Boosting Adversarial Transferability through Augmentation in Hypothesis Space

Yu Guo[1,2] Weiquan Liu[1,3*] Qingshan Xu[4] Shijun Zheng[1,2] Shujun Huang[1,2]
Yu Zang[1,2] Siqi Shen[1,2] Chenglu Wen[1,2] Cheng Wang[1,2*]
[1]Fujian Key Laboratory of Sensing and Computing for Smart Cities, Xiamen University, China
[2]Key Laboratory of Multimedia Trusted Perception and Efficient Computing,
Ministry of Education of China, School of Informatics, Xiamen University, China
[3] College of Computer Engineering, Jimei University, China [4]Nanyang Technological University, Singapore

Abstract

Adversarial examples can mislead deep neural networks with subtle perturbations, causing them to make incorrect predictions. Notably, adversarial examples crafted for one model can also deceive other models, a phenomenon known as the transferability of adversarial examples. To improve transferability, existing studies have designed increasingly complex mechanisms, but the improvements achieved remain relatively limited and are often difficult to adapt to other modalities, further restricting the scalability of these methods. In this work, we observe a mirroring relationship between model generalization and adversarial example transferability. Motivated by this observation, we propose an augmentation-based attack, called ***OPS*** *(**O**perator-**P**erturbation-based **S**tochastic optimization), which constructs a stochastic optimization problem by input transformation operators and random perturbations, and solves this problem to generate adversarial examples with better transferability. Extensive experiments on both images and 3D point clouds demonstrate that OPS significantly outperforms existing state-of-the-art methods in terms of both performance and cost, showcasing the universality and superiority of our approach. The code is available at* https://github.com/the-full/OPS.

1. Introduction

Deep Neural Networks (DNNs) have been integrated into many real-world solutions, such as facial recognition [41, 42, 45], autonomous driving [9, 18, 21]. However, the existence of adversarial examples reveals the vulnerability of these networks. These adversarial examples, generated by adding small perturbations to the original data, can mislead the model into making incorrect predictions. Therefore, to ensure the reliability of DNNs in critical applications, researchers are increasingly focusing on adversarial attack and defense strategies [4, 7, 10, 43, 59] related to DNNs.

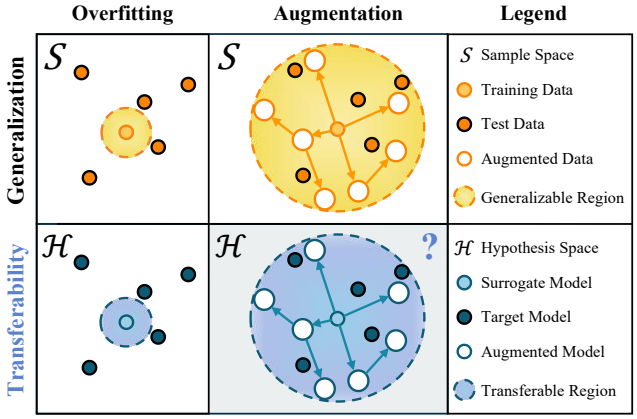

Figure 1. The mirrored relationship between model generalization and adversarial example transferability. The former refers to the model's overfitting to training data, with the research goal of expanding the model's generalization region, while the latter refers to an example's overfitting to the surrogate model, with the goal of expanding the example's transferability region.

Based on the information available to the attacker, adversarial attacks can be divided into two categories: (1) White-box attacks, where the parameters of the target model are fully accessible to the attacker. (2) Black-box attacks, where the attacker has only limited access to the target model. Research by Goodfellow et al. [10] found that adversarial examples crafted for a specific model can also mislead other models, indicating the transferability of adversarial examples. This means that an attacker can create adversarial examples in a white-box setting using a surrogate model and then leverage transferability to conduct a black-box attack. However, the adversarial examples generated in this way often overfit to the surrogate model, resulting in good performance in white-box attacks but relatively poor transferability [17, 59]. To enhance the trans-

*Corresponding author

ferability, various strategies have been developed, including stabilizing the gradient direction [5, 47], seeking flat loss solutions [8, 38], increasing input diversity [1, 23, 60], modifying forward or backward processes [54, 66, 70], and adding regularization terms [12, 25], among others. However, these methods often introduce complex computations while providing only relatively limited performance gains.

In this paper, we introduce two fundamental concepts in machine learning—sample space and hypothesis space—as the universal sets of data and model, while sweeping the interaction details between data and model under the rug. We observe that there is a mirrored relationship between the study of model generalization and the study of transferability. As shown in Fig. 1, the former aims to expand the generalizable region of the model within the sample space, while the latter seeks to enlarge the transferable region of adversarial examples within the hypothesis space. Considering the role of data augmentation in enhancing generalization, we wonder: *Can we perform '**model augmentation**' on the surrogate model to improve transferability?*

We explored this possibility. Specifically, we establish a topological structure in the hypothesis space, showing that the target model lies within a neighborhood of the surrogate model. Thus, an ideal augmentation approach would be to obtain more models by sampling the neighborhood of the surrogate model. However, since the hypothesis space is a function space, directly sampling the neighborhood of the surrogate model is challenging. To address this issue, we propose two simple alternatives: (1) composing the surrogate model with an input transformation operator or (2) combining it with a fixed perturbation. We then use the obtained neighbors to formulate a stochastic optimization problem and generate adversarial examples by solving this problem. We refer to this method as **O**perator-**P**erturbation-based **S**tochastic optimization (**OPS**).

By replacing the input transformation operator, OPS can be easily adapted to different modalities. In this study, we focus on the visual domain and validate the effectiveness of OPS on two visual modalities: images and 3D point clouds. Additionally, to the best of our knowledge, there is currently no publicly available benchmark for 3D point clouds. Therefore, we constructed **3D** **T**ransfer **A**ttack **B**enchmark (**3DTAB**) to comprehensively evaluate our method. Our contributions are summarized as follows:

- We revisit the transferability of adversarial examples from a higher-level perspective and propose a new approach to improve transferability through model augmentation.
- We propose an augmentation-based attack, OPS, which generates adversarial examples with better transferability by probing the neighbors of the surrogate model.
- We establish a benchmark for 3D point cloud adversarial transferability, 3DTAB, providing a comprehensive evaluation framework for subsequent research.
- Extensive experiments on both images and point clouds demonstrate that OPS significantly outperforms baseline methods in both transferability and computational overhead, showcasing its superiority and scalability.

