Generating Transferable 3D Adversarial Point Cloud via Random Perturbation Factorization

Bangyan He[1, 2], Jian Liu[3], Yiming Li[4], Siyuan Liang[1, 2], Jingzhi Li[1, 2, *], Xiaojun Jia[1, 2, *], Xiaochun Cao[5]

[1]SKLOIS, Institute of Information Engineering, CAS, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]Ant Group, Beijing, China
[4]Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, China
[5]School of Cyber Science and Technology, Shenzhen Campus, Sun Yat-sen University, Shenzhen, China
{hebangyan, liangsiyuan, lijingzhi, jiaxiaojun}@iie.ac.cn, rex.lj@antgroup.com,
li-ym18@mails.tsinghua.edu.cn, caoxiaochun@mail.sysu.edu.cn

Abstract

Recent studies have demonstrated that existing deep neural networks (DNNs) on 3D point clouds are vulnerable to adversarial examples, especially under the white-box settings where the adversaries have access to model parameters. However, adversarial 3D point clouds generated by existing white-box methods have limited transferability across different DNN architectures. They have only minor threats in real-world scenarios under the black-box settings where the adversaries can only query the deployed victim model. In this paper, we revisit the transferability of adversarial 3D point clouds. We observe that an adversarial perturbation can be randomly factorized into two sub-perturbations, which are also likely to be adversarial perturbations. It motivates us to consider the effects of the perturbation and its sub-perturbations simultaneously to increase the transferability for sub-perturbations also contain helpful information. In this paper, we propose a simple yet effective attack method to generate more transferable adversarial 3D point clouds. Specifically, rather than simply optimizing the loss of perturbation alone, we combine it with its random factorization. We conduct experiments on benchmark dataset, verifying our method's effectiveness in increasing transferability while preserving high efficiency. [1]

Introduction

Deep neural networks (DNNs) have been widely adopted in 2D (Chen et al. 2018; He et al. 2021; Li et al. 2021b) and 3D vision tasks (Sung et al. 2017; Tu et al. 2020; Xie et al. 2020a). Since many applications are mission-critical, the security of DNNs is of great significance.

Recent studies revealed that DNNs designed for 3D point clouds (Wang et al. 2020; Rao, Lu, and Zhou 2020; Xu et al. 2021) are also vulnerable to *adversarial examples*, similar to that in 2D scenarios (Bai et al. 2020; Treu et al. 2021; Li et al. 2022). This is probably because they all suffer from the identical drawback that the decision-making processes of humans and DNNs are different. Currently, most existing adversarial attacks (Hamdi et al. 2020; Tsai et al. 2020; Wen et al. 2020) against 3D point clouds are under the white-box settings, where the adversaries have complete access to model source files. As such, they can easily generate adversarial perturbations based on the model gradients. However, in real-world scenarios, most victim models are deployed and the adversaries can only query the model. Accordingly, generating transferable adversarial 3D point clouds generated using one model can also fool others is a more realistic threat and worth further consideration.

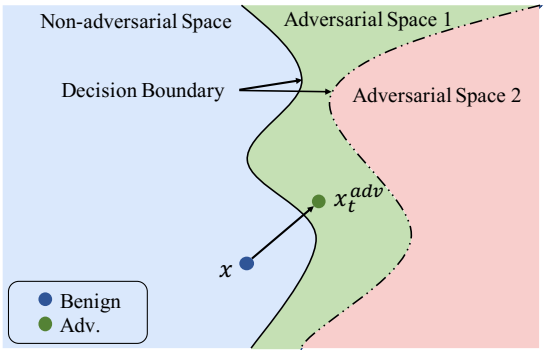
In this paper, we revisit the transferability of adversarial 3D point clouds. Arguably, adversarial examples lying close to the decision boundary are less transferable, for different DNNs may have relatively different boundaries. However, we can hardly regularize the distance between the adversarial example and the decision boundary to generate more transferable examples, for the boundary is complicated and has no analytical form. Meanwhile, we observe that an adversarial perturbation can be randomly factorized into two sub-perturbations, which are also likely to be adversarial perturbations. This phenomenon indicates that the sub-perturbations may also contain helpful information about the decision boundaries and, therefore, can help generate more transferable adversarial examples.

Motivated by this understanding, we propose a simple yet effective attack to increase the transferability of adversarial 3D point clouds. Specifically, we propose to optimize the loss of the perturbation and its sub-perturbations generated by randomized factorization instead of simply optimizing the perturbation alone. As such, the generated samples are further away from the decision boundary and, therefore, more transferable (as shown in Figure 1). Besides, we develop a sampling-based method that only selects one random factorization in each iteration for efficiency. The sequential combination of the effects of different factorizations used in different iterations serves as a 'bagging ensemble', ensuring effectiveness while preserving high efficiency. We hope our work can inspire a deeper understanding of adversarial transferability to help design more robust and secure DNNs.

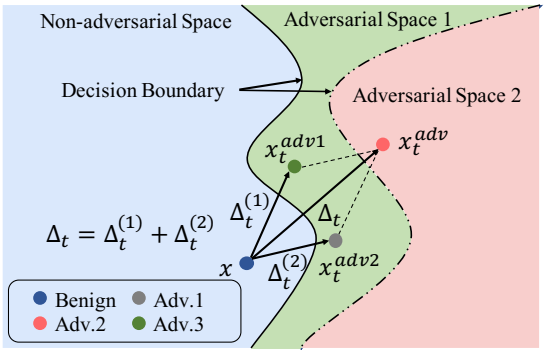
In conclusion, our main contributions are three-fold:

*Correspondence to: Jingzhi Li and Xiaojun Jia.

[1]The code is available on https://github.com/HeBangYan/PF-Attack



(a) Baseline Attack



(b) Ours

Figure 1: The illustration of our method and baseline attacks. Our method requires that the sub-perturbations be adversarial perturbations based on the victim model. We can generate adversarial examples further away from the decision boundary and successfully fool another model, even though the slightly shifted decision surface. In this figure, different colors represent different adversarial spaces and sample points. The solid and dashed lines represent the decision boundaries for the different models. The region to the left of the decision boundary represents the non-adversarial space, and to the right represents the adversarial space.

