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

Zenghao Niu [1] Weicheng Xie [1,2,3] * Siyang Song [4] Zitong Yu [5] Feng Liu [1] Linlin Shen [6]
[1] School of Computer Science & Software Engineering, Shenzhen University, China
[2] Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen, China
[3] Guangdong Provincial Key Laboratory of Intelligent Information Processing, Shenzhen University, China
[4] School of Computer Science, University of Exeter, U.K.
[5] Department of Computing and Information Technology, Great Bay University, China
[6] Computer Vision Institute, School of Artificial Intelligence, Shenzhen University, China
{2300271072@mail., wcxie@}szu.edu.cn
s.song@exeter.ac.uk, zitong.yu@ieee.org, {feng.liu@, llshen@}szu.edu.cn

Abstract

Adversarial attacks present a critical challenge to deep neural networks' robustness, particularly in transfer scenarios across different model architectures. However, the transferability of adversarial attacks faces a fundamental dilemma between Exploitation (maximizing attack potency) and Exploration (enhancing cross-model generalization). Traditional momentum-based methods over-prioritize Exploitation, i.e., higher loss maxima for attack potency but weakened generalization (narrow loss surface). Conversely, recent methods with inner-iteration sampling over-prioritize Exploration, i.e., flatter loss surfaces for cross-model generalization but weakened attack potency (suboptimal local maxima). To resolve this dilemma, we propose a simple yet effective Gradient-Guided Sampling (GGS), which harmonizes both objectives through guiding sampling along the gradient ascent direction to improve both sampling efficiency and stability. Specifically, based on MI-FGSM, GGS introduces inner-iteration random sampling and guides the sampling direction using the gradient from the previous inner-iteration (the sampling's magnitude is determined by a random distribution). This mechanism encourages adversarial examples to reside in balanced regions with both flatness for cross-model generalization and higher local maxima for strong attack potency. Comprehensive experiments across multiple DNN architectures and multimodal large language models (MLLMs) demonstrate the superiority of our method over state-of-the-art transfer attacks. Code is made available at https://github.com/anuin-cat/GGS.

*Corresponding author.

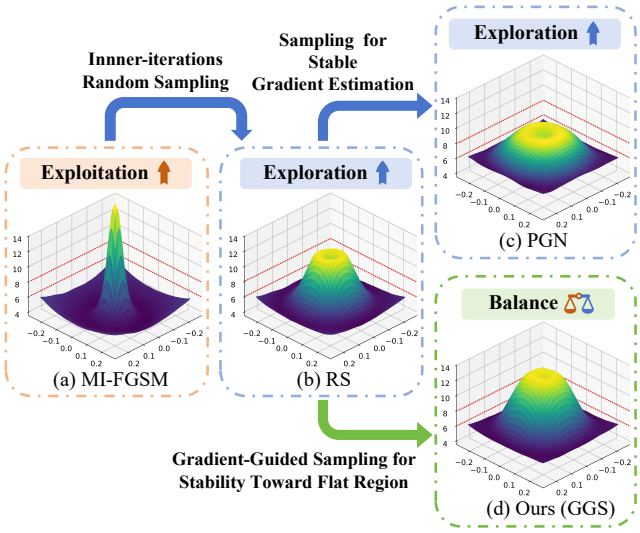


Figure 1. The loss surfaces of (a) MI-FGSM [1] (Momentum iterative fast gradient sign method), (b) RS (Base inner-iteration Random Sampling defined in section 3.2) for enhancing exploration within the neighborhood, (c) PGN [7] (Penalizing Gradient Norm) with Random Sampling (RS) for stable gradient estimation to enhance exploration, and (d) Our GGS (Gradient-Guided Sampling) for efficient sampling to generate gradients that stably towards flat regions with higher local maxima. It shows that, building upon RS, our approach not only maintains a flat loss surface, but also improves the local maximum loss value, compared to PGN, achieving a balance between exploration and exploitation.

1. Introduction

Researches have indicated that adding imperceptible perturbations to input data can easily affect deep neural networks

(DNNs), leading to erroneous predictions in safety-critical domains such as autonomous driving [13, 35], and cybersecurity [15, 28]. To facilitate the development of relevant defense mechanisms, in-depth research on adversarial attack methods is urgently needed.

While adversarial attacks are categorized into white-box and black-box settings, the black-box scenario is more realistic in practical applications, where the attacker lacks knowledge of the target model and relies solely on a surrogate model, aiming to generate adversarial examples that are effective across multiple target models. To enhance the transferability of adversarial examples in this setting, researchers have explored gradient-based methods [1, 7, 32], input transformation methods [2, 33, 37], and generative methods [16, 20, 36] to mitigate overfitting of adversarial examples on surrogate models. Despite significant progress, the effectiveness of attacks on target models remains substantially lower than that on surrogate models, driving the need for more effective attack strategies.

In black-box adversarial attacks, RAP [24] adopted Sharpness-Aware Minimization (SAM [6]) to improve transferability by flattening the loss surface. Building on this, PGN [7] further enhanced the transferability of adversarial examples by penalizing gradient norm and flattening the loss surface. The efficacy of flat loss surface in improving adversarial attack transferability is increasingly supported [4, 7, 38]. To enhance exploration through local flatness, recent approaches, building on MI-FGSM's [1] momentum-based **outer-iterations**, often incorporates **inner-iterations** before each example update [7, 40], i.e., performing multi-step neighborhood sampling for stabilization.

However, these methods enhance Exploration (cross-model generalization) at the cost of compromising Exploitation (attack potency). This is because, although inner-iterations can significantly enhance exploration capability through neighborhood sampling, independent random sampling struggles to obtain stable and effective gradient directions, while potential noise may cause the averaged gradient to deviate from the optimal ascent toward target regions, which exhibit flatter loss surface and higher local maxima. To optimize sampling efficiency, a guided sampling constraint is essential, which should satisfy two criteria: (1) ensuring stable gradient ascent direction and (2) directing gradients toward flatter regions, thereby enabling samples to reach flatter local maxima regions with enhanced transferability, balancing exploitation and exploration.

