Adversarially Robust 3D Point Cloud Recognition Using Self-Supervisions

Anonymous Author(s)

Abstract

3D point cloud data is increasingly used in safety-critical applications such as autonomous driving. Thus, robustness of 3D deep learning models against adversarial attacks is a major consideration. In this paper, we systematically study the impact of various self-supervised learning proxy tasks on different architectures and threat models for 3D point clouds. Specifically, we study MLP-based (PointNet), convolution-based (DGCNN), and transformer-based (PCT) 3D architectures. Through comprehensive experiments, we demonstrate that appropriate self-supervisions can significantly enhance the robustness in 3D point cloud recognition, achieving considerable improvements compared to the standard adversarial training baseline. Our analysis reveals that *local* feature learning is desirable for adversarial robustness since it limits the adversarial propagation between the point-level input perturbations and the model's final output. It also explains the success of DGCNN and the jigsaw proxy task in achieving 3D robustness.

1 Introduction

Point cloud data is one of the most widely used representations in 3D computer vision. It is a versatile data format available from various sensors and computer-aided design (CAD) models. Given such advantages, many deep learning-based 3D perception systems have been proposed [1–6] and achieved great success in safety-critical applications (*e.g.,* autonomous driving) [7–9]. Although deep learning on point clouds has exhibited high performance, they are particularly vulnerable to adversarial attacks [10–12]. Because of the wide applications in safety-critical fields, it is imperative to study the adversarial robustness of point cloud recognition models.

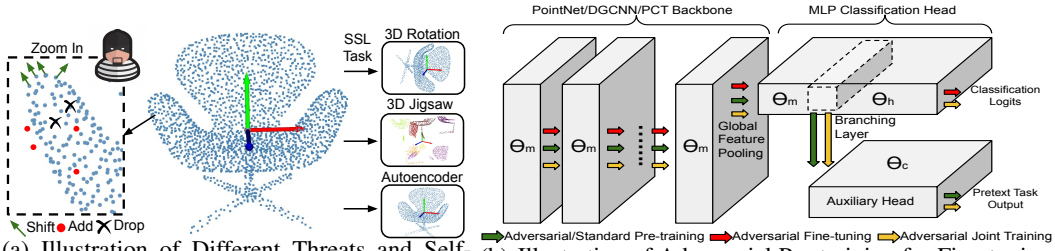
However, there are at present two obstacles on the path reaching robust point cloud recognition:

Architectural Diversity. Deep 3D point cloud recognition is an emerging field. Many 3D architectures with various feature aggregation methods have been proposed. We broadly categorize them into three families based on how networks aggregate the geometric information: multi-layer-perceptron (MLP)-based networks [5, 13], convolutional networks [3, 4, 6, 14–16], and transformer-based networks [17, 18]. These networks have been studied mostly using clean accuracy as their main metric, but an in-depth study of their robustness is lacking.

Vulnerability to Adaptive Attacks. A few defenses against 3D attacks have been recently proposed [19–21]. Some of the methods, however, merely obfuscate attackers by limiting the malicious agents from accessing the defense systems and true gradients [22]. They have been shown vulnerable to adaptive attackers who have the full knowledge of the defenses and can approximate the gradients [21]. Similar to the 2D domain [23], adversarial training (AT) [21] provides more longstanding robustness in 3D even against adaptive attacks [22]. Nevertheless, further robustness improvements on adversarial training are highly desired for practical usage and deployment.

Self-supervised learning (SSL) has been incorporated into adversarial training in 2D image perception models lately. It has shown great potential to enhance adversarial robustness without requiring any additional data or labels [24, 25]. Given such achievements, a natural question emerges: can we mimic the application of SSL to improve adversarial robustness in 3D point cloud recognition? Such a label-free strategy is preferred due to the cost and difficulty of 3D point cloud data annotation [26].

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.



(a) Illustration of Different Threats and Self-Supervised Learning Tasks.

(b) Illustration of Adversarial Pre-training for Fine-tuning (APF) and Adversarial Joint Training (AJT).

Figure 1: Overview of Our Analysis in 3D Point Cloud Classification.

Summary of Our Contributions:

In this paper, we present a systematic analysis of the adversarial robustness in 3D point cloud recognition using self-supervisions on three representative architectures: a multi-layer-perceptron (MLP) network (PointNet) [5], a convolutional network (DGCNN) [15], and a transformer-based network (PCT) [17]. Specifically, we use two strategies to integrate self-supervised learning and adversarial training, including (1) adversarial pre-training for fine-tuning (APF), which uses the SSL tasks only for pre-training, and (2) adversarial joint training (AJT), which jointly trains the SSL task with the recognition task, as shown in Figure 1. To further study the importance of self-supervised tasks for adversarial robustness, we select three representative SSL proxy tasks, including 3D rotation prediction [27], 3D jigsaw [28], and autoencoding [29]. Our key observations are as follows:

- We show that pre-training on SSL tasks improves adversarial robustness of the fine-tuned models. Unlike the 2D domain, where both APF and AJT have enhanced the robustness, our study finds that only APF consistently achieves robustness improvements in 3D. AJT does not always help since the distributional gap between data for SSL and recognition tasks will distract each other in AJT. Evaluation results of various unforeseen attacks further confirm such improvements by APF.
- We find that the convolutional network, *i.e.,* DGCNN, is more robust than the other architectures. Moreover, 3D jigsaw SSL task, which predicts the permutation of 3D point cloud patches, helps achieve stronger robustness than the others. Both convolutional architecture and jigsaw SSL task enforce the model to learn better local semantics. Intuitively, robust local features help limit the propagation of adversarial effect from point-level input perturbations to the model's final output.
- We demonstrate that fine-tuned models from different pre-training tasks have different vulnerabilities, and adversarial examples generated by attacking them do not transfer well among each other. Thus, we further leverage two simple yet powerful ensemble methods to boost the adversarial robustness by a substantial margin. Our best ensemble models achieve robust accuracy of 53.5% (+15.6%), 69.4% (+7.4%) and 57.9% (+8.8%) with PointNet, DGCNN, and PCT on ModelNet40.

2 Analysis Methodology

In this section, we detail our adversarial robustness analysis methodology. We first introduce the principal 3D point cloud recognition architectures and the threat models used in our study. We then introduce two ways to generalize and improve AT using 3D point cloud SSL proxies.

2.1 3D Point Cloud Recognition Models and Threats

We introduce the adopted model designs and the formulations of threats to 3D point clouds below.

