On Adversarial Robustness of 3D Point Cloud Classification under Adaptive Attacks

Jiachen Sun[1]
jiachens@umich.edu
Karl Koenig[1]
kamako@umich.edu
Yulong Cao[1]
yulongc@umich.edu
Qi Alfred Chen[2]
alfchen@uci.edu
Z. Morley Mao[1]
zmao@umich.edu

[1] Computer Science & Engineering
University of Michigan
Ann Arbor, MI, USA

[2] Department of Computer Science
UC Irvine
Irvine, CA, USA

Abstract

3D point clouds are playing pivotal roles in many safety-critical applications like autonomous driving, where adversarially robust 3D deep learning models are desired. In this study, we conduct the first security analysis of state-of-the-art (SOTA) defenses against 3D adversarial attacks and design adaptive evaluations on them. Our **100%** adaptive attack success rates demonstrate that SOTA countermeasures are still fragile. We further present an in-depth study showing how adversarial training (AT) performs in point cloud classification and identify that the required symmetric function (pooling operation) is paramount to 3D models' robustness. Through systematic analysis, we unveil that the default-used fixed pooling (*e.g.,* `MAX` pooling) generally weakens AT's effectiveness. Interestingly, we also discover that *sorting-based* parametric pooling significantly improves the models' robustness. Based on the above insights, we propose `DeepSym`, a deep symmetric pooling operation, to architecturally advance the robustness of PointNet to **47.0%** under AT without sacrificing nominal accuracy, outperforming the original design and a strong baseline by **+28.5%** ($\sim 2.6\times$) and **+6.5%**, respectively.

1 Introduction

Despite the prominent achievements that deep neural networks (DNN) have reached in the past decade, adversarial attacks [41] are becoming the Achilles' heel in modern deep learning deployments, where adversaries generate imperceptible perturbations to mislead the DNN models. The emergence of 3D point cloud applications in safety-critical areas like autonomous driving raises public concerns about the security of their DNN pipelines. Among them, classification is an essential and fundamental task on point clouds. While it seems intuitive to extend convolutional neural networks (CNN) from 2D to 3D for point cloud classification, it is in fact, not a trivial task. The difficulty mainly inherits from that point cloud is a *sparse* and *unordered* set structure that CNN cannot handle. Pioneering point

cloud recognition models [34, 51] address this problem by leveraging a **symmetric function**, which is *permutation-invariant* to the order of points, to aggregate local features, as illustrated in Figure 1. Such a primitive has been universally adopted in many other complex learning tasks like semantic segmentation and object detection [21, 59]. In this paper, we present a systematic study to analyze and enhance its robustness against adversarial attacks.

Our key contributions are ***three-fold*** and summarized below:

• **Adaptive Attacks**. Recent efforts [3, 40, 52] have demonstrated that various deep point cloud models are vulnerable to adversarial attacks, and a few countermeasures have been lately proposed. However, the failure of gradient obfuscation-based defenses in 2D vision tasks motivates us to re-think whether current defense designs provide *true* robustness [43] for 3D point cloud classification. Especially, DUP-Net [63] and GvG-PointNet++ [6] have claimed significant improvements in adversarial robustness. However, we find that both defenses belong to gradient obfuscation through our analysis, hence further design adaptive attacks to break their robustness. Our **100%** attack success rates show that current defense strategies can still be circumvented by adaptive adversaries.

• **Adversarial Training Analysis**. It is widely acknowledged that adversarial training (AT) [27] is a more long-standing defense [5]. We thus perform the *first* rigorous study of AT in point cloud classification to further improve its robustness. Through systematic analysis, we identify that the default used **symmetric function** bottlenecks the effectiveness of AT. Specifically, popular models (*e.g.,* PointNet) utilize fixed pooling operations like `MAX` and `SUM` pooling as their symmetric functions to aggregate features. Different from CNN-based models that usually apply pooling operations with a small sliding window (*e.g.,* 2×2), point cloud classification models leverage pooling operations to aggregate features from a large number of candidates (*e.g.,* 1024). We find that those fixed pooling operations inherently lack *smoothness* and *learnability*, which AT does not favor. Moreover, recent research has presented parametric pooling operations in set learning [48, 52], which also preserve permutation-invariance. We take a step further to systematically study their impacts in models' robustness under AT. Experimental results show that the *sorting-based* pooling design benefits AT well, which outperforms `MAX` pooling, for instance, in adversarial accuracy by +7.3% while maintaining similar nominal accuracy[1].

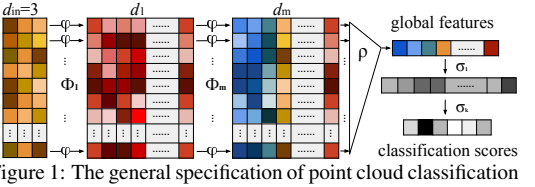


Figure 1: The general specification of point cloud classification $(\sigma \circ \rho \circ \Phi)(\mathbb{X})$, where n is the number of points, d_i is the number of hidden dimensions in the i-th feature map, Φ represents the permutation-equivariant layers, ρ denotes the column-wise symmetric function, and σ is the fully connected layer.

• **Architectural Improvement**. Based on our experimental insights, we further propose `DeepSym`, a sorting-based pooling operation that employs deep learnable layers, to architecturally advance the adversarial robustness of point cloud classification under AT. `DeepSym` is intrinsically flexible and general by design. Experimental results show that `DeepSym` reaches the highest adversarial accuracy in all chosen backbones, which on average, is a +10.8% improvement compared to the default architectures. We also explore the limits of `DeepSym` based on PointNet due to its broad adoption in multiple 3D vision tasks [11]. We obtain the best robustness on ModelNet40, which achieves the adversarial accuracy of 47.0%, significantly outperforming the default `MAX` pooling design by +28.5% ($\sim 2.6\times$). We

[1] In this paper, we use nominal and adversarial accuracy to denote the model's accuracy on clean and adversarially perturbed data, respectively.

demonstrate that PointNet with `DeepSym` also reaches the highest adversarial accuracy of 45.2% under the most efficient AT on ModelNet10 [51], exceeding `MAX` pooling by +17.9% ($\sim 1.7\times$).

