Enhancing Adversarial Transferability by Balancing Exploration and Exploitation with Gradient-Guided Sampling

Supplementary Material

1. Robustness against Defense Methods

To further study the robustness of our method, we selected seven defense strategies for testing, i.e., RP [29], Bit-red [31], JPEG [4], FD [14], NRP [17], DP (DiffPure) [18] and MD (MimicDiffusion) [21]. Table S1 presents the ASR of various methods under these defenses. Table S1 indicates that GGS demonstrates superior performance in overcoming defenses except for DP and NRP, increasing the average ASR by 1.7% compared to state-of-the-art methods, validating its robustness and suggesting its greater applicability and effectiveness in a wide range of scenarios.

Moreover, when comparing the inner-iteration methods using gradient averaging (GRA [32], PGN [6], GGS) with other methods, the minimum difference in average attack success rate is 12% (ANDA:54.2% to GRA: 66.74%), demonstrating the outstanding effectiveness of inner-iteration sampling methods in overcoming defense mechanisms.

Table S1. The average untargeted ASR (%) on nine models under seven defense strategies (RP [29], Bit-red [31], JPEG [4], FD [14], NRP [17], DP [18] and MD [21]) of different methods (MI[1], NI[13], VMI[26], RAP [19], GRA[32], PGN[6], ANDA[5], GI [25], GGS), with the adversarial examples generated on ResNet50.

Method	RP	Bit-red	JPEG	FD	NRP	DP	MD	Avg.
MI	37.8	36.2	33.5	41.0	30.8	19.2	37.5	33.71
NI	39.9	38.0	34.3	42.6	31.3	19.6	38.1	34.83
VMI	54.0	52.9	49.7	53.5	40.8	28.0	50.1	47.00
RAP	53.7	51.8	49.0	36.6	43.0	26.1	50.5	44.39
GRA	74.1	72.9	72.8	72.6	59.9	46.2	68.7	66.74
PGN	77.0	76.3	76.0	75.2	**61.3**	**47.0**	72.4	69.31
ANDA	66.0	62.8	57.9	63.2	41.2	31.1	57.2	54.20
GI	44.7	43.0	40.6	34.0	47.53	21.8	42.7	39.19
GGS	**82.2**	**81.3**	**79.0**	**78.3**	60.6	42.1	**73.9**	**71.06**

2. Attack on Cloud Models

To further evaluate the capabilities of different methods in real-world scenarios, we selected four prominent cloud service providers: Alibaba, Tencent, Baidu, and HUAWEI, and conducted attack tests on their general image labeling services. As shown in Table S2, our GGS achieved the best attack performance across all the four providers, with the final average result being 6% lower than current best result, demonstrating the effectiveness of our GGS in practical applications.

Table S2. Classification success rate (%) of adversarial examples generated by all nine methods (MI-FGSM [1], NI-FGSM [13], VMI-FGSM [26], RAP [19], GRA [32], PGN [6], ANDA [5], GI-FGSM [25], GGS) in the general label classification interfaces provided by four prominent cloud service providers (Alibaba, Tencent, Baidu, HUAWEI). The row "clean" indicates the classification accuracy corresponding to the original, unperturbed samples.

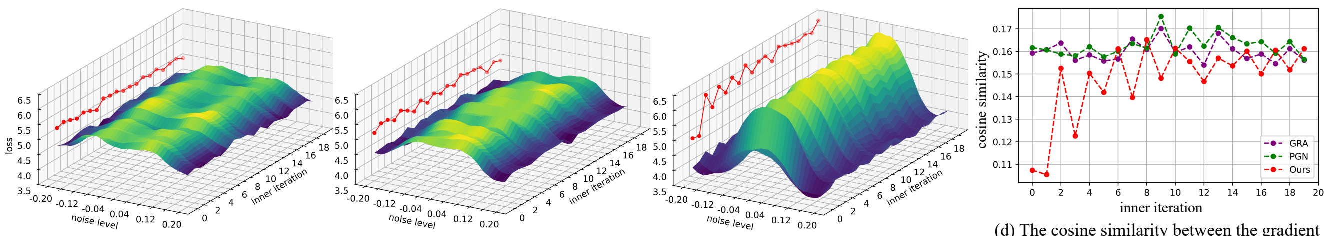
Method	Alibaba	Tencent	Baidu	HUAWEI	Avg. ↓
clean	81.3	47.0	48.5	64.3	60.28
MI CVPR'18 [1]	44.8	24.7	29.7	25.0	31.05
NI ICLR'20 [13]	40.3	26.4	26.4	49.8	35.73
VMI CVPR'21 [26]	35.4	22.8	23.6	47.0	32.20
RAP NeurIPS'22 [19]	22.6	20.6	18.8	47.2	27.30
GRA ICCV'23 [32]	25.3	17.5	14.6	34.2	22.90
PGN NeurIPS'23 [6]	22.2	17.5	14.5	33.8	22.00
ANDA CVPR'24 [5]	24.0	17.9	17.3	38.1	24.33
GI ESWA'24 [25]	37.8	27.2	28.6	48.9	35.63
GGS	**15.7**	**13.9**	**10.9**	**21.7**	**15.55**

3. Loss Surfaces against Inner-Iteration

To further compare the differences in the inner-iteration processes between our method and other inner-iteration sampling methods, we visualized the loss surface against different inner-iteration steps of PGN [6], GRA [32], and GGS, as well as the similarity between the gradients during the inner-iterations and the final gradient averaging, as shown in Fig. S1.

