



Article

Local Pixel Attack Based on Sensitive Pixel Location for Remote Sensing Images

Lu Liu ^{1,2}, Zixuan Xu ¹, Daqing He ², Dequan Yang ³ and Hongchen Guo ^{2,*}¹ School of Computer Science, Beijing Institute of Technology, Beijing 100081, China² School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China³ Information Technology Center, Beijing Institute of Technology, Beijing 100081, China

* Correspondence: guohongchen@bit.edu.cn

Abstract: As deep neural networks (DNNs) are widely used in the field of remote sensing image recognition, there is a model security issue that cannot be ignored. DNNs have been shown to be vulnerable to small perturbations in a large number of studies in the past, and this security risk naturally exists in remote sensing object detection models based on DNNs. The complexity of remote sensing object detection models makes it difficult to implement adversarial attacks on them, resulting in the current lack of systematic research on adversarial examples in the field of remote sensing image recognition. In order to better deal with the adversarial threats that remote sensing image recognition models may confront and to provide an effective means for evaluating the robustness of the models, this paper takes the adversarial examples for remote sensing image recognition as the research goal and systematically studies vanishing attacks against a remote sensing image object detection model. To solve the problem of difficult attack implementation on remote sensing image object detection, adversarial attack adaptation methods based on interpolation scaling and patch perturbation stacking are proposed in this paper, which realizes the adaptation of classical attack algorithms. We propose a hot restart perturbation update strategy and the joint attack of the first and second stages of the two-stage remote sensing object detection model is realized through the design of the attack loss function. For the problem of the modification cost of global pixel attack being too large, a local pixel attack algorithm based on sensitive pixel location is proposed in this paper. By searching the location of the sensitive pixels and constructing the mask of attack area, good local pixel attack effect is achieved. Experimental results show that the average pixel modification rate of the proposed attack method decreases to less than 4% and the vanishing rate can still be maintained above 80%, which effectively achieves the balance between attack effect and attack cost.

Keywords: adversarial examples; remote sensing image object detection; vanishing attack

Citation: Liu, L.; Xu, Z.; He, D.; Yang, D.; Guo, H. Local Pixel Attack Based on Sensitive Pixel Location for Remote Sensing Images. *Electronics* **2023**, *12*, 1987. <https://doi.org/10.3390/electronics12091987>

Academic Editor: Silvia Liberata Ullò

Received: 17 March 2023

Revised: 20 April 2023

Accepted: 20 April 2023

Published: 24 April 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Remote sensing images have been an important data source for natural resource investigation, disaster monitoring, and public safety management. With the increasing computational power and the growing abundance of remote sensing data, deep convolutional neural networks have been more widely used in remote sensing image applications, such as image classification tasks [1–4] and target detection tasks related to image recognition [5–9] in the field of remote sensing. However, the vulnerability of deep neural networks has been exposed in recent years by the results of studies in which they are susceptible to well-designed adversarial samples [10] by attackers. An adversarial sample is a malicious input sample carefully generated by an attacker to deceive a deep learning model using an adversarial attack algorithm. The adversarial sample can cause the model to output wrong prediction results. Adversarial samples have subsequently gained the attention of many researchers in computer vision, and various adversarial attack algorithms have been developed rapidly in the following years. In the development process of adversarial attack

techniques, Goodfellow et al. [11] proposed a fast method for generating adversarial samples, namely FGSM (Fast Gradient Sign Method), where the algorithm is optimized in the direction of the gradient sign generated by the target adversarial sample. The perturbation will always be in the direction of increasing the attack loss. Madry et al. [12] proposed the PGD algorithm, Kurakin et al. [13] proposed the iterative I-FGSM (Iterative-FGSM) method, and Dong et al. [14] proposed the MI-FGSM (Momentum-based I-FGSM) method that uses momentum to accelerate the convergence process. (Momentum-based Iterative FGSM) algorithm to accelerate the convergence process. Carlini and Wagner [15] proposed an optimization-based adversarial sample generation algorithm, C&W, whose innovation lies in defining an objective function that reduces the distance between the adversarial sample and the original sample while increasing the prediction error of the target model.

Remote sensing classification models have been proven unsafe by several researchers in the field of remote sensing image tasks [16–21]. Adversarial attacks against remote sensing image classification models were first proposed by Chen et al. [16]. Their study analyzed the adversarial sample problem in remote sensing image classification systems. The experimental results showed that remote sensing classification models are also vulnerable to adversarial sample attacks, proving the threat of adversarial samples to the security of remote sensing applications. Burnel et al. [18] generated adversarial samples with good transferability using generative adversarial networks. The experimental results in the paper have confirmed that the adversarial samples are still transferable on remote sensing classification models. The generated adversarial samples still have good attacks on black-box models. Czaja et al. [19] include in their research setup some practical factors that an attacker needs to consider when launching physical world attacks, providing the feasibility of physical world attacks. These studies make it possible to construct more realistic adversarial samples and further invisibility of adversarial samples when launching physical-world attacks. Xu et al. [20] conducted a systematic study of adversarial samples on remote sensing classification models, providing a baseline of the effectiveness of adversarial sample attacks on several common remote sensing image classification datasets. These studies only provided technical pavement for the study of adversarial samples on remote sensing image classification models but did not consider how to impose the attacks on remote sensing target detection models with more complex model structures. The studies that exist for adversarial attacks on remote sensing image target detection models include the study of adversarial patches proposed by Den et al. [21]. They disguised the adversarial patch on an aircraft object to help the aircraft escape from the YOLO V2 target detection model. The study has some limitations, firstly, the vanishing attack in the study only targets the aircraft category, and the study is weak in systematicity; in addition, the study only performed the attack on the more easily breached generic single-stage target detector and did not perform the attack study on the advanced remote sensing target detector.

In the field of adversarial samples for remote sensing image models, the research of adversarial attack algorithms for classification models is beginning to take shape. However, the implementation of adversarial attacks on remote sensing object detection models is complex due to the characteristics of remote sensing images that are different from close-up images and the complexity of remote sensing object detection models. It causes the research of adversarial attack algorithms for remote sensing object detectors to be still scarce, and large-scale systematic analysis and research are even more lacking, which is obviously not conducive to the model to deal with the security threats that may appear in the future. We propose well-performing adaptive improved adversarial attack algorithms based on the characteristics of remote sensing object detection models to construct transferable adversarial samples.

In view of the current situation of the lack of research on the countermeasure samples on the remote sensing object detector, this work first realizes the adaptation of the typical white-box attack algorithm, successfully implements the vanishing attack on the model, and builds a comparative baseline for the subsequent research, and then proposes the adaptive improved global pixel attack method and local pixel attack method according to

the characteristics of the remote sensing target detector. Specifically, the contributions of this paper are as follows:

1. We design an attack adaptation method based on interpolation scaling and block-based perturbation superposition to successfully adapt a variety of typical white-box attacks to the remote sensing object detection, and we use a hot restart strategy to effectively solve the perturbation failure problem caused by image splitting preprocessing transformations. Moreover, through the design of the loss function, a joint attack on the two-stage remote sensing target detector RPN module and the classification refinement module is realized;
2. We propose an attack sensitivity mapping algorithm to realize the search for the location of sensitive pixel points and improve the attack effect;
3. We propose a local pixel attack algorithm (Sensitive Pixel Localization C&W, SPL-C&W) based on sensitive pixel point localization. By searching the location of attack-sensitive pixel points in the original image and constructing a technical scheme for mask extraction of the attack region, the local pixel attack is implemented while still adopting the hot restart perturbation update strategy, balancing the attack effect with the overhead of attacking the number of modified pixels.

2. Related Work

2.1. Remote Sensing Image Object Detection

Generic object detectors are divided into two-stage object detectors and single-stage object detectors. The two-stage object detector usually consists of two stages. Take Faster R-CNN [22] as an example, it consists of the following three modules: (1) Backbone network for extracting features. The feature information in the image is extracted by the convolution operation of this module to form a Feature Map and sent to the next stage. (2) Region Proposal Network (RPN). The function of this module is to distinguish the foreground from the background. (3) Classification refinement stage. In this stage, local features are extracted from the proposed regions sent in to achieve further classification refinement and position regression, obtain the specific categories and positions of objects, and obtain the final prediction results through postprocessing. The YOLO [23], as an example of a single-stage object detector, differs from the two-stage object detector in that it does not contain a module that distinguishes explicitly between the foreground and the background but directly regresses the class and location of the target to be detected in the output layer.