- We reveal that the sub-perturbations of an adversarial perturbation are also likely to be adversarial perturbations containing useful information.
- We propose a simple yet effective attack method to generate more transferable adversarial 3D point clouds.
- Experiments on the benchmark dataset verify our attack effectiveness and its resistance to potential defenses.

Related Work

DNNs for 3D Point Cloud Classification

Much of the current work has focused on 2D images (Yang et al. 2020; Su et al. 2020; Xie et al. 2020b). It is worth noting that 2D image data is ordered, whereas 3D point cloud data is unordered, which is the most significant difference between them. As a result, existing 2D methods are difficult to use directly for 3D tasks.

Initially, there were several DNN models (Maturana and Scherer 2015; Qi et al. 2016; Masci et al. 2015; Guo, Zou, and Chen 2015) for 3D point cloud data processing, but most suffered from over-computation, application domain limitations, and information loss. To overcome these problems, **PointNet** (Qi et al. 2017a) has been proposed, which is more concise and efficient than previous methods. Specifically, PointNet uses multiple multi-layer perceptrons (MLPs) to extract features for each point cloud data, aligns these features using T-Net, and then obtains global features through max pooling. However, (Qi et al. 2017b) points out that PointNet cannot capture local features. Therefore, **PointNet++** (Qi et al. 2017b) is proposed to improve on this deficiency. Specifically, PointNet++ can be subdivided into two forms: single-scale **(SSG)** and multi-scale **(MSG)**. **DGCNN** (Wang et al. 2019) proposed the EdgeConv module, which can extract local features of point clouds very well while also achieving excellent performance. Since then, several outstanding models (Ran et al. 2021; Goyal et al. 2021; Xiang et al. 2021) have emerged. Also, as researchers continue to investigate the visual transformer, there are now ways to migrate the structure of the visual transformer to point cloud processing tasks. Also, some methods (Guo et al. 2021; Zhao et al. 2021; Engel, Belagiannis, and Dietmayer 2021) that migrate the structure of the vision transformer to the point cloud processing task.

As the aforementioned point-based DNNs are widely used in various mission-critical applications, the study of their security has become increasingly important.

Adversarial Attacks against Point-Based DNNs

The adversarial attack aims to mislead the DNN into making the wrong decision. In 2D domain, existing attack methods can be divided into white-box attack (Gu, Wu, and Tresp 2021) and black-box attack (Andriushchenko et al. 2020; Dong et al. 2018; Jia et al. 2020). Most of the current research has focused on 2D tasks (Wang et al. 2021; Gu et al. 2022; Fan et al. 2020). Due to the differences between 2D and 3D data structures, these methods cannot be directly applied to 3D tasks.

3D-Adv (Xiang, Qi, and Li 2019) is the first work to investigate the production of adversarial point clouds. They divided adversarial attacks on 3D point clouds into two categories: adversarial point perturbation and adversarial point generation. Most of the subsequent research has been conducted around these two categories (Ma et al. 2020; Zhou et al. 2020; Li et al. 2021a). In this paper, only adversarial point perturbation is discussed. **KNN** (Tsai et al. 2020) incorporates KNN distance into the loss term to generate reasonably shaped adversarial point clouds and can mislead 3D point cloud classifiers in physical world. $GeoA^3$ proposes a geometry-aware loss term to enhance the unpredictability of adversarial point clouds to humans. In addition, they propose an iterative normal projection method to improve the smoothness of the generated adversarial point clouds.

All of the above methods are white-box attacks. However, since white-box attacks require knowledge of all model parameters and are rarely used in real scenarios, they are less practical than black-box attacks. **AdvPC** (Hamdi et al.

2020) makes adversarial point clouds easier to transfer by reconstructing the perturbed inputs through point cloud autoencoders. AdvPC improves the transferability of adversarial point clouds compared to previous methods. However, the transferability of existing methods is still limited and deserves further investigation. We take an optimization-based perspective to improve transferability.

Adversarial Defenses towards Point-Based DNNs

There has been some research on CNN (Jia et al. 2022, 2019; Wu et al. 2021), capsule network (Gu, Tresp, and Hu 2021) and Vision Transformer (Gu, Tresp, and Qin 2022; Wu et al. 2022) with regard to their robustness. Due to the different data structure between 3D and 2D, most 2D defense methods cannot be directly used for 3D tasks. Zhou et al (Zhou et al. 2019) proposed two defence methods, that is, statistical outlier removal (SOR) and simple random sampling (SRS). **SOR** determines the k-nearest neighbour of each point in the point cloud and the distances between the neighbouring points. By calculating the mean $\bar{d}$ and standard deviation σ of these distance values, points with distance values greater than $(\bar{d} + \alpha \cdot \sigma)$ must be eliminated, where α is the hyperparameter value. **SRS** removes certain points from the point cloud with a specific probability.

Revisiting Adversarial 3D Point Clouds

In this section, we explore the properties of adversarial 3D point clouds in the high-dimensional input space. In particular, we focus mainly on the adversarial perturbation since it is closely related to the attack success rate and the transferability of adversarial examples.

Specifically, let $\boldsymbol{P} \in \mathbb{R}^{N\times 3}$ denotes the benign 3D point cloud with ground-truth label y. Given a model f_θ, assume that $\boldsymbol{\Delta}$ is the successfully adversarial perturbation of $\boldsymbol{P}$, $i.e.$, $f_\theta(\boldsymbol{P}+\boldsymbol{\Delta}) \neq y$, each perturbation $\boldsymbol{\Delta}$ can be randomly factorized into different sub-perturbations, as follows:

$$\boldsymbol{\Delta} = \boldsymbol{\Gamma} \odot \boldsymbol{\Delta}_l + (\mathbf{1} - \boldsymbol{\Gamma}) \odot \boldsymbol{\Delta}_r, \tag{1}$$

where $\boldsymbol{\Gamma} \in \{0,1\}^{N\times 3}$ is a mask metric, $\boldsymbol{\Delta}_l$ and $\boldsymbol{\Delta}_r$ is the left and right sub-perturbation, respectively. We explore whether these sub-perturbations can also serve as the successfully adversarial perturbation of $\boldsymbol{P}$. If so, they will also contain much useful information.