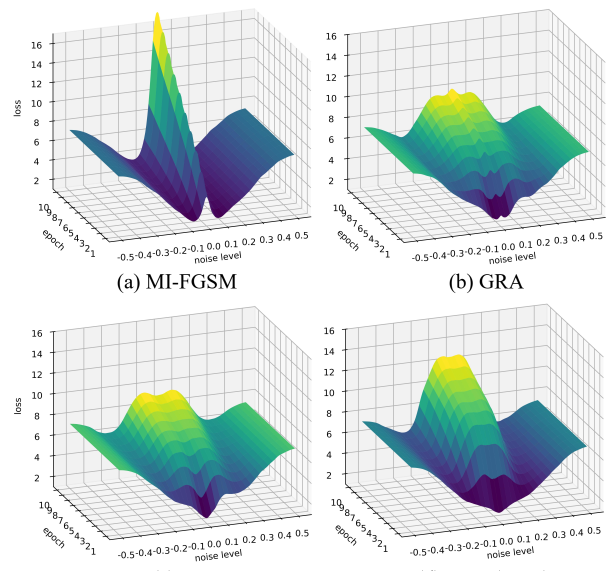
In Fig. S1(d), it shows that the cosine similarity between the generated gradients and the final averaged gradient specific to our GGS gradually increases and stabilizes, as the number of inner-iteration increases, indicating that our method tends to favor the more stable gradient results from the later-stages of inner-interations, whereas PGN and GRA do not exhibit significant differential treatment toward different gradients.

From Fig. S1(a), (b), and (c), it can be seen that, our method enhances exploitation capability by improving sampling efficiency, while maintaining the original exploration capability from inner-iteration sampling, balancing exploration and exploitation. This ensures that adversarial samples improve sampling efficiency without sacrificing cross-model generalization, thereby increasing the attack potential of the final samples.



(a) GRA (b) PGN (c) Ours (GGS) (d) The cosine similarity between the gradient and the sum of the gradients in iterations

Figure S1. Loss surfaces of adversarial examples generated by GRA[32], PGN[6], and our GGS, with increasing inner-iterations on Resnet50. The red line on the right side of part (a)∼(c) represents the maximum value of the loss surface in each iteration. For (d), it represents the cosine similarity between the gradient $\tilde{g}_i$ generated in each inner-iteration and the average of gradients $\sum_{i=1}^{N} \tilde{g}_i / N$.



(a) MI-FGSM (b) GRA (c) PGN (d) Ours (GGS)

Figure S2. The classification loss surfaces of adversarial examples generated by the methods (MI-FGSM [1], GRA [32], PGN [6], GGS) against the number of epochs through four different methods. These adversarial examples are generated and tested on the surrogate model Res50 [9].

4. Loss Surface against Outer-Iteration

Loss Surface: Further, we visualized the changes in the loss surface during the generation of adversarial examples using four different methods (MI-FGSM [1], GRA [32], PGN [6], and Our GGS) on the Res50 [9] model, presenting the results in a two-dimensional fashion in Fig. S2. It can be observed that, our method maintains the flatness of the loss landscape while simultaneously increasing the loss value, as the number of epochs increases. Compared to other methods, we achieve a better balance between the magnitude of the loss values and the flatness of the loss landscape.

5. More Visualizations of Loss Surfaces

To provide a more comprehensive comparison of the loss surfaces and the connections between different methods, we plotted the loss surfaces of all the nine comparison methods along with our GGS and its related variants MGS and RS methods in Fig. S3. According to the loss surfaces shown in Fig. S3, previous methods can be generally categorized into two types: those with larger local maxima and those with flatter loss surfaces. However, our GGS method achieves an excellent balance between these two characteristics by maintaining local flatness while possessing relatively large local maxima. This balance is consequently helpful to achieve strong adversarial transferability.

Additionally, to better distinguish the differences among various inner-iteration methods, we summarized these methods based on the purpose of inner and outer iterations as well as the relationship between them. This summary is presented in Table S3.

6. Attack on Ensemble Models

As shown in Table S4, our method consistently achieves higher transfer attack success rates (ASR) across multiple model architectures, compared to other existing methods, under multiple model settings, demonstrating its strong generalization capabilities. Specifically, for targeted and untargeted attacks in a multiple model ensemble setting, our method increases the average ASR by 5%. For untargeted attacks, it achieved a score of 95.99%, approaching a nearly complete success in attacking all the samples, revealing its strong attack capability in complex model ensemble environments.

7. Comparison of Adversarial Perturbation

To compare the differences in noise intensities produced by recent advanced methods, we conducted both qualitative and quantitative analyses on the adversarial examples generated by GRA [32], PGN [6], ANDA [5], and our GGS.

