

Appendices

A. Details about OPS

A.1. Proofs Omitted in the Methodology Section

We define an open set in the hypothesis space \mathcal{H} based on the open ball defined in Sec 3.2. Specifically, we call a set $H \subseteq \mathcal{H}$ an open set if it satisfies the following property: for any $x \in H$, there exists $r > 0$ such that $\mathcal{B}_r(x) \subseteq H$.

Lemma 1. *A subset H of the hypothesis space \mathcal{H} is a nonempty open set if and only if H can be written as a union of open balls.*

Proof. Suppose H is an open set. Then, for any $x \in H$, there exists $r_x > 0$ such that $\mathcal{B}_{r_x}(x) \subseteq H$. On one hand, $x \in \mathcal{B}_{r_x}(x)$, so:

$$H \subseteq \bigcup_{x \in H} \mathcal{B}_{r_x}(x).$$

On the other hand, $\mathcal{B}_{r_x}(x) \subseteq H$, hence:

$$\bigcup_{x \in H} \mathcal{B}_{r_x}(x) \subseteq H.$$

Therefore, H can be expressed as the union of open balls $\mathcal{B}_{r_x}(x)$. Next, suppose $H = \bigcup_{x \in \mathcal{I}} \mathcal{B}_{r_x}(x)$, where \mathcal{I} is an index set. For any $y \in H$, there exists $x \in \mathcal{I}$ such that $y \in \mathcal{B}_{r_x}(x)$. Let $d(x, y) = s < r_x$. It follows that $\mathcal{B}_{r_x-s}(y) \subseteq \mathcal{B}_{r_x}(x) \subseteq H$. This is because, for any $z \in \mathcal{B}_{r_x-s}(y)$, we have:

$$d(x, z) \leq d(x, y) + d(y, z) < s + (r_x - s) = r_x,$$

which implies $z \in \mathcal{B}_{r_x}(x)$. Thus, the necessity of the condition follows, and H is an open set. \square

Theorem 1. *The neighborhood system defined in Sec 3.2 induces a topology on the hypothesis space.*

Proof. We verify the three axioms of topology one by one:

1. *The empty set \emptyset and the entire space \mathcal{H} are open.*
By definition, \emptyset has no elements and trivially satisfies any condition. The entire space \mathcal{H} contains all open balls $\mathcal{B}_r(x) \subseteq \mathcal{H}$. Therefore, both sets are open.
2. *The union of any collection of open sets is open.*
By Lemma 1, each open set can be written as a union of open balls. Thus, the union of open sets is also a union of open balls, and hence it is open.
3. *The intersection of a finite number of open sets is open.*
Let H_1, H_2, \dots, H_n be open sets. For any $x \in \bigcap_{i=1}^n H_i$, since each H_i is open, there exists $r_i > 0$ such that $\mathcal{B}_{r_i}(x) \subseteq H_i$. Let $r = \min\{r_1, r_2, \dots, r_n\}$. Then $\mathcal{B}_r(x) \subseteq \bigcap_{i=1}^n H_i$, which implies that $\bigcap_{i=1}^n H_i$ is open.

The above confirms that the neighborhood system satisfies the topology axioms. \square

Theorem 2. *If the surrogate model f_θ satisfies the Lipschitz condition, i.e., there exists a constant $C > 0$ such that for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^{H \times W \times C}$,*

$$\|f_\theta(\mathbf{x}_1) - f_\theta(\mathbf{x}_2)\| \leq C\|\mathbf{x}_1 - \mathbf{x}_2\|,$$

then for any perturbation δ , the difference between f_θ and the perturbed model $f'(\mathbf{x}) = f_\theta(\mathbf{x} + \delta)$ satisfies:

$$d(f_\theta, f') \leq C\|\delta\|,$$

where $\|\cdot\|$ denotes a norm.

