Adversarial Attacking and Defensing Modulation Recognition with Deep Learning in Cognitive Radio-Enabled IoT

Zhenju Zhang, Linru Ma, Mingqian Liu, *Member, IEEE*, Yunfei Chen, *Senior Member, IEEE*, and Nan Zhao, *Senior Member, IEEE*

***Abstract*—Modulation recognition using deep learning (DL) can efficiently recognize modulated signals in cognitive radio-enabled Internet of Things (IoT). However, it is vulnerable to the attack of adversarial examples designed by attackers, leading to a decrease in its accuracy. Different adversarial techniques can be used for attacks, but these attacks have limited efficiency. This paper proposes a double loop iterative method. Different from the traditional attack methods, the new method designs an additional external loop iteration for high efficiency. When generating adversarial examples, the initial conditions of each iteration can be updated as the number of iterations changes, so that the adversarial examples can cross the decision boundary of the model as much as possible. In addition, this paper uses knowledge distillation to improve the traditional adversarial training defense, which improves the robustness of the model. Simulation results show that the proposed attack and defense methods have better performance than traditional methods.**

***Index Terms*—Adversarial attack, adversarial training, deep learning, modulation recognition, cognitive radio.**

I. Introduction

COGNITIVE radio (CR) can monitor the idle spectrum available in the environment in real time and enable secondary users to use the spectrum resources of the authorized users in a non-interfering manner [2]–[4]. By improving the efficiency of radio spectrum utilization and communication, CR can solve the problem of increasingly scarce radio spectrum resources and promote more intelligent wireless communication. In the Internet of Things (IoT), CR can help devices adaptively switch between different frequency bands and channels, avoid wasting and interfering with spectrum resources, and improve communication efficiency and reliability among IoT devices. Modulation recognition is used to identify the communication parameters and modulation modes of primary users to alleviate the shortage of spectrum resources, as an important part of CR.

Traditional modulation recognition methods are based on maximum likelihood estimation and statistical pattern recognition, but they are heavily dependent on the prior knowledge of the signal and artificial feature extraction with low accuracy. In recent years, deep learning (DL) has been gradually applied to automatic modulation recognition [5]–[11]. Compared with the traditional modulation recognition methods, the modulation recognition model based on deep neural network (DNN) can effectively extract the characteristics of modulation signals with higher recognition speed and accuracy.

Although DL can automatically extract the features of the signals to recognize them, it still remains unknown how it learns, which makes it less interpretable and the DNN model less secure and more vulnerable to attack. Szegedy *et al.* pointed out that adversarial examples generated by adding carefully designed subtle perturbations to the clean examples can significantly reduce the accuracy of the classifier [12]. Adversarial examples are obtained by adding adversarial perturbations with strong camouflage to clean examples, which can deceive and mislead the recognition model to classify signals incorrectly. The early inferential interpretation of why deep learning is easy to be attacked is its highly nonlinear feature. Goodfellow *et al.* proposed the fast gradient symbol method (FGSM) to attack the convolutional neural network (CNN) classifier [13]. Later, Kurakin *et al.* improved FGSM and proposed an iterative FGSM called the basic iterative method (BIM), which divides the perturbation size in FGSM into multiple segments and iteratively generates adversarial examples [14]. Madry *et al.* added a projection step to BIM, randomly initializing the example under norm constraints, and proposed the projection gradient descent method (PGD) [15]. In order to improve the stability of iteration and the generalization of adversarial examples, Dong *et al.* introduced momentum into the iterative attack and proposed the momentum iterative method (MIM) [16]. Zhang *et al.* explained the transfer characteristics of adversarial examples between different target models, and generated adversarial examples with strong transferability through principal component analysis (PCA) [17].

Adversarial attacks were originally proposed for images, and now have achieved fruitful research results in computer vision and other related fields. For example, in the field of autonomous driving, Xiong *et al.* proposed two multi-source adversarial example attack models, which successfully

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 under Grant 2023-YBGY-249, and in part by the Guangxi Key Research and Development Program under Grant 2022AB46002. *(Corresponding author: Linru Ma.)* An earlier version of this paper was presented in part at the 9th International Conference on Dependable Systems and Their Applications (DSA) [1].

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

L. Ma is with the Institute of Systems Engineering, AMS, Beijing 100071, China (e-mail: malinru@163.com).

Y. Chen is with the Department of Engineering, University of Durham, South Road, Durham, UK, DH1 3LE (e-mail: yunfei.chen@durham.ac.uk).

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

attacked the image and lidar sensing system in autonomous vehicles [18]. Lv *et al.* proposed an adversarial attack method based on the incremental learning for unmanned driving, and achieved a higher attack success rate [19]. However, there are very few studies on adversarial attacks in wireless communication, where the wireless network using DNN is also vulnerable to attacks. In order to improve the robustness of the wireless network model by using adversarial examples, some researchers introduced adversarial attacks into the modulation recognition. Lin *et al.* verified the effectiveness of some gradient-based adversarial attacks on the automatic modulation recognition model, and pointed out that attacks can significantly reduce the accuracy of the target model [20]. Qi *et al.* proposed a detection-tolerant black-box method to attack the modulation classifier and improve the transferability of adversarial attack [21]. Kim *et al.* proposed a channel-aware adversarial attack against classifiers, which shows the vulnerability of classifiers by considering information about channels, transmitter inputs, and classifier models [22]. Liu *et al.* introduced an interference waveform into spectrum sensing systems for data poisoning attack, which significantly reduces the sensing accuracy [23]. Moreover, to counteract adversarial attacks, for common adversarial attacks, researchers have developed some defense mechanisms that can ensure the security of wireless communications. Zhang *et al.* studied a defense method based on training time and running time, which protected the modulation signal classifier based on machine learning from malicious attacks by attackers [24]. Hameed *et al.* proposed a secure wireless communication method that can prevent the attacker from detecting the correct modulation category, which enhances the security of communication between transmitter and receiver [25]. In addition, researchers have pointed out that most of the defense methods proposed in recent years, including active defense and passive defense, can only deal with specific attacks and are difficult to effectively respond to new types of attacks [26]–[28].

This paper examines the attack performance of several traditional attack methods and the defense performance of traditional adversarial training, and proposes a new double loop iterative attack method and a new defense method based on distillation-based adversarial training. The main contributions of this paper are as follows:

- Different high-accuracy modulation recognition models based on DNN on the open source simulation data set are trained to identify and classify the modulation signals to achieve high accuracy.
- A double loop iterative attack method is proposed. By adding an external loop iteration and designing an external iteration step, it initializes the conditions of each attack with the change of loop parameters in the process of generating adversarial examples, which can continuously increase the prediction loss of the recognition model.
- A defensive method based on distillation for adversarial training is proposed. By using complex models to learn the deep adversarial knowledge of adversarial examples, and using knowledge distillation to transfer the knowledge to simple models, it can effectively enhance the robustness of simple models deployed in resource-constrained devices to adversarial examples.
- In the cognitive radio scenario, the proposed attack method can significantly reduce the accuracy of the modulation recognition model, and the proposed defense method can effectively enhance the robustness of the recognition model.

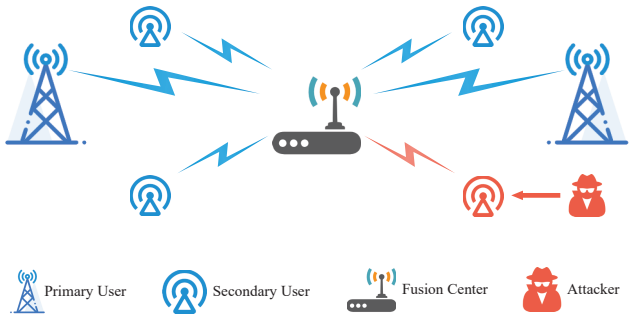


Fig. 1. System model of adversarial attack in cognitive radio-enabled IoT.

The rest of this paper is organized as follows: Section II introduces the system model of adversarial attack on CR and the traditional adversarial attack methods. In Section III, an attack algorithm based on double loop iteration is proposed. In Section IV, knowledge distillation is introduced into adversarial training to enhance the robustness of simple models. In Section V, the feasibility and effectiveness of the proposed attack and defense from multiple aspects are shown via simulation results. Section VI summarizes the paper.

II. System Model

A. Wireless Model

In IoT, the adversarial attack will severely affect the normal operation of the CR system by interfering with modulation recognition, thereby disrupting the reliable transmission of communication, as shown in Fig. 1.

From the attacker's point of view, the adversarial attack can be regarded as the act of disguising the modulation signal. The attacker generates an adversarial perturbation through the attack algorithm, adds it to the modulated signal to form an adversarial example, and transmits it to the target receiver. After receiving the adversarial example, the receiver will automatically identify the modulation type of the example, which may be exploited by the attacker to generate a wrong recognition result. Compared with adding noise to the signal, the example generated by the adversarial attack can maximize the classification loss of the target model, thus reducing the accuracy of the model. In addition, due to the transferability, some adversarial examples designed for the source model can also be used to attack other target models, which increases the risk of attack for other network models [29], [30].