Algorithm Comparison in terms of Perturbation

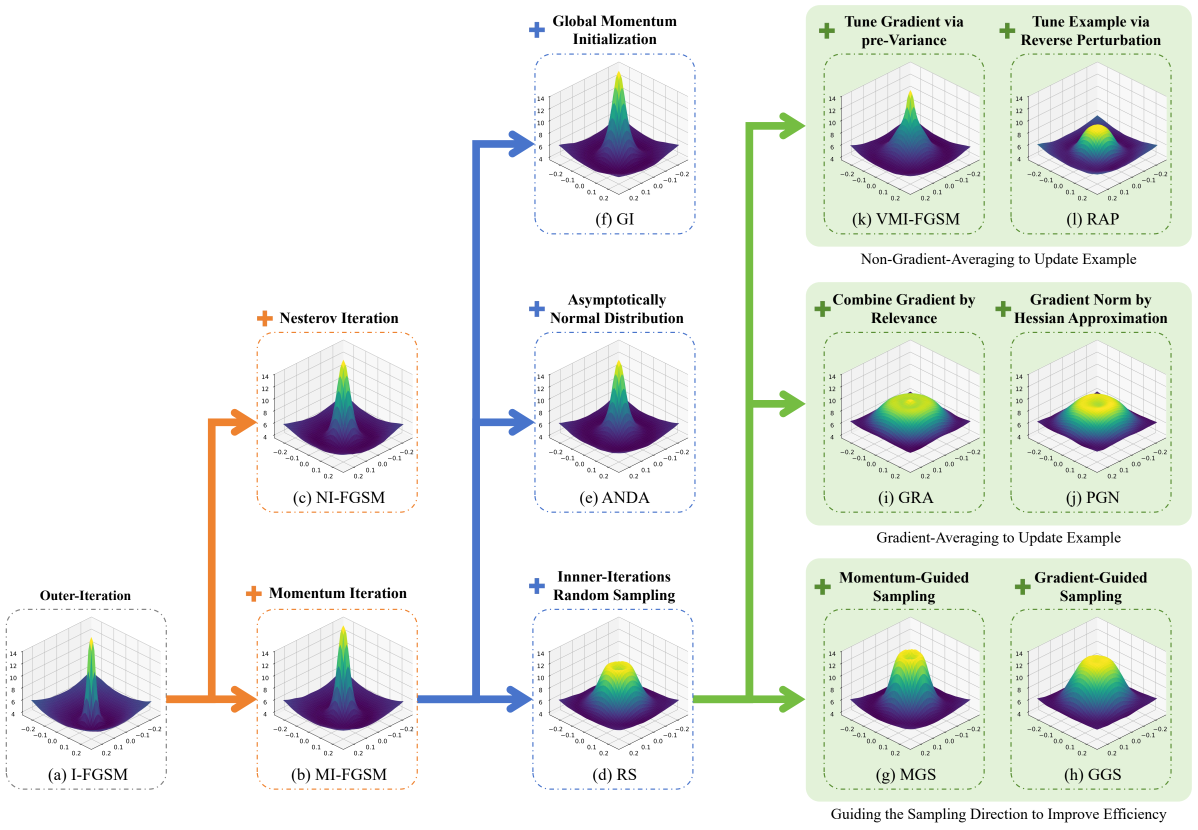


Figure S3. The relationships and comparison among different methods in terms of their average loss surfaces of the 32 randomly selected adversarial examples, corresponding to all the 12 methods (I-FGSM [12], MI-FGSM [1], NI-FGSM [13], VMI-FGSM [26], RAP [19], GRA [32], PGN [6], ANDA [5], GI-FGSM [25], our GGS and its related variants RS and MGS). The x and y axes on loss surface represent the weights of two random directions, respectively, and the z-axis indicates the magnitude of the loss value obtained by testing the adversarial examples, after incorporating the combination of the corresponding two random directions like PGN [6].

Table S3. The comparison of various inner-iteration methods (VMI [26], RAP [19], PGN [6], GRA [32], and our GGS). Out. and In. denote the numbers of outer-iterations and inner-iterations, respectively. pre-grad denote the gradient from previous inner-iteration.

	Out.	In.	Inner-Iteration Purpose	Relationship	Outer-Iteration Purpose
VMI	10	20	gradient variance	Nested	tune gradient via variance from previous iteration
RAP	400	8	reverse perturbation	Conditional Nested	tune example via reverse perturbation
PGN	10	20	gradient norm by Hessian approximation	Nested	update example by momentum
GRA	10	20	averaged gradient	Nested	combine grad. by relevance & update delta by decay
GGS	10	20	averaged gradient lookahead by pre-grad	Nested	update example by momentum

Metrics: We used four metrics, e.g. PSNR, SSIM [8, 28], L2, and LF [16], to evaluate the imperceptibility of adversarial perturbations relative to the clean samples. As shown in Table S5, all methods exhibit relatively similar distortion metric values. The primary reason for this is that all methods utilize a fixed step size coupled with the Fast Gradient Sign Method (FGSM [7]), during each adversarial example updating, which effectively limits the noise intensity within a relatively narrow range.