Remote sensing images differ in several ways from conventional images in real life. Firstly, they possess an enormous size; for example, the images in the DOTA dataset [24] range from 800×800 to 4000×4000 . Secondly, the objects in remote sensing images are generally small and densely arranged, with some appearing in a formation and complex background. Thirdly, these images are primarily captured from an aerial view, leading to immense variability in the size, direction, and shape of the objects. Therefore, developing a remote sensing object detection model requires a unique approach. The rotational object detectors using deep neural networks are found to be suitable for remote sensing object detection due to its progress in deep learning technology and abundant data resources. In recent years, several excellent remote sensing object detectors based on deep neural networks have been developed [5,6,8] such as the two-stage Gliding Vertex [5] model and the ROI-Transformer [6] model, where ROI stands for Region of Interests. The Gliding Vertex model represents the location of the object using four points' offset on a horizontal frame. On the other hand, the ROI-Transformer model uses the two-stage object detector architecture for detection, and its final prediction stage's output is similar to that of a generic object detector—comprising the object's coordinate information and its category label. However, the coordinates for determining the rotation detection frame differ slightly from those of the Gliding Vertex model. Moreover, due to the massive size of remote sensing images, feeding the entire high-resolution image directly into the operation during training or testing is not possible using most GPUs. Therefore, most remote sensing object detection models perform preprocessing image transformations, cutting images into feasible sizes

before training and testing. Overall, these unique characteristics of remote sensing images require the development of novel and robust remote sensing object detection models.

2.2. Adversarial Examples

The adversarial attack optimizes the loss function by adding small perturbations directly to the image pixels through an optimization algorithm, thus causing the modified sample detection results to deviate from the actual label. The attacks can be classified into global pixel attacks and local pixel attacks according to the number of image pixels involved. The global pixel attack involves the whole image, while the local pixel attack involves only some pixels. Almost all traditional attacks against image classification tasks are global pixel attacks, such as FGSM [11], I-FGSM [13], MI-FGSM [14], PGD [12], and C&W [15]. From the analysis of the above-mentioned related studies, it is clear that the specific implementation of category attacks that reduce the classification accuracy of object detectors should be the modules in the detectors involved in the classification process. In this paper, the subsequent research on the vanishing attack for remote sensing object detectors should also focus on these modules of the object detectors to achieve the attack effect. The effect of the attack on the target detector by the antagonistic sample can be briefly described as two: reduction of detector accuracy and vanishing attack. The DAG(Dense Adversary Generation) attack algorithm proposed by Xie et al. [25] and subsequently proposed by Chen et al. [7]. The ShapeShifter algorithm focuses the attack on the classification refinement phase of the two-stage object detector. The common goal of the vanishing attack is to make all objects in the image bypass the detection of the detector. The vanishing attack involves more orders of magnitude and is more difficult than an attack that misclassifies an object in the image. Vanishing attacks also reduce the detector's accuracy, target hiding is often more relevant, and the studies we have conducted have focused on vanishing attacks.

2.3. CAM

Among the interpretable approaches to neural networks, some studies on the visualization of image classification model predictions have commonly used the gradient information flowing back in the model [8,10]. In these studies, CNN predictions are visualized by highlighting pixels in the image that are “more important” for a particular class (i.e., changes in these pixels have the most significant impact on the prediction scores for that class). An example of the Class Activation Mapping (CAM) algorithm [8] on a GoogleNet classification model is shown in Figure 1. CAM uses the gradient information returned on the image classification model to obtain the response value of a layer of the feature map for a particular class, where a higher response value represents a higher contribution to the classification of that class. This way, a class activation map is obtained, and then the image regions most relevant to a particular class can be identified from the map by upsampling it.

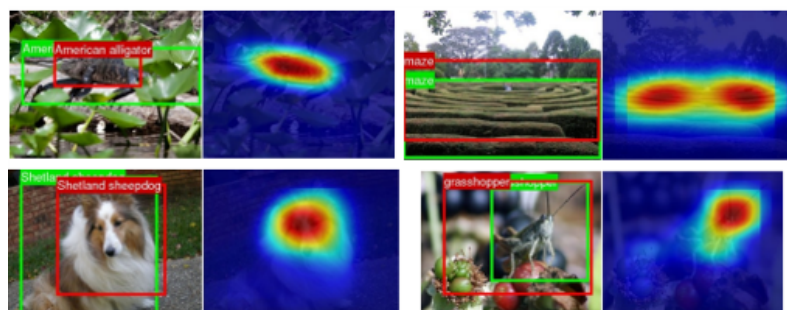


Figure 1. A CAM's example.

3. Attack Strategies

3.1. Problem Definition and Notation for Deep Neural Networks

First, the notation is defined by t to denote the number of the round of the multi-round attack, and the resulting adversarial sample is denoted as x_t^* ; α denotes the step size of the

multi-round attack, which represents the value of pixel modification at each round of the gradient-based attack algorithm; ε denotes the maximum value of pixel modification when using the L_∞ norm constraint; $J(\cdot)$ denotes the loss function of the adversarial attack, and $\nabla_x J(\cdot)$ denotes the gradient obtained by passing the attack loss back to the original image x . Y denotes the correct classification class of the image after the classification model Z , and Y' represents the target class set by the attack.

Our white-box attack is inspired by the formulation of the C&W attack. C&W [15] is an optimization-based adversarial attack algorithm. The algorithm proposes a new loss function that reduces the distance between the adversarial sample and the original sample while increasing the prediction error of the target model, i.e., subject to the L_2 norm. To facilitate optimization, the C&W algorithm first introduces the \tanh space variable w to represent the adversarial sample x^* , and the transformation range of x^* is transformed from $(0, 1)$ to $(-\infty, +\infty)$. The variable w is calculated as follows:

$$x^* = \frac{1}{2}(\tanh(w) + 1). \quad (1)$$

The optimization expression for C&W against attacks consists of two parts, which are formulated as follows:

$$\min \|\delta\|_2^2 + c \cdot f(x^*), \quad (2)$$

here, $\|\delta\|_2^2$ represents the Euclidean distance between the original image and the generated adversarial sample, and $f(\cdot)$ represents the probability distance between the category correctly predicted by the model and the category incorrectly predicted into the attack target. δ and $f(\cdot)$ represent the expressions respectively as follows:

$$\delta = x^* - x, \quad (3)$$

$$f(x^*) = \max(\max\{Z(x^*)_i : i \neq Y'\} - Z(x^*)_{Y'}, -k), \quad (4)$$

where c is a parameter measuring the loss function of the two components and k is the confidence that the model is misclassified, the larger the chance that the adversarial sample will be misclassified, but it will also be harder to find, and a better balance is usually achieved when it is greater than 40. The attack is usually implemented using the Adam optimizer to optimize the process.

3.2. Splitting Preprocessing

Due to the huge size of remote sensing images, most GPUs cannot support putting the whole high-resolution image directly on the GPU for calculation during training or prediction. If the image is scaled directly, some information will be lost. We take a single large image and split it into a collection of multiple subgraph splits that the GPU can process by some rules, i.e., we preprocess the image by splitting. The pseudo-code of the *Split* algorithm is shown in Algorithm 1.

Given a remote sensing image x of size (w, h) , the relevant parameters are as follows: (1) rate represents the rate of image scaling before cutting; (2) the Boolean type parameter padding for whether to perform a 0-pixel padding operation; (3) gap is represented by the distance between two adjacent slices; (4) the size of the single subgraph after cutting. The significance of the cutting parameters is as follows: rate represents the scaling ratio and the expected values are 1.0 and 0.5, where 1.0 means no image scaling and 0.5 means reducing the image size by half. When cutting the image, the split size is smaller than the set sub-size, and when the parameter padding is set to True, the pieces will be transformed by image padding, and the size will be expanded to the sub-size. The scaling transformation and the padding transformation are standard data enhancement tools used in image tasks, and these two parameters can be considered data enhancement operations in the cut preprocessing process.

Algorithm 1 The Framework of *Split***Require:** original remote sensing image x **Require:** $rate, padding, gap, subsize$ **Ensure:** subgraph set $P = \{p_1, p_2, \dots, p_n\}$

```

1: Let  $(left, up) = (0, 0)$ 
2: resize  $x$  according to  $rate$  and update  $(w, h)$ 
3: while  $left \leq w$  do
4:    $up = 0$ 
5:   while  $up \leq h$  do
6:     Split subgraph according to  $(left, up)$ 
7:     Fill the subgraph based on the  $padding$  parameter
8:     Update  $P$  set
9:     if  $up + subsize \geq h$  then
10:      Break
11:    end if
12:    Update  $up$ 
13:  end while
14:  if  $left + subsize \geq w$  then
15:    Break
16:  end if
17:  Update  $left$ 
18: end while
19: return  $P$ 

```

In addition, the target information at the edge of the slices may be missing during the cutting process. Therefore, the parameter gap is set to a certain overlap area between the adjacent subgraphs. This parameter is usually set to around 50% of the cut image size. The implementation process of the cutting preprocessing algorithm can be regarded as the process of sliding window cutting on the image. When updating the top-left coordinate $(left, up)$ of the subgraph corresponding to the original image during each slide, a judgment operation is needed. Taking the update of $left$ as an example, if $left + slide + subsize \leq w$, then the $left$ update is $left + slide$; otherwise, it is updated to $w - subsize$. The update of up is similar. The following is the process of the cutting preprocessing, as shown in the Figure 2. After the cutting preprocessing, we obtain a set of subgraphs $P = \{p_1, p_2, \dots, p_n\}$.