Settings. We adopt $GeoA^3$ to generate adversarial perturbations with a maximum perturbation size $\epsilon \in \{0.18, 0.45\}$. Specifically, the step size is 0.01, the victim model is PointNet, and the number of iterations is 200. We randomly select 250 samples in the ModelNet40 dataset (Wu et al. 2015) that can be successfully attacked and generate their sub-perturbations $\boldsymbol{\Delta}_l$ and $\boldsymbol{\Delta}_r$ according to equation 1. Afterward, we calculate these sub-perturbations' average attack success rate (A-ASR).

Results. As shown in Table 1, most sub-perturbations can still be used as successful adversarial perturbations when attacking the model, regardless of the value of ϵ. These results suggest that sub-perturbations also carry useful information about the decision boundary and may be used to increase the transferability of the adversarial point cloud.

ϵ	0.18		0.45	
Adversarial Example	$\boldsymbol{P}+\boldsymbol{\Delta_l}$	$\boldsymbol{P}+\boldsymbol{\Delta_r}$	$\boldsymbol{P}+\boldsymbol{\Delta_l}$	$\boldsymbol{P}+\boldsymbol{\Delta_r}$
A-ASR	84.80	84.00	91.20	92.80

Table 1: The averaged attack success rate (A-ASR, %) of adversarial examples with sub-perturbations generated by randomized perturbation factorization.

The Proposed Method

Motivated by the above understanding, we propose to use information from sub-perturbations to improve the transferability of the generated adversarial 3D point clouds. Intuitively, the transferability of adversarial examples close to the decision boundary is lower, as different DNNs may have relatively different boundaries. If most of the sub-perturbations of an adversarial perturbation are also adversarial, then this perturbation is more likely to be far from the decision boundary and, therefore, more transferable. Accordingly, in this paper, we propose to optimize both the perturbation and its sub-perturbations. More technical details are given in the following subsections.

Threat Model

In this paper, we focus on the untargeted adversarial attack against DNNs designed for 3D point cloud classifiers. Specifically, the adversaries intend to generate transferable adversarial 3D point clouds based on a local victim model under the white-box settings. The generated adversarial examples should be able to fool other victim models during the inference process. This attack could happen in real-world scenarios where the targeted model is deployed, and the adversaries can only query the model.

Randomized Perturbation Factorization

Generate Perturbations Further Away from the Decision Surface. Intuitively, if the model's output is more similar to a one-hot vector, the further the sample is from the decision boundary. In this paper, we propose to optimize the victim DNN according to its predicted probability. Specifically, let y denotes the ground-truth label of 3D point cloud $\boldsymbol{P}$ and $g(\boldsymbol{P})_y \in \mathbb{R}$ indicate the predicted probability of the y-th class predicted by the victim DNN. $v\left(\boldsymbol{P}, \boldsymbol{\Delta}\right)$ measures the gap between the predicted probability of the ground-truth class and the predicted probability of the top-1 but non-ground-truth class, which can be calculated as follow.

$$v\left(\boldsymbol{P}, \boldsymbol{\Delta}\right) = \left(\max_{y'\neq y} g(\boldsymbol{P}+\boldsymbol{\Delta})_{y'} - g(\boldsymbol{P}+\boldsymbol{\Delta})_y\right)^2. \tag{2}$$

Attack with Randomized Perturbation Factorization. Given a mask matrix $\Gamma \in \mathbb{R}^{N\times 3}$, the probability of an element in Γ having a value of 1 is p and the probability of a value of 0 is $1-p$. Then, the set $\mathcal{S}$ of matrices consisting of all possible Γ can be expressed as:

$$\mathcal{S} = \{\Gamma_{it} | i \in \{1, 2, \ldots, N\}, t \in \{1, 2, 3\}\}.$$

The perturbation factorization function is defined as follows:

$$V\left(\boldsymbol{P}, \boldsymbol{\Delta}\right) = \mathop{\mathbb{E}}_{\Gamma'\sim\mathcal{S}}[v\left(\boldsymbol{P}, \Gamma' \odot \boldsymbol{\Delta}\right) + v\left(\boldsymbol{P}, (1-\Gamma') \odot \boldsymbol{\Delta}\right)]. \tag{3}$$

Algorithm 1: The Main Process of Our PF-Attack.

Input: Benign point cloud $\boldsymbol{P}$, Unit normal vector of benign point cloud $\boldsymbol{n_P}$, number of iteration steps T, step size η, coefficient of ℓ_∞ norm ϵ;

Initialize Perturbation $\boldsymbol{\Delta}_0 \sim \mathcal{N}(0, 0.001)$;
$\boldsymbol{P}'_0 \leftarrow \boldsymbol{P} + \boldsymbol{\Delta}_0$;
$t \leftarrow 0$;
for $t \leq T$ **do**
  $\boldsymbol{\Delta}_{t+1} \leftarrow \boldsymbol{\Delta}_t - \eta \cdot \nabla \ell_{reg}\left(\boldsymbol{P}'_t, \boldsymbol{P}, \boldsymbol{\Delta}_t\right)$;
  $\boldsymbol{\Delta}'_{t+1} \leftarrow \langle \boldsymbol{\Delta}_{t+1}, \boldsymbol{n_P} \rangle \cdot \boldsymbol{n_P}$;
  if $\left|\boldsymbol{\Delta}'_{t+1}\right| \leq \epsilon$ **then**
    $\boldsymbol{P}'_{t+1} \leftarrow \boldsymbol{P} + \boldsymbol{\Delta}'_{t+1}$;
  else
    $\boldsymbol{P}'_{t+1} \leftarrow \boldsymbol{P} + \frac{\boldsymbol{\Delta}'_{t+1}}{\left\|\boldsymbol{\Delta}'_{t+1}\right\|_2} \cdot \epsilon$;
  end if
  $t \leftarrow t + 1$;
end for

Output: $\boldsymbol{P}'_{t+1}$

For equation 3, the attacker needs to sample all possible Γ', which is computation-consuming. Instead, we propose a sampling-based approach that performs multiple iterations, sampling a pair of randomly factorized sub-perturbations each time the perturbation is updated.