Visualization of adversarial examples: We further visualized the adversarial examples generated by different methods and provided a local zoom-in comparison. As shown in Fig. S4, it can be observed that the adversarial noises generated by the four methods do not exhibit significant differences at the scale of the entire image. However, upon closer inspection, PGN [6], GRA [32] and our GGS generate perturbations with lower regularity in shape (fewer continuous groove-like perturbations) . This may be a manifestation of high transferability in the noise patterns, and we hope that future work can uncover the relationship between

Table S4. The average untargeted and targeted ASR (%) on all the nine models, with adversarial examples generated on multiple models (Res50, Inc-v3 and ViT-B). Each data pair (u/w) corresponds to the performances under (untargeted/targeted) attacks. The best and second best results are labeled in **bold** and <u>underline</u>, respectively.

	Attack	**Res50**	Dense121	**Inc-v3**	IncRes-v2	**ViT-B**	PiT-B	Inc-v3$_{ens3}$	Inc-v3$_{ens4}$	IncRes-v2$_{ens}$	Avg.
Res50 Inc-v3 ViT-B	MI	99.4 / 66.7	73.4 / 0.5	100.0 / 91.2	64.3 / 0.3	98.3 / 86.8	51.4 / 0.1	44.6 / 0.0	46.7 / 0.0	36.9 / 0.0	68.33 / 27.29
	NI	99.7 / 72.4	79.9 / 1.0	100.0 / 88.1	71.8 / 0.9	98.5 / 83.6	51.9 / 0.3	48.6 / 0.1	51.6 / 0.1	40.4 / 0.0	71.38 / <u>27.39</u>
	VMI	99.0 / 40.7	84.0 / 2.3	100.0 / 72.4	81.0 / 1.8	97.5 / 60.9	66.8 / 1.1	63.8 / 0.2	65.5 / 0.5	57.4 / 0.4	79.44 / 20.30
	RAP	99.7 / 15.8	94.4 / 1.1	100.0 / 24.6	90.2 / 0.7	99.2 / 30.5	75.2 / 0.2	64.3 / 0.1	66.8 / 0.3	54.0 / 0.0	82.64 / 8.14
	GRA	97.3 / 26.1	93.3 / 4.7	100.0 / 58.6	94.1 / 5.4	95.7 / 48.3	83.0 / 4.2	85.4 / 1.4	85.6 / 2.3	81.7 / 1.8	<u>90.68</u> / 16.98
	PGN	95.8 / 13.6	94.5 / 4.4	100.0 / 44.7	93.9 / 5.2	93.0 / 25.1	82.5 / 3.3	85.1 / 1.3	86.1 / 1.7	82.1 / 1.5	90.33 / 11.20
	ANDA	99.5 / 45.7	96.5 / 5.5	100.0 / 74.6	93.6 / 5.2	99.3 / 67.4	87.2 / 2.8	79.8 / 1.0	81.3 / 1.1	74.4 / 0.6	90.18 / 22.66
	GI	99.7 / 77.8	86.2 / 1.8	100.0 / 93.5	79.1 / 1.7	99.4 / 91.1	64.8 / 0.7	57.4 / 0.3	58.9 / 0.0	49.7 / 0.0	77.24 / 29.66
	GGS	99.3 / 54.1	98.0 / 23.8	100.0 / 78.8	98.2 / 24.9	99.2 / 71.4	94.6 / 19.7	91.9 / 10.6	93.3 / 10.3	89.4 / 8.1	**95.99 / 33.52**

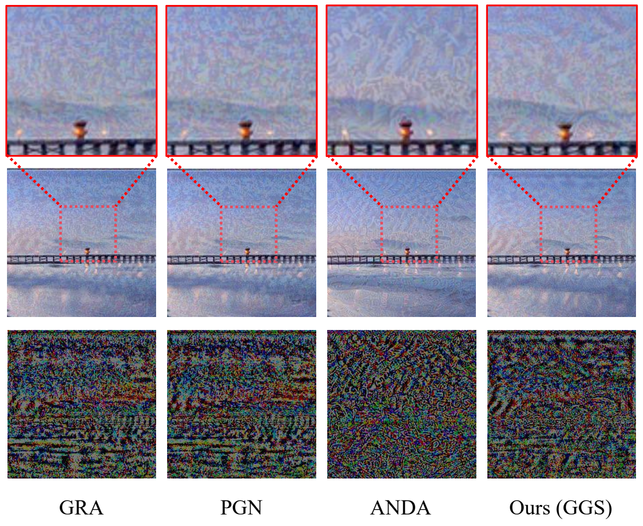


Figure S4. Visualization comparison of adversarial examples generated by four advanced methods (GRA [32], PGN [6], ANDA [5], GGS) using Res50 [9] as the surrogate model. The first row shows the locally magnified regions, the second row displays the corresponding adversarial examples, and the third row presents the adversarial noises.

