Project Gradient Descent Adversarial Attack against Multisource Remote Sensing Image Scene Classification

Yan Jiang, Guisheng Yin, Ye Yuan, and Qingan Da

College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China

Correspondence should be addressed to Guisheng Yin; yinguisheng@hrbeu.edu.cn

Received 27 December 2020; Revised 7 February 2021; Accepted 3 June 2021; Published 12 June 2021

Deep learning technology (a deeper and optimized network structure) and remote sensing imaging (i.e., the more multisource and the more multicategory remote sensing data) have developed rapidly. Although the deep convolutional neural network (CNN) has achieved state-of-the-art performance on remote sensing image (RSI) scene classification, the existence of adversarial attacks poses a potential security threat to the RSI scene classification task based on CNN. The corresponding adversarial samples can be generated by adding a small perturbation to the original images. Feeding the CNN-based classifier with the adversarial samples leads to the classifier misclassify with high confidence. To achieve a higher attack success rate against scene classification based on CNN, we introduce the projected gradient descent method to generate adversarial remote sensing images. Then, we select several mainstream CNN-based classifiers as the attacked models to demonstrate the effectiveness of our method. The experimental results show that our proposed method can dramatically reduce the classification accuracy under untargeted and targeted attacks. Furthermore, we also evaluate the quality of the generated adversarial images by visual and quantitative comparisons. The results show that our method can generate the imperceptible adversarial samples and has a stronger attack ability for the RSI scene classification.

1. Introduction

With the high-speed development of remote sensing technology, we can obtain more multisource and multicategory remote sensing images (RSIs). Those images are more efficiently employed to observe ground objects. RSI encompasses more the abundant spectral features of the ground objects, i.e., the texture and geometric features. Therefore, employing RSI to achieve scene classification has also gathered considerable attention. The purpose of scene classification is to predict the label for a given RSI, e.g., airport, forest, and river. The critical process of RSI scene classification is to extract image features. The existing researches can be mainly divided into two strategies according to the features used: (1) the low-level features and the middle-level global features obtained by the manual extraction and (2) the high-level features extracted automatically by the convolutional neural network (CNN).

In early RSI scene classification research, scene classification mainly relies on handcrafted low-level and middle-level features [1–5], e.g., color, texture, spatial structure, and scale-invariant feature transform [6]. The middle-level features’ extracting process can be regarded as the high-order statistical of the low-level local features [7–9]. Compared with the low-level features, the middle-level features (e.g., bag-of-visual-words model [10]) have expressive content for scene images. However, due to the lack of flexibility and adaptability, using the middle-level features cannot easily distinguish the complex scenes and gain the ideal classification accuracy. Traditional methods can express RSI features to a certain extent [11]. Nevertheless, the above features are manually extracted, and end-to-end scene classification cannot be achieved. The final classification accuracy is mainly subject to the human experience.

To ameliorate the limitations of handcrafted features, more researchers adopted the automatic extracting of RSI features. In recent years, deep learning has achieved
impressive performance in image processing tasks, e.g., image generation, object detection, and image classification. Moreover, the CNN-based classifier has also been employed for RSI scene classification. With an end-to-end framework, CNN performs multilayer nonlinear transformations to extract the high-level image features automatically. These features focus on the overall semantics of RSI rather than the pixels of a local region. Hence, the high-level features obtained by CNN can be more effective for RSI scene classification. For instance, Wang et al. [12] proposed a method based on Residual Network (ResNet) to achieve RSI classification. It breaks the limitation of lacking training samples. Zhang et al. [13] proposed an effective architecture based on CNN and Capsule Network (CapsNet), named CNN-CapsNet. RSI scene classification has a common problem, i.e., the overfitting caused by the deeper network (too many parameters). However, the shallower network has not enough extraction ability of semantic information. To this end, Zhang et al. [14] proposed a simple and efficient network based on the dense convolutional network (DenseNet). DenseNet can improve network performance without increasing the number of network parameters.

Although the above CNN-based methods have an outstanding performance for RSI scene classification, they also face various security risks [15], e.g., data poisoning attacks [16] and adversarial sample attacks. To defend against these attacks, researchers also proposed corresponding malicious detection methods [17–19]. Among these attacks, the classifiers can easily be fooled by the designed adversarial samples and get unexpected results. This phenomenon has raised serious concerns, especially for the security of applications based on deep learning [20–22]. The vulnerability of deep learning is first revealed in [23]. By adding adversarial perturbations to original images, we can obtain deceptive images imperceptible to human eyes. Subsequently, numerous attack methods were proposed. Goodfellow proposed a hypothesis that the existence of adversarial examples may be due to deep model linear property. Based on this hypothesis, they proposed a fast gradient sign method (FGSM) [24], which can modify the pixel values of images. Hence, the modified images can obtain the misclassify result. Later, an iterative version of FGSM, the basic iterative method (BIM) [25], was proposed. BIM can achieve an effective attack by using the generated more imperceptible adversarial perturbation. Then, projected gradient descent (PGD) [26] attack was proposed, which is seen as a variant of BIM. Unlike BIM, PGD adds a step of initializing the uniform random perturbation. Then, it can run more iterations until finding an adversarial example. Also, it replaces the clip operation of BIM on the gradient with the gradient projection [27]. Due to the above improvements, the attack capability of PGD is far superior to both FGSM and BIM. Moreover, PGD is known as the strongest first-order attack. Another strong attack is the Carlini and Wagner (C & W) [28] attack. C & W proposed a family of three attacks that minimize diverse similarity metrics: \( L_0, L_2, \) and \( L_{\infty} \), respectively. C & W achieves the effect that is imperceptible by humans, and they constrain the \( L_0, L_2, \) and \( L_{\infty} \) norm to make the perturbation small.