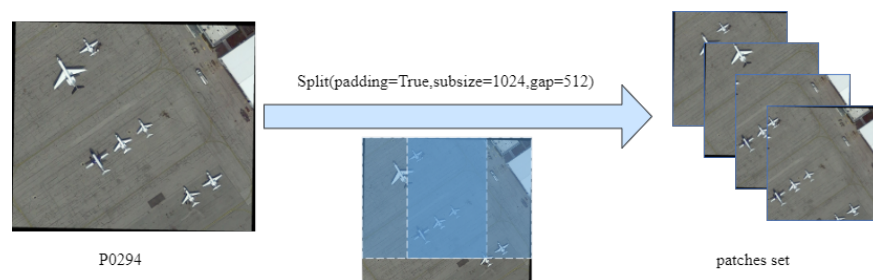


Figure 2. The process of the cutting preprocessing.

3.3. Hot Disturbance Update Strategy

The cutting preprocessing module, while ensuring smooth operation of the remote sensing algorithm, might reduce the effectiveness of the attack. Suppose all slices in the set of subgraph slices are attacked individually. In that case, all the perturbation blocks in the set are added to the original graph at once according to the corresponding positions of the subgraphs to generate the adversarial examples. However, as a result of the many overlaps between subgraphs, the overlap area of the perturbation blocks added in front will be covered by the perturbation blocks added later, resulting in partial loss of perturbation information. Finally, the generated adversarial examples are more biased towards the perturbation blocks added later, which is a kind of “local optimum” in the attack. If

such an adversarial example is preprocessed by cutting and sent to the detector, the detection accuracy of the subgraph slices partially covered by the perturbation may not be reduced, and the accuracy of the final detection results obtained after merging the subgraph detection results will not be significantly reduced. The adversarial examples generated by this perturbation update strategy are less robust when using the cut preprocessing module. Therefore, when adding perturbations to the original image to generate the adversarial examples, instead of a one-time static addition, a dynamic approach should be taken to update the perturbations. To overcome this challenge, we propose a hot restart perturbation update strategy for dynamic updates of adversarial samples.

Hot restart is an approach where, after updating the perturbations for all subgraphs in the set P from the cutting preprocessing stage, the adversarial samples generated are reprocessed through cutting and preprocessing before the next attack round. It is important to note that the non-minimizable image transformation operation cuts off the backpropagation of the attack loss and makes the attack impossible to execute. Therefore, the cut preprocessing used in the attack is rewritten according to the cut preprocessing algorithm. This finer-grained perturbation update strategy can enhance the robustness of the generated perturbations. The drawback is that the update of the slice set is performed before each round of attack, and the perturbation update on the original image brought by a single attack on each slice in the slice set is calculated during each round of attack. Hence, the number of iterations of the attack increases, and the attack speed decreases.

The role of the RPN module in the two-stage target detector is to distinguish the foreground from the background and to fine-tune the object positions. Therefore, we can not only attack the classification refinement module to make it misclassified as a background class when performing a vanishing attack, but also attack the RPN module. The RPN module eventually sorts out the wrong proposed region (i.e., the background region) and sends it to the second stage, thus achieving the effect of the vanishing attack. The RPN module is understood as a binary classification model that distinguishes between foreground and background, and the confidence probability c is understood as the probability that the category is foreground. To accommodate the hot-restart strategy, the final attack process consists of two parts, the attack RPN module and the attack classification refinement module. Justifiably, the joint loss function J consists of two parts, J_{rpn} and J_{roi} . The loss function equation of the attack is expressed as follows:

$$J = \beta J_{rpn} + J_{roi}, \quad (5)$$

$$J_{roi} = - \sum_{i=1}^m \log p(y'_i | b_i). \quad (6)$$

$$J_{rpn} = - \sum_{c \in \{c_i | c_i > obj\}}^n \log(1 - c). \quad (7)$$

where β is the parameter to measure the proportion of J_{rpn} and J_{roi} . J_{roi} represents the loss of the attack classification refinement module, where the $b = \{b_1, b_2, \dots, b_m\}$ denotes the number of detection frames sent to the second stage after filtering in the RPN stage, n . Y denotes the correct classification class of the image after the classification model. $Y' = \{y'_1, y'_2, \dots, y'_n\}$ denotes the set attack target class label. Here is the background class with the label 0.

3.4. Sensitive Pixel Point Selection

The hot restart perturbation update strategy proposed above effectively solves the perturbation failure problem caused by the image cut preprocessing module in the remote sensing target detection model. It implements a global pixel attack with a high attack success rate on the remote sensing target detection model. However, the whole image is the range of modified pixels involved in the global pixel attack. The overhead of modified pixels in this attack can be accepted for conventional images with small sizes. If the size of a remote sensing

image is 3000×3000 , the number of modified pixels involved in a global pixel attack is 9,000,000, which is a large number. In order to balance the attack effect and the overhead, the number of pixels involved in the attack needs to be reduced as much as possible. Inspired by the Class Activation Mapping (CAM) algorithm [8] introduced above, an Attack Sensitivity Mapping (ASM) algorithm is proposed here, and the algorithm pseudo-code is shown in Algorithm 2. This algorithm can search for the pixel points in the original image that are more sensitive to the attack during the attack, and being more sensitive to the attack means that the modification of these pixels contributes more to the attack effect.

Algorithm 2 Attack Sensitivity Mapping (ASM)

Require: original image x , gradient graph on the original image $\nabla_x J$

Require: $K, kernel$

Ensure: coordinates set of Top K sensitive pixels $L = \{l_1, l_2, \dots, l_k\}$

```

1: for each  $x_{(i,j)} \in \nabla_x J$  do
2:   Calculate the sensitivity  $s$  of each pixel
3: end for
4: Update sensitivity matrix  $S$ 
5:  $S' = conv(S, kernel)$ 
6:  $Pixels = \text{Sort}_{desc}(S')$ 
7: Select Top K points from Pixels Set
8: for each  $k \in \text{Top } K$  do
9:   Map the index of  $k$  back to  $S$ 
10:  Obtain sensitive pixels position coordinates
11:  Update Set  $L$ 
12: end for
13: return  $L$ 

```

The algorithm first performs the calculation of the attack sensitivity s . After back-propagating the attack loss J to the original image x , each pixel point $x_{(i,j)}$ is given its gradient, and the direction of the gradient represents the direction where the attack loss decreases fastest at this pixel point. The magnitude of the gradient represents the value of the decrease. The ASM algorithm determines the sensitivity of each pixel point in the image to the attack based on the magnitude of the gradient obtained by backpropagating the attack loss J to the original image x along the model network s . The expression for s is given as follows:

$$s = \sum_n^c \left| \left(\frac{\partial J}{\partial x} \right)_{x_{(i,j)}} \right|, \quad (8)$$

where c represents the number of channels of the image, for a grayscale map with a single channel, $c = 1$, s is ultimately equal to the absolute value of the gradient size on the pixel point; for a color map with three channels of RGB, $c = 3$, s is ultimately equal to the sum of the absolute values of the gradient sizes of the three channels on the pixel point. So far, we can obtain the sensitivity matrix S with the same size as the original image, $S_{(i,j)}$ corresponds to the attack sensitivity information on the pixel $x_{(i,j)}$ in the original image.

The sensitivity matrix S can be regarded as a numerical sensitivity map. In order to make the selection result of sensitive pixel points more robust, the sensitivity matrix S is convolved here before the selection of sensitive pixel points. For the convolution operation, a 1×1 convolution kernel is used with a size of $kernel$, which is usually an odd number. The stride is set to 1. The matrix S' is obtained after the convolution process. All the points in the matrix S' are sorted in descending order according to their values, and the top K points are filtered out to obtain the *Top K* point set; for each point in the *Top K* point set, its position index is mapped back to S before the convolution process, thus obtaining the coordinate information of each point corresponding to $kernel \times kernel$ in the original image. The coordinate information of each point corresponds to $kernel \times kernel$ of sensitive pixels

in the original image. After all the points in the *Top K* point set are processed, the coordinate information of the sensitive pixel points is retained and used for the construction of the attack region mask in next section. Finally, $L = \{l_1, l_2, \dots, l_n\}$ is used to denote the set of coordinates of the sensitive pixel points selected from the original image.