The adversarial example $\boldsymbol{x}^*$ refers to the example formed by deliberately adding the subtle perturbation $\boldsymbol{\eta}$ to the input $\boldsymbol{x}$, which will cause the recognition model to make an incorrect prediction. In many cases, $\boldsymbol{x}^*$ looks very similar to $\boldsymbol{x}$, and

human observers cannot notice the difference between them. Thus, the adversarial example can be denoted as

$$\boldsymbol{x}^* = \boldsymbol{x} + \boldsymbol{\eta}. \tag{1}$$

When performing an untargeted attack, the goal is to maximize the loss between the prediction probability distribution of the recognition model for the constructed example and the true label of the original example, which can be expressed as

$$\max_{\boldsymbol{\eta}:\|\boldsymbol{\eta}\|\leq\varepsilon} \mathcal{L}\left(F\left(\boldsymbol{x}+\boldsymbol{\eta}\right), \boldsymbol{l}\right), \tag{2}$$

where ε represents the maximum value that the adversarial perturbation can reach under the norm constraint, $\mathcal{L}$ represents the loss of the target recognition model, $\boldsymbol{l}$ represents the true label of the original example $\boldsymbol{x}$, and $F\left(\boldsymbol{x}\right)$ represents the composite operation between n_l different output layers of the recognition network model with

$$F(\boldsymbol{x}) = F_{n_l} \circ F_{n_l-1} \circ \cdots \circ F_1(\boldsymbol{x}), \tag{3}$$

and its output is the recognition probability distribution for the input $\boldsymbol{x}$.

Although the loss maximization problem in (2) is difficult to solve, [13] linearizes the loss function near $(\boldsymbol{x}, \boldsymbol{l})$ to give

$$\widetilde{\mathcal{L}}(\boldsymbol{\eta}) = \mathcal{L}\left(F(\boldsymbol{x}), \boldsymbol{l}\right) + \nabla_{\boldsymbol{x}}\mathcal{L}\left(F(\boldsymbol{x}), \boldsymbol{l}\right)\boldsymbol{\eta}. \tag{4}$$

The optimal solution to (4) is to maximize the twist in ε along the sign direction of the loss gradient $\nabla_{\boldsymbol{x}}\mathcal{L}$ to get

$$\boldsymbol{\eta}^* = \varepsilon \cdot \operatorname{sign}\left(\nabla_{\boldsymbol{x}}\mathcal{L}\left(F(\boldsymbol{x}), \boldsymbol{l}\right)\right). \tag{5}$$

In the adversarial attack, the norm can be used to uniformly regulate the range of perturbation generated by the attack algorithm, which is a constraint on the perturbation. The L_p norm of the perturbation $\boldsymbol{\eta}$ can be expressed as

$$\|\boldsymbol{\eta}\|_p = \left(\sum_{i=1}^{n_s} \|\eta_i\|^p\right)^{\frac{1}{p}}. \tag{6}$$

Common norms include L_0, L_2 and L_∞. For adversarial perturbation, L_0 represents the number of sampling points of non-zero perturbation, L_2 represents the Euclidean distance between examples before and after perturbation, and L_∞ represents the maximum value of perturbation at all sampling points. In the above attack, when there is a norm constraint $\|\boldsymbol{\eta}\|_p \leq \varepsilon$ for any p, the optimal solution of $\boldsymbol{\eta}$ can be generalized as

$$\boldsymbol{\eta}^* = \varepsilon \cdot \operatorname{sign}\left(\nabla_{\boldsymbol{x}}\mathcal{L}\left(F(\boldsymbol{x}), \boldsymbol{l}\right)\right)\left(\frac{\left|\nabla_{\boldsymbol{x}}\mathcal{L}\left(F(\boldsymbol{x}), \boldsymbol{l}\right)\right|}{\left\|\nabla_{\boldsymbol{x}}\mathcal{L}\left(F(\boldsymbol{x}), \boldsymbol{l}\right)\right\|_q}\right)^{\frac{1}{p-1}}, \tag{7}$$

where q is the dual of p, and

$$\frac{1}{p} + \frac{1}{q} = 1. \tag{8}$$

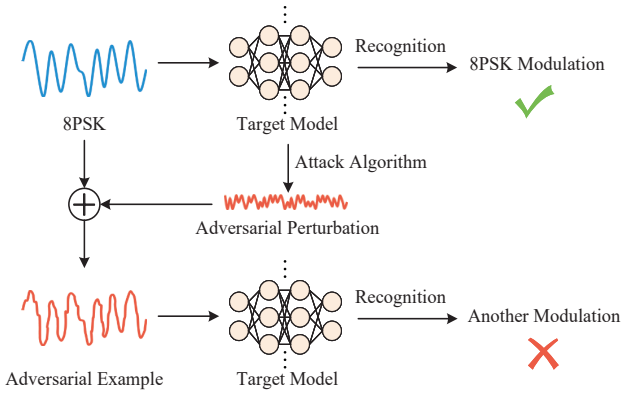


Fig. 2. The process of adversarial attack.

B. Adversarial Attack Models

FGSM, BIM, PGD and MIM are all attack methods using infinite norm, which generate adversarial examples under the constraint of L_∞ norm. They determine the direction of adversarial perturbation according to the loss gradient of the target model, and generate an adversarial example by adding a certain perturbation in this direction, thus attacking the model. For example, a trained model can correctly recognize an 8PSK signal as an 8PSK modulation type, but it will recognize the adversarial example generated by the attacker using the model information and the attack algorithm as another modulation type, as shown in Fig. 2.

1) FGSM: When generating an adversarial example, FGSM obtains the attack direction by calculating the loss gradient, and then adds a fixed step size in this direction as the adversarial perturbation level and adds it to the clean example. FGSM is extremely fast in generating adversarial examples because it does not require multiple iterations, but it cannot repeatedly query model parameters to enhance the attack performance. FGSM can be expressed as

$$\begin{cases} \boldsymbol{\eta} = \varepsilon \cdot \operatorname{sign}\left(\nabla_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{x}, \boldsymbol{l})\right), \\ \boldsymbol{x}^* = \boldsymbol{x} + \boldsymbol{\eta}, \end{cases} \tag{9}$$

where $\nabla_{\boldsymbol{x}}\mathcal{L}$ represents the loss gradient of the target model for the input.

2) BIM: Compared with FGSM, BIM divides the direction and size of the adversarial examples into multiple segments, which solves the problem that FGSM cannot update examples by accessing the model multiple times. If the attack process contains N iterations, the iteration step is $\alpha = \varepsilon/N$. BIM can be expressed as

$$\begin{cases} \boldsymbol{x}_0^* = \boldsymbol{x}, \\ \boldsymbol{x}_{n+1}^* = \operatorname{Clip}_{\boldsymbol{x},\varepsilon}\left\{\boldsymbol{x}_n^* + \alpha \cdot \operatorname{sign}\left(\nabla_{\boldsymbol{x}_n^*}\mathcal{L}\left(\boldsymbol{x}_n^*, \boldsymbol{l}\right)\right)\right\}, \\ \boldsymbol{x}^* = \boldsymbol{x}_N^*, \end{cases} \tag{10}$$

where n represents the number of current iterations, and $\operatorname{Clip}_{\boldsymbol{x},\varepsilon}\{\cdot\}$ denotes that the examples are restricted to $[\boldsymbol{x}-\varepsilon, \boldsymbol{x}+\varepsilon]$. Based on FGSM, BIM segments the overall perturbation level, which can increase the loss of the model

by using the information of the target model in the iterative process of generating adversarial examples, but it increases the computational complexity.

3) PGD: On the basis of BIM, PGD adds a projection step to randomly initialize adversarial examples under norm constraint, and uses the initial point of noise to generate adversarial examples with stronger attack performance, which can be expressed as

$$\boldsymbol{x}_{n+1}^{*}=\prod_{\boldsymbol{x}+\boldsymbol{S}}\left(\boldsymbol{x}_{n}^{*}+\alpha\cdot\operatorname{sign}\left(\nabla_{\boldsymbol{x}_{n}^{*}}\mathcal{L}\left(\boldsymbol{x}_{n}^{*},\boldsymbol{l}\right)\right)\right), \tag{11}$$

where $\boldsymbol{S}$ denotes the random perturbation introduced to the original examples under norm constraint.

4) MIM: By introducing the concept of momentum into adversarial attacks, MIM solves the problem of over-fitting and local optimal solution in the optimization process, and has good aggression and generalization. MIM enhances the stability of the perturbation direction during the attack process by accumulating the loss gradient in the iterative process in a certain proportion, which can be expressed as

$$\begin{cases}\boldsymbol{x}_{0}^{*}=\boldsymbol{x},\boldsymbol{g}_{0}=\mathbf{0},\\ \boldsymbol{g}_{n+1}=\mu\cdot\boldsymbol{g}_{n}+\dfrac{\nabla_{\boldsymbol{x}_{n}^{*}}\mathcal{L}\left(\boldsymbol{x}_{n}^{*},\boldsymbol{l}\right))}{\left\|\nabla_{\boldsymbol{x}_{n}^{*}}\mathcal{L}\left(\boldsymbol{x}_{n}^{*},\boldsymbol{l}\right)\right\|_{1}},\\ \boldsymbol{x}_{n+1}^{*}=\operatorname{Clip}_{\boldsymbol{x},\varepsilon}\left\{\boldsymbol{x}_{n}^{*}+\alpha\cdot\operatorname{sign}\left(\boldsymbol{g}_{n+1}\right)\right\},\\ \boldsymbol{x}^{*}=\boldsymbol{x}_{N}^{*},\end{cases} \tag{12}$$

where $\boldsymbol{g}_n$ represents the gradient accumulated during the iteration, and μ represents the attenuation coefficient of $\boldsymbol{g}_n$.

C. Modulation Recognition

1) Data Set: In order to better study the effectiveness of adversarial attacks on CR, we select the RADIOML2016.10B dataset designed by DeepSiG [31]. The data set contains 1.2 million signal examples with a length of 128, and is composed of 10 modulation signals under different SNRs. It contains eight digital signals: 8 phase shift keying (8PSK), quadrature phase shift keying (QPSK), binary phase shift keying (BPSK), Gaussian frequency shift keying (GFSK), continuous phase frequency shift keying (CPFSK), pulse amplitude modulation 4 (PAM4), quadrature amplitude modulation 16 (QAM16) and QAM64, and two analog signals: wide band frequency modulation (WBFM) and double sideband amplitude modulation (AM-DSB). The data set contains twenty SNRs. The SNR of the modulated signal varies from -20 dB to 18 dB, and the interval is 2 dB. We use 80% and 20% of the examples in the data set as training set and test set, respectively. Using the in-phase component and quadrature component in the data set, we can express the time domain expression of a signal as

$$S\left(t\right)=Icos\left(2\pi ft\right)+Qsin\left(2\pi ft\right), \tag{13}$$

where I and Q are in-phase component and quadrature component, respectively, and f is the carrier frequency.