Table S5. Perceptibility metrics of adversarial examples generated by different methods on Res50 [9], including Attack Success Rate (ASR) as a measure of untargeted attack capability (average ASR across nine models), and the metrics of the imperceptibility, such as conventional L2 norm, Structural Similarity (SSIM [8, 28]) , Peak Signal-to-Noise Ratio (PSNR), and average distortion of Low-Frequency Component (LF [16]).

Method	ASR	PSNR ↑	SSIM ↑	L2 ↓	LF ↓
GRA ICCV'23	73.41	<u>28.31</u>	<u>0.711</u>	<u>17.90</u>	12.93
PGN NeurIPS'23	<u>76.53</u>	28.07	0.701	18.39	13.24
ANDA CVPR'24	64.67	**28.60**	**0.711**	**17.32**	**12.12**
GGS	**82.08**	28.16	0.697	18.22	<u>12.48</u>

noise morphology and transferability.

8. GGS vs. NCS with different random seeds

Table S6 shows the mean and avg. range from 10 independent runs with random seeds. GGS surpassed NCS [20] using Res50 and ViT-B as surrogate models.

Table S6. Average untargeted ASR (%) of NCS and GGS.

	Attack	**Res50**	Dense121	**Inc-v3**	IncRes-v2	**ViT-B**	PiT-B	Avg.
Res50	NCS	96.16	87.00	80.20	73.46	53.58	67.74	76.36 ± 0.27
	GGS	99.30	95.78	89.46	85.92	60.14	80.18	**85.13** ± 0.18
Inc-v3	NCS	69.38	84.52	99.76	91.20	32.54	45.10	**70.42** ± 0.30
	GGS	68.42	86.72	100.00	95.72	27.64	39.16	69.61 ± 0.74
ViT-B	NCS	72.18	82.00	79.34	72.72	98.56	85.32	81.69 ± 0.15
	GGS	81.26	90.12	86.68	81.84	100.0	92.50	**88.73** ± 0.50

9. Inner-Iterations vs. Efficiency analysis

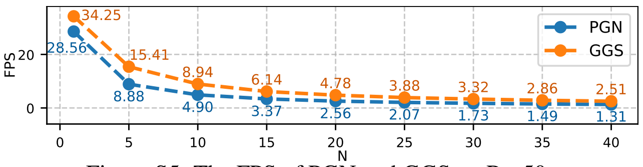


Figure S5. The FPS of PGN and GGS on Res50.

Fig. S5 shows PGN and Our method's time consumption scales with inner iterations, while GGS maintaining 2.5 FPS compare with PGN's 1.3 FPS at 40 iterations. N=20 was set to ensure usability and fair comparison with other methods.

10. Inference Speeds and GPU Memory Usage

To compare the differences in efficiency and overhead of various methods for generating adversarial examples, we conducted a detailed evaluation of the inference speed and GPU memory usage of all nine methods, as shown in Table S7. Since the MI-FGSM [1], NI-FGSM [13] and GI-FGSM [25] methods do not involve an inner-iteration process, we focused primarily on the remaining six methods that include

Table S7. Comparison of inference speeds in terms of Frames Per Second (FPS) and GPU Memory Usage (GB) of different methods. The official default batch size (bs) of ANDA [5] is set to 1, RAP is set to 16 due to memory constraints, while for other methods, the default batch size (bs) is set to 64 for Res50 [9] and Inc-v3 [22], and 32 for ViT-B [3]. The MI-FGSM [1], NI-FGSM [13] and GI-FGSM [25] do not include an inner-iteration process, thus resulting in a faster inference speed.

Method	FPS ↑			GB ↓		
	Res50	Inc-v3	ViT-B	Res50	Inc-v3	ViT-B
MI CVPR'18 [1]	43.5	52.6	27.0	8.37	5.96	5.47
NI ICLR'20 [13]	47.6	41.7	26.3	8.36	5.96	5.47
GI ESWA'24 [25]	31.3	27.8	18.2	8.38	6.02	5.47
VMI CVPR'21 [26]	4.7	6.4	<u>1.8</u>	<u>8.60</u>	<u>6.14</u>	5.56
RAP NeurIPS'22 [19]	0.25	0.15	0.13	21.3	20.5	21.5
GRA ICCV'23 [32]	<u>4.7</u>	<u>6.4</u>	1.8	8.76	6.35	5.55
PGN NeurIPS'23 [6]	2.5	3.6	1.0	8.93	6.26	5.55
ANDA CVPR'24 [5]	3.6	2.4	1.4	**3.23**	**2.42**	**4.57**
GGS	**4.9**	**6.6**	**1.9**	8.78	6.24	<u>5.53</u>

