# Supplementary Materials for Boosting Adversarial Transferability by Achieving Flat Local Maxima

**Anonymous Author(s)**
Affiliation
Address
`email`

## A Detailed Proof of Corollary 1

In this work, we introduce a penalized gradient norm to the original loss function, which helps the adversarial examples to achieve a flat maximum. Then we randomly sample an example $x'$ in the neighborhood of the adversarial example $x^{adv}$ and simplify the objective function as follows:

$$\max_{x^{adv}\in\mathcal{B}_\epsilon(x)} \mathcal{L}(x^{adv},y;\theta) = \left[J(x',y;\theta) - \lambda\cdot\|\nabla_{x'}J(x',y;\theta)\|_2\right], \quad \text{s.t.} \quad x'\in\mathcal{B}_\zeta(x^{adv}). \tag{1}$$

Gradient-based attacks require calculating the gradient of the objective function during practical optimization, thus the gradient of the current loss function (1) can be expressed as:

$$\begin{aligned}\nabla_{x^{adv}}\mathcal{L}(x^{adv},y;\theta) &= \nabla_{x'}J(x',y;\theta) - \lambda\cdot\nabla_{x'}(\|\nabla_{x'}J(x',y;\theta)\|_2)\\ &= \nabla_{x'}J(x',y;\theta) - \lambda\cdot\nabla^2_{x'}J(x',y;\theta)\cdot\frac{\nabla_{x'}J(x',y;\theta)}{\|\nabla_{x'}J(x',y;\theta)\|_2}.\end{aligned} \tag{2}$$

In practice, it is computationally expensive to directly optimize Eq. (2), since we need to calculate the Hessian matrix. In this work, we approximate the second-order Hessian matrix using the finite difference method to accelerate the attack process. Specifically, local Taylor expansion would be employed to approximate the operation results between the Hessian matrix and the gradient vector.

### A.1 Proof of Theorem 1

*Proof.* According to the Taylor expansion, we have

$$\nabla_x J(x+\Delta x,y;\theta) = \nabla_x J(x,y;\theta) + \nabla^2_x J(x,y;\theta)\Delta x + O(\|\Delta x\|^2), \tag{3}$$

where $\Delta x = \alpha\cdot v$, $\alpha$ is a small step size, $v$ is a normalized gradient direction vector. Here, we denote $v = -\frac{\nabla_x J(x,y;\theta)}{\|\nabla_x J(x,y;\theta)\|_2}$.

Therefore, the second-order Hessian matrix can be approximated by the first-order gradient as follows:

$$\nabla^2_x J(x,y;\theta) \approx \frac{\nabla_x J(x+\alpha\cdot v,y;\theta) - \nabla_x J(x,y;\theta)}{\alpha\cdot v}. \tag{4}$$

□

### A.2 Proof of Corollary 1

*Proof.* From Eqs. (2) and (4), the gradient of the loss function $\mathcal{L}(\cdot)$ can be expressed as:

$$\begin{aligned}\nabla_{x^{adv}}\mathcal{L}(x^{adv},y;\theta) &= \nabla_{x'}J(x',y;\theta) - \lambda\cdot\nabla^2_{x'}J(x',y;\theta)\cdot\frac{\nabla_{x'}J(x',y;\theta)}{\|\nabla_{x'}J(x',y;\theta)\|_2} \\ &\approx \nabla_{x'}J(x',y;\theta) - \lambda\cdot\frac{\nabla_{x'}J(x'+\alpha\cdot v,y;\theta)-\nabla_{x'}J(x',y;\theta)}{\alpha\cdot v}\cdot(-v) \\ &\approx \nabla_{x'}J(x',y;\theta) + \lambda\cdot\frac{\nabla_{x'}J(x'+\alpha\cdot v,y;\theta)-\nabla_{x'}J(x',y;\theta)}{\alpha} \\ &\approx (1-\frac{\lambda}{\alpha})\cdot\nabla_{x'}J(x',y;\theta)+\frac{\lambda}{\alpha}\cdot\nabla_{x'}J(x'+\alpha\cdot v,y;\theta).\end{aligned} \tag{5}$$

We introduce a balanced coefficient $\delta$ and denote it as $\delta = \frac{\lambda}{\alpha}$. Hence, the gradient of the objective function (1) at the $t$-th iteration can be approximated as:

$$\nabla_{x_t^{adv}}\mathcal{L}(x_t^{adv},y;\theta) \approx (1-\delta)\cdot\nabla_{x_t'}J(x_t',y;\theta)+\delta\cdot\nabla_{x_t'}J(x_t'+\alpha\cdot v,y;\theta). \tag{6}$$