Overall Objective Function. We aim to produce adversarial point clouds $\boldsymbol{P}'$ for f to misclassify them. The function we are trying to minimize is as follows:

$$\min_{\boldsymbol{\Delta}} \ell_{reg} = -\ell_{cls}\left(f\left(\boldsymbol{P}'\right), y_{true}\right) + \tau \cdot \ell_{PF}\left(\boldsymbol{P}', \boldsymbol{P}, \boldsymbol{\Delta}\right) \quad \text{s.t.} \left\|\boldsymbol{\Delta}\right\|_\infty \leq \epsilon, \tag{4}$$

where ℓ_{cls} denotes cross-entropy loss function, ϵ denotes coefficient of ℓ_∞ norm and τ is a penalty parameter. The full objective of ℓ_{PF} can be expressed as

$$\ell_{PF}\left(\boldsymbol{P}', \boldsymbol{P}, \boldsymbol{\Delta}\right) = \ell_{CD}\left(\boldsymbol{P}', \boldsymbol{P}\right) + \beta \cdot \ell_{pf}\left(\boldsymbol{P}, \boldsymbol{\Delta}\right), \tag{5}$$

where β is a hyper-parameter to balance the different constraints, $\ell_{CD}\left(\boldsymbol{P}', \boldsymbol{P}\right)$ denotes Chamfer loss and the expression for $\ell_{pf}\left(\boldsymbol{\Delta}, \boldsymbol{P}\right)$ is shown below.

$$\ell_{pf} = v\left(\boldsymbol{P}, \Gamma' \odot \boldsymbol{\Delta}\right) + v\left(\boldsymbol{P}, (1 - \Gamma') \odot \boldsymbol{\Delta}\right) + v\left(\boldsymbol{P}, \boldsymbol{\Delta}\right). \tag{6}$$

For intuitive understanding, we show in Algorithm 1 how our PF-Attack method generates an adversarial point cloud. Before iteration, we generate an initial perturbation with a mean of 0 and variance of 0.001 that conforms to a normal distribution, *i.e.*, $\boldsymbol{\Delta}_0 \sim N\left(0, 0.001\right)$. Next, we update the perturbation using gradient descent and use the projection method to make the adversarial point cloud $\boldsymbol{P}'_{t+1}$ smoother. Then, we limit the perturbation size using ℓ_∞ norm.

Figure 2 shows some of the adversarial point clouds generated by our method.

Experiments

Experimental Setup

Dataset. We used ModelNet40 (Wu et al. 2015), a widely used dataset, to train the model and evaluate the performance of each attack method. ModelNet40 has a total of 12,311 CAD models containing 40 different object categories, of which 9,843 samples were used for training and 2,468 for testing. Following prior works (Xiang, Qi, and Li 2019; Wen et al. 2020; Hamdi et al. 2020), 25 samples were randomly extracted from each of the ten object categories of ModelNet40, using a total of 250 samples used to generate the adversarial point cloud, each of which contained 1,024 points.

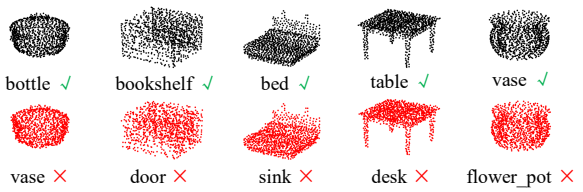


Figure 2: Visualization. Benign point clouds are in the top row (shown in black) and adversarial point clouds are in the bottom row (shown in red).

Models. Following prior works (Xiang, Qi, and Li 2019; Wen et al. 2020; Hamdi et al. 2020), the models we attack are PointNet (Qi et al. 2017a), PointNet++(SSG) (Qi et al. 2017b), PointNet++(MSG) (Qi et al. 2017b) and DGCNN (Wang et al. 2019). Each model was trained normally.

Attack Settings. We use 3D-Adv (Xiang, Qi, and Li 2019), KNN (Tsai et al. 2020), $GeoA^3$ (Wen et al. 2020), AdvPC (Hamdi et al. 2020) as the baselines for our experiments. The hyper-parameters of the PF-Attack were set to: $\eta = 0.01, \tau = 10, \beta = 0.5, p = 0.5, T = 200, \epsilon \in \{0.18, 0.45\}$. We use Adam optimizer (Kingma and Ba 2015). 3D-Adv, KNN, AdvPC initialize the perturbation twice. Our method and $GeoA^3$ initialize the perturbation once.

Evaluation Metric

First, we give two definitions: ❶ A *victim model* is a model used to generate adversarial point clouds; ❷ A *transfer model* is a model that is attacked by adversarial point clouds, but the transfer model is not used to generate adversarial point clouds. Then, we will evaluate our experiments using the following two metrics.

Attack Success Rate (ASR). Given N adversarial point clouds $x_i, i \in \{1, 2, \ldots, N\}$, their ground-truth labels are $y_i^{true} i \in \{1, 2, \ldots, N\}$ respectively. For intuitively understanding, ASR can be expressed as follows.

$$\text{ASR} = \frac{1}{N} \cdot \left(\sum_{i=1}^{N} T\left(f\left(x_i\right)\right)\right), \tag{7}$$

where

$$T\left(f\left(x_i\right)\right) = \begin{cases} 0, & \text{if} f\left(x_i\right) = y_i^{true} \\ 1, & \text{if} f\left(x_i\right) \neq y_i^{true} \end{cases}.$$

Notably, the higher the attack success rate on the transfer model, the better the adversarial point cloud's transferability.

Transferability Score. To evaluate the overall transferability, following (Hamdi et al. 2020), we calculated the average ASRs of the adversarial point clouds on all transfer models and used this as the transferability score.