2 Background and Related Work

3D Point Cloud Classification. Early works attempt to classify point clouds by adapting deep learning models in the 2D space [29, 38, 39, 42, 47, 60]. PointCNN [24] tries to address the unorderness problem by learning a permutation matrix, which is, however, still non-deterministic. DeepSets [51] and PointNet [34] pioneer to achieve end-to-end learning on point cloud classification and formulate a general specification (Figure 1) for point cloud learning. PointNet++ [35] and DGCNN [49] build upon PointNet set abstraction to better learn local features by exploiting k-nearest neighbors. Lately, DSS [28] generalizes DeepSets to enable complex functions in set learning. Besides, ModelNet40 [51] is the most popular dataset for benchmarking point cloud classification, which consists of 12,311 CAD models belonging to 40 categories. Their numerical range is normalized to $[-1,1]$.

Adversarial Attacks and Defenses on Point Clouds. Numerous attacks have been widely studied for various tasks in 2D [4, 13, 20, 27, 53, 55, 57], including projected gradient descent (PGD) [27] and C&W attacks [4]. Xiang *et al.* [52] perform the first study to extend C&W attack [4] to 3D point cloud classification. Wen *et al.* [50] improve the loss function in C&W attack to realize attacks with smaller perturbations and [12] present black-box attacks on point cloud classification. Recently, [63] and [6] propose to defend against adversarial point clouds by input transformation and adversarial detection. Besides, [25] conduct a preliminary investigation on extending countermeasures in the 2D space to defend against naïve attacks like FGSM [10] on point cloud data. Liu *et al.* [26] propose to certify the robustness of point cloud recognition with a threat model only considering the number of modified points. In this work, we first design adaptive attacks to break existing defenses and analyze the adversarial robustness of point cloud classification under adversarial training constrained by widely recognized L^p norms.

3 Breaking SOTA Point Cloud Defenses

3.1 Adaptive Attacks on DUP-Net

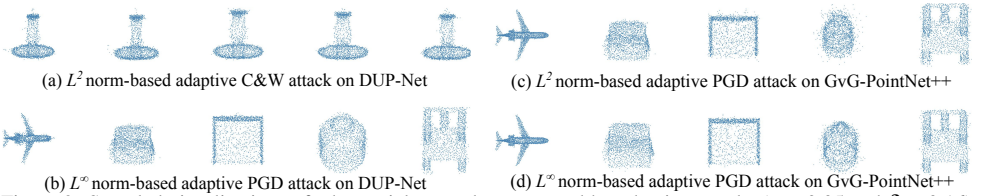
DUP-Net [63] (ICCV'19) presents a denoiser layer and upsampler network structure to defend against adversarial point cloud classification. The denoiser layer $g : \mathbb{X} \to \mathbb{X}'$ leverages kNN (k-nearest neighbor) for outlier removal. Specifically, the kNN of each point $\boldsymbol{x}_i$ in point cloud $\mathbb{X}$ is defined as $knn(\boldsymbol{x}_i, k)$ so that the average distance d_i of each point $\boldsymbol{x}_i$ to its kNN is denoted as:

$$d_i = \frac{1}{k} \sum_{\boldsymbol{x}_j \in knn(\boldsymbol{x}_i,k)} ||\boldsymbol{x}_i - \boldsymbol{x}_j||_2 \ , \ i = \{1,2,\ldots,n\} \tag{1}$$

where n is the number of points. The mean $\mu = \frac{1}{n}\sum_{i=1}^{n} d_i$ and standard deviation $\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(d_i-\mu)^2}$ of all these distances are computed to determine a distance threshold as $\mu + \alpha \cdot \sigma$ to trim the point clouds, where α is a hyper-parameter. As a result, the denoised

Table 1: Adversarial accuracy under adaptive attacks on PU-Net and DUP-Net. For the denoiser layer g, $k = 2$ and $\alpha = 1.1$ are set the same as [63]. † denotes the attack setups evaluated in the original DUP-Net paper [63].

Attack Method	Adversarial Accuracy			Mean L^2 Norm Distance
	PointNet (f)	PU-Net ($f \circ p$)	DUP-Net ($f \circ p \circ g$)	
Clean point cloud	88.3%	87.5%	86.3%	0.0
C&W attack on f †	**0.0%**	23.9%	84.5%	0.77
C&W attack on $f \circ p$	2.3%	**0.0%**	74.7%	0.71
Adaptive attack on $f \circ p \circ g$	1.1%	0.8%	**0.0%**	1.62
PGD attack ($\varepsilon = 0.01$)	7.1%	5.9%	5.4%	-
PGD attack ($\varepsilon = 0.025$)	3.5%	2.8%	2.1%	-
PGD attack ($\varepsilon = 0.05$)	1.3%	1.0%	0.8%	-
PGD attack ($\varepsilon = 0.075$)	**0.0%**	**0.0%**	**0.0%**	-



(a) L^2 norm-based adaptive C&W attack on DUP-Net (c) L^2 norm-based adaptive PGD attack on GvG-PointNet++

(b) L^∞ norm-based adaptive PGD attack on DUP-Net (d) L^∞ norm-based adaptive PGD attack on GvG-PointNet++

Figure 2: Sampled visualizations of adversarial examples generated by adaptive attacks ($\varepsilon = 0.05$ and $\delta = 0.16$). More visualizations can be found in Appendix A.2.

point cloud is represented as $\mathbb{X}' = \{\boldsymbol{x}_i \mid d_i < \mu + \alpha \cdot \sigma\}$. The denoised point cloud $\mathbb{X}'$ will be further fed into PU-Net [59], defined as $p : \mathbb{X}' \to \mathbb{X}''$, to upsample $\mathbb{X}'$ to a fixed number of points. Combined with the base classifier f, the integrated DUP-Net can be noted as $(f \circ p \circ g)(\mathbb{X})$. The underlying hypothesis is that g will eliminate the adversarial perturbations and p will re-project the denoised off-manifold point cloud to the natural manifold.