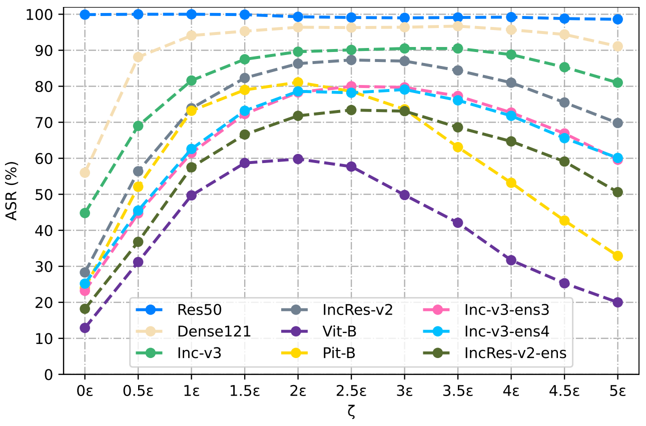


Figure S6. Untargeted attack success rate (ASR, %) on different models (Res50 [9], Dense121 [11], Inc-v3 [22], IncRes-v2 [23], ViT-B [3], PiT-B [10], Inc-v3$_{\text{ens3}}$ [24], Inc-v3$_{\text{ens4}}$ [24], IncRes-v2$_{\text{ens}}$ [24]) with different values of ζ, with Res50 as a surrogate model. The maximum perturbation $\epsilon = 16/255$.

an inner-iteration process for a fair comparison. It can be observed that our GGS method exhibits the fastest inference speed, and the GPU memory usage does not obviously increase. Notably, due to algorithmic limitations, ANDA [5] can be only configured with a batch size of 1, resulting in a reduced GPU memory usage.

11. Sensitivity Analysis of Hyperparameters

Sensitivity Analysis of ζ. In Fig. S6, we analyze the impact of the upper bound neighborhood size, determined by parameter ζ, on the attack success rate (ASR) in black-box settings. The experiment employs uniform sampling to reduce bias from uneven sample distribution. As ζ gradually increases to $2.0 \times \epsilon$, the transferability performance of normally trained models is improved and reaches its peak, while the transferability of adversarially trained models continues to rise. However, when ζ exceeds $3.5 \times \epsilon$, the adversarial transferability of the eight black-box models begins to decline. To achieve a balance between normally and adversarially trained models in transferability, we select $\zeta = 2.0 \times \epsilon$ for the experiments.

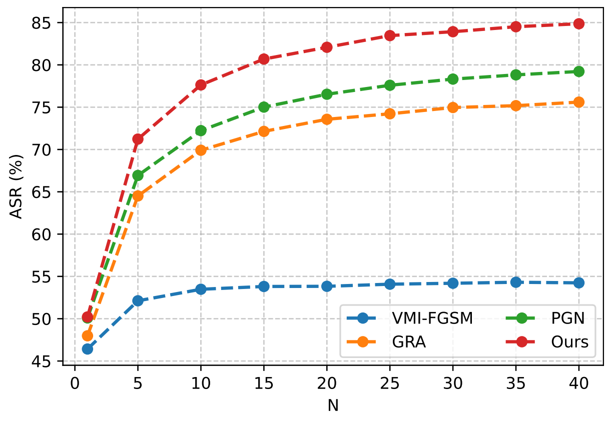


Figure S7. The average attack success rate on nine models with different N (inner-iterations numbers) across different methods (VMI-FGSM [26], GRA [32], PGN [6], our GGS).

Sensitivity Analysis of N. We further conducted experimental evaluations to assess the impact of varying the number of inner-iterations N on different methods. The results are presented in Fig. S7. It can be observed that, the attack success rate for all the methods has been largely improved when the number of inner-iterations is within 15. As the number of inner-iterations increases, the effectiveness of GRA [32], PGN [6], and our GGS method can be further enhanced. Considering both time overhead and effectiveness, the selection of 20 inner-iterations (N=20) provides a balanced benefit.

12. Detailed Hyperparameter Settings

For all methods, we set the maximum perturbation of the parameter $\epsilon = 16/255$. For all methods, except RAP, we set the number of outer-iteration is $T = 10$, the step size is $\alpha = \epsilon/T$. For VMI-FGSM [26], we configure the number of sampled examples as $N = 20$ and set the upper bound of neighborhood size to $\beta = 1.5 \times \epsilon$; For RAP, the step size is $\alpha = 2.0/255$, with the number of iterations $K = 400$, an inner-iteration number $T = 8$, and a late-start $K_{LS} = 100$. The size of neighborhoods is $\epsilon_n = 16/255$. For GRA [32], the number of sampled examples is $N = 20$, the sample quantity is $m = 20$, the upper bound factor of sample range is $\beta = 3.5$ and the attenuation factor is $\eta = 0.94$; For PGN [6], the number of sampled examples is $N = 20$, the balanced coefficient is $\delta = 0.5$, and the upper bound factor of sample range is $\zeta = 3.0 \times \epsilon$. For ANDA [5], we strictly adhere to the hyperparameter settings specified in its official code repository. For GI-FGSM [25], the pre-convergence iterations $P = 5$, and the global search factor $S = 10$.

13. Detailed Results when our GGS is Combined with Different Methods

Due to space limit in the main text, we only present the average attack success rates of our GGS combined with different methods. More detailed results, including targeted and untargeted attack across all nine models, are provided in Table S8.