Victim Network	Attack Method	$\epsilon = 0.18$				$\epsilon = 0.45$			
		PointNet	PointNet++ (MSG)	PointNet++ (SSG)	DGCNN	PointNet	PointNet++ (MSG)	PointNet++ (SSG)	DGCNN
PointNet	3D-Adv	100*	8.4	10.4	6.8	100*	8.8	9.6	8.0
	KNN	100*	9.6	10.8	6.0	100*	9.6	8.4	6.4
	$GeoA^3$	100*	20.0	19.6	7.2	100*	23.6	20.8	7.2
	AdvPC	98.8*	20.4	27.6	22.4	98.8*	18.0	26.8	20.4
	PF-Attack	100*	**49.6**	**61.5**	**24.8**	100*	**59.3**	**64.8**	**28.8**
PointNet++ (MSG)	3D-Adv	6.8	100*	28.4	11.2	7.2	100*	29.2	11.2
	KNN	6.4	100*	22.0	8.8	6.4	100*	23.2	7.6
	$GeoA^3$	4.4	100*	14.4	6.4	4.4	100*	13.6	6.0
	AdvPC	13.2	97.2*	54.8	**39.6**	18.4	98*	58.0	**39.2**
	PF-Attack	**17.2**	100*	**67.0**	24.3	**19.2**	100*	**75.0**	27.5
PointNet++ (SSG)	3D-Adv	7.6	9.6	100*	6.0	7.2	10.4	100*	7.2
	KNN	6.4	9.2	100*	6.4	6.8	7.6	100*	6.0
	$GeoA^3$	5.2	10.4	100*	2.2	4.8	9.2	100*	4.0
	AdvPC	12.0	27.2	99.2*	**22.8**	14.0	30.8	99.2*	**27.6**
	PF-Attack	**13.9**	**47.4**	100*	19.6	**15.7**	**56.9**	100*	23.0
DGCNN	3D-Adv	9.2	11.2	31.2	100*	9.6	12.8	30.4	100*
	KNN	7.2	9.6	14.0	99.6*	6.8	10.0	11.2	99.6*
	$GeoA^3$	4.4	27.2	27.6	100*	4.4	26.8	25.6	100*
	AdvPC	19.6	46.0	64.4	94.8*	**32.8**	48.8	64.4	97.2*
	PF-Attack	**21.3**	**60.9**	**74.8**	100*	26.1	**79.7**	**85.8**	100*

Table 2: Transfer Attack on ModelNet40. Measure performance in terms of attack success rate (%). The results of 3D-Adv, KNN and AdvPC are reported in (Hamdi et al. 2020). Number in bold indicates the best. * indicates the white-box model.

$\epsilon \downarrow$	Transferability Score (%)				
	3D-Adv	KNN	$GeoA^3$	AdvPC	PF-Attack
0.18	12.2	9.7	12.4	30.8	**40.2**
0.45	12.6	9.2	12.5	33.3	**46.8**

Table 3: Transferability Score on ModelNet40.

Attack Method $\downarrow$	$\epsilon = 0.18$	$\epsilon = 0.45$
3D-Adv	7.5	7.47
GeoA3	21.6	23.6
PF-Attack	51.28	59.32

Table 4: Transfer Attack on Point Transformer. Measure performance in terms of attack success rate (%). Adversarial point clouds are generated on PointNet.

Attack Transferability

We conducted a series of comparative experiments to compare the transferability of the adversarial point clouds generated by our method and baselines. The results are shown in Table 2. In most cases, the adversarial point cloud generated by our method achieves better transferability, $i.e.$, higher ASRs on the transfer models. As shown in Table 3, the overall metastability of our method outperforms the baseline. In addition, we used the adversarial point cloud generated on PointNet to attack the Point Transformer. as shown in Table 4, our method achieves better attack success rates.

Discussion

Ablation Studies

To investigate the effect of different loss terms and hyperparameters on the transferability of adversarial point clouds, we conducted a series of ablation studies on loss terms, regularization coefficient β, and generation probability p.

Loss Term. We begin by conducting ablation studies by deleting or replacing a single term in equation 5 and then compare the attack success rates. In (Xiang, Qi, and Li 2019) and (Wen et al. 2020), they only consider the non-symmetric Hausdorff distance, and we do the same. Table 5 demonstrates that, L_{PF}, which containing perturbation factorization method, plays a much larger role in transferability.

Regularization Coefficient β. We then study the impact of the value of regularization coefficient β on the attack success rates, and the β is selected in $\{0.1, 0.3, 0.5, 0.7, 0.9\}$. Figure 3 shows the results. When both ϵ and β are fixed, among the three models of PointNet++ (SSG), PointNet++ (MSG), and DGCNN, the attack success rate of the adversarial point cloud generated by our method on PointNet is the highest on PointNet++ (SSG), PointNet++ (MSG) comes next, and DGCNN is the lowest. The β value can be likened to the step size in the PGD method and we restrict it to change between 0 and 1, so it is reasonable that the β value has little effect on the transferability of the adversarial point cloud. As shown in Figure 3, $\beta = 0.5$ is a compromise value, so we chose it in our comparison experiment.

Adjustment of Loss Term	Attack Success Rate(%)							
	$\epsilon = 0.18$				$\epsilon = 0.45$			
	PointNet	PointNet++ (MSG)	PointNet++ (SSG)	DGCNN	PointNet	PointNet++ (MSG)	PointNet++ (SSG)	DGCNN
with all two loss terms	**100***	**43.59**	**58.6**	**25.64**	**100***	**55.93**	**63.14**	**30.08**
L_{CD} replaced by L_{HD}	100*	36.17	52.8	23.83	100*	45.96	57.02	24.68
w/o L_{pf}	100*	10.92	12.6	3.78	100*	13.08	13.5	4.22

Table 5: Ablation studies of replacing or deleting single loss term. Measure performance in terms of attack success rate (%). The attack success rate is reported by attacking other models using the replaced or deleted single loss term. The victim model is PointNet. Number in bold indicates the best. * indicates the white-box model.