Although C & W can generate more precise perturbations, the computational efficiency of C & W is slower. By contrast, PGD is a commonly used attack method that is stable and efficient at present.

The research on the adversarial samples for RSI scene classification is not thoroughly studied yet. Czaja et al. [29] provided a preliminary analysis of the adversarial examples for remote sensing data. However, their analysis is only under the targeted attack. Xu et al. [30] used FGSM and L-BFGS [23] attacks to generate the adversarial samples for RSI scene classification. They tested these above two attacks under the targeted and the untargeted attacks. We call it Xu2020 in this paper. Additionally, they utilized an adversarial training strategy to increase the resistibility of deep models that faced adversarial examples. Burnel et al. [31] proposed a different approach based on a modified Wasserstein generative adversarial network (GAN) [32] to generate natural adversarial hyperspectral images. Chen et al. [33] tested two adversarial attack algorithms (i.e., FGSM and BIM) on two classifiers trained on different remote sensing data sets. They found that a higher similarity in the feature space is more straightforward for generating the corresponding adversarial examples. Although the above methods can easily fool the classifier, these attack methods could not represent a state-of-the-art performance according to the attack capacity. FGSM attack has the limitation of its one-step characteristics. Hence, it has a low success rate on large public data sets. Unlike FGSM, BIM has lower iteration numbers than PGD, so its attack effect lags behind PGD.

Based on the above analysis, we utilize the PGD attack to generate adversarial samples. Then, we test the generated images on eight state-of-the-art classifiers to prove our method’s effectiveness in this paper. The main contributions are summarized as follows:

1. We apply a first-order attack (i.e., PGD) to generate the adversarial samples. This method can generate more accurate perturbation by uniform random perturbation as initialization, then running several BIM iterations to find an adversarial example.

2. To demonstrate our method’s effectiveness, we employ three benchmark multiclass remote sensing scene data sets imaged by different sensors. Also, eight CNN-based classifiers are attacked under the untargeted and the targeted settings. Compared with the Xu2020 method, our method can make the attacked classifiers much lower classification accuracy. This phenomenon has revealed that the current state-of-the-art classifiers still have potential security risks in the multisource RSI scene classification task.

3. Furthermore, we also provide visual comparisons of the generated perturbations and the adversarial samples. For the more objective visual quality evaluation, the peak signal-to-noise ratio (PSNR) is employed as the evaluation index. The results indicate that our method can generate imperceptible adversarial samples and has a stronger attack ability for current state-of-the-art classifiers.
The rest of this paper is organized as follows. Section 2 introduces the main strategies for generating adversarial examples and the detailed expression of the PGD attack under untargeted and targeted settings. Section 3 shows three RSI scene data sets used in this paper and the corresponding experimental results. Conclusions are summarized in Section 4.

2. Method

Adversarial attacks principally followed these two strategies. (1) We maximize the classifier’s loss function under the $L_\infty$ norm constraint by the small constant value. This operation will cause the classifier to misclassify. (2) Using the $L_\infty$ norm constraint by the small constant value. This operation is an efficient one-step attack. This attack used an $\epsilon$-bounded constraint and the sign function to get its perturbation set. $\mathcal{D}$ is the cross-entropy loss. Usually, $L_\infty$ and $L_2$ norms are currently used as the constraint that bound the perturbation to be small. In this paper, [34] mentioned that $L_\infty$ is the optimal distance metric for the image classification task, and [35] argues that the distillation is secure under this distance metric. Thus, we choose PGD-$L_\infty$ as our attack method (Algorithm 1).

Generally, adversarial attacks can fool the CNN classifier under untargeted and targeted settings. The targeted attack means that the attacker assigns the classification result to a specific label. Unlike the targeted attack, the untargeted attack aims to make the classifier mispredict the given image’s label but does not necessarily specify the prediction into a specific label. The targeted and the untargeted settings of our attack method can be formulated as

$$\text{target: } x_{\text{adv}}^{i+1} = \prod_{x \in S} \left[ x_{\text{adv}}^i + \alpha \text{ sign} (\nabla_x \mathcal{L} (x, y, \theta)) \right],$$

$$\text{untarget: } x_{\text{adv}}^{i+1} = \prod_{x \in S} \left[ x_{\text{adv}}^i + \alpha \text{ sign} (\nabla_x \mathcal{L} (x, y, \theta)) \right].$$

3. Experiments and Results

3.1. Experimental Setup. We tested the Xu2020 method and our method on three RSI scene classification data sets, i.e., UC Merced (UCM) data set [10], WHU-RS19 data set [36], and AID data set [37]. Figures 2–4 show the examples of the three data sets.