Model Variants. We use a shared multi-layer-perceptron-based network PointNet [5], a convolutional network Dynamic Graph CNN (DGCNN) [15], and a transformer-based network Point Cloud Transformer (PCT) [17] as our primary backbone architectures, denoted as $\mathcal{M}_{\boldsymbol{\theta}_m}$. Specifically, PointNet directly aggregates learned features from each point to form a global embedding for final recognition. By doing so, the point-level features are independent of each other before global pooling. PointNet embraces high efficiency that has been widely utilized as a base operation in complex tasks [8, 30]. DGCNN instead builds a graph based on k-nearest neighbors (kNN) and uses a variant of continuous convolution (EdgeConv) on edges to enable local feature learning. The combination of local and global representation learning helps DGCNN achieve a higher clean accuracy. PCT extends the Transformer [31], the dominant framework in natural language processing, for point cloud recognition. The core of PCT is the full attention mechanism, which establishes a much more

flexible scheme, where each point has the potential to affect every other point in the point cloud. The classification head $\mathcal{H}_{\boldsymbol{\theta}_h}$ parameterized with $\boldsymbol{\theta}_h$ for these backbones is an MLP and the part segmentation head $\mathcal{H}_{\boldsymbol{\theta}_h}$ is a set of 1×1 convolutions. We use $\mathcal{F}_{\boldsymbol{\theta}_f}$ parameterized with $\boldsymbol{\theta}_f$ ($\boldsymbol{\theta}_f := [\boldsymbol{\theta}_m; \boldsymbol{\theta}_h]$) to represent the overall model architecture, consisting of the stacked backbone $\mathcal{M}$ and recognition head $\mathcal{H}$, where $\mathcal{F} = \mathcal{M} \circ \mathcal{H}$. Given the input point cloud $\boldsymbol{x}$, the model $\mathcal{F}$ aims to predict the corresponding label $\boldsymbol{y}$, where $\boldsymbol{y} = \mathcal{F}(\boldsymbol{x})$. More details of the architectures are in the supplements.

Threats. There are mainly three types of threats against point cloud perception models, which could be abstracted as point shifting (PS), dropping (PD), and adding (PA) attacks (Figure 1(a)). We formally define these threats within ℓ_p projected gradient descent (PGD) style attacks. First, we assume a PS adversary is able to shift all existing points within a ℓ_p norm ball:

$$\boldsymbol{x}_{s+1} = \Pi_{\boldsymbol{x}+\mathcal{S}}(\boldsymbol{x}_s + \alpha \cdot \mathrm{sign}(\nabla_{\boldsymbol{x}_s}\mathcal{L}(\boldsymbol{x}_s, \boldsymbol{y}; \mathcal{F}))); \quad \boldsymbol{x}_0 = \boldsymbol{x} + \mathrm{U}(-\epsilon, \epsilon) \tag{1}$$

where $\boldsymbol{x}_s$ is the adversarial example in the s-th iteration, Π is the projection function to project the adversarial example to the pre-defined perturbation space $\mathcal{S}$, α is the attack step size, and $\mathrm{U}(-\epsilon, \epsilon)$ represents a uniform distribution from $-\epsilon$ to ϵ. Second, we allow a PD adversary to discard the top k salient points [32] in each iteration until a total drop of N points:

$$\boldsymbol{x}_{s+1} = \boldsymbol{x}_s \ / \ \arg\ \mathrm{top}k_{\boldsymbol{x}\in\boldsymbol{x}_s}\ \mathrm{saliency}(\boldsymbol{x}_s, \boldsymbol{y}; \mathcal{F}); \quad \text{if} \ \ k \times s < N \tag{2}$$

where $\mathrm{saliency}()$ calculates the importance scores of the input points and $/$ denotes the dropping operation. We detail $\mathrm{saliency}()$ in the supplements due to space limits. Third, a PA adversary is capable of adding new points bounded by an ℓ_p norm ball to the original point cloud. PA randomly initializes N new points from the original point coordinates and only perturb the added points. We follow the same setting as PS to formulate the perturbation:

$$\boldsymbol{x}_{s+1} = \Pi_{\boldsymbol{x}+\mathcal{S}}(\boldsymbol{x}_s + \alpha \cdot \mathrm{sign}(\nabla_{\boldsymbol{x}_s}\mathcal{L}([\boldsymbol{x}_{ori}; \boldsymbol{x}_s], \boldsymbol{y}; \mathcal{F}))); \ \boldsymbol{x}_0 = \mathrm{sample}(\boldsymbol{x}_{ori}, N) + \mathrm{U}(-\epsilon, \epsilon) \tag{3}$$

where $\boldsymbol{x}_{ori}$ is the original point cloud and $\mathrm{sample}()$ initializes N new points denoted as $\boldsymbol{x}_0$. We believe the chosen threats mostly cover the existing attack surfaces on point cloud data. It is also beneficial to study how different threats would affect the adversarial robustness in point cloud recognition under our adversarial training strategies which will be introduced next.

2.2 Adversarial Training with Self-Supervisions

We first introduce the chosen 3D self-supervised learning methods, followed by two strategies to incorporate these pretext tasks in adversarial training.

3D Self-Supervised Learning. The primary goal of self-supervised learning (SSL) is to learn effective feature representations with unlabeled data. Given a pretext task P_t , the pre-training process is still conducted in a supervised manner with self-generated data $\boldsymbol{x}^t$ and label $\boldsymbol{y}^t$ from pristine data $\boldsymbol{x}$, where $(\boldsymbol{x}^t, \boldsymbol{y}^t) = P_t(\boldsymbol{x})$. Therefore, a target loss function $\mathcal{L}_t(\boldsymbol{x}^t, \boldsymbol{y}^t; \mathcal{F}^t_{\boldsymbol{\theta}_t})$ will be minimized during the optimization, where $\boldsymbol{\theta}_t$ consists of the shared backbone parameters $\boldsymbol{\theta}_m$ and customized branch parameters $\boldsymbol{\theta}_c$ (*i.e.,* $\boldsymbol{\theta}_t := [\boldsymbol{\theta}_m; \boldsymbol{\theta}_c]$). We utilize the following 3D SSL tasks in our study (Figure 1(a)).

• *3D Rotation* [27]: Similar to the rotation task in 2D vision [33], the data and label are generated by rotating the original point clouds to pre-defined angles η in the 3D space. Therefore, the problem is to correctly predict 3D rotation angles *w.r.t.* the input point cloud. The I seobjective function can be formulated as a cross-entropy (CE) loss: $\mathcal{L}_{rotation} = \mathrm{CE}(\boldsymbol{x}^{rotation}, \boldsymbol{y}^{rotation}; \mathcal{F}^t)$.

• *3D Jigsaw* [28]: Different from the jigsaw task in 2D vision [34] which is defined as a classification problem, 3D jigsaw solicits a segmentation model. A point cloud is evenly divided to k^3 small cubes and shuffled to different positions. Points inside each small cube are assigned to a label signaling its original position. The problem, thus, is to correctly predict the original cube position of each point. Similarly, its objective is to minimize a CE loss: $\mathcal{L}_{jigsaw} = \mathrm{CE}(\boldsymbol{x}^{jigsaw}, \boldsymbol{y}^{jigsaw}; \mathcal{F}^t)$.

• *Autoencoder* [29, 35]: An autoencoder utilizes an encoder $z = E(\boldsymbol{x})$ to learn a compact representation and a decoder $D(E(\boldsymbol{x}))$ to reconstruct the point cloud. We utilize different backbones as the encoder $E(\cdot)$ and FoldingNet [29] as the decoder $D(\cdot)$ due to its satisfactory performance. We use three different positional encodings: plane, 3D sphere, and 3D gaussian in our experiments. we use the Chamfer distance [36] as the reconstruction loss: $\mathcal{L}_{ae} = \mathrm{Chamfer}(D(E(\boldsymbol{x}^{ae})), \boldsymbol{x}^{ae}; \mathcal{F}^t)$.

