

An Efficient Two-Stage Black-Box Sparse Adversarial Attack Method Based on Intelligent Optimization

Ting Mei¹, Hui Lu¹(✉), Shiqi Wang¹, Ruoliu Zhang¹

¹ School of Electronic and Information Engineering, Beihang University, Beijing, 100191, China
mluhui@buaa.edu.cn

Abstract. Sparse adversarial attacks have attracted increasing attention due to the advantage of low attack costs by limiting the number of modified pixels. However, some sparse attacks assume full access to information from deep neural networks (DNNs), often necessitating a large number of queries, making them impractical. Other methods only constrain the number of perturbed pixels, regardless of the size of the individual perturbation to each pixel, resulting in easily detectable in vision. To overcome these issues, we propose a two-stage black-box sparse attack approach that efficiently generates adversarial examples with small distortions. The proposed method first employs sparse attacks to generate an initial improved perturbation vector that meets the confidence score threshold, using Genetic Algorithm (GA). Subsequently, the size of the initial sparse perturbation vector is optimized to identify the final adversarial example with smaller perturbations through the application of Particle Swarm Optimization (PSO). The experimental results demonstrate that our method can achieve attack success rates comparable to the state-of-the-art black-box sparse attack method within the same budget while introducing more imperceptible distortions. This holds for untargeted and targeted attacks on CIFAR-10 classifiers trained conventionally and adversarially.

Keywords: black attack, sparse attack, confidence scores, intelligent optimization.

1 Introduction

Although DNNs have made impressive achievements in computer vision tasks such as image classification [1], speech recognition [2], and face recognition [3], recent studies have revealed that DNNs can be vulnerable to relatively small perturbations on original images, which brings out security concerns in safety-critical tasks [4,5]. Since existing works have shown that adversarial examples can strengthen the robustness of DNNs via adversarial training [6], increasing numbers of researchers have devoted themselves to proposing better adversarial attack methods.

From the perspective of the number of modified pixels, two types of attack methods are included: one is dense attacks [7-13] that perturb all pixels and restrict the magnitude of perturbations, and the other is sparse attacks [14-24] that change only a

small number of pixels. Although dense attacks have been proven effective, developing novel sparse attack methods is necessary due to their provision of insight into DNNs' interpretability and their practicality in the physical world.

Current sparse attack methods can be classified into two major categories: white-box sparse attacks [14-20] and black-box sparse attacks [21-24]. The former can obtain all information on the target DNNs including the parameters and structures, which is infeasible in practical scenarios. In this case, perturbations usually are determined by obtaining gradient information via back-propagation, which tends to require multiple access to the DNNs. The latter only needs access to the confidence scores for each class outputted by DNNs. In this scenario, the number of perturbed pixels is limited but the size of distortions is unbounded, the pixel-element huge perturbations caused by which hardly fool human eyes.

Considering the above limitations of existing sparse attacks, a two-stage black-box sparse adversarial attack method is proposed. In our dual-stage approach, we employ intelligent optimization techniques to generate adversarial examples that are independent of the target model's internal structure. The process initiates with the generation of an enhanced sparse adversarial example in the first stage, which involves unbounded modifications to individual pixels. Subsequently, these sparse perturbations are refined through optimization in the second stage. The two steps ensure the sparsity and invisibility of the perturbation vector together.

To summarize, the main contributions of this paper are as follows:

1. We propose a two-stage black-box sparse adversarial attack method to generate sparse and more imperceptible adversarial perturbations.
2. In the first stage, we propose to incorporate a class confidence score to determine whether the perturbed pixel position obtained by the black-box sparse attack through GA is an important region. This aims to enhance the quality of the initial sparse perturbation vector.
3. In the second stage, we put forward two strategies within the PSO framework to enhance the optimization of the initial sparse perturbation vector. The first strategy involves discarding a portion of the perturbations and setting the global optimum to the all-zero vector during the PSO initialization phase, considering both the large initial perturbation and the smaller, targeted perturbation. The second strategy is an adaptive mutation approach aimed at avoiding local optima during the optimization process.
4. To evaluate the effectiveness of the proposed method, adversarial attacks on DNN classifiers conventionally and adversarially trained on the CIFAR-10 datasets are conducted considering both targeted and untargeted attack scenarios. Experiments indicate that our approach outperforms the state-of-the-art black-box sparse attack in terms of the size of perturbation and imperceptibility.

The rest of this paper is organized as follows. The related work about sparse adversarial attacks is introduced in Section 2, Some basic concepts and the proposed black-box sparse adversarial attack method are elaborated in Section 3, the experimental studies are given in Section 4, and the conclusions are drawn in Section 5.

2 Related Work

Existing sparse attacks that modify only a small number of pixels include two types of methods according to different levels of access to model information.