2) Target Model: In order to examine the effects of different attacks and highlight the performance of the proposed attack method, ResNet is selected as the target model. ResNet was proposed by He *et al.*, using the "shortcut connection" connection method, which can easily extract the features of input examples and has been widely used in automatic recognition [32]. It uses the residual function to optimize the learning process, which makes it easy to deepen without degrading performance. Therefore, we use ResNet as the target model for modulation recognition and generation of adversarial examples. The residual stack structure of ResNet is shown in Fig. 3, and its overall structure is shown in Table I.

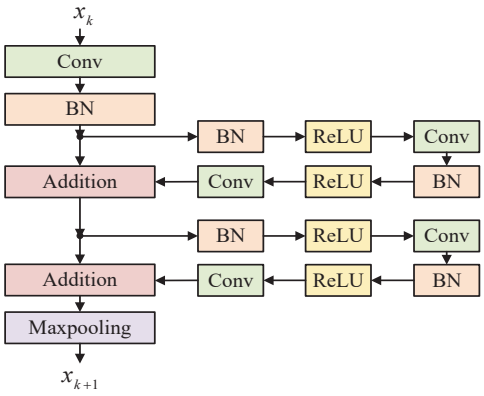


Fig. 3. Residual stack structure.

TABLE I

ResNet Network Layout

Layer	Output shape
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

Before training the network, we set the training batch and the initial learning rate to 1024 and 0.001, respectively, and set the learning rate as an automatic update mechanism to make the network converge faster. After 100 rounds of network training, the test set is input into the trained network to test its recognition accuracy.

III. Double Loop Iterative Method

In this section, we will propose a new method that takes advantage of double loop iteration for attacks.

A. Motivation

In multi-classification tasks, cross entropy loss is often used to optimize the model, which characterizes the difference between the predicted values of the model and the true labels of the inputs. The cross entropy loss of the model during training can be expressed as

$$\mathcal{L}\left(\boldsymbol{x},\boldsymbol{l}\right)=-\frac{1}{N_1}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}l_{ij}\left(\boldsymbol{x}_i\right)\log_2\left(p_{ij}\left(\boldsymbol{x}_i\right)\right), \tag{14}$$

where N_1 is the number of input signals, N_2 is the number of categories of signals, $l_{ij}(\boldsymbol{x})$ is the true label of the input, and $p_{ij}(\boldsymbol{x})$ is the prediction probability of the model.

For FGSM, BIM, PGD and MIM, whether they use one-step iteration or multi-step iteration to generate adversarial perturbation, all of them have only one loop iteration layer. They determine the perturbation direction by calculating the loss gradient of the target model, and add a fixed-size perturbation in this direction to generate an adversarial example to attack the target model.

However, In the process of iterative attack, the adversarial example generated by the direction and size of the adversarial perturbation may not be sufficient to make the loss of the model reach the threshold. If the examples generated by them after the iteration process cannot fool the classifier model, then the attack will fail. Therefore, we analyze how to improve the attack performance of the adversarial example by adjusting the direction and size of the perturbation in the following.

B. Double Loop Iterative Attack

In this paper, we determine the local direction of iteration by the accumulation of momentum. By continuously accumulating the current and previous gradients, it can generate adversarial examples more stably, and these examples have strong transferability. The direction of iteration can be expressed as

$$\boldsymbol{g}_{n+1} = \mu \cdot \boldsymbol{g}_n + \frac{\nabla_{\boldsymbol{x}_n^*} \mathcal{L}(\boldsymbol{x}_n^*, \boldsymbol{l}))}{\left\| \nabla_{\boldsymbol{x}_n^*} \mathcal{L}(\boldsymbol{x}_n^*, \boldsymbol{l}) \right\|_1}. \tag{15}$$

We provide the impact of the direction and size of perturbation on the model's prediction results in Proposition 1.

Proposition 1: The prediction result of the model for an adversarial example is

$$\begin{cases} y_p(\boldsymbol{x}^*) = y_t, & \mathcal{L}(\boldsymbol{x}, \alpha, \boldsymbol{g}) \leq \mathcal{L}_T \\ y_p(\boldsymbol{x}^*) \neq y_t, & \mathcal{L}(\boldsymbol{x}, \alpha, \boldsymbol{g}) > \mathcal{L}_T \end{cases} \tag{16}$$

where y_p is the predicted class of the model for the adversarial example, y_t is the true class, α is the size of the perturbation, $\boldsymbol{g}$ is the accumulated gradient used to determine the direction of the perturbation, and $\mathcal{L}_T$ is the model reach the threshold.

Proof: See Appendix A. ■

By adjusting the iteration step size and accumulated gradient during the iterative process, the prediction loss of the model can be increased, which guides the example to cross the decision boundary of the model and be misclassified.

We consider adding an outer loop iteration layer to the momentum iteration and using the generated initial adversarial example as the new input to continuously increase the loss of the target model within a limited number of iterations. Adding an outer loop iteration layer is not simply increasing the number of momentum iterations, because the iteration conditions are initialized at the beginning of each new outer loop, including clearing the accumulated gradients and setting the new iteration step size for that loop.

Remark 1: After adding the external layer, when maximizing the loss $\mathcal{L}(\boldsymbol{x}_n^*, \boldsymbol{l})$, the example moves continuously in the decision domain, and each move is close to or even cross the decision boundary. Each time a new external loop begins, the momentum goes to zero and begins to accumulate again, so that the effect of the previous gradient information on the direction of the perturbation is reduced, making it more flexible to find the direction.

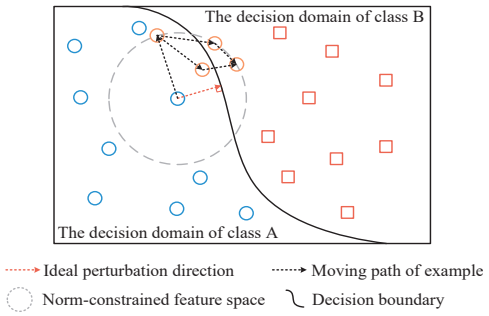


Fig. 4. Moving process of adversarial example in decision domain.

Ideally, the perturbation direction of an adversarial example in the decision domain is the direction of the original example perpendicular to the decision boundary. However, in the iterative optimization process, there are often some process that skips the global optimal point or hovers near the local optimal point. Therefore, it is necessary to take some necessary means to re-activate the optimization process. By increasing the external loop iteration layer and initializing the step, the adversarial example is more likely to cross the decision boundary. To clearly analyze the effect of the proposed method in the decision domain, we consider the momentum iteration process of each internal loop as a whole, with an arrow representing the example positions before and after the process, as shown in Fig. 4. It shows the movement of the example in the decision domain when the adversarial example is generated using double loop iteration. After each movement, the maximum distance between the example and the original example can be expressed as an infinite norm

$$\left\| \boldsymbol{x}_n^* - \boldsymbol{x} \right\|_\infty = \max_{1 \leq i \leq n_s} |(\boldsymbol{x}_n^* - \boldsymbol{x})_i|, \tag{17}$$

where n_s represents the number of sampling points of the input signal.

Remark 2: Since the size of the adversarial perturbation is constrained by the norm of $\left\| \boldsymbol{x}^* - \boldsymbol{x} \right\|_\infty \leq \varepsilon$, the movement of the example in the decision domain is constrained in the feature space mapped by the norm of the adversarial perturbation, and each example can only move at the boundary or inside the space. The purpose of the adversarial attack is to make the example in the space cross the decision boundary as much as possible, as in Fig. 4 where the example moves in different paths.

When the iteration step size in the external loop is set under the norm constraint, the step size should be not less than the momentum iteration step size to ensure the effectiveness of the attack in the internal loop. In addition, the adversarial examples should be able to adjust the targeting of the attack with the iteration process during the attack, so the examples should determine the approximate position and the specific position to maximize the model loss in the early and late

TABLE II

Recognition Results of ResNet with Different Loop Parameters

N	5			10			25	100
M	1	3	5	1	5	10	1	1
Accuracy(%)	53.9	48.3	46.8	50.4	46.3	45.2	50.3	49.8

iterations, respectively, which means that the iteration step size in the external loop is decreased. Therefore, the iteration step size α_m in the external loop can be set to

$$\begin{aligned} \alpha_m &= \frac{(M-m+1)\cdot\varepsilon}{N}, \\ s.t. \quad & 1 \le M \le N, \\ & 1 \le m \le M, \end{aligned} \tag{18}$$

where M and N represent the number of external loop iterations and the number of internal loop iterations, respectively, and m represents the number of completed external loop iterations. When $M = N$, α_m satisfies $\varepsilon/N \le \alpha_m \le \varepsilon$ and decreases with the increase of m.

Remark 3: It can be seen from (18) that the iteration step size gradually decreases with the increase of the current number of external loops m. Therefore, the iteration step is updated each time a new external loop begins. The step is larger when m is small in order to generate the initial fuzzy perturbation, and gradually decreases as m becomes larger to fine-tune the perturbation.