The detailed description of the pretext tasks can be found in the supplements.

Adversarial Pre-training for Fine-tuning (APF). As introduced in §1, adversarial training (AT) [23, 37, 38] has been demonstrated to be one of the most longstanding and practical defenses. We thus

enable AT in both pre-training and fine-tuning stages:

$$\arg\min_{\boldsymbol{\theta}} \quad \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}\left[\max_{\sigma\in\mathbb{S}} \mathcal{L}(\boldsymbol{x}+\sigma,\boldsymbol{y},\boldsymbol{\theta})\right] \tag{4}$$

where $\mathcal{L} \in \{\mathcal{L}_t, \mathcal{L}_f\}$ for loss functions in pre-training (t) and fine-tuning (f) stages, σ is the adversarial perturbations, and $\mathbb{S}$ represents its manipulation space. AT essentially solves a min-max problem. In the inner loop, the optimizer tries to find adversarial examples that maximize the target loss, and the outer loop updates the network parameters to correctly recognize the generated adversarial examples. In contrast, standard training (ST) is simply $\arg\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}\left[\mathcal{L}(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\theta})\right]$.

In the pre-training stage of APF, we leverage both standard and adversarial training to get the pre-trained backbones $\mathcal{M}_{\boldsymbol{\theta}_m}$ and $\mathcal{M}^{adv}_{\boldsymbol{\theta}_m}$. Given a pre-trained backbone parameterized by $\boldsymbol{\theta}_m$, in the second stage, we adversarially fine-tune all $\boldsymbol{\theta}_f := [\boldsymbol{\theta}_m; \boldsymbol{\theta}_h]$ for the recognition task, as illustrated in Figure 1(b). The network branches at the penultimate vector for the rotation task and the first global feature [21] for the jigsaw and autoencoder tasks since they use the segmentation head.

Adversarial Joint Training (AJT). Besides pre-training for fine-tuning, joint training is another way to apply SSL. The objective function is formulated as:

$$\arg\min_{\boldsymbol{\theta}_m;\boldsymbol{\theta}_h;\boldsymbol{\theta}_c} \quad \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}\left[\max_{\sigma\in\mathbb{S}} \mathcal{L}_f(\boldsymbol{x}+\sigma,\boldsymbol{y},\boldsymbol{\theta}_f)\right] + \lambda\cdot\mathcal{L}_t(\boldsymbol{x}^t,\boldsymbol{y}^t,\boldsymbol{\theta}_t) \tag{5}$$

where λ is a hyperparameter to balance the SSL and recongnition tasks. Two tasks share the same backbone $\boldsymbol{\theta}_m$ with two different branches, parameterized by $\boldsymbol{\theta}_h$ and $\boldsymbol{\theta}_c$, respectively. We also enable dual batch normalization [39] in AJT for $\boldsymbol{x}$ and $\boldsymbol{x}_t$ since they should belong to different underlying distributions. We use two model-agnostic tasks, *i.e.,* 3D rotation and jigsaw in AJT.

Similarly, in our AJT analysis, all $\mathcal{L}_t$ and $\mathcal{L}_f$ can be formulated as CE loss. We empirically set $\lambda = 1$ and leverage the same branching point with APF. The whole network is trained to predict the supervised task with the original head and the SSL task with the auxiliary head (Figure 1(b)).

3 Experiments and Results

In this section, we present our experimental setups and results. We first introduce the adopted datasets and adversarial settings. We leverage PS as the primary threat in our study since we find it is the most powerful adversary and the other two adversaries easier to detect. Next, we detail our extensive evaluation of two fundamental point cloud recognition tasks: classification and part segmentation.

3.1 Evaluation Setups

Datasets. We leverage four datasets ($\mathcal{D}$): ModelNet40 [40] (40 classes), ModelNet10 [40] (10 classes), ScanObjectNN [41] (15 classes), and ShapeNetPart [42] throughout our experiments. Specifically, the first three datasets are utilized for the classification task and the last one is for the part segmentation task. ModelNet and ShapeNetPart are captured from CAD models, and ScanObjectNN is scanned and extracted from real-world indoor scenes. For each point cloud, we randomly sample 1024 points and normalize it to an edge-length-2 cube ($[-1, 1]$) for experimentation. We follow the default split of training and test sets in [5] and [43]. For $\mathcal{D}_t$, we randomly sample $\boldsymbol{y}_t$ from the pre-defined label sets and further generate $\boldsymbol{x}_t$ based on $\boldsymbol{y}_t$ in each iteration. Specifically, we choose $\eta = 6, 18$ and $k = 3, 4$ for rotation and jigsaw tasks, followed by the suggestion of [27] and [28].

Adversary. As introduced in §2.1, for PS, we exploit 7-step and 200-step ℓ_∞ PGD attacks [23] targeting the cross-entropy loss for adversarial training and testing, respectively. We follow Sun *et al.* [21] to empirically set the perturbation boundary $\epsilon = 0.05$ ($||\sigma||_\infty \leq 0.05$). Note that, different from discrete RGB values in 2D images, point cloud's features (coordinates) are continuous. We thus utilize PGD step size $\alpha = 0.01$ and $\alpha = 0.005$ in the training and testing phases, respectively. We allow a stealthy perturbation of $N = 100$ points for PD and PA since we believe a larger number of modification would be easy to detect. We drop the most $k = 14, 5$ salient points every iteration in training and testing phases for PD. The attack setups in PA are the same as those in PS since they utilize similar ℓ_∞ PGD perturbation methods. We exploit PS as the primary threat for the classification and part segmentation tasks, and leverage the other two threats to further demonstrate our robustness improvements in the classification task.

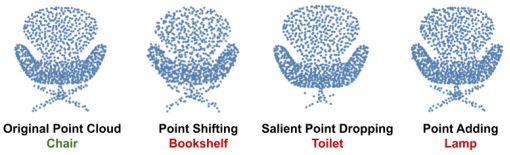


Figure 2: Adversarial Examples of Different Threats.

3.2 Point Cloud Classification

In this section, we first introduce the setups of our point cloud classification analysis. Next, we introduce the detailed study of APF and AJT. We further evaluate our robust models with unforeseen attacks to demonstrate that our robustness improvements are non-trivial. We find that pre-trained models from different SSL tasks preserve different vulnerabilities; hence we use simple yet powerful ensemble methods to boost the robustness. Lastly, we evaluate our methods in different threats to demonstrate their generality.

Training Details. All pre-trained and fine-tuned models in APF are trained using Adam [44]. We use batch sizes of 32 for PointNet and DGCNN, and 128 for PCT. The initial learning rate is set to 0.001 for PointNet and DGCNN, and 5×10^{-4} for PCT. Both pre-training and fine-tuning take 250 epochs, where a $10\times$ decay happens at the 100-th, 150-th, and 200-th epoch. We leverage the same training setups in AJT. All experiments are done on 1 to 4 NVIDIA V100 GPUs [45].