Proof. By definition, the difference between f_θ and f' is:

$$d(f_\theta, f') = \int |\mathcal{J}(f_\theta(\mathbf{x}), y) - \mathcal{J}(f_\theta(\mathbf{x} + \delta), y)| \cdot p(\mathbf{x}) \, d\mathbf{x}.$$

Let $C_f > 0$ be the Lipschitz constant of f_θ and let $C_{\mathcal{J}} > 0$ be the Lipschitz constant of \mathcal{J} . Then, we have:

$$|\mathcal{J}(f_\theta(\mathbf{x}), y) - \mathcal{J}(f_\theta(\mathbf{x} + \delta), y)| \leq C_{\mathcal{J}} \cdot \|f_\theta(\mathbf{x}) - f_\theta(\mathbf{x} + \delta)\|.$$

Applying the Lipschitz condition of f_θ :

$$\|f_\theta(\mathbf{x}) - f_\theta(\mathbf{x} + \delta)\| \leq C_f \|\delta\|.$$

Substituting this back, we obtain:

$$|\mathcal{J}(f_\theta(\mathbf{x}), y) - \mathcal{J}(f_\theta(\mathbf{x} + \delta), y)| \leq C_{\mathcal{J}} \cdot C_f \|\delta\|.$$

Thus,

$$d(f_\theta, f') \leq \int C_{\mathcal{J}} \cdot C_f \|\delta\| \cdot p(\mathbf{x}) \, d\mathbf{x}.$$

Since $\int p(\mathbf{x}) \, d\mathbf{x} = 1$, we have:

$$d(f_\theta, f') \leq C_{\mathcal{J}} \cdot C_f \|\delta\|.$$

Defining $C = C_{\mathcal{J}} \cdot C_f$, we conclude:

$$d(f_\theta, f') \leq C\|\delta\|.$$

This completes the proof. \square

A.2. Discussion: OPS vs. IT-Based Methods

For convenience of expression, we introduce the notation **IT** (Input-Transformation) here. As described in the Related Work, the guiding principle of IT-based methods is input diversity. Specifically, multiple transformed images are generated through different ITs and then fed into the surrogate model to compute the average gradient for updating the adversarial example. This leads to two limitations: (1) **Input expansion** results in computational graph expansion, requiring larger GPU memory. (2) **Gradient coupling**

of similar operators. Most methods apply ‘different ITs’, which are essentially similar transformations under different parameter settings. The gradients obtained from similar operators are quite similar, leading to lower computational efficiency.

In OPS, our goal is not to increase input diversity, but rather to emphasize increasing gradient diversity while ensuring the effectiveness of the gradient. Therefore, the operator set \mathcal{P} in OPS contains different types of IT operators, and a random operator is selected each time to compute the gradient. This is similar to the shuffle operation in ACE [6], which not only avoids input inflation but also achieves gradient decoupling, thereby surpassing all the baseline methods in terms of memory usage and computational efficiency.

A.3. Details about the Basic Operator Set

We approximate the original dataset using the 1,000 samples employed for attacks to estimate $d(f_\theta, f_\theta \circ op)$. Based on the estimation results, we select a batch of operators that induce relatively small perturbations to the surrogate model, forming the basic operator set \mathcal{P}_b . Specifically: For operators in OPS_{img} , the selection criterion is $d(f_\theta, f_\theta \circ op) < 2$. While for operators in OPS_{pc} , the selection criterion is $d(f_\theta, f_\theta \circ op) < 0.05$.

Basic Operator Set for OPS_{img}

The basic operator set for image data includes three categories of transformations:

1. **Common Data Augmentation Transformations.** Including vertical flip, horizontal flip, vertical shift, horizontal shift, and rotation. For rotation, the angles are selected as: $\pm 5^\circ, \pm 15^\circ, \pm 45^\circ, \pm 90^\circ, 180^\circ$
2. **Scaling Transformations Inspired by SIM.** Instead of using the scaling factor $\gamma = 1/2^i$ as in SIM, we adopt a simplified approach with $\gamma = 1/i$, where $i \in \{2, 3, 4, 5, 6, 7, 8\}$.
3. **Random Resizing and Padding Inspired by DIM.** The resizing operation uses the following scale factors: 1.1, 1.3, 1.5, 1.7, 1.9, 2.1, 2.3, 2.5, 2.7, 2.9.

