electronics

MDPI

Article

Sparse Adversarial Attacks against DL-Based Automatic Modulation Classification

Zenghui Jiang, Weijun Zeng *, Xingyu Zhou, Peilun Feng, Pu Chen, Shenqian Yin, Changzhi Han and Lin Li

College of Communication Engineering, Army Engineering University of PLA, Qin Huai District, Nanjing 210007, China; jqianxi@163.com (Z.J.); universezhou@sina.cn (X.Z.); eternityer_j@126.com (P.F.); puchen0127@163.com (P.C.); ysqyaqian@126.com (S.Y.); hanchangzhi1234@163.com (C.H.); 18519873080@163.com (L.L.)

* Correspondence: zwj3103@126.com

Abstract: Automatic modulation recognition (AMR) serves as a crucial component in domains such as cognitive radio and electromagnetic countermeasures, acting as a significant prerequisite for the efficient signal processing of receivers. Deep neural networks (DNNs), despite their effectiveness, are known to be vulnerable to adversarial attacks. This vulnerability has inspired the introduction of subtle interference to wireless communication signals—interference so minuscule that it is difficult for the human eye to discern. Such interference can mislead eavesdroppers into erroneous modulation pattern recognition when using DNNs, thereby camouflaging communication signal modulation patterns. Nonetheless, the majority of current camouflage methods used for electromagnetic signal modulation recognition rely on a global perturbation of the signal. They fail to consider the local agility of signal disturbance and the concealment requirements for bait signals that are intercepted by the interceptor. This paper presents a generator framework designed to produce perturbations with sparse properties. Furthermore, we introduce a method to reduce spectral loss, which minimizes the spectral difference between adversarial perturbation and the original signal. This method makes perturbation more challenging to monitor, thereby deceiving enemy electromagnetic signal modulation recognition systems. The experimental results validated that the proposed method significantly outperformed existing methods in terms of generation time. Moreover, it can generate adversarial signals characterized by high deceivability and transferability even under extremely sparse conditions.

Keywords: modulation recognition; wireless security; deep learning; neural networks; adversarial attack; adversarial examples

Citation: Jiang, Z.; Zeng, W.; Zhou, X.; Feng, P.; Chen, P.; Yin, S.; Han, C.; Li, L. Sparse Adversarial Attacks against DL-Based Automatic Modulation Classification. *Electronics* **2023**, *12*, 3752. https://doi.org/10.3390/electronics12183752

Academic Editor: Hirokazu Kobayashi

Received: 9 July 2023
Revised: 22 August 2023
Accepted: 24 August 2023
Published: 5 September 2023

1. Introduction

As various communication systems have emerged, electromagnetic data have exhibited complex and diverse characteristics such as randomness, heterogeneity, and vastness. In the cooperative communication field, communication signals employ different modulation methods to meet user requirements and fully utilize channel capacity. Modulation recognition technology allows the collection and extraction of signal features like the signal spectrum, instantaneous amplitude, and instantaneous phase, thus facilitating efficient signal processing. In non-cooperative communication, modulation recognition significantly alleviates the scarcity of spectrum resources, serving as an important technique in both civil and military applications [1,2]. Deep learning (DL), in its ascendancy, has fostered considerable interest in automatic modulation recognition (AMR) technology based on DL in recent years [3,4]. Compared to traditional methods, DL employs deep neural networks (DNNs) as classifiers and offers several advantages:

- The utilization of vast amounts of communication data can considerably enhance modulation recognition accuracy.

- Automated feature extraction circumvents the constraints of professional knowledge and manual experience.
- The potential to continuously benefit from the rapid iteration and evolution of DL tools provides a way to tackle intricate modulation recognition problems [5].

However, as research on deep learning technology continues to deepen, it brings both convenience and security risks. There exist malicious third parties who can intercept the transmission signals of a transmitter, utilizing automatic modulation recognition technology to identify the transmitter's modulation method, modulation parameters, and demodulation information. Subsequently, they can modulate information based on the receiver's modulation method and forward it to the receiver, leading to theft and the alteration of information. Consequently, it has become increasingly imperative to explore effective ways of enhancing the disguise of communication signal modulation modes, making this an urgent research topic.

Researchers have found that DNNs are vulnerable to adversarial example attacks [6]. Adversarial samples are input examples that have been altered by small, often imperceptible, perturbations to mislead the network into producing incorrect outputs with high confidence. These are especially prevalent in deep learning image classification tasks, where perturbations, virtually undetectable to the human eye, are added to recognized images, leading the model to generate erroneous results. This susceptibility of DNNs to subtle interference inspired us to apply a similar strategy to wireless communication signals. We introduce interference that is almost imperceptible to the human eye and can mislead an enemy's DNN into incorrect modulation pattern recognition, thus achieving the goal of camouflaging communication signal modulation patterns [7]. Such a scenario includes a sender, a legitimate receiver, and a third-party eavesdropper. The sender sends the signal, the receiver receives the signal, and the eavesdropper intercepts the modulated signal and employs a deep-learning-based modulation recognition model to demodulate it before employing subsequent attack methods. At this time, a small perturbation is added to the signal at the sender, so that the eavesdropper cannot correctly identify the modulation mode, in order to achieve the purpose of hiding the modulation mode.

In this scenario, most previous methods did not consider the concealment of adversarial signals and relied directly on the adversarial attack method to generate adversarial signals. The definition of imperceptibility in image adversarial examples is not applicable to signal adversarial examples, which often use indicators such as the bit error rate to measure the quality of signals during communication. Therefore, we hoped to minimize the number of perturbation points in order to minimize the impact on communication [8]. Moreover, previous research has indicated that adversarial perturbations can lead to adversarial examples failing to retain the same spectral shape as the original signal [9]. Such alterations to the signal spectrum shape can compromise the effectiveness of deception, as eavesdroppers can employ pre-processing stages to diminish the impact of perturbations, e.g., by using filters. Additionally, these spectral shape changes might alert eavesdroppers to the occurrence of attacks [10].