After getting the direction and size of the iteration, in the $(n+1)$-th iteration in the internal loop, we can get the adversarial perturbations by

$$\boldsymbol{\eta}_{n+1} = \alpha_m \cdot \operatorname{sign}\left(\mu \cdot \boldsymbol{g}_n + \frac{\nabla_{\boldsymbol{x}_n^*}\mathcal{L}\left(\boldsymbol{x}_n^*, \boldsymbol{l}\right)}{\left\|\nabla_{\boldsymbol{x}_n^*}\mathcal{L}\left(\boldsymbol{x}_n^*, \boldsymbol{l}\right)\right\|_1}\right) \tag{19}$$

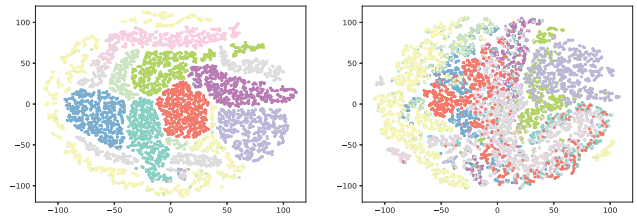
and the adversarial examples by

$$\boldsymbol{x}_{n+1}^* = \operatorname{Clip}_{\boldsymbol{x},\varepsilon}\left\{\boldsymbol{x}_n^* + \boldsymbol{\eta}_{n+1}\right\}. \tag{20}$$

After the end of the double loop iteration, the loss of the model to these examples is often greater than that of the single loop iteration, which makes the model more vulnerable to attack and recognizes these examples as another wrong class.

To study the influence of loop parameters on attack effectiveness, we select different combinations of external loop number and internal loop number to generate adversarial examples. The recognition results of ResNet for these examples are shown in Table II. It can been seen that when $M = 1$ and N gradually increase, the attack performance increases, but the increase is smaller. When N is constant and M increases, the attack performance increases. In addition, the adversarial examples generated when $M = N = 5$ decrease the accuracy of the model more than when $N = 25$ and $M = 1$, and the adversarial examples generated when $M = N = 10$ decrease the accuracy of the model more than when $N = 100$ and $M = 1$. It shows that the proposed method has a better effect than simply increasing the overall number of iterations.

To preliminarily study whether the attack caused the expected damage to the target model, t-SNE is used to visualize the characteristics of the clean examples and the corresponding adversarial examples when $M = N = 10$, as shown in Fig. 5. It can be seen from Fig. 5 that the characteristics of clean examples in the model have obvious regional characteristics, which is conducive to the correct classification of modulated signals by the model. In the case of a non-target attack, the proposed attack method makes the features of examples pass through the decision boundary of the model, so that the generated adversarial examples overlap each other between different feature regions, which can greatly fool the recognition model.



(a) Clean Examples (b) Adversarial Examples

Fig. 5. t-SNE visualizations of the features of clean examples and corresponding adversarial examples in the recognition model.

Compared with other traditional attacks, the proposed attack method mainly adds an external loop, so we call it the double loop iterative method (DLIM). Algorithm 1 summarizes the detailed steps of the DLIM attack algorithm. To facilitate the implementation of the algorithm, the number of loops m starts from zero, then $\alpha_m = (M-m)\cdot\varepsilon/N$.

Algorithm 1 Double Loop Iterative Method

Input: Original input example $\boldsymbol{x}$; true label $\boldsymbol{l}$; loss $\mathcal{L}$ of a classifier.

Input: The perturbation constraint ε and decay factor μ; external iterations M and internal iterations N.

Output: An adversarial example x^* with $\|\boldsymbol{x}^* - \boldsymbol{x}\|_\infty \le \varepsilon$.

$\boldsymbol{x}_0^* = \boldsymbol{x}$;
for $m = 0$ to $M-1$ **do**
  $\boldsymbol{g}_0 = 0$; $\boldsymbol{x}_0^* = \boldsymbol{x}_m^*$; $\alpha_m = (M-m)\cdot\varepsilon/N$;
  for $n = 0$ to $N-1$ **do**
    Input $\boldsymbol{x}_n^*$ to classifier and obtain the loss gradient $\nabla_{\boldsymbol{x}_n^*}\mathcal{L}\left(\boldsymbol{x}_n^*, \boldsymbol{l}\right)$;
    Update $\boldsymbol{g}_{n+1}$ by accumulating the velocity vector in the gradient direction as

$$\boldsymbol{g}_{n+1} = \mu \cdot \boldsymbol{g}_n + \frac{\nabla_{\boldsymbol{x}_n^*}\mathcal{L}\left(\boldsymbol{x}_n^*, \boldsymbol{l}\right))}{\left\|\nabla_{\boldsymbol{x}_n^*}\mathcal{L}\left(\boldsymbol{x}_n^*, \boldsymbol{l}\right)\right\|_1};$$

    Update $\boldsymbol{x}_{n+1}^*$ by applying the sign gradient as

$$\boldsymbol{x}_{n+1}^* = \operatorname{Clip}_{\boldsymbol{x},\varepsilon}\left\{\boldsymbol{x}_n^* + \alpha_m \cdot \operatorname{sign}\left(\boldsymbol{g}_{n+1}\right)\right\};$$

  end for
  $\boldsymbol{x}_{m+1}^* = \boldsymbol{x}_N^*$;
end for
return $\boldsymbol{x}^* = \boldsymbol{x}_M^*$.

To compare with the traditional methods and highlight the improvement, we express the process of the DLIM algorithm

in an external loop as follows

$$\begin{cases} \boldsymbol{x}_0^* = \boldsymbol{x}, \boldsymbol{g}_0 = \mathbf{0}, \alpha_m = (M-m)\cdot\varepsilon/N, \\ \boldsymbol{g}_{n+1} = \mu\cdot\boldsymbol{g}_n + \dfrac{\nabla_{\boldsymbol{x}_n^*}\mathcal{L}\left(\boldsymbol{x}_n^*, \boldsymbol{l}\right)}{\left\|\nabla_{\boldsymbol{x}_n^*}\mathcal{L}\left(\boldsymbol{x}_n^*, \boldsymbol{l}\right)\right\|_1}, \\ \boldsymbol{x}_{n+1}^* = \mathrm{Clip}_{\boldsymbol{x},\varepsilon}\left\{\boldsymbol{x}_n^* + \alpha_m\cdot\mathrm{sign}\left(\boldsymbol{g}_{n+1}\right)\right\}, \\ \boldsymbol{x}^* = \boldsymbol{x}_N^*. \end{cases} \tag{21}$$

(21) represents the generation process of adversarial examples in an external loop. The difference from MIM is that the initialization of the loop conditions and the increase of an iterative step size determined by the double loop parameters.

C. Attack Performance Metrics

When implementing an adversarial attack, the purpose is to use small perturbations that the receiver cannot perceive to make the model misclassify and try to cover up the traces of the attack on the waveform while ensuring the attack effect. Therefore, the main metrics to evaluate the attack performance are the attack success rate and the perceptibility of the adversarial perturbation.

The attack success rate of adversarial examples can be reflected by the accuracy of the recognition model. The more the recognition accuracy decreases after the attack, the higher the attack success rate will be. The ratio of perturbation power to noise power and the ratio of perturbation power to signal power can be used to describe the perceptibility of perturbation, which are called the perturbation-to-noise ratio (PNR) and the perturbation-to-signal ratio (PSR), respectively [34]. The relationship between PNR, PSR and SNR satisfies $\mathrm{PNR} = \mathrm{PSR} \times \mathrm{SNR}$ with [35]:

$$\mathrm{PNR}\left[\mathrm{dB}\right] = \frac{\mathbb{E}\left[\left\|\varepsilon\right\|_2^2\right]}{\mathbb{E}\left[\left\|x\right\|_2^2\right]}\left[\mathrm{dB}\right] + \mathrm{SNR}\left[\mathrm{dB}\right], \tag{22}$$

where $\mathbb{E}$ is the expectation. According to the definition of PNR, the larger the PNR, the higher the added perturbation level. When $\mathrm{PNR} \leq 0$ dB, it shows that the order of the perturbation is equal to or even lower than the noise level. At this time, we can consider the perturbation to be imperceptible. For example, for the modulation signal $\boldsymbol{x}$ with an amplitude of 0.01 used in this paper, when $\mathrm{SNR} = 10$ dB, if the perturbation perception is not visible, it is necessary that

$$\frac{\mathbb{E}\left[\left\|\varepsilon\right\|_2^2\right]}{\mathbb{E}\left[\left\|x\right\|_2^2\right]}\left[\mathrm{dB}\right] + 10\left[\mathrm{dB}\right] \leq 0\left[\mathrm{dB}\right], \tag{23}$$

then the solution $\varepsilon \leq 0.0032$. This indicates that the perturbation has strong concealment when the maximum perturbation level does not exceed 0.0032 under the above conditions.

In the PNR expression, $\|\varepsilon\|_2^2$ is the maximum perturbation power that the adversarial example can achieve at a certain sampling point, which limits the visibility of the adversarial perturbation before generating the adversarial example. When using different attack methods to generate adversarial examples under the same norm constraint, in order to compare the performances of different attacks, it is necessary to quantitatively analyze the similarity between the generated examples and the clean examples. Zhao *et al.* proposed the fitting difference (FD) to quantify the degree of change of clean examples after the adversarial attack, which can be used to measure the concealment of attacks [36]. After an attack, the FD between the adversarial example and the original example can be expressed as

$$FD\left(\boldsymbol{s}, \boldsymbol{s}^*\right) = \frac{\sum_{i=1}^{L_s}\left(s_i - s_i^*\right)^2}{\sum_{i=1}^{L_s}\left(s_i - \bar{s}\right)^2}, \tag{24}$$

where L_s is the length of the original example, $\boldsymbol{s}$ and $\boldsymbol{s}^*$ are the original example and the corresponding adversarial example, respectively, and $\bar{s}$ is the average of the original example, that is

$$\bar{s} = \frac{1}{L_s}\sum_{i=1}^{L_s} s_i. \tag{25}$$

In general, when $\mathrm{FD} \to 0$, the waveform of the adversarial example is very similar to the original example, indicating that the adversarial attack has good concealment. On the contrary, when FD increases, the adversarial example waveform gradually deviates from the original example waveform, and the adversarial attack is easy to detect. Therefore, the perceptual invisibility of attacks can be analyzed by comparing the FD values of the adversarial examples generated by different attacks between the original examples.