In the white-box scenario, attackers tend to make use of the gradients of target DNNs to generate adversarial examples. Papernot et al. [14] construct an adversarial saliency map of the gradient based on the logit of each class to obtain the pixel positions that have the greatest contribution to misclassification. Perturbations only are added to these key pixels in each iteration until the attack is successful. Then, NT-JSMA is proposed by Wiyatno and Xu [15] to conduct the untargeted attacks. Both methods require multiple visits to the target DNN to get a derivative of each pixel. Carlini and Wagner [9] propose the method ($C&W - l_0$) that eliminates the least important perturbations from an adversarial example different from adding key perturbations to a benign sample. Sparse-Fool introduced by Modas et al. [16] exploits the low mean curvature of the decision boundary to calculate the perturbations iteratively. During each update, Deepfool [8] is used to calculate the boundary point and project perturbation onto a point in the affine plane to obtain the next-generation solution. This method is efficient but not sparse enough and cannot achieve a targeted attack. Fan et al. [17] propose SAPF by modeling a sparse attack as a Mixed Integer Programming (MIP) task. This approach aims to optimize both the binary selection factor and the continuous perturbation magnitudes of all pixels simultaneously. The authors also impose cardinality constraints on the selection factor to explicitly regulate the sparsity l_0 . Dong et al. [18] propose the GreedyFool algorithm, which selects k most suitable pixels iteratively for modification according to the gradient information until the attack is successful. Then the greedy strategy is used to reduce the unimportant pixels as much as possible to further improve sparsity. Zhu et al. [19] introduce the Homotopy attack, which generates increasingly sparse perturbations by adding a weighted l_0 norm penalty term to the optimized loss function. The evolutionary homotopy algorithm is leveraged to optimize the weight of the penalty term and the perturbation values. Tian et al. [20] propose a constrained evolutionary algorithm based on two populations to perform sparse attacks, with one population used to find adversarial examples and the other population used to minimize the l_0 and l_2 distances from the original image.

In the black-box scenario, only the outputted confidence scores can be accessed. Narodytska and Kasiviswanathan [21] use a new local search-based technique to construct a numerical approximation of the network gradient, which is then carefully used to construct a small set of pixels in the image for perturbation. Su et al. [22] adopt differential evolution to find one or several perturbed pixels and the corresponding perturbation size. This reflects that adversarial attacks can still be achieved in very limited scenarios, although the attack success rate is not very high. In contrast to the sparse attack approaches minimizing the l_0 distance, Sparse-RS, introduced by Croce et al. [23], employs a random search technique to determine the perturbed pixels. This method fixes the number of perturbed pixels while enabling unlimited perturbation on each pixel. This is very efficient, but it is easy to detect due to large perturbations. The SA-MOO algorithm, introduced by Williams and Li [24], is a multi-target attack

method. Building upon Sparse-RS [23], it increases the likelihood of reducing the added perturbations to zero. Additionally, SA-MOO introduces a novel dominance criterion to evaluate solutions, giving precedence to minimizing the loss function followed by minimizing its l_2 norm. This approach achieves l_0 minimization when l_2 is minimized. SA-MOO is the most advanced black-box sparse attack method, which can achieve a high attack success rate and obtain adversarial examples with minimal perturbations. However, the method allows unlimited perturbations on individual pixels, leading to perceptibility issues, particularly noticeable in smaller image datasets such as CIFAR-10.

3 Methodology

In this section, we first introduce some basic concepts of adversarial attacks in Section 3.1 and then elaborate on the steps of the proposed algorithm in Section 3.2.

3.1 Adversarial Attacks

Let $f: X \in [0,1]^{h \times w \times c} \rightarrow R^K$ be a DNN classifier that assigns a benign image $x \in X$ of height h , width w and channel c to class $\underset{i \in \{1,2,\dots,K\}}{\operatorname{argmax}} f_i(x)$, where K is the number of class labels and the dimensions of the output vector. Then, an untargeted attack can be formulated as the following optimization problem:

$$\underset{i \in \{1,2,\dots,K\}}{\operatorname{argmax}} f_i(x + \vec{\delta}) \neq y, \vec{\delta} \in R^{h \times w \times c}, \min_{\vec{\delta}} D_p(x, x + \vec{\delta}) \quad (1)$$

Where y represents the ground truth class label of the original image x , $\vec{\delta}$ signifies the perturbation vector, and $D_p(x, x + \vec{\delta})$ indicates the discrepancy between the original image x and the perturbed image $x + \vec{\delta}$ typically reflecting l_0 , l_2 or l_∞ distances. It is worth noting that $f_i(x)$ in this paper are the confidence score of class i , not logit. It is easy to achieve a successful attack but more detectable with a larger D_p . Conversely, a smaller D_p typically indicates greater imperceptibility but may result in a failed attack. In terms of sparse attacks, the number of modified pixels l_0 should be limited. Similarly, the goal of a targeted attack is to craft a perturbation vector $\vec{\delta}$ such that

$$\underset{i \in \{1,2,\dots,K\}}{\operatorname{argmax}} f_i(x + \vec{\delta}) = y_t, \vec{\delta} \in R^{h \times w \times c}, \min_{\vec{\delta}} D_p(x, x + \vec{\delta}) \quad (2)$$