Basic Operator Set for OPS_{pc}

For point cloud data, the basic operator set comprises three rigid transformations and one point perturbation operator:

1. **Scaling.** The scaling operation adjusts the size of the point cloud by a scaling factor s . The scaling factors chosen are:

$$s \in \{0.8, 0.85, 0.9, 0.95, 1.05, 1.1, 1.15\}.$$

2. **Rotation.** The rotation operation applies rotations along the three axes (x , y , and z) of the point cloud. The rotation angles α , β , and γ are chosen from the following

set of parameter combinations:

$$(\alpha, \beta, \gamma) \in \{(0.5, -1.0, 0), (0, 0, 0.5), (-1.0, 0.5, 0), (1.5, 0, -1.0), (1.5, 0, -0.5), (1.0, 1.5, -0.5), (-1.0, 1.5, 1.0)\}.$$

3. **Translation.** The translation operation shifts the entire point cloud along the x , y , and z axes. The translation vectors selected are:

$$(x, y, z) \in \{(0.02, 0.05, -0.04), (0.04, 0, 0), (0, 0.01, -0.05), (-0.04, 0.03, 0), (-0.04, 0.03, -0.03), (0.03, 0.05, 0.01), (-0.03, 0, 0.05)\}.$$

4. **Point Perturbation.** The point perturbation operator selects a random point within the point cloud and applies a scaling factor r to it. The available scaling factors are:

$$r \in \{0, 0.3, 0.6, 0.9, 1.2, 1.5\}.$$

B. 3D Transfer Attack Benchmark

In the field of 3D adversarial transferability research, there is currently a lack of publicly available evaluation benchmarks. This absence hinders fair comparisons between different methods and leads to poor reproducibility of results. To address this issue, we propose **3DTAB (3D Transfer Attack Benchmark)**, a comprehensive benchmark that incorporates the latest advancements in 3D point cloud classification and adversarial attack research. It provides a unified framework for model training and attack evaluation, enabling consistent and reliable comparisons across methods. Furthermore, as white-box attacks can be regarded as a special case of the transfer attack where the surrogate model is identical to the target model, 3DTAB also supports the evaluation of white-box attack techniques.

B.1. Threat Model

The default threat model in 3DTAB is as follows:

1. **Untargeted attacks** under the **transfer attack**. The attacker generates adversarial examples using a surrogate model and then applies them to attack the target model.
2. The size of the perturbation Δ satisfies a **hard constraint**, with the default setting being $\|\Delta\|_\infty \leq 0.06$.
3. The attack method used by the attacker is **point perturbation**.

B.2. Dataset Setting

Following previous studies, we used the preprocessed ModelNet40 [40] from Qi et al. for model training and attack method evaluation. For model training, we employed

farthest point sampling to sample each sample to 1024 points and maintained consistency in data augmentation. For the evaluation of attack methods, we randomly selected 25 samples from each category, totaling 1000 samples, to form a small dataset for attack evaluation, referred to as **ModelNet40-mini**.

B.3. Model Setting

Previous research has generally employed small-scale models, utilizing PointNet [26], PointNet++ (SSG), PointNet++ (MSG) [27], and DGCNN [36], for a total of 4 models. In this study, we expand this number to 20, encompassing the research results from 14 studies including PointConv [38], CurveNet [25], PointCAT [15], PT [45], PCT [9], PointMLP [24], VN-PointNet [3], among others. The training settings and accuracy for each model are shown in Tab. 2

B.4. Attacker and Defender

We have integrated commonly used baselines and the latest attack methods in 3D point cloud attacks, including: 3D-Adv [42], KNN [30], GeoA³ [37], HiT-Adv [23], AdvPC [10], AOF [20], and PF-Attack [11]. For defense methods, we provide four commonly used defenses: SOR, SRS, Dup-Net [46], and IF-Defense [41]. We are also actively porting image-based transfer attacks to point clouds, with ported methods including: FGSM [8], MI [4], NI [17], VMI [32], EMI [33], GRA [28], SIM [18], IR [34], etc. As seen, most of these are gradient-based methods, as many input transformation-based methods are closely tied to the modality and are difficult to adapt to point clouds.