To address the above issues, this paper proposes a method called spectrum-similarity sparse adversarial attack (SSAA). Specifically, the perturbations are first decoupled into signal position masks and amplitude vectors, and then a generator is utilized to jointly optimize and generate these two vectors. By multiplying the two vectors, adversarial perturbations with transferability and sparsity properties can be generated. Compared to dense perturbations, highly sparse attacks on resistant perturbations are more dangerous, because they are less likely to be detected. Additionally, this paper introduces a spectrum deception loss metric to maintain consistency during the training process and achieve the goal of reducing spectrum variation.

Our contributions are summarized as follows:

- Unlike previous work, a new attack method is proposed based on generators, called SSAA. It can generate adversarial signals with sparse and transitive properties.

- Considering the damage inflicted by adversarial perturbations on the spectrum of adversarial signals, spectral loss is introduced to maintain the spectral consistency of adversarial perturbations during training.
- The performance of the proposed method was evaluated in RADIOML 2016.10a, and it was shown to significantly reduce the generation time of perturbations compared to existing methods and could achieve a better fooling rate and transferability than existing methods in extremely sparse situations.

The remaining portion of this document is organized as follows: Section 2 presents the preliminary concepts of adversarial attacks. Section 3 formulates the problem and provides a detailed explanation of the proposed method. Section 4 provides the results, the list of relevant research papers, and a brief description of the observed trends. Section 5 summarizes the paper and then states the challenges and limitations of the proposed method.

2. Related Work

Adversarial examples. Adversarial examples were initially introduced by Szegedy et al. [6]. They demonstrated that by adding imperceptible perturbations to an image, according to certain rules, even a well-performing convolutional neural network (CNN) model can be easily misled. These perturbations trigger the classifier network to dramatically change its prediction, leading to a highly confident yet incorrect classification. These perturbed samples are known as adversarial examples and can cause neural networks to make erroneous classifications. Attacks that employ such adversarial examples are typically referred to as adversarial attacks.

Assuming that x is the original In-phase and Quadrature (IQ) signal, an adversarial example for modulation recognition can be defined as follows:

$$\tilde{x} = x + \theta, \quad s.t.\ f(x_{adv})! = f(x),\ ||\theta||_p < \varepsilon, \tag{1}$$

where f denotes the classifier, θ is the adversarial perturbation, and ε represents the upper limit of the perturbation scale.

Adversarial attacks. Adversarial attacks can be categorized based on several classification criteria. One way to distinguish between these attacks is by looking at the classification results they yield: targeted attacks and non-targeted attacks. Targeted attacks are designed to mislead the model into predicting a sample as a specific, incorrect label. Conversely, non-targeted attacks aim to deceive the model into predicting a sample as any label, provided that it is different from the correct one.

Adversarial attacks can also be subdivided, based on the level of knowledge the attacker has about the target model, into white-box and black-box attacks. White-box attacks occur when the attacker has comprehensive access to all information about the target model. This includes, but is not limited to, its structure, parameters, training set, and test set. Typically, white-box attacks result in the most effective model compromise. On the other hand, black-box attacks transpire when the attacker lacks access to the structure and parameters of the target model. In this case, attacks are designed and implemented by interacting with the model through normal inputs. Typically, in communication, it is challenging for the attacker to learn the input/output information of the target model, making this a characteristic black-box attack scenario.

In order to achieve better attack effectiveness, various types of adversarial example attacks have been proposed, such as the Fast Gradient Sign Method (FGSM) [11], C&W attacks [12], Projected Gradient Descent (PGD) [13], DeepFool [14], and Universal Adversarial Perturbation (UAP) [15]. In early research, adversarial examples were almost all generated on known models, known as white-box attacks. Considering real-world situations, in this paper, more emphasis is placed on how to effectively implement models in black-box settings, such as transferable black-box adversarial attacks and decision-based black-box adversarial attacks. This paper mainly considers generator-based transferable black-box adversarial attacks. To accomplish the objective of camouflage modulation, a

substitute model was constructed, and a white-box attack method was employed to generate adversarial examples. Subsequently, the transferability of these adversarial examples was utilized to mislead the target model.

Generator-based adversarial examples are generated by a signal-to-signal generator architecture, which can learn to generate perturbation signals that are indistinguishable from the original signals. The authors of [16] proposed ATN, which takes the original signal as input and generates adversarial examples through the generator, while the authors of [17] proposed GAP, which generates adversarial perturbations from the original signal. The difference between these methods is that ATN directly generates adversarial examples, while GAP only generates adversarial perturbations. The authors of [18] used an adversarial generative network (GAN) to generate adversarial examples that were more difficult to distinguish; the authors of [19] considered generating unrestricted adversarial examples using a conditional generator without any prior information. The above-mentioned methods were all focused on global perturbation. In this paper, a spectral similarity sparse generator architecture is proposed to generate adversarial examples with spectral similarity sparsity for signals.

Sparse adversarial attack. In the current phase of adversarial attacks, most are based on the ℓ_2 and ℓ_∞ norms, resulting in waveforms grounded on global interference. This approach is not conducive to ensuring reliable communication processes and the concealment of deceptive signals. To address this issue, a sparse adversarial attack method based on the ℓ_0 norm is considered, such as a single-pixel attack [20] that achieves an extremely sparse attack by altering just one point in the sample.