Analysis. The upsampler network p (*i.e.,* PU-Net) is differentiable and can be integrated with the classification network f. Therefore, we find $f \circ p$ is vulnerable to gradient-based adaptive attacks. Although the denoiser layer g is not differentiable, we treat it as a *deterministic masking*: $\mathcal{M}(\boldsymbol{x}_i) = \mathbf{1}_{d_i < \mu + \alpha \cdot \sigma}$ so that the approximated gradients (BPDA) [1] can still flow through the masked points. By involving $\mathcal{M}(\boldsymbol{x}_i)$ into the iterative adversarial optimization process, we are able to find adversarial examples with high fidelity.

$$\nabla_{\boldsymbol{x}_i}(f \circ p \circ g)(\mathbb{X})|_{\boldsymbol{x}_i = \hat{\boldsymbol{x}}} \approx \nabla_{\boldsymbol{x}_i}(f \circ p)(\mathbb{X})|_{\boldsymbol{x}_i = \hat{\boldsymbol{x}} \cdot \mathcal{M}(\hat{\boldsymbol{x}})} \tag{2}$$

Experimentation. We leverage the official codebase[2] of DUP-Net for experimentation. Specifically, a PointNet [34] trained on ModelNet40 is used as the classifier f. For the PU-Net, the upsampled number of points is 2048, and the upsampling ratio is 2. For the adaptive attacks, we exploit targeted L^2 norm-based C&W attack and untargeted L^∞ norm-based PGD attack with 200 iterations (PGD-200). Detailed setups are elaborated in Appendix A.1.

Discussion. As presented in Table 1, adaptive C&W attacks achieve 100% success rates on DUP-Net. Though the mean distance of adversarial examples targeting DUP-Net is larger than those targeting PU-Net, they are almost indistinguishable, as visualized in Appendix A.2. We find that naïve PGD attacks are also effective since the upsampler network is sensitive to L^∞ norm-based perturbations, which also showcase the fragility of the defense pipeline. The design of DUP-Net is similar to ME-Net [58] in the 2D space, which has been shown vulnerable to adaptive attacks [43]. We demonstrate that such input transformation-based defenses cannot offer true robustness to point cloud classification, either.

3.2 Adaptive Attacks on GvG-PointNet++

GvG-PointNet++ [6] (CVPR'20) introduces gather vectors in the 3D point clouds as an adversarial indicator to detect adversarial perturbations. The original PointNet++ [35] aggregates local features $\boldsymbol{f}_i$ hierarchically to make final classification. Gather vectors $\boldsymbol{v}_i$ are learned from local features $\boldsymbol{f}_i$ to indicate the global center $\boldsymbol{c}_i$ of a point cloud sample. If the indicated global center $\boldsymbol{c}_i$ is far away from the ground-truth global center $\boldsymbol{c}_g$, the corresponding local feature $\boldsymbol{f}_i$ will be masked out:

$$\boldsymbol{c}_i = \boldsymbol{x}_{ci} + \boldsymbol{v}_i \; ; \quad \mathcal{M}_i = \mathbf{1}_{||\boldsymbol{c}_g - \boldsymbol{c}_i|| < r} \; ; \quad \mathbb{F}_g = \{\boldsymbol{f}_i \cdot \mathcal{M}_i\} \tag{3}$$

where $\boldsymbol{x}_{ci}$ is the geometry center of the local point set, r is the distance threshold to mask the local feature, and $\mathbb{F}_g$ is the cleaned feature set for final classification. To train GvG-PointNet++, it is necessary to optimize an auxiliary loss to correctly learn the gather vectors besides the default cross-entropy (xent) loss:

[2] https://github.com/RyanHangZhou/DUP-Net

Table 2: Adversarial accuracy under L^p norm-based adaptive attacks on GvG-PointNet++. ε and δ are the perturbation boundaries. † denotes the attack setups evaluated in the original GvG-PointNet++ paper [6].

Target Loss	Adversarial Accuracy (L^∞)				Adversarial Accuracy (L^2)		
	$\varepsilon = 0.01$	$\varepsilon = 0.025$	$\varepsilon = 0.05$	$\varepsilon = 0.075$	$\delta = 0.08$	$\delta = 0.16$	$\delta = 0.32$
$\mathcal{L}_{xent}$†	30.6%	21.4%	5.4%	1.8%	25.2%	16.9%	15.4%
$\mathcal{L}_{adv}$	20.1%	12.6%	2.2%	**0.0%**	**7.5%**	4.4%	**2.1%**
$\mathcal{L}_{gather}$	**17.9%**	**8.1%**	**0.0%**	**0.0%**	8.5%	**4.1%**	2.7%

Table 3: Adversarial robustness of models with fixed pooling operations under PGD-200 at $\varepsilon = 0.05$.

