Dual-Flow: Transferable Multi-Target, Instance-Agnostic Attacks via *In-the-wild* Cascading Flow Optimization

Yixiao Chen[1,*], **Shikun Sun**[1,*], **Jianshu Li**[2], **Ruoyu Li**[2], **Zhe Li**[2], **Junliang Xing**[1]
[1]Tsinghua University, [2]Ant Group
{chenyixi22, ssk21}@mails.tsinghua.edu.cn,
{jianshu.l, ruoyu.li, lizhe.lz}@antgroup.com, jlxing@tsinghua.edu.cn

Abstract

Adversarial attacks are widely used to evaluate model robustness, and in black-box scenarios, the transferability of these attacks becomes crucial. Existing generator-based attacks have excellent generalization and transferability due to their instance-agnostic nature. However, when training generators for multi-target tasks, the success rate of transfer attacks is relatively low due to the limitations of the model's capacity. To address these challenges, we propose a novel Dual-Flow framework for multi-target instance-agnostic adversarial attacks, utilizing Cascading Distribution Shift Training to develop an adversarial velocity function. Extensive experiments demonstrate that Dual-Flow significantly improves transferability over previous multi-target generative attacks. For example, it increases the success rate from Inception-v3 to ResNet-152 by 34.58%. Furthermore, our attack method shows substantially stronger robustness against defense mechanisms, such as adversarially trained models. The code of Dual-Flow is available at: https://github.com/Chyxx/Dual-Flow.

1 Introduction

Deep neural networks (DNNs) are highly vulnerable to adversarial attacks [53, 22, 3, 19], which can significantly compromise their reliability. Among these, targeted black-box attacks—where adversaries manipulate a model into misclassifying an input as a specific target class without direct access to the model—are the most challenging and impactful [1, 2, 39].

Adversarial attacks can be classified into instance-specific and instance-agnostic approaches. Instance-specific attacks [12, 67, 16] optimize perturbations for each input image using victim model gradients but often suffer from poor transferability. Instance-agnostic attacks [65, 42, 69, 17] generalize perturbations over the dataset, leading to stronger black-box transferability. These methods typically rely on universal adversarial perturbations [41, 70] or generative models [46, 43].

Generative model-based attacks can be further divided into single-target [42, 18, 61] and multi-target [69, 17] approaches. While single-target attacks achieve high success rates, they require training a separate model per target class, making them impractical for large-scale attacks. Multi-target attacks address this by conditioning a single generator on target labels but often suffer from reduced transferability and weak robustness against adversarial defenses.

Diffusion models [29, 51, 48] offer strong generative capabilities, making them a promising tool for adversarial attacks. However, current diffusion-based attacks are all instance-specific methods, which

[*]Equal contribution; the order of co-first authors is interchangeable.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

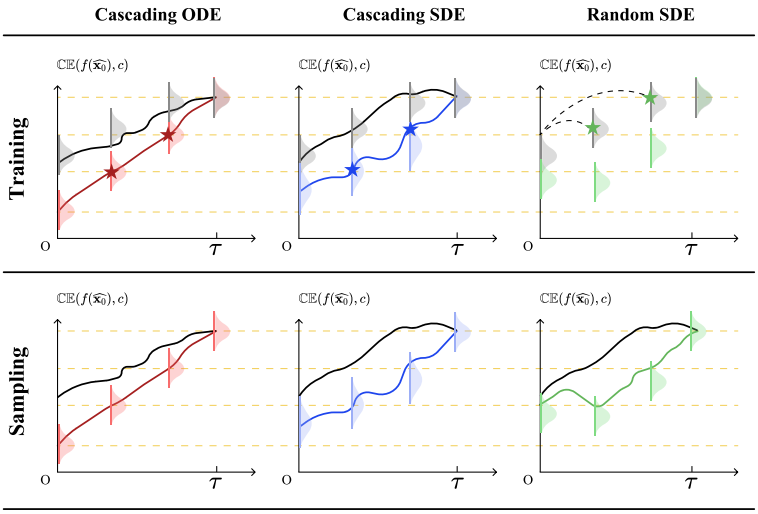


Figure 1: The comparison between Cascading ODE, Cascading SDE, and Random SDE for the second flow. The star shape represents the input for training the reverse flow. Notably, the Random SDE is observed to optimize in an incorrect distribution.

means they need to access the target model's gradient information during inference. Moreover, choosing between stochastic and deterministic sampling methods for adversarial perturbation generation remains an open challenge.

To address these challenges, we propose a novel **Dual-Flow framework** for multi-target instance-agnostic adversarial attacks. Our approach integrates (1) a pretrained diffusion model to generate an intermediate perturbation distribution as the forward flow and (2) a fine-tuned lightweight LoRA-based velocity function as reverse flow for targeted adversarial refinement. We introduce **Cascading Distribution Shift Training** to improve attack capability and employ **dynamic gradient clipping** to enforce the ℓ_∞ constraint.

As illustrated by the Cascading ODE and Cascading SDE in Figure 1, we first follow the black trajectory to introduce a slight perturbation to the image. Then, we follow either the red or blue trajectory to generate an altered image, effectively exploiting this process to attack the target model. Our main contributions are:

- **First application of flow-based ODE velocity training for adversarial attacks**, extending diffusion-based techniques beyond conventional score function training.
- **Dual-Flow algorithm**, integrating a pretrained diffusion-based forward ODE with a fine-tuned adversarial velocity function for structured perturbation generation.
- **Theoretical contribution on cascading improvement mechanism**, demonstrating how our method facilitates cascading improvements at later timesteps.

Extensive experiments demonstrate that our attack method achieves state-of-the-art black-box transferability in multi-target scenarios and exhibits high robustness against defense mechanisms.

2 Preliminary

2.1 Instance-Agnostic Attacks

Instance-agnostic attacks [39, 33, 60, 46, 69, 43, 42] learn perturbations based on data distributions rather than individual instances. These approaches, employing universal adversarial perturbations[41, 70] or generative models, have demonstrated superior transferability. This paper primarily focuses on the latter due to its greater flexibility and attack effectiveness.

Early generative model-based methods were primarily single-target attacks[65, 42, 43, 18, 61], requiring a separate model to be trained for each target class. Although these models exhibited high attack capabilities, the excessive training overhead limited their applicability when many target classes needed to be attacked. Recent research has proposed several multi-target attack methods[69, 17] that condition the perturbation generative model on class labels[69] or text embeddings[17] of classes. These approaches allow a single model to be trained to attack multiple target classes, significantly reducing the training overhead.

Consider a white-box image classifier characterized by the parameter θ, denoted as $f : \mathcal{X} \to \mathcal{Y}$, where the input space $\mathcal{X} \subset \mathbb{R}^{C\times H\times W}$ corresponds to the image domain, and the output space $\mathcal{Y} \subset \mathbb{R}^L$ represents the confidence scores across various classes. Here, L denotes the total number of classes. Given an original image $\mathbf{x} \in \mathcal{X}$ and a target class $c \in \mathcal{C}$, the goal of transferable multi-target generative attack is to generate the perturbation $\boldsymbol{\delta} = G(\mathbf{x}, c)$ and the pertubed image $\mathbf{x}^{\boldsymbol{\epsilon}} = \mathbf{x} + \boldsymbol{\delta}$, in such a way that an unseen victim model F predicts c for the perturbed image, *i.e.*, $\arg\max_{i\in\mathcal{C}} F(\mathbf{x}^{\boldsymbol{\epsilon}})_i = c$. Here G is the generator trained on the known source model f_θ. To ensure that the manipulated images remain visually indistinguishable from the originals, the perturbation is constrained using the l_∞ norm such that $\|\mathbf{x} - \mathbf{x}^{\boldsymbol{\epsilon}}\|_\infty = \|\boldsymbol{\delta}\|_\infty < \epsilon$.

2.2 Diffusion Models and Flow-based Models

Diffusion models [50, 29, 51, 52] have emerged as powerful generative models, particularly for continuous data such as audio [32] and images [48]. Recently, Flow-based generative models [36, 15, 37], developed directly from ordinary differential equations, have also gained significant momentum. Given their strong generative capabilities, exploring their applications in adversarial attacks is natural.

Sampling Algorithms. One of the most appealing aspects of diffusion models is the flexibility in designing the sampling process. The generation process of diffusion models primarily follows two formulations: one based on the Stochastic Differential Equation (SDE) and the other on the Ordinary Differential Equation (ODE). Each approach has strengths and weaknesses, making them suitable for different scenarios.

Diffusion Models in Adversarial Attack. Currently, diffusion models have found some applications in adversarial attacks. Some utilize diffusion models to create unrestricted adversarial examples[5, 8], while others perform instance-specific attacks[68, 4]. However, they all rely on iterative optimization using the classifier model's gradients while generating adversarial examples, thus not qualifying as instance-agnostic attacks.

3 Dual-Flow for Adversarial Attack

We propose a Dual-Flow pipeline designed to transform an image $\mathbf{x} \in \mathcal{X}$ through a perturbed distribution $\mathcal{X}_\tau$ and ultimately into a constrained output space $\mathcal{X}^\epsilon$, which is $\{\mathbf{x}^\epsilon | \exists \mathbf{x} \in \mathcal{X}, \|\mathbf{x}^\epsilon - \mathbf{x}\|_\infty < \epsilon\}$. By construction, $\mathcal{X}^\epsilon$ enforces a maximum ℓ_∞ perturbation of ϵ. Our method is instance-agnostic, which means that once training is completed, our model can generate adversarial examples during inference without requiring any access to the classifier model.

Specifically, we leverage the original ODE-based diffusion flow to map $\mathcal{X}$ to $\mathcal{X}_\tau$. Using a pretrained diffusion model's velocity function $\mathbf{v}_\phi(\cdot,\cdot)$ and a given input image $\mathbf{x} \sim \mathcal{X}$, we select a fixed timestep $\tau \in (0, 1)$. The perturbed image $\mathbf{x}_\tau \sim \mathcal{X}_\tau$ is obtained by integrating the following equation:

$$\frac{\partial}{\partial t}\Phi(\mathbf{x}, t) = \mathbf{v}_\phi(\Phi(\mathbf{x}, t), t), \quad \Phi(\mathbf{x}, 0) \sim \mathcal{X}, \tag{1}$$

from $t = 0$ to $t = \tau$.