□

## B Visualization of Loss Surfaces

**Implementation details.** Given that the adversarial example $x^{adv}$ typically has a large number of dimensions, visualizing the loss function against all dimensions becomes infeasible. To this end, we randomly select two directions, denoted as $r_1$ and $r_2$, from a Gaussian distribution with the same dimension as $x^{adv}$. Next, we calculate the loss change by varying the magnitudes of $k_1$ and $k_2$, representing the scaling factors applied to $r_1$ and $r_2$, respectively, which enables us to visualize the loss function using a two-dimensional plot. This approach provides a slice of the loss function, allowing us to analyze its behavior and understand the impact of perturbations along different directions.

**Visualization of loss surfaces for more adversarial examples.** We visualize five randomly selected images in the ImageNet-compatible dataset. The adversarial examples are generated by various gradient-based attack methods on Inc-v3. As shown in Fig. 1, we can observe that our method generates visually similar adversaries as other attacks. However, our method demonstrates the capability to guide adversarial examples towards larger and smoother flat regions. This observation substantiates the effectiveness of our PGN method in generating adversarial examples that reside within flat regions, thereby shedding light on the potential role of flat local maxima in generating transferable adversarial examples.

## C Combined with Gradient-based Attacks

Our PGN attack method can also be combined with various gradient-based attacks. The core of our method involves updating gradients by interpolating the first-order gradients from two samples to approximately minimize the gradient norm. In contrast, conventional gradient-based methods typically utilize a single example for gradient updates. To evaluate the efficacy of our strategy, we incorporate this interpolation approach into previous gradient-based methods, such as I-FGSM (BIM), MI-FGSM, NI-FGSM, VMI-FGSM, EMI-FGSM, and RAP. To simplify the experimental setup, we omitted random sampling and directly substituted the gradient update process of these methods with our proposed strategy.

The experimental results are presented in Table 1. Notably, when our gradient update strategy is integrated, there is a remarkable improvement in the adversarial transferability of the gradient-based attack methods within the black-box setting. For example, RAP alone achieves an average success rate of $67.51\%$ across the seven models. However, when combined with our PGN method, the average success rate rises to 75.30%, exhibiting a significant improvement of $7.79\%$. This outcome underscores the robust scalability of our approach, as it seamlessly integrates with existing methodologies to further amplify the success rate of transfer-based attacks.