Using the ASM algorithm to search for sensitive pixel points in the original image, the sensitive pixel point localization maps for different values of K at $\text{kernel} = 1$ are shown in Figure 3. The K is taken as $K = 10, 50, 100, 500, 1000$, and the number of sensitive pixels accounts for 0.12%, 0.56%, 1.06%, 4.03%, and 5.35% of the original image, respectively. The distribution pattern of highly sensitive pixels in the image can be perceived from the changes of sensitive pixel location maps based on different K values. That is, most of the pixels that are highly sensitive to the attack are distributed in a particular area of the image containing the object to be detected, which should be focused on when performing local pixel attacks.

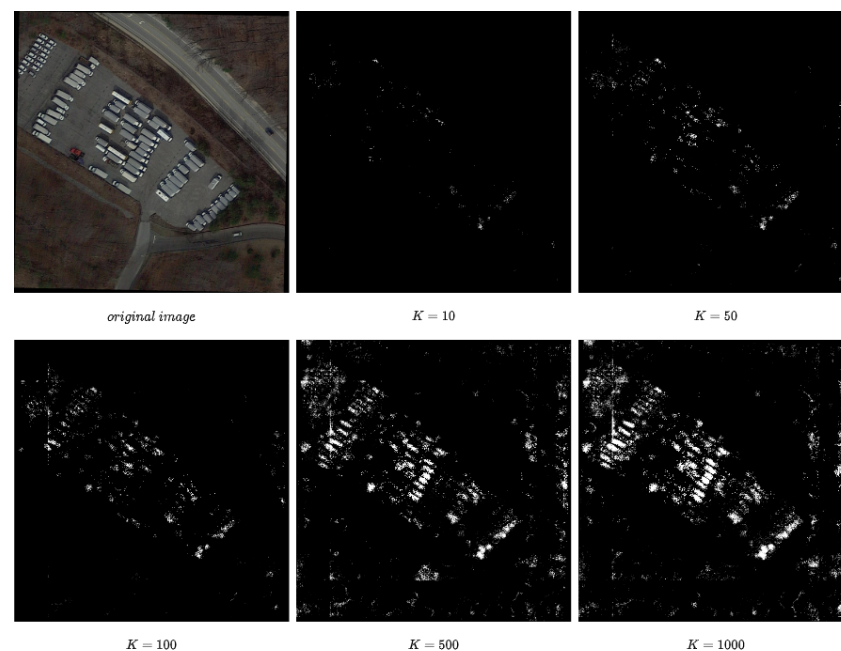


Figure 3. When $\text{Kernel} = 1$, the images of sensitive pixel location under different K values.

3.5. Construction of the Attack Area Mask

After searching the position information of *Top K* sensitive pixel points by the ASM algorithm, the position mask about these K pixel points can be constructed directly by setting the pixel value of the corresponding position to 1 and the remaining positions to 0, as shown in Figure 4. Suppose we use the obtained *Top K* sensitive pixel point position masks as attack region masks straightly. In that case, observation of the constructed position mask shows that there will be many isolated sensitive pixel points, especially at the edges of the image. It might be detrimental to future attack studies that might be conducted in the physical world. In addition, we can find that most pixel points susceptible to the attack are distributed within a specific area of the image containing the object to be detected. This phenomenon inspires us to construct the attack region mask by directly selecting the region containing the object to be detected. The detection frame information in the detection result of the original image by the target detection model can be used for construction. It is worth noting that the target detection model will use the contextual information to correct the classification results, so the attack range is only limited to the inside of the detection frame and may not be optimal; in addition, as the remote sensing target detection model adopts rotating frame detection, the detection frame is mostly a directed rectangular frame, so the edges of some objects with regular shapes will most likely overlap with the detection frame, and thus will not be included in the detection frame mask and finally, not in the attack area. Such a situation exists for the vehicle class in the DOTA dataset, and the contribution of

the object's edge contour feature to the detection result cannot be ignored. Based on the above two points, a slight expansion of the detection frame area should be considered when constructing the mask. The graphical morphology method of expansion is used to slightly expand all the detection frame areas in the image so that part of the contextual information and the edge contours of the object can also be included in the mask range after processing. The parameter $kernel_0$ is the expanded convolution kernel used to perform the expansion operation. This scheme can eventually build a detection frame mask with a regular shape, but the number of pixels involved in the attack will be more significant.

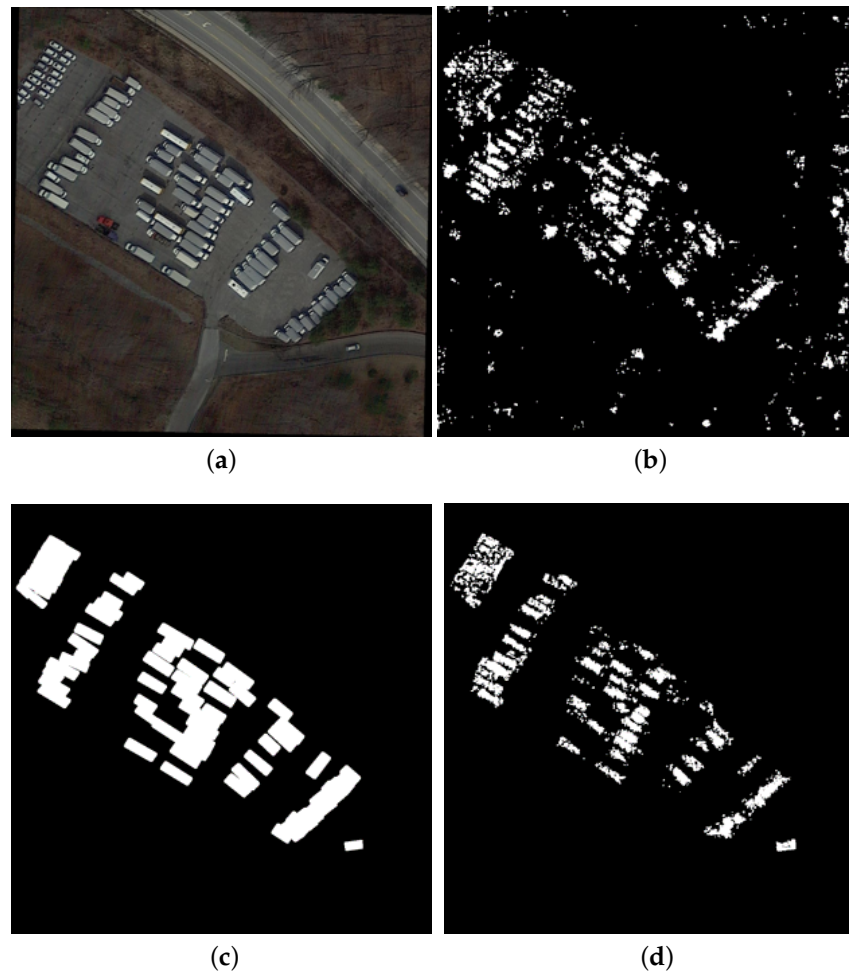


Figure 4. Attack area mask. (a) Original image; (b) $kernel = 3, K = 500$; (c) $M_b, kernel_0 = 7$; (d) M .

In summary, by fusing the ideas of the above two schemes, an attack region mask scheme that uses the detection frame mask to limit further the search range of *Top K* sensitive pixel points are designed here, and this attack region mask scheme will eventually filter out fewer sensitive pixel points. The detailed process of attack region mask construction is described below: firstly, the post-inflated detection frame mask M_b is constructed based on the detection frame information, and the gradient map $\nabla_x J$ is obtained by using M_b to mask the gradient map after passing back the attack loss to the original map, which is expressed as the following:

$$\nabla_x J' = \nabla_x J \odot M_b. \quad (9)$$

The ASM algorithm runs then, by which the search for *Top K* sensitive pixel points can be performed in the area already restricted by M_b so that the localized sensitive pixel points are more distributed in the object itself and around the object. The attack area masks

M can be constructed by constructing the localization map after obtaining the location information of *Top K* sensitive pixel points.