14. Details of the attack on MLLMs

Specifically, (1) 4o and mini represent gpt-4o-2024-08-06 and gpt-4o-mini-2024-07-18, respectively; (2) pro and flash represent gemini-1.5-pro-002 and gemini-1.5-flash-002, respectively; (3) sonnet represent claude-3-5-sonnet-20240620. For the system prompt, to better study the image content understanding capabilities of MLLMs while simplifying the experimental process, we converted the multi-classification problem into a binary classification one. Specifically, we input both the image label name and the image itself into the MLLMs, prompting the MLLMs to determine whether the image content belongs to the corresponding label, and provide a confidence level for its judgment. For all the five MLLMs, we uniformly applied the following system prompts:

```
You are a strict image classification expert system.
     Execute the following instructions:

1. Determine whether the main content of the image
     belongs to the given label through visual analysis
     only
2. Strictly return results according to the following
     JSON schema:
   {"match": 1 or 0, "confidence": integer between 1-10}
3. Judgment rules:
   - "match": 1 if completely matching the label, 0 if
       not matching
   - "confidence": Objectively assess the credibility
       of your judgment based on image clarity, content
       features, etc.

Key constraints:
    - Prohibited from outputting any text other than JSON
    - Ensure output can be directly parsed by standard
        JSON parsers
    - Output must be valid JSON format
    - Please do not return empty content, you must give
        match and confidence, even if the picture is
        blank

Example output:
    {"match": 1, "confidence": 8}
```

References

[1] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018. 1, 2, 3, 4, 5

[2] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4312–4321, 2019. 7

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 5

[4] Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M Roy. A study of the effect of jpg compression on adversarial images. *arXiv preprint arXiv:1608.00853*, 2016. 1

[5] Zhengwei Fang, Rui Wang, Tao Huang, and Liping Jing. Strong transferable adversarial attacks via ensembled asymptotically normal distribution learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24841–24850, 2024. 1, 2, 3, 4, 5, 6

[6] Zhijin Ge, Hongying Liu, Wang Xiaosen, Fanhua Shang, and Yuanyuan Liu. Boosting adversarial transferability by achieving flat local maxima. *Advances in Neural Information Processing Systems*, 36:70141–70161, 2023. 1, 2, 3, 4, 5

[7] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations, ICLR*, 2015. 3

[8] Muhammad Zaid Hameed and Andras Gyorgy. Perceptually constrained adversarial attacks. *arXiv preprint arXiv:2102.07140*, 2021. 3, 4

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2, 4, 5

[10] Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11936–11945, 2021. 5

[11] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 5

Table S8. The untargeted and targeted ASR (%) on all nine models with the adversarial examples generated on ResNet50, when combined our GGS with input transformation-based methods (DIM[30], TIM[2], SIM[13], Admix[27], SSM[15]). Each data pair (u/w) corresponds to the performances under (original method / variant combined with our GGS).

Method	Res50	Dense121	Inc-v3	IncRes-v2	Vit-B	PiT-B	Inc-v3$_{\text{ens3}}$	Inc-v3$_{\text{ens4}}$	IncRes-v2$_{\text{ens}}$	Avg.
					untargeted attack					
DIM / +ours	97.6 / 98.6	72.8 / 96.3	60.3 / 94.3	50.8 / 93.7	25.6 / 75.3	41.5 / 88.2	39.3 / 89.6	40.9 / 88.6	32.9 / 86.6	51.30 / **90.13**
TIM / +ours	99.9 / 99.3	60.0 / 96.0	49.9 / 90.7	33.8 / 87.5	15.7 / 63.6	26.6 / 79.9	28.1 / 84.3	30.1 / 83.1	23.3 / 80.1	40.82 / **84.94**
SIM / +ours	100.0 / 99.6	71.8 / 98.8	56.2 / 95.4	40.6 / 94.2	22.0 / 71.9	36.2 / 89.3	32.3 / 89.5	35.2 / 88.4	26.9 / 84.0	46.80 / **90.12**
Admix / +ours	100.0 / 99.7	82.7 / 99.2	65.4 / 95.9	51.4 / 95.0	28.5 / 73.6	46.8 / 89.0	41.9 / 89.5	43.1 / 88.3	33.2 / 85.5	54.78 / **85.50**
SSM / +ours	97.9 / 98.9	89.7 / 96.1	81.4 / 92.2	77.6 / 88.8	50.1 / 47.4	67.3 / 70.2	70.0 / 80.5	69.3 / 79.3	64.7 / 73.3	74.22 / **80.74**
GRA / +ours	96.9 / 99.2	88.6 / 96.8	81.8 / 91.0	75.8 / 85.7	45.3 / 44.0	62.6 / 66.4	71.5 / 77.5	70.9 / 76.6	67.3 / 71.0	73.41 / **78.69**
PGN / +ours	98.6 / 99.8	91.3 / 97.4	85.0 / 91.3	78.5 / 88.8	49.7 / 55.2	67.8 / 78.4	74.9 / 81.9	72.9 / 80.7	70.1 / 75.6	76.53 / **83.23**
					targeted attack					
DIM / +ours	76.1 / 83.3	1.1 / 26.7	0.1 / 9.9	0.2 / 13.3	0.0 / 6.7	0.1 / 11.1	0.0 / 7.7	0.0 / 7.5	0.1 / 8.1	8.63 / **19.37**
TIM / +ours	98.9 / 89.7	0.2 / 28.9	0.0 / 7.9	0.1 / 9.7	0.0 / 4.1	0.0 / 8.6	0.0 / 6.1	0.0 / 5.5	0.1 / 5.8	11.03 / **18.48**
SIM / +ours	99.1 / 95.8	0.9 / 52.5	0.1 / 17.6	0.0 / 21.8	0.0 / 10.9	0.1 / 22.1	0.0 / 11.6	0.0 / 13.4	0.1 / 10.3	11.14 / **28.44**
Admix / +ours	97.3 / 96.6	3.8 / 57.6	0.0 / 19.0	0.1 / 23.5	0.0 / 11.9	0.2 / 24.9	0.0 / 15.2	0.1 / 14.0	0.0 / 11.8	11.28 / **30.50**
SSM / +ours	63.5 / 69.1	6.2 / 18.1	1.2 / 5.3	1.8 / 5.6	0.3 / 1.3	0.6 / 4.0	0.5 / 3.5	0.8 / 3.0	0.7 / 3.0	8.40 / **12.54**
GRA / +ours	67.7 / 83.2	3.9 / 19.9	0.9 / 4.5	1.7 / 5.6	0.2 / 0.9	0.6 / 3.0	0.7 / 1.4	1.1 / 2.4	0.7 / 1.9	8.61 / **13.64**
PGN / +ours	49.5 / 73.3	4.7 / 19.2	1.4 / 5.1	1.7 / 5.7	0.4 / 1.5	1.0 / 3.2	0.6 / 2.2	1.1 / 2.5	1.2 / 2.4	6.84 / **12.79**