Model Adaptation to AT. First, PointNet [5] leverages exponential learning rate decay, and DGCNN and PCT utilize cosine annealing learning rate decay. Through preliminary experiments, we find that a piecewise decay of the learning rate improves the default baseline by a noticeable margin. Second, two T-Nets in PointNet [5] will also make AT unstable since they holistically modify the point cloud and features. We thus remove them in our experiments. Third, PCT default leverage farthest point sampling (FPS) to sample and group local points, which will cause AT unstable since the perturbed point cloud samples different anchor points in each iteration. We replace FPS with EdgeConv on each point. It is worth noting that such a change will not modify the application of self-attention in PCT. As presented in Table 1[1], the modifications will both improve the standard and adversarial training baseline, and we introduce the details in the supplements.

Table 1: Summarized Results (%) on ModelNet40 by Model Adaptation.

	PointNet		DGCNN		PCT	
	CA	RA	CA	RA	CA	RA
Default ST	88.0	0.0	91.3	2.9	91.9	0.0
Selected ST	88.7	0.3	91.5	3.2	92.1	2.3
Default AT	87.1	33.6	90.6	58.7	88.4	44.2
Selected AT	87.7	37.9	90.6	62.0	89.7	49.1

3.2.1 Self-Supervised Pre-training Helps Adversarial Fine-tuning

We systematically evaluate all configurations in APF under PS attack. As introduced earlier, we use standard and adversarial training to get the pre-trained models. From Table 2, we can make several interesting observations. First, we find that our APF strategy generally enhances the adversarial robustness. The best-fine-tuned models achieve 14.2%, 5.4%, and 2.2% robustness improvements in PointNet, DGCNN, and PCT on ModelNet40, respectively. The enhancements on the real-world dataset, ScanObjectNN, *i.e.,* 1.8%, 10.4%, and 6.9% in PointNet, DGCNN, and PCT, are also significant, which demonstrate the generality of APF. Second, we find that DGCNN outstands to be the most robust architecture, consistently achieving $\sim$15% stronger robustness than the other two models on both ModelNet40 and ScanObjectNN. Lastly, jigsaw-based APF offers more robustness improvements than the other two methods while maintaining slightly higher clean accuracy (CA). Specifically, jigsaw-based APF, on average, further boosts DGCNN's RA by 2.8%, 2.2%, and 2.7% on three datasets, respectively.

Insights. Different from 2D images that possess both texture and shape information, 3D point clouds naturally bias towards shape. In 2D image space, it is widely recognized that local and global features correspond to the texture and shape information, respectively [46]. Recent studies have demonstrated that appreciation of global/shape features can help improve model robustness on image classification [47]. However, we find some distinctions in point cloud recognition. As mentioned above, PointNet with only global feature learning will be easily affected by the perturbed points (Table 2). Due to the sparsity of point clouds, the local feature actually represents the smoothness of the object's surface. Thus, learning robust local features is critical for correctly recognizing a perturbed point cloud, as it limits the adversarial effect propagation to the model output.

As summarized above, DGCNN achieves the strongest robustness under AT, attributed to the hierarchical usage of EdgeConv [15]. EdgeConv dynamically aggregates local features by exploiting kNN. Such an aggregation method has the ability to calibrate the adversarial effect in the local feature learning stage. Although transformer-based architectures have gained tremendous visibility recently [48], we find that PCT does not have a major robustness improvement compared to PointNet. Self-attention increases the capacity of the model architecture, but it also enlarges the receptive field

[1]The 1-st and 2-nd highest accuracy among fine-tuned models in each column are noted, and we use the same mark throughout this paper.

Table 2: Evaluation Results (%) of Adversarial Pre-training for Fine-tuning and Task Ensembles.

		ModelNet40						ScanObjectNN						ModelNet10					
		PointNet		DGCNN		PCT		PointNet		DGCNN		PCT		PointNet		DGCNN		PCT	
Pretext Task	Parameters	CA	RA	CA	RA	CA	RA	CA	RA	CA	RA	CA	RA	CA	RA	CA	RA	CA	RA
AT Baseline	N/A	87.7	37.9	90.6	62.0	89.7	49.1	69.9	23.7	74.4	30.9	72.4	20.5	96.6	79.7	98.1	86.3	97.4	80.0
3D Rotation	$\eta = 6$	87.2	48.0	91.4	63.6	90.2	50.7	69.1	24.5	75.7	32.9	72.6	20.6	96.8	79.0	97.7	84.9	97.2	80.4
	$\eta = 18$	87.2	48.3	91.1	64.1	90.2	49.5	69.5	25.0	73.8	32.2	72.5	20.1	97.1	79.3	98.5	85.3	97.8	80.3
Adversarial 3D Rotation	$\eta = 6$	87.6	42.1	90.8	61.8	90.4	50.8	69.6	25.3	75.0	36.8	71.6	28.7	97.0	79.9	97.7	87.5	98.0	82.2
	$\eta = 18$	87.4	45.7	90.9	62.9	90.4	50.1	69.3	24.5	75.0	36.3	73.1	26.9	97.0	79.7	98.0	88.2	97.4	83.7
3D Jigsaw	$k = 3$	87.6	50.1	90.0	67.4	90.4	51.1	70.8	25.5	79.0	33.8	73.4	23.2	96.8	80.0	98.0	89.6	97.8	81.5
	$k = 4$	87.6	50.9	90.1	65.3	90.3	50.2	70.2	25.4	76.2	35.3	73.8	24.6	96.7	80.2	98.0	89.0	97.7	81.9
Adversarial 3D Jigsaw	$k = 3$	88.2	52.1	89.6	65.8	89.8	51.3	69.0	24.8	77.5	41.3	72.5	26.3	97.0	80.6	98.5	90.5	97.4	83.5
	$k = 4$	87.8	50.5	89.9	65.3	89.6	51.0	69.9	25.5	76.1	40.6	73.1	27.4	97.0	80.5	98.0	89.1	97.3	83.9
Autoencoder	sphere	87.4	50.0	89.9	62.8	90.2	50.7	69.9	25.1	76.1	36.0	71.3	24.1	97.0	80.5	98.2	86.8	97.1	80.1
	plane	87.1	48.8	90.1	62.2	90.2	50.2	69.4	25.5	76.2	35.6	71.1	22.6	96.8	80.8	97.8	87.6	97.0	80.1
	gaussian	87.4	48.9	90.8	63.3	89.7	50.3	69.7	23.8	75.6	35.8	71.3	24.8	96.8	80.5	97.8	86.4	97.1	80.1
Adversarial Autoencoder	sphere	87.1	49.7	90.0	62.2	90.3	50.0	70.4	25.2	75.2	36.2	72.6	22.2	96.7	80.4	97.5	87.3	97.5	82.1
	plane	86.9	46.6	89.7	61.8	89.7	50.0	69.2	24.0	75.6	38.0	73.3	21.6	97.0	80.6	98.0	86.1	97.7	82.5
	gaussian	87.1	48.5	90.7	62.7	90.2	50.5	68.8	25.0	74.7	36.3	72.6	23.4	97.0	80.2	97.8	88.4	97.4	83.2
Max Ensemble		**88.5**	**53.5**	**91.4**	**69.4**	**90.5**	55.4	**70.8**	**27.3**	**79.1**	**42.6**	74.0	**28.9**	**97.2**	**82.5**	98.5	**91.0**	**98.1**	**85.2**
Mean Ensemble		88.4	52.5	91.4	68.7	90.4	**57.9**	70.8	26.9	79.0	41.9	**74.2**	28.4	97.0	82.4	**98.6**	90.9	98.0	84.9

of the model [49]. In PCT, each point can influence every other point's feature, which will potentially increase the model's fragility [50].