IV. Adversarial Training Based on Knowledge Distillation

In this section, we will apply knowledge distillation to adversarial training according to the distribution characteristics of the model's prediction probability for adversarial examples by proposing a new defense method for adversarial training based on knowledge distillation.

A. Adversarial Training

Adversarial training (AT) uses adversarial examples to adjust the parameters of the target model, which is one of the most direct and effective defense methods [37]. After adversarial training, the model can learn the adversarial features of adversarial examples within the model, thereby resisting similar attacks. For multi-classification tasks, under the infinite norm constraint $\left\|\boldsymbol{x}^* - \boldsymbol{x}\right\|_\infty \leq \varepsilon$, the parameters of a robust model can be obtained by

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{l})\sim\mathcal{D}}\left[\max_{\left\|\boldsymbol{x}^* - \boldsymbol{x}\right\|_\infty \leq \varepsilon} \mathcal{L}\left(f\left(\boldsymbol{x}^*, \boldsymbol{\theta}\right), \boldsymbol{l}\right)\right], \tag{26}$$

where $\boldsymbol{x}$ and $\boldsymbol{l}$ represent the original input and the true label, respectively, $\mathcal{D}$ is the distribution of the input, $\boldsymbol{\theta}$ is the model parameter, and $\mathcal{L}\left(\cdot\right)$ is the prediction loss of the model.

The adversarial training process is actually a min-max game, which aims to achieve the best balance between the accuracy and robustness of the model. The max optimization problem in (26) can be regarded as the process of generating adversarial examples, which maximizes the prediction loss of the model

by adding adversarial perturbations satisfying norm constraints to the input examples. The min optimization problem is used to minimize the overall expected risk of the model, which can be regarded as the process of training the model with adversarial examples. The most robust parameters for these examples can be found under the condition that the prediction results conform to the original data distribution. For the maximization process in (26), the adversarial examples $\boldsymbol{x}_{PGD}^{*}$ obtained by using the PGD can greatly improve the robustness of the model to general first-order attacks [15]. At this point, (26) can be rewritten as

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{l})\sim\mathcal{D}} \left[\mathcal{L}\left(f\left(\boldsymbol{x}_{PGD}^{*},\boldsymbol{\theta}\right),\boldsymbol{l}\right)\right]. \tag{27}$$

B. Knowledge Distillation

The basic idea of knowledge distillation is to use a complex teacher model to guide the training of a simple student model, which is a technique for compressing and optimizing models [38]. In knowledge distillation, the teacher model is usually a complex DNN trained on a large-scale data set, which can learn the deep features of the data and has high accuracy and generalization ability. The student model is usually a simplified neural network with a simple structure and a small number of parameters. The student model is faster, takes up fewer resources than the teacher model in training and reasoning, and is more suitable for some devices or scenarios with limited storage and computing resources.

Through knowledge distillation, the student model can learn the precise decision boundary of the teacher model. The student model takes the output probability distribution of the teacher model as a soft label during training, and minimizes the Kullback-Leibler divergence between its predicted output and the output probability distribution of the teacher model, so as to better simulate the decision-making process and knowledge representation of the teacher model. When the distillation temperature is T, the prediction probability of the model for the input is

$$q_k(\boldsymbol{x}) = \frac{\exp\left(z_k(\boldsymbol{x})/T\right)}{\sum_{k=1}^{K}\exp\left(z_k(\boldsymbol{x})/T\right)}, \tag{28}$$

where $z_i(\boldsymbol{x})$ is the logit value that the model predicts the input as class i, and K is the number of categories of signals. Then the predictive soft label of the teacher model for the input can be expressed as a probability distribution

$$\boldsymbol{l}_t^{soft} = \left(q_1, q_2, \cdots, q_K\right). \tag{29}$$

Then, the student model can be trained by using the soft label instead of the true label, so as to transfer the knowledge of the teacher model to the student model.

C. Distillation-Based Adversarial Training

In order to ensure the concealment of adversarial examples, attackers often generate adversarial examples in the adversarial area near the original example [20]. Due to the strong learning ability of complex models, adversarial training using complex models can learn the adversarial information contained in adversarial examples. However, in some specific practical applications, such as edge devices, the storage and computing resources of devices are very limited, which limits the deployment of complex models to these devices, so the use of simple models is necessary. In order to make the simple model deployed in the device have better robustness, the adversarial knowledge learned by the complex model can be transferred to the simple model through knowledge distillation, so that the simple model also has strong defense performance. The process of adversarial training based on knowledge distillation is shown in Fig. 6.

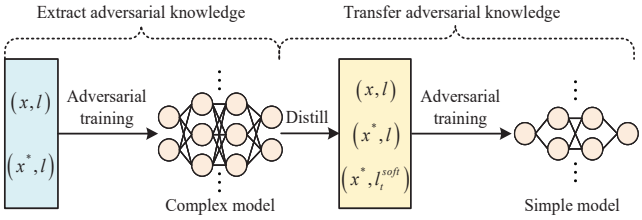


Fig. 6. Adversarial training based on knowledge distillation.

Fig. 6 shows the brief process of the proposed distillation-based adversarial training (DAT), including the extraction and transfer of adversarial information contained in the adversarial examples. After adversarial training using clean and adversarial examples, the complex model can learn adversarial knowledge from the input, which is reflected in the decision boundary and prediction probability distribution of the model. Then, through knowledge distillation, the adversarial knowledge learned by the complex model is transferred to the simple model.

When training the simple model, traditional adversarial training uses one-hot encoding as the label, which will overfit the network. Label smoothing can avoid network overconfidence by adjusting the probability distribution of the original label, so that the predicted value of the model is not excessively concentrated in the category with high probability. The kth probability value in the smoothed label can be expressed as

$$\tilde{l}(k) = (1-\beta)\cdot\delta_{k,l} + \beta\cdot u(k), \tag{30}$$

where β represents the smoothing coefficient, $\delta_{k,l}$ represents the distribution of the one-hot encoding, and $u(k)$ represents the decay probability distribution and is often uniformly distributed $u(k) = 1/K$.

Proposition 2: After label smoothing, the model's prediction loss for adversarial examples can be expressed as

$$\mathcal{L}\left(\boldsymbol{x}^{*},\tilde{\boldsymbol{l}}\right) = (1-\beta)\cdot\mathcal{L}\left(\boldsymbol{x}^{*},\boldsymbol{l}\right) + \beta\cdot\mathcal{L}\left(\boldsymbol{x}^{*},\boldsymbol{u}\right). \tag{31}$$

When the model is over-fitting, a certain prediction probability of the model output is close to 1, and the distribution loss $\mathcal{L}\left(\boldsymbol{x}^{*},\boldsymbol{u}\right)$ between the prediction probability distribution and u will increase, which is conducive to preventing the model from over-learning.

Proof: See Appendix B. ■

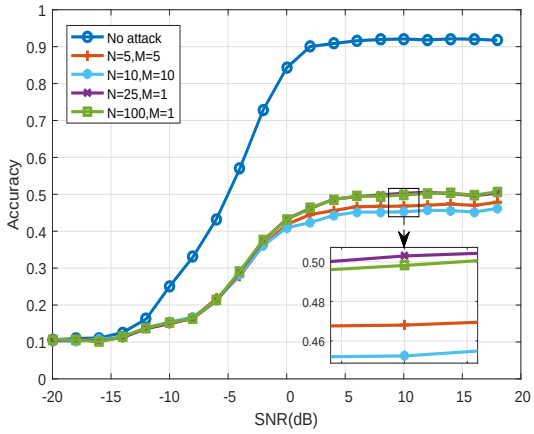


Fig. 7. Accuracy of ResNet with different loop parameters.

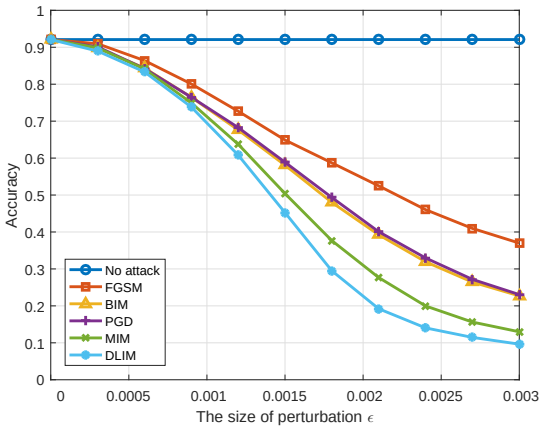


Fig. 8. Accuracy of ResNet under different perturbation constraints.

In this paper, we use the prediction results of the complex model to adjust $u(k)$ in (30) to better utilize the adversarial information learned by this model. We assign the smoothing values based on the probability values other than the true category probability q_t in $\boldsymbol{l}_t^{soft}$, setting $u(k)$ as

$$u(k) = (1 - \delta_{k,l}) \cdot \frac{q_k}{1 - q_t}, \quad k = 1, 2, \cdots, K. \tag{32}$$

Then, the training loss of the simple model can be expressed as

$$\mathcal{L}_s = (1 - \lambda) \cdot \mathcal{L}(\boldsymbol{x}, \boldsymbol{l}) + \lambda \cdot \mathcal{L}\left(\boldsymbol{x}^*, \tilde{\boldsymbol{l}}\right) \cdot T^2, \tag{33}$$

where λ represents the proportion of adversarial examples used in adversarial training, and $\mathcal{L}(\cdot)$ represents the cross entropy loss in (14). After training, the simple model will contain the adversarial information learned by the complex model from the adversarial examples, so it has good robustness to adversarial attacks.