UCM data set includes 2,100 remote sensing scene images. This data set consists of 21 scene classes, and each class contains 100 samples with a size of 256 $\times$ 256 pixels. All images were collected from the National Map Urban Area Imagery collection, including multiple towns across the United States. The 21 land-use classes are agricultural, airplane, baseball diamond, beach, buildings, chaparral, dense residential, forest, freeway, golf course, harbor, intersection, medium-density residential, mobile home park, overpass, parking lot, river, runway, sparse residential, storage tanks, and tennis courts. The WHU-RS19 data set was collected from Google Earth (Google Inc.). The WHU-RS19 has 19 different scene classes, and each class contains 50 images with the size of 600 $\times$ 600 pixels. The 19 scene classes are airport, beach, bridge, commercial area, desert, farmland, football field, forest, industrial area, meadow, mountain, park, parking lot, pond, port, railway station, residential area, river, and viaduct. AID data set is the biggest data set among the three data sets, containing 30 scene classes. Each class has around 220–420 images. This data set includes...
Figure 1: The overall procedure of our method.

Input:
(1) A set of test remote sensing images $X = \{x_1, \ldots, x_n\}$ and the corresponding labels $Y = \{y_1, \ldots, y_n\}$.
(2) A classifier $f$ with parameter $\theta$.
(3) Number of epochs $t$; a small constant that restricts the perturbation parameter $\alpha$;

Output: Adversarial example, $x_{adv}$;
(1) Let $x_{adv} = x_{adv}^1, t = 0$,
(2) while iter < $t$ do
(3) Compute $\mathcal{L} = \sum_{i=1}^n \mathcal{L}(x, y, \theta)$
(4) Compute $x_{adv}^{i+1} = x_{adv}^i + t \times L$
(5) Compute perturbed = $x_{adv}^i - x_{i+1}$
(6) Compute perturbed = clip(perturbed, $-\epsilon, \epsilon$)
(7) $t + 1$
(8) end while

Return $x_{adv}$

Algorithm 1: Adversarial attack strategy.
10,000 remote sensing scene images. The 30 scene classes contain airport, bare land, baseball field, beach, bridge, center, church, commercial, dense residential, desert, farmland, forest, industrial, meadow, medium residential, mountain, park, parking, playground, pond, port, railway station, resort, river, school, sparse residential, square, stadium, storage tanks, and viaduct.

To prove the strong attack ability of our method, we conducted the experiments in three parts. Firstly, we compared the classification accuracies of the Xu2020 method (used in [30]) with those of our method. Both attacks were tested in the untargeted and targeted settings. Then, we also showed the classification accuracy specific to each class under the targeted setting. Finally, we visualized the generated adversarial images and the corresponding perturbations. For the more objective evaluation, we used the peak signal-to-noise ratio (PSNR) index to analyze the generated images of the Xu2020 and our method.

In this paper, we first randomly picked 20% of labeled images as the training set. The rest images are employed as the test set. All experiments were randomly repeated three times. In our experiment, we used Xu2020 as the compared method. Eight state-of-the-art CNN-based classifiers were employed to test our method. These classifiers were VGG-16 [38], GoogLeNet [39], InceptionV3 [40], ResNet-18 [41], ResNet-50 [41], ResNet-101 [41], DenseNet-121 [42], and DenseNet-201 [42], respectively. They were pretrained on the ImageNet [43] data set. Adam was used as the optimizer to train the classifiers with a batch size of 32. The training epochs were set as 40 with a learning rate initialized to $5 \times 10^{-5}$ at the first 20 epochs and then the learning rate decay to $1 \times 10^{-5}$ at the last 20 epochs. The experimental environment configuration is shown in Table 1.

### 3.2. Experimental Results of Untargeted Attack

In this section, we show the classification accuracies of eight different classifiers before and after the attack. The overall accuracy (OA, i.e., the number of accurately classified images divided by the number of entire test sets) is employed to evaluate different methods quantitatively. OA gap is the gap between the classification accuracy of the clean test set and that of the Xu2020 method and our method test set. For the untargeted and targeted attacks, the parameter $\varepsilon$ was set to be 0.01 for the Xu2020 method and our method. Moreover, because Xu2020 is a one-step attack, we set the number of steps to 1. For our method, the number of steps was set to be 40.

Tables 2–4 show the untargeted attack results against three types of input (clean, perturbed by Xu2020, perturbed by our method ran for 40 steps).