Moreover, we also find that jigsaw-based APF is the most effective method to improve adversarial robustness, aligning well with our above insights. Jigsaw SSL makes the model learn to reassemble the randomly displaced local point clusters, where the model is enforced to learn the displaced local features. Meanwhile, to correctly reconstruct the point cloud, jigsaw SSL also requires the model to capture the global and holistic semantics. Nevertheless, rotation and autoencoder-based pre-training methods focus more on global feature learning. Therefore, we believe jigsaw-based APF is a perfect candidate to strengthen the association between local and global features in point cloud learning, hence improving the adversarial robustness under APF.

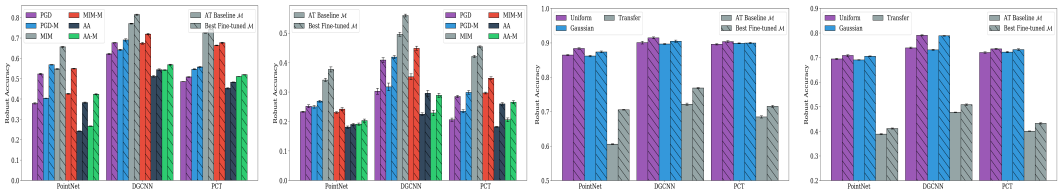
3.2.2 Adversarial Joint Training Does not Always Improve Robustness

We leverage two model-agnostic SSL tasks in AJT. As presented in Table 3, AJT can still enhance the robustness in PointNet and DGCNN. For instance, AJT improves their RA by 1.8% and 8.0% on ScanObjectNN, respectively. However, AJT overall cannot outperform APF in point cloud recognition. Especially, we find AJT even degrades the RA of PCT compared to the standard AT.

Table 3: Evaluation Results (%) of Adversarial Joint Training.

		ModelNet40						ScanObjectNN						ModelNet10					
		PointNet		DGCNN		PCT		PointNet		DGCNN		PCT		PointNet		DGCNN		PCT	
Pretext Task	Parameters	CA	RA	CA	RA	CA	RA	CA	RA	CA	RA	CA	RA	CA	RA	CA	RA	CA	RA
AT Baseline	N/A	87.7	37.9	90.6	62.0	89.7	49.1	69.9	23.7	74.4	30.9	72.4	20.5	96.6	79.7	98.1	86.3	97.4	80.0
3D Rotation	$\eta = 6$	86.8	45.0	91.2	60.7	89.5	44.3	67.8	24.3	74.2	37.8	72.3	20.3	96.6	79.0	98.1	86.3	97.8	73.8
	$\eta = 18$	86.5	46.4	91.3	62.0	88.9	42.9	68.7	25.1	76.2	37.2	72.1	19.8	97.0	79.9	97.9	85.7	98.1	75.6
3D Jigsaw	$k = 3$	87.6	42.5	91.0	62.3	90.2	43.1	69.4	25.5	77.1	38.9	72.1	20.7	96.8	79.8	98.4	87.9	97.7	76.8
	$k = 4$	87.2	46.7	91.1	61.7	89.8	40.9	70.0	24.6	75.9	38.4	73.7	20.8	96.8	77.9	98.0	88.6	97.1	78.0

Insights. We find this also to be related to the natural characteristic of point cloud data. Although SSL can help models learn strong priors and context information, it is still a *separate* learning task. Rotated and disassembled images still preserve similar local features to the original images since the RGB values do not change, so that the auxiliary optimization in AJT will not distract AT but help models learn robust global features [24]. However, point cloud models take point coordinates xyz as input. Rotated and disassembled point clouds have significant variations in their coordinates' numeric values. Although we apply dual batch normalization [51] to migrate the feature heterogeneous problems, such discrimination will consequently distract model learning in AJT, and thus hurt the RA performance. The usage of self-attention in PCT will further expand this impact since it introduces a global receptive field [50].



(a) ModelNet40. (b) ScanObjectNN. (c) ModelNet40. (d) ScanObjectNN.

Figure 3: Evaluation Results of Unforeseen Attacks. The first two plots present the evaluation results of unforeseen adversarial attacks, where "-M" denotes the attacks on margin loss. The rest plots present the evaluation results of noise and transfer attacks.

3.2.3 Robustness against Unforeseen Attacks and Noises

We have so far shown that APF generally improves the robustness of point cloud classification. In this section, we leverage unforeseen attacks to further demonstrate that the enhancements are non-trivial. We select the fine-tuned models with the highest RA in Table 2, and all results here are averaged from five runs using different random seeds. We present the results on ModelNet10 in the supplements.

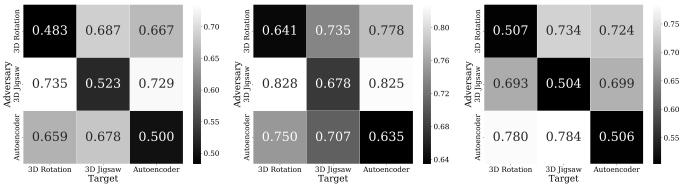
We leverage unforeseen attacks: Auto Attack (AA) [52], Momentum Iterative Method (MIM) [53], and PGD with both cross-entropy and margin loss (*i.e.,* $\mathcal{L}_{\text{margin}} = Z(\boldsymbol{x})_y - \max_{i \neq y} Z(\boldsymbol{x})_i$, where $Z(\boldsymbol{x})$ is the logit output of the target classifier). We elaborate the details of AA and MIM in the supplements. All attacks use 200 steps to find the potential adversarial examples. As shown in Figure 3(a) and 3(b), the best fine-tuned models consistently achieve higher RA than models from standard AT with 2.6% - 15.3% and 1.0% - 10.1% improvements on ModelNet40 and ScanObjectNN, respectively. We find that DGCNN still outperforms the other two models achieving the highest RA even against the strongest Auto Attacks [52].

Moreover, we exploit the transfer attack to confirm the effectiveness of APF. We fine-tune five models using the same (best) setting but different random seeds for transfer-based attacks and test the transferability of generated adversarial examples among different models. As Figure 3(c) and 3(d) show, adversarial examples cannot transfer well and there is a $\sim$10% gap in RA compared to attacks on target models, which further validates the real robustness improvements.

Lastly, We find that Gaussian and uniform noises are ineffective in breaking model robustness, and the RA is therefore close to the CA. As shown in Figure 3(c) and 3(d), the fine-tuned models still outperform standard AT baseline by 0.1% - 2.2% and 1.0% - 5.7% on ModelNet40 and ScanObjectNN, respectively, in RA.