V. Simulation Results and Discussion

In this section, we will analyze the performance of the proposed DLIM and DAT through simulation.

A. Double Loop Parameters

To study the influence of the double loop parameters on the performance of the proposed attack method and test the effectiveness of adding an external loop layer, we select different values of the internal and external loop parameters to attack the target model. Under the condition of maximum perturbation value $\varepsilon = 0.0015$, the adversarial examples generated by different loop parameters are input into the trained ResNet to test the recognition accuracy of the model for these examples. We generate adversarial examples to attack the target model in batches using the test set and the attack algorithm, and take the average of the accuracy of the target model after being attacked, as shown in Fig. 7.

In Fig. 7, we show the impact of loop parameters on the attack performance of DLIM. The accuracy of the model without attack is 0.921. We analyze the accuracy of the recognition model when SNR $= 10$ dB. When $N = M = 5$, the recognition accuracy is 0.468. When $N = 25$ and $M = 1$, the recognition accuracy is 0.503, which is 3.5% higher than that when $N = M = 5$. When $N = M = 10$, the recognition accuracy is 0.452. When $N = 100$ and $M = 1$, the recognition accuracy is 0.498, which is 4.6% higher than that when $N = M = 10$. Therefore, increasing the external loop layer does not simply increase the number of overall iterations. It can increase the prediction loss of the target model as much as possible by initializing the iterative conditions, and generate adversarial examples with stronger attack performance under the norm constraint.

For the traditional attack algorithm, which has only one loop, its algorithmic complexity is $\mathcal{O}(N)$. We can see from Algorithm 1 that the algorithmic complexity of the proposed DLIM is $\mathcal{O}(MN)$. Therefore, DLIM has the same algorithmic complexity at $M = N = 10$ as the traditional algorithm at $N = 100$, but it has a more significant attack performance as can be seen in Fig. 7.

B. Perturbation Level

Next, we study the influence of perturbation on the attack performance for $N = M = 10$ and SNR $= 10$ dB. We select different perturbation values in the interval $[0, 0.003]$ to generate adversarial examples, and use ResNet to identify these examples. The accuracy of the model is shown in Fig. 8.

In Fig. 8, we show the impact of perturbation constraint on the performance of different attack methods. After being attacked, the accuracy decreases with the increase of perturbation constraint. It can be seen from the decrease in recognition accuracy that among the four traditional attacks, the attack performances of BIM and PGD are basically the same, both stronger than FGSM, and MIM is the strongest. The proposed DLIM has the strongest attack performance. For the adversarial examples generated by different attack methods, DLIM has the best attack effect when the perturbation constraint is the same. If the accuracy of the model is the same after different attacks, the perturbation required by DLIM is smaller, which means better concealment. When $\varepsilon = 0.0021$, the attack performance of the adversarial examples generated by DLIM

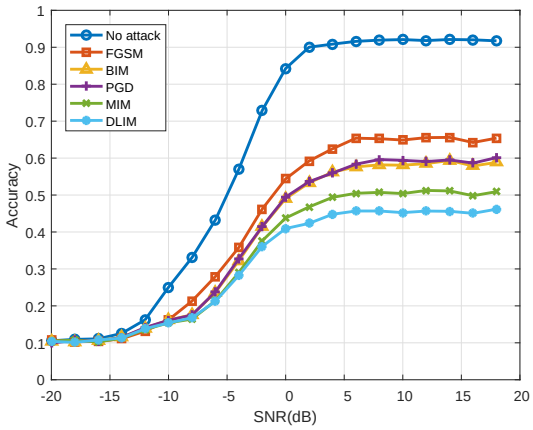


Fig. 9. Accuracy curves of ResNet at different SNRs.

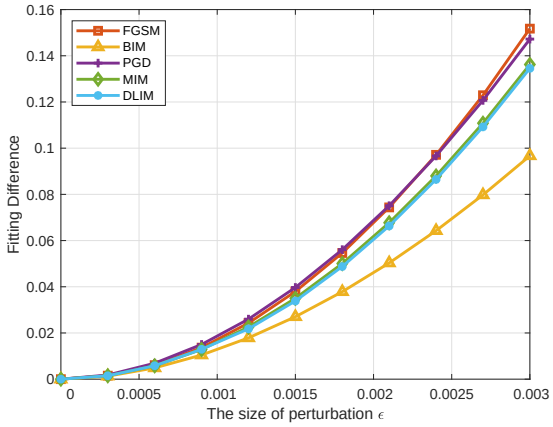


Fig. 10. FD values of attacks under different perturbation constraints.

TABLE III

Reduced Accuracy of ResNet at Different SNRs (%)

	FGSM	BIM	PGD	MIM	DLIM
−6dB	15.3	19.6	19.5	21.9	**22.1**
−4dB	21.1	24.7	24.2	28.0	**28.7**
−2dB	26.7	31.5	31.4	35.3	**36.8**
0dB	29.9	35.2	34.8	40.5	**43.3**
2dB	30.8	36.7	36.4	43.2	**47.7**

is stronger than that generated by MIM, which makes the accuracy of the recognition model decrease by 8.5% over the MIM attack.

C. Signal-to-Noise Ratio

When $N = M = 10$ and $\varepsilon = 0.0015$, adversarial examples of different SNRs are generated by the attack methods, and they are input into the recognition model to study the influence of SNR on the attacks.

In Fig. 9, we show the relationship between the recognition accuracy and SNR when the model is attacked. When SNR is very small, the noise will submerge the adversarial perturbation we designed, which will lead to a small feedback of the model for the perturbation, so that there is no significant difference in the accuracy of the model after different attacks. As the SNR increases, the advantage of the proposed attack method increases. After the curve converges, when SNR is the same, DLIM makes the accuracy decrease the most. When $\mathrm{SNR} = 12$ dB, the recognition accuracy reduced by DLIM is 5.6% higher than that of MIM.

To clearly show the effect of the attack in relation to the SNR, we selected the reduced accuracy of the target model at -6 dB, -4 dB, -2 dB, 0 dB and 2 dB after being attacked as the attack effect, as shown in Table III.

We can see from Table III that when the accuracy of the model decreases close to 30%, the SNRs at which the FGSM, BIM, PGD, MIM, and DLIM are located are 0 dB, -2 dB, -2 dB, -4 dB, and -4 dB, respectively. Meanwhile, when the accuracy of the model decreases close to 43%, the SNRs at which the MIM and DLIM are located are 2 dB and 0 dB, respectively. Therefore, DLIM requires the smallest SNR when making the target model lose the same accuracy. In addition, DLIM has the best attack results compared to other methods at each of the selected SNRs, which indicates that DLIM outperforms other methods at low SNRs. In fact, from (22), when the PNR is certain, the perturbation value can be appropriately increased as the SNR decreases, which will enhance the performance of the attack, as will be introduced below.

D. Perceptual Invisibility

In the previous experiments, we compared the performances of different attacks and found that the proposed DLIM has the best attack performance. In order to evaluate the perceptual invisibility of attacks, we will compare the waveform similarity between clean examples and adversarial examples.

We use FD to quantitatively evaluate the perceived invisibility of different attack methods. We select the test set of the data set to generate adversarial examples and calculate the average FD between them and clean examples. The results are shown in Fig. 10. From Fig. 10, it can be seen that the FD of BIM is the smallest and the fitting effect is the best, MIM and the proposed DLIM are second and basically the same. It shows that these three methods produce fewer traces of attack, good waveform similarity, and low risk of attack being detected. However, when $\varepsilon \leq 0.003$, these attack methods all satisfy the perturbation's imperceptible criterion. From Fig. 9, when $\mathrm{SNR} = 10$ dB and $\varepsilon = 0.0015$, the attack success rate of DLIM is 12.8% and 5.2% higher than that of BIM and MIM respectively.

When $\mathrm{SNR} = 10$ dB and $\varepsilon = 0.0015$, according to (22), $\mathrm{PNR} = -6.5$ dB, which satisfies the imperceptible standard $\mathrm{PNR} \leq 0$ dB of the perturbation, indicating that the perturbation has good concealment. After using DLIM to generate adversarial examples of different modulation signals, we draw the time domain waveforms of these examples and the corresponding clean examples according to (13), as shown in Fig. 11. It can be seen from Fig. 11 that the waveform of these signals changes little before and after the attack, but this small change can deceive the recognition model to classify the signals incorrectly.