To further map $\mathcal{X}_\tau$ to $\mathcal{X}^\epsilon$, we fine-tune a LoRA-based score function[30], yielding a new velocity function $\mathbf{v}_\theta$. Then by integrating another equation:

$$\frac{\partial}{\partial t}\Psi(\mathbf{x}, t) = \mathbf{v}_\theta(\Psi(\mathbf{x}, t), t), \quad \Psi(\mathbf{x}, \tau) \sim \mathcal{X}_\tau, \tag{2}$$

from $t = \tau$ to $t = 0$.

To train this second flow, we introduce a novel *Cascading Distribution Shift Training* strategy, which addresses the challenges posed by the inaccessibility of intermediate distributions during training.

Finally, to ensure that outputs remain within $\mathcal{X}^\epsilon$, we apply dynamic gradient stops during training, coupled with hard clipping operations at the final timestep. This approach allows for richer intermediate representations while maintaining the required perturbation bounds.

Details of our approach are provided in the following subsections.

3.1 A Construction of Better Extend Flow

Firstly, we construct an extended flow based on j, which is the negative value of the cross-entropy function:

$$j = -\mathbb{CE}(f(\mathbf{x}), c), \tag{3}$$

where f is the source model and c is the target label.

Proposition 1 (Morse Flow Construction)**.** *Under mild assumptions on the $\mathcal{X}^\epsilon$ and the function j, there exists $\epsilon > 0$, a unique smooth flow.*

$$\Phi : \mathcal{X}^\epsilon \times [0, \epsilon] \to \mathcal{X}^\epsilon, \tag{4}$$

satisfying:

$$\begin{aligned} \frac{d}{dt}\Phi(\mathbf{x}, t) &= \mathbf{v}(\Phi(\mathbf{x}, t)), \\ \Phi(\mathbf{x}, 0) &= \mathbf{x}, \end{aligned} \tag{5}$$

such that:

1. $\mathbf{v} = \alpha(\mathbf{x})\nabla_{\mathbf{X}} j(\mathbf{x})$ *almost everywhere*
2. $j(\Phi(\mathbf{x}, \epsilon)) \geq j(\mathbf{x})$ *for all $x \in \mathcal{X}^\epsilon$, and $>$ holds almost everywhere if j is not trivial*
3. *Each $\Phi(\cdot, t) : \mathcal{X}^\epsilon \to \mathcal{X}^\epsilon$ is a diffeomorphism*

A detailed proof is provided in Appendix A.1. Proposition 1 indicates we can find better extend flow to hack j by ∇j. In the following paragraphs, we will construct a concrete algorithm to realize it along with an existing flow.

3.2 Cascading Distribution Shift Training

Although the *in-the-wild* ODE trajectory is deterministic, obtaining exact intermediate samples remains a significant challenge, which poses difficulties for our approach. To address this issue, we propose the **Cascading Distribution Shift Training Algorithm**, specifically designed to enhance adversarial attack efficacy through two key mechanisms: (1) Enforcing a cascading hacking effect, where each perturbation step incrementally contributes to misleading the source model. (2) Ensuring the final perturbation follows the prescribed ℓ_∞ constraint. Further details are provided in Algorithm 1, where l denotes the conditional input in real diffusion models.

Our proposed training framework not only circumvents the challenge of inaccessible intermediate timesteps but also offers additional advantages. Proposition 2 formalizes that our algorithm progressively refines a coarse-to-fine representation, thus effectively leveraging information from ahead timesteps. More concretely, due to the continuity of the ODE, our training algorithm enables a cascading optimization within an increasingly refined space.

Theorem 2 (Cascading Improvement at Adjoint Timesteps)**.** *Consider two consecutive timesteps $t, t-\delta$. Following Algorithm 1, when comparing the cases with and without updating θ at t, updating θ results in an equal or lower cross-entropy for $\widehat{\mathbf{x}_0}$ at $t-\delta$ when δ is sufficiently small and all functions are smooth.*

One crucial consideration is constraining the final result within $\mathcal{X}^\epsilon$. There are two primary methods to achieve this. The first approach incorporates the original ODE trajectory during model tuning,

Algorithm 1 Cascading Distribution Shift Training

Input: $\tau = N\delta$, stepsize δ, model param. ϕ, θ, source model f, target labels set $\mathcal{C}$, training dataset $\{\mathbf{I}^i\}_{i\in\mathcal{I}}$, learning rate l_r
Initialize $\theta = \phi$.
repeat
  for $i \in \mathcal{I}$ **do**
    get $\mathbf{x}_0 = \mathbf{I}^i$
    sample $c \sim \mathcal{C}$
    for $t = 1$ **to** N **do**
      $\mathbf{x}_{t\delta} = \mathbf{x}_{(t-1)\delta} + \mathbf{v}_\phi(\mathbf{x}_{(t-1)\delta}, (t-1)\delta, \varnothing)\delta$
    end for
    for $t = N$ **to** 1 **do**
      $\mathbf{x}_{(t-1)\delta} = \mathbf{x}_{t\delta} - \mathbf{v}_\theta(\mathbf{x}_{t\delta}, t\delta, c)\delta$
      $\widehat{\mathbf{x}_0} = \mathbf{x}_{t\delta} - \mathbf{v}_\theta(\mathbf{x}_{t\delta}, t\delta, c)t\delta$
      $\widehat{\mathbf{x}_0} = \text{clip}\,(\widehat{\mathbf{x}_0}, \mathbf{x} - \epsilon, \mathbf{x} + \epsilon)$
      $\theta = \theta - l_r \cdot \nabla_\theta(\mathbb{CE}(f(\widehat{\mathbf{x}_0}), c))$
    end for
  end for
until $\mathbf{v}_\theta$ convergence
Return: Dual-Flow $\{\mathbf{v}_\phi, \mathbf{v}_\theta\}$

Algorithm 2 Dual-Flow Sampling

Input: $\tau = N\delta$, stepsize δ, image $\mathbf{I}$, target label c, Dual-Flow $\{\mathbf{v}_\phi, \mathbf{v}_\theta\}$
$\mathbf{x} = \mathbf{I}$.
for $t = 1$ **to** N **do**
  $\mathbf{x}_{t\delta} = \mathbf{x}_{(t-1)\delta} + \mathbf{v}_\phi(\mathbf{x}_{(t-1)\delta}, (t-1)\delta, \varnothing)\delta$
end for
for $t = N$ **to** 1 **do**
  $\mathbf{x}_{(t-1)\delta} = \mathbf{x}_{t\delta} - \mathbf{v}_\theta(\mathbf{x}_{t\delta}, t\delta, c)\delta$
end for
$\mathbf{x}_0 = \text{clip}\,(\mathbf{x}_0, \mathbf{x} - \epsilon, \mathbf{x} + \epsilon)$
Return: $\mathbf{x}_0$

ensuring the output remains close to the original ODE flow. The second approach enforces the ℓ_∞ constraint or applies gradient clipping to suppress the influence of out-of-range image regions, guaranteeing that only in-range rewards contribute to model optimization. Our experiments show that dynamic gradient clipping yields the best performance among these methods.

Given the fixed model capacity, the Cascading Distribution Shift Training algorithm ensures greater consistency between the training and sampling processes, improving performance. This is visually illustrated in Figure 1.

3.3 Dual-Flow Sampling

During the sampling process, following the proposed training algorithm, a given image $\mathbf{x} \in \mathcal{X}$ is first mapped to $\mathbf{x}_\tau$ via Eq. (1). Subsequently, it is transformed into an intermediate state $\mathbf{x}'_0$ using Eq. (2). Finally, a hard truncation is applied to obtain the qualified perturbed sample $\mathbf{x}_0 \in \mathcal{X}^\epsilon$.

3.4 Deterministic Flow vs. Stochastic Flow

An important consideration is the choice between a deterministic flow, modeled by an ODE, and a stochastic flow, modeled by an SDE. This decision primarily depends on the second flow, as the first is inherited from a pretrained diffusion model.

When translating the distribution $\mathcal{X}^\tau$ to $\mathcal{X}^\epsilon$, a simple rescaling and noise injection into $\mathcal{X}^\epsilon$ is insufficient to fully recover $\mathcal{X}^\tau$. Consequently, this transformation falls outside the standard diffusion-based framework.

Table 1: Attack success rates (%) for multi-target attacks on normally trained models using the ImageNet NeurIPS validation set. The perturbation budget is constrained to $l_\infty \leq 16/255$. * indicates white-box attacks. The results are averaged across 8 different target classes, and the overall average on the far right is computed solely for black-box attacks.