As shown in Table 2, we can see that all classifiers have excellent performance on three data sets before the attack. Generally, the OA values of eight classifiers are almost around 90%. Among these different classifiers, the classifiers with more complicated layers have higher accuracy than the classifiers with fewer layers. For instance, ResNet-101 has gained about 5% advancement in terms of OA than VGG-16 on the UCM data set. However, the OA of different classifiers all decreased dramatically after the attack.

As shown in Table 4, for the biggest data set (i.e., AID data set), the OA values of VGG-16 and DenseNet-201 are 20.97% and 53.88% after being attacked by Xu2020.
Figure 4: Examples of AID data set. (a) Airport. (b) Bare land. (c) Baseball field. (d) Beach. (e) Bridge. (f) Center. (g) Church. (h) Commercial. (i) Dense residential. (j) Desert. (k) Farmland. (l) Forest. (m) Industrial. (n) Meadow. (o) Medium residential. (p) Mountain. (q) Park. (r) Parking. (s) Playground. (t) Pond. (u) Port. (v) Railway station. (w) Resort (x) River. (y) School. (z) Sparse residential. (a1) Square. (b1) Stadium. (c1) Storage tanks. (d1) Viaduct.

Table 1: Experimental environment configuration.

| Designation       | Information                                      |
|-------------------|--------------------------------------------------|
| Operating system  | Ubuntu 18.04 LTS                                 |
|                   | CPU: Intel Core i7-9700 K 3.60 GHz, 32 GB RAM    |
| System configuration | NVIDIA GeForce RTX 2080Ti 11G                     |
| Python library    | Cuda 10.1                                        |
|                   | Pytorch 1.3.1                                    |
|                   | Torchvision 0.4.2                                |
|                   | Numpy 1.19.2                                     |
|                   | Opencv-python 4.4.0                              |
|                   | Advertorch 0.2.3                                 |

Table 2: Classification accuracy of the UCM data set under the untargeted attack.

| Models       | OA of clean | OA of Xu2020 | OA of ours | OA gap of Xu2020 | OA gap of ours |
|--------------|-------------|--------------|------------|------------------|---------------|
| VGG-16       | 91.15 ± 1.03| 24.46 ± 0.95 | 4.84 ± 0.73| −66.69           | −86.31         |
| GoogLeNet    | 95.52 ± 0.62| 37.32 ± 1.73 | 5.93 ± 0.75| −58.19           | −89.58         |
| InceptionV3  | 94.78 ± 0.32| 57.24 ± 1.03 | 10.00 ± 1.25| −37.54           | −84.78         |
| ResNet-18    | 94.80 ± 0.22| 31.05 ± 1.33 | 1.92 ± 1.14| −63.75           | −92.88         |
| ResNet-50    | 96.27 ± 0.10| 51.29 ± 1.27 | 2.60 ± 1.21| −44.98           | −93.67         |
| ResNet-101   | 96.15 ± 0.26| 50.22 ± 4.27 | 2.66 ± 0.62| −45.93           | −93.49         |
| DenseNet-121 | 95.58 ± 0.67| 50.08 ± 1.53 | 5.73 ± 1.51| −45.50           | −89.84         |
| DenseNet-201 | 95.91 ± 0.73| 51.62 ± 0.68 | 5.65 ± 1.55| −44.29           | −90.26         |
| Models    | OA of clean | OA of Xu2020 | OA of ours | OA gap of Xu2020 | OA gap of ours |
|-----------|-------------|--------------|------------|------------------|----------------|
| VGG-16    | 91.03 ± 0.85| 17.09 ± 2.79 | 0.24 ± 0.24| −73.94           | −90.79         |
| GoogLeNet | 93.25 ± 1.54| 31.52 ± 0.73 | 2.06 ± 0.48| −61.74           | −91.19         |
| InceptionV3| 91.72 ± 0.57| 45.90 ± 1.29 | 3.69 ± 0.88| −45.82           | −88.03         |
| ResNet-18 | 91.52 ± 1.21| 22.42 ± 2.79 | 1.66 ± 0.44| −69.09           | −89.86         |
| ResNet-50 | 93.90 ± 0.44| 41.05 ± 2.75 | 2.02 ± 1.41| −52.85           | −91.88         |
| ResNet-101| 92.14 ± 0.87| 42.95 ± 0.32 | 2.06 ± 0.00| −49.19           | −90.08         |
| DenseNet-121| 93.05 ± 0.69| 43.60 ± 1.62 | 3.64 ± 0.97| −49.45           | −89.41         |
| DenseNet-201| 94.26 ± 0.53| 51.52 ± 3.76 | 4.81 ± 1.25| −42.75           | −89.45         |