C. More experimental results

C.1. Experiments on Vision-Language Models

We conducted experiments on CLIP’s zero-shot classification task (Tab. 1) to assess the impact of our adversarial examples. The results demonstrate that our examples cause a greater disturbance to CLIP’s performance compared to existing methods, highlighting the effectiveness of our approach in transferring adversarial attacks to vision-language models.

Methods	No Attack	VMI	NCS	L2T	OPS-5	OPS-30
Accuracy(%)	77.2	37.3	24.9	16.7	13.7	5.9
Change(↓)	0.0	-39.9	-52.3	-60.5	-63.5	-71.3

Table 1. The impact of different methods on Clip accuracy.

C.2. Experiments on Different Surrogate Models

To analyze the impact of different surrogate models on the transferability of adversarial examples, we conducted ma-

trix experiments on ResNet-18, ResNet-101 [12], ResNext-50 [43], DenseNet-121 [14], ViT [5], PiT [13], ViFormer [1], and Swin [22], and reported their attack success rates, as shown in Tab. 3, Tab. 4, Tab. 5, Tab. 6, Tab. 7, Tab. 8, Tab. 9, Tab. 10. The experimental results show that the proposed method exhibits excellent attack transferability across all surrogate models and significantly outperforms existing methods on all target models. This result strongly validates the effectiveness of our method in generating adversarial examples across architectures and tasks.

Model	LR	Epochs	Batch Size	Optimizer	Scheduler	Acc (%)	Acc-mini (%)
PointNet (2017)	0.001	200	600	Adam	CosineAnnealingLR	89.71	91.8
PointNet++ (SSG) (2017)	0.001	200	100	Adam	CosineAnnealingLR	91.13	95.0
PointNet++ (MSG) (2017)	0.001	200	50	Adam	CosineAnnealingLR	92.75	96.1
PointCNN (2018)	0.001	200	800	Adam	CosineAnnealingLR	89.18	89.3
PointConv (2019)	0.001	200	150	Adam	CosineAnnealingLR	91.65	93.0
DGCNN (2019)	0.001	200	50	Adam	CosineAnnealingLR	92.54	95.8
RSCNN (2019)	0.001	200	50	Adam	LambdaLR	91.73	94.5
CurveNet (2020)	0.001	200	200	Adam	CosineAnnealingLR	92.30	95.4
PT (2021)	0.001	200	50	Adam	StepLR	91.13	94.0
PCT (2021)	0.001	200	50	Adam	StepLR	91.69	94.6
VN-PointNet (2021)	0.01	200	50	SGD	StepLR	80.06	78.0
VN-DGCNN (2021)	0.1	250	40	SGD	CosineAnnealingLR	89.02	89.0
PointMLP (2022)	0.1	300	40	SGD	CosineAnnealingLR	93.40	96.5
PointMLP (Elite) (2022)	0.1	300	40	SGD	CosineAnnealingLR	92.83	95.7
PT-v1-26 (2021)	0.05	200	50	SGD	MultiStepLR	90.60	89.2
PT-v1-38 (2021)	0.05	200	50	SGD	MultiStepLR	90.80	94.4
PT-v1-50 (2021)	0.05	200	50	SGD	MultiStepLR	92.10	94.4
PT-v2 (2022)	0.05	200	50	SGD	MultiStepLR	92.83	94.5
Point-PN (2023)	0.001	300	50	Adam	CosineAnnealingLR	91.73	94.0
PointCAT (2024)	0.01	250	50	SGD	CosineAnnealingLR	92.18	95.3

Table 2. Training settings and accuracy for different models. We report the accuracy on ModelNet40 (**Acc**) and ModelNet40-mini (**Acc-mini**). The PT-v1 series is derived from the implementation of PT [45] in Pointcept [2]. For detailed optimizer and scheduler settings, please refer to the code.