Source	Method	Inc-v3	Inc-v4	Inc-Res-v2	Res-152	DN-121	GoogleNet	VGG-16	Average
Inc-v3	MIM	99.90*	0.80	1.00	0.40	0.20	0.20	0.30	0.48
	TI-MIM	98.50*	0.50	0.50	0.30	0.20	0.40	0.40	0.38
	SI-MIM	99.80*	1.50	2.00	0.80	0.70	0.70	0.50	1.03
	DIM	95.60*	2.70	0.50	0.80	1.10	0.40	0.80	1.05
	TI-DIM	96.00*	1.10	1.20	0.50	0.50	0.50	0.80	0.77
	SI-DIM	90.20*	3.80	4.40	2.00	2.20	1.70	1.40	2.58
	Logit	99.60*	5.60	6.50	1.70	3.00	0.80	1.50	3.18
	SU	99.59*	5.80	7.00	3.35	3.50	2.00	3.94	4.26
	C-GSP	93.40*	66.90	66.60	41.60	46.40	40.00	45.00	51.08
	CGNC	96.03*	59.43	48.06	42.48	62.98	51.33	52.54	52.80
	Dual-Flow	90.08*	**77.19**	**66.76**	**77.06**	**82.64**	**73.01**	**67.09**	**73.96**
Res-152	MIM	0.50	0.40	0.60	99.70*	0.30	0.30	0.20	0.38
	TI-MIM	0.30	0.30	0.30	96.50*	0.30	0.40	0.30	0.32
	SI-MIM	1.30	1.20	1.60	99.50*	1.00	1.40	0.70	1.20
	DIM	2.30	2.20	3.00	92.30*	0.20	0.80	0.70	1.53
	TI-DIM	0.80	0.70	1.00	90.60*	0.60	0.80	0.50	0.73
	SI-DIM	4.20	4.80	5.40	90.50*	4.20	3.60	2.00	4.03
	Logit	10.10	10.70	12.80	95.70*	12.70	3.70	9.20	9.87
	SU	12.36	11.31	16.16	95.08*	16.13	6.55	14.28	12.80
	C-GSP	37.70	47.60	45.10	93.20*	64.20	41.70	45.90	47.03
	CGNC	53.39	51.53	34.24	95.85*	85.66	62.23	63.36	58.40
	Dual-Flow	**69.58**	**71.92**	**56.10**	92.39*	**85.73**	**73.65**	**67.59**	**70.76**

A natural solution is to construct an *in-the-wild* ODE (as we do) or SDE that maps $\mathcal{X}^\tau$ to $\mathcal{X}^\epsilon$. In the ODE formulation, we define a velocity function $\mathbf{v}_\theta$ that satisfies Eq. (2). A similar approach applies to the SDE case, where stochastic noise facilitates the distribution transition.

When comparing from the perspectives of randomness and determinism, as shown in Figure 1, we label our framework as **Cascading ODE** and implement two SDE-based training algorithms. One injects noise at a random timestep within $(0, \tau)$, labeled as **Random SDE**, while the other first directly adds noise at τ and then reverses the flow using DDPM, resulting in a weak cascading relationship, labeled as **Cascading SDE**. While SDE-based training algorithms more closely resemble original diffusion models, they present two key challenges.

First, **Cascading SDE** introduces a random term, which may make it more difficult to construct the Cascading Improvement relationship as stated in Proposition 2 for **Cascading ODE**. Second, sampling with SDEs tends to produce unstable results, where larger step sizes exacerbate accumulated errors, further impacting reliability, as demonstrated in our experiments.

As for **Random SDE**, it exhibits the worst performance because when sampling $\mathbf{x}_t$ by directly adding noise, the distribution of $\mathbf{x}_t$ remains unchanged. Consequently, even slight training of the reverse flow leads to a distribution mismatch, as illustrated by the star in the last column of Figure 1.

4 Experiments

4.1 Experimental Settings

Dataset. Following [69, 18, 17], we train the model on the ImageNet training set[10] and evaluate the attack performance using ImageNet-NeurIPS (1k) dataset proposed by NeurIPS 2017 adversarial competition[44].

Victim Models. We consider various naturally trained models, including Inception-v3 (Inc-v3) [55], Inception-v4 (Inc-v4) [56], Inception-ResNet-v2 (Inc-Res-v2) [56], ResNet-152 (Res-152) [27], DenseNet-121 (DN-121) [31], GoogleNet [54], and VGG-16 [49].

Table 2: Attack success rates (%) for single-target attacks against normally trained models on ImageNet NeurIPS validation set. Note that CGNC† and Dual-Flow† denote the single-target variants of CGNC and our proposed Dual-Flow, respectively. The perturbation budget is constrained to $l_\infty \leq 16/255$. * indicates white-box attacks. The results are averaged across 8 different target classes, and the overall average on the far right is computed solely for black-box attacks.

Source	Method	Inc-v3	Inc-v4	Inc-Res-v2	Res-152	DN-121	GoogleNet	VGG-16	Average
Inc-v3	GAP	86.90*	45.06	34.48	34.48	41.74	26.89	34.34	36.16
	CD-AP	94.20*	57.60	60.10	37.10	41.60	32.30	41.70	45.07
	TTP	91.37*	46.04	39.37	16.40	33.47	25.80	25.73	31.14
	DGTA-PI	94.63*	67.95	55.03	50.50	47.38	47.67	48.11	52.77
	CGNC†	98.84*	74.76	64.48	62.00	78.94	69.06	70.74	70.00
	Dual-Flow	90.08*	77.19	66.76	77.06	82.64	73.01	67.09	73.96
	Dual-Flow†	91.41*	**78.85**	**70.59**	**79.12**	**83.36**	**77.52**	**71.29**	**76.79**
Res-152	GAP	30.99	31.43	20.48	84.86*	58.35	29.89	39.70	35.14
	CD-AP	33.30	43.70	42.70	96.60*	53.80	36.60	34.10	40.70
	TTP	62.03	49.20	38.70	95.12*	82.96	65.09	62.82	60.13
	DGTA-PI	66.83	53.62	47.61	96.48*	86.61	68.29	69.58	65.42
	CGNC†	68.86	69.45	45.71	98.61*	**91.14**	69.83	68.05	68.84
	Dual-Flow	69.58	71.92	56.10	92.39*	85.73	73.65	67.59	70.76
	Dual-Flow†	**72.25**	**74.35**	**58.44**	93.65*	87.61	**75.45**	**71.11**	**76.12**

Table 3: Attack success rates (%) for multi-target attacks against robust models on ImageNet NeurIPS validation set. The perturbation budget $l_\infty \leq 16/255$. The results are averaged on 8 different target classes.

Source	Method	Inc-v3$_{adv}$	IR-v2$_{ens}$	Res50$_{SIN}$	Res50$_{IN}$	Res50$_{fine}$	Res50$_{Aug}$	Average
Inc-v3	MIM	0.16	0.10	0.20	0.27	0.44	0.19	0.23
	TI-MIM	0.21	0.19	0.33	0.49	0.68	0.31	0.37
	SI-MIM	0.13	0.19	0.26	0.43	0.63	0.29	0.32
	DIM	0.11	0.09	0.16	0.33	0.39	0.19	0.21
	TI-DIM	0.15	0.13	0.16	0.21	0.33	0.14	0.19
	SI-DIM	0.19	0.21	0.43	0.71	0.84	0.46	0.47
	Logit	0.30	0.30	0.70	1.23	3.14	0.86	1.09
	SU	0.49	0.41	0.84	1.75	3.55	1.04	1.35
	C-GSP	20.41	18.04	6.96	33.76	44.56	21.95	24.28
	CGNC	24.36	22.54	8.85	40.83	52.18	22.85	28.60
	Ours	**51.54**	**55.62**	**45.86**	**74.56**	**78.54**	**67.56**	**62.28**
Res-152	MIM	0.19	0.15	0.28	1.58	2.75	0.78	0.96
	TI-MIM	0.61	0.73	0.50	2.51	4.75	1.76	1.81
	SI-MIM	0.24	0.24	0.39	0.66	0.84	0.36	0.46
	DIM	0.63	0.37	0.94	8.50	14.22	3.77	4.74
	TI-DIM	0.23	0.30	0.28	0.76	1.49	0.49	0.59
	SI-DIM	0.71	0.71	0.75	2.73	3.89	1.37	1.69
	Logit	1.15	1.18	1.65	6.70	15.46	5.93	5.34
	SU	2.12	1.20	1.95	7.53	21.14	6.95	6.82
	C-GSP	14.60	16.01	16.84	60.30	65.51	42.88	36.02
	CGNC	22.21	26.71	29.83	79.80	84.05	63.75	51.06
	Ours	**44.50**	**54.09**	**59.35**	**83.05**	**84.28**	**76.35**	**66.94**

For further evaluation, we also analyze the performance of our method on robustly trained models. These include adv-Inception-v3 (Inc-v3$_{adv}$) [22], ens-adv-Inception-ResNet-v2 (IR-v2$_{ens}$) [26], and several robustly trained ResNet-50 models. The ResNet-50 variants are: Res50$_{SIN}$ (trained on stylized ImageNet), Res50$_{IN}$ (trained on a mixture of stylized and Nature ImageNet), Res50$_{fine}$ (further fine-tuned with an auxiliary dataset [21]), and Res50$_{Aug}$ (trained with advanced data augmentation techniques from Augmix [28]).

Baseline Methods. We compare our attack with several attack methods. For instance-specific attacks, we consider MIM [12], DIM [66], SIM [35], DIM [13], Logit [71], and SU [62]. For instance-agnostic attacks, we consider C-GSP [69], CGNC [17], GAP [46], CD-AP [42], TTP [43], and DGTA-PI[18]. Among them, C-GSP [69]and CGNC [17]are multi-target generative attacks, and the others are single-target generative attacks. For SU attack [62], we choose to compare with its best version DTMI-Logit-SU. For CGNC [17], we also consider its single target variant and compare it to a single target attack method.

Implementation Details. We adopt stable-diffusion [48] as our pre-trained diffusion model. We set $\tau = 0.25$ and $N = 6$ for training and testing. The LoRA rank is 16.

Following previous work [69, 18, 17], we choose Res-152 and Inc-v3 as source models to train our model. The perturbation budget ϵ is 16/255. We conduct 50k steps of training for multi-target tasks. To compare our method with other single-target attacks, we further fine-tune our model for an additional 10k steps to specialize in a single target class(more details provided in Appendix D.2). For multi-target training, we use a learning rate of 2.5×10^{-5} and a total batch size of 8 (distributed across two NVIDIA RTX 3090 GPUs, each with 24GB memory and batch size 4). Training under this setting takes approximately one day to complete. For single-target fine-tuning, we set the learning rate to 1×10^{-5} and a batch size of 4, conducted on a single NVIDIA RTX 3090 GPU, which requires approximately 4 hours. In total, the experiments involve approximately 160 GPU hours.

4.2 Transferability Evaluation

We assess the effectiveness of our proposed Dual-Flow for black-box target attacks through a series of experiments. To ensure consistency with previous work [69, 18, 17], we select eight distinct target classes [70] to conduct the target black-box attack testing protocol. We use the average attack success rate (ASR) across 8 target classes as an evaluation metric.