2. Related Work

2.1. Transfer-based Attack in Image

The transferability of adversarial examples was first identified in the pioneering work on adversarial attacks, FGSM [10]. Its iterative version, I-FGSM [17], achieves better white-box attack performance but suffers from lower transferability. MI [5] suggested that this issue arises from the algorithm getting stuck in local maxima, introducing momentum techniques to solve it. This led to several momentum-based attacks like NI [22], VMI [47], and GRA [37]. Meanwhile, RAP [38] enhanced transferability by searching adversarial examples with a flat loss landscape, but this requires more computational overhead. Later methods like PGN [8] and NCS [39] have partially addressed this issue and achieved higher transferability.

On the other hand, enhancing input diversity through transformations has become one of the most effective approaches to improve transferability. DIM [60] first proposed and applied this idea by adjusting images to random sizes and then padding them to a fixed size. Subsequent research has largely focused on finding input transformations that can enhance transferability, such as SIM [23], Admix [48], and DeCoWA [24]. More recently, SIA [50] and BSR [46] decompose images into blocks and apply different transformations at the block level, further enhancing diversity. Additionally, some works have attempted to predict the most suitable transformations for different samples. For instance, AITL [62] uses a network to predict the most effective transformation combinations, and L2T [69] learns the optimal transformation trajectory for each sample.

Our method, OPS, utilizes input transformations but neither aims to enhance input diversity nor customizes transformations for individual samples. Instead, it emphasizes gradient diversity by randomly selecting transformations in each iteration. This makes the optimization more computationally efficient (see the discussion in **Appendix**).

2.2. Adversarial Defense in Image

Researchers have been working to eliminate the threat posed by adversarial attacks and have developed various defense methods. Adversarial training [10, 31, 43, 53] incorporates adversarial examples into the training set to enhance the robustness of models. For example, Tramèr et al. [43] used adversarial examples generated on different models for ensemble adversarial training, improving the model's robustness against transfer-based attacks. Input purification methods [20, 34, 35] aim to restore adversarial examples

to natural samples. For instance, Liao et al. [20] trained a High-level Representation Guided Denoiser (HGD) to remove adversarial noise from samples. Unlike the above empirical defenses, certified defenses [4, 11, 63] provide provable guarantees within a certain radius. For example, Cohen et al. [4] trained an ImageNet classifier with a provable defense radius based on randomized smoothing (RS).

2.3. Adversarial Attack Research in 3D Point Cloud

3D-Adv [59] is the first study on 3D point cloud adversarial attacks, based on the C&W attack [2]. Many subsequent studies have focused on generating adversarial examples with more natural shapes and making them harder for humans to detect by adding regularization terms. For example, KNN [44] incorporates knn distance into the loss function, GeoA^3 [52] proposes a geometry-aware loss term, and HiT-ADV [30] introduces an imperceptibility score to control perturbations in regions less sensitive to human vision. However, AdvPC [12] points out that these methods produce adversarial examples that tend to overfit specific models, leading to poor transferability. It then introduced adversarial loss for reconstructing point clouds to alleviate overfitting. Most of the following research has followed this approach, with AOF [25] introducing adversarial loss for low-frequency components and PF-Attack [13] introducing adversarial loss for sub-perturbations.

Most 2D defense methods cannot be directly applied to 3D tasks, but the defense strategies are universal. Liu et al. [26] extended adversarial training [10] to the 3D domain. Zhou et al. [68] observed that adversarial point clouds always contain outliers and proposed two simple input purification methods, SRS and SOR. Furthermore, they introduced DUP-Net, which achieved better defense results by adding an upsampling network after SOR. In contrast, IF-Defense [58] focuses on the implicit function representation of 3D objects, using an occupancy network [32] to recover clean point clouds from the attacked point clouds. Certified defenses have also been explored in point cloud research. Liu et al. [27] introduced the first certified defense for point clouds based on random sampling.

3. Methodology

In this section, we primarily focus on image classification tasks to illustrate the design concept of the algorithm. This does not imply that our method is limited to the image modality. In fact, our approach can be readily extended to 3D point clouds (described in Sec. 3.6).

3.1. Preliminaries

We refer to the model used by the attacker to craft adversarial examples as the surrogate model, and the model being attacked as the target model. The classical definition of adversarial examples includes two parts:

- **Adversarial:** adversarial example $\mathbf{x}^{adv}$ can mislead target model.
- **Imperceptible:** it is difficult for humans to determine whether an image is an adversarial example.

In white-box attacks, the surrogate model and the target model are the same model. Attackers typically model it as a box-constrained problem as shown below:

$$\begin{aligned} \arg\max_{\mathbf{x}^{adv}} \quad & \mathcal{J}(f_\theta(\mathbf{x}^{adv}), y) \\ \text{s.t.} \quad & \|\mathbf{x}^{adv} - \mathbf{x}^{ori}\|_\infty < \epsilon, \end{aligned} \tag{1}$$

where $f_\theta(\cdot)$ is the surrogate model with learnable parameters θ, $\mathcal{J}(x, y)$ represents the cross-entropy loss function, $\mathbf{x}^{ori}$ is the clean example with groud-truth label y. The attacker aims to search an adversarial example $\mathbf{x}^{adv}$ that misleads f_θ within the ℓ_∞ neighborhood of $\mathbf{x}^{ori}$. Therefore, the radius ϵ is also referred to as the attack budget. In practice, the perturbation $\mathbf{\Delta} = \mathbf{x}^{adv} - \mathbf{x}^{ori}$ is typically treated as the optimization variable, and the problem is solved by iteratively updating the perturbation. For instance, the perturbation update formula in I-FGSM [17] is:

$$\mathbf{\Delta}_{i+1} = \mathbf{\Delta}_i + \text{Clip}_\epsilon \left\{ \alpha \cdot \text{Sign}\left(\nabla_{\mathbf{x}} \mathcal{J}\left(f_\theta(\mathbf{x}^{ori} + \mathbf{\Delta}_i), y \right) \right) \right\}. \tag{2}$$

Here, $\text{Clip}_\epsilon\{\cdot\}$ denotes the operation of clipping the portion of the perturbation that exceeds the budget ϵ back to ϵ, and α is the iteration step size. For an iterative process of T steps, it is usually set as $\alpha = \epsilon/T$.

In the transfer attack paradigm, the surrogate model and the target model are different models, with the latter is typically not just one. Let the set of target models be denoted as $V = \{f_1, f_2, \ldots\}$. We can restate the objective function of Eq. (1) in the following form:

$$\arg\max_{\mathbf{x}^{adv}} \quad \mathbb{E}_{f_i \in V} \left[\mathcal{J}(f_i(\mathbf{x}^{adv}), y) \right]. \tag{3}$$

The problem is that the attacker cannot access the information of the target model set V, making Eq. (3) intractable. However, we notice that this problem is similar to few-shot learning in model generalization research. Given that data augmentation has proven to be an effective solution to this problem, could we adopt a similar approach in the study of transferability and conduct 'model augmentation'? To do so, we need to analyze the distribution of the target models and the surrogate model.