Pooling Operation	Nominal Accuracy			Adversarial Accuracy		
	PointNet	DeepSets	DSS	PointNet	DeepSets	DSS
MAX	80.5%	71.1%	78.8%	16.1%	21.8%	21.5%
SUM	76.3%	54.1%	73.3%	**25.1%**	**24.8%**	**25.3%**
MEDIAN	**84.6%**	**72.7%**	**82.4%**	7.5%	11.0%	9.3%

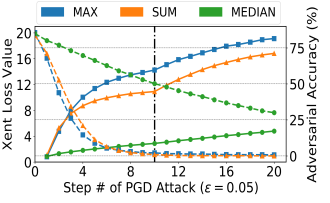


Figure 3: Cross-entropy (xent) loss and adversarial accuracy of PGD attack on PointNet with three fixed pooling operations (each data point is averaged over 100 runs).

$$\mathcal{L}_{total} = \mathcal{L}_{xent} + \lambda \cdot \mathcal{L}_{gather}, \quad \mathcal{L}_{gather} = \sum_{i=1}^{n'} ||\boldsymbol{c}_i - \boldsymbol{c}_g||_1 \tag{4}$$

where n' is the number of local features and λ is a hyper-parameter.

Analysis. Dong *et al.* [6] evaluate white-box adversaries on GvG-PointNet++ with naïve L^2 norm-based PGD attacks. Specifically, only $\mathcal{L}_{xent}$ is utilized as the objective loss in the adversarial optimization process so that the masking $\mathcal{M}_i$ will hinder the gradient propagation. However, since $\mathcal{M}_i$ is learned from the network itself, it is highly possible to further break this obfuscation with $\mathcal{L}_{gather}$ considered. We thus formulate the first adaptive attack as an optimization problem with the objective function:

$$\mathcal{L}_{adv} = \mathcal{L}_{xent} - \beta \cdot \mathcal{L}_{gather} \tag{5}$$

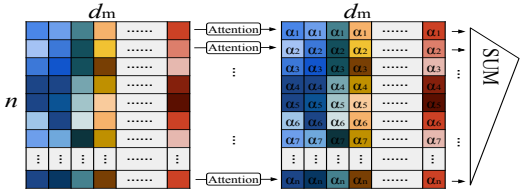
where β is a tunable hyper-parameter. By maximizing $\mathcal{L}_{adv}$ with L^2 norm-based PGD attacks, adversaries not only strive to enlarge the adversarial effect but also minimize the perturbations on gather vectors. We also design the second attack to make PGD attacks *only* target $\mathcal{L}_{gather}$. Such perturbations will potentially affect most gather vector predictions to make $\boldsymbol{v}_i$ masked out so that the rest features are insufficient for the final classification.

Experimentation. We train GvG-PointNet++ based on single-scale grouped PointNet++ [35] on ModelNet40 and set $r = 0.08$ and $\lambda = 1$ as suggested by [6]. The model is trained by Adam [19] optimizer with 250 epochs using batch size = 16, and the initial learning rate is 0.01. For the adaptive attack, we use 10-step binary search to find a appropriate β. The setup of L^2 norm-based PGD attacks is identical to [6], and we also leverage L^∞ norm-based PGD-200 in the evaluation. Detailed setups are elaborated in Appendix A.1.

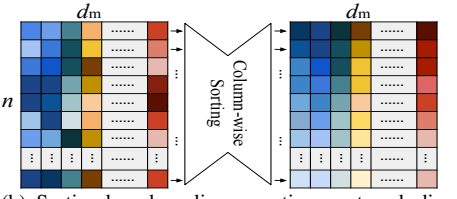
Discussion. As presented in Table 2, both adaptive PGD attacks achieve high success rates on GvG-PointNet++. We also observe that the L^∞ norm-based PGD attack is more effective on $\mathcal{L}_{gather}$ since L^∞ norm perturbations assign the same adversarial budget to each point, which can easily influence a large number of gather vector predictions. However, it is hard for the L^2 norm-based PGD attack to affect so many gather vector predictions because it prefers to perturb key points (*i.e.*. points with larger gradients) rather than the whole point set. GvG-PointNet++ leverages DNN to detect adversarial perturbations, which is similar to MagNet [30] in the 2D space. We validate that adversarial detection also fails to provide true robustness under adaptive adversaries in point cloud classification.

4 Adversarial Training in Point Cloud Classification

We have so far demonstrated that SOTA defenses against adversarial point clouds are still vulnerable to adaptive attacks. In this section, we conduct the first thorough study showing how adversarial training (AT) performs in point cloud classification.



(a) Attention-based pooling operations apply self-attention to each point-level feature vector $\boldsymbol{f}_i$. The learned weight α_i is multiplied with each element in $\boldsymbol{f}_i$, and the aggregated feature is computed by a column-wise summation.



(b) Sorting-based pooling operations sort each dimension to re-organize the feature set into an ordered matrix $\widetilde{\boldsymbol{F}}$ to which complex operations (*e.g.*, CNN) can be applied to aggregate features.

Figure 4: Design philosophy of attention-based and sorting-based pooling operations.

4.1 Adversarial Training Preliminaries

Madry *et al*. [27] formulate AT as a saddle point problem in Equation 6, where $\mathcal{D}$ is the underlying data distribution, $\mathcal{L}(\cdot,\cdot,\cdot)$ is the loss function, $\boldsymbol{x}$ is the training data with its label $\boldsymbol{y}$, $\boldsymbol{\varepsilon}$ is the adversarial perturbation, and $\mathbb{S}$ denotes the boundary of such perturbations.

$$\arg\min_{\boldsymbol{\theta}} \quad \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}}\left[\max_{\boldsymbol{\varepsilon}\in\mathbb{S}} \mathcal{L}(\boldsymbol{x}+\boldsymbol{\varepsilon},\boldsymbol{y},\boldsymbol{\theta})\right] \tag{6}$$

Adversarial Training Setups. We choose PointNet [34][3] as the primary backbone due to its extensive adoption in various 3D learning tasks [21, 36]. We also select DeepSets [61] and DSS [28] to show the generality of our analysis. As illustrated in Section 3 and demonstrated by [27], L^∞ norm-based PGD attack tends to be a universal first-order adversary. Thus, we select PGD-7 into the training recipe, and empirically set the maximum per-point perturbation $\varepsilon = 0.05$ out of the point cloud range $[-1,1]$. We follow the default PointNet training setting[3] to train all models on the ModelNet40 training set. In the evaluation phase, we utilize PGD-200 to assess their robustness on the ModelNet40 validation set with the same adversarial budget $\varepsilon = 0.05$. Meanwhile, we also report the nominal accuracy on the clean validation set. Each PGD attack starts from a random point in the allowed perturbation space. More details can be found in Appendix B due to space constraints.