Multi-Target Black-Box Attack. We initially conduct attacks on normally trained models to evaluate the performance of multi-target attacks. The results in Table 1 show that our proposed Dual-Flow method exhibits significantly superior transferability, outperforming state-of-the-art instance-specific and instance-agnostic methods. Specifically, our method achieves an average ASR improvement of 21.16% and 12.36% over CGNC [17] using Inc-v3 and Res-152 as source models, respectively, on black-box models. Notably, instance-specific methods, despite higher success rates in white-box settings, tend to overfit the source models' classification boundaries, resulting in poor performance when transferred to black-box models.

Single-Target Black-Box Attack. To further evaluate our method's effectiveness, we compare it with other state-of-the-art instance-agnostic single-target attacks. Multi-target attacks are inherently more challenging than single-target ones, disadvantaging our model in such comparisons. To ensure fairness, we applied a masked fine-tuning technique similar to CGNC [17], allowing us to fine-tune our model separately for each target class and create single-target variants.

The results in Table 2 show that after fine-tuning, Dual-Flow† achieves higher attack success rates and generally outperforms leading single-target methods. Notably, our method excels in average black-box attack capability even without individual fine-tuning for the eight target classes. This demonstrates our approach's significant capacity and effectiveness in multi-target attacks, reducing the need for separate models for each target class in resource-constrained scenarios.

4.3 Attack Under Defense Strategies

To demonstrate the robustness of our proposed Dual-Flow, we evaluate its performance against several widely used defense mechanisms.

Robustly Trained Networks. We first consider attacking six robustly trained networks, with results in Table 3. Attacking robustness-augmented models is challenging, as previous methods see a significant drop in success rates. However, our approach consistently misleads black-box classifiers into predicting the specified classes, showing marked improvement over earlier multi-target methods. Notably, using Inc-v3 as the source model, the average attack success rate against the six robust models increases significantly from 28.60% to 62.28%, highlighting our method's effectiveness.

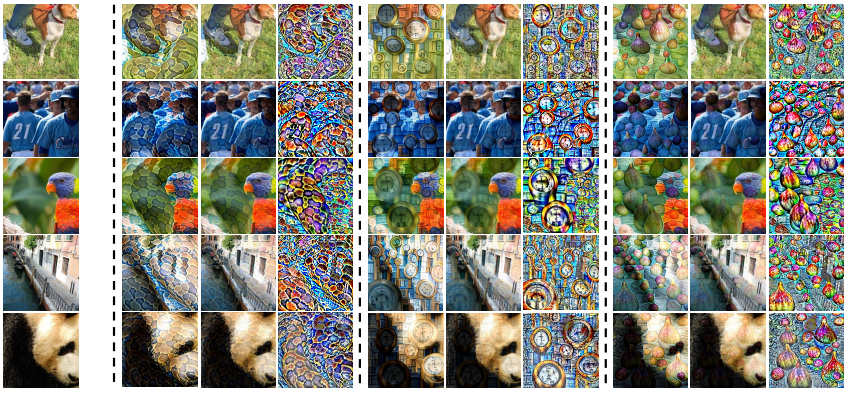
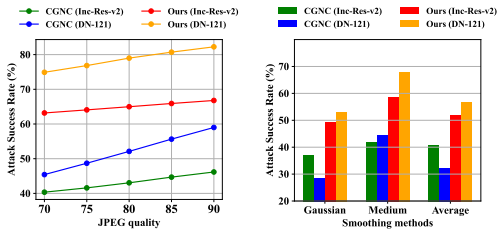


Figure 2: Visualization results of different input images targeting various classes. For each text prompt of the target class, the left column displays the adversarial examples generated before clipping, the middle column shows the adversarial examples after clipping, and the right column presents the corresponding adversarial perturbations, which represent the differences between the clipped adversarial examples and the original images. Note that the perturbations are scaled to a range between 0 and 1. The source model used is Inc-v3.

Input Process Defense. We compared our method's performance with the state-of-the-art multi-target attack method CGNC against input preprocessing defenses, such as image smoothing [11] and JPEG compression [14]. As shown in Figure 3, our method consistently outperforms CGNC under these defenses. For example, using Inc-v3 as the source model and DN-121 as the target model, our method achieves a 52.99% success rate under Gaussian smoothing, compared to CGNC's 28.27%. This highlights the superior effectiveness of our approach in overcoming input preprocessing defenses.



(a) JPEG compression (b) Input smoothing

Figure 3: A comparison of CGNC and our method regarding attack success rates against various input processing defense strategies. The results against JPEG compression are shown in (a), while (b) presents the outcomes against different input smoothing methods. Inc-v3 is the source model and Inc-Res-v2, along with DN-121, are the target models.

4.4 Visualization

To gain a deeper understanding of the effectiveness of our method, we visualized both the unclipped and clipped samples generated by our approach. Additionally, for consistency with other perturbation-based attack methods, we visualized the equivalent adversarial perturbations, defined as the pixel differences between the clipped samples and the clean images. As illustrated in Figure 2, our method first transforms the original image into one that maintains a similar layout and color scheme but becomes semantically closer to the target class. This transformed image is subsequently clipped to ensure its pixel differences from the original image remain within the epsilon bound. Visually, the clipped image retains substantial semantic features of the target class. Notably, the adversarial perturbations also exhibit distinct semantic patterns aligned with the target class, further validating the effectiveness of our approach.

4.5 Ablation Study

To further validate the effectiveness of our chosen Cascading ODE, we conducted a series of ablation experiments in this section. Here, Dual-Flow-co represents the original method, Dual-Flow-cs denotes the Cascading SDE variant, and Dual-Flow-rs denotes the Random SDE variant. During the reverse process at inference time, these variants em-

ploy either the DDPM scheduler or the DDIM scheduler. As shown in Table 4, our method significantly outperforms the other variants in white-box and black-box transfer attacks.

Furthermore, we compared the impact of different sampling steps N during the reverse process. As shown in Figure 4, increasing the sampling steps steadily increases success rates for both Dual-Flow-co and Dual-Flow-cs. However, for Dual-Flow-rs, the success rate quickly declines as the inference steps increase, supporting our analyses in Section 3.4 and Figure 1.

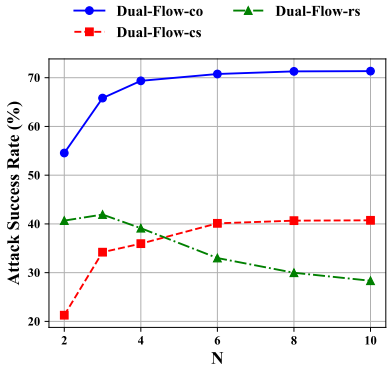


Figure 4: The multi-target black-box attack success rates of several variants of our method. The source model used is Res-152.

5 Discussion

Potential Advantage of SDE. Although our experiments with Cascading SDE have not yet surpassed the performance of Cascading ODE, we believe that methods based on Schrödinger bridges [9] have the potential to bring significant improvements. Schrödinger bridge formulations provide a principled way to learn stochastic transport maps, which could offer better control over the reverse trajectory and enhance the stability of the cascading distribution shift.

Table 4: The multi-target attack success rates of several variants of our method. The source model used is Res-152. The white-box attack success rate refers to the performance on Res-152, while the black-box attack success rate represents the average performance across six black-box models.

Method	White-box	Black-box
Dual-Flow-co	92.39	70.76
Dual-Flow-cs + DDIM	68.56	40.12
Dual-Flow-cs + DDPM	74.58	46.04
Dual-Flow-rs + DDIM	55.39	33.00
Dual-Flow-rs + DDPM	29.19	14.86

6 Conclusion

We have introduced Dual-Flow, a novel framework for highly transferable multi-target adversarial attacks. By employing Cascading Distribution Shift Training to develop an adversarial velocity function, our approach addresses the limitations of existing methods. Extensive experimental results demonstrate that Dual-Flow achieves remarkable improvements in transferability and robustness compared to previous multi-target generative attacks. These findings highlight the potential of Dual-Flow as a powerful tool for evaluating and improving the robustness of generative models.

Acknowledgments and Disclosure of Funding

This work was supported in part by the Natural Science Foundation of China under Grant No. 62222606 and the Ant Group Security and Risk Management Fund. We sincerely thank Qixin Wang for her assistance in creating the figures for this paper.

References

[1] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *European conference on computer vision*, pp. 484–501. Springer, 2020.

[2] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *International conference on machine learning*, pp. 284–293. PMLR, 2018.

[3] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pp. 39–57. Ieee, 2017.

[4] Jianqi Chen, Hao Chen, Keyan Chen, Yilan Zhang, Zhengxia Zou, and Zhenwei Shi. Diffusion models for imperceptible and transferable adversarial attack. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[5] Xinquan Chen, Xitong Gao, Juanjuan Zhao, Kejiang Ye, and Cheng-Zhong Xu. Advdiffuser: Natural adversarial example synthesis with diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4562–4572, 2023.

[6] Zhaoyu Chen, Bo Li, Shuang Wu, Kaixun Jiang, Shouhong Ding, and Wenqiang Zhang. Content-based unrestricted adversarial attack. *Advances in Neural Information Processing Systems*, 36: 51719–51733, 2023.

[7] Zhengsu Chen, Lingxi Xie, Jianwei Niu, Xuefeng Liu, Longhui Wei, and Qi Tian. Visformer: The vision-friendly transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 589–598, 2021.

[8] Xuelong Dai, Kaisheng Liang, and Bin Xiao. Advdiff: Generating unrestricted adversarial examples using diffusion models. In *European Conference on Computer Vision*, pp. 93–109. Springer, 2025.

[9] Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling. *Advances in Neural Information Processing Systems*, 34:17695–17709, 2021.

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

[11] Gavin Weiguang Ding, Luyu Wang, and Xiaomeng Jin. Advertorch v0. 1: An adversarial robustness toolbox based on pytorch. *arXiv preprint arXiv:1902.07623*, 2019.