| Models    | OA of clean | OA of Xu2020 | OA of ours | OA gap of Xu2020 | OA gap of ours |
|-----------|-------------|--------------|------------|------------------|----------------|
| VGG-16    | 90.43 ± 0.27| 20.97 ± 0.65 | 1.31 ± 0.74| −69.46           | −89.12         |
| GoogLeNet | 91.97 ± 0.28| 28.92 ± 0.44 | 0.42 ± 0.16| −63.05           | −91.55         |
| InceptionV3| 92.26 ± 0.16| 46.80 ± 0.44 | 4.49 ± 0.47| −45.46           | −87.77         |
| ResNet-18 | 92.22 ± 0.31| 16.32 ± 1.13 | 0.15 ± 0.04| −75.90           | −92.07         |
| ResNet-50 | 93.54 ± 0.14| 33.99 ± 1.08 | 0.23 ± 0.19| −59.55           | −93.31         |
| ResNet-101| 93.80 ± 0.21| 36.47 ± 1.93 | 0.34 ± 0.15| −57.34           | −93.46         |
| DenseNet-121| 93.66 ± 0.15| 42.93 ± 2.65 | 0.97 ± 0.12| −50.74           | −92.69         |
| DenseNet-201| 94.14 ± 0.17| 53.88 ± 1.19 | 2.89 ± 0.80| −40.25           | −91.25         |

| Models    | OA of clean | OA of Xu2020 | OA of ours | OA gap of Xu2020 | OA gap of ours |
|-----------|-------------|--------------|------------|------------------|----------------|
| VGG-16    | 91.15 ± 0.10| 38.04 ± 0.38 | 29.98 ± 0.42| −53.11           | −61.17         |
| GoogLeNet | 95.52 ± 0.62| 51.21 ± 0.45 | 24.48 ± 0.97| −44.31           | −71.04         |
| InceptionV3| 94.78 ± 0.32| 66.09 ± 1.29 | 37.69 ± 1.95| −28.70           | −57.10         |
| ResNet-18 | 94.80 ± 0.22| 42.89 ± 0.67 | 16.52 ± 0.70| −51.91           | −78.28         |
| ResNet-50 | 96.27 ± 0.10| 60.84 ± 0.70 | 21.38 ± 0.19| −35.43           | −74.89         |
| ResNet-101| 96.15 ± 0.26| 60.93 ± 4.97 | 24.70 ± 2.02| −35.22           | −71.45         |
| DenseNet-121| 95.58 ± 0.67| 63.61 ± 1.23 | 26.14 ± 0.86| −31.97           | −69.43         |
| DenseNet-201| 95.91 ± 0.73| 64.17 ± 1.07 | 26.10 ± 0.63| −31.75           | −69.81         |

| Models    | OA of clean | OA of Xu2020 | OA of ours | OA gap of Xu2020 | OA gap of ours |
|-----------|-------------|--------------|------------|------------------|----------------|
| VGG-16    | 91.03 ± 0.85| 30.83 ± 2.30 | 26.61 ± 1.74| −60.20           | −64.42         |
| GoogLeNet | 93.25 ± 1.54| 47.80 ± 1.42 | 22.49 ± 0.76| −45.45           | −70.76         |
| InceptionV3| 91.72 ± 0.57| 57.02 ± 1.02 | 31.45 ± 0.75| −34.70           | −60.27         |
| ResNet-18 | 91.52 ± 1.21| 37.46 ± 1.65 | 15.91 ± 1.14| −54.05           | −75.60         |
| ResNet-50 | 93.90 ± 0.44| 51.64 ± 3.20 | 24.42 ± 2.19| −42.25           | −69.48         |
| ResNet-101| 92.14 ± 0.87| 54.72 ± 1.02 | 27.58 ± 1.65| −37.43           | −64.56         |
| DenseNet-121| 93.05 ± 0.69| 59.63 ± 1.66 | 28.05 ± 0.24| −33.42           | −65.00         |
| DenseNet-201| 94.26 ± 0.53| 65.18 ± 2.50 | 32.53 ± 0.64| −29.08           | −61.73         |

| Models    | OA of clean | OA of Xu2020 | OA of ours | OA gap of Xu2020 | OA gap of ours |
|-----------|-------------|--------------|------------|------------------|----------------|
| VGG-16    | 90.43 ± 0.27| 25.25 ± 0.41 | 22.55 ± 1.95| −65.18           | −67.88         |
| GoogLeNet | 91.97 ± 0.28| 44.08 ± 0.41 | 19.28 ± 0.74| −47.89           | −72.69         |
| InceptionV3| 92.26 ± 0.16| 63.56 ± 0.18 | 31.81 ± 0.49| −28.70           | −60.45         |
| ResNet-18 | 92.22 ± 0.31| 29.12 ± 0.86 | 13.92 ± 0.38| −63.10           | −78.30         |
| ResNet-50 | 93.54 ± 0.14| 30.03 ± 0.78 | 17.15 ± 0.41| −43.51           | −76.39         |
| ResNet-101| 93.80 ± 0.21| 53.76 ± 2.67 | 18.81 ± 0.74| −40.04           | −75.00         |
| DenseNet-121| 93.66 ± 0.15| 62.80 ± 1.69 | 24.33 ± 1.45| −30.87           | −69.33         |
| DenseNet-201| 94.14 ± 0.17| 74.03 ± 1.12 | 26.92 ± 1.84| −20.11           | −67.21         |
Figure 5: Each category classification accuracy of UCM under the targeted attacks of Xu2020 and our method.