An example attack region mask constructed according to the above process is shown in Figure 4. Figure 4b represents an example map of sensitive pixel location obtained directly according to the ASM algorithm, where $kernel = 3$ and $K = 500$, and the number of sensitive pixels finally selected accounts for 6.09% of all pixels in the original image; Figure 4c represents an example map of detection frame mask constructed according to the detection frame information. Figure 4d represents the final attack region mask constructed according to the attack region mask scheme in this section, with the number of pixels in the attack region accounting for 4.73% of the original image. From Figure 4, it can be seen that the attack region mask constructed according to the scheme designed in this section can, on the one hand, make the attack focus more on the region where the object itself is located, and on the other hand, filter out the pixel points from these regions that are more sensitive to the attack, further reducing the overhead of modifying the number of pixels in the attack.

3.6. SPL-C&W

The process of implementing the vanishing attack by the Sensitive Pixel Localization C&W(SPL-C&W) algorithm proposed in conjunction with the above schemes is shown in Figure 5. The SPL-C&W algorithm performs the vanishing attack against the second-stage classification refinement module of the two-stage target detector.

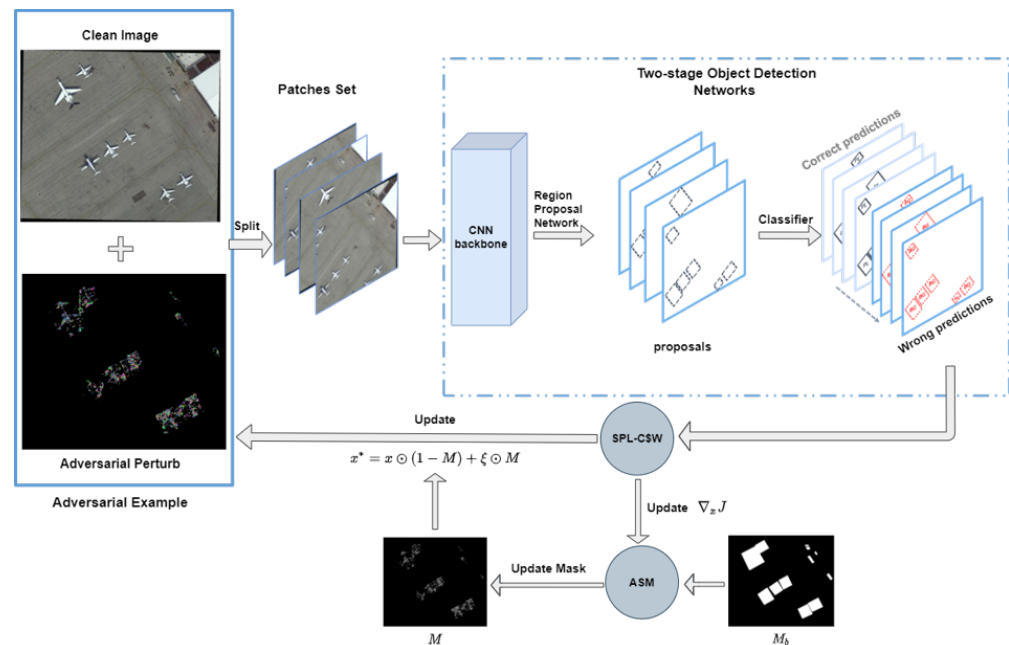


Figure 5. The overview of SPL C&W.

In order to generate an adversarial sample in the attack with splitting preprocessing, a hot restart perturbation update strategy is used in the algorithm. The ASM algorithm is utilized to search sensitive pixel points and construct the attack region mask during each round, explicitly targeting slicing in a single attack. The attack region mask makes it possible to update the perturbations only in the attack region and always keep the same initial value as the original image x for the pixel points outside the attack region. This adversarial sample optimization process can be expressed as follows:

$$x_t^* = x \odot (1 - M_t) + \xi_t \odot M_t, \quad (10)$$

where t represents the number of rounds of attacks, and ξ_t is used to denote the perturbation generated in the t th round of attacks, M_t represents the attack region mask updated from this round, and x_t^* represents the adversarial samples generated in the t th round of attacks.

In the multi-round iterative attack, the attack region mask is first updated at every single attack, and then the adversarial samples are optimized within the limits of the mask. In this way, the local pixel attack based on sensitive pixel point localization is finally achieved. The algorithmic framework of the SPL-C&W attack is shown in Algorithm 3.

Algorithm 3 The Framework of SPL-C&W

Require: real image x ; the detector D ; mask of the detection boxes M_b ; patches set $P = \{p_1, p_2, \dots, p_n\}$; proposal regions set $B = \{B_1, B_2, \dots, B_n\}$ on P ; detected label set $Y = \{Y_1, Y_2, \dots, Y_n\}$; adversarial target label set $Y' = \{Y'_1, Y'_2, \dots, Y'_n\}$

Require: iterations T ; attack parameters c and k ; ASM parameters K and $kernel$; construct mask parameters $kernel_0$

Ensure: adversarial example x^*

- 1: Let $x_0^* = x$; $P^* = Split(x_0^*)$
- 2: **for** $t = 0 \dots T - 1$ **do**
- 3: $P_t^* = Split(x_t^*)$
- 4: Input P_t^* to D
- 5: Update Set B_t , Set Y_t and Set Y'_t
- 6: **if** $Y_t == Y'_t$ **then**
- 7: **Break**
- 8: **end if**
- 9: **for** each patch $p_{t_n}^* \in P_t^*$ **do**
- 10: Input $p_{t_n}^*$ to D
- 11: Attack $p_{t_n}^*$ by Attack method and backpropagation Loss J
- 12: Optimize the ξ_t and obtain the gradient $\nabla_x J(p_{t_n}^*, Y'_{t_n})$
- 13: $\nabla_x J(p_{t_n}^*, Y'_{t_n})' = \nabla_x J(p_{t_n}^*, Y'_{t_n}) \odot M_b$
- 14: // Search for the locations of sensitive pixels by ASM algorithm
- 15: $L = ASM(x, \nabla_x J(p_{t_n}^*, Y'_{t_n})')$
- 16: Update Mask M according to L
- 17: Update x_t^* according to $x_t^* = x \odot (1 - M) + \xi_t \odot M$
- 18: **end for**
- 19: **end for**
- 20: **return** $x^* = x_T^*$

4. Evaluation

In this section, the SPL-C&W algorithm is used to perform the vanishing attack on two white-box remote sensing target detection models, and the rationality of the sensitive pixel point localization and attack region mask construction scheme proposed in this chapter is verified through comparative experiments; subsequently, it is verified that the SPL-C&W local pixel attack algorithm proposed in this chapter can balance between the attack success rate and the attack modification pixel overhead through comparison experiments with the attack effect of global pixel attack.

4.1. Setup

The experiments in this section are based on Pytorch and are conducted on a Linux server with a GeForce RTX2080 Ti graphics card and 64 GB of RAM. The adopted dataset uses validation set images from the DOTA V 1.0 remote sensing image dataset. The local pixel attack takes a long time to perform multiple searches for sensitive pixel points and the construction of attack region masks in each round of attack, and 100 images from the validation set are randomly selected for the experiments here.

In this chapter, the Gliding Vertex [5] with ROI-Transformer [6] detector is still used as the local white-box model on which the local pixel attack algorithm based on the feedback of sensitive pixel points is implemented. The R3Det [26] and BBAAVectors [27] are used as the local black-box model, and Gliding Vertex and ROI-Transformer are mutually black-box models to verify the transferability of the generated adversarial samples. The four models

of the experimental attack can operate normally locally before being attacked, and the detection performance on the 100 randomly selected images in this chapter is good, with the Gliding Vertex achieving 87.41% mAP (Mean Average Precision), the ROI-Transformer detector achieving 82.88% mAP, and the R3Det achieving 87.04% mAP, and BBAVectors can reach 88.79% mAP.

To compare with the global pixel attack method without ASM, the attack we use has the same parameters as follows: (1) normalize the original image from the pixel range of $[0, 255]$ to $[0, 1]$; (2) the attacks are constrained by the L_2 norm; (3) the maximum number of attack rounds $T = 50$ for multi-round attacks; (4) the model misclassification confidence $k = 50$ and the loss function parameter $c = 10$. It should be noted that in the experiment, a convolution operation was performed on the gradient map using the attack sensitivity algorithm with a convolution kernel of $kernel = 3$, and another parameter $K = 500$ was used for the ASM algorithm. The inflated convolution kernel $kernel_0 \in [1, 3, 5, 7, 9]$ detects the box mask when constructing the attack region mask.

4.2. Effectiveness Analysis of Sensitive Pixel Location

In order to verify the effectiveness of the sensitive pixel points positioned in the attack and the rationality of the attack region mask scheme, this section constructs different attack region masks for comparison experiments. It implements SPL-C&W local pixel attacks on the white-box Gliding Vertex model and ROI-Transformer model to analyze the differences in attack effects.