4.2 AT Meets Fixed Pooling Operations

As illustrated in Figure 1, point cloud classification models fundamentally follow a general specification $(\sigma\circ\rho\circ\Phi)(\mathbb{X})$. Φ represents a set of permutation-equivariant layers to learn local features from each point. ρ is a column-wise symmetric function to aggregate the learned local features into a global feature, and σ are fully connected layers for final classification. PointNet, DeepSets, and DSS leverage different Φ for local feature learning, but all depend on **fixed pooling operations** as their ρ. Specifically, `MAX` pooling is by default used in DeepSets [61] (for point cloud classification) and PointNet [34], while DSS utilizes `SUM` pooling [28]. We also additionally select `MEDIAN` pooling due to its robust statistic feature [14]. Though models with fixed pooling operations have achieved satisfactory accuracy under standard training, they face various difficulties under AT. As presented in Table 3, models with `MEDIAN` pooling achieve better nominal accuracy among fixed pooling operations, but much worse adversarial accuracy, while `SUM` pooling performs contrarily. Most importantly, none of them reaches a decent balance between nominal and adversarial accuracy.

Insights. AT consists of two stages: 1) *inner maximization* to find the worst adversarial examples and 2) *outer minimization* to update model parameters. Fixed pooling operations essentially leverage a *single* statistic to represent the distribution of a feature dimension [31].

[3] https://github.com/charlesq34/pointnet

Table 4: Adversarial robustness of models with parametric pooling operations under PGD-200 at $\varepsilon = 0.05$.

Pooling Operation	Nominal Accuracy			Adversarial Accuracy		
	PointNet	DeepSets	DSS	PointNet	DeepSets	DSS
ATT	73.5%	52.3%	72.8%	22.1%	23.2%	23.9%
ATT-GATE	75.1%	63.9%	73.3%	23.2%	24.8%	26.1%
PMA	73.9%	51.9%	72.5%	25.4%	20.9%	23.9%
FSPool	**82.8%**	73.8%	81.5%	29.8%	25.3%	26.1%
SoftPool	79.8%	72.1%	80.2%	30.1%	24.9%	26.5%
DeepSym (ours)	82.7%	**74.2%**	**81.6%**	**33.6%**	**26.9%**	**31.4%**

Table 5: Adversarial robustness of PointNet with different pooling operations under attacks at $\varepsilon = 0.05$.

Pooling Operation	White-box Attack			Black-box Attack		
	FGSM	BIM	MIM	SPSA	NES	Evolution
MAX	72.8%	24.3%	23.5%	69.2%	67.1%	53.4%
MEDIAN	**77.6%**	23.3%	14.5%	71.1%	65.2%	57.8%
SUM	44.4%	33.5%	37.5%	65.3%	62.3%	52.7%
ATT	43.1%	33.1%	35.0%	68.1%	64.8%	55.9%
ATT-GATE	43.9%	34.2%	33.9%	70.2%	65.9%	55.8%
PMA	47.2%	31.9%	30.1%	67.2%	64.1%	53.4%
FSPool	61.3%	45.4%	48.0%	**72.8%**	71.9%	69.9%
SoftPool	62.1%	47.6%	45.1%	69.2%	68.5%	70.0%
DeepSym (ours)	61.4%	**52.5%**	**55.4%**	72.4%	**72.1%**	**73.1%**

Although MEDIAN pooling, as a robust statistic, intuitively should enhance the robustness, we find it actually hinders the inner maximization stage from making progress. We utilize L^∞ norm-based PGD attack to maximize the xent loss of standard trained model with three fixed pooling operations. Figure 3 validates that MEDIAN pooling struggles to maximize the loss, so that it fails to find the worst adversarial examples in the first stage with limited steps. Though MAX and SUM pooling are able to achieve higher loss value, they encounter challenges in the second stage. MAX pooling backward propagates gradients to a *single* point at each dimension so that the rest $\frac{n-1}{n}$ features do not contribute to model learning. Since n is oftentimes a large number (*e.g.,* 1024), the huge information loss and non-smoothness will fail AT [56]. While SUM pooling realizes a smoother backward propagation, it lacks discriminability because by applying the same weight to each element, the resulting representations are strongly biased by the adversarial perturbations. Thus, with SUM pooling, the models cannot generalize well on clean data (Table 3).

4.3 AT Meets Parametric Pooling Operations

Recent studies have also presented trainable **parametric pooling operations** for different tasks in set learning, which are also qualified to be the symmetric function ρ. We further group them into two classes: 1) *attention-based* and 2) *sorting-based* pooling. It is worth noting we are the *first* to analyze their robustness in point cloud classification models.