[12] 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*, pp. 9185–9193, 2018.

[13] 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*, pp. 4312–4321, 2019.

[14] 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.

[15] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

[16] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1625–1634, 2018.

[17] Hao Fang, Jiawei Kong, Bin Chen, Tao Dai, Hao Wu, and Shu-Tao Xia. Clip-guided generative networks for transferable targeted adversarial attacks. In *European Conference on Computer Vision*, pp. 1–19. Springer, 2025.

[18] Weiwei Feng, Nanqing Xu, Tianzhu Zhang, and Yongdong Zhang. Dynamic generative targeted attacks with pattern injection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16404–16414, 2023.

[19] Kuofeng Gao, Yang Bai, Jiawang Bai, Yong Yang, and Shu-Tao Xia. Adversarial robustness for visual grounding of multimodal large language models. In *ICLR Workshop*, 2024.

[20] Lianli Gao, Yaya Cheng, Qilong Zhang, Xing Xu, and Jingkuan Song. Feature space targeted attacks by statistic alignment. *arXiv preprint arXiv:2105.11645*, 2021.

[21] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[22] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[23] Benjamin Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Herve Jegou, and Matthijs Douze. Levit: A vision transformer in convnet's clothing for faster inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12259–12269, October 2021.

[24] Jiangfan Han, Xiaoyi Dong, Ruimao Zhang, Dongdong Chen, Weiming Zhang, Nenghai Yu, Ping Luo, and Xiaogang Wang. Once a man: Towards multi-target attack via learning multi-target adversarial network once. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5158–5167, 2019.

[25] Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer. *Advances in neural information processing systems*, 34:15908–15919, 2021.

[26] Jie Hang, Keji Han, Hui Chen, and Yun Li. Ensemble adversarial black-box attacks against deep learning systems. *Pattern Recognition*, 101:107184, 2020.

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pp. 630–645. Springer, 2016.

[28] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

[29] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[30] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[31] 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*, pp. 4700–4708, 2017.

[32] Rongjie Huang, Jiawei Huang, Dongchao Yang, Yi Ren, Luping Liu, Mingze Li, Zhenhui Ye, Jinglin Liu, Xiang Yin, and Zhou Zhao. Make-an-audio: Text-to-audio generation with prompt-enhanced diffusion models. *arXiv preprint arXiv:2301.12661*, 2023.

[33] Zelun Kong, Junfeng Guo, Ang Li, and Cong Liu. Physgan: Generating physical-world-resilient adversarial examples for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14254–14263, 2020.

[34] Qizhang Li, Yiwen Guo, and Hao Chen. Yet another intermediate-level attack. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pp. 241–257. Springer, 2020.

[35] Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E Hopcroft. Nesterov accelerated gradient and scale invariance for adversarial attacks. *arXiv preprint arXiv:1908.06281*, 2019.

[36] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[37] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[38] Yantao Lu, Yunhan Jia, Jianyu Wang, Bai Li, Weiheng Chai, Lawrence Carin, and Senem Velipasalar. Enhancing cross-task black-box transferability of adversarial examples with dispersion reduction. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 940–949, 2020.

[39] Jinqi Luo, Tao Bai, and Jun Zhao. Generating adversarial yet inconspicuous patches with a single image (student abstract). In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[40] TorchVision maintainers and contributors. Torchvision: Pytorch's computer vision library. `https://github.com/pytorch/vision`, 2016.

[41] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1765–1773, 2017.

[42] Muhammad Muzammal Naseer, Salman H Khan, Muhammad Haris Khan, Fahad Shahbaz Khan, and Fatih Porikli. Cross-domain transferability of adversarial perturbations. *Advances in Neural Information Processing Systems*, 32, 2019.

[43] Muzammal Naseer, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Fatih Porikli. On generating transferable targeted perturbations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7708–7717, 2021.

[44] NeurIPS. `https://www.kaggle.com/c/nips-2017-defense-against-adversarial-attack/data`. Kaggle, 2017.

[45] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification. In *International Conference on Machine Learning (ICML)*, 2022.

[46] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4422–4431, 2018.

[47] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

[48] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

[49] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[50] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

[51] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[52] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[53] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[54] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[55] 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*, pp. 2818–2826, 2016.

[56] 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*, volume 31, 2017.

[57] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention, 2021.

[58] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Herv'e J'egou. Going deeper with image transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 32–42, October 2021.

[59] 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*, pp. 1924–1933, 2021.

[60] Xiaosen Wang, Kun He, and John E Hopcroft. At-gan: A generative attack model for adversarial transferring on generative adversarial nets. *arXiv preprint arXiv:1904.07793*, 3(4):3, 2019.

[61] Zhibo Wang, Hongshan Yang, Yunhe Feng, Peng Sun, Hengchang Guo, Zhifei Zhang, and Kui Ren. Towards transferable targeted adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20534–20543, 2023.

[62] Zhipeng Wei, Jingjing Chen, Zuxuan Wu, and Yu-Gang Jiang. Enhancing the self-universality for transferable targeted attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12281–12290, 2023.

[63] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

[64] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision, 2020.

[65] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018.

[66] 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*, pp. 2730–2739, 2019.

[67] Yifeng Xiong, Jiadong Lin, Min Zhang, John E Hopcroft, and Kun He. Stochastic variance reduced ensemble adversarial attack for boosting the adversarial transferability. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14983–14992, 2022.

[68] Haotian Xue, Alexandre Araujo, Bin Hu, and Yongxin Chen. Diffusion-based adversarial sample generation for improved stealthiness and controllability. *Advances in Neural Information Processing Systems*, 36:2894–2921, 2023.

[69] Xiao Yang, Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Boosting transferability of targeted adversarial examples via hierarchical generative networks. In *European Conference on Computer Vision*, pp. 725–742. Springer, 2022.

[70] Chaoning Zhang, Philipp Benz, Tooba Imtiaz, and In So Kweon. Understanding adversarial examples from the mutual influence of images and perturbations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14521–14530, 2020.

[71] Zhengyu Zhao, Zhuoran Liu, and Martha Larson. On success and simplicity: A second look at transferable targeted attacks. *Advances in Neural Information Processing Systems*, 34: 6115–6128, 2021.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims made in the abstract and introduction are supported in Section 4.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss limitations in the Appendix.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: In the Appendix we provide the complete proofs.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide full experimental details for reproducing the main results in Section 4 and Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We will release our data, models and codebase in the final version of the paper.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide full details in Section 4 and Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide statistical significance information in Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide sufficient information on the computer resources in Section 4 and Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We provide discussion about potential societal impacts in Appendix.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The benchmarks and data splits are publicly available. All licenses are respected.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Assets will be released, and all instructions and details will be included for reproduction.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Proofs

A.1 Proof of Morse Flow Construction

Proposition 3 (Morse Flow Construction)**.** *Let $B \subset \mathbb{R}^n$ be a bounded open set with smooth boundary, and let $j : B \to \mathbb{R}$ be a smooth Morse function that extends to $C^\infty(\overline{B})$. There exists $\varepsilon > 0$, a smooth vector field $X \in \mathfrak{X}(B)$, and a unique smooth flow*

$$\Phi : B \times [0, \varepsilon] \to B$$

satisfying:

$$\frac{d}{dt}\Phi(x,t) = X(\Phi(x,t)),$$
$$\Phi(x,0) = x,$$

such that:

1. *$j(\Phi(x,\varepsilon)) \geq j(x)$ for all $x \in B$*
2. *Each $\Phi(\cdot,t) : B \to B$ is a diffeomorphism*
3. *Trajectories remain bounded away from ∂B for $t \in [0,\varepsilon]$*

Constructive Proof. We proceed through coordinated geometric and analytic constructions.

Step 1: Geometric Preparations

1. *Smooth Defining Function*: By the smooth boundary assumption, there exists $\mu \in C^\infty(\overline{B}, [0,\infty))$ with:
   - $\mu^{-1}(0) = \partial B$
   - $\nabla\mu(x) \neq 0$ for $x \in \partial B$
   - $\mu(x) \sim \text{dist}(x, \partial B)$ near ∂B

   For explicit construction, take $\mu(x) = f(\text{dist}(x, \partial B))$ where $f \in C^\infty([0,\infty))$ satisfies $f(r) = r$ near 0.

2. *Critical Point Isolation*: Since j is Morse on compact $\overline{B}$, its critical points $\mathscr{C}(j) = \{p_1, \ldots, p_N\}$ are finite and non-degenerate. Choose pairwise disjoint neighborhoods $U_i \ni p_i$ with:
$$\overline{U_i} \subset B \setminus \partial B \quad \text{and} \quad \overline{U_i} \cap \mathscr{C}(j) = \{p_i\}$$

Step 2: Vector Field Construction

1. *Partition of Unity*: Let $\{\rho_i\}_{i=1}^N$ be smooth functions with:
$$\text{supp}(\rho_i) \subset U_i, \quad 0 \leq \rho_i \leq 1, \quad \sum_{i=1}^N \rho_i \leq 1$$
   Define the cutoff function:
$$\eta(x) := 1 - \sum_{i=1}^N \rho_i(x)$$
   Note $\eta \equiv 0$ near critical points and $\eta \equiv 1$ outside $\bigcup U_i$.

2. *Decay Modulation*: Fix $m \geq n+1$. Define the boundary decay factor:
$$\mu_m(x) := \mu(x)^m$$
   This ensures sufficient regularity at ∂B.

3. *Synthesized Vector Field*: Define

$$X(x) := \eta(x)\mu_m(x)\nabla j(x)$$

This field vanishes at critical points and near ∂B.

Step 3: Flow Analysis

Boundary Avoidance: For $x \in B$, let $r(t) = \mu(\Phi(x,t))$. Compute:

$$\frac{dr}{dt} = \nabla\mu(\Phi) \cdot X(\Phi) = \eta(\Phi)\mu_m(\Phi)\nabla\mu(\Phi) \cdot \nabla j(\Phi)$$

Using $|\nabla\mu \cdot \nabla j| \leq C$ near ∂B:

$$\left|\frac{dr}{dt}\right| \leq C\eta(\Phi)\mu(\Phi)^{m+1} \leq Cr(t)^{m+1}$$

Solutions to $\dot{r} \leq Cr^{m+1}$ satisfy it will never reach 0 in finite time, establishing boundary avoidance.