Attack Method	Attack Success Rate (%)							
	$\epsilon = 0.18$				$\epsilon = 0.45$			
	PointNet	PointNet++ (MSG)	PointNet++ (SSG)	DGCNN	PointNet	PointNet++ (MSG)	PointNet++ (SSG)	DGCNN
3D-Adv	100*	8.4	10.4	6.8	100*	8.8	9.6	8.0
PF-3D-Adv	100*	17.0	20.9	6.5	100*	20.2	21.0	8.5
KNN	100*	9.6	10.8	6.0	100*	9.6	8.4	6.4
PF-KNN	100*	46.2	61.1	27.8	100*	60.0	68.9	28.9
$GeoA^3$	100*	20.0	19.6	7.2	100*	23.6	20.8	7.2
PF-$GeoA^3$	100*	37.9	47.4	24.1	100*	45.7	54.3	25.2

Table 6: Ablation Studies of Incorporating Existing Methods. Measure performance in terms of attack success rate (%). The victim model is PointNet. * indicates the white-box model. The results of 3D-Adv, KNN are reported in (Hamdi et al. 2020).

$\epsilon_\infty = 0.18$			$\epsilon_\infty = 0.45$		
No Defense	SRS	SOR	No Defense	SRS	SOR
100.00	51.28	38.03	100.00	56.78	39.41

Table 7: Resistance to potential defense methods. Measure performance in terms of attack success rate (%). The victim model is PointNet. In SRS, 128 points were randomly removed. In SOR, the value of α is 1.1, and k is 2.

Generation Probability p**.** Finally, we study the impact of the value of generation probability p on the success rates, and the p are selected in $\{0, 0.1, 0.3, 0.5, 0.7, 0.9, 1\}$. Figure 4 shows the result. When the generation probability p equals 0 or 1, the equation 6 does not consider the perturbation factorization. Therefore, when the value of p is 0 or 1, the transferability of the generated adversarial point cloud will be worse than when the value of p is 0.1, 0.3, 0.5, 0.7, or 0.9. When ϵ is equal to 0.18 and β is fixed, among the three models of PointNet++ (SSG) (Qi et al. 2017b), PointNet++ (MSG) (Qi et al. 2017b), and DGCNN (Wang et al. 2019), the attack success rate of the adversarial point cloud generated by our method on PointNet is the highest on PointNet++ (SSG), PointNet++ (MSG) comes next, and DGCNN is the lowest. When ϵ is equal to 0.45 and p is equal to 0.9, the adversarial point cloud generated by our method on PointNet has a higher attack success rate in PointNet++ (MSG) than PointNet++ (SSG). We choose $p = 0.5$ as the hyperparameter value because the transferability of the adversarial point cloud generated when $p = 0.5$ is worse than that of 0.1, 0.3, 0.7, 0.9. Even in this case, our method performance is still better than baselines.

	$\epsilon = 0.18$			$\epsilon = 0.45$		
K	2	3	4	2	3	4
ASR	61.5	51.1	50.2	64.8	63.4	60.9

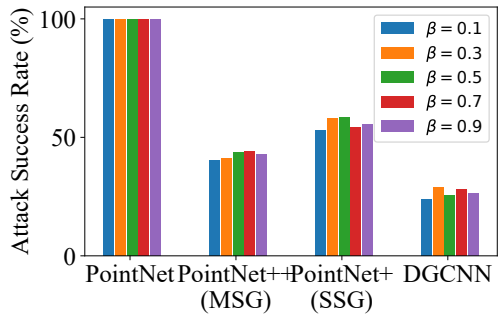
Table 8: Extending into K sub-perturbations. Measure performance in terms of attack success rate (%). The victim model is PointNet. The transfer model is PointNet++(MSG).

Attack Method	3D-Adv	KNN	$GeoA^3$	PF-Attack
Running Time	0.18	0.16	0.24	0.3

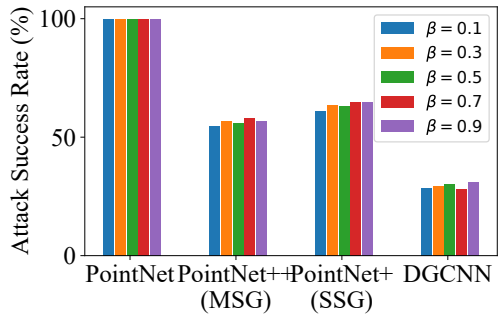
Table 9: Time Comparison. The victim model is PointNet. The number of iterations is 200. Running Time is the average time (in minutes) taken to attack each point cloud.

Incorporation with Existing Attacks

Following (Xie et al. 2019; Wang and He 2021; Dong et al. 2018), we not only pay attention to the transferability of

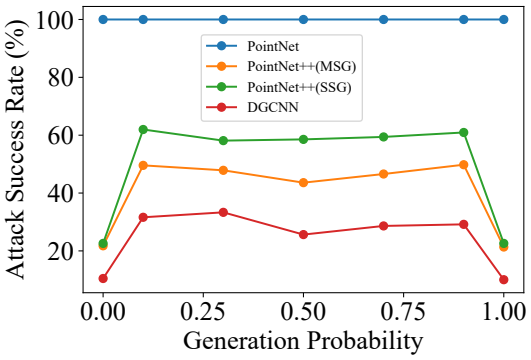


(a) $\epsilon = 0.18$

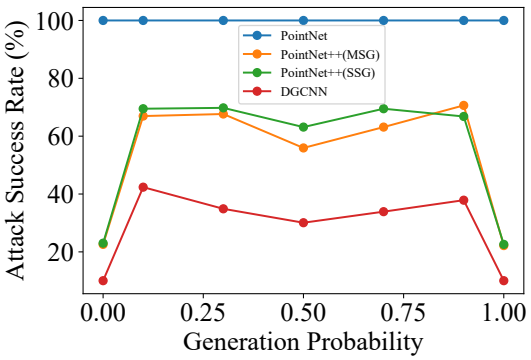


(b) $\epsilon = 0.45$

Figure 3: Ablation studies in β. Measure performance in terms of attack success rate (%). The value of p is fixed at 0.5. The adversarial point clouds are generated on PointNet.