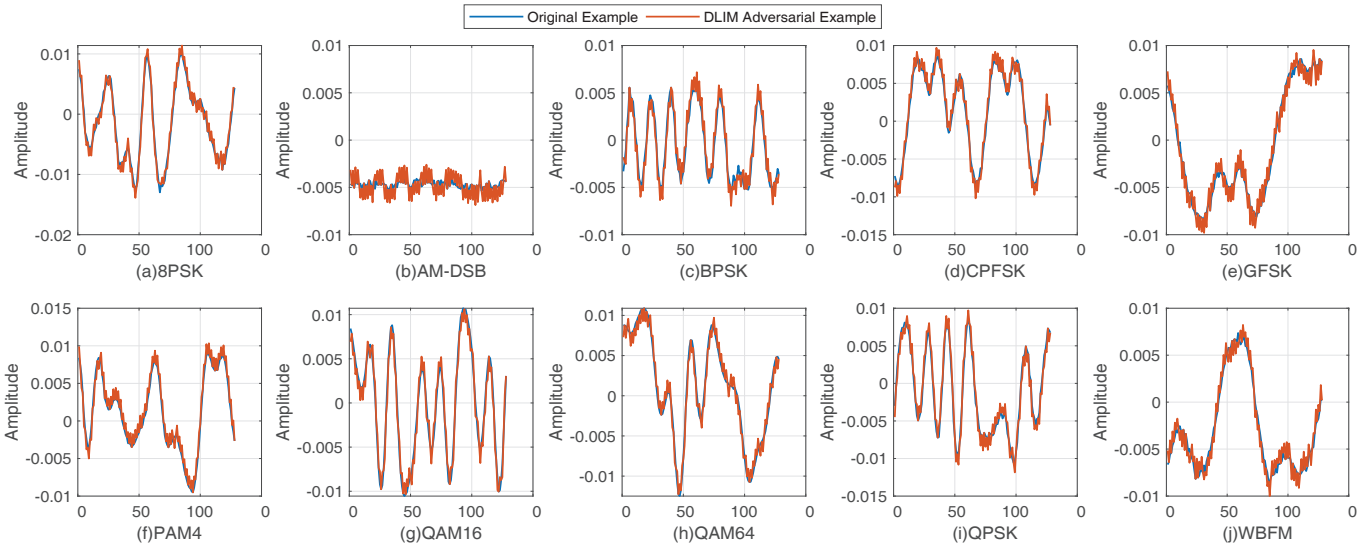
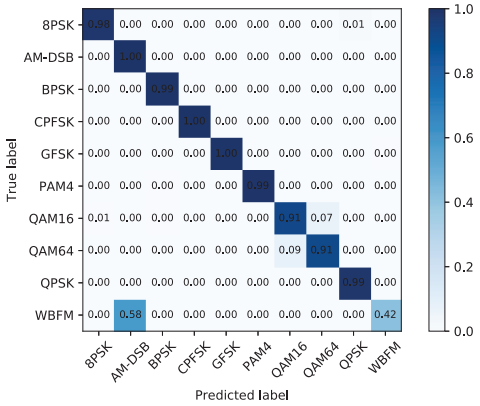


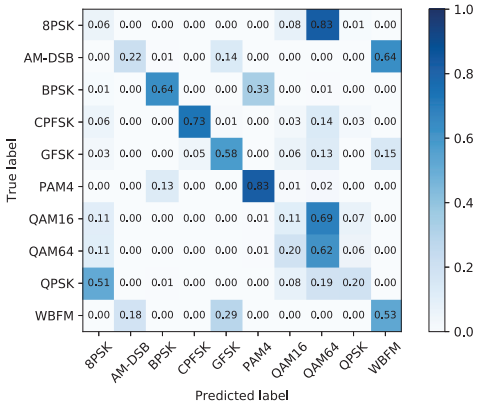
Fig. 11. Waveform of modulation signals before and after DLIM attack.

The confusion matrix can intuitively show the classification results of the model for adversarial examples after being attacked, which is helpful in analyzing the difficulty of each type of modulation signal being attacked. In order to test the effectiveness of the proposed attack method, the confusion matrix of the model before and after the attack can be compared. Using $\mathrm{SNR} = 10$ dB and $\varepsilon = 0.0015$, DLIM is used to generate adversarial examples, which are input into the model. The result is shown in Fig. 12.

It can be seen from Fig. 12 (a) that the trained ResNet model has good recognition ability. From the probability value on the diagonal of the confusion matrix in Fig. 12 (b), it can be seen that the prediction matrix of the model becomes confused after being attacked, and it is easy to misidentify the categories of most modulated signals, which indicates that DLIM has strong attack performance.



(a) No Attack



(b) DLIM Attack

Fig. 12. Confusion matrix of ResNet under MIM and DLIM attacks.

E. *Adversarial Defense*

To verify the effectiveness of the proposed adversarial distillation defense method DAT, we choose VTCNN with a simple structure as the target model. The network consists of only two convolutional layers and two fully connected layers. It consumes very small storage and computing resources and can be used in some resource-constrained devices. To show the effectiveness of the proposed defense method, we have performed normal training, adversarial training and distillation-based adversarial training on VTCNN to obtain VTCNN, VTCNN-AT, and VTCNN-DAT, respectively. Using $\mathrm{SNR} = 10$ dB and $\varepsilon = 0.0015$, we implement different attacks on the trained VTCNN, VTCNN-AT and VTCNN-DAT. The recognition results of the models are shown in Table IV.

It can be seen from Table IV that due to the simple structure of VTCNN, the model has limited accuracy and is vulnerable to attacks. Before the adversarial training, the accuracy of the model has been greatly reduced after being attacked, showing a strong vulnerability to adversarial examples. After the adversarial training, the VTCNN-AT obtained by the traditional adversarial training method can significantly improve the accuracy of the model. Compared with VTCNN-AT, the VTCNN-DAT obtained by the proposed DAT method can further improve the accuracy of the model by about 5%, and

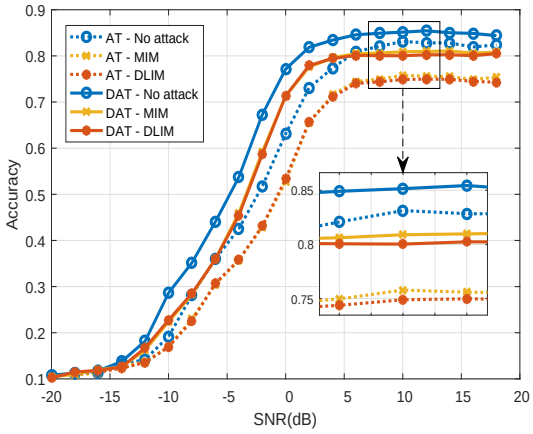


Fig. 13. Accuracy of VTCNN under different defenses.

TABLE IV
The Recognition Accuracy of Different Models (%)

	No attack	FGSM	BIM	PGD	MIM	DLIM
VTCNN	**86.5**	31.3	26.9	27.4	21.8	18.5
VTCNN – AT	83.1	73.4	76.6	75.9	75.8	74.9
VTCNN – DAT	85.2	**79.4**	**81.6**	**80.8**	**80.9**	**80.0**

weaken the damage of the attack to the model.

In addition, we use MIM and the proposed DLIM to study the defense effect of the model at different SNRs, as shown in Fig. 13.

We can see from Fig. 13 that the recognition performance of the model trained with the proposed DAT is better than AT at different SNRs, both for clean examples and for adversarial examples. Therefore, the proposed defense method can effectively improve the robustness of the model.

VI. Conclusion

This paper has studied the security of the vulnerable automatic modulation recognition model in cognitive radio-enabled IOT. We have proposed a double loop iteration method by adding an external loop iteration layer and designing an external iteration step to update the initial conditions of the iterative attack. In addition, for simple models in devices with limited storage and computing resources, we have proposed an adversarial training method based on knowledge distillation. Simulation results show that the proposed attack method has better attack performance than traditional attacks when the perturbation is perceptually invisible, and that the proposed defense method can improve the defense performance of the traditional adversarial training.

Appendix A
Proof of Proposition 1

The prediction probability of a network with weight vector and bias vector $\boldsymbol{\omega}$ and b, respectively, for a certain adversarial example $\boldsymbol{x}^*$, is

$$p_i\left(\boldsymbol{x}^*\right)=\frac{\exp\left(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}^*+b\right)}{\sum\limits_{k=1}^{K}\left(\exp\left(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}^*+b\right)\right)}, \tag{34}$$

where K is the number of classes of signals

Since $\boldsymbol{x}^*=\boldsymbol{x}+\boldsymbol{\eta}$ and the true label $l_i(\boldsymbol{x}^*)=l_i(\boldsymbol{x})$, the prediction loss can be expressed as

$$\begin{aligned}\mathcal{L}\left(\boldsymbol{x}^*,\boldsymbol{l}\right)&=-\sum_{k=1}^{K}l_k\left(\boldsymbol{x}^*\right)\log_2\left(p_k\left(\boldsymbol{x}^*\right)\right)\\&=-\sum_{k=1}^{K}l_k\left(\boldsymbol{x}\right)\log_2\left(\frac{\exp\left(\boldsymbol{\omega}^{\mathrm{T}}\left(\boldsymbol{x}+\boldsymbol{\eta}\right)\right)}{\sum\limits_{k=1}^{K}\left(\exp\left(\boldsymbol{\omega}^{\mathrm{T}}\left(\boldsymbol{x}+\boldsymbol{\eta}\right)\right)\right)}\right)\\&=-\sum_{k=1}^{K}l_k\left(\boldsymbol{x}\right)\log_2\left(\frac{\exp\left(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}\right)\exp\left(\alpha\boldsymbol{\omega}^{\mathrm{T}}\mathrm{sign}(\boldsymbol{g})\right)}{\sum\limits_{k=1}^{K}\left(\exp\left(\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{x}\right)\exp\left(\alpha\boldsymbol{\omega}^{\mathrm{T}}\mathrm{sign}(\boldsymbol{g})\right)\right)}\right)\end{aligned} \tag{35}$$

where α is the size of perturbation, $\boldsymbol{g}$ is the accumulated gradient and $\mathrm{sign}(\boldsymbol{g})$ represents the direction of perturbation. Since the network parameters $\boldsymbol{\omega}$ of the trained model are fixed, the prediction loss of the model for an adversarial example $\boldsymbol{x}^*$ generated from a clean example $\boldsymbol{x}$ is related to α and $\boldsymbol{g}$, that is

$$\mathcal{L}\left(\boldsymbol{x}^*,\boldsymbol{l}\right)=\mathcal{L}\left(\boldsymbol{x},\alpha,\boldsymbol{g}\right). \tag{36}$$

[39] indicates that when the model loss does not exceed the predicted loss threshold $\mathcal{L}_T$, the model will make a correct prediction. The predicted class of the model for an adversarial example $\boldsymbol{x}^*$ is

$$y_p\left(\boldsymbol{x}^*\right)=\arg\max_{y_k}\left\{p\left(\boldsymbol{y}|\boldsymbol{x}^*\right)\right\},\quad k=1,2,\cdots,K, \tag{37}$$

so

$$\begin{cases}y_p\left(\boldsymbol{x}^*\right)=y_t, & \mathcal{L}\left(\boldsymbol{x},\alpha,\boldsymbol{g}\right)\le\mathcal{L}_T\\ y_p\left(\boldsymbol{x}^*\right)\neq y_t, & \mathcal{L}\left(\boldsymbol{x},\alpha,\boldsymbol{g}\right)>\mathcal{L}_T\end{cases} \tag{38}$$

where y_p represents the predicted class of the model for the adversarial example, and y_t represents the true class.