For (1) stable gradient direction, a sufficiently small neighborhood sampling followed by gradient calculation generally yields correct ascent directions. Yet, how can we further stabilize this direction? The Nesterov Accelerated Gradient (NAG) [23] enhances stability by firstly updating samples along the current momentum (lookahead) before acquiring gradients, providing more stable directions. Therefore, momentum-based direction constraints are promising for sampling, as the additional random magnitude can be used to preserve the sampling randomness. (2) However, directly applying NAG's momentum mechanism during inner-iteration sampling imposes strong constraints on subsequent directions, restricts the exploration capability for flat regions. To address this, we replace momentum with the gradient from the previous inner-iteration, named Gradient-Guided Sampling (GGS). This adjustment ensures that sampling to align with gradient ascent directions while reducing the impact from early-stage instability. Our GGS can achieve stable sampling directions after brief oscillations, consistently targeting flat loss regions with larger local maxima regardless of the initial sampling quality.

As shown in Fig. 1(d), our GGS generates adversarial examples with flatter loss surface and higher local maxima compared to RS, i.e., it does not sacrifice local maxima in the pursuit of a flatter surface compared to PGN, thereby achieving a balance between exploration and exploitation to facilitate attack transferability. Additionally, since our approach represents a complementary improvement over Random Sampling (RS), it is compatible with RS-based methods, such as inner-iteration random sampling and gradient averaging, including PGN [7] and GRA [40]. In summary, this study makes the following contributions:

- We propose Gradient-Guided Sampling (GGS), a novel inner-iteration sampling strategy that effectively balances exploration (cross-model generalization) and exploitation (attack potency). By leveraging the gradient from the previous inner-iteration to guide sampling directions, GGS achieves stable gradient ascent toward flat loss regions with higher local maxima.
- The GGS framework is compatible with existing inner-iteration random sampling-based methods, enhancing their sampling efficiency and further improving the transferability of the adversarial examples they generate.
- Extensive experimental results as well as comprehensive visualizations demonstrate the efficacy of our GGS, covering targeted and non-targeted attacks in cross-architecture black-box scenarios, non-targeted attacks on multimodal large language models (MLLMs) and commercial cloud functions.

2. Related Work

Transferability is a critical property in adversarial attacks, allowing adversarial examples generated from a surrogate model to deceive unseen target models in black-box settings. This characteristic makes transfer-based attacks highly practical in real-world applications where target models are inaccessible, motivating our focus on transferable adversarial attacks.

2.1. Transferable Attack

Gradient-based Attack. Gradient-based attacks aim to generate adversarial examples by exploiting the gradients of a surrogate model to maximize the loss. Classic methods such as Fast Gradient Sign Method (FGSM) [9] and Projected Gradient Descent (PGD) [22] optimize adversarial perturbations to fool the surrogate model. However, these attacks often overfit the surrogate model, resulting in limited transferability to target models. To address this, advanced techniques such as momentum-based (MI-FGSM [1]), Nesterov Iterative-based (NI-FGSM [19]), variance-tuned (VMI-FGSM [32]), gradient relevance-based (GRA[40]), momentum initialization (GI-FGSM [30]) and distribution-based methods (ANDA[5]) have been introduced to smooth out the optimization process and improve transferability across models.

Input Transformation. To mitigate the overfitting issue inherent in gradient-based methods, input transformation techniques have been proposed to enhance the diversity of adversarial examples. Techniques like translation, resizing, padding, image mixup (e.g., DIM [37], SIM [19], TIM [2], Admix[33]) and other advanced image transformation methods (e.g., SSM[21], SIA[34], STM[8], L2T[41], BSR[31]) introduce randomness to the input data before generating perturbations. By applying such transformations during the attack process, more effective transferable adversarial examples can be generated.

2.2. Flat Maxima

The generalization ability of models has been suggested to possibly have a certain association with flat minima [12]. More in-depth research and exploration have gradually confirmed this perspective [6, 17, 18, 39]. Moreover, flat maxima also have been validated to be effective for enhancing the generalization and transferability of adversarial examples [24]. Flat maxima refers to regions in the loss landscape, where small changes in the model parameters result in minimal changes in the loss, making adversarial examples generated in these regions less sensitive to the specific decision boundaries of models. This property contrasts with "sharp maxima", which are highly sensitive to small perturbations and can lead to overfitting to the surrogate model, reducing transferability capacity.

In the context of transfer-based attacks, works like Reverse Adversarial Perturbation (RAP) [24] leverage flat maxima to generate adversarial examples, PGN [7] adopts a first-order procedure to approximate the Hessian/vector product, largely improving computational efficiency. They are not only effective on the surrogate model but also more resilient when attacking unseen target models. These methods mitigate overfitting, via making adversarial examples lie in regions where the loss function remains stable across different models (e.g., flat maxima regions), improving attack success rates in black-box settings. However, achieving flat maxima requires not only attention to "flatness" but also preservation of the "maxima" to ensure sufficient attack strength during transfer attack, which poses a significant challenge.

2.3. Inner-Iteration Sampling

To enhance the generalization capability of adversarial examples, recent methods have introduced inner-iteration sampling [7, 24, 32]. Specifically, an inner iterative process is inserted before updating the examples. This process involves neighborhood sampling to acquire domain information during inner-iterations which are used to adjust gradients [32] or neighborhood search is incorporated to locate local minima and enhance their loss values [24]. Furthermore, works like [7, 40] enhance the flatness of the loss landscape by averaging gradients from inner-iterations to update the examples.

Additionally, neighborhood sampling can facilitate examples to escape from sharp local maxima regions, thereby avoiding overfitting to the surrogate model. However, directly averaging the gradients from inner-iterations will reduce the maximum value of the loss surface, since random sampling produces unstable gradient directions that fail to consistently direct to flat regions with higher local maxima. Thus, enhancing inner-iteration sampling efficiency to stabilize the final gradient has become a critical challenge.

3. Methodology

3.1. Preliminaries

Transferable Adversarial Attacks. For a given example (x, y), a surrogate model f_θ and multiple target models f_{φ_k} for $k \in \{1, 2, ..., K\}$, the attacker's goal is to find $x^{adv} = x + \delta$ using f_θ to make $f_{\varphi_k}(x^{adv}) \neq y$, where $\|\delta\|_p < \epsilon$, ϵ denotes the maximum magnitude of the specified perturbation and $\|\cdot\|_p$ denotes the ℓ_p-norm. For targeted attacks, we simply adapt the symbols and labels in the target function: $f_{\varphi_k}(x^{adv}) = y_t$, where y_t denotes the target label. Meanwhile, to facilitate a clearer representation of the norm constraints against x^{adv} as below, we define $\mathcal{B}_\epsilon(x) = \{x' : \|x' - x\|_p \leq \epsilon\}$ to denote the ϵ-ball of an input image x. For the purpose of the above attack, we generally need to maximize the following objective function to generate the transferable adversarial examples:

$$\max_{x^{adv} \in \mathcal{B}_\epsilon(x)} \mathcal{L}\left(x^{adv}, y\right), \tag{1}$$

where $\mathcal{L}(\cdot)$ represents the loss function. In the above flow, we assume that f_θ is known as a white-box model and f_{φ_k} is unknown as a black-box model. Similar to the process of training a neural network, the white-box model can be seen as the training data, with the adversarial examples serving

as the optimization parameters, while the black-box model acts as the test data.