3.2. Hypothesis Space

A neural network designed for M-class image classification can be represented as a function $h\colon \mathbf{x} \to z$, where $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ denotes the input image, and its distribution is given by $p(\mathbf{x})$. The output $z \in \mathbb{R}^M$ represents the predicted scores from h. We define the hypothesis space $\mathcal{H}$ as the set of all functions that conform to this form.

We define a functional $\mathcal{F}(h) = \int \mathcal{J}(h(\mathbf{x}), y) \cdot p(\mathbf{x})\, \mathrm{d}\mathbf{x}$ to quantify the performance of each hypothesis $h \in \mathcal{H}$ under the given data distribution. On this basis, we introduce a pseudo-metric $d(h_1, h_2) = |\mathcal{F}(h_1) - \mathcal{F}(h_2)|$ to measure the difference between two hypotheses. Furthermore, for any hypothesis h, we define the neighborhood centered at h with radius r as the open ball $\mathcal{B}_r(h) = \{h' \mid d(h, h') < r\}$. It can be proven that such a neighborhood system induces a topological structure (see **Appendix**). Our key insight is that both the surrogate model f_θ and target models $f_i \in V$ perform well on classification tasks. Therefore, they are included in a set $H = \{h \mid \mathcal{F}(h) < a\}$, where a is a small value. Thus, we have $V \subset \mathcal{B}_a(f_\theta)$. Therefore, an ideal augmentation approach is to obtain more models by sampling the neighborhood of the surrogate model. That is, replace Eq. (3) with:

$$\arg\max_{\mathbf{x}^{adv}} \quad \mathbb{E}_{h \in \mathcal{B}_a(f_\theta)} \left[\mathcal{J}(h(\mathbf{x}^{adv}), y) \right]. \tag{4}$$

However, since the hypothesis space is a function space, directly sampling the neighborhood of the surrogate model is challenging. Therefore, we take a step back: we perturb the surrogate model f_θ in the hypothesis space to obtain neighboring models f'. Next, we introduce two simple yet effective augmentation methods.

3.3. Operator Sampling

We consider perturbations of the form $f' = f_\theta \circ op$, where $op \colon \mathbb{R}^{H \times W \times C} \to \mathbb{R}^{H \times W \times C}$ is an input transformation operator, and $\circ$ denotes composition. Our goal is to construct a set of operators $\mathcal{P}$, corresponding to a set of neighboring surrogate models, whose distribution within the neighborhood is relatively balanced. In each iteration, we randomly select N_p operators from $\mathcal{P}$ for augmentation.

We first select a set of transformation operators that induce small perturbations to the surrogate model, forming the basic operator set $\mathcal{P}_b$. Then, we obtain some expanded sets through composition operations to increase the number of available operators. We define the k-th order extension set $\mathcal{P}_b^k$ of $\mathcal{P}$ as:

$$\mathcal{P}_b^k = \{op_1 \circ op_2 \circ \cdots \circ op_k \mid op_1, op_2, \ldots, op_k \in \mathcal{P}_b\}. \tag{5}$$

In particular, we define $\mathcal{P}_b^1 = \mathcal{P}_b$.

Clearly, the higher the order, the more operators are available, and the greater the perturbation on the surrogate model. Therefore, we control the number of operators at different orders to be equal, in order to maintain a hierarchical balance. Specifically, we introduce an integer list K_{list} to specify the order of each extension set, and randomly sample $|\mathcal{P}_b|$ operators from each selected extension set to form $\mathcal{P}$. We use the notation $S(A, n)$ to denote randomly sampling n times from the set A. Therefore, we have:

$$\mathcal{P} = \bigcup_{k \in K_{\text{list}}} S(\mathcal{P}_b^k, |\mathcal{P}_b|). \tag{6}$$

Algorithm 1 OPS

Input: Original image $\mathbf{x}^{ori}$; Ground truth label y; Attack budget ϵ; Iterations T and decay factor μ; Level list K_{list}; Radius list R_{list}; Number of operator samples N_p; Number of perturbation samples N_e.
Output: : Adversarial example $\mathbf{x}^{adv}$.

$\alpha = \epsilon/T$, $m_0 = 0$, $\mathbf{\Delta}_0 = \mathbf{0}$.
for $i = 0$ to $T - 1$ **do**
  $\bar{g} = \nabla_{\mathbf{x}} \mathcal{J}(f(\mathbf{x}^{ori} + \mathbf{\Delta}_i), y)$.
  Initialize the perturbation set $\mathcal{D}$ by Eq. (7).
  for δ in $S(\mathcal{D}, N_e)$ **do**
    Initialize the operator set $\mathcal{P}$ by Eq. (6).
    for op in $S(\mathcal{P}, N_p)$ **do**
      $\bar{g} = \bar{g} + \nabla_{\mathbf{x}} \mathcal{J}(f(op(\mathbf{x}^{ori} + \mathbf{\Delta}_i + \delta)), y)$.
    end for
  end for
  Calculate average gradient $\bar{g} = \frac{\bar{g}}{N_e \times N_p + 1}$.
  $m_{i+1} = \mu \cdot m_i + \frac{\bar{g}}{\|\bar{g}\|_1}$.
  $\mathbf{\Delta}_{i+1} = \mathbf{\Delta}_i + \text{Clip}_\epsilon\{\alpha \cdot \text{Sign}(m_{t+1})\}$.
end for
return $\mathbf{x}^{adv} = \mathbf{x}^{ori} + \mathbf{\Delta}_T$.

3.4. Perturbation Sampling

Another form is to add a perturbation δ before the input of f_θ, i.e., $f'(\mathbf{x}) = f_\theta(\mathbf{x} + \delta)$. Although this is possible, we did not include it in the operators discussed in the previous section because this form has better properties. In fact, it can be proven that if f_θ satisfies the Lipschitz condition, then $d(f_\theta, f') \leq C\delta$, where C is a fixed constant (see **Appendix**).

Similar to the previous section, our approach is to uniformly sample the same number of perturbations from a series of open balls with different radii $\mathcal{B}_r = \{\delta \mid \|\delta\|_\infty < r\}$, forming the perturbation set $\mathcal{D}$. Then, in each iteration, we randomly sample N_e perturbations from $\mathcal{D}$ for augmentation. We introduce the list R_{list} to specify the radii of the sequence of open balls and set the number of samples within each ball to N_e. Thus, $\mathcal{D}$ can be expressed as:

$$\mathcal{D} = \bigcup_{r \in R_{\text{list}}} S(\mathcal{B}_r, N_e). \tag{7}$$

3.5. Attack Algorithm

Based on the above discussion, formally, we ultimately replace Eq. (3) with:

$$\arg\max_{\mathbf{x}^{adv}} \quad \mathbb{E}_{(\mathcal{D},\mathcal{P})} \mathbb{E}_{(\delta, op) \in \mathcal{D} \times \mathcal{P}} \left[\mathcal{J}\left(f_\theta(op(\mathbf{x}^{adv} + \delta), y \right) \right]. \tag{8}$$

We adopt MI [5] to solve this problem. The complete algorithm is shown as Algorithm 1. We name it **O**perator-**P**erturbation-based **S**tochastic Optimization (**OPS**).