Where y_t is the target class label. Therefore, we employ the margin loss in untargeted attacks and the cross-entropy loss in targeted attacks following SA-MOO [24] as follows.

$$L(f, x + \vec{\delta}, y) = f_y(x + \vec{\delta}) - \underset{i \neq y}{\operatorname{argmax}} f_i(x + \vec{\delta}) \quad (3)$$

$$L(f, x + \vec{\delta}, y_t) = -\log f_{y_t}(x + \vec{\delta}) \quad (4)$$

Recent sparse attacks either involve accessing the gradient information of the target DNN to determine the perturbed pixels or limiting the number of perturbed pixels while permitting unbounded perturbations. Our objective is to identify the positions of sparse perturbed pixels without the knowledge of the gradient information of the target DNN, followed by the search for perturbations characterized by restricted l_0 and small l_2 distances. The fitness functions for untargeted and targeted attacks are employed to quantify the success of an attack with small perturbations, as illustrated below.

$$F(f, x + \vec{\delta}, y) = C_1 * \max\left(f_y(x + \vec{\delta}) - \underset{i \neq y}{\operatorname{argmax}} f_i(x + \vec{\delta}), 0\right) + \|\vec{\delta}\|_2 \quad (5)$$

$$F(f, x + \vec{\delta}, y_t) = C_1 * \max\left(\underset{i \neq y_t}{\operatorname{argmax}} f_i(x + \vec{\delta}) - f_{y_t}(x + \vec{\delta}), 0\right) + \|\vec{\delta}\|_2 \quad (6)$$

Where C_1 indicates a large positive number used to punish unsuccessful attacks.

3.2 Two-Stage Sparse Attack

The proposed Two-Stage Sparse Attack method (TSSA) aims to identify perturbed key pixels in the first stage using sparse attacks that allow unbounded or zero perturbation on each perturbed pixel. Optimization can be achieved by modifying a portion of perturbed pixels in the current solution along with the corresponding perturbation values. This approach differs from gradient-based methods as it does not depend on a continuous set for optimization. GA is a method of optimization based on natural selection and genetic mechanisms. It explores and optimizes solution space through crossover, mutation (introducing new changes while retaining some features), and selection. Therefore, GA with discrete encoding is introduced to search for the initial sparse perturbation vector.

In the second stage, the perturbation size of the initial perturbation vector is fine-tuned while maintaining adversarial. PSO simulates the foraging behavior of birds and searches for optimal solutions through collaboration and information sharing. Due to its rapid convergence capability, PSO efficiently reaches optimal solutions and adapts well to high-dimensional space. This is achieved by continuously adjusting particle velocities and positions like simulating a real gradient update direction. Consequently, PSO with continuous encoding is introduced to optimize the initial perturbation vector. The diagram of the optimization search is shown in Fig. 1.

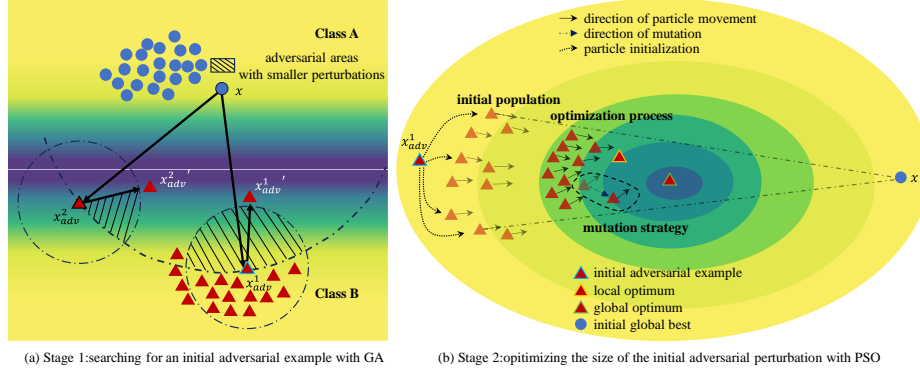


Fig. 1. Diagram of TSSA. (a) indicates the diagram of a classifier classifying two classes (color-coded; the darker the lower the confidence scores). It searches for an initial adversarial example with a higher adversarial class confidence score, which is in a better position for the next optimization. (b) describes the steps and strategies of searching for a better solution based on the initial perturbation vector with PSO.