Nesterov Accelerated Gradient (NAG). Due to the similarity between model training task and adversarial example generation, improving performance in black-box settings can be viewed as enhancing model generalization in model training tasks. Consequently, some commonly used optimization methods, such as Momentum [25] and Nesterov Accelerated Gradient (NAG) [23], can be introduced to improve the generalization of adversarial examples across black-box models through a more stable optimization process.

NI-FGSM [19] was the first to incorporate the principles of NAG into adversarial attack tasks. This enables the algorithm to acquire directional information in advance, reducing oscillations and achieving stable convergence. Specifically, momentum is computed at each step as follows:

$$\begin{aligned} v_t &= \gamma \cdot v_{t-1} + \frac{\nabla_x \mathcal{L}\left(\tilde{x}_t, y\right)}{\left\|\nabla_x \mathcal{L}\left(\tilde{x}_t, y\right)\right\|_1}, \\ \tilde{x}_t &= \underbrace{x_{t-1}^{adv} + \alpha \cdot \gamma \cdot v_{t-1}}_{\text{lookahead point}}, \end{aligned} \tag{2}$$

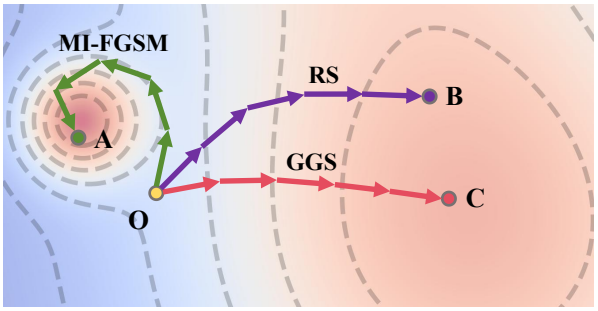
where v_t represents the momentum term, $x_0^{adv} = x$, α stands for the step size, γ denotes the momentum decay factor, and $\|\cdot\|_1$ denotes the ℓ_1-norm of the given variable. Unlike traditional momentum-based methods, NI-FGSM utilizes the estimated position of examples rather than their current actual position, during gradient computation, i.e., introducing a **lookahead** mechanism into the algorithm.

3.2. Motivation

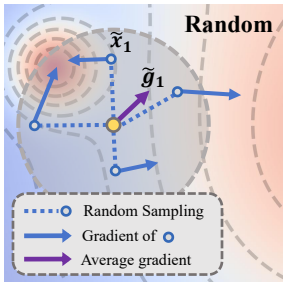
Recent transfer-based adversarial attack methods have introduced inner-iteration random sampling, to enhance algorithm stability [7, 40] or acquire neighborhood information [24, 32]. It can be noted that these methods have increased the flatness of the loss surface, as illustrated in Fig.1(c), with additional loss surface plots for other methods provided in the supplementary materials. If we isolate the inner-iteration random sampling (RS) component, it can be expressed in the following form:

$$\begin{aligned} v_t &= \gamma v_{t-1} + \frac{\sum_{i=1}^{N} \tilde{g}_i}{\left\|\sum_{i=1}^{N} \tilde{g}_i\right\|_1}, \\ \tilde{g}_i &= \nabla_x \mathcal{L}\left(\tilde{x}_i, y\right), \text{ with } \tilde{x}_i = \underbrace{x_{t-1}^{adv} + \tilde{p}}_{\text{sampling point}}, \end{aligned} \tag{3}$$

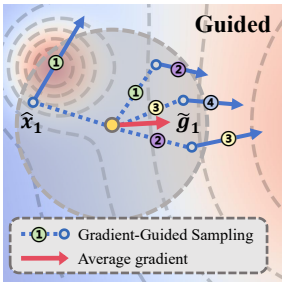
where $\sum_{i=1}^{N}$ denotes the output of the inner-iteration process; N denotes the number of inner-iterations; γ denotes the momentum decay factor; $\tilde{g}_0, \tilde{p} \sim \text{Uniform}(-\zeta, \zeta)$, are uniform random noises with the same size as x.



(a) Optimization Trajectories of Various Methods



(b) 1st step process of RS

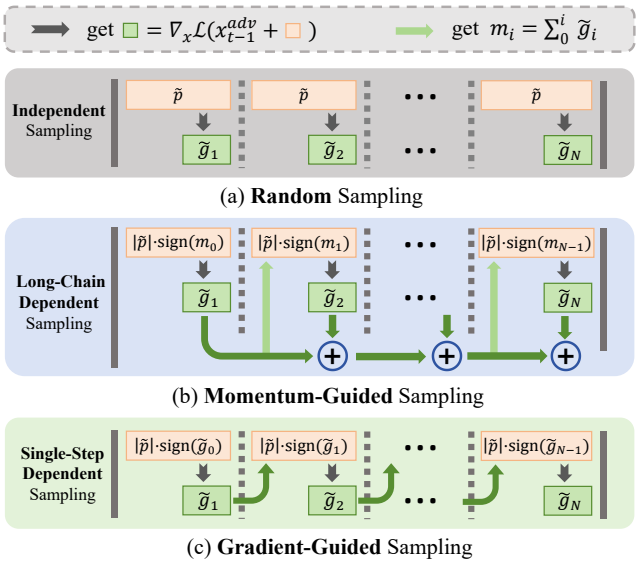


(c) 1st step process of GGS

Figure 2. (a) Outer-Iteration processes of MI-FGSM (green), RS (purple) and our GGS (red). (i) MI-FGSM prioritizes rapid ascent directions, thereby enhancing exploitation and facilitating access to sharp local maxima regions. (ii) RS introduces inner-iterations and enhances exploration capability through neighborhood search, thereby enabling access to flat local maxima regions. (iii) GGS incorporates gradient constraints into RS, simultaneously enhancing both exploration and exploitation capabilities, thereby enabling the faster convergence to the centers of flat local maxima regions. (b) RS is performed within a neighborhood of the current example, and uses the average gradient of all sampled points as the final gradient. (c) Building upon RS, GGS enables examples to have stable gradient directions (② ③ ④) toward the centers of flat local maxima regions, following an initial period of brief oscillation (①). It uses the gradient direction from the previous inner-iteration as the guidance, while the randomness is maintained by setting the gradient magnitude with a random distribution.