Several methods have been proposed in this regard: The authors of [21] proposed the JSMA attack method, which identifies the influence of features on output classification by selecting the most effective pixels on the adversarial saliency map. The authors of [22] suggested projecting the adversarial noise generated by PGD onto the ℓ_0 sphere to achieve a sparse attack, proposing the PGD_0 attack method. Drawing on the idea of decoupling perturbations into direction and amplitude vectors, the authors of [23] generated waveforms with low intercept properties. The authors of [8] proposed an enhancement method based on differential evolution to construct sparse attacks that incorporate constraints related to visual differences and recoverability into the optimization process, with the aim of deceiving deep-learning-based AMR classifiers. The authors of [24] proposed a geometry-inspired sparse attack, Sparsefool, which utilizes the low mean curvature of the boundary to effectively calculate adversarial perturbations. This method can swiftly compute the sparse disturbance and can be efficiently extended to high-dimensional data.

However, the Sparsefool method relies on the gradient information of the target model to generate samples, making it highly targeted and prone to overfitting, resulting in poor transferability. In this paper, we propose decoupling perturbation into signal position masks and amplitude vectors and then using a generator to create adversarial examples with transferability and sparsity properties.

3. Spectrum-Similarity Sparse Adversarial Attacks

3.1. Problem Analysis

Modulation recognition can be seen as a classification problem with M modulation schemes. In the communication process, the received signal at the receiver end is denoted as

$$r = hx + \sigma, \tag{2}$$

where x represents the signal modulated by the transmitter in a specific modulation scheme, h denotes channel gain, and σ refers to additive white Gaussian noise (AWGN). The objective of any modulation classifier is to determine $f(r)$ given the received signal r, where f is the classifier.

For adversarial attacks targeted at modulation recognition, the goal is to obtain

$$\min_{\theta} ||\theta||_p + \lambda \mathcal{L}(f(x+\theta), y), \quad s.t. \ \|\theta\|_\infty \leq \varepsilon, \tag{3}$$

where θ denotes the adversarial perturbation, $\mathcal{L}$ represents the negative cross-entropy function, ε is the upper limit of the perturbation scale, λ is a hyperparameter controlling the importance (weight) of the loss function, and y denotes the true label. The aforementioned are only applicable for non-targeted attacks. For targeted attacks, the equations should be modified to $y = y_t$, where $\mathcal{L}$ should be set as the positive cross-entropy function and y_t represents the target category.

Depending on the purpose, the value of p can be specified differently. For example, when $p = 2$ (e.g., C&W [12]), adversarial perturbations can be added at all pixels, which is called a dense attack. On the other hand, if $p = 0$, the above problem will only cause interference at a few points, which is called a sparse attack. For sparse attacks, the aforementioned problem is described as

$$\min_{\theta} ||\theta||_0 + \lambda \mathcal{L}(f(x+\theta), y), \quad s.t. \ \|\theta\|_\infty \leq \varepsilon. \tag{4}$$

Equation (4) is an NP-hard problem. To address this issue, the ℓ_0 norm is approximated by the ℓ_1 norm. Then, we decouple the perturbation θ as follows:

$$\theta = \delta \odot \rho, \quad s.t. \delta \in \{0,1\}^{2\times N}, \rho \in (-1,1)^{2\times N}, \tag{5}$$

where δ denotes the vector of the binary mask, N represents the length of the signal, ρ denotes the vector of perturbation magnitudes, and $\odot$ represents the element-wise product. Therefore, by jointly optimizing δ and ρ, we can achieve the goal of optimizing θ.

Using this factorization, Equation (4) can be optimized as follows:

$$\min_{\theta} ||\delta \odot \rho||_1 + \lambda \mathcal{L}(f(x+\delta \odot \rho), y), \quad s.t. \ \|\delta \odot \rho\|_\infty \leq \varepsilon, \tag{6}$$

where the ℓ_0 norm above is approximately replaced with the ℓ_1 norm.

3.2. Framework

Figure 1 shows the overall framework, consisting of generator G and alternative model f. Generator G takes the original signal as input, outputs adversarial perturbations θ, and then inputs $x + \theta$ to the target classifier to calculate the adversarial loss, which represents the inverse value of the distance between the predicted and real label (non-targeted attack) or the distance between the predicted and target category y_t (targeted attack). The generator consists of an encoder and two decoders, and the encoder Enc is used to implement feature encoding by encoding x as $z = enc(x)$, which is then fed to two decoders, Dec1 and Dec2.

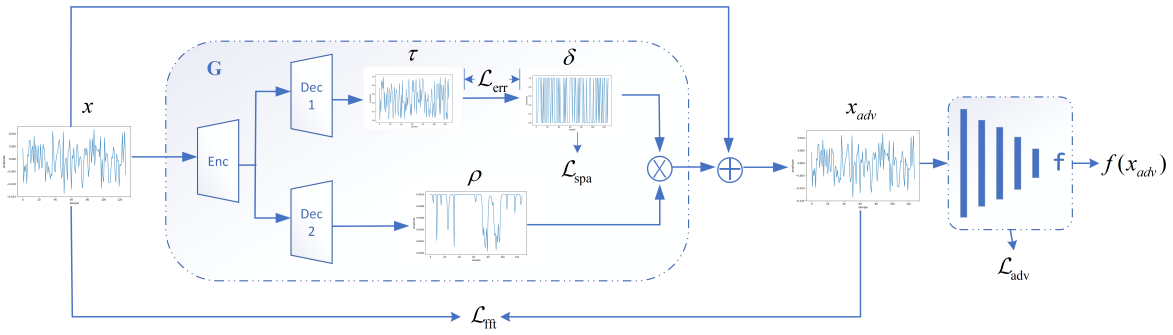

Figure 1. Overall framework.