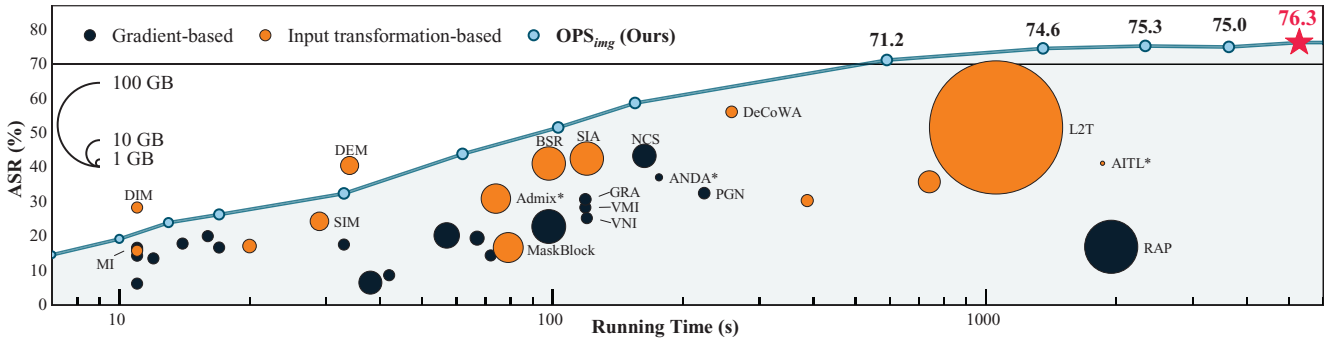

Figure 2. Attack success rate (ASR) (%) of adversarial examples crafted on ResNet-18 (CNN-based) when tested on ViT (Transformer-based). Each disk represents a different attack algorithm, with the size of the disk indicating the maximum memory overhead used during the entire attack process, and the x-axis representing the algorithm's runtime. For OPS, the first three disks from left to right correspond to OPS(4, 0, 0), OPS(7, 0, 0), and OPS(10, 0, 0), followed by 10 disks for OPS$(10, N_e, N_p)$, with the specific parameter settings being $N_e = N_p = 1, 2, 3, 4, 5, 10, 15, 20, 25, 30$. The method marked with * indicates that it only supports input with a batch size of 1.

Remark. The computational overhead of the algorithm primarily arises from the backpropagation during the attack. In OPS, the number of times this operation is performed is $T \cdot (N_p \cdot N_e + 1)$, so we use the notation OPS(T, N_p, N_e) to distinguish between different configurations.

3.6. Extension to 3D Point Clouds

The discussion in this section is not restricted to specific modalities. Our experiments include both the image version of OPS and the point cloud version of OPS, which we distinguish as OPS$_{img}$ and OPS$_{pc}$. The primary difference between the two lies in the basic operator set $\mathcal{P}_b$. For OPS$_{img}$, $\mathcal{P}_b$ comprises three types of transformations: (1) commonly used data augmentation transformations, (2) scale scaling used in SIM [23], and (3) random resizing and padding used in DIM [60]. For OPS$_{pc}$, due to limited research, we primarily use rigid body transformations. More details can be found in **Appendix**.

4. Experiments

4.1. Implementation

We use the publicly available benchmark TransferAttack [49] to evaluate OPS$_{img}$ and our implemented **3D Transfer Attack Benchmark** (**3DTAB**) to evaluate OPS$_{pc}$. 3DTAB is implemented in PyTorch and incorporates the latest research in point cloud classification and 3D adversarial domains. For specific details, please refer to **Appendix**. We report results on a workstation equipped with an Intel i9-10980XE (36) @ 4.600GHz and 128GB of memory. Our experiments were conducted on an RTX 3090 GPU.

4.2. Evaluation on Image Classification

We conduct empirical evaluations on the ImageNet dataset to evaluate the effectiveness of OPS$_{img}$. Further details on the experimental setup and analysis are presented below.

Dataset. Following previous works [46, 69], we evaluate OPS$_{img}$ on 1,000 images belonging to 1,000 categories from the validation set of the ImageNet dataset [40]. Unless otherwise specified, the batch size for the attacks in the experiments is uniformly set to 32.

Models. We evaluate OPS$_{img}$ on both normally trained models and robust models. For the normally trained models, we select eight widely used architectures: ResNet-18 [14], ResNet-101 [14], ResNeXt-50 [61], DenseNet-121 [16], ViT [6], PiT [15], Visformer [3], and Swin [29]. All models are pretrained on the ImageNet dataset. For the robust models, we consider three defense strategies: adversarial training [43, 53], provable defense [4], and input purification [20, 34], covering five different defense methods.

Baseline. OPS uses input transformations for augmentation, which are related to input transformation-based methods, while MI, used to solve the stochastic optimization problem, is related to gradient-based methods. Therefore, we focus on these two categories. Among the gradient-based methods, we compare four approaches: VMI [47], GRA [37], PGN [8], and NCS [39]. For input transformation methods, we also compare four approaches: DeCoWA [24], SIA [50], BSR [46], and L2T [69].

Parameters Settings. We keep the parameter settings of all baseline methods consistent with those in their respective papers. For OPS, the hyperparameters can be divided into two parts. The first part pertains to the hyperparameters for constructing the stochastic optimization problem in Eq. (8). We set the order list $K_{\text{list}} = [1, 2, 3, 4]$ and the radius list $R_{\text{list}} = [0.5, 1.0, 1.5, 2.0, 2.5, 3.0] \times \epsilon$. The variable parameters N_e and N_p are set to 5 by default. The second part involves the hyperparameters for solving the optimization problem in Eq. (8). We follow the default settings of MI, setting the perturbation budget $\epsilon = 16$, the number of iterations $T = 10$, the step size $\alpha = 1.6$, and the decay factor $\mu = 1$.