As shown in Fig. 1, the inner-iteration RS possesses a strong capability to enhance the flatness of the loss surface compared to MI-FGSM. Compared to RS, current methods such as PGN [7] can further improve the flatness of the loss surface, however, they unwillingly reduce the local maxima of the loss surface, limiting its attack potential. This reduction in local maxima can be attributed to the unstable gradients generated by the completely random sampling strategy, as illustrated in Fig. 2(b). Specifically, the averaged gradient roughly points towards flat regions, while it struggles to consistently align with the stable direction to the center of flat maxima regions, thereby reducing updating efficiency.

To acquire stable gradient ascent direction and improve the efficiency of inner-iteration sampling, we define two es-



(a) **Random** Sampling

(b) **Momentum-Guided** Sampling

(c) **Gradient-Guided** Sampling

Figure 3. The differences among three inner-iteration sampling guidance methods: (a) Random sampling in Eq. 3, with each inner-iteration sampling being independent; (b) Momentum-guided sampling in Eq. 4, where sampling direction depends on the cumulative average of all previous gradients, creating long-chain dependencies, while using random sampling for maintaining the randomness; (c) Gradient-guided sampling in Eq. 5, where sampling direction relies solely on the gradient direction of the previous iteration, establishing single-step dependencies.

sential characteristics that a well-balanced gradient direction should possess: (1) alignment with the stable gradient ascent direction for improving Exploitation capacity, and (2) pointing to loss regions with flatter surface for improving Exploration capacity.

3.3. Gradient-Guided Sampling

(1) Stable gradient ascent direction: A sampled gradient in a sufficiently small neighborhood of the current example is probably a gradient ascent direction, while how can we ensure that this gradient ascent direction is stable? The Nesterov Accelerated Gradient (NAG) technique can precisely address this issue. It first allows the example to "lookahead" along the current momentum direction, and then acquires the gradient based on this new position, thereby identifying a more stable ascent direction [19, 23]. To leverage the superior stability of NAG based on random sampling, we perform the magnitude sampling along the momentum direction with an uniform distribution, and named this approach as Momentum-Guided Sampling (MGS). Compared to RS in Eq. 3, we only need to replace $\tilde{x}_i$ with $\bar{x}_i$ as follows:

$$\bar{x}_i = \underbrace{x_{t-1}^{adv} + |\tilde{p}| \cdot \text{sign}(m_{i-1})}_{\text{lookahead sampling point}}, \tag{4}$$

where $|\tilde{p}|$ denotes the element-wise absolute value of $\tilde{p}$, $\text{sign}(\cdot)$ denotes the sign function, which outputs either 1 or -1, and momentum decay is set to 1 for simplicity, so that $m_i = \sum_{k=1}^{i} \tilde{g}_k$.

(2) Pointing to regions with better flatness: As shown in Fig. 3(b), if we use momentum for 'lookahead', the early unstable sampling in the inner-iterations will impose excessive constraints on subsequent sampling directions, leading to long-chain dependencies that severely impair RS's ability in exploring flat regions. Thus, for our GGS, we substitute the momentum term in Eq. 4 with the gradient obtained from the previous iteration, where the lookahead mechanism is maintained with the gradient ascent direction. Specifically, we only need to replace $\tilde{x}_i$ in Eq. 3 with $\hat{x}_i$ as follows:

$$\hat{x}_i = \underbrace{x_{t-1}^{adv} + |\tilde{p}| \cdot \text{sign}(\tilde{g}_{i-1})}_{\text{lookahead sampling point}}, \tag{5}$$

Fig. 3 further sheds light on the differences among three inner-iteration sampling guidance methods. As shown in Fig. 3(c), GGS largely alleviates long-chain dependencies compared with Momentum-Guided Sampling in Fig. 3(b), while ensuring the sampling direction to be align with the gradient ascent direction, compared with random sampling in Fig. 3(a). In addition, as shown in Fig. 2(c), GGS will achieve stable sampling directions (② ③ ④) after a brief period of oscillation (①), when the initial sampling points are unstable.

For clarity, our attack process is shown in Algorithm 1.

Algorithm 1 The attacking process of our GGS.

Input: A clean image $x^{C\times H\times W}$ with ground-truth label y, and the loss function $\mathcal{L}(\cdot)$; the magnitude of perturbation ϵ; the sampling radius ζ; the number of outer iterations, T; the decay factor γ; the number of inner iterations, N; $\tilde{g}_0, \tilde{p} \in \mathbb{R}^{C\times H\times W}$

Output: An adversarial example x^{adv};

Randomly sample $\tilde{g}_0 \sim \text{Uniform}(-\zeta, \zeta)$;
$v_0 = 0$, $x_0^{adv} = x$, $\alpha = \epsilon/T$;
while $t \leftarrow 1$ to T (outer-iteration) **do**
  while $i \leftarrow 1$ to N (inner-iteration) **do**
    Randomly sample noises $\tilde{p} \sim \text{Uniform}(-\zeta, \zeta)$;
    Get the lookahead example $\hat{x}_i$ by Eq. (5);
    Get the gradient $\tilde{g}_i = \nabla_x \mathcal{L}(\hat{x}_i, y)$
  end while
  Update $v_t = \gamma \cdot v_{t-1} + \frac{\sum_{i=1}^{N} \tilde{g}_i}{\left\|\sum_{i=1}^{N} \tilde{g}_i\right\|_1}$
  Update $x_t^{adv} = \Pi_{\mathcal{B}_\epsilon(x)}\left[x_{t-1}^{adv} + \alpha \cdot \text{sign}\left(v_t\right)\right]$;
end while
$x^{adv} = x_T^{adv}$

Table 1. The average untargeted and targeted ASR (%) on the nine models (Res50 [10], Dense121 [14], Inc-v3 [26], IncRes-v2 [27], ViT-B [3], PiT-B [11], and Inc-v3$_{ens3}$, Inc-v3$_{ens4}$, IncRes-v2$_{ens}$ [29]), with adversarial examples generated on a single model (Res50, Inc-v3 or 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 underline, respectively.