Dec1 decodes the input z and outputs it as $\tau \in (0,1)^{2\times N}$, representing the probability of adding interference at each position. To obtain the binary position mask (a

two-dimensional array, where 1 indicates that the position should be perturbed), we set the position with a probability greater than p_s to 1 and the position with a probability less than p_s to 0 as follows:

$$\delta_{i,j} = \begin{cases} 1 & \tau_{i,j} > p_s \\ 0 & \tau_{i,j} \leq p_s \end{cases} \tag{7}$$

Obviously, such quantization operations will lead to the disappearance of gradients. By introducing a randomization operation [25], probability quantization is performed during the training phase, with each point randomly using quantization operations. Otherwise, the original values are used to preserve some gradients, and this operation is not taken during the testing phase.

Dec2 decodes the input z and outputs it as ρ, which represents the vector of perturbation magnitudes. The tanh function is used to limit ρ to $(-1, 1)$ and then multiply hyperparameter ε to control the ℓ_∞ norm of ρ.

Finally, we use the generator to generate δ and ρ and then multiply these by Equation (5) to obtain θ; adversarial example x_{adv} is obtained from Equation (1).

3.3. Loss Functions

Adversarial loss. The purpose of adversarial loss is to maximize the ability of adversarial examples to deceive eavesdroppers for classification. For example, for non-targeted attacks, the aim is to generate θ, which prevents eavesdroppers from correctly classifying $x + \theta$. We use the C&W loss function as the countermeasure loss, which is calculated by the confidence of the eavesdropper in the real label. The classification of the signal received by the eavesdropper is determined by the softmax output. When the source class is no longer the most likely class determined by the classifier, this decrease in confidence may lead to successful aimless attacks. Adversarial loss is defined as

$$\mathcal{L}_{adv}(x_{adv}, y, f) = \max\{f_y(x_{adv}) - \max_{i \neq y}\{f_i(x_{adv})\}, -\kappa\}, \tag{8}$$

where κ is the hyperparameter that is used to control the attack intensity, and $f(*)$ is the softmax layer output of the target model. For a targeted attack, the loss function is as follows:

$$\mathcal{L}_{\text{adv}}(x_{adv}, y_t, f) = \max\{\max_{i \neq y_t}\{f_i(x_{adv})\} - f_{y_t}(x_{adv}), -\kappa\}, \tag{9}$$

where y_t is the target category of the attack.

Spectrum loss. The loss measurement of the adversarial examples in the frequency domain is used to maintain the spectral integrity of the adversarial examples in the training, so that the eavesdropper can effectively avoid signal filtering and other pre-processing steps. Specifically, the original signal and disturbance are converted from the time domain to the frequency domain using the fast Fourier transform, and the gap between their frequency domains is calculated. The gap is determined using a mean square error (MSE) function, a regression loss function used to determine the difference between expected and actual values:

$$\mathcal{L}_{fft}(\hat{x}, \hat{\theta}) = \frac{1}{n}\sum_{i=1}^{n}(\hat{x} - \hat{\theta})^2, \tag{10}$$

where $\hat{x}$ and $\hat{\theta}$ are the frequency domain representations of the original signal and disturbance, respectively.

Sparse loss. For signal deception, it is necessary to ensure that the added disturbance is imperceptible and as small as possible. The purpose is to make deceptive signals more like legitimate signals deceiving eavesdroppers, thereby interfering with the other party's decisions, delaying and reducing the effectiveness of the other party's threat level assessment and adversarial decision making. This makes it difficult for the target model to accurately process and analyze the intercepted signal, ensuring that it does not affect the receiver's reception performance. Sparse perturbation is achieved by introducing the norm

of ℓ_0. However, for decoupled perturbations δ and ρ, ρ is continuous and difficult to solve. Sparsity is mainly controlled by δ, but the ℓ_0 norm is non-differentiable, so it needs to be approximated and replaced with the $\ell_1$norm:

$$\mathcal{L}_{spa}(\delta) = \|\delta\|_1. \tag{11}$$

Quantization loss. When using the ℓ_1 norm to approximate the replacement of the ℓ_0 norm, there is a random quantization step, which can lead to significant quantization errors between training and testing, resulting in poor test results. To solve this problem, it is necessary to reduce this quantization error and bring the training and testing results closer together. We introduce the mean square error of the quantization error as a loss to encourage the reduction of the quantization error between the training and testing stages:

$$\mathcal{L}_{\text{err}}(\tau, \delta) = \|\tau - \delta\|_2 \tag{12}$$

To summarize, the total loss function is shown below.

$$\mathcal{L}_{all} = \mathcal{L}_{adv} + \lambda_f \mathcal{L}_{fft} + \lambda_s \mathcal{L}_{spa} + \lambda_e \mathcal{L}_{err} \tag{13}$$

where λ_f, λ_s, and λ_e are hyperparameters that control the importance of spectrum loss, sparse loss, and quantization loss, respectively. The samples that λ_f encourages the model to generate are similar to the original sample's spectrum, while the samples that λ_s and λ_e encourage the model to generate are sparse.

4. Experiments

The experiment was implemented using the Python programming language and the Pytorch framework. The hardware used was an RTX3060ti graphics card. The following paragraphs introduce the setup of the experiment, including the database, target model, generator model, benchmark method, and parameter settings.

Database. In order to facilitate the evaluation of the performance of the proposed method, the experimental dataset was taken from the public dataset RADIOML 2016.10a [26]. This dataset was generated using the GNU Radio tool and is stored in IQ signal format, with the float data type and data dimensions of 2×128. It has a sample size of 220,000 and a signal-to-noise ratio range of -20 dB to 18 dB, and it includes 11 common modulation methods, of which 8 are digital modulation methods and 3 are analog modulation methods. The center frequency offset, sampling rate offset, additive white Gaussian noise, multipath, and fading of the stochastic process are comprehensively considered, which is similar to real communication scenarios, and it could be used to evaluate and compare the performance of the algorithm in various signal and noise environments.