Category	Attack	ResNet-18 ⟹							
		ResNet-18	ResNet-101	ResNext-50	DenseNet-121	ViT	PiT	Visformer	Swin
Gradient-based	VMI (2021)	**100.0***	62.2	65.7	89.5	28.3	39.4	53.2	59.1
	GRA (2022)	**100.0***	66.2	72.0	94.2	30.7	41.0	54.8	63.6
	PGN (2023)	**100.0***	69.3	71.7	94.9	32.5	43.5	56.7	65.7
	NCS (2024)	**100.0***	78.8	82.2	96.9	43.3	53.7	67.9	74.8
Input transformation-based	SIA (2023)	**100.0***	86.4	89.9	99.3	42.5	59.1	77.2	76.7
	DeCoWA (2024)	**100.0***	84.8	88.0	98.7	56.1	65.2	79.8	79.8
	BSR (2024)	**100.0***	85.1	88.4	98.6	41.1	58.1	76.1	75.6
	L2T (2024)	**100.0***	88.6	90.0	98.5	51.6	64.7	79.5	79.9
OPS$_{img}$ (Ours)	OPS(10, 5, 5)	99.8*	88.8	90.3	98.6	58.7	68.1	81.4	82.1
	OPS(10, 10, 10)	**100.0***	94.8	95.6	99.7	71.2	77.1	88.7	89.9
	OPS(10, 30, 30)	**100.0***	**97.1**	**97.5**	**99.9**	**76.3**	**82.1**	**92.6**	**93.6**

Table 1. Attack success rates (%) of different attack algorithms on seven normally trained image classifiers, using ResNet-18 as the surrogate model. The best results are highlighted in bold, and * indicates white-box attacks.

Category	Attack	ResNet-18 ⟹							
		Inc-v3 (ens3)	Inc-v4 (ens4)	IncRes-v2 (ens)	ResNet-50 (fast-2px)	ResNet-50 (fast-4px)	ResNet-50 (RS)	ResNet-101 (HGD)	ResNet-101 (NRP)
Gradient-based	VMI (2021)	57.1	55.8	45.0	31.0	35.6	26.2	54.0	42.0
	GRA (2022)	66.5	66.9	51.8	40.1	40.9	34.6	59.4	53.6
	PGN (2023)	68.5	68.2	54.1	41.7	41.2	35.3	63.2	57.3
	NCS (2024)	78.0	77.6	63.0	41.7	40.8	36.0	73.9	64.2
Input transformation-based	SIA (2023)	87.5	87.4	78.5	46.2	44.4	35.3	86.9	57.3
	DeCoWA (2024)	79.5	77.5	64.7	37.0	38.9	28.7	80.3	52.3
	BSR (2024)	77.3	76.7	62.8	36.9	38.6	27.1	78.7	48.2
	L2T (2024)	86.7	87.1	76.7	44.1	43.0	33.6	87.6	62.2
OPS$_{img}$ (Ours)	OPS(10, 5, 5)	89.9	91.4	85.0	51.1	47.1	42.7	90.6	69.7
	OPS(10, 10, 10)	96.0	96.0	92.4	56.8	50.3	50.9	95.9	81.9
	OPS(10, 30, 30)	**97.3**	**97.8**	**95.5**	**61.2**	**52.2**	**57.8**	**97.8**	**86.4**

Table 2. Attack success rates (%) of different attack algorithms on eight image classifiers with various defenses, using ResNet-18 as the surrogate model. The best results are highlighted in bold.

4.2.1 Evaluation on Normally Trained Models

We first evaluate the adversarial transferability of OPS$_{img}$ on models trained normally. As shown in Tab. 1, compared to recent attack methods, our method achieves the highest transferability across all target models. Even the lightest configuration OPS$_{img}$(10, 5, 5) surpasses all baseline methods. Furthermore, in the most challenging cross-architecture attack scenario (***ResNet18*** $\Rightarrow$ ***ViT***), OPS$_{img}$ achieves an attack success rate of over 75%, far surpassing other methods. Delving deeper, we use this attack scenario as a lens to highlight the significant advantages of our method in terms of both performance and computational overhead in Fig. 2. Due to memory constraints, we set the batch size for L2T to 2, and correspondingly, scaling up its memory usage by a factor of 16.

Experiment Analysis. We observe that DIM and DEM surpassed our curve, with the latter being an improved algorithm based on the former. This phenomenon is primarily due to the fact that most operators are not as effective as DIM. The study by Zhao et al. indicates that DIM performs better in terms of input diversity. When N_p is relatively small, our method has a greater probability of not sampling DIM operators, resulting in lower outcomes. However, our method clearly has a higher ceiling, which also suggests that subsequent research can quickly evaluate the effectiveness of their methods based on our curve.

4.2.2 Evaluation on Robust Models

We further evaluate the effectiveness of OPS$_{img}$ on robust models. We consider three ensemble adversarial training models [43], two fast adversarial training models [53], a certified defense based on randomized smoothing (RS) [4], and two input purification methods: High-level Representation Guided Denoiser (HGD) [20] and Neural Representation Purifier (NRP) [34]. As shown in Tab. 2, Our method once again achieves the highest transferability across all approaches, demonstrating the practicality and effectiveness of OPS$_{img}$ in a wide range of applications.

Category	Attack	Used Budget (↓)	PointNet ⟹ PointNet	PointConv	DGCNN	PointCNN	CurveNet	Point-PN	PT-v1	PT-v2	RSCNN
White box-Attack	KNN	0.17	99.4*	21.3	18.4	11.3	34.5	31.2	16.4	29.7	25.5
	GeoA^3	0.08	85.7*	35.6	26.8	30.7	27.1	46.3	27.5	36.7	28.0
	HiT-ADV	0.11	90.3*	11.4	4.7	11.5	5.0	29.0	13.2	8.0	8.0
Transfer-Attack	AdvPC	**0.06**	**99.6***	39.7	19.7	23.6	20.4	37.3	15.8	36.6	16.1
	AOF	0.09	98.8*	64.8	41.9	50.0	39.1	59.8	28.6	60.9	35.3
	PF-Attack	0.16	93.5*	68.3	38.1	48.1	43.1	**61.8**	34.2	57.6	41.1
OPS$_{pc}$ (Ours)	OPS $_{\epsilon=0.06}$	**0.06**	93.4*	**76.8**	**50.0**	**67.5**	**44.8**	60.6	**38.2**	**67.3**	**47.2**
	OPS $_{\epsilon=0.18}$	0.18	100.0*	97.5	97.1	96.4	96.3	97.3	91.6	97.5	95.2

Table 3. Attack success rates (%) of different attack algorithms on nine normally trained point cloud classifiers, using PointNet as the surrogate model. The best results are highlighted in bold, and * indicates white-box attacks. We evaluated the attack budget used by adversarial examples generated by different methods. We use OPS $_{\epsilon=0.06}$ to compare with other methods, while the results of OPS $_{\epsilon=0.18}$ are provided for reference only.