Category	Attack	ResNet-18 \implies							
		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
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

Table 3. 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-101 \implies							
		ResNet-18	ResNet-101	ResNext-50	DenseNet-121	ViT	PiT	Visformer	Swin
Gradient-based	VMI (2021)	62.3	88.4*	65.3	66.5	32.8	43.6	49.6	51.3
	GRA (2022)	80.9	92.5*	81.9	81.9	56.6	66.8	70.6	70.2
	PGN (2023)	86.5	96.5*	89.3	88.7	63.4	72.4	78.3	77.9
	NCS (2024)	80.2	91.3*	81.7	81.8	59.7	68.1	73.0	72.3
Input transformation-based	SIA (2023)	86.4	95.0*	90.0	89.0	53.1	71.8	79.2	78.0
	DeCoWA (2024)	93.1	96.9*	91.6	94.5	65.0	76.4	84.7	82.8
	BSR (2024)	84.6	94.5*	88.3	88.2	49.9	69.2	76.8	74.3
OPS_{img} (Ours)	OPS(10, 5, 5)	91.8	95.1*	91.7	92.7	75.6	80.8	85.4	85.2
	OPS(10, 10, 10)	96.2	97.6*	95.8	97.0	85.4	88.3	92.2	90.6

Table 4. Attack success rates (%) of different attack algorithms on seven normally trained image classifiers, using ResNet-101 as the surrogate model. The best results are highlighted in bold, and * indicates white-box attacks.

Category	Attack	ReNext-50 \implies							
		ResNet-18	ResNet-101	ResNext-50	DenseNet-121	ViT	PiT	Visformer	Swin
Gradient-based	VMI (2021)	61.2	60.3	90.5*	63.4	28.7	41.6	49.2	50.0
	GRA (2022)	80.6	83.1	92.7*	82.7	55.6	65.3	70.7	72.1
	PGN (2023)	86.4	88.4	97.5*	89.0	58.5	72.4	77.1	77.9
	NCS (2024)	81.7	82.8	93.3*	83.4	57.5	67.9	72.9	73.7
Input transformation-based	SIA (2023)	82.9	85.0	96.7*	87.2	43.1	63.5	73.8	72.1
	DeCoWA (2024)	89.7	87.4	96.9*	91.7	58.3	71.7	80.9	77.8
	BSR (2024)	81.0	83.3	96.0*	86.0	39.7	60.0	71.4	69.7
OPS_{img} (Ours)	OPS(10, 5, 5)	91.9	90.5	96.3*	92.8	72.0	80.3	85.0	84.6
	OPS(10, 10, 10)	96.4	95.5	98.6*	96.9	82.7	88.4	91.9	91.5

Table 5. Attack success rates (%) of different attack algorithms on seven normally trained image classifiers, using ReNext-50 as the surrogate model. The best results are highlighted in bold, and * indicates white-box attacks.