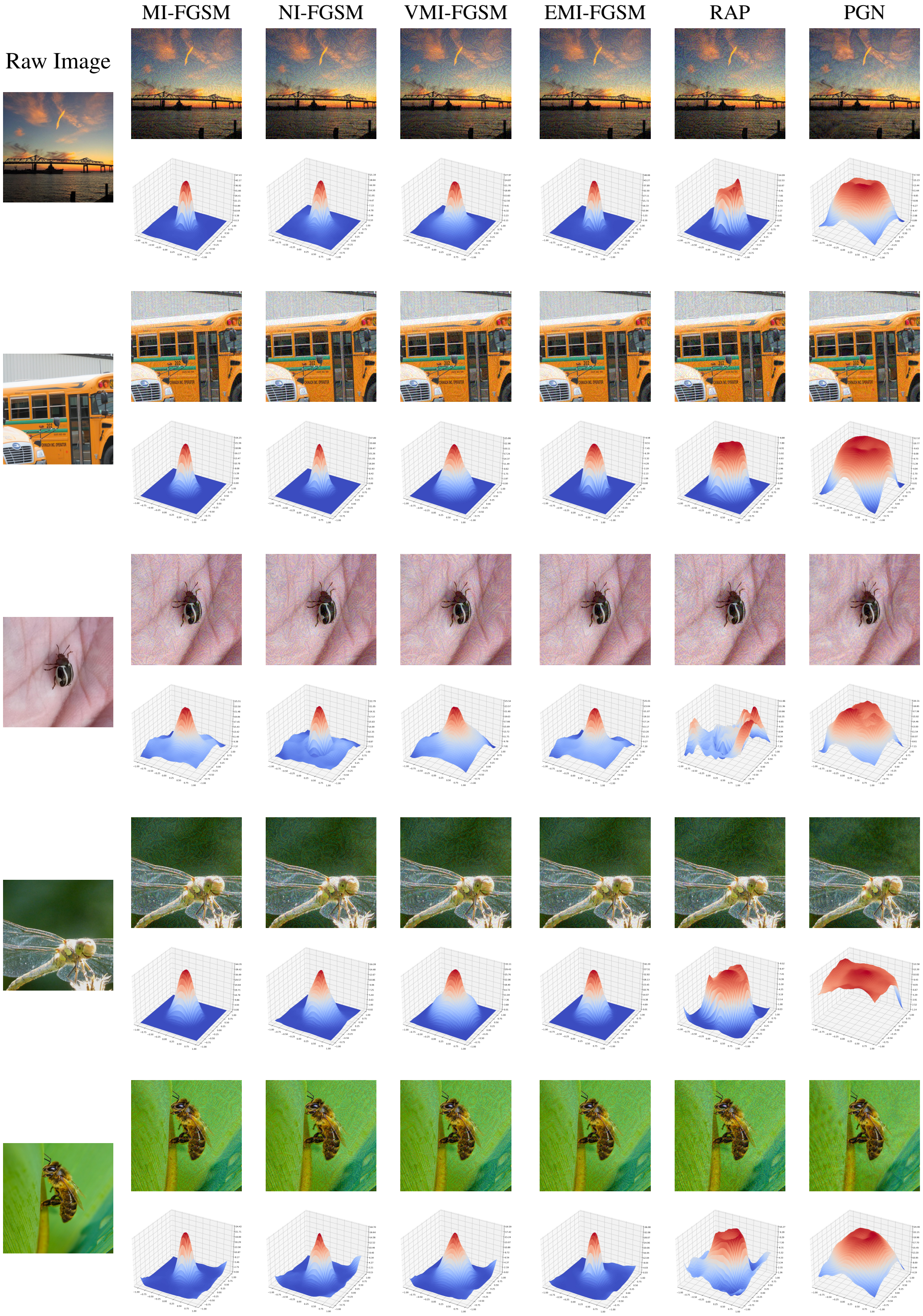

Figure 1: Visualization of adversarial examples with their corresponding loss surfaces along two random directions. Here, we randomly sampled five images and generated the adversarial examples on Inc-v3. The loss surfaces are also calculated on Inc-v3.

Table 1: Untargeted attack success rates (%) of our PGN method, when it is integrated with I-FGSM (BIM), MI-FGSM, NI-FGSM, VMI-FGSM, EMI-FGSM, and RAP, respectively. The adversarial examples are generated on Inc-v3. * indicates the white-box model.

| Attack | Inc-v3 | Inc-v4 | IncRes-v2 | Res-101 | Inc-v3$_{ens3}$ | Inc-v3$_{ens4}$ | IncRes-v2$_{ens}$ | Avg. |
|---|---|---|---|---|---|---|---|---|
| BIM | **100.0*** | 26.8 | 19.8 | 38.7 | 11.7 | 11.7 | **5.9** | 31.22 |
| PGN-BIM | **100.0*** | **37.9** | **28.6** | **44.7** | **13.3** | **13.0** | 5.1 | **35.45** |
| MI | **100.0*** | 49.7 | 47.1 | 61.9 | 22.3 | 23.4 | 10.9 | 46.72 |
| PGN-MI | **100.0*** | **65.6** | **61.4** | **71.1** | **24.7** | **25.7** | **12.2** | **53.50** |
| NI | **100.0*** | 61.9 | 60.0 | 69.6 | 22.9 | 23.2 | 12.1 | 51.85 |
| PGN-NI | **100.0*** | **73.0** | **69.3** | **74.7** | **25.8** | **26.2** | **12.4** | **56.57** |
| VMI | **100.0*** | 74.8 | 70.2 | 75.8 | 41.1 | 40.7 | 24.4 | 62.50 |
| PGN-VMI | **100.0*** | **81.9** | **78.4** | **80.3** | **48.6** | **49.0** | **30.2** | **68.28** |
| EMI | **100.0*** | 81.1 | 76.9 | 80.4 | 33.4 | 32.7 | 16.4 | 62.45 |
| PGN-EMI | **100.0*** | **86.1** | **83.6** | **83.9** | **46.8** | **44.5** | **29.4** | **69.72** |
| RAP | 99.9 | 82.4 | 79.3 | 81.4 | 48.1 | 44.3 | 27.6 | 67.51 |
| PGN-RAP | **100.0*** | **90.3** | **88.6** | **86.7** | **58.2** | **54.8** | **37.3** | **75.30** |