Figure 6: Each category classification accuracy of WHU-RS19 under the targeted attacks of Xu2020 and our method.
Meanwhile, they are 1.31% and 2.89% after being attacked by our method, respectively.

As shown in Tables 2–4, our method’s adversarial images have the lower OA values for the bigger dataset. Specifically, for the UCM data set (a small size data set with 21 classes), the OA value of DenseNet-201 is 5.65% after being attacked by our method. For the WHU-RS19 and the AID data sets, the OA values of DenseNet-201 are only 4.81% and 2.89% after being attacked by our method, respectively. These experimental results demonstrate the effectiveness of our method for different attacked classifiers and different data sets.

Also, we can find an interesting phenomenon from Tables 2–4. The OA gap between the clean and the adversarial test sets is highly related to the classifiers’ depth for Xu2020. This phenomenon shows that the complicated classifiers tend to have a more robust and more stable property. However, the experimental results show that our method always has a stable OA gap value no matter the classifiers’ depth. In summary, our method is not affected by the depth of the classifiers and can attack various classifiers stably.

3.3. Experimental Results of Targeted Attack. Furthermore, we also performed our method and the compared method under the targeted attack. Tables 5–7 represent the corresponding OA values.

In our experiments, we set the targeted attack categories as each category every time. The number of attacks was the number of the entire categories of the whole data set. The experimental results demonstrate that the classifiers also have vulnerability under the targeted attack. From Tables 5–7, we discover that the targeted attack’s OA values are higher than those of the untargeted attack. For instance, for the RS19 data set, compared with the untargeted attack, the OA value has reached 26.61% on VGG-16 after being attacked by our method. However, the OA value has dropped to 0.24% on the same classifier under the untargeted attack. This phenomenon indicates that the targeted attack is more complicated than the untargeted attack.

We can find the common points from Tables 5–7. The OA gap between the shallow model (i.e., VGG-16) and the deep model (i.e., DenseNet-201) has a big difference. Specifically, for the AID data set under the targeted attack, the OA value is 25.25% on VGG-16 after being attacked by Xu2020. However, OA has reached about 74.03% on DenseNet-201. A similar phenomenon also can be found under the untargeted attack. The OA gap between VGG-16 and DenseNet-201 is around 30% on VGG-16. In contrast, our method shows a stable OA gap between the shallow and deep models. Specifically, under the targeted attack, the OA value is 22.55% on VGG-16 after being attacked by our method, and the OA value is 26.92% on DenseNet-201. The difference between shallow and deep models only is 4.37%.
As shown in Tables 5–7, our method has no significant fluctuation between the OA gaps under targeted and untargeted attacks. This phenomenon proves that our method is more beneficial than Xu2020 under targeted and untargeted attacks.

To further analyze the adversarial attack’s capacity, we provided each category classification accuracy (CA) after being attacked by Xu2020 and our method in Figures 5–7. We can know that the CA values of Xu2020 are much lower than those of our method in each class. For instance, for the UCM data set, the CA value is 64.35% after being attacked by Xu2020. However, the CA value is 23.06% after being attacked by our method. As shown in Figures 5–7, the CA values are around 60% after being attacked by Xu2020 on each category in all three data sets. The gap between the highest and the lowest CA values was only 8.8% from Figure 5. However, the CA values have fluctuated after being attacked by our method. The possible reason is that the images’ feature distributions of different categories have similar distributions.

An interesting phenomenon can be found in Figures 5–7. The CA values vary dramatically when the same classifiers are attacked by the same attack method but used different data sets. For the WHU-RS19 data set, the gap between the highest and the lowest CA values is 43.03% attacked by our method. However, for the bigger data set (UCM), the gap between the highest CA and the lowest CA values was only 29.46% when using our method. This phenomenon may be because the deep model is easily overfitting by training on a small data set.

3.4. Visualization of Adversarial Samples. In this section, we visualized adversarial samples generated by Xu2020 and our method. To further compare the generated images’ quality, we also demonstrated the generated perturbations against the DenseNet-201 classifier on the UCM data set, as shown in Figure 8 (the untargeted attack) and Figure 9 (the targeted attack). These visualization results indicate that the
perturbations generated by our method are more imperceptible to human eyes compared with Xu2020. The step of Xu2020 was set to 1, and the step of our method was set to be 40. \(\varepsilon\) is set to be 0.01. Under the targeted attack, we set the target category to “airport”. The second and the fourth lines of Figures 8 and 9 represent the perturbations generated by the Xu2020 and our method, respectively. We can find that most of the perturbations generated by Xu2020 are more evident than those generated by our method, especially in the third and fifth perturbations of the second line under the targeted attack.