Category	Attack	DenseNet-121 \implies							
		ResNet-18	ResNet-101	ResNext-50	DenseNet-121	ViT	PiT	Visformer	Swin
Gradient-based	VMI (2021)	88.4	74.1	75.2	99.9*	37.7	47.1	62.8	61.3
	GRA (2022)	97.3	87.9	89.4	99.9*	52.4	63.9	76.5	79.1
	PGN (2023)	96.5	88.5	90.0	100.0*	54.2	66.2	77.9	80.8
	NCS (2024)	97.2	91.4	91.9	99.9*	62.2	70.7	82.9	84.5
Input transformation-based	SIA (2023)	98.8	91.4	94.4	99.9*	48.3	65.9	83.4	81.5
	DeCoWA (2024)	97.9	88.4	90.3	100.0*	58.3	68.4	82.6	81.0
	BSR (2024)	98.7	89.7	93.3	100.0*	48.8	65.6	82.7	79.4
OPS_{img} (Ours)	OPS(10, 5, 5)	98.7	93.4	94.1	99.9*	65.0	74.5	86.7	85.0
	OPS(10, 10, 10)	99.3	97.1	97.5	100.0*	75.3	83.9	92.8	91.8

Table 6. Attack success rates (%) of different attack algorithms on seven normally trained image classifiers, using DenseNet-121 as the surrogate model. The best results are highlighted in bold, and * indicates white-box attacks.

Category	Attack	ViT \implies							
		ResNet-18	ResNet-101	ResNext-50	DenseNet-121	ViT	PiT	Visformer	Swin
Gradient-based	VMI (2021)	58.9	46.5	50.1	59.7	98.1*	55.6	56.8	67.7
	GRA (2022)	71.3	64.8	65.9	72.2	97.5*	71.4	71.2	78.2
	PGN (2023)	76.3	69.4	69.2	77.7	97.5*	75.4	76.0	81.8
	NCS (2024)	75.3	68.0	69.6	75.9	96.6*	75.3	75.6	82.5
Input transformation-based	SIA (2023)	77.9	73.7	74.7	80.8	97.7*	82.2	81.4	85.9
	DeCoWA (2024)	81.1	72.9	75.8	83.2	91.4*	81.9	81.4	81.4
	BSR (2024)	76.1	70.5	72.2	77.9	94.3*	78.8	76.7	79.9
OPS_{img} (Ours)	OPS(10, 5, 5)	83.7	78.2	80.1	84.9	94.9*	83.9	83.9	85.3
	OPS(10, 10, 10)	90.3	86.8	88.7	91.8	96.9*	91.1	90.6	92.8

Table 7. Attack success rates (%) of different attack algorithms on seven normally trained image classifiers, using ViT as the surrogate model. The best results are highlighted in bold, and * indicates white-box attacks.

Category	Attack	PiT \implies							
		ResNet-18	ResNet-101	ResNext-50	DenseNet-121	ViT	PiT	Visformer	Swin
Gradient-based	VMI (2021)	59.8	50.5	53.2	60.4	45.6	96.0*	61.3	63.4
	GRA (2022)	73.7	68.7	70.4	74.4	67.0	94.4*	77.0	77.4
	PGN (2023)	76.2	70.3	72.2	75.8	68.8	93.7*	76.7	78.7
	NCS (2024)	75.4	71.4	72.2	77.9	69.6	94.1*	77.5	79.3
Input transformation-based	SIA (2023)	80.1	77.3	80.0	83.4	72.4	99.2*	88.2	88.3
	DeCoWA (2024)	83.9	77.1	81.4	88.5	76.5	97.4*	89.1	89.9
	BSR (2024)	78.9	73.2	77.8	81.8	67.1	98.6*	85.2	86.6
OPS_{img} (Ours)	OPS(10, 5, 5)	88.9	85.9	86.2	89.5	85.1	95.8*	91.2	91.8
	OPS(10, 10, 10)	94.3	91.3	91.4	94.4	91.5	98.3*	95.4	95.6

Table 8. Attack success rates (%) of different attack algorithms on seven normally trained image classifiers, using PiT as the surrogate model. The best results are highlighted in bold, and * indicates white-box attacks.