3.2.4 Attack Transferability and Task Ensemble

Since we leverage three pre-text tasks for self-supervised pre-training, we also test the transferability of adversarial examples generated by models pre-trained on different SSL tasks. As shown in Figure 4, different fine-tuned models preserve different vulnerabilities, and there is at least $\sim$10% gain of RA when transferring attacks on models fine-tuned from different tasks on ModelNet40. We thus combine the models and further test the ensemble model's robustness. We use two ensemble methods, which are taking the max and mean value of the stacked logits from individual models *i.e.,* $\mathrm{L}^i_{ensemble} = \max/\mathrm{mean}([\mathrm{L}^i_{rotation}, \mathrm{L}^i_{jigsaw}, \mathrm{L}^i_{autoencoder}])$ where L^i denotes the i-th value in the logit L. By doing so, we form a "wider" model where the gradients can still propagate smoothly back to the input. Therefore, we also exploit the 200-step PGD attack to test their robustness. As presented in the last two rows of Table 2, both ensemble models can further boost the RA by a significant margin while maintaining similar clean accuracy. For example, our best ensemble models achieve robust accuracy of 53.5% (+15.6%), 69.4% (+7.4%) and 57.9% (+8.8%) with PointNet, DGCNN, and PCT on ModelNet40 respectively. It is worth noting that our fine-tuning strategy does not enforce



(a) PointNet. (b) DGCNN. (c) PCT.

Figure 4: Robust Accuracy on Transfer Attacks among Fine-tuned Models from Different SSL Tasks on ModelNet40.

a diversity regulation, unlike [25], which further demonstrates that pre-training on different pretext tasks indeed brings distinct context information into pre-trained models.

3.2.5 Robustness against Other Point Cloud Threats

As mentioned before, we also consider two other threats that specifically target point cloud recognition. As we have demonstrated that jigsaw-based APF reaches the best robustness enhancement under point shifting (PS), we replace PS attack in AT with salient point dropping (PD) and point adding (PA) to test the effectiveness of jigsaw-based APF strategy.

Table 4: Evaluation (%) of PD on ScanObjectNN.

		PointNet		DGCNN		PCT	
Pretext Task	Parameters	CA	RA	CA	RA	CA	RA
ST Baseline	N/A	69.8	48.5	79.3	60.9	74.2	56.1
AT Baseline	N/A	70.9	64.2	76.2	77.1	73.3	73.7
3D Jigsaw	$k = 3$	71.3	65.2	81.4	80.0	76.8	76.9
	$k = 4$	72.5	66.3	81.2	80.7	77.8	80.1
Adversarial 3D Jigsaw	$k = 3$	71.1	64.7	78.7	79.5	75.4	76.2
	$k = 4$	70.9	65.7	79.3	80.4	75.9	78.3

Table 5: Evaluation (%) of PA on ScanObjectNN.

		PointNet		DGCNN		PCT	
Pretext Task	Parameters	CA	RA	CA	RA	CA	RA
ST Baseline	N/A	69.8	57.6	79.3	38.0	74.2	39.6
AT Baseline	N/A	69.0	59.4	77.4	63.5	72.1	56.8
3D Jigsaw	$k = 3$	70.2	59.9	82.3	66.3	75.7	60.4
	$k = 4$	70.4	60.6	83.1	66.6	76.4	58.7
Adversarial 3D Jigsaw	$k = 3$	70.4	62.5	77.5	67.8	75.2	64.4
	$k = 4$	70.7	59.4	79.5	68.2	74.4	65.5

We find that jigsaw-based APF can still strengthen the robustness under these two threats on three datasets. Especially, as shown in Table 4 and 5, DGCNN still achieves the highest RA, which on average outperforms PointNet and PCT by 14.8% and 2.3% in PD-AT, and 6.6% and 4.9% in PA-AT, respectively, which is consistent with our previous findings. Moreover, as PD and PA tend to be much weaker adversaries than the PS attack, the enhancements mainly appear in the challenging real-world dataset, ScanObjectNN, where jigsaw-based APF boosts the RA of DGCNN by 3.6% and 4.7% under PD and PA. The key insights still hold in ModelNet datasets under these two threats, and we present the detailed results in the supplements due to space constraints.

3.3 Point Cloud Part Segmentation

In addition to object classification, we conduct the *first* study on analyzing the robustness of the point cloud part segmentation on ShapeNetPart [42] with 17 classes of 3D objects. Apart from semantic segmentation that all samples from a dataset share the same set of labels, part segmentation task assigns a unique label set for each class of objects. For instance, the class of chair has the label set: {seat, back, arm, leg} and the airplane has {wing, body, tail, engine}, as illustrated in Figure 5. We use the same backbone with a segmentation head for this task and mean inter-over-union (mIoU) [43] as the evaluation metric. Especially since each class possesses its own label set, the models also take the category (one-hot vector) as input. As we find APF is an effective and efficient strategy, we use the PS attack with the same AT and APF setups as the classification task to train the models (§ 3.1).

Table 6: Evaluation Results (%) of the Part Segmentation Task.

		PointNet		DGCNN		PCT	
Pretext Task	Parameters	C-mIoU	R-mIoU	C-mIoU	R-mIoU	C-mIoU	R-mIoU
ST Baseline	N/A	83.2	32.5	84.2	42.0	84.2	34.6
AT Baseline	N/A	79.2	62.7	82.9	69.5	82.6	67.8
3D Rotation	$\eta = 6$	79.1	63.0	82.8	69.3	82.8	67.7
	$\eta = 18$	79.3	63.0	82.9	69.5	82.9	67.5
Adversarial 3D Rotation	$\eta = 6$	79.3	62.9	82.9	69.2	82.2	67.2
	$\eta = 18$	79.1	62.6	82.9	69.0	81.0	65.7
3D Jigsaw	$k = 3$	79.4	63.0	82.7	70.1	82.8	68.1
	$k = 4$	79.4	63.1	82.9	70.3	82.4	68.4
Adversarial 3D Jigsaw	$k = 3$	79.7	62.5	82.8	69.7	81.1	65.8
	$k = 4$	79.9	62.7	82.7	70.3	82.0	66.7

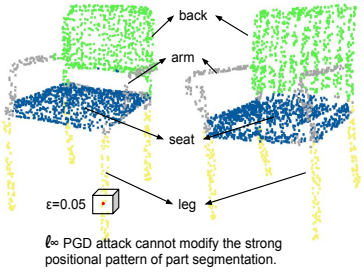


Figure 5: Illustration of the Point Cloud Part Segmentation Using "Chair".

Table 6 summarizes the quantitative results. To our surprise, we find that part segmentation is a more robust task than classification. The robust mIoU is relatively high even in the ST baseline. Moreover, DGCNN still achieves the highest clean and robust mIoU in part segmentation, further confirming its supremacy in point cloud learning. Furthermore, jigsaw-based APF can also promote the robustness of part segmentation by 0.4%, 0.8%, and 0.6% in PointNet, DGCNN, and PCT, respectively.