Attention-based Pooling Operations. An attention module can be abstracted as mapping a query and a set of key-value pairs to an output, making the models learn and focus on the critical information [2]. Figure 4(a) shows the one example of attention-based pooling, ATT [15], which leverages a compatibility function to learn point-level importance. The aggregated global feature is computed as a column-wise weighted sum of the local features. Let $\mathbb{F} = \{\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_n\}$ be a set of features, ATT aggregates the global feature $\boldsymbol{g}$ by:

$$\boldsymbol{g} = \sum_{i=1}^{n} a_i \cdot \boldsymbol{f}_i\,, \quad a_i = \frac{\exp(\boldsymbol{w}^\top \cdot \tanh(\boldsymbol{V} \cdot \boldsymbol{f}_i^\top))}{\sum_{j=1}^{n} \exp(\boldsymbol{w}^\top \cdot \tanh(\boldsymbol{V} \cdot \boldsymbol{f}_j^\top))} \tag{7}$$

where $\boldsymbol{w} \in \mathbb{R}^{L\times 1}$ and $\boldsymbol{V} \in \mathbb{R}^{L\times d_m}$ are learnable parameters. We leverage ATT, ATT-GATE [15], and PMA [22] in our study and detail their design and our implementation in Appendix B.3.

Sorting-based Pooling Operations. Sorting has been recently considered in the set learning literature due to its permutation-invariant characteristic, as shown in Figure 4(b). For example, let $\boldsymbol{F} \in \mathbb{R}^{n\times d_m}$ be the matrix version of the feature set $\mathbb{F}$, FSPool [62] aggregates $\boldsymbol{F}$ by feature-wise sorting in a descending order:

$$\widetilde{\boldsymbol{F}}_{i,j} = sort_{\downarrow}(\boldsymbol{F}_{:,j})_i\,; \quad g_j = \sum_{i=1}^{n} \boldsymbol{W}_{i,j} \cdot \widetilde{\boldsymbol{F}}_{i,j} \tag{8}$$

where $\boldsymbol{W} \in \mathbb{R}^{n\times d_m}$ are learnable parameters. Therefore, the pooled representation is column-wise weighted sum of $\widetilde{\boldsymbol{F}}$. We also utilize SoftPool [48] in our study, and its implementation details are elaborated in Appendix B.3 due to space constraints.

4.3.1 Experimental Analysis

Table 4 summarizes the results of AT with different parametric pooling operations. To meet the requirement of permutation-invariance, attention-based pooling is restricted to learn *point-level* importance. For example, `ATT` applies the same weight to all dimensions of a point embedding. As a result, attention barely improves the pooling operation's expressiveness as it essentially re-projects the point cloud to a single dimension (*e.g.,* $\boldsymbol{f}_i \rightarrow a_i$ in `ATT`) and differentiates them based on it, which limits their discriminability. Therefore, little useful information can be learned from the attention module, explaining their similar performance with `SUM` pooling that applies the same weight to each point under AT, as presented in Table 4. On the other hand, sorting-based pooling operations naturally maintain permutation-invariance as $sort_{\downarrow}(\cdot)$ re-organizes the unordered feature set $\mathbb{F}$ to an ordered feature map $\widetilde{\boldsymbol{F}}$. Thus, `FSPool` and `SoftPool` are able to further apply *feature-wise* linear transformation and CNN. The key insight is that feature dimensions are mostly independent of each other, and each point expresses to a different extent in every dimension. By employing feature-wise learnable parameters, the gradients also flow smoother through sorting-based pooling operations. Table 4 validates that sorting-based pooling operations achieve much better adversarial accuracy, *e.g.,* on average, +7.3% better than `MAX` pooling while maintaining comparable nominal accuracy.

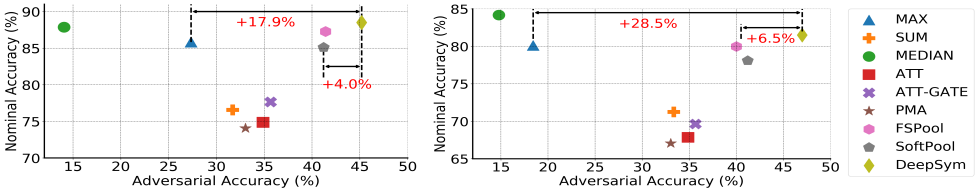
5 Improving Robustness with `DeepSym`

In the above analysis, we have shed light that sorting-based pooling operations benefit AT in point cloud classification. We hereby explore to further improve the sorting-based pooling design based on existing arts. First, we notice that both `FSPool` and `SoftPool` apply $sort_{\downarrow}(\cdot)$ right after a ReLU function [32]. However, ReLU activation leads to half of neurons being zero [9], which will make $sort_{\downarrow}(\cdot)$ unstable. Second, recent studies have shown that AT appreciates deeper neural networks [24]. Nevertheless, `FSPool` only employs one linear layer to aggregate features, and `SoftPool` requires d_m to be a small number. The reason is that scaling up the depth in these existing sorting-based pooling designs requires exponential growth of parameters, which will make the end-to-end training intractable.

To address the above limitations, we propose a simple yet effective pooling operation, `DeepSym`, that embraces the benefits of sorting-based pooling and also applies deep learnable layers to the pooling process. Given a feature set after ReLU activation $\mathbb{F} \in {\mathbb{R}_+}^{n \times d_m}$, `DeepSym` first applies another linear transformation to re-map $\mathbb{F}$ into $\mathbb{R}^{n \times d_m}$ so that $\boldsymbol{f}'_i = \boldsymbol{W} \cdot \boldsymbol{f}_i{}^{\top}$ where $\boldsymbol{W} \in \mathbb{R}^{d_m \times d_m}$ and $\mathbb{F}' = \{\boldsymbol{f}'_1, \boldsymbol{f}'_2, \ldots, \boldsymbol{f}'_n\}$. Let $\boldsymbol{F}'$ be the matrix version of $\mathbb{F}'$, `DeepSym` further sorts $\boldsymbol{F}'$ in a descending order (Equation 8) to get $\widetilde{\boldsymbol{F}}'$. Afterwards, a column-wise shared MLP will be applied on the sorted feature map $\widetilde{\boldsymbol{F}}'$:

$$g_j = \mathrm{MLP}(\widetilde{\boldsymbol{F}}'_{:,j}), \quad j = \{1, 2, \ldots, d_m\} \tag{9}$$

to learn the global feature representation $\boldsymbol{g}$. Each layer of the MLP composes of a linear transformation, a batch normalization [17], and a ReLU activation function. Compared to `FSPool` that applies different linear transformations to different dimensions, `DeepSym` employs a *shared* MLP to different dimensions. By doing so, `DeepSym` deepens the pooling process to be more capable of digesting the adversarial perturbations. `DeepSym` can also address the problem of `SoftPool` that is only achievable with limited d_m because the MLP is shared by all the feature channels so that it can scale up to a large number of d_m with little complexity increases. Moreover, `DeepSym` is intrinsically flexible and general. For



(a) PGD-1 adversarial training on ModelNet10. (b) PGD-20 adversarial training on ModelNet40.