Defense	Attack	PointNet ⟹ PointNet	PointConv	DGCNN	PointCNN	CurveNet	Point-PN	PT-v1	PT-v2	RSCNN
Dup-Net	KNN	22.6*	44.0	38.5	77.0	13.5	51.0	81.0	30.0	14.0
	GeoA^3	42.9*	67.0	56.5	80.5	32.0	66.0	83.5	55.5	29.0
	HiT-ADV	53.5*	47.5	41.5	80.5	11.0	53.5	81.0	31.0	13.0
	AdvPC	36.7*	75.5	54.0	84.5	16.3	66.0	81.0	51.5	16.0
	AOF	30.4*	50.2	30.7	37.8	31.3	48.1	26.0	33.4	24.6
	PF-Attack	56.0*	87.0	75.0	87.0	38.5	77.0	83.0	68.0	38.5
	OPS$_{pc}$ (Ours)	**88.0***	**94.0**	**79.5**	**88.0**	**45.5**	**78.5**	**86.0**	**72.0**	**53.0**
IF-Defense	KNN	15.9*	23.0	17.0	25.7	17.4	29.3	18.0	19.5	13.6
	GeoA^3	31.2*	37.8	27.7	35.1	27.8	39.6	27.1	31.2	25.7
	HiT-ADV	40.0*	36.7	22.3	31.6	21.2	41.8	22.7	29.1	19.1
	AdvPC	17.8*	34.0	22.1	29.6	18.5	35.7	20.7	24.2	15.9
	AOF	30.4*	50.2	30.7	37.8	31.3	48.1	26.0	33.4	24.6
	PF-Attack	38.2*	55.1	36.3	44.7	38.5	51.8	30.9	40.8	35.1
	OPS$_{pc}$ (Ours)	**61.7***	**61.4**	**43.7**	**45.7**	**41.1**	**56.7**	**35.5**	**43.7**	**40.7**

Table 4. Attack success rates (%) of different attack algorithms against point cloud defenses Dup-Net and IF-Defense, using PointNet as the surrogate model. The best results are highlighted in bold, and * indicates white-box attacks.

4.3. Evaluation on 3D Point Cloud Classification

In this section, we evaluate the effectiveness of OPS$_{pc}$ on the ModelNet40 dataset, with further details regarding the experimental setup and analysis provided below.

Dataset. We use the preprocessed ModelNet40 [36, 57] for model training and evaluation of the attack methods. Unlike previous studies [12, 13, 25], we expand the number of test samples from 250 to 1,000. Specifically, we randomly select 25 samples from each category for evaluation.

Model. Similar to the experimental setup in images, we evaluate OPS$_{pc}$ on both normally trained models and robust models. For the normally trained models, we conduct experiments using nine different architectures, including: PointNet [36], DGCNN [51], PointConv [55], PointCNN [19], CurveNet [33], PT-v1 [65], PT-v2 [56], RSCNN [28] and Point-PN [64]. For the robust models, we consider two classic defense methods: DUPNet [68] and IF-Defense [58].

Baseline. The baseline we adopt consists of two parts: (1) White-box attack in 3D point cloud: KNN [44], GeoA^3 [52], HiT-Adv [30] and (2) Transfer-based attack in 3D point cloud: AdvPC [12], AOF [25], PF-Attack [13].

Parameters Settings. Following previous works [13], We constrain the perturbation norm for each point to be less than 0.18. We maintain the parameter settings for all baseline methods consistent with those in their respective papers. For OPS$_{pc}$, the settings are largely the same as those for OPS$_{img}$, with the only modification being the radius list $R_{\text{list}} = [0.5, 0.75, 1.0, 1.25] \times \epsilon$.

Evaluation on Normally Trained Models. Similar to the experiments in the images, we first compare against the baseline methods on the normally trained models. As shown in Tab. 3, the results of the default setting of OPS$_{pc}$ (i.e., OPS(10, 5, 5)) have significantly surpassed all baseline methods. We believe this is partly due to the insufficient use of the budget in previous methods. To ensure a fair com-

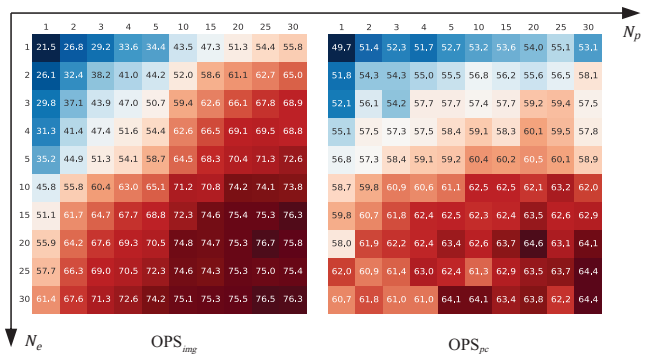

Figure 3. Heatmap visualization of the impact of parameters N_e and N_p on attack success rate (%). N_e and N_p are set to 1, 2, 3, 4, 5, 10, 15, 20, 25, and 30, respectively.

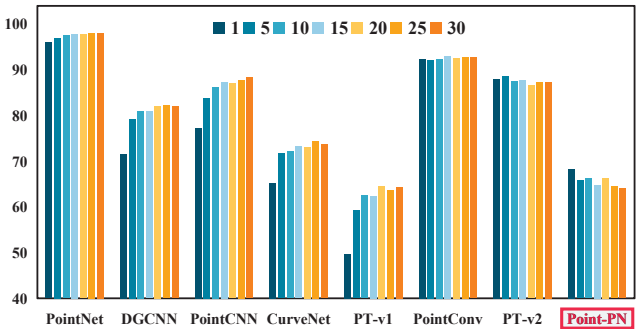

Figure 4. Attack success rate (%) of OPS_{pc} across different 3D point cloud models under varying computational overhead (N_e = N_p = 1, 5, 10, 15, 20, 25, 30)

parison, we limit the budget to 0.06 and compare it again with the baseline methods. The results indicate that OPS_{pc} achieves a higher attack success rate with a lower budget, demonstrating the superiority of our method.

Evaluation on Robust Models. For robust models, we still set the budget of OPS_{pc} to 0.06 for comparison with the baseline methods. As shown in Tab. 4, OPS_{pc} achieves the best results under the defenses of Dup-Net and IF-Defense. Even in a white-box setting, OPS_{pc} demonstrates a significant advantage. We notice that after defense with Dup-Net, the attack success rate of most models does not decrease but rather increases. This may be attributed to the upsampling operation of Dup-Net, which increases the number of points in the upsampled point cloud to four times that of the original. This significantly alters the neighborhood information of the point cloud, causing varying degrees of interference with the model's predictions.

4.4. Ablation study

Among all the parameters in the OPS algorithm, the two variable parameters N_e and N_p have the most direct impact on the results. Ideally, as the number of samples increases, the solution accuracy of Eq. (8) will gradually improves, resulting in higher transferability. We investigate this in depth in both images and point clouds. In the experiments of this section, we set the budget for OPS_{pc} to $\ell_\infty = 0.06 \approx 16/255$.