Category	Attack	Visformer \implies							
		ResNet-18	ResNet-101	ResNext-50	DenseNet-121	ViT	PiT	Visformer	Swin
Gradient-based	VMI (2021)	71.2	63.5	66.8	73.5	54.3	70.2	97.1*	76.6
	GRA (2022)	79.2	74.9	77.1	80.5	70.6	79.0	93.3*	82.1
	PGN (2023)	84.1	80.1	81.0	85.2	76.5	82.9	95.7*	85.4
	NCS (2024)	84.2	81.7	82.1	87.1	76.6	83.5	96.6*	87.3
Input transformation-based	SIA (2023)	87.0	83.6	86.1	90.7	68.9	88.5	99.2*	92.7
	DeCoWA (2024)	93.1	86.0	89.6	95.1	75.9	91.0	99.4*	93.6
	BSR (2024)	84.7	79.3	82.0	89.3	60.9	86.6	99.2(89.6
OPS_{img} (Ours)	OPS(10, 5, 5)	95.1	92.6	93.4	95.6	88.3	93.8	98.5*	94.8
	OPS(10, 10, 10)	97.5	96.0	97.0	98.6	93.0	96.5	99.4*	97.8

Table 9. Attack success rates (%) of different attack algorithms on seven normally trained image classifiers, using Visformer as the surrogate model. The best results are highlighted in bold, and * indicates white-box attacks.

Category	Attack	Swin \implies							
		ResNet-18	ResNet-101	ResNext-50	DenseNet-121	ViT	PiT	Visformer	Swin
Gradient-based	VMI (2021)	58.7	45.0	49.4	58.6	45.7	54.0	59.9	97.4*
	GRA (2022)	84.1	72.7	76.1	84.7	75.2	80.1	84.3	99.5*
	PGN (2023)	86.4	75.2	80.2	86.9	77.4	83.8	87.7	99.4*
	NCS (2024)	86.9	80.3	82.1	88.9	81.8	85.5	91.0	99.2*
Input transformation-based	SIA (2023)	80.6	67.8	72.9	81.2	56.9	77.8	84.4	99.0*
	DeCoWA (2024)	91.7	79.5	83.1	92.0	70.3	89.3	91.4	98.2*
	BSR (2024)	82.1	71.7	76.4	84.2	57.6	83.1	86.4	98.2*
OPS_{img} (Ours)	OPS(10, 5, 5)	93.5	88.0	89.7	93.2	85.4	91.7	93.9	98.9*
	OPS(10, 10, 10)	97.5	95.5	96.2	97.6	93.7	96.8	98.5	99.5*

Table 10. Attack success rates (%) of different attack algorithms on seven normally trained image classifiers, using Swin as the surrogate model. The best results are highlighted in bold, and * indicates white-box attacks.