	Attack	**Res50**	Dense121	**Inc-v3**	IncRes-v2	**ViT-B**	PiT-B	Inc-v3$_{ens3}$	Inc-v3$_{ens4}$	IncRes-v2$_{ens}$	Avg.
Res50	**MI** CVPR'18	99.8 / 98.1	54.9 / 0.2	44.2 / 0.0	28.0 / 0.0	11.8 / 0.0	22.4 / 0.0	22.8 / 0.0	25.3 / 0.0	19.6 / 0.0	36.53 / 10.92
	NI ICLR'20	100.0 / 95.1	63.1 / 0.2	47.2 / 0.0	32.3 / 0.0	12.9 / 0.0	23.7 / 0.0	23.0 / 0.0	27.5 / 0.0	19.0 / 0.0	38.74 / 10.59
	VMI CVPR'21	99.8 / 96.6	71.2 / 2.8	58.1 / 0.5	49.4 / 0.2	31.3 / 0.1	47.7 / 0.5	43.9 / 0.0	45.4 / 0.1	37.7 / 0.1	53.83 / 11.21
	RAP NeurIPS'22	99.6 / 26.4	86.8 / 0.5	68.4 / 0.0	52.6 / 0.0	22.2 / 0.0	40.5 / 0.1	36.2 / 0.0	36.7 / 0.0	25.5 / 0.0	52.06 / 3.00
	GRA ICCV'23	96.9 / 67.7	88.6 / 3.9	81.8 / 0.9	75.8 / 1.7	45.3 / 0.2	62.6 / 0.6	71.5 / 0.7	70.9 / 1.1	67.3 / 0.7	73.41 / 8.61
	PGN NeurIPS'23	98.6 / 49.5	91.3 / 4.7	85.0 / 1.4	78.5 / 1.7	49.7 / 0.4	67.8 / 1.0	74.9 / 0.6	72.9 / 1.1	70.1 / 1.2	76.53 / 6.84
	ANDA CVPR'24	99.9 / 97.6	87.5 / 5.4	73.6 / 0.4	66.3 / 0.8	42.3 / 0.1	61.8 / 1.1	51.0 / 0.1	52.9 / 0.1	46.7 / 0.1	64.67 / 11.74
	GI ESWA'24	100.0 / 98.5	68.7 / 0.4	55.6 / 0.1	37.6 / 0.1	15.3 / 0.0	29.0 / 0.0	28.8 / 0.0	30.4 / 0.1	22.4 / 0.0	43.09 / 11.02
	GGS	99.3 / 90.7	95.9 / 28.6	89.6 / 7.2	86.4 / 8.6	60.2 / 3.4	81.0 / 8.5	77.5 / 3.7	77.4 / 4.3	71.4 / 4.0	**82.08** / **17.67**
Inc-v3	MI	32.1 / 0.0	49.6 / 0.1	100.0 / 95.3	56.1 / 0.1	13.0 / 0.0	17.3 / 0.0	34.3 / 0.0	38.0 / 0.0	26.3 / 0.0	40.74 / 10.61
	NI	41.4 / 0.0	60.5 / 0.1	100.0 / 75.9	66.9 / 0.1	13.4 / 0.0	19.2 / 0.0	36.8 / 0.0	40.6 / 0.1	29.5 / 0.0	45.37 / 8.47
	VMI	44.9 / 0.1	65.6 / 0.1	99.9 / 76.6	72.7 / 0.4	18.3 / 0.0	25.3 / 0.0	48.9 / 0.1	52.5 / 0.0	40.9 / 0.0	52.11 / 8.59
	RAP	58.0 / 0.0	78.4 / 0.2	100.0 / 16.4	82.1 / 0.1	15.4 / 0.0	23.8 / 0.0	44.1 / 0.1	47.0 / 0.0	33.9 / 0.0	53.63 / 1.87
	GRA	53.3 / 0.1	75.4 / 0.4	99.8 / 67.5	85.4 / 1.8	21.4 / 0.0	30.5 / 0.1	64.9 / 0.1	65.9 / 0.3	56.1 / 0.1	61.41 / 7.82
	PGN	56.0 / 0.0	79.4 / 0.2	100.0 / 50.3	87.9 / 0.9	23.2 / 0.0	33.4 / 0.0	65.9 / 0.0	67.0 / 0.1	55.6 / 0.0	63.16 / 5.72
	ANDA	51.3 / 0.3	76.3 / 0.4	100.0 / 87.4	82.5 / 0.6	20.9 / 0.0	31.8 / 0.1	58.2 / 0.2	61.2 / 0.2	49.3 / 0.0	59.06 / 9.91
	GI	39.9 / 0.0	59.3 / 0.3	100.0 / 97.1	62.6 / 0.5	14.6 / 0.0	19.9 / 0.0	39.2 / 0.0	40.9 / 0.0	31.9 / 0.0	45.37 / **10.88**
	GGS	69.1 / 1.1	86.9 / 1.9	100.0 / 76.0	95.8 / 6.5	27.9 / 0.0	39.7 / 0.1	70.0 / 0.7	73.9 / 1.4	60.1 / 0.5	**69.27** / 9.80
ViT-B	MI	41.7 / 0.0	55.6 / 0.0	58.1 / 0.0	39.4 / 0.0	100.0 / 99.9	50.1 / 0.0	34.2 / 0.0	36.3 / 0.0	29.9 / 0.0	49.48 / 11.10
	NI	42.4 / 0.0	58.3 / 0.0	59.5 / 0.0	44.7 / 0.1	100.0 / 99.0	50.0 / 0.2	36.2 / 0.0	37.6 / 0.0	31.0 / 0.0	51.08 / 11.03
	VMI	54.8 / 0.6	65.9 / 0.4	63.1 / 0.1	50.9 / 0.3	100.0 / 99.0	67.6 / 1.5	45.3 / 0.1	44.7 / 0.0	38.7 / 0.2	59.00 / 11.36
	RAP	62.1 / 0.1	79.6 / 0.2	73.7 / 0.0	62.3 / 0.0	99.9 / 57.0	66.2 / 0.1	48.5 / 0.0	51.2 / 0.0	42.6 / 0.0	65.12 / 6.38
	GRA	64.9 / 1.0	77.6 / 1.1	74.7 / 0.8	68.8 / 0.8	99.3 / 91.7	80.9 / 4.7	61.5 / 0.2	64.0 / 0.1	57.7 / 0.2	72.16 / 11.18
	PGN	69.3 / 0.4	81.1 / 0.7	78.6 / 0.4	70.2 / 0.5	99.1 / 82.2	84.8 / 4.4	65.6 / 0.1	67.5 / 0.3	61.4 / 0.2	75.29 / 9.91
	ANDA	66.2 / 0.3	79.0 / 0.5	75.5 / 0.1	65.7 / 0.3	100.0 / 99.9	79.0 / 1.4	55.3 / 0.1	56.7 / 0.1	49.9 / 0.2	69.70 / 11.44
	GI	55.8 / 0.1	69.6 / 0.2	67.1 / 0.0	52.5 / 0.1	100.0 / 100.0	61.0 / 0.3	45.4 / 0.0	46.3 / 0.0	38.0 / 0.0	59.52 / 11.19
	GGS	80.8 / 6.0	89.9 / 6.5	87.4 / 4.6	82.3 / 4.6	100.0 / 99.2	92.7 / 27.3	74.0 / 2.1	74.5 / 2.3	68.4 / 2.4	**83.33** / **17.22**

4. Experiments

4.1. Experimental Settings

Dataset: Our experiments are conducted on an ImageNet-compatible dataset that is widely used in adversarial attack research. The dataset includes 1,000 images, each with a size of 299 × 299 × 3.