Generation of the initial sparse perturbation vector using GA. The class confidence scores of DNN output can better reflect the relationship between the sample and the DNN decision boundary. We argue that when facing similar distortions, adversarial examples with higher adversarial class confidence scores have greater potential to find adversarial examples with smaller perturbations as shown in Fig. 1 (a). Therefore, the adversarial class confidence score of the perturbed sample is used to determine the initial sparse perturbation vector by adding unbounded or zero perturbations to finite pixels. The following steps are the same as SA-MOO [24] as shown below. The details of generating the initial sparse perturbation vector using GA under untargeted attacks are shown in Algorithm 1, while targeted attacks follow similar steps. The difference is that the confidence score threshold \mathcal{C} represents the difference between the true class confidence score and the adversarial class confidence score of the perturbed sample under untargeted attacks. However, in the case of targeted attacks, the adversarial class confidence score of the perturbed sample is represented.

Initial population. The initialization process involves randomly selecting k indices from $h \times m$ pixels and applying perturbations $(1, -1, 0)$ to the three channels with probabilities of $(\frac{1-pr_0}{2}, \frac{1-pr_0}{2}, pr_0)$, where pr_0 represents zero-sampling probability.

Crossover. The crossover operator aims to generate a superior offspring individual by combining the strengths of two high-performing individuals. This is accomplished by swapping pixel locations and their associated perturbation values between the two parent individuals.

Mutation. The primary objective of the mutation operator is to introduce a controlled level of randomness to enhance the diversity within the population. This is achieved

by randomly selecting a subset of perturbed pixels and resetting their perturbation values to zero. Subsequently, an equivalent number of unperturbed pixels are chosen and initialized with unbounded or zero perturbations. The adoption of a dynamic method, which adjusts the mutation rate as the attack process advances, has demonstrated superior efficiency compared to maintaining a constant mutation rate throughout the optimization process.

Evaluation. A perturbation vector with perturbed locations M and perturbed values Δ is reshaped to match the shape of input x and added into x to generate a perturbed sample. The fitness values of the candidate solution are evaluated by Equation (3) and Equation (4) under the untargeted and targeted attacks, respectively.

Selection. The purpose of the selection operator is to select outstanding individuals as parents for crossover and mutation, aiming to move closer to the optimum. Consequently, solutions with lower fitness values L or in cases where L is equal, solutions with smaller l_2 values will be prioritized.

Algorithm 1. Generation of the initial sparse perturbation vector using GA

```

1: Input: input  $x$ , ground truth label  $y$  or target label  $y_t$ , query budget  $N$ , sparsity
    $k$ , population size  $s_1$ , zero-sampling probability  $pr_0$ , confidence scores threshold
    $C$ , query budget threshold  $N_T$ , initial query budget  $Q = 0$ 
2:  $M \leftarrow k$  perturbed pixel indices
3:  $\Delta \leftarrow$  perturbation values
   //initial population
4:  $P, P' \leftarrow \{(M_1, \Delta_1), \dots, (M_{s_1}, \Delta_{s_1})\}$ 
   //fitness value evaluation
5:  $L, L' \leftarrow \{L(x, (M_1, \Delta_1)), \dots, L(x, (M_{s_1}, \Delta_{s_1}))\}$ 
6:  $Q = Q + s_1$ 
7: while  $Q \leq N$  do
8:   for  $i \leftarrow 0; i < s/2; i \leftarrow i + 1$  do
9:      $p_1, p_2 \leftarrow$  sorting  $L$  //selecting parents
10:     $p'_1, p'_2 \leftarrow$  Crossover( $p_1, p_2$ )
11:     $p''_1, p''_2 \leftarrow$  Mutation( $p'_1, p'_2$ )
12:     $P' = P' \cup p'_1 \cup p'_2$ 
13:     $L' = L' \cup L(p''_1) \cup L(p''_2)$ 
14:   $P = P'$ 
15:   $L = L'$ 
16:  optimal value  $O = \min_j (L_1, \dots, L_{2s})$ 
17:   $P(L), P'(L') \leftarrow$  fitness sorting  $P(L)$ 
18:   $Q = Q + s_1$ 
19:  if  $Q \leq N_T$  and  $O < C$  or  $Q > N_T$  and  $O < 0$  then
20:     $N_1 = N - Q$ 
21:  return  $(M_j, \Delta_j), N_1$ 

```

Generation of the final sparse perturbation vector using PSO. The modified pixels, identified as the critical feature region, are determined by the initial sparse perturbation vector that meets the confidence scores condition. Subsequently, the perturbation values of the initial sparse perturbation vector are further optimized to explore a more effective adversarial vector in this section. The details of generating the final sparse perturbation vector using PSO are shown in Algorithm 2 and Fig. 1 (b).