Step 4: Monotonicity & Diffeomorphism

1. *Energy Gain*: Along trajectories:

$$\frac{d}{dt}j(\Phi(x,t)) = \nabla j(\Phi) \cdot X(\Phi) = \eta(\Phi)\mu_m(\Phi)\|\nabla j(\Phi)\|^2 \geq 0$$

Thus j is non-decreasing, with strict increase except at critical points.

2. *Flow Diffeomorphisms*: The differential $D\Phi(x,t)$ satisfies:

$$\frac{d}{dt}D\Phi(x,t) = DX(\Phi(x,t))D\Phi(x,t)$$

Since X is smooth with bounded derivatives on $\overline{B}$, Grönwall's inequality gives:

$$\|D\Phi(x,t)\| \leq \exp\left(\int_1^{1+\varepsilon} \|DX(\Phi(x,s))\|ds\right) < \infty$$

Thus $\Phi(\cdot,t)$ remains locally diffeomorphic, and properness follows from boundary avoidance.

Step 5: Isotopy Synthesis

The time-ε map $\Phi(0,\varepsilon)$ provides the required isotopy through diffeomorphisms. □

A.2 Proof of Cascading Improvement at Adjoint Timesteps

Proposition 4 (Cascading Improvement at Adjoint Timesteps)**.** *Consider two consecutive timesteps $t, t-\delta$. Following Algorithm 1, when comparing the cases with and without updating θ at t, updating θ results in an equal or lower cross-entropy for $\widehat{\mathbf{x}_0}$ at $t-\delta$ when δ is sufficiently small and all functions are smooth.*

Proof. We want to show

$$\mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^2), c\big) \;\leq\; \mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^1), c\big)$$

for sufficiently small δ with the following statement:

$$\widehat{\mathbf{x}_0}^1 \;=\; \mathbf{x}_{t-\delta} \;-\; \mathbf{v}_\theta\big(\mathbf{x}_{t-\delta}, t-\delta\big)\,(t-\delta)$$

and

$$\widehat{\mathbf{x}_0}^2 \;=\; \mathbf{x}_{t-\delta} \;-\; \mathbf{v}_{\theta+\Delta\theta}\big(\mathbf{x}_{t-\delta}, t-\delta\big)\,(t-\delta),$$

where

$$\Delta\theta \;=\; -\,l_r\,\nabla_\theta\Big(\mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^0), c\big)\Big), \quad \widehat{\mathbf{x}_0}^0 \;=\; \mathbf{x}_t \;-\; \mathbf{v}_\theta\big(\mathbf{x}_t, t\big)\,t.$$

Step 1. Relating $\widehat{\mathbf{x}_0}^1$ and $\widehat{\mathbf{x}_0}^0$. Since $\mathbf{x}_{t-\delta}$ is close to $\mathbf{x}_t$ for small δ, the smoothness of $\mathbf{v}_\theta(\cdot,\cdot)$ implies

$$\|\widehat{\mathbf{x}_0}^1 \;-\; \widehat{\mathbf{x}_0}^0\| \;=\; \Big\|\big[\mathbf{x}_{t-\delta} - \mathbf{v}_\theta(\mathbf{x}_{t-\delta}, t-\delta)\,(t-\delta)\big] \;-\; \big[\mathbf{x}_t - \mathbf{v}_\theta(\mathbf{x}_t, t)\,t\big]\Big\|$$

can be made arbitrarily small by taking δ sufficiently small (and using continuity/Lipschitz arguments). Consequently,

$$\nabla_\theta \mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^1), c\big) \quad \text{and} \quad \nabla_\theta \mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^0), c\big)$$

are also close for small δ.

Step 2. First-order comparison at $t-\delta$. By a first-order expansion of $\mathbf{v}_{\theta+\Delta\theta}$ around θ and the smoothness of $\mathbf{v}_\theta(\cdot,\cdot)$, we have

$$\mathbf{v}_{\theta+\Delta\theta}(\mathbf{x}_{t-\delta}, t-\delta) \;=\; \mathbf{v}_\theta(\mathbf{x}_{t-\delta}, t-\delta) \;+\; \nabla_\theta \mathbf{v}_\theta(\mathbf{x}_{t-\delta}, t-\delta)\,\Delta\theta \;+\; \mathcal{O}\big(\|\Delta\theta\|^2\big).$$

Hence,

$$\widehat{\mathbf{x}_0}^2 - \widehat{\mathbf{x}_0}^1 \;=\; -\Big[\mathbf{v}_{\theta+\Delta\theta}(\mathbf{x}_{t-\delta}, t-\delta) - \mathbf{v}_\theta(\mathbf{x}_{t-\delta}, t-\delta)\Big]\,(t-\delta) \;\approx\; -(t-\delta)\,\nabla_\theta \mathbf{v}_\theta\big(\mathbf{x}_{t-\delta}, t-\delta\big)\,\Delta\theta.$$

Step 3. Cross-entropy decrease. Using the smoothness of the cross-entropy and another first-order expansion,

$$\begin{aligned}
&\mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^2), c\big) \;-\; \mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^1), c\big) \\
&\quad\approx\; \big\langle \nabla_{\widehat{\mathbf{x}_0}} \mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^1), c\big),\; \widehat{\mathbf{x}_0}^2 \;-\; \widehat{\mathbf{x}_0}^1 \big\rangle \;+\; \mathcal{O}\big(\|\widehat{\mathbf{x}_0}^2 - \widehat{\mathbf{x}_0}^1\|^2\big) \\
&\quad\approx\; \big\langle \nabla_\theta \mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^1), c\big),\; \Delta\theta \big\rangle \;+\; \mathcal{O}\big(\|\Delta\theta\|^2, \|\delta\|\big).
\end{aligned}$$

By definition of the gradient step $\Delta\theta = -l_r\,\nabla_\theta \mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^0), c\big)$ and the fact that $\nabla_\theta \mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^0), c\big)$ is close to $\nabla_\theta \mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^1), c\big)$ for small δ, the above inner product is non-positive up to higher-order (small) terms. Concretely,

$$\big\langle \nabla_\theta \mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^1), c\big),\; -l_r\,\nabla_\theta \mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^0), c\big) \big\rangle \;\le\; 0$$

when δ is sufficiently small so that these gradients align (up to small errors). Therefore,

$$\mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^2), c\big) \;\le\; \mathbb{CE}\big(f(\widehat{\mathbf{x}_0}^1), c\big),$$

which completes the proof. □

B Related Works

B.1 Targeted and Untargeted Attacks

Targeted Attacks. The objective of targeted attacks is to force the classifier to output a specified label. In other words, the attacker seeks to cause the model to produce incorrect classification results and aims for the result to be a specific target class. This type of attack is more hazardous due to its ability to manipulate the model's output precisely but is typically more challenging to execute.

Untargeted Attacks. The goal of untargeted attacks is to make the classifier output any incorrect label. The attacker merely needs to mislead the model so that its classification result does not match the true label. Despite having lower requirements, untargeted attacks can still have severe consequences in certain situations.

B.2 White-Box and Black-Box Attacks

White-Box Attacks. White-box attacks assume that the attacker has complete access to the target model, including its architecture, parameters, and gradient information. Using this information, the attacker can generate efficient adversarial examples through iterative optimization methods.

Black-Box Attacks. Black-box attacks assume that the attacker does not have access to the internal information of the target model. A common method to implement black-box attacks is to utilize transferability, where adversarial examples are first generated against a known source model and then used to attack the unknown target model.

B.3 Instance-Specific and Instance-Agnostic Attacks

Instance-Specific Attacks. Instance-specific attacks [12, 20, 16, 67, 59, 38, 34] and *instance-agnostic* generate adversarial perturbations for specific input samples. The attacker uses gradient information from the target model and iterative optimization algorithms to create minimal perturbations that achieve the attack on a given sample. Such attacks usually have high success rates on individual samples but lack generalization and transferability.

Instance-Agnostic Attacks. Instance-agnostic attacks [65, 39, 33, 42, 43, 18] do not target specific input samples but instead learn universal adversarial perturbations or generative functions based on data distribution. These attack methods have better generalization across different samples, thus exhibiting stronger transferability.

B.4 Subcategories of Instance-Agnostic Attacks

Instance-agnostic attacks can be further subdivided into the following categories:

Universal Adversarial Perturbations. These methods learn a universal perturbation [41, 70] applicable to the entire dataset. The classifier can be misled by superimposing this perturbation on any input sample.

Generative Models. Generative attacks [46, 42] train a generator that, upon receiving an input sample, can produce specific adversarial perturbations. This approach often surpasses universal adversarial perturbations regarding flexibility and attack efficacy.

B.5 Single-Target and Multi-Target Attacks

Single-Target Attacks. Single-target attacks train an individual generative model for each target class [42, 43, 18, 61]. Although these models achieve high success rates for single-target classes, the training cost becomes substantial when the number of target classes is large, thereby limiting practical usability.

Multi-Target Attacks. Multi-target attacks simultaneously train the attack capabilities for multiple target classes within a single model [24, 69, 17]. Class labels or text embeddings are typically used as conditional inputs to generate corresponding adversarial perturbations. This method significantly reduces training costs and enhances feasibility in real-world applications.

C Comparison with Other Diffusion-Based Methods

Currently, diffusion models have been explored in several adversarial attack methods [5, 68, 6, 4, 8]. Among them, ACA [6] and DiffAttack [4] leverage DDIM inversion to obtain the latent space of diffusion models and optimize adversarial examples within this latent space. AdvDiffuser [5], DiffPGD [68], and AdvDiff [8] incorporate adversarial guidance during the reverse denoising process of diffusion models. Notably, although these methods are diffusion-based, they are all instance-specific attacks, requiring access to the target classifier's gradient information during inference for each input sample to perform the attack.