Appendix B
Proof of Proposition 2

We denote $p\left(x^*\right)$ as p, and substitute (30) into the cross entropy loss of the model to obtain

$$\begin{aligned}\mathcal{L}\left(\boldsymbol{x}^*,\tilde{\boldsymbol{l}}\right)&=-\sum_{k=1}^{K}\tilde{l}_k\cdot\log_2\left(p_k\right)\\&=-\sum_{k=1}^{K}\left(\left(1-\beta\right)\cdot\delta_{k,l}+\beta\cdot u(k)\right)\cdot\log_2\left(p_k\right)\\&=\left(1-\beta\right)\left[-\sum_{k=1}^{K}\delta_{k,l}\log_2\left(p_k\right)\right]+\beta\left[-\sum_{k=1}^{K}u(k)\log_2\left(p_k\right)\right]\\&=\left(1-\beta\right)\cdot\mathcal{L}\left(\boldsymbol{x}^*,\boldsymbol{l}\right)+\beta\cdot\mathcal{L}\left(\boldsymbol{x}^*,\boldsymbol{u}\right)\end{aligned} \tag{39}$$

REFERENCES

[1] M. Liu and Z. Zhang, "Adversarial attacks on deep neural network based modulation recognition," in *Proc. 9th Int. Conf. Depend. Syst. Appl.*, 2022, pp. 1053-1054.

[2] A. A. Khan, M. H. Rehmani and M. Reisslein, "Cognitive radio for smart grids: Survey of architectures, spectrum sensing mechanisms, and networking protocols," *IEEE Commun. Surv. Tutorials*, vol. 18, no. 1, pp. 860-898, 1st Quart. 2016.

[3] Y. Aborahama and M. S. Hassan, "On the stochastic modeling of the holding time of SUs to PU channels in cognitive radio networks," *IEEE Trans. Cognit. Commun. Netw.*, vol. 6, no. 1, pp. 282-295, Mar. 2020.

[4] X. Hao, T. Yang, Y. Hu, H. Feng and B. Hu, "An adaptive matching bridged resource allocation over correlated energy efficiency and AoI in CR-IoT system," *IEEE Trans. Green Commun. Netw.*, vol. 6, no. 1, pp. 583-599, Mar. 2022.

[5] T. J. O'Shea, T. Roy and T. C. Clancy, "Over-the-air deep learning based radio signal classification," *IEEE J. Sel. Top. Signal Process.*, vol. 12, no. 1, pp. 168-179, Feb. 2018.

[6] S. Chang, S. Huang, R. Zhang, Z. Feng and L. Liu, "Multitask-learning-based deep neural network for automatic modulation classification," *IEEE Internet Things J.*, vol. 9, no. 3, pp. 2192-2206, Feb. 2022.

[7] S. Kim, H.-Y. Yang and D. Kim, "Fully complex deep learning classifiers for signal modulation recognition in non-cooperative environment," *IEEE Access*, vol. 10, pp. 20295-20311, 2022.

[8] K. Bu, Y. He, X. Jing and J. Han, "Adversarial transfer learning for deep learning based automatic modulation classification," *IEEE Signal Process. Lett.*, vol. 27, pp. 880-884, 2020.

[9] P. Ghasemzadeh, M. Hempel and H. Sharif, "GS-QRNN: A high-efficiency automatic modulation classifier for cognitive radio IoT," *IEEE Internet Things J.*, vol. 9, no. 12, pp. 9467-9477, Jun. 2022.

[10] Y. Wang, J. Bai, Z. Xiao, H. Zhou and L. Jiao, "MsmcNet: A modular few-shot learning framework for signal modulation classification," *IEEE Trans. Signal Process.*, vol. 70, pp. 3789-3801, 2022.

[11] 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, Jun. 2023.

[12] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna and R. Fergus, "Intriguing properties of neural networks," in *Proc. Int. Conf. Learn. Represent.*, pp. 1-10, May 2015.

[13] I. Goodfellow, J. Shlens and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. Int. Conf. Learn. Represent.*, pp. 189-199, Mar. 2015.

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

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

[16] Y. Dong *et al.*, "Boosting adversarial attacks with momentum," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, pp. 9185-9903, Mar. 2018.

[17] Y. Zhang, X. Tian, Y. Li, X. Wang and D. Tao, "Principal component adversarial example," *IEEE Trans. Image Process.*, vol. 29, pp. 4804-4815, 2020.

[18] Z. Xiong, H. Xu, W. Li and Z. Cai, "Multi-source adversarial sample attack on autonomous vehicles," *IEEE Trans. Veh. Technol.*, vol. 70, no. 3, pp. 2822-2835, Mar. 2021.

[19] H. Lv, M. Wen, R. Lu and J. Li, "An adversarial attack based on incremental learning techniques for unmanned in 6G scenes," *IEEE Trans. Veh. Technol.*, vol. 70, no. 6, pp. 5254-5264, Jun. 2021.

[20] Y. Lin, H. Zhao, X. Ma, Y. Tu and M. Wang, "Adversarial attacks in modulation recognition with convolutional neural networks," *IEEE Trans. Reliab.*, vol. 70, no. 1, pp. 389-401, Mar. 2021.

[21] P. Qi, T. Jiang, L. Wang, X. Yuan and Z. Li, "Detection tolerant black-box adversarial attack against automatic modulation classification with deep learning," *IEEE Trans. Reliab.*, vol. 71, no. 2, pp. 674-686, Jun. 2022.

[22] B. Kim, Y. E. Sagduyu, K. Davaslioglu, T. Erpek and S. Ulukus, "Channel-aware adversarial attacks against deep learning-based wireless signal classifiers," *IEEE Trans. Wireless Commun.*, vol. 21, no. 6, pp. 3868-3880, Jun. 2022.

[23] 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. Reliab.*, vol. 72, no. 2, pp. 431-444, Jun. 2023.

[24] L. Zhang, S. Lambotharan, G. Zheng, G. Liao, A. Demontis and F. Roli, "A hybrid training-time and run-time defense against adversarial attacks in modulation classification," *IEEE Wireless Commun. Lett.*, vol. 11, no. 6, pp. 1161-1165, Jun. 2022.

[25] M. Z. Hameed, A. György and D. Gündüz, "The best defense is a good offense: Adversarial attacks to avoid modulation detection," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, pp. 1074-1087, 2021.

[26] D. Wang, C. Li, S. Wen, S. Nepal and Y. Xiang, "Defending against adversarial attack towards deep neural networks via collaborative multi-task training," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 953-965, Mar. 2022.

[27] X. Yuan, P. He, Q. Zhu and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 9, pp. 2805-2824, Sept. 2019.

[28] N. Akhtar, A. Mian, N. Kardan and M. Shah, "Advances in adversarial attacks and defenses in computer vision: A survey," *IEEE Access*, vol. 9, pp. 155161-155196, 2021.

[29] Z. Che *et al.*, "SMGEA: A new ensemble adversarial attack powered by long-term gradient memories," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 3, pp. 1051-1065, Mar. 2022.

[30] Y. Zhong and W. Deng, "Towards transferable adversarial attack against deep face recognition," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, pp. 1452-1466, 2021.

[31] DeepSig, "Deepsig dataset: Radioml 2016.10b," [Online] Available: *https://www.deepsig.io/datasets*, 2016.

[32] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, pp. 770-778, 2016.

[33] C. Liu, M. Salzmann, T. Lin, R. Tomioka and S. Süsstrunk, "On the loss landscape of adversarial training: Identifying challenges and how to overcome them," in *Adv. Neural Inf. Process. Syst.*, 2020.

[34] M. Sadeghi and E. G. Larsson, "Adversarial attacks on deep-learning based radio signal classification," *IEEE Wireless Commun. Lett.*, vol. 8, no. 1, pp. 213-216, Feb. 2019.

[35] R. Sahay, C. G. Brinton and D. J. Love, "A deep ensemble-based wireless receiver architecture for mitigating adversarial attacks in automatic modulation classification," *IEEE Trans. Cognit. Commun. Netw.*, vol. 8, no. 1, pp. 71-85, Mar. 2022.

[36] H. Zhao, Y. Lin, S. Gao and S. Yu, "Evaluating and improving adversarial attacks on DNN-based modulation recognition," in *IEEE Glob. Commun. Conf.*, pp. 1-5, 2020.

[37] X. Jia, Y. Zhang, B. Wu, J. Wang and X. Cao, "Boosting fast adversarial training with learnable adversarial initialization," *IEEE Trans. Image Process.*, vol. 31, pp. 4417-4430, 2022.

[38] B. Heo, M. Lee, S. Yun and J. Y. Choi, "Knowledge transfer via distillation of activation boundaries formed by hidden neurons," in *AAAI Conf. Artif. Intell.*, pp. 3779-3787, 2019.

[39] 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.*, early access, Apr. 04, 2023, doi: 10.1109/TITS.2023.3262347.