Models: We select six normally pre-trained models, i.e., ResNet50 (Res50) [10], DenseNet121 (Dense121) [14], Inception-v3 (Inc-v3) [26], InceptionResNet-v2 (IncRes-v2) [27], VisionTransformer-Base (ViT-B) [3], and Pooling-based VisionTransformer-Base (PiT-B) [11], along with three models pre-trained on adversarial examples Inception-v3$_{ens}$ (Inc-v3$_{ens}$), Inception-v4$_{ens}$ (Inc-v4$_{ens}$) and InceptionResNet-v2$_{ens}$ (IncRes-v2$_{ens}$) [29], to evaluate the performance of different methods.

Baselines: To provide a comprehensive evaluation of our method, we choose eight popular gradient-based adversarial attack methods as baselines: MI-FGSM [1], NI-FGSM [19], VMI-FGSM [32], RAP [24], GRA [40], PGN [7], ANDA [5] and GI [30]. Additionally, we integrate our method with five input transformation-based methods, i.e., DIM [37], TIM [2], SIM [19], Admix [33], and SSM [21] to verify the generalizability of our approach.

Hyperparameters: For our GGS, we set the maximum perturbation $\epsilon = 16/255$, the number of outer-iterations $T = 10$, the step size $\alpha = \epsilon/T$, the number of inner-iterations $N = 20$, and the upper bound factor of sample range $\zeta = 2.0 \times \epsilon$. More hyperparameter explanation for other methods is provided in the Supplementary Material.

4.2. Attack on Different Models

4.2.1. Classification model

As shown in Table 1, our method consistently outperforms state-of-the-art (SOTA) approaches. Compared to the best-performing competitor, it improves the average attack success rate (ASR) by over 5% in both untargeted and targeted attacks across most architectures, except for Inception-v3. Furthermore, our GGS significantly enhances attack performance on the surrogate model, achieving 4%–25% higher ASR in targeted attacks than other sampling methods like PGN [7] and GRA [40]. This demonstrates GGS's ability to boost attack potency while maintaining strong cross-model generalization. Additional details on ensemble setting can be found in the supplementary materials.

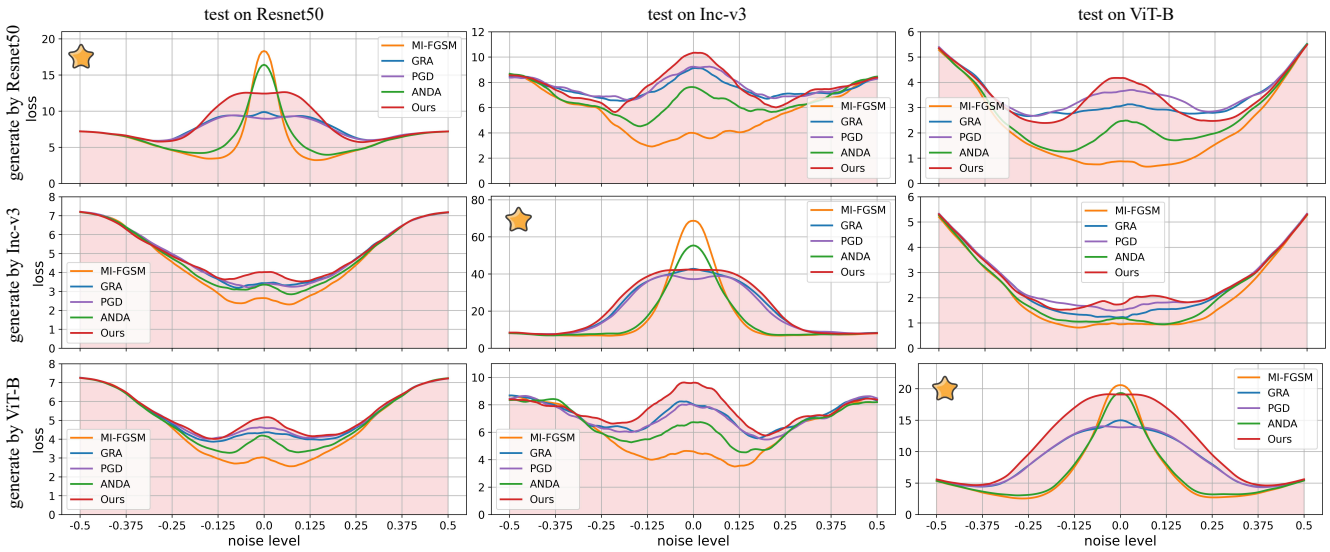


Figure 4. The loss surfaces of adversarial examples generated by different methods (MI-FGSM [1], GRA [40], PGN [7], ANDA [5]), based on three different model architectures (ResNet50[10], Inception-v3[26], and ViT-B[3]) along random directions with varying strengths like [7]. In the images marked with a star in the upper left corner, adversarial examples are generated and tested on the same model, indicating white-box testing. In contrast, unmarked images represent adversarial examples generated on one model and tested on another, indicating black-box testing. We have also highlighted the regions covered by our loss surface using a red background for easier visualization.

4.2.2. Multimodal Large Language Models

The widespread adoption of Multimodal Large Language Models (MLLMs) has raised significant security concerns. We evaluated our method on five prominent MLLMs using adversarial examples generated under an ensemble setting, as shown in Table 2. Our GGS method achieved superior attack performance, reducing the average CSR by over 9% compared to the strongest baseline, which demonstrates its excellent cross-model applicability.