Algorithm 2. Generation of the final sparse perturbation vector using PSO

```

1: Input: the initial perturbation vector  $\vec{\delta} = \Delta = (\delta_0, \delta_1, \dots, \delta_{k-1})$ , population size
    $s_2$ , query budget  $N_1$ , the discarded perturbed pixels ratio  $pr_1$ , the retained per-
   turbations ratio  $pr_2$ , velocity threshold  $B$ , iterations between mutation intervals  $I$ 
//initialize particle swarm
2: for  $i \leftarrow 0 ; i < s_2 ; i \leftarrow i + 1$  do
3:   for  $j \leftarrow 0 ; j < k ; j \leftarrow j + 1$  do
4:     if  $random() < pr_1$  then
5:        $\delta_j^i = \delta_j \times pr_2$ 
6:      $p_{best}^i = \vec{\delta}^i$ 
7:  $g_{best} = [0, 0, \dots, 0]^k$ 
//Optimization
8: for  $n \leftarrow 1 ; n < \frac{N_1}{s_2} ; n \leftarrow n + 1$  do
9:   for  $i \leftarrow 0 ; i < s_2 ; i \leftarrow i + 1$  do
10:     $v^i = wv^i + C_1r_1(p_{best}^i - \vec{\delta}^i) + C_2r_2(g_{best} - \vec{\delta}^i)$  // update velocity
11:     $v^i = clip(v^i, -B, B)$ 
12:     $\delta^i = \delta^i + v^i$  // update position
13:     $x^i = clip(x + \vec{\delta}^i, 0, 1)$ 
14:    if  $F(x^i) < F(p_{best}^i)$  then
15:       $p_{best}^i = \vec{\delta}^i$ 
16:    if  $F(x^i) < F(g_{best})$  then
17:       $g_{best} = \vec{\delta}^i$ 
// Adaptive mutation strategy
18: if  $n \% I = 0$  then
19:   for  $i \leftarrow 0 ; i < s_2 ; i \leftarrow i + 1$  do
20:      $prob = (\frac{N_1}{s_2} - n) / \frac{N_1}{s_2}$ 
21:     if  $random() < prob$  then
22:       for  $j \leftarrow 0 ; j < k ; j \leftarrow j + 1$  do
23:         if  $random() < prob$  then
24:           if  $\delta_j^i = 0$  then
25:              $\delta_j^i = prob \times 0.05$ 
26:           else
27:              $\delta_j^i = prob \times 0.5 \times \delta_j^i$ 
28: return the best sparse perturbation vector  $g_{best}$ 

```

Initialize particle swarm. The perturbation vector has $3k$ dimensions, each corresponding to the magnitude of a perturbation at a pixel location obtained in the first stage. When optimizing for smaller perturbations, initialization of particles involves discarding part of the perturbations to improve solutions. Specifically, each particle's perturbed pixel retains pr_2 's perturbations with probability pr_1 . To guide the particle swarm towards small perturbations, the initial global best is set to an all-zero vector, considering the initial sparse and large perturbation vector.

Optimization. Due to the limitation of the number of queries to the target DNN, we optimize the solution with a relatively small population and multiple iteration rounds. The value of the inertia weight w is determined through a linearly decreasing function that considers the number of queries made to the target DNN as shown below.

$$w = w \times \left(1 - \frac{Q}{N_1}\right) \quad (7)$$

Where Q indicates the number of queries submitted to the target DNN and N_1 is the query budget. At the same time, to control the size of the perturbation, the speed each update is limited. The fitness values of the updated particles are evaluated to update new p_{best} and g_{best} by Equation (5) and (6) under the untargeted and targeted attacks, respectively.

Adaptive Mutation Strategy. The strategy of adaptive mutation, inspired by the dynamic mutation strategies in GA, aims to increase the diversity of particles, enabling them to escape the local optima and approach the global optimum. As the number of iterations increases, both the discarded perturbed pixels ratio and the retained perturbation values ratio gradually decrease. Meanwhile, a specific channel of perturbed pixels that has a perturbation size of zero is subjected to an additional perturbation that gradually decreases as the number of iterations increases, which helps explore better solutions.

4 Experiments

In this section, the proposed method TSSA is evaluated by experiments to verify its effectiveness. We first outline the experimental setup including dataset and model settings, parameters settings, and evaluation metrics in Section 4.1. To determine the effectiveness of each strategy, an ablation study is conducted in Section 4.2. Then, the hyperparameter selection is carried out in Section 4.3 to get better solutions. Finally, TSSA is compared with the state-of-the-art black-box sparse attack SA-MOO to demonstrate its superiority in Section 4.4.

4.1 Experimental Setup

Dataset and Model Settings. The proposed method TSSA is conducted on DNNs trained on CIFAR-10. 1000 correctly classified images of the test set are chosen to implement both untargeted and targeted attacks. For targeted attacks, a random target class is chosen, distinct from the ground truth label of each image. The max number of model queries is set to 1000 following the work of Phoenix et al. [24]. Two adversarial trained models AT0, AT1 and a conventionally trained model Standard are selected as attack models, consistent with SA-MOO [24]. At the same time, another two models are also chosen to validate the generalization of our method. The specific accuracy information of these models is shown in the following Table 1.

Table 1. Accuracy of models attacked.

