# Adversarial Attack and Defense on Point Sets

Jiancheng Yang* Qiang Zhang* Rongyao Fang* Bingbing Ni† Jinxian Liu
Shanghai Jiao Tong University
{jekyll4168, zhangqiang2016, lucas_fang, nibingbing, liujinxian}@sjtu.edu.cn

Qi Tian
Huawei Noahs Ark Lab
tian.qi1@huawei.com

## Abstract

*Emergence of the utility of 3D point cloud data in critical vision tasks (*e.g.*, ADAS) urges researchers to pay more attention to the robustness of 3D representations and deep networks. To this end, we develop an attack and defense scheme, dedicated to 3D point cloud data, for preventing 3D point clouds from manipulated as well as pursuing noise-tolerable 3D representation. A set of novel 3D point cloud attack operations are proposed via pointwise gradient perturbation and adversarial point attachment / detachment. We then develop a flexible perturbation-measurement scheme for 3D point cloud data to detect potential attack data or noisy sensing data. Extensive experimental results on common point cloud benchmarks demonstrate the validity of the proposed 3D attack and defense framework.* [1]

## 1. Introduction

The popularity of 3D sensors, *e.g.*, LiDAR and RGB-D cameras, raises a number of research concerns with 3D vision. Increasing accessible data makes data-driven deep learning approaches practical to be used in many fields, such as autopilot [37, 22], robotics [13, 5] and graphics [35, 15, 31]. Particularly, point cloud is one of the most natural data structures to represent the 3D geometry. Typical CNN-based approaches [26, 24] not only require pre-rendering the sparse point clouds into unnecessarily voluminous representations, but also introduce quantization artifacts [23]. PointNet [23] and DeepSet [36] pioneer the direction of learning representations on the raw point clouds. By learning representation from *permutation-invariant* and *size-varying* **point sets**, this idea is shown to be effective and efficient, and has achieved remarkable success [25, 17, 32].

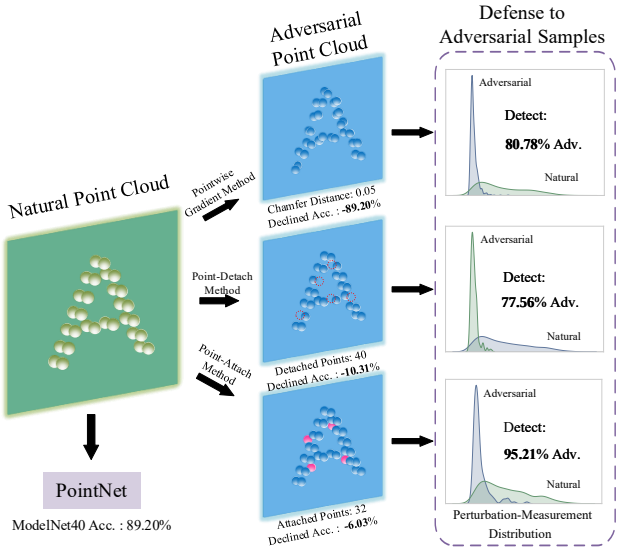


Figure 1. **Framework of the proposed adversarial attack and defense on point clouds.** A well-tuned PointNet [23] is vulnerable to our adversarial attacks. Fortunately, the proposed defense strategies detect most ($> 75\%$) of these adversarial samples, with help of the attacked PointNet. Best viewed in color.

Considering the popularity and potential use in numerous crucial applications of point clouds, *e.g.* self-driving, it is urgent to address the safety issues. It is proven that even high-performance deep image classifiers are vulnerable to the adversarial attacks [28, 29, 2], where visually similar samples are generated to mislead the networks to classify incorrectly. We are interested in the safety and robustness of the deep nets given adversarial point cloud samples, especially when learning is based on the raw point clouds.

It is expected that point cloud classification networks are fragile to *pointwise gradient* guided optimization adversar-

*Equal contribution.
†Corresponding author.
[1]Preprint work in progress.

ial attacks (e.g., FGSM-like methods [8]), due to the generality of these attack methods. We are showing that these pointwise adversarial noises generate visually similar samples in 3D space. Besides, considering the realworld 3D sensors, the common issues are sensing noises and vanishing points. To this regard, we generate extreme sensing noises by adversarial *point-attach* method, and remove possibly vanishing essential points by adversarial *point-detach* method. As will be shown later, these prudent attached or detached points are likely to change the decision of well tuned networks. Theoretically, these points are highly related to the *critical points* [23] existing in the models.