The descriptions of several groups of attack area masks used in the experiments are as follows: (1) Small squares of size 3×3 are randomly selected in the image several times to construct area masks. Then, local pixel attacks are performed, symbolically denoted as *randomK*, and K represents the number of randomly selected squares. The randomly selected 5000 and 10,000 small squares in the image correspond to the average modification rates of 4.27% and 8.26% in the images participating in the experiments, respectively, and the attacks have the same average image modification rates on the two white-box models; (2) using the ASM algorithm, the convolutional operation is performed using a convolutional kernel of $kernel = 3$ with parameters $k = 500$, constructing a position mask as the attack region mask for each round of SPL-C&W attack, symbolically denoted as *conv3*. The average image modification rate of the final attack is 7.63% on the Gliding Vertex model and 7.33% on the ROI-Transformer model; (3) Using the attack region mask scheme, the detection frame mask restriction ASM algorithm is added, searching the area range of *Top K* sensitive pixel points. The detection box is not inflated here first to exclude the influence of other factors, and the parameter $kernel_0 = 1$ is symbolically represented as *box1conv3*. The final attack has an average image modification rate of 2.82% on the Gliding Vertex model and 2.29% on the ROI-Transformer model. Figure 6 shows the comparison of the local pixel attack effect under the above different attack region masks.

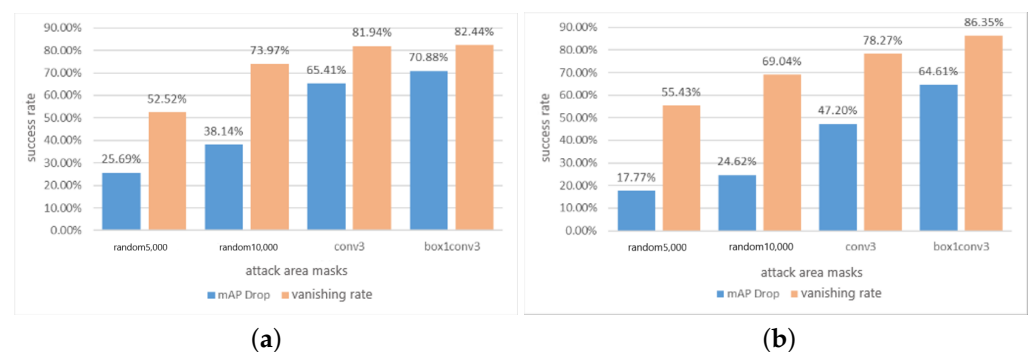


Figure 6. Comparison of attack effects based on different attack area masks. (a) Gliding Vertex ; (b) ROI-Transformer.

Figure 6a shows the result of attacking the Gliding Vertex model, and Figure 6b shows the result of attacking the ROI-Transformer model. From Figure 6, it can be intuitively seen that the attack effect of the local pixel attack based on *conv3* mask is better than that of the local pixel attack based on *random10,000* mask for the Gliding Vertex model and the ROI-Transformer model with less than 1% difference in the attack modification rate. The mAP Drop of the attack is 27.27% and 22.28% higher on the two models, and the vanishing rate is 7.97% and 9.2% higher, respectively. The experimental results effectively demonstrate the effectiveness of the selected sensitive pixel points in enhancing the effect of local pixel attacks.

In addition, it can be learned from Figure 6 that the *box1conv3* mask constructed according to the attack region mask scheme reduces the attack modification rate in the SPL-C&W attack. In contrast, the attack effect performs the best in several groups of comparison experiments. With less than 2% difference in the attack modification rate, the attack improves the mAP Drop on the two models by 45.19% and 46.84% for the Gliding Vertex model and ROI-Transformer model respectively, and the vanishing rate by 29.92% and 30.92%; compared with the SPL-C&W attack based on the *conv3* mask, for the Gliding Vertex and ROI-Transformer models, the attack on the two models has a 4.81% and 5.04% lower modification rate, 5.47% and 17.41% higher mAP Drop, and vanishing rate increased by 0.5% and 8.08% respectively. The SPL-C&W attack based on *box1conv3* mask can achieve more than 80% vanishing rate on two models with less than 3% modification rate only. The experimental results further prove the effectiveness of the selected sensitive pixel points on the local pixel attack effect and verify the superiority of the attack area mask scheme.

4.3. Attack on Remote Sensing Object Models

After verifying the effectiveness of the attack sensitivity mapping algorithm and the constructed attack region mask proposed in this chapter, it is necessary to explore the attack effect of the SPL-C&W attack. Firstly, we analyze the effect of different detection frame expansion parameters $kernel_0$ on the attack effect of SPL-C&W attack. As shown in Figure 7, the changes of mAP Drop and vanishing rate when the detection frame expansion parameter $kernel_0$ takes different values during the construction of the attack area mask for the local pixel attack with two white-box models. Table 1 shows the average modification rate of the attack on the image for different values of $kernel_0$.

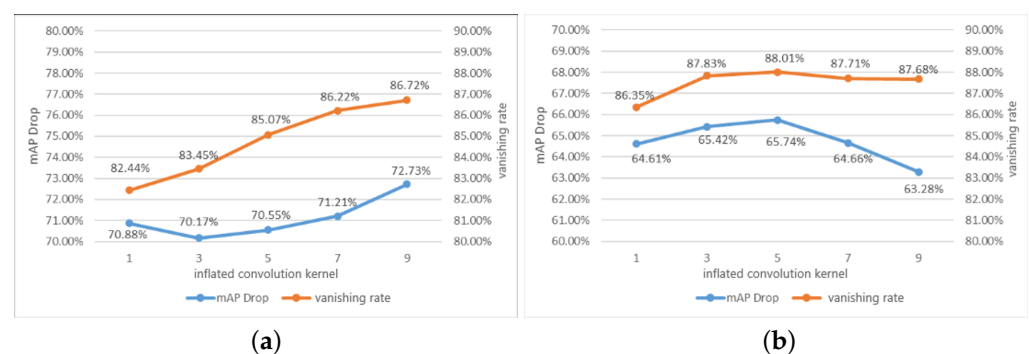


Figure 7. Comparison of attack effects under different size of $kernel_0$. (a) Gliding Vertex; (b) ROI-Transformer.

Table 1. Average modification rate of local pixel attack under different $kernel_0$.

Model	1	3	5	7	9
Gliding Vertex	2.82%	2.95%	3.12%	3.24%	3.37%
ROI-Transformer	2.29%	2.41%	2.49%	2.60%	2.72%

After finding the appropriate detection frame mask parameters, it is necessary to analyze whether the local pixel attack implemented by the SPL-C&W algorithm can achieve

a good balance between the success rate of the attack and the attack modification pixel overhead with the appropriate parameters. A comparison of the attack effectiveness between the global pixel attack SW-C&W and the local pixel attack SPL-C&W is performed, as shown in Table 2 for the comparison of the attack success rate against the sample in the white-box case. From Table 2, we can learn that the difference between the mAP Drop of the SPL-C&W attack on the two white-box models and the SW-C&W attack is less than 5%. The maximum vanishing rate is reduced by about 11%, but the modification rate can be reduced from the original 100% to less than 4%. The loss of the attack effect brought by the decrease in the number of attack pixels is within the acceptable range.

Table 2. Comparison of global and local pixel attacks on two models.

Model	Attack	mAP Drop	Vanishing Rate	Modification Rate
Gliding Vertex	SW-C&W	76.12%	97.70%	100%
	SPL-C&W	72.73%	86.72%	3.37%
ROI-Transformer	SW-C&W	65.43%	94.74%	100%
	SPL-C&W	65.74%	88.01%	2.49%

As shown in Figures 8 and 9, the comparison of the transferability of the adversarial samples generated by the global pixel attack SW-C&W and the local pixel attack SPL-C&W on the black-box model is shown. From Figures 8 and 9, we can see that the difference between the adversarial samples generated by SW-C&W and SPL-C&W on the two white-box models is less than 0.1% on the local black-box R3Det model, which is related to the fact that the R3Det model itself is easier to break single-stage target detection model; the ROI-Transformer model as a black-box model, the transferability of the adversarial samples generated by SPL-C&W is even improved compared with SW-C&W, the mAP Drop is improved by 3.62%, and the vanishing rate improves by 20.23%; when the Gliding Vertex model is used as a black-box model, the attack of the adversarial samples generated by SPL-C&W For the local black-box BBAVectors model, the mAP Drop of the adversarial samples generated by the SPL-C&W algorithm attacking the Gliding Vertex model and the adversarial samples generated by the ROI-Transformer model decreased by 5.34% and 4.06%, respectively. Moreover, 4.06% and the vanishing rate decrease by 4.55% and 0.56%, respectively, and the loss of transferability of the adversarial samples due to the reduction of the number of attack pixels is within the acceptable range.