To evaluate visual quality more objectively, we employ the PSNR to compare the generated adversarial images, as shown in Figures 8 and 9. The PSNR is a common index, which can measure the similarity between the original clean and the adversarial images. The calculation of PSNR is given as

\[
\text{PSNR} = 10 \times \log_{10} \left( \frac{\text{MAX}_f^2}{\text{MSE}} \right),
\]

where \(\text{MAX}_f\) represents the maximum possible pixel value of the image; \(\text{MSE}\) is the mean squared error. The higher value of PSNR indicates that the adversarial image’s distribution is closer to the original clean image’s distribution. If the PSNR value is higher than 40 dB, it means that the image quality is excellent. If the PSNR value is between 30 and 40 dB, it usually indicates that the image quality is good, which means that the distortion of the generated image is detectable but acceptable. The PSNR values are provided on the top of the third and fifth lines of Figures 8 and 9.

Figure 8 presents the adversarial images and perturbations generated by Xu2020 and our method. We can see that the PSNR values of Xu2020 are slightly lower than those of our method. Hence, our method’s distribution of the adversarial images is closer to those of original clean images. This inference also can be confirmed from the generated perturbations in Figure 9. These results show that the images generated by our method have a better trade-off between the attack strength and the image quality.

In summary, our method can maintain its attack strength for the different classifiers and the models with different depths. Also, our method is more beneficial than Xu2020 under untargeted and targeted attacks. Moreover,
our method’s quality of generated adversarial images has less distortion than Xu2020.

4. Conclusions

In this paper, we discussed the adversarial sample problem on the RSI scene classification task. Although the classifiers based on CNN have a state-of-the-art performance on the classification task, they have vulnerability toward adversarial samples. This may cause security risks for the RSI scene classification application. Moreover, our method performs better than the compared method under targeted and untargeted attacks. Specifically, our method has a stronger attack strength and better attack stability. We also analyzed the PSNR values of the generated adversarial images, and the results indicated that our method’s adversarial images have better image quality.

Adversarial samples are a potential threat to the RSI scene classification task. We will further investigate the effective defense or detection strategies against the adversarial samples for RSI image classification in our future work.

Data Availability

The data used to support the findings of this study are available from the website given by corresponding papers.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was funded by the Fundamental Research Funds for the Central Universities (3072020CFQ0602, 3072020CFP0604, 3072020CFP0601, and 3072021CFP0609) and 2019 Industrial Internet Innovation and Development Engineering (KY1060020002 and KY1060020008).

References

[1] G. Cheng, J. Han, L. Guo, Z. Liu, S. Bu, and J. Ren, “Effective and efficient midlevel visual elements-oriented land-use classification using VHR remote sensing images,” IEEE Transactions on Geoscience and Remote Sensing, vol. 53, no. 8, pp. 4238–4249, 2015.
[2] G. Cheng, J. Han, L. Guo et al., “Object detection in remote sensing imagery using a discriminatively trained mixture model,” ISPRS Journal of Photogrammetry and Remote Sensing, vol. 85, pp. 32–43, 2013.
[3] E. Actual, “Remote sensing image retrieval with global morphological texture descriptors,” IEEE Transactions on Geoscience and Remote Sensing, vol. 52, no. 5, pp. 3023–3034, 2014.
[4] Y. Yang and S. Newsam, “Comparing SIFT descriptors and gabor texture features for classification of remote sensed imagery,” in Proceedings of the IEEE International Conference on Image Processing (ICIP), pp. 1852–1855, San Diego, CA, USA, October 2008.
[5] X. Huang, L. Zhang, and L. Wang, “Evaluation of morphological texture features for mangrove forest mapping and species discrimination using multispectral IKONOS imagery,” IEEE Geoscience and Remote Sensing Letters, vol. 6, no. 3, pp. 393–397, 2009.
[6] D. G. Lowe and G. David, “Distinctive image features from scale-invariant keypoints,” International Journal of Computer Vision, vol. 60, no. 2, pp. 91–110, 2004.
[7] S. Chen and Y. Tian, “Pyramid of spatial relations for scene-level land use classification,” IEEE Transactions on Geoscience and Remote Sensing, vol. 53, no. 4, pp. 1947–1957, 2015.
[8] G. Cheng, L. Guo, T. Zhao, J. Han, H. Li, and J. Fang, “Automatic landslide detection from remote-sensing imagery using a scene classification method based on BoVW and pLSA,” International Journal of Remote Sensing, vol. 34, no. 1, pp. 45–59, 2013.
[9] H. Sridharan and A. Cheriyadat, “Bag of lines (BoL) for improved aerial scene representation,” IEEE Geoscience and Remote Sensing Letters, vol. 12, no. 3, pp. 676–680, 2015.
[10] Y. Yang and S. Newsam, “Bag-of-visual-words and spatial extensions for land-use classification,” in Proceedings of the SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp. 270–279, San Jose, CA, USA, November 2010.
[11] B. Luo, S. Jiang, and L. Zhang, “Indexing of remote sensing images with different resolutions by multiple features,” IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 6, no. 4, pp. 1899–1912, 2013.
[12] M. Wang, X. Zhang, X. Niu, F. Wang, and X. Zhang, “Scene classification of high-resolution remotely sensed image based on ResNet,” Journal of Geovisualization and Spatial Analysis, vol. 3, no. 2, pp. 1–9, 2019.
[13] W. Zhang, P. Tang, and L. Zhao, “Remote sensing image scene classification using CNN-CapsNet,” Remote Sensing, vol. 11, no. 5, p. 494, 2019.
[14] J. Zhang, C. Lu, X. Li, H.-J. Kim, and J. Wang, “A full convolutional network based on DenseNet for remote sensing scene classification,” Mathematical Biosciences and Engineering, vol. 16, no. 5, pp. 3345–3367, 2019.
[15] J. Qiu, Z. Tian, C. Du, Q. Zuo, S. Su, and B. Fang, “A survey on access control in the age of internet of things,” IEEE Internet of Things Journal, vol. 7, no. 6, pp. 4682–4696, 2020.
[16] M. Li, Y. Sun, H. Lu, S. Maharjan, and Z. Tian, “Deep reinforcement learning for partially observable data poisoning attack in crowd sensing systems,” IEEE Internet of Things Journal, vol. 7, no. 7, pp. 6266–6278, 2020.
[17] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, “CorrAUC: a malicious BOT-IoT traffic detection method in IOT network using machine learning techniques,” IEEE Internet of Things Journal, vol. 8, no. 5, pp. 3242–3254, 2020.
[18] Y. Sun, Z. Tian, M. Li et al., “Honeypot identification in softwarized industrial cyber-physical systems,” IEEE Transactions on Industrial Informatics, vol. 17, no. 8, pp. 5542–5551, 2020.
[19] C. Luo, Z. Tan, G. Min et al., “A novel web attack detection system for internet of things via ensemble classification,” IEEE Transactions on Industrial Informatics, vol. 17, no. 8, pp. 5810–5818, 2020.
[20] L. Jiang, K. Qiao, R. Qin et al., “Cycle-consistent adversarial GAN: the integration of adversarial attack and defense,” Security and Communication Networks, vol. 2020, Article ID 3608173, 9 pages, 2020.
[21] F. Yu, L. Wang, X. Fang, and Y. Zhang, “The defense of adversarial example with conditional generative adversarial
networks,” Security and Communication Networks, vol. 2020, Article ID 3932584, 12 pages, 2020.