Model	AT0	AT1	Standard	[9] ¹	Resnet20
Accuracy	0.8948	0.8750	0.9478	0.7877	0.9260

Parameters settings. We maintain the same parameters settings as Phoenix et al [24] during the search for the initial perturbation vector. Specifically, we keep the sparsity $k = 24$, population size $s_1 = 2$, and zero-sampling probability $pr_0 = 0.3$. The query budget threshold N_T is set to 300 and 500 under untargeted attacks and targeted attacks respectively. This is to make the initial perturbation vector meet the confidence condition and give more access times to the subsequent optimization. We set the population size $s_2 = 10$, velocity threshold $B = 0.8$, iterations between mutation intervals $I = 10$ and constant $C_1 = 1000$ in Equation (5) and (6) for the search of the final perturbation vector.

Evaluation Metrics. We assess the efficacy of algorithms in generating adversarial examples by utilizing all available queries. The attack success rate (ASR), the average adversarial perturbations l_2 with the limited perturbed pixels, as well as the average structural similarity index measure (SSIM) which measures the similarity between the original image and the corresponding adversarial example, are the main three metrics of evaluating the performance.

4.2 Ablation Study

To demonstrate the contribution of each strategy, an ablation study is launched on two models trained with conventional and adversarial methods respectively. 100 images classified correctly by the target DNNs are chosen to carry out the untargeted attacks and the targeted attacks. The following Table 2 describes the selection of relevant hyperparameters: the confidence score threshold C , the discarded perturbed pixels ratio pr_1 and the retained perturbations ratio pr_2 .

¹ The same DNN model as in the $C\&W$ attack.

Table 2. Selection of hyperparameters in ablation study.

Model	Untargeted attack			Targeted attack		
	pr_1	pr_2	C	pr_1	pr_2	C
[9]	0.2	0.6	-0.8	0.2	0.6	0.7
AT1	0.1	0.8	-0.5	0.1	0.8	0.5

Table 3. The effectiveness of strategies under untargeted attack

Model	Untargeted attack				
	Strategy	ASR	l_2	l_2^2	SSIM
[9]	<i>No</i>	0.99	2.9376	9.7027	0.9003
	<i>1st</i>	0.99	2.4673	6.9524	0.9205
	<i>2nd</i>	0.99	2.0415	5.1573	0.9403
	<i>3rd</i>	0.99	1.5435	3.0945	0.9573
AT1	<i>Nost</i>	0.87	3.3917	13.2197	0.8780
	<i>1st</i>	0.82	2.5203	7.6941	0.9123
	<i>2nd</i>	0.82	2.3213	6.5863	0.9231
	<i>3rd</i>	0.84	2.2800	6.5248	0.9277

Table 4. The effectiveness of strategies under targeted attack

Model	Targeted attack				
	Strategy	ASR	l_2	l_2^2	SSIM
[9]	<i>No</i>	0.87	3.4712	13.0160	0.8639
	<i>1st</i>	0.88	3.2716	11.7259	0.8759
	<i>2nd</i>	0.88	3.0410	10.6371	0.8874
	<i>3rd</i>	0.88	2.5590	7.8523	0.9042
AT1	<i>No</i>	0.40	3.9033	16.0045	0.8355
	<i>1st</i>	0.36	3.2270	11.7782	0.8712
	<i>2nd</i>	0.36	3.2128	11.7055	0.8716
	<i>3rd</i>	0.36	3.0522	10.5273	0.8769

No indicates the proposed algorithm with no strategies. *1st* shows the strategy of using confidence scores as the condition of searching for the initial adversarial vector in GA. Strategy *2nd* demonstrates setting the initial global best g_{best} of particle swarm to the all-zero vector. *3rd* expresses the proposed adaptive mutation strategy.

Experiments show that all three strategies are quite efficient in improving the size of adversarial perturbation for models trained with the conventional method as shown in Table 3 and Table 4. The implementation of strategy *1st* improves the quality of the initial perturbation vector. This allows for more opportunities to find smaller perturbations while still maintaining adversarial. The strategy *2nd* guides the initial phase of optimization to help the particle swarm find a solution with less disturbance. Implementing strategy *3rd* in the particle swarm optimization process increases diversity, allowing for escaping local optima and exploring better solutions.

Strategy *1st* has a greater impact on reducing perturbations in defense models than strategies *2nd* and *3rd*. However, strategies *2nd* and *3rd* have additional enhance-

ment effects when combined with strategy 1st, resulting in better perturbation vectors overall.

4.3 Hyperparameter Selection

The key parameters that affect the performance of our algorithm are pr_1 , pr_2 , C . To simplify the process of obtaining a better set of parameters, non-duplicate sampling is used for pr_1 and pr_2 . This results in multiple groups (pr_1, pr_2), with the values of pr_1 and pr_2 ranging from 0.1 to 0.7 with an interval of 0.1 for untargeted attacks (0.1 to 0.8 for targeted attacks). For each group (pr_1, pr_2), we conduct a grid search for C , with values ranging from -0.9 to -0.4 for untargeted attacks and from 0.4 to 0.9 for targeted attacks, with an interval of 0.1 on model [9]. For model AT1, the search is conducted with values ranging from -0.7 to -0.2 for untargeted attacks and from 0.3 to 0.8 for targeted attacks.