The experimental results effectively demonstrate that the local pixel attack proposed in this chapter reduces the attack overhead at the expense of a small fraction of the attack success rate and transferability. The average modification rate of the image is reduced by more than 96% relative to the global pixel attack.

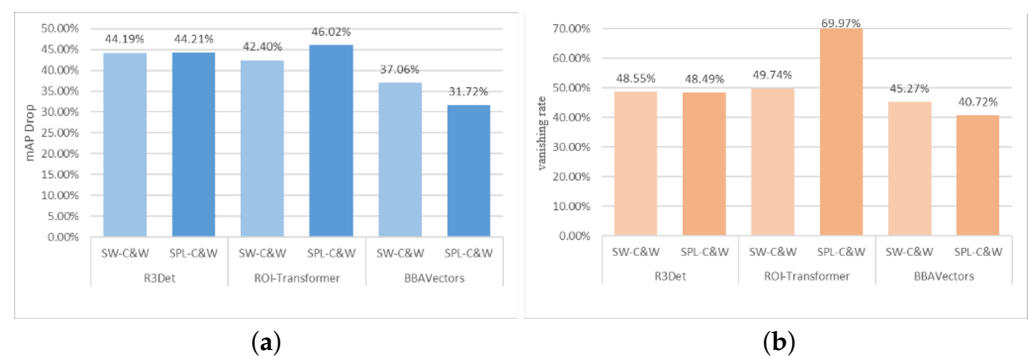


Figure 8. Adversarial Sample Transferability Tests Generated by Attacking the Gliding Vertex. (a) mAP Drop; (b) vanishing rate.

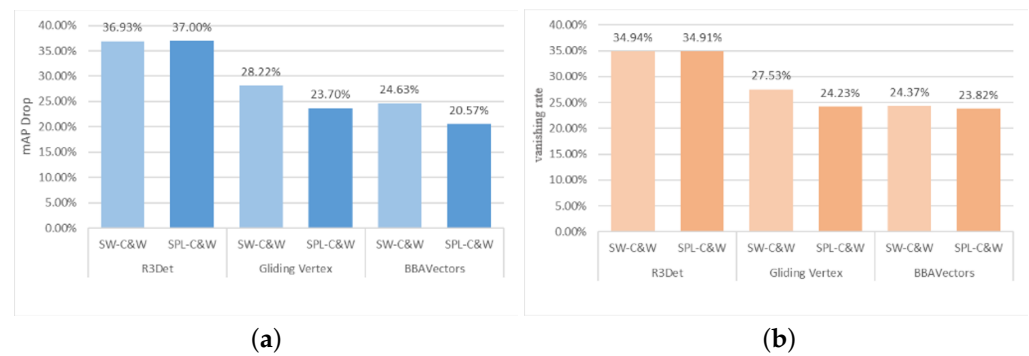


Figure 9. Adversarial sample transferability test generated by attacking the ROI-Transformer. (a) mAP Drop; (b) vanishing rate.

Figures 10 and 11 present the visual detection results of the SPL-C&W attack on the two white-box models obtained for the adversarial samples. The first column of each row shows the visual detection result of the original image on the model, and the second column shows the final generated noise map. The third column shows the visual detection result of the adversarial sample on the model. From Figures 10 and 11, we can see that after applying the vanishing attack on the two white-box models using the SPL-C&W method, the effect of hiding all objects in the image by attacking only some pixels in the image is successfully achieved.

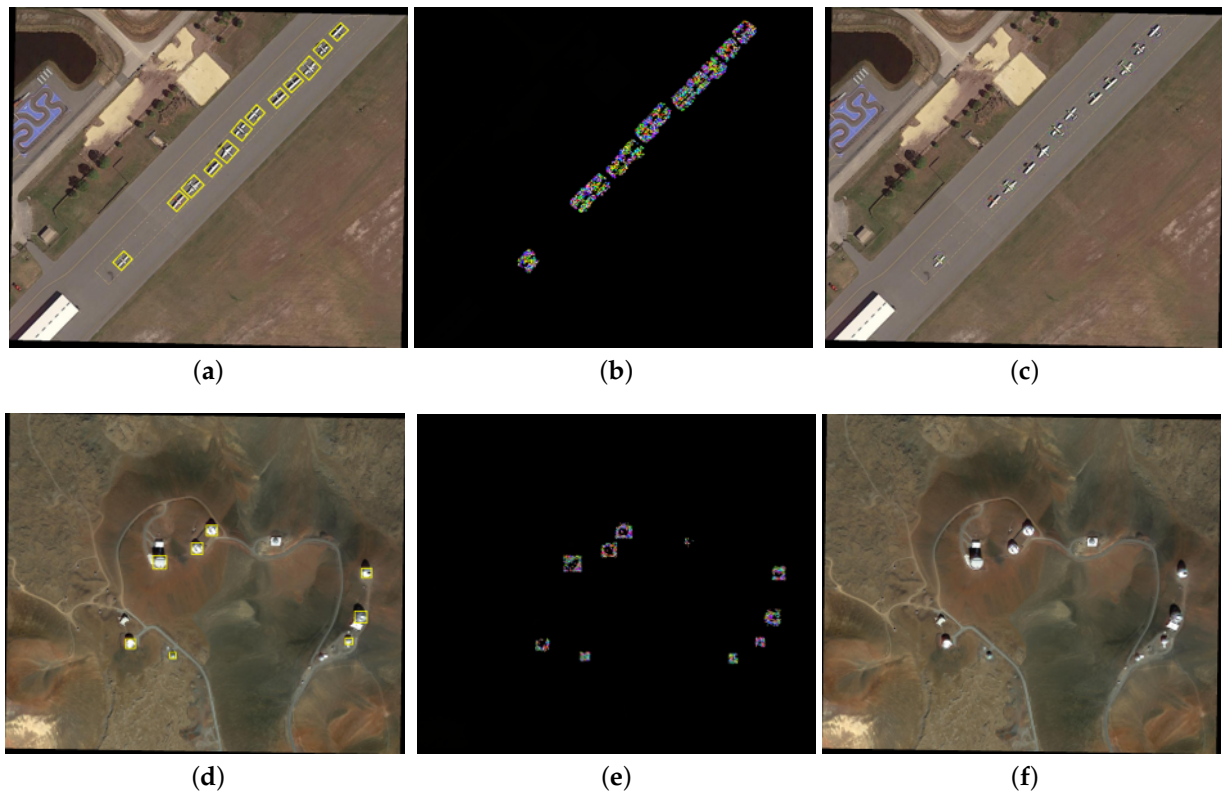


Figure 10. Results visualization for Gliding Vertex model. (a) Original image; (b) Perturbation image; (c) Adversarial image; (d) Original image; (e) Perturbation image; (f) Adversarial image.

