We would like to thank all reviewers for their valuable comments. We appreciate that the novelty and effectiveness of our method are well recognized by R2, R3, R5. All the minor grammar errors will be fixed in the final version.

R2:*Dataset specific GAN for distortion map* : Thanks, really good question. Inspired by it, we find that it is unnecessary to train GAN for each dataset. A GAN trained on ImageNet (GAN-ImageNet) also works on other datasets. If we generate the distortion map with GAN-ImageNet for CIFAR10 images by resizing, the mean perturbed pixel number is 6.01 and the median is still 4, almost the same as the result in our paper. We will add such analysis in the final version.

R2:l_1*-norm problem*: Sorry, this is our negligence that we only consider the traditional l_∞-norm method and SparseFool in this part, we will improve it in the final version. But here we want to emphasize that to the best of our knowledge, we are the first to quantitatively prove the importance of direction for sparse attack(in the ablation study part).

R2:*Comparing with PGD$_0$ l_0+l_∞*: In the main experiments, since we consider both sparsity and invisibility, we think it is fairer to compare with the invisible version of the previous method (SparseFool with ϵ constrain and PGD$_0$ with σ-map). Here we report the result of PGD$_0$-l_∞ with 200 pixels budget and $\epsilon = 10, 100$. Its fooling rate is 40.62% and 80.14% respectively, still lower than our GF. More results will be added to the final version.

R2:κ *for σ-PGD$_0$*: In Tab.1 and Tab.2, we already set $\kappa = \epsilon$ to control the max value of the perturbation for σ-PGD$_0$.

R2:*Why PGD$_0$ performs worse than it in [11]*: All the results are obtained by running the official code released by [11]. There are three key possible reasons that lead to the difference: 1) Data difference, different from traditional dense attack, sparse attack performance is more sensitive to the input (some images only need perturb 1 pixel while some need 100 pixels). [11] only uses 1000 test images, while we test the whole CIFAR10 test set. We asked for the advice from the author of [11] and we both think this possibly leads to the performance difference. 2) The metric difference, we found that [11] treat clean images classified incorrectly by the model as the successful attack samples. But we ignore these images when computing the fooling rate (it will be a bit lower for the same results), this also influences the result slightly. 3). Parameters difference, as there are some parameters of PGD$_0$ not reported in [11], we use the default parameters of its official code, this may also influence the result.

R2:*Sparsity comparison with CornerSearch[11]*: CornerSearch(CS) is a black-box method by traversing all the pixels, so it is very slow. In contrast, since our method is a white-box method, there does not exist a proper or fair comparison for success rate and speed. For CIFAR10 with $\epsilon = 255$, the mean pixel and time used by CS are 3.49 and 6.29sec, while our GF is 5.98 and 0.114sec. CS performs better than our GF a bit but significantly slower. For ImageNet with $\epsilon = 255$, the mean pixel and time used by CS are 104.3 and 644.95sec, but our GF is only 62.13 and 5.25sec.

R2:*Invisibility Comparison with σ-CornerSearch[11]*: For invisibility, we need 12000 ImageNet images for the invisibility evaluation, due to the short rebuttal time and σ-CS is relatively slow, we failed to generate enough samples. We will add this result to the final version.

R3:*Confusing notations*: Thanks for your suggestion, we will reorganize the logic of the method part and add more necessary diagrams in the final version. The g_t is the gradient of x_t^{adv} at the t iteration. $\mathbf{m}$ is a binary mask denoting whether a pixel is selected or not. ϱ is the distortion map generated by GAN and $\mathbf{p}$ is calculated from ϱ for the balance between invisibility and sparsity, visual results about ϱ and $\mathbf{p}$ are shown in the supplementary materials.

R3:*Lack of related work*: Sparse adversarial attack is a new direction and related works are relatively few, so we only introduced them in the Introduction part due to the page limit. More related works will be added in the final version.

R3:*Attack other models*: On ImageNet($\epsilon = 255$), our GF fool VGG16, ResNet50 and DenseNet161 with 21.98, 25.83, 34.69 pixels on average respectively. For SF, it needs 85.40, 103.61, 137.94 pixels respectively, nearly $4\times$ than us.

R3,R4:*Black-box transferability*: On ImageNet($\epsilon = 255$), we transfer from DenseNet161 to Vgg16 and ResNet50. For SparseFool[26], its fooling rate is 26.76% and 15.38%, while our GF is significantly better with 40.33% and 30.67%.

R4:*Similar to DeepFool and GAN-based attack method [1,2]?*: ***No, we cannot agree on the comment***. Our method is totally different from DeepFool and the cited methods from the hypothesis and task perspective: 1) DeepFool's hypothesis is that, if the input is perturbed with the direction toward the nearest hyperplane in each iteration, the classifier can be fooled with the smallest dense perturbation under $l_{1,2}$-norm. *But for our GreedyFool, the hypothesis is that the pixel that has largest value of the gradient in each iteration influences the prediction most, if we perturb it, we can fool the classifier with the smallest perturbation pixel number under l_0-norm.* 2) The task of [1] is generating adversarial samples by GAN, and the task of [2] is how to generate adversarial samples to fool the generative model. In contrast, for our GAN-based distortion map part, *it instead acts as the guidance to select the pixel with the minimal modification visibility and we train it by adding a global perturbation noise to it in an adversarial way*.

R4:*Comparison to One Pixel-Attack*: One Pixel Attack(1-PA) is a black box attack with high computation cost and needs predefined pixels budget. To attack ResNet18 on ImageNet with 50 pixels budget, the success rate of 1-PA is 60.87% with 120sec per image. While the success rate of our GF is 88.75% with only 0.6sec per image.

R5:*Comparison to other distortion map*: We have already shown the visual result and corresponding analysis in the supplementary materials. Here we further report the quantitative result by replacing the GAN-based distortion map with the σ-map and generate adversarial samples with $\epsilon = 10$. The median perturbation number and SRM detection rate of the σ-GF is 301.50 and 57.20%, which is worse than our GF which is 222.50 and 54.00% respectively.