On the other hand, we pursue defense strategies to prevent the proposed adversarial attacks. Based on the observation that the adversarial point clouds destabilize the model output, we explore the intrinsic **robustness of the attacked models**. A *perturbation-measurement* defense framework is developed to detect the adversarial samples, which in principle defines a family of easy-to-implement methods. The proposed perturbations are related to *input transformation* [10] and *stochastic activation* [6] methodology, however either the shattered or stochastic gradients "give a false sense of security" [2]. To this end, instead of utilizing the perturbed samples as defense directly, we measure the statistics of the perturbed outputs of the attacked models to detect the adversarial samples. By changing the threat model with low costs, we detect most of the adversarial samples generated by our attack methods.

The main contributions of this paper are threefold: we 1) address the safety issues of adversarial attacks on point clouds, with three attack methods; 2) propose a flexible perturbation-measurement defense framework to detect the adversarial point cloud samples, with help of the attacked models; and 3) analyze the transferability of adversarial point clouds between different point cloud networks, together with that between point clouds and grids (CNNs). Our findings could benefit the deep network design for point cloud in terms of representation, robustness and reliability. Our code for reproducing the attack and defense results will be soon available.

## 2. Preliminaries

### 2.1. Deep Learning on Point Clouds

Point clouds are widely involved in modern 3D vision research. Shape classification on ModelNet40 dataset [33] is one of important tasks that benchmark this area. Voxel-based methods [26, 24], which classify pre-rendered point clouds with 2D / 3D CNNs, require computation-intensive rendering and (regular or sparse) convolution operation, hindering the real-time application in practice.

Alternatively, learning directly on point sets provides several advantages on speed and accuracy, with the requirements of 1) *permutation-invariance* and 2) ability to process *size-varying* data. PointNets [23] and DeepSets [36] pioneer this direction, and it has become a mainstream on reasoning about point clouds. To make the networks able to consume point sets, they use a stack of symmetric functions based on (max-)pooling to build the neural networks. PointNet++ [25] extends PointNet by introducing hierarchical structure with spatial nearest-neighbor graphs, and achieves better classification performance. Subsequent methods [17, 14] improve the core operations in the hierarchical structure, and DGCNN [32] introduces a dynamic way to build spatial-neighbor graphs. We refer these networks as *Point Cloud Networks (PC-Nets)* for simplicity.

The core operator $f : \mathbb{R}^{N\times c} \to \mathbb{R}^c$ of PointNets is as follows:

$$f(X) = \gamma \max_{x_i \in X} \{h(x_i)\}, \tag{1}$$

where $X \in \mathbb{R}^{N\times c}$ denotes a point set, $x_i \in \mathbb{R}^c$ denotes a single point (a row) in $X$, $\gamma$ and $h$ are two learnable neural networks. It is proven that this simple function is able to approximate any continuous set function [23] defined under *Hausdorff distance*,

$$\mathcal{D}_H(S, S') = \max_{s'\in S'} \min_{s\in S} l_2(s, s'). \tag{2}$$

Moreover, for a $K$-dimension max-pooling based aggregation, it is also proven that there exists $N_C$ points in the input points ($N_C \leq K$) responsible for the final output, *i.e.*, the output does not change by removing any remaining points. For this reason, the $N_C$ points are called the *critical points*. The authors declare robustness on "small corruptions or extra noise points" [23] thanks to the property of critical points. On the contrary, we argue that the PointNets are vulnerable in adversarial setting; besides, attack on the critical points is much successful.

### 2.2. Adversarial Samples and Adversarial Attack

*Adversarial sample* was first formalized by Szegedy *et al*. [28]. Conceptually, an adversarial sample $X_a$ is generated by deliberately perturbing a natural sample $X$, which misleads model $f$ to output incorrectly. We focus on untargeted attack in this study; In other words, for some distance metric $\mathcal{D}$ and constrain $\epsilon \in \mathbb{R}$,

$$f(X_a) \neq f(X), \quad s.t. \mathcal{D}(X_a, X) < \epsilon. \tag{3}$$

Generating adversarial samples is called *adversarial attack*. There has been numerous works focusing on attack on image classification, such as fast gradient sign method (FGSM) [8], least-likely-class iterative method [16] and one-pixel attack [27]. Neural networks are proven to be vulnerable to the adversarial attacks, especially in a *white-box*

setting, where the adversary has full access, including hidden layers and gradients, to the models.