In contrast, our method, once trained, does not require any further information from the classifier, enabling more efficient generation of adversarial examples.

Furthermore, since these prior methods either only support untargeted attacks[5, 68, 6, 4] or focus on unrestricted adversarial examples[5, 6, 4, 8], their settings are not directly compatible with ours, making direct comparisons infeasible.

D Method Details

D.1 Target Class Condition Representation

For each label c in the target label set $\mathcal{C}$, we first obtain its class description and format it into a text condition using the template "a photo of a {class}"[47]. Subsequently, we utilize CLIP's text encoder to derive this textual input's embedding $\mathbf{e}$. Finally, this embedding is fed into our model via cross-attention mechanisms:

$$\begin{aligned} Q = \mathbf{z}W_Q, K = \mathbf{e}W_K, V = \mathbf{e}W_V, \\ Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d}}) \cdot V, \end{aligned} \tag{6}$$

where $\mathbf{z} \in \mathbb{R}^{d_z}$ denotes the flattened intermediate features of the unet model, $W_Q \in \mathbb{R}^{d_z \times d}$, $W_K \in \mathbb{R}^{d_e \times d}$, $W_V \in \mathbb{R}^{d_e \times d}$ are learnable parameters.

By employing this approach, we can leverage the rich semantic priors associated with the target classes embedded in the pre-trained diffusion model, thereby facilitating a more effective training process.

Algorithm 3 Single-Target Fine-Tuning Mechanism

Input: $\tau = N\delta$, stepsize δ, model param. ϕ, θ, model f, target label c, training dataset $\{\mathbf{I}^i\}_{i \in \mathcal{I}}$, learning rate l_r
repeat
  for $i \in \mathcal{I}$ **do**
    get $\mathbf{x}_0 = \mathbf{I}^i$
    for $t = 1$ **to** N **do**
      $\mathbf{x}_{t\delta} = \mathbf{x}_{(t-1)\delta} + \mathbf{v}_\phi(\mathbf{x}_{(t-1)\delta}, (t-1)\delta, \varnothing)\delta$
    end for
    for $t = N$ **to** 1 **do**
      $\mathbf{x}_{(t-1)\delta} = \mathbf{x}_{t\delta} - \mathbf{v}_\theta(\mathbf{x}_{t\delta}, t\delta, c)\delta$
      $\widehat{\mathbf{x}_0} = \mathbf{x}_{t\delta} - \mathbf{v}_\theta(\mathbf{x}_{t\delta}, t\delta, c)t\delta$
      get random mask M
      $\widehat{\mathbf{x}_0} = \mathbf{x}_0 + M \cdot (\widehat{\mathbf{x}_0} - \mathbf{x}_0)$
      $\widehat{\mathbf{x}_0} = \text{clip}\,(\widehat{\mathbf{x}_0}, \mathbf{x} - \epsilon, \mathbf{x} + \epsilon)$
      $\theta = \theta - l_r \cdot \nabla_\theta(\mathbb{CE}(f(\widehat{\mathbf{x}_0}), c))$
    end for
  end for
until $\mathbf{v}_\theta$ convergence
Return: Dual-Flow $\{\mathbf{v}_\phi, \mathbf{v}_\theta\}$

D.2 Fine-Tuning on Single-Target Tasks

We fine-tune our model for single-target tasks to enhance its performance further. Specifically, we fix the target label during training, enabling the model to focus on targeted attacks for a specific label. To mitigate the perturbations being confined to some areas of the image, which can reduce the robustness and transferability of adversarial examples in single-target training, we apply the mechanism introduced in [17].

In detail, we generate a random mask M of the same size as the image, where several randomly positioned square pixel areas are set to 0, and the rest are set to 1. By multiplying this mask with the perturbation, we ensure the generated adversarial samples remain consistent with the original image in the masked square areas. This forces the model to create adversarial patterns distributed across the entire image rather than being localized to specific regions, as illustrated in Algorithm 3.

Like other single-target methods, we must fine-tune a separate model for each target. However, due to our model's powerful capabilities in multi-target attacks, once the model is trained on the

multi-target task, it requires only a few additional steps to adapt to each single-target task. This results in significantly lower training overhead compared to other methods.

E Computational Cost

For training the multi-target Dual-Flow (e.g., using Res-152 as the source model), we require approximately 24 hours of training on 2 NVIDIA RTX 3090 GPUs, which is equivalent to 48 GPU hours. To further fine-tune on single-target tasks, we need an additional 4 hours per class on a single NVIDIA RTX 3090 GPU, totaling 4 hours $\times$ 8 classes = 32 GPU hours. Once training is complete, our method only requires 15 minutes of inference on a single GPU to generate adversarial samples for 8 target classes across the entire ImageNet NeurIPS validation set.

Table 5: Training and inference time for Dual-Flow with Res-152 as the source model.

Train (only multi-target)	Train (multi-target + single-target)	Inference
48 GPU hours	80 GPU hours	15 min

We highlight that once our method completes training, it can rapidly generate adversarial samples with high transferability and robustness without requiring any gradient information from the classifier models.

F More Experiments

Evaluation on Transformer Models. We tested our attack method's success rate when transferred to transformer models. Specifically, we utilized Res152 as the source model. The results, included in Table 6, demonstrate that despite the fundamental architectural differences between transformer models and our source model based on convolutional networks (Res152), our method maintains a high attack success rate, significantly outperforming baseline methods. This further corroborates the advantage of our method in terms of transferability.

Table 6: Attack success rates (%) for multi-target attacks on transformer models. The source model is Res-152.

Method	ViT-B/16 [64]	CaiT-S/24 [58]	Visformer-S [7]	DeiT-B [57]	LeViT-256 [23]	TNT-S [25]
C-GSP	11.78	32.00	36.60	35.58	37.85	31.00
CGNC	19.46	54.56	58.70	59.90	57.53	48.40
Dual-Flow	**36.39**	**74.24**	**76.72**	**78.50**	**79.34**	**67.86**

Evaluation on DiffPure. We evaluate our attack method using Diffusion Models for Adversarial Purification (DiffPure)[45]. The experimental results show the attack success rates of our method under various DiffPure t^* settings and compare them with the baseline method. As illustrated in Table 7, the baseline method is easily nullified by the purification process, whereas our method maintains a significant success rate. This further demonstrates the robustness of our approach.

Table 7: Attack success rates (%) for multi-target attacks on normally trained models with DiffPure. The source model is Res-152.

t^*	Method	Inc-v3	Inc-v4	Inc-Res-v2	Res-152	DN-121	GoogleNet	VGG-16
0.05	CGNC	16.26	19.91	8.53	67.76	49.81	21.79	29.81
	Dual-Flow	49.60	51.92	37.56	79.50	70.20	51.30	52.26
0.10	CGNC	2.41	2.96	1.25	14.65	10.10	3.46	4.59
	Dual-Flow	24.31	25.32	18.76	48.05	39.81	25.06	24.78
0.15	CGNC	0.47	0.46	0.34	1.84	1.46	0.62	0.92
	Dual-Flow	7.20	7.87	5.89	16.70	13.72	7.71	8.34

Table 8: Attack success rates (%) for multi-target attacks on normally trained models using the ImageNet validation set. The perturbation budget is constrained to $l_\infty \leq 16/255$. * indicates white-box attacks. The results are averaged across 8 different target classes, and the overall average on the far right is computed solely for black-box attacks.

Source	Method	Inc-v3	Inc-v4	Inc-Res-v2	Res-152	DN-121	GoogleNet	VGG-16	Average
Inc-v3	CGNC	96.59*	57.82	46.84	44.13	65.90	53.40	56.27	54.06
	Dual-Flow	89.89*	**75.74**	**65.05**	**75.73**	**82.75**	**72.21**	**66.20**	**72.95**
Res-152	CGNC	56.00	50.37	32.26	96.44*	86.69	63.84	63.90	58.84
	Dual-Flow	**69.75**	**72.53**	**54.11**	92.70*	**86.71**	**74.08**	**68.22**	**70.90**

Transferability Evaluation On ImageNet Validation Set. In addition to the evaluation on the ImageNet-NeurIPS (1k) dataset[44], we conducted an assessment of our attack method on the ImageNet validation set (50k)[10] and compared it with the state-of-the-art multi-target attack method, CGNC[17]. The experimental results presented in Table 8 indicate that our method achieves a significantly higher average black-box attack success rate than CGNC, demonstrating its superior transferability. This outcome is consistent with the results observed on the ImageNet-NeurIPS (1k) dataset.

Evaluation on Smaller Perturbation Budgets. To validate the effectiveness of our method when adversarial perturbations are more imperceptible, we conducted experiments at lower perturbation budgets for black-box targeted attacks. We used the $\epsilon = 16/255$ versions of trained models (CGNC and Dual-Flow) and clipped the generated samples to meet the $\epsilon = 12/255$ and $\epsilon = 8/255$ limits. The results are shown in Tab 9. The results show that our method achieves better attack performance than the baseline across various perturbation budgets.

Table 9: Attack success rates (%) for multi-target attacks on normally trained models with different perturbation budgets. The source model is Inc-v3.

ϵ	Method	Inc-v4	Inc-Res-v2	Res-152	DN-121	GoogleNet	VGG-16
16/255	CGNC	59.43	48.06	42.48	62.98	51.33	52.54
	Dual-Flow	**77.19**	**66.76**	**77.06**	**82.64**	**73.01**	**67.09**
12/255	CGNC	43.23	30.34	28.48	46.96	33.65	37.95
	Dual-Flow	**55.80**	**41.82**	**54.41**	**63.19**	**47.66**	**44.94**
8/255	CGNC	13.33	6.54	6.95	14.65	7.51	10.29
	Dual-Flow	**23.86**	**14.04**	**21.21**	**27.68**	**13.39**	**15.55**

Evaluation on Single-Target Attacks against Robustly Trained Models. We validate the effectiveness of our attack method in single-target settings against the robust models mentioned in the main text and compare it with other single-target attacks. As shown in Table 10, the conclusions are similar to those in the multi-target case in the main text, demonstrating that our method can effectively mislead robustly trained classifiers.