Figure 5: Adversarial robustness of PointNet with various pooling operations under PGD-200 at $\varepsilon = 0.05$.

example, it clearly generalizes `MAX` and `SUM` pooling by specific weight vectors. Therefore, it can also theoretically achieve universality with $d_m \geq n$ [46] while being more expressive in its representation and smoother in gradients propagation. To deal with the variable-size point clouds, `DeepSym` adopts column-wise linear interpolation in $\widetilde{\boldsymbol{F}}'$ to form a continuous feature map and then re-samples it to be compatible with the trained MLP [18]. Last but not least, `DeepSym` is by design flexible with its own number of layers and number of pooled features from each dimension. It is easy to tune the depth and width of `DeepSym` ($O(1)$) across different learning tasks, which is extremely hard for other pooling operations to achieve. For example, it requires $O(n^2)$ complexity for `FSPool` to tune its architectural parameters.

5.1 Evaluations

As `DeepSym` is naturally flexible, we implement a 5-layer `DeepSym` with $[512, 128, 32, 8, 1]$ hidden neurons on three backbone networks and adversarially train them on ModelNet40 the same way introduced in Section 4.1. Table 4 shows that almost all models with `DeepSym` reach the best results in both nominal and adversarial accuracy, outperforming the default architecture by +10.8%, on average. Taking PointNet as an example, `DeepSym` (33.6%) improves the adversarial accuracy by +17.5% ($\sim 2.1\times$) compared to the original `MAX` pooling architecture. Besides, `DeepSym` also achieves a +3.5% improvement in adversarial accuracy compared to `FSPool` and `SoftPool`. Overall, we demonstrate that `DeepSym` can benefit AT significantly in point cloud classification.

We further leverage various white- and black-box adversarial attacks to cross-validate the robustness improvements of `DeepSym` in PointNet. Specifically, we exploit well-known FGSM [41], BIM [20], and MIM [7] as the white-box attack methods. We set the adversarial budget $\varepsilon = 0.05$, and leverage 200 steps for the iterative attacks, as well. For the black-box attacks, we choose two score-based methods: SPSA [45] and NES [16], and a decision-based evolution attack [8]. We still select $\varepsilon = 0.05$ and allow 2000 queries to find each adversarial example. The detailed setups are elaborated in Appendix C.1. As Table 5 shows, PointNet with `DeepSym` consistently achieves the highest adversarial accuracy under white-box attacks, except for FGSM since it is a single-step method that has limited ability to find adversarial examples (Figure 3). Besides, we find the black-box attacks are not as effective as the white-box attacks, which also demonstrate that AT with `DeepSym` improves the adversarial robustness in point cloud classification without gradient obfuscation [5].

5.2 Exploring the Limits of `DeepSym`

There is a trade-off between the training cost and adversarial robustness in AT. Increasing the number of PGD attack steps can create harder adversarial examples [27], which could further improve the model's robustness. However, the training time also increases linearly with the number of attack iterations increasing. Due to PointNet's broad adoption [11], we here analyze how it performs under various AT settings. Specifically, we exploit the most efficient AT with PGD-1 on ModelNet10 [51], a dataset consisting of 10 categories with

4899 objects, and a relatively expensive AT with PGD-20 on ModelNet40 to demonstrate the effectiveness of DeepSym. Other training setups are identical to Section 4.1.

Figure 5 shows the results of the robustness of adversarially trained PointNet with various pooling operations under PGD-200. We demonstrate that PointNet with DeepSym still reaches the highest adversarial accuracy of 45.2% under AT with PGD-1 on ModelNet10, which outperforms the original MAX pooling by +17.9% ($\sim 1.7\times$) and SoftPool by +4.0%. Surprisingly, PointNet with DeepSym also achieves the best nominal accuracy of 88.5%. Moreover, DeepSym further advances itself under AT with PGD-20 on ModelNet40. Figure 5(b) shows that PointNet with DeepSym reaches the highest 47.0% adversarial accuracy, which have +28.5% ($\sim 2.6\times$) and +6.5% improvements compared to MAX pooling and SoftPool, respectively while maintaining competent nominal accuracy. Other ablation studies are presented in Appendix C due to space constraints, which further show the general effectiveness of DeepSym under various training and evaluation settings.

6 Discussion and Conclusion

We have so far demonstrated that DeepSym has helped achieve significant robustness improvement compared to prior arts. we discuss more implications of our study and future research directions.

Universal Robustness. By investigating prior SOTA defenses [6, 63], we find it alarming to claim universal robustness in point cloud classification, which has been well shown in 2D vision tasks [5], as such general defenses may give a false sense of security. In this work, we primarily target defenses against L^∞ norm-based adversaries. Recent studies present sophisticated attacks [44, 64] that are not bounded by formal distances. We expect future defensive studies that provide *true* robustness against general attacks.