References

- [1] Zhengsu Chen, Lingxi Xie, Jianwei Niu, Xuefeng Liu, Longhui Wei, and Qi Tian. Visformer: The vision-friendly transformer. In *ICCV*, pages 589–598, 2021. 3
- [2] Pointcept Contributors. Pointcept: A codebase for point cloud perception research. <https://github.com/Pointcept/Pointcept>, 2023. 4
- [3] Congyue Deng, Or Litany, Yueqi Duan, Adrien Poulenc, Andrea Tagliasacchi, and Leonidas J Guibas. Vector neurons: A general framework for so (3)-equivariant networks. In *ICCV*, pages 12200–12209, 2021. 3, 4
- [4] 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. 3
- [5] 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. 3
- [6] Eric et al. Accelerated coordinate encoding: Learning to relocalize in minutes using rgb and poses. In *CVPR*, 2023. 2
- [7] Zhijin Ge, Hongying Liu, Wang Xiaosen, Fanhua Shang, and Yuanyuan Liu. Boosting adversarial transferability by achieving flat local maxima. *NeurIPS*, 36:70141–70161, 2023. 4, 5, 6
- [8] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015. 3
- [9] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 7:187–199, 2021. 3, 4
- [10] Abdullah Hamdi, Sara Rojas, Ali Thabet, and Bernard Ghanem. Advpc: Transferable adversarial perturbations on 3d point clouds. In *ECCV*, pages 241–257. Springer, 2020. 3
- [11] 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
- [12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 3
- [13] 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. 3
- [14] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, pages 4700–4708, 2017. 3
- [15] Qidong Huang, Xiaoyi Dong, Dongdong Chen, Hang Zhou, Weiming Zhang, Kui Zhang, Gang Hua, Yueqiang Cheng, and Nenghai Yu. Pointcat: Contrastive adversarial training for robust point cloud recognition. *IEEE TIP*, 33:2183–2196, 2024. 3, 4
- [16] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *NeurIPS*, 31, 2018. 4
- [17] Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E Hopcroft. Nesterov accelerated gradient and scale invariance for adversarial attacks. In *ICLR*, 2020. 3
- [18] Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E Hopcroft. Nesterov accelerated gradient and scale invariance for adversarial attacks. In *ICLR*, 2020. 3
- [19] 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. 4, 5, 6
- [20] Binbin Liu, Jinlai Zhang, and Jihong Zhu. Boosting 3d adversarial attacks with attacking on frequency. *IEEE Access*, 10:50974–50984, 2022. 3
- [21] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *CVPR*, pages 8895–8904, 2019. 4
- [22] 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. 3
- [23] 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
- [24] Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Rethinking network design and local geometry in point cloud: A simple residual mlp framework, 2022. 3, 4
- [25] AAM Muzahid, Wanggen Wan, Ferdous Sohel, Lian Yao 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. 3, 4
- [26] 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. 2, 3, 4
- [27] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *NeurIPS*, 30, 2017. 3, 4
- [28] 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. 3, 4, 5, 6
- [29] Chunlin Qiu, Yiheng Duan, Lingchen Zhao, and Qian Wang. Enhancing adversarial transferability through neighborhood conditional sampling, 2024. 4, 5, 6
- [30] Tzungyu Tsai, Kaichen Yang, Tsung-Yi Ho, and Yier Jin. Robust adversarial objects against deep learning models. In *AAAI*, pages 954–962, 2020. 3
- [31] Kunyu Wang, Xuanran He, Wenxuan Wang, and Xiaosen Wang. Boosting adversarial transferability by block shuffle and rotation. In *CVPR*, pages 24336–24346, 2024. 4, 5, 6
- [32] Xiaosen Wang and Kun He. Enhancing the transferability of adversarial attacks through variance tuning. In *CVPR*, pages 1924–1933, 2021. 3, 4, 5, 6

- [33] Xiaosen Wang, Jiadong Lin, Han Hu, Jingdong Wang, and Kun He. Boosting adversarial transferability through enhanced momentum, 2021. [3](#)
- [34] Xin Wang, Jie Ren, Shuyun Lin, Xiangming Zhu, Yisen Wang, and Quanshi Zhang. A unified approach to interpreting and boosting adversarial transferability. In *ICLR*, 2021. [3](#)
- [35] Xiaosen Wang, Zeliang Zhang, and Jianping Zhang. Structure invariant transformation for better adversarial transferability. In *ICCV*, pages 4607–4619, 2023. [4](#), [5](#), [6](#)
- [36] 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. [3](#), [4](#)
- [37] 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](#)
- [38] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *CVPR*, pages 9621–9630, 2019. [3](#), [4](#)
- [39] 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. [4](#)
- [40] 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. [2](#)
- [41] 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](#)
- [42] Chong Xiang, Charles R Qi, and Bo Li. Generating 3d adversarial point clouds. In *CVPR*, pages 9136–9144, 2019. [3](#)
- [43] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, pages 1492–1500, 2017. [3](#)
- [44] 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. [4](#)
- [45] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *ICCV*, pages 16259–16268, 2021. [3](#), [4](#)
- [46] Hang Zhou, Kejiang Chen, Weiming Zhang, Han Fang, Wenbo Zhou, and Nenghai Yu. Dup-net: Denoiser and up-sampler network for 3d adversarial point clouds defense. In *ICCV*, pages 1961–1970, 2019. [3](#)