On the numbers of samples. We first investigate the impact of N_e and N_p on the success rate in specific attack scenarios. For OPS_{img}, we use the attack scenario ***ResNet-18*** $\Rightarrow$ ***ViT***. For OPS_{pc}, we adopt the attack scenario ***PointNet*** $\Rightarrow$ ***PT-v1***. As shown in Fig. 3, the transferability of OPS_{img} increases with the growth of both parameters, reaching a plateau around 20, which aligns with our expectations. The heatmap indicates that there is no significant priority between the two parameters; when the computational load is comparable, whether we increase N_e or N_p first has a negligible impact on the results. Therefore, in practice, we set $N_e = N_p$. On the other hand, although OPS_{pc} shows a similar trend, the increase in N_e brings a greater improvement than the increase in N_p. This may be related to the relatively uniform set of operators we used. The basic operator set of OPS_{pc} primarily consists of rigid transformations, which leads to homogenization of the neighbors detected (prob), resulting in poorer transferability of the generated examples in the hypothesis space.

On different cross-architecture attacks. We further investigate the impact of computational cost on attack success rates in more cross-architecture scenarios. As shown in Fig. 4, for most models, the attack success rate steadily increases with the amount of computation, then fluctuates around an upper limit. Point-PN, however, exhibits an entirely opposite trend: its attack success rate decreases as computation increases. This model represents recent work in parameter-free learning for point clouds and differs significantly in design from other models, potentially offering insights into novel defense methods.

5. Conclusion

In this study, we observed a mirroring relationship between the research on model generalization and transferability, and explored the possibility of enhancing the surrogate model within the hypothesis space to improve transferability. We designed an augmentation-based attack, OPS, which enhances the surrogate model through input transformation operators and random perturbations. Extensive experiments show that, compared to existing state-of-the-art attacks, OPS offers significant advantages in both transferability and computational efficiency. By shifting the focus from local interactions between data and models to their global relationship, we gain insights into the intrinsic link between transferability and generalization, providing a new perspective for future work.

Acknowledgements. This work is supported in part by the National Natural Science Foundation of China (No. 62401225, 62471415); the Natural Science Foundation of Xiamen, China (No. 3502Z202472018); the Fujian Provincial Natural Science Foundation of China (No. 2024J01115, 2023J01004); the Jimei University Scientific Research Start-up Funding Project (No. ZQ2024034); the Fundamental Research Funds for the Central Universities (No. 20720230033); the PDL (No. 2022-PDL-12); the Xiaomi Young Talents Program.

References

[1] Junyoung Byun, Seungju Cho, Myung-Joon Kwon, Hee-Seon Kim, and Changick Kim. Improving the transferability of targeted adversarial examples through object-based diverse input. In *CVPR*, pages 15244–15253, 2022. 2

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

[3] Zhengsu Chen, Lingxi Xie, Jianwei Niu, Xuefeng Liu, Longhui Wei, and Qi Tian. Visformer: The vision-friendly transformer. In *ICCV*, pages 589–598, 2021. 5

[4] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*, pages 1310–1320. PMLR, 2019. 1, 3, 5, 6

[5] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *CVPR*, pages 9185–9193, 2018. 2, 4

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

[7] Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia, 2018. Association for Computational Linguistics. 1

[8] Zhijin Ge, Hongying Liu, Wang Xiaosen, Fanhua Shang, and Yuanyuan Liu. Boosting adversarial transferability by achieving flat local maxima. *NeurIPS*, 36:70141–70161, 2023. 2, 5, 6

[9] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, pages 3354–3361. IEEE, 2012. 1

[10] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015. 1, 2, 3

[11] Sven Gowal, Krishnamurthy Dj Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. Scalable verified training for provably robust image classification. In *ICCV*, pages 4842–4851, 2019. 3

[12] Abdullah Hamdi, Sara Rojas, Ali Thabet, and Bernard Ghanem. Advpc: Transferable adversarial perturbations on 3d point clouds. In *ECCV*, pages 241–257. Springer, 2020. 2, 3, 7

[13] Bangyan He, Jian Liu, Yiming Li, Siyuan Liang, Jingzhi Li, Xiaojun Jia, and Xiaochun Cao. Generating transferable 3d adversarial point cloud via random perturbation factorization. In *AAAI*, pages 764–772, 2023. 3, 7

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 5

[15] Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. In *ICCV*, pages 11936–11945, 2021. 5

[16] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, pages 4700–4708, 2017. 5

[17] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC, 2018. 1, 2, 3

[18] Wen Li, Yuyang Yang, Shangshu Yu, Guosheng Hu, Chenglu Wen, Ming Cheng, and Cheng Wang. Diffloc: Diffusion model for outdoor lidar localization. In *CVPR*, pages 15045–15054, 2024. 1

[19] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *NeurIPS*, 31, 2018. 7

[20] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. In *CVPR*, pages 1778–1787, 2018. 2, 3, 5, 6

[21] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2019. 1

[22] Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E Hopcroft. Nesterov accelerated gradient and scale invariance for adversarial attacks. In *ICLR*, 2020. 2

[23] Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E Hopcroft. Nesterov accelerated gradient and scale invariance for adversarial attacks. In *ICLR*, 2020. 2, 5

[24] Qinliang Lin, Cheng Luo, Zenghao Niu, Xilin He, Weicheng Xie, Yuanbo Hou, Linlin Shen, and Siyang Song. Boosting adversarial transferability across model genus by deformation-constrained warping. In *AAAI*, pages 3459–3467, 2024. 2, 5, 6

[25] Binbin Liu, Jinlai Zhang, and Jihong Zhu. Boosting 3d adversarial attacks with attacking on frequency. *IEEE Access*, 10:50974–50984, 2022. 2, 3, 7

[26] Daniel Liu, Ronald Yu, and Hao Su. Extending adversarial attacks and defenses to deep 3d point cloud classifiers. In *ICIP*, pages 2279–2283. IEEE, 2019. 3

[27] Hongbin Liu, Jinyuan Jia, and Neil Zhenqiang Gong. Pointguard: Provably robust 3d point cloud classification. In *CVPR*, pages 6186–6195, 2021. 3

[28] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *CVPR*, pages 8895–8904, 2019. 7

[29] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 10012–10022, 2021. 5

[30] Tianrui Lou, Xiaojun Jia, Jindong Gu, Li Liu, Siyuan Liang, Bangyan He, and Xiaochun Cao. Hide in thicket: Generating imperceptible and rational adversarial perturbations on 3d point clouds. In *CVPR*, pages 24326–24335, 2024. 3, 7