We explore white-box adversarial attacks on point clouds. The main difference is that images are fixed-dimension **grids**, while point clouds are represented as **sets**. A set is orderless and unstructured, thus distance between sets can be very differently defined from that of grids. In our experiments, asymmetric *Chamfer pseudo-distance* ($\mathcal{D}_C$) [1] is used as a main measurement,

$$\mathcal{D}_C(S, S') = \frac{1}{||S'||} \sum_{s' \in S'} \min_{s \in S} l_2(s, s'), \tag{4}$$

where $||\cdot||$ denotes points number in a set. Note the Chamfer distance (Eq. 4) is not so strict as Hausdorff distance (Eq. 2): the latter is less tolerant of outliers in the target set $S'$. In practice, outlier noises are very common in 3D sensing, that is the reason why we choose Chamfer distance.

However, neither the Hausdorff distance nor the Chamfer distance measures the number of points changed. Thereby, apart from modifying natural samples point-wisely, it is feasible to perturb point sets by attaching or detaching some points, which is generally impossible for images. For this case, we define the *number change measurement*,

$$\Delta N_{S,S'} = \text{abs}(||S|| - ||S'||). \tag{5}$$

### 2.3. Defense to Adversarial Attack

Several prior works develop defense strategies against adversarial attacks, these works can be regarded as either of two directions: 1) Improve the ability of attacked models to classify adversarial samples correctly. For instance, adversarial training, *i.e.*, attaching training data with adversarial samples [8, 21]; or using an additional network to "denoise" adversarial samples [9]. 2) Detect adversarial samples and reject them, *e.g.*, LID [19] suggests a characteristic to detect adversarial samples [2].

In this paper, we mainly focus on attack and defense on PC-Nets, on the ModelNet40 and MNIST dataset. Only white-box, untargeted attacks are addressed. Our defense strategy follows the "detect-and-refuse" approaches. We note there is a parallel study [34] working on the adversarial attack on point sets, while we develop a novel defense framework and more attack methods, *e.g.*, point detachment, as well as discuss the transferability of the adversarial samples.

## 3. Attack Methods

### 3.1. Principles and Notations

We represent an input point cloud as $X \in \mathbb{R}^{N \times c}$, where $N$ denotes the number of points, and $c$ denotes the dimension of a single point ($c = 3$ for the 3D space in our work). The set of all point clouds in the dataset is defined as $\mathbb{X}$. $f(\cdot)$ is the point cloud network output probability scores of different classes, and $c^*(\cdot)$ is the true label.

Consequently, the attack problem setting is described as follows: by changing input data $X$ to $X_a = T_\epsilon(X)$, where $T_\epsilon$ denotes an attack function, given a certain attack budget $\epsilon$ under distance metric $\mathcal{D}$, the goal is to decrease classification accuracy of attacked model. We define $\mathbb{B}_\epsilon$ as the set of adversarial samples under budget $\epsilon$,

$$\mathbb{B}_\epsilon : \{X_a \| \ f(X) = f^*(X) \text{ and } f(X_a) \neq f^*(X_a)\}. \tag{6}$$

The attack performance is evaluated by post-attack model accuracy,

$$\eta_\epsilon = \frac{||\{T(X) \mid \arg\max f(T(X)) = c^*(X)\}||}{||\mathbb{X}||}. \tag{7}$$

### 3.2. Pointwise Gradient Method

Inspired by the success of gradient-guided attack methods (*e.g.*, FGSM [8]) on natural images, we first generate the adversarial point clouds by pointwise gradient guided perturbation. Given attack budget $\epsilon$ under the Chamfer distance (Eq. 4), an adversarial sample is obtained by

$$X_a = T_{PG}(X), \quad s.t. \mathcal{D}_C(X_a, X) < \epsilon, \tag{8}$$

where $T_{PG}(x)$ is an iterative perturbation based on pointwise gradient descent. To decrease the model maximum output in the untargeted-attack setting, we obtain its gradient on the input, via backward pass of output of the ground truth class,

$$\nabla X_a = \frac{\partial f^{(t)}(X_a)}{\partial X_a}, \quad t = c^*(X). \tag{9}$$

For FGSM, an adversarial sample is obtained by:

$$X_{n+1} = X_n - \alpha \text{sign} \nabla X_a, \tag{10}$$

where function $sign(x) = \mathbf{1}(x \geq 0) - \mathbf{1}(x < 0)$ ensures that the magnitude of the single-step change is maximized with the $\epsilon$-restriction under certain distance $\mathcal{D}$.