[12] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC, 2018. 3

[13] Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E Hopcroft. Nesterov accelerated gradient and scale invariance for adversarial attacks. In *International Conference on Learning Representations, ICLR*, 2020. 1, 3, 4, 5, 7

[14] Zihao Liu, Qi Liu, Tao Liu, Nuo Xu, Xue Lin, Yanzhi Wang, and Wujie Wen. Feature distillation: Dnn-oriented jpeg compression against adversarial examples. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 860–868. IEEE, 2019. 1

[15] Yuyang Long, Qilong Zhang, Boheng Zeng, Lianli Gao, Xianglong Liu, Jian Zhang, and Jingkuan Song. Frequency domain model augmentation for adversarial attack. In *European conference on computer vision*, pages 549–566. Springer, 2022. 7

[16] Cheng Luo, Qinliang Lin, Weicheng Xie, Bizhu Wu, Jinheng Xie, and Linlin Shen. Frequency-driven imperceptible adversarial attack on semantic similarity. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15315–15324, 2022. 3, 4

[17] 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. 1

[18] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification. *arXiv preprint arXiv:2205.07460*, 2022. 1

[19] Zeyu Qin, Yanbo Fan, Yi Liu, Li Shen, Yong Zhang, Jue Wang, and Baoyuan Wu. Boosting the transferability of adversarial attacks with reverse adversarial perturbation. *Advances in neural information processing systems*, 35:29845–29858, 2022. 1, 3, 5

[20] Chunlin Qiu, Yiheng Duan, Lingchen Zhao, and Qian Wang. Enhancing adversarial transferability through neighborhood conditional sampling. *arXiv preprint arXiv:2405.16181*, 2024. 4

[21] Kaiyu Song, Hanjiang Lai, Yan Pan, and Jian Yin. Mimicdiffusion: Purifying adversarial perturbation via mimicking clean diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24665–24674, 2024. 1

[22] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 5

[23] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI conference on artificial intelligence*, 2017. 5

[24] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *ICLR*, 2018. 5

[25] Jiafeng Wang, Zhaoyu Chen, Kaixun Jiang, Dingkang Yang, Lingyi Hong, Pinxue Guo, Haijing Guo, and Wenqiang Zhang. Boosting the transferability of adversarial attacks with global momentum initialization. *Expert Systems with Applications*, 255:124757, 2024. 1, 3, 4, 5, 6

[26] Xiaosen Wang and Kun He. Enhancing the transferability of adversarial attacks through variance tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1924–1933, 2021. 1, 3, 5

[27] Xiaosen Wang, Xuanran He, Jingdong Wang, and Kun He. Admix: Enhancing the transferability of adversarial attacks.

In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16158–16167, 2021. 7

[28] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 3, 4

[29] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*, 2017. 1

[30] Cihang Xie, Zhishuai Zhang, Yuyin Zhou, Song Bai, Jianyu Wang, Zhou Ren, and Alan L Yuille. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2730–2739, 2019. 7

[31] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017. 1

[32] Hegui Zhu, Yuchen Ren, Xiaoyan Sui, Lianping Yang, and Wuming Jiang. Boosting adversarial transferability via gradient relevance attack. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4741–4750, 2023. 1, 2, 3, 4, 5