[22] W. Elmasry, A. Akbulut, and A.H. Zaim, “Deep learning approaches for predictive masquerade detection,” Security and Communication Networks, vol. 2018, Article ID 9327215, 24 pages, 2018.

[23] C. Szegedy, W. Zaremba, I. Sutskever et al., “Intriguing properties of neural networks,” in Proceedings of the International Conference on Learning Representations (ICLR), Banff, Canada, April 2014.

[24] I. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” in Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, May 2015.

[25] A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial machine learning at scale,” in Proceedings of the International Conference on Learning Representations (ICLR), Toulon, France, April 2017.

[26] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” in Proceedings of the International Conference on Learning Representations (ICLR), Vancouver, Canada, April 2018.

[27] H. Kannan, A. Kurakin, and I. Goodfellow, “Adversarial logit pairing,” in Proceedings of the International Conference on Learning Representations (ICLR), Mountain View, CA, USA, March 2018.

[28] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” in Proceedings of the IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, May 2017.

[29] W. Czaja, N. Fendley, M. Pekala, C. Ratto, and I. J. Wang, “Security and Communication Networks,” Security and Communication Networks, vol. 2020, Article ID 3932584, 12 pages, 2020.

[30] Y. Xu, B. Du, and L. Zhang, “Assessing the threat of adversarial examples on deep neural networks for remote sensing scene classification: attacks and defenses,” IEEE Transactions on Geoscience and Remote Sensing, vol. 59, no. 2, pp. 1604–1617, 2020.

[31] J. C. Burnel, K. Fatras, and N. Courtly, “Generating natural adversarial hyperspectral examples with a modified wasserstein GAN,” in Proceedings of the Computer & Electronics Security Applications Rendezvous, Rennes, France, November 2019.

[32] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in Proceedings of the International Conference on Machine Learning (ICML), pp. 214–223, Sydney, Australia, August 2017.

[33] L. Chen, G. Zhu, Q. Li, and H. Li, “Adversarial example in remote sensing image recognition,” 2019, http://arxiv.org/abs/1910.13222.

[34] D. Warde-Farley and I. Goodfellow, “Adversarial perturbations of deep neural networks,” Perturbations, Optimization, and Statistics, pp. 311–342, MIT Press, Cambridge, MA, USA, 2016.

[35] N. Papernot and P. McDaniel, “On the effectiveness of defensive distillation,” 2016, http://arxiv.org/abs/1607.05113.

[36] G. S. Xia, W. Yang, J. Delon et al., “Structural high-resolution satellite image indexing,” in Proceedings of the ISPRS Technical Commission VII Symposium on Advancing Remote Sensing Science, pp. 298–303, Vienna, Austria, July 2010.