Insights. We attribute the natural robustness of part segmentation to the intrinsic feature of point clouds. Since all point clouds in ShapeNetPart are sampled from CAD models, they are posed to face the same direction. Thus, models may memorize the 3D space corresponding to the specific label during learning. Taking the class of chair as an example, the points whose $z \in [-1, 0]$ most

likely belong to the "leg" label. However, ℓ_∞-based adversary with a $\epsilon = 0.05$ might not be able to perturb this strong pattern (Figure 5). We also envision that the robustness of part segmentation may help and transfer to other complex learning tasks if appropriately used. Moreover, DGCNN with jigsaw-based APF still delivers the best performance, which further generalizes our previous findings on the importance of balancing local and global features in point cloud learning.

4 Related Work

In this section, we review a few topics related to the adversarial robustness of 3D perception: 3D deep learning, self-supervised learning, and adversarial attacks and defenses.

4.1 Deep Learning on 3D Point Clouds

In image-based perception, there has been stellar progress on architectures and learning algorithms for convolutional neural networks. However, in 3D perception, there has been no consensus on the type of convolutions for 3D data partially because there is no standard data format for 3D perception. Yet, point clouds are the most commonly used data format because both 3D scanner raw data and triangular meshes can easily be converted into a point cloud. Some of the earlier 3D networks use dense voxel grids for perception [1, 2, 54, 55], which discretize a point cloud to voxel grids for classification, segmentation, and object detection. Later, to overcome cubic memory complexity, various types of data structures have been proposed: octrees [3, 4], surfaces [56], graphs [15, 57], point sets [5, 14, 17], and sparse tensors [6, 16]. In this work, we focus on 3D pint cloud preception with PointNet [5], DGCNN [15], and PCT [17] as our evaluation backbones since they are widely used and achieve state-of-the-art results for point cloud recognition [58].

4.2 Self-Supervised Learning Approaches

Self-supervised or unsupervised learning (SSL) techniques have shown effective in learning powerful representation without manual efforts, where the labels are generated from the data itself [59–61]. The embeddings pre-trained through self-supervision could be utilized for fine-tuning multiple downstream supervised tasks with better generalization and calibration [62, 63]. Because of the impressive success of SSL in 2D computer vision [33], recent studies have also proposed several self-supervisions for 3D point clouds including autoencoders [29], generative adversarial networks (GANs) [35], 3D rotation [27], and 3D jigsaw [28]. Those primitives hold promises to improve the clean accuracy of point cloud learning. This work systematically explores whether and how SSL could be leveraged to improve the adversarial robustness in point cloud recognition.

4.3 Adversarial Attacks and Defenses

Despite the accomplishments that DNNs have achieved, adversarial attacks [64] are becoming the major obstacle in real-world deep learning deployments, especially in safety-critical areas [11, 65]. Numerous attacks have been widely studied for various tasks in 2D [66, 66–71] and 3D [12, 72, 73] domain. To address this problem, many defense methods have been proposed to enhance the robustness against adversarial attacks in the 2D [74–78] and 3D domain [19–21]. However, most of them including adding randomization [20, 79, 80], model distillation [77], adversarial detection [78], and input transformation [19, 74–76, 81] have been compromised by adaptive attacks [21, 22, 82]. Certified methods are recently applied to 3D, but they focus on threat models with modified point numbers [83] and isometric transformations [84], which are applicable to ℓ_∞ norm. Adversarial training (AT) [23, 37, 38, 85], on the other hand, provides one of the most longstanding and practical defenses. Various methods have been proposed to improve AT in the 2D domain [51, 86, 87]. Jeddi *et al.* establish a fast adversarial fine-tuning scheme [88]. Hendrycks *et al.* [24] have shown to jointly train the adversarial loss and SSL loss to improve the robustness. Chen *et al.* [25] have applied adversarial pre-training and fine-tuning strategy to improve the robustness. In this work, we explore to apply and improve AT in 3D point cloud recognition.

5 Conclusion

In this work, we systematically explore the impact of self-supervised learning (SSL) on the adversarial robustness in 3D point cloud recognition. We find tangible robustness improvements by the adversarial pre-training for fine-tuning strategy. We also experimentally show that robust local features are critical to achieving robustness in 3D, explaining the success of DGCNN and the jigsaw proxy task. Our results shed light for future research on designing more robust models and SSL schemes for 3D point clouds. By providing empirical evaluations on the robustness, our study motivates future research in developing theoretical guarantees on the robustness in the 3D domain.

References

[1] Dominic Zeng Wang and Ingmar Posner. Voting for voting in online point cloud object detection. In *Robotics: Science and Systems*, volume 1, pages 10–15607. Rome, Italy, 2015.

[2] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015.

[3] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[4] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 36(4):1–11, 2017.

[5] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[6] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019.

[7] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking. *CVPR*, 2021.

[8] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019.

[9] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020.

[10] Yulong Cao, Chaowei Xiao, Benjamin Cyr, Yimeng Zhou, Won Park, Sara Rampazzi, Qi Alfred Chen, Kevin Fu, and Z Morley Mao. Adversarial sensor attack on lidar-based perception in autonomous driving. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 2267–2281, 2019.

[11] Jiachen Sun, Yulong Cao, Qi Alfred Chen, and Z. Morley Mao. Towards robust lidar-based perception in autonomous driving: General black-box adversarial sensor attack and countermeasures. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 877–894. USENIX Association, August 2020.

[12] Chong Xiang, Charles R Qi, and Bo Li. Generating 3d adversarial point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9136–9144, 2019.

[13] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[14] P. Hermosilla, T. Ritschel, P-P Vazquez, A. Vinacua, and T. Ropinski. Monte carlo convolution for learning on non-uniformly sampled point clouds. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2018)*, 37(6), 2018.

[15] 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 Transactions On Graphics (tog)*, 38(5):1–12, 2019.

[16] Benjamin Graham and Laurens van der Maaten. Submanifold sparse convolutional networks. *arXiv preprint arXiv:1706.01307*, 2017.

[17] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *arXiv preprint arXiv:2012.09688*, 2020.

[18] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. *arXiv preprint arXiv:2012.09164*, 2020.

[19] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1961–1970, 2019.

[20] Xiaoyi Dong, Dongdong Chen, Hang Zhou, Gang Hua, Weiming Zhang, and Nenghai Yu. Self-robust 3d point recognition via gather-vector guidance. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11513–11521. IEEE, 2020.

[21] Jiachen Sun, Karl Koenig, Yulong Cao, Qi Alfred Chen, and Z. Morley Mao. On the adversarial robustness of 3d point cloud classification, 2020.

[22] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, pages 274–283. PMLR, 2018.

[23] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[24] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. *arXiv preprint arXiv:1906.12340*, 2019.

[25] Tianlong Chen, Sijia Liu, Shiyu Chang, Yu Cheng, Lisa Amini, and Zhangyang Wang. Adversarial robustness: From self-supervised pre-training to fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 699–708, 2020.

[26] Charles R Qi, Yin Zhou, Mahyar Najibi, Pei Sun, Khoa Vo, Boyang Deng, and Dragomir Anguelov. Offboard 3d object detection from point cloud sequences. *arXiv preprint arXiv:2103.05073*, 2021.

[27] Omid Poursaeed, Tianxing Jiang, Quintessa Qiao, Nayun Xu, and Vladimir G Kim. Self-supervised learning of point clouds via orientation estimation. *arXiv preprint arXiv:2008.00305*, 2020.