## D The Number of Sampled Examples $N$

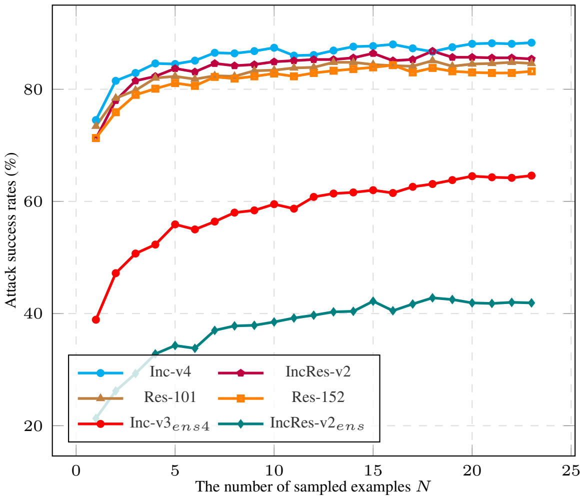

Figure 2: Untargeted attack success rates (%) on six black-box models with the different number of sampled samples $N$. The adversarial examples are generated by PGN on Inc-v3.

In this study, we employ random sampling of multiple examples and calculate the average gradients of these examples to mitigate the variance resulting from random sampling during the iterative process. To investigate the influence of the number of sampled examples, denoted as $N$, we conduct ablation experiments to analyze this parameter. As illustrated in Figure 2, when $N = 1$, our method demonstrates the lowest level of transferability. However, as we increase the value of $N$, the transferability exhibits rapid improvement until $N = 12$, after which it gradually converges for normally trained models. Notably, when $N > 12$, a slight performance improvement can still be achieved by increasing the number of sampled examples in our PGN method. To strike a balance between transferability and computational overhead, we set $N = 20$ in our work. This observation further substantiates that sampling random examples from the vicinity of the adversarial example effectively facilitates neighborhood exploration. Consequently, it stabilizes the gradient update process and encourages the discovery of flatter regions by the adversarial example.

## E Attack Defense Models

In this subsection, besides normally trained models and adversarially trained models, we further validate the effectiveness of our methods on other defenses, including Bit-Red [7], ComDefend [2], JPEG [1], HGD [4], R&P [6], and NIPS-r3 [5]. The adversarial examples are generated on an ensemble of Inc-v3, Inc-v4, and IncRes-v2, and the weight for each model is $1/3$.

The experimental results are displayed in Table 2. In the context of ensemble models, it is evident that our algorithm can considerably enhance existing attack methods. For instance, VMI, EMI, and RAP achieve average success rates of $57.65\%$, $65.65\%$, and $76.43\%$ respectively, against the six defense models. In contrast, our proposed PGN method achieves an average success rate of $85.25\%$, surpassing them by $27.6\%$, $19.6\%$, and $8.82\%$ respectively. This notable improvement demonstrates the remarkable effectiveness of our proposed method against adversarially trained models as well as other defense models. Consequently, it poses a more substantial threat to advanced defense models. These findings further validate that the discovery of adversarial examples within flat regions can significantly enhance the transferability of adversarial attacks.

Table 2: Untargeted attack success rates (%) on six defense models. The adversarial examples are crafted on the ensemble models, *i.e.* Inc-v3, Inc-v4 and IncRes-v2.

| Attack | HGD | R&P | NIPs-r3 | Bit-Red | JPEG | ComDefend | AVG. |
|---|---|---|---|---|---|---|---|
| MI | 24.8 | 22.2 | 29.9 | 23.8 | 49.5 | 54.9 | 34.18 |
| NI | 22.3 | 23.1 | 29.3 | 23.9 | 50.8 | 56.9 | 34.38 |
| VMI | 54.3 | 50.6 | 58.1 | 39.0 | 71.5 | 72.4 | 57.65 |
| EMI | 64.8 | 60.1 | 69.1 | 47.6 | 74.1 | 78.2 | 65.65 |
| RAP | 72.3 | 73.1 | 81.0 | 54.6 | 88.1 | 89.5 | 76.43 |
| PGN | **82.5** | **83.6** | **88.3** | **72.1** | **91.3** | **93.7** | **85.25** |

## F Attack Success Rates on CIFAR-10

To further illustrate the effectiveness of our PGN method on different datasets, we conduct experiments on CIFAR-10 [3]. we set the hyperparameters as follows: maximum perturbation $\epsilon = 8/255$, number of iterations $T = 10$, and step size $\alpha = 1/255$. We compare our PGN method with various gradient-based attacks, including MI-FGSM, NI-FGSM, VMI-FGSM, EMI-FGSM, and RAP. The adversarial examples are generated on VGG-16, ResNet-50, and DenseNet-121 models, respectively. The results, presented in Table 3, clearly demonstrate that our PGN method can enhance the attack transferability on the CIFAR-10 dataset. This outcome supports our motivation that adversarial examples located in flat local regions tend to exhibit better transferability across diverse models. Moreover, our attack method showcases superior performance when applied to other datasets, reinforcing its versatility and effectiveness.