The results are shown in Fig. 2. ASR in each subgraph should ideally be similar to one another. However, due to the impact of random numbers, there may be minor variations when different hyperparameters are used. The hyperparameters mostly influence the perturbation size. It is noticeable that smaller perturbations are typically obtained when the initial adversarial examples have high confidence scores for the adversarial class. To achieve better attack results for the comparative experiments, we choose parameters (pr_1, pr_2, C) that achieve both a high attack success rate and a small perturbation. For the conventional models under untargeted and targeted attacks, the parameters (0.3,0.5,-0.9) and (0.7,0.7,0.9) are used. For the defense models under untargeted and targeted attacks, the parameters (0.7,0.7,-0.7) and (0.4,0.4,0.7) are chosen.

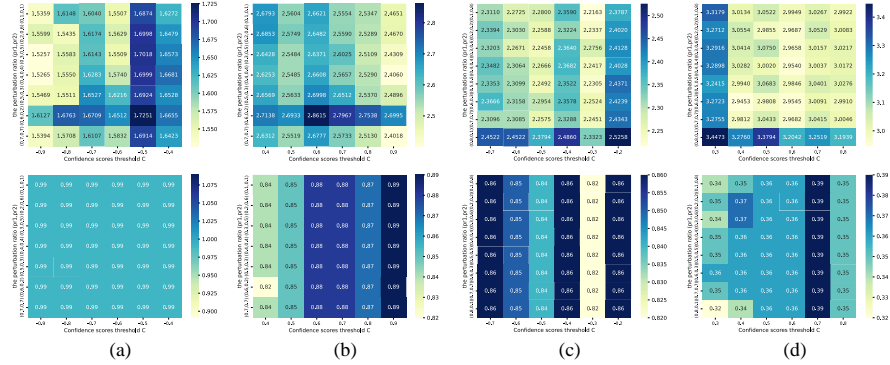


Fig. 2. (a) and (b) show the average l_2 norm (first row) and ASR (second row) for different parameter combinations, respectively, when launching untargeted and targeted attacks on the model [9]. (c) and (d) describe the attack situation of model AT1.

4.4 Comparison

The proposed method is compared with the advanced black-box sparse attack method SA-MOO [24] to demonstrate its advancement. The following experiments are carried out under the above optimal parameters. The SA-MOO [24] algorithm follows the experimental setup described in the original paper. Some adversarial images are represented in Fig. 3.

The experimental results under untargeted and targeted attacks are shown in Table 5 and Table 6 respectively. It is evident that adversarially trained models consistently exhibit lower attack success rates and larger adversarial perturbations compared to conventionally trained models across the two distinct attack methods. This observation further indicates the effectiveness of enhancing model robustness through adversarial training. The proposed method TSSA achieves similar attack success rates with SA-MOO in almost all DNNs. In adversarially and conventionally trained models, the improvement of adversarial perturbations using TSSA are almost greater than 0.5 and 0.3 under untargeted attacks and close to 0.5 in almost all models under targeted attacks. The decrease in disturbance is also manifested in the increase of SSIM. This implies that our method TSSA can provide a significantly more accurate assessment of model robustness. Furthermore, the difference in the mean magnitude of perturbations of the models shows some fluctuations compared to SA-MOO, but the relative size relationship remains the same.



Fig. 3. This illustrate displays adversarial images and their corresponding perturbations generated by two different algorithms, the proposed method TSSA and SA-MOO. (a) and (b) depict untargeted and targeted attacks on the Standard model, while (c) and (d) represent untargeted and targeted attacks on the defense model AT1.

Table 5. Comparison Results under untargeted attack

Model	SA-MOO				TSSA			
	ASR	l_2	l_2^2	SSIM	ASR	l_2	l_2^2	SSIM
AT0	0.736	2.9047	9.8199	0.9048	0.757	2.3368	6.7425	0.9208
AT1	0.817	2.8073	9.1856	0.9101	0.836	2.3039	6.6331	0.9255
Standard	0.997	1.7214	3.5227	0.9595	0.998	1.4525	2.5976	0.9647
[9]	0.997	1.8028	3.9198	0.9528	0.992	1.4980	2.9356	0.9587
Resnet20	1.000	1.5822	2.9593	0.9639	1.000	1.2797	2.0491	0.9705

Table 6. Comparison Results under targeted attack

Model	SA-MOO				TSSA			
	ASR	l_2	l_2^2	SSIM	ASR	l_2	l_2^2	SSIM
AT0	0.350	3.5021	13.2865	0.8661	0.347	3.0442	10.4608	0.8814
AT1	0.414	3.3927	12.4996	0.8703	0.402	2.8401	9.0950	0.8897
Standard	0.958	2.4743	6.8114	0.9277	0.937	2.0075	4.6524	0.9417
[9]	0.923	2.8472	8.9909	0.8999	0.916	2.5427	7.4834	0.9069
Resnet20	0.981	2.4692	6.8750	0.9277	0.974	2.0030	4.6998	0.9410

5 Conclusion

In this paper, we propose a two-stage black-box sparse attack method called TSSA. This method determines the perturbed pixels based on the confidence score of the adversarial class. Additionally, two new strategies, namely zeroing and adaptive mutation are implemented in the PSO algorithm to reduce perturbations. As a result, our method achieves a more imperceptible attack compared with the state-of-the-art method SA-MOO. This method also provides a useful tool to evaluate and strengthen the robustness of DNNs.