[28] Jonathan Sauder and Bjarne Sievers. Self-supervised deep learning on point clouds by reconstructing space. *arXiv preprint arXiv:1901.08396*, 2019.

[29] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–215, 2018.

[30] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[32] Tianhang Zheng, Changyou Chen, Junsong Yuan, Bo Li, and Kui Ren. Pointcloud saliency maps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1598–1606, 2019.

[33] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[34] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*, pages 69–84. Springer, 2016.

[35] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pages 40–49. PMLR, 2018.

[36] Gunilla Borgefors. Distance transformations in arbitrary dimensions. *Computer vision, graphics, and image processing*, 27(3):321–345, 1984.

[37] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020.

[38] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.

[39] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L Yuille, and Quoc V Le. Adversarial examples improve image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 819–828, 2020.

[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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[41] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1588–1597, 2019.

[42] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016.

[43] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[44] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[45] NVIDIA V100 TENSOR CORE GPU. https://www.nvidia.com/en-us/data-center/v100/, 2020.

[46] Anh Bui, Trung Le, He Zhao, Paul Montague, Olivier deVel, Tamas Abraham, and Dinh Phung. Improving adversarial robustness by enforcing local and global compactness. *arXiv preprint arXiv:2007.05123*, 2020.

[47] Prasad Chalasani, Jiefeng Chen, Amrita Roy Chowdhury, Somesh Jha, and Xi Wu. Concise explanations of neural networks using adversarial training. *arXiv*, pages arXiv–1810, 2018.

[48] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[49] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International conference on machine learning*, pages 7354–7363. PMLR, 2019.

[50] Chong Xiang, Arjun Nitin Bhagoji, Vikash Sehwag, and Prateek Mittal. Patchguard: Provable defense against adversarial patches using masks on small receptive fields. *arXiv preprint arXiv:2005.10884*, 2020.

[51] Cihang Xie and Alan Yuille. Intriguing properties of adversarial training at scale. In *International Conference on Learning Representations*, 2020.

[52] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning*, pages 2206–2216. PMLR, 2020.

[53] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Discovering adversarial examples with momentum. *arXiv preprint arXiv:1710.06081*, 2017.

[54] Shuran Song and Jianxiong Xiao. Deep Sliding Shapes for amodal 3D object detection in RGB-D images. In *CVPR*, 2016.

[55] Lyne P. Tchapmi, Christopher B. Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese. Segcloud: Semantic segmentation of 3d point clouds. In *International Conference on 3D Vision (3DV)*, 2017.

[56] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. Tangent convolutions for dense prediction in 3d. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3887–3896, 2018.

[57] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6411–6420, 2019.

[58] 3D Point Cloud Classification Benchmark on ModelNet40. https://paperswithcode.com/sota/3d-point-cloud-classification-on-modelnet40, 2021.

[59] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1920–1929, 2019.

[60] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pages 609–616, 2009.

[61] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015.

[62] Dumitru Erhan, Aaron Courville, Yoshua Bengio, and Pascal Vincent. Why does unsupervised pre-training help deep learning? In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 201–208. JMLR Workshop and Conference Proceedings, 2010.

[63] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6707–6717, 2020.

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

[65] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018.

[66] 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.

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

[68] Chenglin Yang, Adam Kortylewski, Cihang Xie, Yinzhi Cao, and Alan Yuille. Patchattack: A black-box texture-based attack with reinforcement learning. In *European Conference on Computer Vision*, pages 681–698. Springer, 2020.

[69] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. Adversarial examples for semantic segmentation and object detection. In *International Conference on Computer Vision*. IEEE, 2017.

[70] Lifeng Huang, Chengying Gao, Yuyin Zhou, Cihang Xie, Alan Yuille, Changqing Zou, and Ning Liu. Universal physical camouflage attacks on object detectors, 2019.

[71] Lifeng Huang, Chengying Gao, Yuyin Zhou, Cihang Xie, Alan L Yuille, Changqing Zou, and Ning Liu. Universal physical camouflage attacks on object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 720–729, 2020.

[72] Yuxin Wen, Jiehong Lin, Ke Chen, and Kui Jia. Geometry-aware generation of adversarial and cooperative point clouds. 2019.

[73] Abdullah Hamdi, Sara Rojas, Ali Thabet, and Bernard Ghanem. Advpc: Transferable adversarial perturbations on 3d point clouds. In *European Conference on Computer Vision*, pages 241–257. Springer, 2020.

[74] Yuzhe Yang, Guo Zhang, Dina Katabi, and Zhi Xu. Me-net: Towards effective adversarial robustness with matrix estimation. *arXiv preprint arXiv:1905.11971*, 2019.

[75] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.

[76] Mitali Bafna, Jack Murtagh, and Nikhil Vyas. Thwarting adversarial examples: An l_0-robustsparse fourier transform. *arXiv preprint arXiv:1812.05013*, 2018.

[77] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE symposium on security and privacy (SP)*, pages 582–597. IEEE, 2016.

[78] Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 135–147, 2017.

[79] Daniel Liu, Ronald Yu, and Hao Su. Extending adversarial attacks and defenses to deep 3d point cloud classifiers. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 2279–2283. IEEE, 2019.

[80] Guneet S Dhillon, Kamyar Azizzadenesheli, Zachary C Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Anima Anandkumar. Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*, 2018.

[81] Nicolas Papernot and Patrick McDaniel. Extending defensive distillation. *arXiv preprint arXiv:1705.05264*, 2017.

[82] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *arXiv preprint arXiv:2002.08347*, 2020.

[83] Hongbin Liu, Jinyuan Jia, and Neil Zhenqiang Gong. Pointguard: Provably robust 3d point cloud classification. *arXiv preprint arXiv:2103.03046*, 2021.

[84] Tobias Lorenz, Anian Ruoss, Mislav Balunović, Gagandeep Singh, and Martin Vechev. Robustness certification for point cloud models. *arXiv preprint arXiv:2103.16652*, 2021.

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

[86] Cihang Xie, Mingxing Tan, Boqing Gong, Alan Yuille, and Quoc V Le. Smooth adversarial training. *arXiv preprint arXiv:2006.14536*, 2020.

[87] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, pages 7472–7482. PMLR, 2019.

[88] Ahmadreza Jeddi, Mohammad Javad Shafiee, and Alexander Wong. A simple fine-tuning is all you need: Towards robust deep learning via adversarial fine-tuning. *arXiv preprint arXiv:2012.13628*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]

(b) Did you describe the limitations of your work? [Yes] Please refer to §5. We explicitly mention that we provide empirical robustness without theoretical guarantees.

(c) Did you discuss any potential negative societal impacts of your work? [N/A] We believe there is no potential negative societal impacts of this work.

(d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...

(a) Did you state the full set of assumptions of all theoretical results? [N/A]

(b) Did you include complete proofs of all theoretical results? [N/A]

3. If you ran experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Please refer to §3.2 and our supplements.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Please refer to Figure 3 and our supplements for Table 2's results.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Please refer to §3.2.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes]

(b) Did you mention the license of the assets? [Yes] We mention the licenses in the supplements.

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] We provide information in the supplements.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] We believe there is no personally identifiable information or offensive content in the used datasets.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]