Table 2. Classification success rate (CSR, %) of adversarial examples generated by different methods in Multimodal Large Language Models (MLLMs) from three major providers (OpenAI, Google, Anthropic) under ensemble settings (Res50 [10], Inc-v3 [26], ViT-B [3]). Row "clean" indicates the CSR on the clean samples. Avg. represents the average CSR across the five MLLMs.

Method	GPT		Gemini		Claude	Avg. ↓
	4o	mini	pro	flash	sonnet	
Clean	77.1	84.0	85.6	80.9	68.3	79.18
MI CVPR'18 [1]	77.0	77.3	81.8	76.5	61.0	74.72
NI ICLR'20 [19]	75.0	73.9	83.1	76.8	62.4	74.24
VMI CVPR'21 [32]	68.8	66.6	78.4	73.2	63.1	70.02
RAP NeurIPS'22 [24]	64.3	62.0	75.1	66.1	60.2	65.54
GRA ICCV'23 [40]	57.1	52.2	70.9	65.6	46.7	58.50
PGN NeurIPS'23 [7]	56.2	49.0	69.6	64.1	45.1	56.80
ANDA CVPR'24 [5]	59.9	56.8	74.9	69.3	50.5	62.28
GI ESWA'24 [30]	71.5	72.8	79.4	73.2	63.8	72.14
GGS	**43.1**	**37.8**	**61.1**	**55.7**	**40.0**	**47.54**

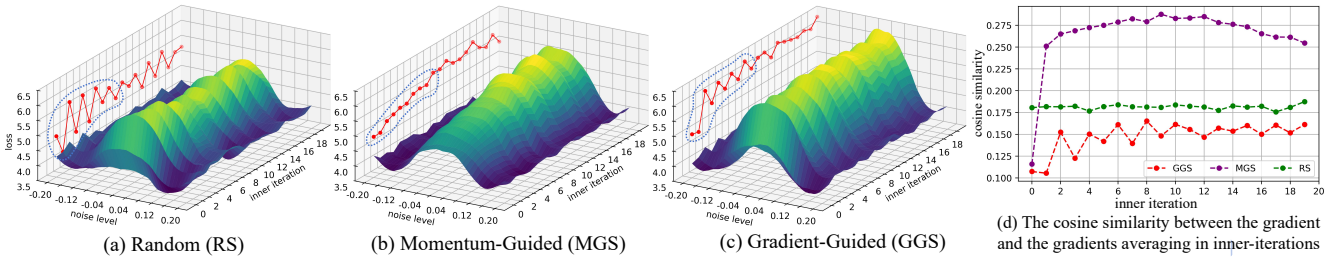
4.2.3. Loss Surface Visualization against Noise Levels

To shed light on the capacity of our GGS in locating flat maxima regions for adversarial examples, we compared the loss surfaces of adversarial examples generated by different attack methods in Fig. 4. Each curve represents the average loss of 32 randomly selected adversarial examples, while the center of each image (noise level = 0) indicates the average loss of clean adversarial examples. In white-box settings (marked with a star), GGS consistently finds flatter maxima compared to baselines. Consequently, in black-box testing, GGS achieves the highest loss values (images without star), indicating its superior transfer attack capability. Additionally, the GGS loss surface (highlighted in red) almost entirely encompasses those of other methods, confirming its effectiveness in balancing strong attack potency with cross-model generalization.

4.3. Compatibility with Other Methods

Attack with Inner-Iteration Sampling Methods: Our Gradient-Guided Sampling (GGS) is compatible with existing inner-iteration RS-based approaches. When integrated with SOTA approaches like GRA [40] and PGN [7], GGS largely improves attack success rates (ASR). As shown in Table 3, untargeted ASR improved by 5.2% (GRA) and 6.7% (PGN), while targeted ASR increased by 5% (GRA) and 6% (PGN), demonstrating GGS's compatibility.

Attack with Input Transformations Methods: To study the compatibility of our GGS with existing input transformation techniques, we integrated it with five meth-



(a) Random (RS) (b) Momentum-Guided (MGS) (c) Gradient-Guided (GGS) (d) The cosine similarity between the gradient and the gradients averaging in inner-iterations

Figure 5. Loss surfaces of adversarial examples generated by different sampling strategies, Random Sampling (RS), MGS and our GGS, with increasing inner-iteration on Resnet50. The red line on the left side of part (a)∼(c) represents the maximum value of the loss surface in each iteration, and the blue dashed line highlight these values in early-stage inner iteration. For (d), it represents the cosine similarity between the gradient $\tilde{g}_i$ generated in each inner-iteration and the average of gradients $\sum_{i=1}^{N} \tilde{g}_i/N$.

Table 3. The average 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[37], TIM[2], SIM[19], Admix[33], SSM[21]) and inner-iteration RS-based methods (GRA [40], PGN [7]).

Class	Method	Untargeted Avg.	Targeted Avg.
Random Sampling (RS)	GRA / +ours	73.41 / **78.69**	8.61 / **13.64**
	PGN / +ours	76.53 / **83.23**	6.84 / **12.79**
Input Transformation	DIM / +ours	51.30 / **90.13**	8.63 / **19.37**
	TIM / +ours	40.82 / **84.94**	11.03 / **18.48**
	SIM / +ours	46.80 / **90.12**	11.14 / **28.44**
	Admix / +ours	54.78 / **85.50**	11.28 / **30.50**
	SSM / +ours	74.22 / **80.74**	8.40 / **12.54**

ods of this type: DIM [37], TIM [2], SIM [19], Admix [33], and SSM [21]. Adversarial examples generated on ResNet50 were tested on nine diverse models. Table 3 shows GGS significantly improved adversarial transferability, increasing untargeted ASR by 6%–43% and targeted ASR by 4%–19%. For instance, SIM's untargeted ASR improved by 43% when combined with GGS. More details are in the supplementary material.

4.4. Ablation Study of Proposed Components

Table 4. The average untargeted ASR on ResNet50 and eight target models for Random Sampling (RS), Momentum-Guided Sampling (MGS), and our Gradient-Guided Sampling (GGS).

	Sampling Guided by			Res50	Others
	rand	momentum	gradient		
(a) RS	✓	-	-	97.3	63.74
(b) MGS	-	✓	-	97.5	58.79
(c) GGS	-	-	✓	**99.3**	**79.93**

Ablation study compares RS, MGS, and GGS via loss surface analysis (Fig. 5) and attack success rates (Table 4).