Target model. We selected VT-CNN2 [27], CLDNN [28], and Resnet [29] as the target models. VT-CNN2 is a CNN model optimized and improved for the RADIOML2016.10a dataset (with layers, network parameters, and CNN initial weights modified for the dataset), consisting of multiple convolutional layers and fully connected layers. CLDNN is a network composed of a convolutional neural network, short-term memory network, and deep neural network. CNN is good at reducing frequency-domain changes. LSTM can provide long-term memory, which is widely used in the time domain. DNN is suitable for mapping features to independent space. By combining CNN, LSTM, and DNN together in a network, we could obtain better performance than with a single network.

Baselines. We chose PGD_0 and Sparsefool as comparison methods. PGD_0 is a commonly used PGD attack method implemented under ℓ_0 norm constraints. Sparsefool is a geometry-inspired sparse attack that uses the low mean curvature of the boundary to effectively calculate the antagonistic disturbance. This method can quickly calculate the sparse disturbance and can be effectively extended to high-dimensional data.

Attack settings. For PGD_0, it was necessary to pre-set the number of interference points and calculate the successful fooling rates under such pre-set conditions. This method

does not find adversarial examples for each test sample. Therefore, we directly pre-defined a sparse number similar to that of the proposed method. Sparsefool dynamically adjusts the number of perturbed points by adjusting the hyperparameters, but because it runs until the attack is successful or the number of iterations ends, we jointly controlled the hyperparameters and the number of iterations to make the proportion of perturbed points as close as possible for fairness. For the proposed method, we could control the sparsity and the fooling rates by controlling the hyperparameters in the loss function. Next, we set the probability threshold p_s = 0.5, and the random quantization probability was set to 0.5. The hyperparameter settings were as follows: $\lambda_f = 1 \times 10^{-3}$, $\lambda_s = 1 \times 10^{-4}$, and $\lambda_e = 1 \times 10^{-4}$.

4.1. Comparison of Attack Methods

First of all, we conducted a comparison of the attack success rates among the different attack methods with ℓ_∞ norm constraints. As shown in Table 1, for a fair comparison, all attack methods were adjusted to a similar sparsity. Then, the perturbation generation time, attack fooling rate, and transferability of the various attack methods were compared. The transfer rate in the following text represents the misclassification rate of network model B for perturbed signals based on network model A.

Table 1. $\ell_\infty = 0.1$ constrained fooling rates and generation time comparison on RADIOML 2016.10a dataset. The best results are shown in bold, and all methods were adjusted to a similar sparsity. * represents white-box settings.

Source Model	Attack Method	Generation Time (s)	Sparsity (%)	VT-CNN2 (%)	CLDNN (%)	Resnet (%)
VT-CNN2	PGD_0	0.207	2.344	63.983 *	33.217	42.154
	Sparsefool ($\lambda = 3$)	0.128	1.621	99.091 *	22.672	27.520
	Sparsefool ($\lambda = 9$)	0.102	3.215	**100.00** *	28.557	30.295
	SSAA (Ours)	**0.007**	2.226	88.211 *	**72.310**	**76.198**
CLDNN	PGD_0	0.208	2.344	33.660	69.249 *	31.165
	Sparsefool ($\lambda = 3$)	0.286	2.219	32.165	99.926 *	34.487
	Sparsefool ($\lambda = 9$)	0.239	4.274	39.498	**100.00** *	35.198
	SSAA (Ours)	**0.008**	2.636	**74.192**	89.516 *	**78.684**

From the results, when $\ell_\infty = 1$ and VT-CNN2 was used as the source model, it was observed that the proposed method significantly reduced the generation time of perturbations compared to the PGD_0 and Sparsefool methods. The time taken for PGD_0 to generate adversarial examples was approximately 0.207 s, and the sp was 0.128 s, while the proposed method only took 0.007 s, which was several orders of magnitude faster than the first two methods. Secondly, compared to the PGD_0 method, the proposed method had better white-box fooling rates and better transferability. The transfer rate of the PGD_0 method was only 33.217%, while the transfer rate of the proposed method was 72.310%. Compared to the Sparsefool method, the proposed method still displayed better transferability. Note that since most normal communication environments are black-box environments, the proposed method is more suitable for practical applications. When using CLDNN as the source model, there were similar results. Specifically, in this case, the generation time of the Sparsefool method doubled, while the proposed method's generation time only increased by 0.001.

Therefore, the results indicate that a method is provided to achieve an efficient fooling rate and sparsity. Moreover, the method still possesses excellent transferability, effectively attacking black-box models.

4.2. Fooling Rate under Different Situations

In this section, we consider the comparison of the fooling rate and transfer rate in different situations.

Firstly, the impact of the signal-to-noise ratio on our proposed method is considered. A good algorithm should exhibit stable and excellent fooling and transfer performance under various signal-to-noise ratios. Therefore, the fooling rate and transfer rate under different signal-to-noise ratios was evaluated. The experiment was conducted on the RADIOML 2016.10a dataset, using VT-CNN2 as the source model, with a sparsity of 2.226% and an $\ell_\infty = 0.1$ constraint. The experimental results are shown in Table 2. It can be seen that the fooling rate decreased with the signal-to-noise ratio, but overall it showed a stable state. Under a sparsity of 2.226% and a signal-to-noise ratio of -6 dB, the fooling rate could reach 91.645%, and when transferred to CLDNN, the fooling rate was also 76.465%. When the signal-to-noise ratio increased to 18dB, the fooling rate only decreased by 4.39%, proving the stability and superiority of the proposed method in facing various signal-to-noise ratios.