Table 10: Attack success rates (%) for single-target attacks against robust models on ImageNet NeurIPS validation set. The perturbation budget $l_\infty \leq 16/255$. The results are averaged on 8 different target classes. The source model is Res-152.

Method	Inc-v3$_{adv}$	IR-v2$_{ens}$	Res50$_{SIN}$	Res50$_{IN}$	Res50$_{fine}$	Res50$_{Aug}$
GAP	5.72	4.51	7.33	71.04	83.64	52.07
CD-AP	3.77	6.48	7.09	63.72	76.79	49.67
TTP	27.99	26.08	24.61	72.47	74.51	70.96
DGTA-PI	31.10	30.07	27.70	77.13	80.55	76.78
CGNC†	31.55	33.63	33.31	**88.34**	**89.74**	72.96
Dual-Flow†	**47.32**	**56.22**	**60.66**	84.15	85.18	**78.56**

G Victim Model Details

All the victim models we used employ the official weights provided for the 1K-class ImageNet dataset [10]. For normally trained models (including transformer models), we directly call these models and their weights through the torchvision [40] or timm [63] libraries. For robust models (adversarially trained models):

- Inc-v3$_{adv}$ and IR-v2$_{ens}$, we call their official weights through the timm library.
- Res50$_{SIN}$, Res50$_{IN}$, and Res50$_{fine}$, we use the weights provided by the open-source repository Texture-vs-Shape [21].
- Res50$_{Aug}$, we use the weights provided by the open-source repository AugMix [28].

We report the accuracy of these models on the ImageNet NeurIPS validation set in Tab 11.

Table 11: Accuracy (%) of victim models on ImageNet NeurIPS validation set.

Type	Model & Accuracy						
Normal	Inc-v3 95.1	Inc-v4 94.7	Inc-Res-v2 97.3	Res-152 94.5	DN-121 90.7	GoogleNet 87.0	VGG-16 87.2
Robust	Inc-v3$_{adv}$ 86.7	IR-v2$_{ens}$ 94.5	Res50$_{SIN}$ 68.4	Res50$_{IN}$ 89.5	Res50$_{fine}$ 93.2	Res50$_{Aug}$ 94.1	
Transformer	ViT-B/16 84.4	CaiT-S/24 98.1	Visformer-S 96.5	DeiT-B 96.9	LeViT-256 94.7	TNT-S 91.3	

H More Ablation Studies

Ablation on LoRA Fine-tuning. To demonstrate that the adversarial attack capability of our model stems from LoRA fine-tuning, we evaluated the model's performance when performing attacks directly without applying LoRA. As shown in Table 12, the original model fails to achieve effective adversarial attacks, which validates the necessity of incorporating LoRA for fine-tuning.

Table 12: Attack success rates comparing the model with and without LoRA. The source model is Res-152.

Method	Inc-v3	Inc-v4	Inc-Res-v2	Res-152	DN-121	GoogleNet	VGG-16
w/o LoRA	0.075	0.075	0.05	0.05	0.075	0.05	0.075
w/ LoRA	69.58	71.92	56.10	92.39	85.73	73.65	67.59

Ablation on Dynamic Gradient Clipping. To evaluate the effect of dynamic gradient clipping, we designed a variant that does not apply dynamic clipping during training and only clips the outputs to satisfy the norm constraints during inference. As shown in Table XXX, if the model is not trained with clipping, it fails to adapt to the norm constraints at inference time and thus cannot perform effective attacks.

Table 13: Attack success rates comparing the model with and without Dynamic Gradient Clipping. The source model is Res-152.

Method	Inc-v3	Inc-v4	Inc-Res-v2	Res-152	DN-121	GoogleNet	VGG-16
w/o clip	1.85	2.32	1.96	2.84	3.51	1.82	1.19
w/ clip	69.58	71.92	56.10	92.39	85.73	73.65	67.59

Ablation on Loss. We designed a variant: during training, we use a new L_2 loss function to make the model's output as close as possible to the original ODE trajectory, ensuring it does not deviate too far from the original ODE flow. We call this variant Dual-Flow-L_2. The experimental results in Table 14 indicate that the attack capability of this variant is not ideal, as described in Section 3.2.

Table 14: Comparison of Dual-Flow and Dual-Flow-L_2. The source model is Res-152.

Method	Inc-v3	Inc-v4	Inc-Res-v2	Res-152	DN-121	GoogleNet	VGG-16	Average
Dual-Flow	69.58	71.92	56.10	92.39	85.73	73.65	67.59	70.76
Dual-Flow-L_2	54.90	56.26	42.94	86.80	78.41	57.71	46.36	56.10

I More Analysis

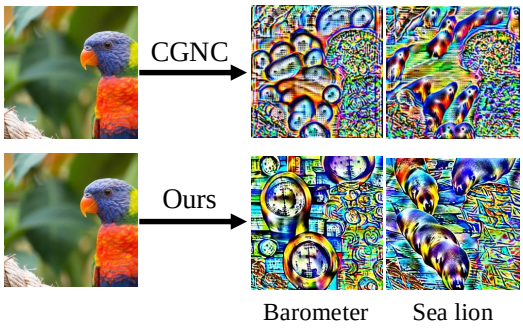


Figure 5: Visualization results comparing the adversarial perturbations generated by our method with those produced by CGNC.

Table 15: Comparison of different types of adversarial inputs. CGNC-P and Dual-Flow-P represent the adversarial perturbations generated by CGNC and our method, respectively, while Dual-Flow-A denotes the unclipped adversarial samples produced by our method. The adversarial perturbations are scaled to a range between 0 and 1 before input into the classifier.

Source	Method	Inc-v3	Inc-v4	Inc-Res-v2	Res-152	DN-121	GoogleNet	VGG-16
Inc-v3	CGNC-P	56.80*	19.15	22.06	10.56	13.14	14.56	3.41
	Dual-Flow-P	86.55*	81.20	74.55	74.55	70.66	76.15	55.10
	Dual-Flow-A	99.12*	95.31	92.79	97.39	96.80	95.62	87.95
Res-152	CGNC-P	23.61	25.72	39.07	58.29*	39.09	37.47	17.21
	Dual-Flow-P	68.44	81.48	78.26	86.29*	80.44	74.59	55.35
	Dual-Flow-A	94.38	96.60	94.62	99.19*	97.92	93.54	90.85

Semantic Adversarial Attack. We compared the adversarial perturbations generated by our method and those produced by the state-of-the-art multi-target attack method, CGNC[17]. As illustrated in Figure 5, the visual results indicate that while CGNC's perturbations contain some semantic features of the target class, they are primarily confined to small, repetitive patterns. In contrast, our method generates perturbations that are semantically more representative of the complete target class.

To further validate this observation, we directly input the adversarial perturbations generated by CGNC and our method into the target classifier. As shown in Table 15, our adversarial perturbations alone can induce the classifier to predict the target class with a relatively high probability. Conversely, the perturbations produced by CGNC exhibit a lower likelihood, particularly when transferred to black-box models. This demonstrates that our perturbations incorporate more semantic features of the target class.

Moreover, as depicted in Figure 2, the unclipped samples generated by our method are semantically very close to the target class. We also input these unclipped samples directly into the target classifier. Table 15 shows that these samples are highly likely to be classified as the target class. This confirms that semantic proximity to the target class effectively increases attack success rates.

These findings collectively suggest that our attack method's robustness and transferability result from embedding substantial target class semantics into the images, thereby reducing dependence on specific target model decision boundaries.

J More Visualization

Figure 6: Visualization results of different input images targeting various classes. For each text prompt of the target class, the left column displays the adversarial examples generated before clipping, the middle column shows the adversarial examples after clipping, and the right column presents the corresponding adversarial perturbations, which represent the differences between the clipped adversarial examples and the original images. Note that the perturbations are scaled to a range between 0 and 1. The source model used is Inc-v3.

K Limitations

While our proposed Dual-Flow framework demonstrates strong adversarial attack performance and transferability, several limitations remain. First, the training process involves additional computational overhead compared to some simpler attack methods, which may affect scalability to extremely large datasets or models. Second, although we observe improved semantic consistency in generated perturbations, there is still room to enhance the interpretability and controllability of adversarial patterns in more complex scenarios. Lastly, our evaluations focus primarily on standard image classification benchmarks; extending the approach to other tasks or modalities warrants further investigation.

L Statistical Significance

To rigorously evaluate the reliability of our attack success rates, we partitioned the test dataset into 5 disjoint subsets, each containing 200 images. We computed the attack success rate for each subset independently and derived the overall 95% confidence intervals (CIs) for our method. These confidence intervals provide a quantitative measure of variability and statistical uncertainty. We further compared our results against the baseline method by examining the overlap of their respective confidence intervals. As shown in Table 16, the non-overlapping intervals observed in our experiments indicate that the improvement in attack success rate achieved by our method is statistically significant with high confidence.

Table 16: Attack success rates with 95% confidence intervals over 5 data splits, compared to the baseline method. The source model is Res-152.

Method	Inc-v3	Inc-v4	Inc-Res-v2	DN-121	GoogleNet	VGG-16
CGNC	53.39±2.77	51.53±2.52	34.24±1.72	85.66±1.53	62.23±2.20	63.36±2.95
Dual-Flow	69.58±3.70	71.92±3.27	56.10±3.31	85.73±1.57	73.65±2.83	67.59±2.28

M Societal Impacts

Our work contributes to a deeper understanding of adversarial vulnerabilities in deep learning models, which is essential for improving model robustness and security. By developing more effective attack methods, we provide valuable tools for evaluating and strengthening defenses against malicious exploitation. However, as with any adversarial technique, there is a potential risk of misuse in compromising AI systems. We advocate for responsible use of such methods strictly within research and security auditing contexts, and encourage the community to develop corresponding mitigation strategies to safeguard AI applications.