Downstream Tasks. While the focus of this work is the adversarial robustness in point cloud classification, DeepSym could be also used in downstream tasks (*e.g.* segmentation [33] and object detection [21, 37]). However, there are no piratical AT strategies for such downstream tasks as their complexities makes AT's computational cost unacceptable. Therefore, it is hard to quantify the effectiveness of DeepSym *w.r.t.* the robustness in those tasks. We plan to bridge the gap between AT and 3D point cloud downstream tasks in our future work.

Set Learning. Point cloud recognition is a particular case of set learning [23], where set elements themselves adhere to their own symmetries [28]. DeepSym is a general pooling design that fits set learning as well. We hope our provided findings and insights will encourage more research on the adversarial robustness of set learning.

To conclude, in this work, we perform the *first* rigorous study on the adversarial robustness of point cloud classification. We design adaptive attacks and demonstrate that SOTA defenses fail to provide true robustness. Moreover, we conduct a thorough analysis of how the required symmetric function affects the AT performance in point cloud classification. We are the first to identify that the fixed pooling generally weakens the models' robustness under AT, and on the other hand, *sorting-based* parametric pooling benefits AT well. Lastly, we propose DeepSym that further architecturally advances the adversarial accuracy of PointNet to 47.0% under AT, outperforming the original design and a strong baseline by +28.5% ($\sim 2.6\times$) and +6.5%.

7 Acknowledgements

We thank our area chairs and anonymous reviewers for their insightful comments. This project is partially supported by NSF grants CMMI-2038215, CNS-1930041, CNS-1932464, and CNS-1929771.

References

[1] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 274–283, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL http://proceedings.mlr.press/v80/athalye18a.html.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] 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.

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. *2017 IEEE Symposium on Security and Privacy (SP)*, May 2017. doi: 10.1109/sp.2017.49. URL http://dx.doi.org/10.1109/SP.2017.49.

[5] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.

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

[7] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018.

[8] Yinpeng Dong, Qi-An Fu, Xiao Yang, Tianyu Pang, Hang Su, Zihao Xiao, and Jun Zhu. Benchmarking adversarial robustness on image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 321–331, 2020.

[9] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

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

[11] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[12] Abdullah Hamdi, Sara Rojas, Ali Thabet, and Bernard Ghanem. Advpc: Transferable adversarial perturbations on 3d point clouds. *arXiv preprint arXiv:1912.00461*, 2019.

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

[14] Peter J Huber. *Robust statistics*, volume 523. John Wiley & Sons, 2004.

[15] Maximilian Ilse, Jakub Tomczak, and Max Welling. Attention-based deep multiple instance learning. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2127–2136, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL http://proceedings.mlr.press/v80/ilse18a.html.

[16] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. *arXiv preprint arXiv:1804.08598*, 2018.

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[18] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015.

[19] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[20] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[21] 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.

[22] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning*, pages 3744–3753, 2019.

[23] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3744–3753, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/lee19d.html.

[24] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on χ-transformed points. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 828–838, 2018.

[25] 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.

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

[27] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

[28] Haggai Maron, Or Litany, Gal Chechik, and Ethan Fetaya. On learning sets of symmetric elements. *arXiv preprint arXiv:2002.08599*, 2020.

[29] 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.

[30] 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.

[31] Naila Murray and Florent Perronnin. Generalized max pooling. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2473–2480, 2014.

[32] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[33] Anh Nguyen and Bac Le. 3d point cloud segmentation: A survey. In *2013 6th IEEE conference on robotics, automation and mechatronics (RAM)*, pages 225–230. IEEE, 2013.

[34] 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.

[35] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017.

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

[37] 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.

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

[39] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.

[40] 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. ISBN 978-1-939133-17-5. URL https://www.usenix.org/conference/usenixsecurity20/presentation/sun.

[41] 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.

[42] 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.

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

[44] Tzungyu Tsai, Kaichen Yang, Tsung-Yi Ho, and Yier Jin. Robust adversarial objects against deep learning models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 954–962, 2020.

[45] Jonathan Uesato, Brendan O'Donoghue, Aaron van den Oord, and Pushmeet Kohli. Adversarial risk and the dangers of evaluating against weak attacks. *arXiv preprint arXiv:1802.05666*, 2018.

[46] Edward Wagstaff, Fabian B Fuchs, Martin Engelcke, Ingmar Posner, and Michael Osborne. On the limitations of representing functions on sets. *arXiv preprint arXiv:1901.09006*, 2019.

[47] 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.

[48] Yida Wang, David Joseph Tan, Nassir Navab, and Federico Tombari. Softpoolnet: Shape descriptor for point cloud completion and classification. *arXiv preprint arXiv:2008.07358*, 2020.

[49] 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)*, 2019.

[50] Yuxin Wen, Jiehong Lin, Ke Chen, and Kui Jia. Geometry-aware generation of adversarial and cooperative point clouds. *arXiv preprint arXiv:1912.11171*, 2019.

[51] 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 (CVPR)*, June 2015.

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

[53] 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.

[54] Cihang Xie and Alan Yuille. Intriguing properties of adversarial training at scale. *arXiv preprint arXiv:1906.03787*, 2019.

[55] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. Adversarial examples for semantic segmentation and object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1369–1378, 2017.

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

[57] 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.

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

[59] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2018.

[60] Tan Yu, Jingjing Meng, and Junsong Yuan. Multi-view harmonized bilinear network for 3d object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 186–194, 2018.

[61] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3391–3401. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/6931-deep-sets.pdf.

[62] Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. Fspool: Learning set representations with featurewise sort pooling. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJgBA2VYwH.

[63] 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 (ICCV)*, October 2019.

[64] Hang Zhou, Dongdong Chen, Jing Liao, Kejiang Chen, Xiaoyi Dong, Kunlin Liu, Weiming Zhang, Gang Hua, and Nenghai Yu. Lg-gan: Label guided adversarial network for flexible targeted attack of point cloud based deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.