Table 2. The fooling rate under different signal-to-noise ratios. The source model was VT-CNN2, and attacks were performed on the RADIOML 2016.10a dataset, with an $\ell_\infty = 0.1$ constraint.

Target Model	SNR (dB)				
	−6	**0**	**6**	**12**	**18**
VT-CNN2 (%)	91.645	89.259	88.169	87.159	87.255
CLDNN (%)	76.465	75.835	74.849	71.899	71.609

Secondly, the fooling rate and transfer rate of the proposed method under different ℓ_∞ norm values were compared and evaluated. The performance comparison of the testing algorithm under changes in the interference upper limit was conducted on the RADIOML 2016.10a dataset, with VT-CNN2 as the source model and the sparsity adjusted to similar values.

The experimental results are shown in Table 3. As the ℓ_∞ norm decreased, the fooling rate showed a decreasing trend. In the case of $\ell_\infty = 0.001$, the proposed algorithm had a fooling rate of 44% and a transfer rate of 38%, indicating that the proposed algorithm still had a certain effect even in the case of an extremely low ℓ_∞ norm.

Table 3. The fooling rate with different ℓ_∞ constraints. The source model was VT-CNN2, and attacks were performed on the RADIOML 2016.10a dataset.

Target Model	ℓ_∞		
	0.1	**0.01**	**0.001**
VT-CNN2 (%)	88.211	75.167	44.154
CLDNN (%)	72.310	66.655	38.131

4.3. Comparison of Power Spectral Density

As previously reported, adversarial perturbations can cause significant changes in the signal spectrum, thereby greatly reducing the effectiveness of deception. It is expected that the spectrum of adversarial signals will be as similar to that of the original signal as possible, in order to avoid being detected and increase attack effectiveness. Therefore, spectral loss was introduced in this work to reduce this spectral variation. Next, this similarity was confirmed by the power spectral density (PSD) of the original signal and the adversarial signal. Note that the spectral loss also had an effect in the phase diagram. Only PSD is shown, because PSD can better indicate the effect of the proposed algorithm.

The comparative experiment was conducted on the RADIOML 2016.10a dataset, using VT-CNN2 as the source model, with sparsity adjusted to similar values, an $\ell_\infty = 0.1$ constraint, a signal-to-noise ratio of 10, and the BPSK modulation method. The PSDs of the various methods are shown in Figure 2. From the results, it can be seen that the adversarial

signals generated by the PGD_0 method and Sparsefool method presented significant differences in the spectrum from the original signal, resulting in the PSD being more serrated in the central lobe and showing significant differences in the sidelobe content. For our proposed method, there was almost no difference in the main lobe content compared to the original signal, and there was only a slight difference in the side lobe content compared to the original signal. The reason for this was that the PGD_0 and Sparsefool methods did not consider the issue of spectral differences in the process of generating interference, only pursuing a lower sparsity and higher fooling rate. In practical applications, this would greatly increase the likelihood of the attack being detected and defended against. For our proposed method, by introducing spectral loss and controlling the disturbance in the frequency domain to make the disturbance appear more in the lobes, we helped to improve the robustness of the attack to filtering and detection operations.

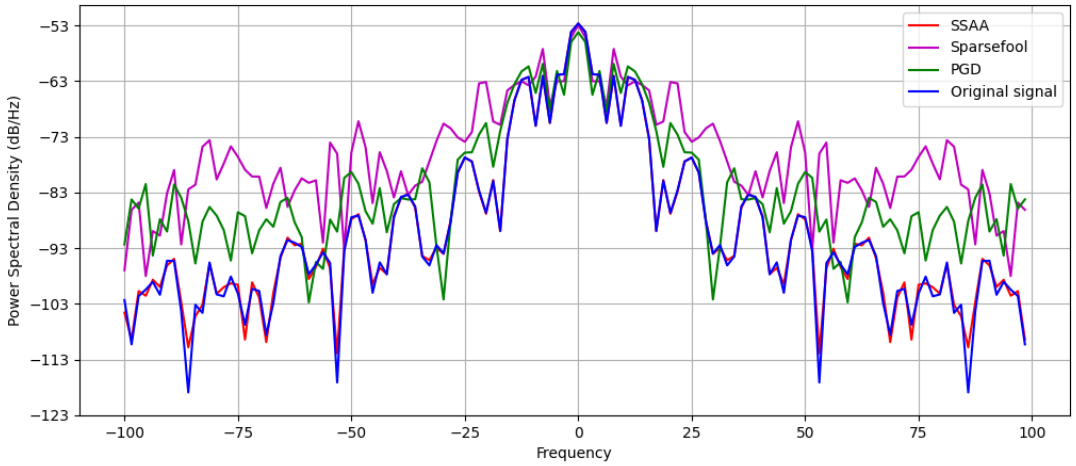

Figure 2. Comparison of the PSD of the adversarial signal generated by PGD_0, Sparsefool, and SSAA with the PSD of the original signal.

4.4. Ablation Study

In a further investigation, the contribution of key factors (such as spectral loss, sparsity, and quantization loss) in the proposed method was considered. The experiment was conducted on the RADIOML 2016.10a dataset, using VT-CNN2 as the source model, with an $\ell_\infty = 0.1$ constraint.

The impact of spectrum loss. In the following, the impact of adding and not adding spectral loss on the PSD of the proposed algorithm was investigated. The method of not adding spectral loss involved setting λ_f to 0. The results are shown in Figure 3. The experimental sparsity in the figure was 2.226%, and the modulation method was BPSK. When spectral loss was not used, although in very sparse situations, the PSD of the adversarial signal and the original signal also differed greatly. For the PSD when using spectral loss, it can be observed from the results that spectral loss guided the training of the model to maintain the spectral shape of the original signal.