Table 3: Untargeted attack success rates (%) on the CIFAR-10 dataset for the attack methods in the single model setting. The adversarial examples are crafted on VGG-16, ResNet-50 (Res-50), and DenseNet-121, respectively.

| Attack | MobileNet | VGG-19 | GoogLeNet | Inc-v3 | DenseNet-121 | DenseNet-169 | Res-34 | Res-50 |
|---|---|---|---|---|---|---|---|---|
| MI | 52.18 | 57.56 | 47.29 | 52.74 | 40.96 | 42.40 | 41.72 | 41.93 |
| NI | 56.13 | 61.36 | 49.19 | 54.87 | 37.61 | 39.74 | 38.74 | 38.90 |
| VMI | 66.14 | 68.05 | 60.89 | 65.63 | 55.62 | 57.21 | 55.35 | 56.46 |
| EMI | 70.69 | 74.36 | 66.78 | 70.56 | 59.98 | 63.04 | 60.47 | 61.83 |
| RAP | 77.98 | 78.43 | 73.41 | 77.86 | 68.30 | 69.74 | 65.48 | 66.27 |
| PGN | **85.97** | **86.73** | **82.82** | **85.59** | **72.48** | **74.66** | **71.62** | **72.95** |

(a) Untargeted attack success rates (%) for the adversarial examples crafted on VGG-16.

| Attack | MobileNet | VGG16 | VGG19 | GoogLeNet | Inc-v3 | DenseNet-121 | DenseNet-169 | Res-34 |
|---|---|---|---|---|---|---|---|---|
| MI | 70.42 | 67.37 | 65.8 | 63.06 | 69.02 | 72.39 | 73.34 | 67.78 |
| NI | 71.97 | 65.57 | 63.76 | 63.28 | 69.13 | 71.03 | 72.78 | 65.02 |
| VMI | 77.60 | 74.27 | 73.26 | 71.11 | 75.99 | 76.80 | 77.68 | 73.54 |
| EMI | 80.11 | 78.66 | 77.43 | 76.34 | 78.12 | 79.68 | 80.12 | 77.24 |
| RAP | 86.92 | 84.24 | 83.46 | 80.68 | 81.75 | 83.54 | 84.98 | 81.36 |
| PGN | **90.88** | **88.68** | **88.07** | **85.79** | **89.53** | **89.93** | **90.91** | **87.19** |

(b) Untargeted attack success rates (%) for the adversarial examples crafted on Res-50.

| Attack | MobileNet | VGG16 | VGG19 | GoogLeNet | Inc-v3 | DenseNet-169 | Res-34 | Res-50 |
|---|---|---|---|---|---|---|---|---|
| MI | 63.47 | 60.64 | 60.08 | 57.39 | 63.35 | 71.09 | 61.99 | 67.37 |
| NI | 66.86 | 61.80 | 60.85 | 60.30 | 66.54 | 75.92 | 61.57 | 69.25 |
| VMI | 71.28 | 68.49 | 68.01 | 65.40 | 70.62 | 75.51 | 68.42 | 72.50 |
| EMI | 74.36 | 75.66 | 73.54 | 70.41 | 75.23 | 78.94 | 73.64 | 77.45 |
| RAP | 79.98 | 80.22 | 78.68 | 76.39 | 80.55 | 84.25 | 78.65 | 80.03 |
| PGN | **86.73** | **85.12** | **84.66** | **81.82** | **86.26** | **88.35** | **83.30** | **86.21** |

(c) Untargeted attack success rates (%) for the adversarial examples crafted on DenseNet-121.

# References

[1] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten. Countering adversarial images using input transformations. In *International Conference on Learning Representations*, 2018.

[2] Xiaojun Jia, Xingxing Wei, Xiaochun Cao, and Hassan Foroosh. Comdefend: An efficient image compression model to defend adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6084–6092, 2019.

[3] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

[4] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1778–1787, 2018.

[5] Muzammal Naseer, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Fatih Porikli. A self-supervised approach for adversarial robustness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 262–271, 2020.

[6] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. In *International Conference on Learning Representations*, 2018.

[7] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *Network and Distributed System Security Symposium*, 2018.