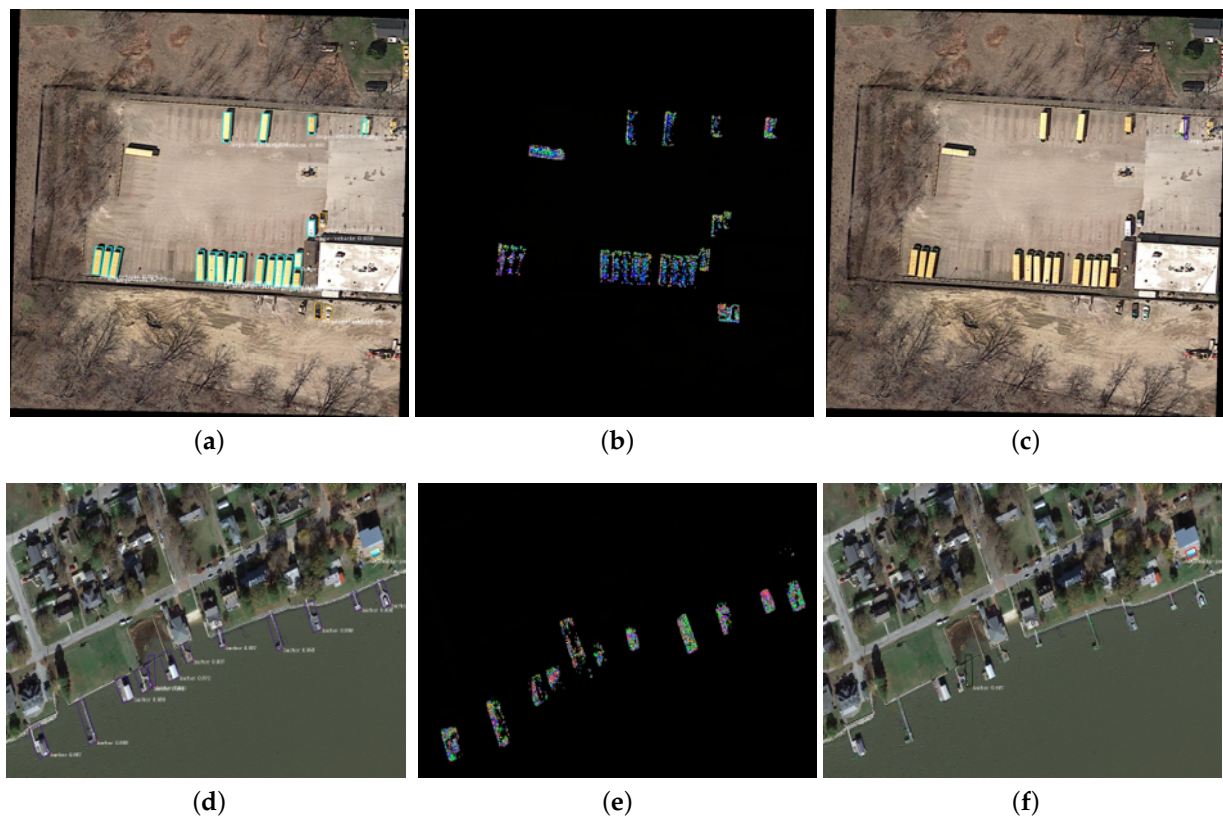


Figure 11. Results visualization for ROI-Transformer model. (a) Original image; (b) Perturbation image; (c) Adversarial image; (d) Original image; (e) Perturbation image; (f) Adversarial image.

5. Conclusions

In this work, we propose a local pixel attack method SPL-C&W applicable to remote sensing target detection models and implement the vanishing attack on several remote sensing object detectors. We design an attack adaptation method based on cut blocks' perturbation superposition and successfully adapt a typical adversarial attack algorithm to a remote sensing target detection model. A generalized hot restart perturbation update strategy is used in the algorithm, which can be used in combination with various attack algorithms to effectively solve the perturbation failure problem caused by cutting preprocessing modules. We implement the search for the location of attack-sensitive pixel points in the image by designing an attack sensitivity mapping method. The attack region mask scheme with the detection frame information limits the search range of sensitive pixel points, reducing the overhead of attack modification pixels. The experimental results show the superiority of the adversarial samples generated by our method in terms of attack effect and transferability, laying a solid foundation for the subsequent research on adversarial samples in the field of remote sensing image recognition and providing reference and guidance for the development of more secure and robust remote sensing image recognition models in the future.

Author Contributions: Conceptualization, methodology, L.L.; validation, Z.X. and H.G.; formal analysis, D.Y.; investigation, Z.X.; resources, H.G.; data curation, D.H.; writing—original draft preparation, L.L.; writing—review and editing, Z.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been supported by Beijing Science and Technology Program No. Z211100004121009.

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

References

1. Hu, F.; Xia, G.S.; Hu, J.; Zhang, L. Transferring Deep Convolutional Neural Networks for the Scene Classification of High-Resolution Remote Sensing Imagery. *Remote Sens.* **2015**, *7*, 14680–14707. [\[CrossRef\]](#)
2. Zhang, F.; Du, B.; Zhang, L. Scene Classification via a Gradient Boosting Random Convolutional Network Framework. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 1793–1802. [\[CrossRef\]](#)
3. Chen, Y.; Jiang, H.; Li, C.; Jia, X.; Ghamisi, P. Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 6232–6251. [\[CrossRef\]](#)
4. Cheng, G.; Yang, C.; Yao, X.; Guo, L.; Han, J. When Deep Learning Meets Metric Learning: Remote Sensing Image Scene Classification via Learning Discriminative CNNs. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 2811–2821. [\[CrossRef\]](#)
5. Xu, Y.; Fu, M.; Wang, Q.; Wang, Y.; Chen, K.; Xia, G.S.; Bai, X. Gliding Vertex on the Horizontal Bounding Box for Multi-Oriented Object Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 1452–1459. [\[CrossRef\]](#) [\[PubMed\]](#)
6. Ding, J.; Xue, N.; Long, Y.; Xia, G.S.; Lu, Q. Learning RoI Transformer for Oriented Object Detection in Aerial Images. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 2844–2853. [\[CrossRef\]](#)
7. Chen, S.T.; Cornelius, C.; Martin, J.; Chau, D.H.P. ShapeShifter: Robust Physical Adversarial Attack on Faster R-CNN Object Detector. In Proceedings of the Machine Learning and Knowledge Discovery in Databases, Dublin, Ireland, 10–14 September 2018; Berlingerio, M., Bonchi, F., Gärtner, T., Hurley, N., Ifrim, G., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2019; pp. 52–68. [\[CrossRef\]](#)
8. Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; Torralba, A. Learning Deep Features for Discriminative Localization. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 2921–2929. [\[CrossRef\]](#)
9. Qian, W.; Yang, X.; Peng, S.; Yan, J.; Guo, Y. Learning Modulated Loss for Rotated Object Detection. *Proc. AAAI Conf. Artif. Intell.* **2021**, *35*, 2458–2466. [\[CrossRef\]](#)
10. Simonyan, K.; Vedaldi, A.; Zisserman, A. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *arXiv* **2014**, arXiv:1312.6034.
11. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and Harnessing Adversarial Examples. In Proceedings of the ICLR, San Diego, CA, USA, 7–9 May 2015.
12. Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. Towards Deep Learning Models Resistant to Adversarial Attacks. In Proceedings of the ICLR, Vancouver, BC, Canada, 30 April–3 May 2018.
13. Kurakin, A.; Goodfellow, I.; Bengio, S. Adversarial examples in the physical world. In Proceedings of the ICLR, Toulon, France, 24–26 April 2017.
14. Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; Li, J. Boosting Adversarial Attacks with Momentum. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 9185–9193. [\[CrossRef\]](#)
15. 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, 22–24 May 2017; pp. 39–57. [\[CrossRef\]](#)
16. Chen, L.; Zhu, G.; Li, Q.; Li, H. Adversarial Example in Remote Sensing Image Recognition. *arXiv* **2020**, arXiv:1910.13222.
17. Ai, S.; Voundi Koe, A.S.; Huang, T. Adversarial perturbation in remote sensing image recognition. *Appl. Soft Comput.* **2021**, *105*, 107252. [\[CrossRef\]](#)
18. Burnel, J.C.; Fatras, K.; Flamary, R.; Courty, N. Generating Natural Adversarial Remote Sensing Images. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 1–14. [\[CrossRef\]](#)
19. Czaja, W.; Fendley, N.; Pekala, M.; Ratto, C.; Wang, I.J. Adversarial examples in remote sensing. In Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Seattle, WA, USA, 6–9 November 2018; pp. 408–411. [\[CrossRef\]](#)
20. Xu, Y.; Ghamisi, P. Universal Adversarial Examples in Remote Sensing: Methodology and Benchmark. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 1–15. [\[CrossRef\]](#)
21. Den Hollander, R.; Adhikari, A.; Tolios, I.; Van Bekkum, M.; Bal, A.; Hendriks, S.; Kruithof, M.; Gross, D.; Jansen, N.; Perez, G.; et al. Adversarial patch camouflage against aerial detection. In Proceedings of the Artificial Intelligence and Machine Learning in Defense Applications II, Online Only, UK, 21–25 September 2020; p. 11. [\[CrossRef\]](#)
22. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [\[CrossRef\]](#) [\[PubMed\]](#)
23. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788. [\[CrossRef\]](#)
24. Xia, G.S.; Bai, X.; Ding, J.; Zhu, Z.; Belongie, S.; Luo, J.; Datcu, M.; Pelillo, M.; Zhang, L. DOTA: A Large-Scale Dataset for Object Detection in Aerial Images. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3974–3983. [\[CrossRef\]](#)

25. Xie, C.; Wang, J.; Zhang, Z.; Zhou, Y.; Xie, L.; Yuille, A. Adversarial Examples for Semantic Segmentation and Object Detection. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 1378–1387. [\[CrossRef\]](#)
26. Yang, X.; Yan, J.; Feng, Z.; He, T. R3Det: Refined Single-Stage Detector with Feature Refinement for Rotating Object. *Proc. AAAI Conf. Artif. Intell.* **2021**, *35*, 3163–3171. [\[CrossRef\]](#)
27. Yi, J.; Wu, P.; Liu, B.; Huang, Q.; Qu, H.; Metaxas, D. Oriented Object Detection in Aerial Images with Box Boundary-Aware Vectors. In Proceedings of the 2021 IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 3–8 January 2021; pp. 2149–2158. [\[CrossRef\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.