As shown in Fig. 5(a), Random Sampling (RS) yields a relatively flat loss surface due to its exploration capability but it suffers from significant oscillation in loss values. Introducing Momentum-Guided Sampling (MGS) (Fig. 5(b)) stabilizes the loss surface over iterations, but its long-chain dependency causes high similarity between later gradients and the final average gradient (Fig. 5(d)), severely limiting exploration. Our GGS resolves MGS's long-chain dependency and mitigates early-stage noisy gradients (Fig. 5(b)(c), blue dashed lines). The low gradient similarity in Fig. 5(d) reflects the enhanced exploration, while the flat yet higher-loss surface in Fig. 5(c) demonstrates the exceptional sampling efficiency of our GGS.

Quantitatively, as shown in Table 4, MGS marginally improves ASR of the surrogate model by 0.2%, but reducing transfer attack capability by 5%. In contrast, GGS increases both white-box and transfer ASR by 2% and 6%, respectively, validating its superior sampling efficiency.

5. Conclusion and Perspectives

This paper introduces Gradient-Guided Sampling (GGS), an easy-to-implement inner-iteration sampling strategy, for transferable adversarial attacks that effectively resolves the dilemma between Exploitation (attack potency) and Exploration (cross-model generalization). By guiding the sampling direction with the gradient from the previous inner iteration, GGS enables stable ascent toward flat loss regions with higher local maxima. Our GGS has the merit of enhancing exploitation capability without compromising exploration capability, effectively balancing these two aspects and addressing the inherent limitations of random sampling (RS) methods, as verified by extensive experiments and visualizations as well as the comparison with related state-of-the-art methods. GGS is also revealed to be highly compatible with existing methods.

While our GGS is highly compatible with gradient-averaging methods (e.g., GRA [40], PGN [7]), we aim to refine it to also support non-gradient-averaging techniques (e.g., VMI-FGSM [32], RAP [24]).

Acknowledgements

The work was supported by the National Natural Science Foundation of China under grants no. 62276170, 82261138629, 62306061, the Science and Technology Project of Guangdong Province under grants no. 2023A1515010688, the Science and Technology Innovation Commission of Shenzhen under grant no. JCYJ20220531101412030, Open Research Fund from Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ) under Grant No. GML-KF-24-11, and Guangdong Provincial Key Laboratory under grant no. 2023B1212060076.

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, 6, 7

[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. 2, 3, 6, 8

[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. 6, 7

[4] Yan Fang, Zhongyuan Wang, Jikang Cheng, Ruoxi Wang, and Chao Liang. Promoting adversarial transferability with enhanced loss flatness. In *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1217–1222. IEEE, 2023. 2

[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. 3, 6, 7

[6] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations, ICLR*, 2021. 2, 3

[7] 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, 6, 7, 8

[8] Zhijin Ge, Fanhua Shang, Hongying Liu, Yuanyuan Liu, Liang Wan, Wei Feng, and Xiaosen Wang. Improving the transferability of adversarial examples with arbitrary style transfer. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 4440–4449, 2023. 3

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

[10] 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. 6, 7

[11] 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. 6

[12] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural computation*, 9(1):1–42, 1997. 3

[13] SM Mostaq Hossain, Shampa Banik, Trapa Banik, and Ashfak Md Shibli. Survey on security attacks in connected and autonomous vehicular systems. In *2023 IEEE International Conference on Computing (ICOCO)*, pages 295–300. IEEE, 2023. 2

[14] 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. 6

[15] Martin Husák, Jana Komárková, Elias Bou-Harb, and Pavel Čeleda. Survey of attack projection, prediction, and forecasting in cyber security. *IEEE Communications Surveys & Tutorials*, 21(1):640–660, 2018. 2

[16] Mintong Kang, Dawn Song, and Bo Li. Diffattack: Evasion attacks against diffusion-based adversarial purification. In *Advances in Neural Information Processing Systems*, pages 73919–73942. Curran Associates, Inc., 2023. 2

[17] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *ICLR*, 2017. 3

[18] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31, 2018. 3

[19] 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. 3, 4, 5, 6, 7, 8

[20] Decheng Liu, Xijun Wang, Chunlei Peng, Nannan Wang, Ruimin Hu, and Xinbo Gao. Adv-diffusion: Imperceptible adversarial face identity attack via latent diffusion model. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(4):3585–3593, 2024. 2

[21] 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. 3, 6, 8

[22] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations, ICLR*, 2018. 3

[23] Yurii Nesterov. A method for unconstrained convex minimization problem with the rate of convergence o (1/k2). In *Dokl. Akad. Nauk. SSSR*, page 543, 1983. 2, 4, 5

[24] 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. 2, 3, 4, 6, 7, 8

[25] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986. 4

[26] 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. 6, 7

[27] 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. 6

[28] Chee-Wooi Ten, Govindarasu Manimaran, and Chen-Ching Liu. Cybersecurity for critical infrastructures: Attack and defense modeling. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 40(4):853–865, 2010. 2

[29] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *ICLR*, 2018. 6

[30] 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. 3, 6, 7

[31] Kunyu Wang, Xuanran He, Wenxuan Wang, and Xiaosen Wang. Boosting adversarial transferability by block shuffle and rotation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24336–24346, 2024. 3

[32] 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. 2, 3, 4, 6, 7, 8

[33] 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. 2, 3, 6, 8

[34] Xiaosen Wang, Zeliang Zhang, and Jianping Zhang. Structure invariant transformation for better adversarial transferability. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4607–4619, 2023. 3

[35] Han Wu, Syed Yunas, Sareh Rowlands, Wenjie Ruan, and Johan Wahlström. Adversarial driving: Attacking end-to-end autonomous driving. In *2023 IEEE Intelligent Vehicles Symposium (IV)*, pages 1–7. IEEE, 2023. 2

[36] Chaowei Xiao, Bo Li, Jun yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 3905–3911, 2018. 2

[37] 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. 2, 3, 6, 8

[38] Yinhu Xu, Qi Chu, Haojie Yuan, Zixiang Luo, Bin Liu, and Nenghai Yu. Enhancing adversarial transferability from the perspective of input loss landscape. In *International Conference on Image and Graphics*, pages 254–266. Springer, 2023. 2

[39] Yang Zhao, Hao Zhang, and Xiuyuan Hu. Penalizing gradient norm for efficiently improving generalization in deep learning. In *International Conference on Machine Learning*, pages 26982–26992. PMLR, 2022. 3

[40] 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. 2, 3, 4, 6, 7, 8

[41] Rongyi Zhu, Zeliang Zhang, Susan Liang, Zhuo Liu, and Chenliang Xu. Learning to transform dynamically for better adversarial transferability. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24273–24283, 2024. 3