Acknowledgement. This work is supported by the National Natural Science Foundation of China under Grant No. 62371030 and the Beijing Municipal Natural Science Foundation under Grant No.4222008.

References

1. Fang X., Bai H., Guo Z., Shen B., Hoi S., Xu Z.: DART: Domain-Adversarial Residual-Transfer networks for unsupervised cross-domain image classification. *Neural Networks*, vol. 127, 182–192 (2020).
2. Hinton, G., et al.: Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine* 29(6), 82–97 (2012).

3. Wang, H., et al.: CosFace: Large Margin Cosine Loss for Deep Face Recognition. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5265-5274 (2018).
4. Paschali, M., Conjeti, S., Navarro, F., Navab, N.: Generalizability vs. robustness: investigating medical imaging networks using adversarial examples. In: Medical Image Computing and Computer Assisted Intervention–MICCAI 2018, pp. 493–501 (2018).
5. Eykholt, K., et al.: Robust physical-world attacks on deep learning visual classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1625–1634 (2018).
6. Bai, T., Luo, J., Zhao, J., Wen, B., Wang, Q. Recent Advances in Adversarial Training for Adversarial Robustness. In: Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, {IJCAI-21}, pp. 4312–4321(2021).
7. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014).
8. Moosavi-Dezfooli, S.M., Fawzi, A., Frossard, P.: Deepfool: a simple and accurate method to fool deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2574–2582 (2016).
9. Carlini, N., Wagner, D.: Towards evaluating the robustness of neural networks. In: 2017 IEEE Symposium on Security and Privacy, pp. 39–57. IEEE (2017).
10. Kurakin, A., Goodfellow, I.J., Bengio, S.: Adversarial examples in the physical world. In: Artificial intelligence safety and security, pp. 99–112 (2018).
11. Dong, Y., et al.: Boosting adversarial attacks with momentum. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 9185–9193 (2018).
12. Chen, J., Gu, Q.: Rays: A ray searching method for hard-label adversarial attack. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1739–1747 (2020).
13. Zhang, Q., Zhang, C., Li, C., Song, J., Gao, L.: Practical no-box adversarial attacks with training-free hybrid image transformation. arXiv preprint arXiv:2203.04607 (2022).
14. Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A.: The limitations of deep learning in adversarial settings. In: 2016 IEEE European Symposium on Security and Privacy (EuroS&P), pp. 372–387. IEEE (2016).
15. Wiyatno R., Xu, A.: Maximal Jacobian-based Saliency Map Attack. arXiv preprint arXiv:1808.07945 (2018).
16. Modas, A., Moosavi-Dezfooli, S. M., Frossard, P.: SparseFool: A Few Pixels Make a Big Difference. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 9079–9088 (2019).
17. Fan, Y., et al.: Sparse Adversarial Attack via Perturbation Factorization. In: Computer Vision-ECCV 2020, pp. 35–50 (2020).
18. Dong, X., et al.: GreedyFool: distortion-aware sparse adversarial attack. In: Proceedings of the 34th International Conference on Neural Information Processing Systems, vol. 33, pp. 11226–11236 (2020).
19. Zhu, M., Chen, T., Wang, Z.: Sparse and Imperceptible Adversarial Attack via a Homotopy Algorithm. In: Proceedings of the 38th International Conference on Machine Learning, vol. 139, pp. 12868–12877 (2021).
20. Tian, Y., et al.: Imperceptible and Sparse Adversarial Attacks via a Dual-Population-Based Constrained Evolutionary Algorithm. IEEE Transactions on Artificial Intelligence 4(2), 268–281 (2023).

21. Narodytska, N., Kasiviswanathan, S.: Simple Black-Box Adversarial Attacks on Deep Neural Networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 1310–1318 (2017).
22. Su, J., Vargas, D.V., Sakurai, K.: One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation* 23(5), 828–841 (2019).
23. Croce, F., Andriushchenko, M., Singh, N., Flammarion, N., Hein, M. Sparse-RS: A Versatile Framework for Query-Efficient Sparse Black-Box Adversarial Attacks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 6437–6445 (2022).
24. Williams, P.N., Li, K.: Black-Box Sparse Adversarial Attack via Multi-Objective Optimisation CVPR Proceedings. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 12291–12301 (2023).