The impact of sparsity. Then, the results under three different sparsities were explored while keeping the other parameters unchanged: $\lambda_s = 1 \times 10^{-4}$, $\lambda_s = 1 \times 10^{-5}$, and $\lambda_s = 1 \times 10^{-6}$. The results are shown in Table 4, indicating that the proposed algorithm could dynamically adjust the sparsity of anti-interference by adjusting λ_s. As λ_s decreased, the sparsity, fooling rate, and transfer rate all increased. At $\lambda_s = 1 \times 10^{-6}$, the fooling rate reached 99.415%. This indicates that even in the case of limited modification points, modifying the points affected the fooling rate and transfer rate, and the proposed algorithm could achieve an excellent fooling rate and transfer rate even in the case of few modification points.

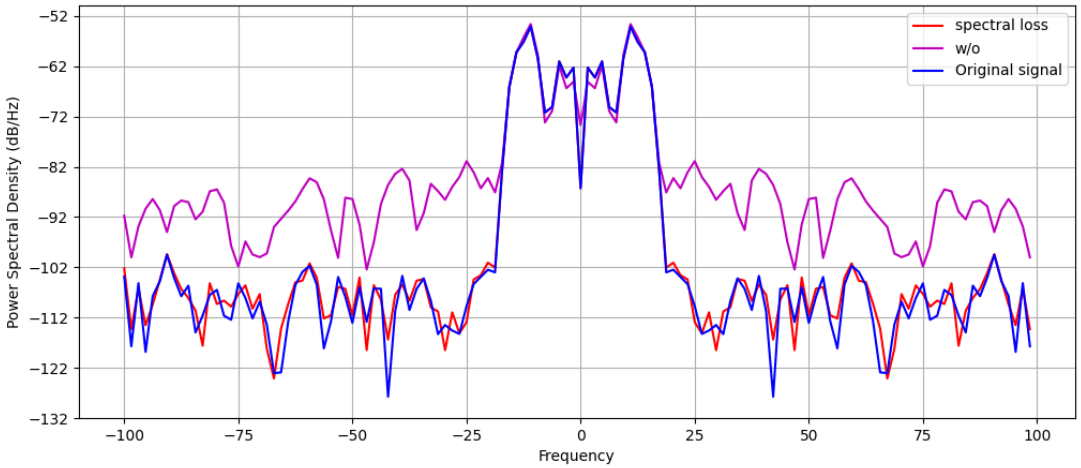

Figure 3. Comparison of the PSD of the adversarial signal generated with and without spectral loss with the PSD of the original signal.

Table 4. Comparison of fooling rates under different sparse hyperparameter settings. The source model was VT-CNN2, and attacks were performed on the RADIOML 2016.10a dataset, with an $\ell_\infty = 0.1$ constraint.

Method	Sparsity (%)	VT-CNN2 (%)	CLDNN (%)
SSAA ($\lambda_s = 1 \times 10^{-4}$)	2.226	88.211	72.310
SSAA ($\lambda_s = 1 \times 10^{-5}$)	4.687	94.158	84.441
SSAA ($\lambda_s = 1 \times 10^{-6}$)	7.427	99.415	90.246

The impact of quantization loss. Finally, the impact of quantization losses on the proposed method was investigated. We considered the impact of both quantization and non-quantization losses on the test results in the VTCNN2 and CLDNN target models. As shown in the results in Table 5, the proposed method could attack in a low-sparsity form without using quantization loss, but the fooling rate and transfer rate in the test results were not as good as in the case where quantization loss was used. When the model was VTCNN2, without using quantization loss, the fooling rate decreased by 13.725%, with an increase in sparsity. The above results demonstrate the effectiveness of quantization loss.

Table 5. Comparison of fooling rates with and without spectral loss on the RADIOML 2016.10a dataset, with an $\ell_\infty = 0.1$ constraint.

Method	Model	Sparsity (%)	Fooling Rate (%)
quantization loss	VT-CNN2	2.226	88.211
w/o quantization loss	VT-CNN2	2.756	74.486
quantization loss	CLDNN	2.636	89.516
w/o quantization loss	CLDNN	2.815	72.310

5. Conclusions

In this paper, a framework was proposed for generating adversarial signals with sparse and spectral similarity, taking into account the concealment of adversarial signals and the changes in the signal spectrum caused by adversarial perturbations in scenarios where adversarial attacks are used to prevent eavesdropping. The framework can generate samples with better fooling and transfer rates than existing methods, and unlike other sparse attack methods, the difference between the spectrum of the generated adversarial signal and the spectrum of the original signal is very small. Specifically, the proposed method decouples the disturbance into position masks and amplitude vectors and then

optimizes the two terms using a generator architecture, introducing a spectral loss to reduce the spectral difference from the original signal during the training process.

First of all, the current proposed method still has shortcomings in terms of generation ability, and its convergence ability needs to be improved. In future work, better generator architectures will be considered, such as adversarial generation networks and diffusion models. Secondly, using only the mean square error to calculate the similarity between spectra may be biased. In the next plan, different functions, such as the Huber function and Fréchet distance, can be evaluated to calculate this similarity. In addition, the impact of adversarial perturbations in communication channels should be considered, which may affect the deception ability of adversarial examples.