[31] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019. 2

[32] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, pages 4460–4470, 2019. 3

[33] AAM Muzahid, Wanggen Wan, Ferdous Sohel, Lianyao Wu, and Li Hou. Curvenet: Curvature-based multitask learning deep networks for 3d object recognition. *IEEE/CAA Journal of Automatica Sinica*, 8(6):1177–1187, 2020. 7

[34] Muzammal Naseer, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Fatih Porikli. A self-supervised approach for adversarial robustness. In *CVPR*, pages 262–271, 2020. 2, 5, 6

[35] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Animashree Anandkumar. Diffusion models for adversarial purification. In *International Conference on Machine Learning*, pages 16805–16827. PMLR, 2022. 2

[36] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 652–660, 2017. 7

[37] Zeyu Qin, Yanbo Fan, Yi Liu, Li Shen, Yong Zhang, Jue Wang, and Baoyuan Wu. Boosting the transferability of adversarial attacks with reverse adversarial perturbation. *NeurIPS*, 35:29845–29858, 2022. 2, 5, 6

[38] Zeyu Qin, Yanbo Fan, Yi Liu, Li Shen, Yong Zhang, Jue Wang, and Baoyuan Wu. Boosting the transferability of adversarial attacks with reverse adversarial perturbation. *NeurIPS*, 35:29845–29858, 2022. 2

[39] Chunlin Qiu, Yiheng Duan, Lingchen Zhao, and Qian Wang. Enhancing adversarial transferability through neighborhood conditional sampling, 2024. 2, 5, 6

[40] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115:211–252, 2015. 5

[41] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823, 2015. 1

[42] Luchuan Song, Zheng Fang, Xiaodan Li, Xiaoyi Dong, Zhenchao Jin, Yuefeng Chen, and Siwei Lyu. Adaptive face forgery detection in cross domain. In *ECCV*, pages 467–484. Springer, 2022. 1

[43] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *ICLR*, 2018. 1, 2, 5, 6

[44] Tzungyu Tsai, Kaichen Yang, Tsung-Yi Ho, and Yier Jin. Robust adversarial objects against deep learning models. In *AAAI*, pages 954–962, 2020. 3, 7

[45] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, pages 5265–5274, 2018. 1

[46] Kunyu Wang, Xuanran He, Wenxuan Wang, and Xiaosen Wang. Boosting adversarial transferability by block shuffle and rotation. In *CVPR*, pages 24336–24346, 2024. 2, 5, 6

[47] Xiaosen Wang and Kun He. Enhancing the transferability of adversarial attacks through variance tuning. In *CVPR*, pages 1924–1933, 2021. 2, 5, 6

[48] Xiaosen Wang, Xuanran He, Jingdong Wang, and Kun He. Admix: Enhancing the transferability of adversarial attacks. In *ICCV*, pages 16158–16167, 2021. 2

[49] Xiaosen Wang, Zeyuan Yin, Zeliang Zhang, Kunyu Wang, Zhijin Ge, and Yuyang Luo. Transferattack. https://github.com/Trustworthy-AI-Group/TransferAttack/tree/main, 2023. Accessed: 2024-07-26. 5

[50] Xiaosen Wang, Zeliang Zhang, and Jianping Zhang. Structure invariant transformation for better adversarial transferability. In *ICCV*, pages 4607–4619, 2023. 2, 5, 6

[51] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM TOG*, 38(5): 1–12, 2019. 7

[52] Yuxin Wen, Jiehong Lin, Ke Chen, CL Philip Chen, and Kui Jia. Geometry-aware generation of adversarial point clouds. *IEEE TPAMI*, 44(6):2984–2999, 2020. 3, 7

[53] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. In *ICLR*, 2020. 2, 5, 6

[54] Dongxian Wu, Yisen Wang, Shu-Tao Xia, James Bailey, and Xingjun Ma. Skip connections matter: On the transferability of adversarial examples generated with resnets. In *ICLR*, 2020. 2

[55] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *CVPR*, pages 9621–9630, 2019. 7

[56] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. *NeurIPS*, 35:33330–33342, 2022. 7

[57] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, pages 1912–1920, 2015. 7

[58] Ziyi Wu, Yueqi Duan, He Wang, Qingnan Fan, and Leonidas J. Guibas. If-defense: 3d adversarial point cloud defense via implicit function based restoration, 2021. 3, 7

[59] Chong Xiang, Charles R Qi, and Bo Li. Generating 3d adversarial point clouds. In *CVPR*, pages 9136–9144, 2019. 1, 3

[60] 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 *CVPR*, pages 2730–2739, 2019. 2, 5

[61] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, pages 1492–1500, 2017. 5

[62] Zheng Yuan, Jie Zhang, and Shiguang Shan. Adaptive image transformations for transfer-based adversarial attack. In *ECCV*, pages 1–17. Springer, 2022. 2

[63] Huan Zhang, Hongge Chen, Chaowei Xiao, Sven Gowal, Robert Stanforth, Bo Li, Duane Boning, and Cho-Jui Hsieh. Towards stable and efficient training of verifiably robust neural networks. In *ICLR*, 2020. 3

[64] Renrui Zhang, Liuhui Wang, Yali Wang, Peng Gao, Hongsheng Li, and Jianbo Shi. Starting from non-parametric networks for 3d point cloud analysis. In *CVPR*, pages 5344–5353, 2023. 7

[65] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *ICCV*, pages 16259–16268, 2021. 7

[66] Zhengyu Zhao, Zhuoran Liu, and Martha Larson. On success and simplicity: A second look at transferable targeted attacks. *NeurIPS*, 34:6115–6128, 2021. 2

[67] Zhengyu Zhao, Hanwei Zhang, Renjue Li, Ronan Sicre, Laurent Amsaleg, Michael Backes, Qi Li, and Chao Shen. Revisiting transferable adversarial image examples: Attack categorization, evaluation guidelines, and new insights, 2023. 6

[68] Hang Zhou, Kejiang Chen, Weiming Zhang, Han Fang, Wenbo Zhou, and Nenghai Yu. Dup-net: Denoiser and upsampler network for 3d adversarial point clouds defense. In *ICCV*, pages 1961–1970, 2019. 3, 7

[69] Rongyi Zhu, Zeliang Zhang, Susan Liang, Zhuo Liu, and Chenliang Xu. Learning to transform dynamically for better adversarial transferability. In *CVPR*, pages 24273–24283, 2024. 2, 5, 6

[70] Yao Zhu, Yuefeng Chen, Xiaodan Li, Kejiang Chen, Yuan He, Xiang Tian, Bolun Zheng, Yaowu Chen, and Qingming Huang. Toward understanding and boosting adversarial transferability from a distribution perspective. *IEEE TIP*, 31: 6487–6501, 2022. 2