(a) $\epsilon = 0.18$



(b) $\epsilon = 0.45$

Figure 4: Ablation studies in p. Measure performance in terms of attack success rate (%). The value of β is fixed at 0.5. The adversarial point clouds are generated on PointNet.

the method itself but also pay attention to the attack transferability of our method incorporated with other methods. We have used 3D-Adv, KNN, and $GeoA^3$ to incorporate our PF-Attack, which we called PF-3D-Adv, PF-KNN, and PF-$GeoA^3$, respectively. The results are shown in Table 6. It is not difficult to find that the transferability of the adversaial point clouds that generated by original attack methods have been improved to a certain extent after incorporating with our method. It is worth mentioning that incorporated method fails mainly because the original method failed, and these adversarial examples may be located near the decision surface (of the victim model), which may be different from the decision surface of the transfer model.

Resistance to Potential Defenses

To investigate the resistance of our method to potential defenses, we used two methods, SOR (Zhou et al. 2019) and SRS (Zhou et al. 2019), with the victim model being PointNet. Table 7 shows the result. By observation, our method maintained some ASR after SOR and SRS defense, suggesting that they were both resistant to the defence mechanism.

Extension to K Sub-perturbations

Intuitively, we can extend our random factorization to the case with more sub-perturbations by introducing more random variables. However, as Table 8 shows, the extension doesn't necessarily lead to better performance. This failure is may be because the sub-perturbations are less likely to be adversarial ones as K increases. We will explore how to find the best K and its theoretical foundations in our future work.

Limitations

Firstly, the ASRs under transfer models are still lower than 100%, although our method has already significantly improved. Secondly, as shown in Table 9, our method brings additional computational and memory costs since we introduce additional variables. These costs are negligible compared to our performance improvement.

Conclusion

In this paper, we proposed a simple yet effective method to generate more transferable adversarial 3D point clouds. Specifically, we proposed to jointly optimize the perturbation and its sub-ones, motivated by the understanding that sub-perturbations also contain useful information about the decision boundary. Our conducted experiments verified that our method can generate more transferable examples compared with all baseline attacks. Our method can be incorporated with existing attacks to increase their transferability.

Acknowledgements

Supported by the National Key R&D Program of China under Grant 2019YFB1406500, National Natural Science Foundation of China (No. 62025604, 62132006), Beijing Natural Science Foundation (No. M22006).

References

Andriushchenko, M.; Croce, F.; Flammarion, N.; and Hein, M. 2020. Square attack: a query-efficient black-box adversarial attack via random search. In *ECCV*.

Bai, J.; Chen, B.; Li, Y.; Wu, D.; Guo, W.; Xia, S.-t.; and Yang, E.-h. 2020. Targeted attack for deep hashing based retrieval. In *ECCV*.

Chen, L.; Papandreou, G.; Kokkinos, I.; Murphy, K.; and Yuille, A. L. 2018. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.*

Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; and Li, J. 2018. Boosting Adversarial Attacks With Momentum. In *CVPR*.

Engel, N.; Belagiannis, V.; and Dietmayer, K. 2021. Point transformer. *IEEE Access*.

Fan, Y.; Wu, B.; Li, T.; Zhang, Y.; Li, M.; Li, Z.; and Yang, Y. 2020. Sparse Adversarial Attack via Perturbation Factorization. In Vedaldi, A.; Bischof, H.; Brox, T.; and Frahm, J., eds., *ECCV*.

Goyal, A.; Law, H.; Liu, B.; Newell, A.; and Deng, J. 2021. Revisiting Point Cloud Shape Classification with a Simple and Effective Baseline. In Meila, M.; and Zhang, T., eds., *ICML*.

Gu, J.; Tresp, V.; and Hu, H. 2021. Capsule network is not more robust than convolutional network. In *CVPR*.

Gu, J.; Tresp, V.; and Qin, Y. 2022. Are vision transformers robust to patch perturbations? In *ECCV*.

Gu, J.; Wu, B.; and Tresp, V. 2021. Effective and efficient vote attack on capsule networks. *ICLR*.

Gu, J.; Zhao, H.; Tresp, V.; and Torr, P. H. 2022. SegPGD: An Effective and Efficient Adversarial Attack for Evaluating and Boosting Segmentation Robustness. In *ECCV*.

Guo, K.; Zou, D.; and Chen, X. 2015. 3D Mesh Labeling via Deep Convolutional Neural Networks. *ACM Trans. Graph.*

Guo, M.-H.; Cai, J.-X.; Liu, Z.-N.; Mu, T.-J.; Martin, R. R.; and Hu, S.-M. 2021. Pct: Point cloud transformer. *Computational Visual Media*.

Hamdi, A.; Rojas, S.; Thabet, A.; and Ghanem, B. 2020. Advpc: Transferable adversarial perturbations on 3d point clouds. In *ECCV*.

He, J.; Erfani, S.; Ma, X.; Bailey, J.; Chi, Y.; and Hua, X.-S. 2021. alpha-IoU: A Family of Power Intersection over Union Losses for Bounding Box Regression. In *NeurIPS*.

Jia, X.; Wei, X.; Cao, X.; and Foroosh, H. 2019. Comdefend: An efficient image compression model to defend adversarial examples. In *CVPR*.

Jia, X.; Wei, X.; Cao, X.; and Han, X. 2020. Adv-watermark: A novel watermark perturbation for adversarial examples. In *ACM MM*.

Jia, X.; Zhang, Y.; Wu, B.; Ma, K.; Wang, J.; and Cao, X. 2022. LAS-AT: Adversarial Training with Learnable Attack Strategy. In *CVPR*.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In Bengio, Y.; and LeCun, Y., eds., *ICLR*.

Li, X.; Chen, Z.; Zhao, Y.; Tong, Z.; Zhao, Y.; Lim, A.; and Zhou, J. T. 2021a. PointBA: Towards Backdoor Attacks in 3D Point Cloud. In *CVPR*.

Li, Y.; Wu, B.; Feng, Y.; Fan, Y.; Jiang, Y.; Li, Z.; and Xia, S.-T. 2022. Semi-supervised robust training with generalized perturbed neighborhood. *Pattern Recognition*, 124: 108472.