Author Contributions: Conceptualization, Z.J. and W.Z.; methodology, Z.J.; software, Z.J. and S.Y.; validation, S.Y., C.H. and L.L.; formal analysis, Z.J. and L.L.; investigation, Z.J. and P.F.; resources, W.Z. and P.C.; data curation, X.Z.; writing—original draft preparation, Z.J.; writing—review and editing, W.Z. and X.Z.; visualization, P.F. and C.H.; supervision, W.Z. and P.C.; project administration, Z.J.; funding acquisition, W.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China under grant No. 62001515.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Huang, Z.; Yang, J.; Wang, X.; Cui, X.; Wang, F. Survey of modulation recognition algorithms in non-cooperative communication. *Sci. Technol. Rev.* **2019**, *37*, 55–62.
2. Zebarjadi, M.; Teimouri, M. Non-cooperative burst detection and synchronisation in downlink TDMA-based wireless communication networks. *IET Commun.* **2019**, *13*, 863–872. [CrossRef]
3. O'Shea, T.J.; Roy, T.; Clancy, T.C. Over-the-air deep learning based radio signal classification. *IEEE J. Sel. Top. Signal Process.* **2018**, *12*, 168–179. [CrossRef]
4. O'shea, T.; Hoydis, J. An introduction to deep learning for the physical layer. *IEEE Trans. Cogn. Commun. Netw.* **2017**, *3*, 563–575. [CrossRef]
5. Shi, Q.; Karasawa, Y. Automatic modulation identification based on the probability density function of signal phase. *IEEE Trans. Commun.* **2012**, *60*, 1033–1044. [CrossRef]
6. Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; Fergus, R. Intriguing properties of neural networks. *arXiv* **2013**, arXiv:1312.6199.
7. Hameed, M.Z.; György, A.; Gündüz, D. The best defense is a good offense: Adversarial attacks to avoid modulation detection. *IEEE Trans. Inf. Forensics Secur.* **2020**, *16*, 1074–1087. [CrossRef]
8. Ma, H.; Yang, S.; He, G.; Wu, R.; Hao, X.; Li, T.; Feng, Z. Faking Signals to Fool Deep Neural Networks in AMC via Few Data Points. *IEEE Access* **2021**, *9*, 124425–124433. [CrossRef]
9. Flowers, B.; Buehrer, R.M.; Headley, W.C. Communications aware adversarial residual networks for over the air evasion attacks. In Proceedings of the MILCOM 2019—2019 IEEE Military Communications Conference (MILCOM), Norfolk, VA, USA, 12–14 November 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 133–140.
10. Wang, C.; Wei, X.; Fan, J.; Hu, Y.; Yu, L. Universal Attack against Automatic Modulation Classification DNNs Under Frequency and Data Constraints. *IEEE Internet Things J.* **2023**, *10*, 12938–12950. [CrossRef]
11. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. *arXiv* **2014**, arXiv:1412.6572.
12. Carlini, N.; Wagner, D. Towards evaluating the robustness of neural networks. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–25 May 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 39–57.
13. Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv* **2017**, arXiv:1706.06083.
14. Moosavi-Dezfooli, S.M.; Fawzi, A.; Frossard, P. Deepfool: A simple and accurate method to fool deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 21–26 July 2016; pp. 2574–2582.
15. Moosavi-Dezfooli, S.M.; Fawzi, A.; Fawzi, O.; Frossard, P. Universal adversarial perturbations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1765–1773.
16. Baluja, S.; Fischer, I. Adversarial transformation networks: Learning to generate adversarial examples. *arXiv* **2017**, arXiv:1703.09387.

17. Poursaeed, O.; Katsman, I.; Gao, B.; Belongie, S. Generative adversarial perturbations. In Proceedings of the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4422–4431.
18. Xiao, C.; Li, B.; Zhu, J.Y.; He, W.; Liu, M.; Song, D. Generating adversarial examples with adversarial networks. *arXiv* **2018**, arXiv:1801.02610.
19. Song, Y.; Shu, R.; Kushman, N.; Ermon, S. Constructing unrestricted adversarial examples with generative models. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 8312–8323.
20. Su, J.; Vargas, D.V.; Sakurai, K. One pixel attack for fooling deep neural networks. *IEEE Trans. Evol. Comput.* **2019**, *23*, 828–841. [CrossRef]
21. Papernot, N.; McDaniel, P.; Jha, S.; Fredrikson, M.; Celik, Z.B.; Swami, A. The limitations of deep learning in adversarial settings. In Proceedings of the 2016 IEEE European Symposium on Security and Privacy (EuroS&P), Saarbrucken, Germany, 21–24 March 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 372–387.
22. Croce, F.; Hein, M. Sparse and imperceivable adversarial attacks. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 4724–4732.
23. Xie, H.; Tan, J.; Zhang, X.; Ji, N.; Liao, H.; Yu, Z.; Xiang, X.; Liu, N. Low-Interception Waveform: To Prevent the Recognition of Spectrum Waveform Modulation via Adversarial Examples. In Proceedings of the 2021 XXXIVth General Assembly and Scientific Symposium of the International Union of Radio Science (URSI GASS), Rome, Italy, 28 August–4 September 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–4.
24. Modas, A.; Moosavi-Dezfooli, S.M.; Frossard, P. Sparsefool: A few pixels make a big difference. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 9087–9096.
25. He, Z.; Wang, W.; Dong, J.; Tan, T. Transferable sparse adversarial attack. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 14963–14972.
26. O'shea, T.J.; West, N. Radio machine learning dataset generation with gnu radio. In Proceedings of the GNU Radio Conference, Boulder, CO, USA, 12–16 September 2016; Volume 1.
27. O'Shea, T.J.; Corgan, J.; Clancy, T.C. Convolutional radio modulation recognition networks. In Proceedings of the Engineering Applications of Neural Networks: 17th International Conference, EANN 2016, Aberdeen, UK, 2–5 September 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 213–226.
28. West, N.E.; O'shea, T. Deep architectures for modulation recognition. In Proceedings of the 2017 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN), Baltimore, MD, USA, 6–9 March 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–6.
29. Liu, X.; Yang, D.; El Gamal, A. Deep neural network architectures for modulation classification. In Proceedings of the 2017 51st Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, USA, 29 October–1 November 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 915–919.