Li, Y.; Zhang, S.; Wang, Z.; Yang, S.; Yang, W.; Xia, S.-T.; and Zhou, E. 2021b. Tokenpose: Learning keypoint tokens for human pose estimation. In *ICCV*.

Ma, C.; Meng, W.; Wu, B.; Xu, S.; and Zhang, X. 2020. Efficient Joint Gradient Based Attack Against SOR Defense for 3D Point Cloud Classification. In Chen, C. W.; Cucchiara, R.; Hua, X.; Qi, G.; Ricci, E.; Zhang, Z.; and Zimmermann, R., eds., *MM '20: The 28th ACM International Conference on Multimedia, Virtual Event / Seattle, WA, USA, October 12-16, 2020*.

Masci, J.; Boscaini, D.; Bronstein, M. M.; and Vandergheynst, P. 2015. Geodesic Convolutional Neural Networks on Riemannian Manifolds. In *ICCV*.

Maturana, D.; and Scherer, S. A. 2015. VoxNet: A 3D Convolutional Neural Network for real-time object recognition. In *IROS*.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *CVPR*.

Qi, C. R.; Su, H.; Nießner, M.; Dai, A.; Yan, M.; and Guibas, L. J. 2016. Volumetric and Multi-view CNNs for Object Classification on 3D Data. In *CVPR*.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In Guyon, I.; von Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *NeurIPS*.

Ran, H.; Zhuo, W.; Liu, J.; and Lu, L. 2021. Learning inner-group relations on point clouds. In *ICCV*, 15477–15487.

Rao, Y.; Lu, J.; and Zhou, J. 2020. Global-local bidirectional reasoning for unsupervised representation learning of 3d point clouds. In *CVPR*.

Su, Z.; Fang, L.; Kang, W.; Hu, D.; Pietikäinen, M.; and Liu, L. 2020. Dynamic group convolution for accelerating convolutional neural networks. In *ECCV*.

Sung, M.; Su, H.; Kim, V. G.; Chaudhuri, S.; and Guibas, L. J. 2017. Complementme: weakly-supervised component suggestions for 3D modeling. *ACM Trans. Graph.*

Treu, M.; Le, T.-N.; Nguyen, H. H.; Yamagishi, J.; and Echizen, I. 2021. Fashion-guided adversarial attack on person segmentation. In *CVPR*.

Tsai, T.; Yang, K.; Ho, T.; and Jin, Y. 2020. Robust Adversarial Objects against Deep Learning Models. In *AAAI*.

Tu, J.; Ren, M.; Manivasagam, S.; Liang, M.; Yang, B.; Du, R.; Cheng, F.; and Urtasun, R. 2020. Physically Realizable Adversarial Examples for LiDAR Object Detection. In *CVPR*.

Wang, X.; and He, K. 2021. Enhancing the Transferability of Adversarial Attacks Through Variance Tuning. In *CVPR*.

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2019. Dynamic Graph CNN for Learning on Point Clouds. *ACM Trans. Graph.*

Wang, Y.; Tan, D. J.; Navab, N.; and Tombari, F. 2020. Softpoolnet: Shape descriptor for point cloud completion and classification. In *ECCV*.

Wang, Z.; Guo, H.; Zhang, Z.; Liu, W.; Qin, Z.; and Ren, K. 2021. Feature importance-aware transferable adversarial attacks. In *ICCV*.

Wen, Y.; Lin, J.; Chen, K.; Chen, C. L. P.; and Jia, K. 2020. Geometry-Aware Generation of Adversarial Point Clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wu, B.; Gu, J.; Li, Z.; Cai, D.; He, X.; and Liu, W. 2022. Towards efficient adversarial training on vision transformers. In *ECCV*.

Wu, B.; Pan, H.; Shen, L.; Gu, J.; Zhao, S.; Li, Z.; Cai, D.; He, X.; and Liu, W. 2021. Attacking adversarial attacks as a defense. *arXiv preprint arXiv:2106.04938*.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3D ShapeNets: A deep representation for volumetric shapes. In *CVPR*.

Xiang, C.; Qi, C. R.; and Li, B. 2019. Generating 3D Adversarial Point Clouds. In *CVPR*.

Xiang, T.; Zhang, C.; Song, Y.; Yu, J.; and Cai, W. 2021. Walk in the cloud: Learning curves for point clouds shape analysis. In *ICCV*.

Xie, C.; Zhang, Z.; Zhou, Y.; Bai, S.; Wang, J.; Ren, Z.; and Yuille, A. L. 2019. Improving Transferability of Adversarial Examples With Input Diversity. In *CVPR*.

Xie, S.; Gu, J.; Guo, D.; Qi, C. R.; Guibas, L.; and Litany, O. 2020a. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *ECCV*.

Xie, Z.; Zhang, Z.; Zhu, X.; Huang, G.; and Lin, S. 2020b. Spatially adaptive inference with stochastic feature sampling and interpolation. In *ECCV*.

Xu, M.; Ding, R.; Zhao, H.; and Qi, X. 2021. PAConv: Position Adaptive Convolution With Dynamic Kernel Assembling on Point Clouds. In *CVPR*.

Yang, T.; Zhu, S.; Chen, C.; Yan, S.; Zhang, M.; and Willis, A. 2020. Mutualnet: Adaptive convnet via mutual learning from network width and resolution. In *ECCV*.

Zhao, H.; Jiang, L.; Jia, J.; Torr, P. H.; and Koltun, V. 2021. Point transformer. In *ICCV*.

Zhou, H.; Chen, D.; Liao, J.; Chen, K.; Dong, X.; Liu, K.; Zhang, W.; Hua, G.; and Yu, N. 2020. Lg-gan: Label guided adversarial network for flexible targeted attack of point cloud based deep networks. In *CVPR*.

Zhou, H.; Chen, K.; Zhang, W.; Fang, H.; Zhou, W.; and Yu, N. 2019. DUP-Net: Denoiser and Upsampler Network for 3D Adversarial Point Clouds Defense. In *ICCV*.