In our experiments, we use a different iteration formula based on $l_2$-normalized gradients [20],

$$X_{n+1} = X_n - \alpha \frac{\nabla X_a}{l_2(\nabla X_a)}, \tag{11}$$

which leads to better attack performance and a more stable convergence. The following proposition declares a theoretical guarantee for universal feasibility of the Pointwise Gradient Method.

**Proposition 1.** *Given any point cloud dataset* $\mathbb{X}$, $\exists \epsilon \in \mathbb{R}$, $\forall X \in \mathbb{X}$ *s.t.* $\arg\max f(X) = c^*(X)$, $\exists X_a = T_{PG}(X)$ : $\mathcal{D}_C(X, X_a) < \epsilon$ *and* $\arg\max f(X_a) \neq c^*(X)$.

[2] Though the authors emphasize that it "is not intended as a defense" in personal communication [2].

### 3.3. Point-Detach Method

Pointwise Gradient Method adds perturbation to every point, which is hardly possible in practice. We investigate a more realistic scenario by detaching a few points; in physical world, point vanishing is common in 3D sensing, due to occlusion and scale issues. Although prior works [23, 25, 17] declare robustness on various numbers of input points, we consider this scenario in an adversarial setting, which can be regarded as a variant of *one-pixel attack* [27] on the point clouds. Since the Chamfer distance does not measure the points missing from the original set (*i.e.*, $\mathcal{D}_C \equiv 0$ when detaching points), we define the attack budget as the number change measurement $\Delta N$ (Eq. 5), thus an adversarial point-detach sample is denoted as

$$X_a = T_{PD}(X) \subseteq X, \quad s.t. \Delta N_{X,X_a} \leq N_d, \tag{12}$$

where $T_{PD}$ defines a point-detach perturbation.

For PointNets [23], we develop a point-detach strategy ulitizing the *critical point* property (see Section 2.1). Recall that the model output changes if and only if the missing point is one of the critical points. As the critical points bounded by the $K$-dimension max-pooling layer are class-agnostic, to efficiently achieve the untargeted attack, we define a *class-dependent importance* $\mathcal{I}^{(i)}$ via Taylor first-order approximation of point $i$'s contribution.

Denote $H_X \in \mathbb{R}^{N\times K}$ as neural networks features with input $X$, before max-pooling aggregation, its gradient matrix $w.r.t.$ the true class output is

$$\nabla H_X = \frac{\partial}{\partial H_X} f^{(t)}(X), \quad t = c^*(X). \tag{13}$$

Note that $\nabla H_X \in \mathbb{R}^{N\times K}$ is a sparse matrix with non-zero only at the critical points, *i.e.*,

$$\nabla H_X^{(i,j)} \neq 0, \, iff \, i = \arg\max_k H_X^{(k,j)} \, for \, j = 1, .., K. \tag{14}$$

To count the value change $\Delta$ of channel $i$ if its critical point is detached, we introduce a *substitute vector* $G \in \mathbb{R}^K$, where $G^{(j)}$ is the second largest value the channel $j$ (the $j_{th}$ column of $H$), then

$$\Delta^{(j)} = \max_i H^{(i,j)} - G^{(j)}, \tag{15}$$

thereby, using a first-order Taylor approximation, the class-dependent importance of point $i$ is,

$$\mathcal{I}^{(i)} = \sum_j^K \nabla H_X^{(i,j)} \cdot \Delta^{(j)}. \tag{16}$$

To confuse the attacked network, we apply a greedy strategy, to iteratively detach the most important point dependent on the true class, until we have detached $N_d$ points. Note that in every iteration, the importance order of the remaining points may change. For this reason, we re-compute the class-dependent importance for each remaining point for every iteration, which makes our point-detach method an $O(N \cdot N_d)$ algorithm.

### 3.4. Point-Attach Method

Similarly, by attaching a few points at appropriate positions, we expect another variant of one-pixel attack on point clouds,

$$X_a = T_{PA}(X) \supseteq X. \tag{17}$$

where $T_{PD}$ is attaching some points to the original point cloud data with the restriction:

$$\Delta N_{X,X_a} \leq N_a, \quad \mathcal{D}_C(X_a, X) < \epsilon. \tag{18}$$

Define $X_a = X \cup X'_a$, where $X'_a$ is the attached points, thereby $\mathcal{D}_C(X'_a, X) = \mathcal{D}_C(X_a, X)$, and $||X'_a|| \leq N_a$. We initialize $X'_a$ randomly, following Eq. 9, we replace $\nabla X$ by,

$$\nabla X'_a = \frac{\partial[f^{(t)}(X_a) + \lambda * \mathcal{D}_c(X, X'_a)]}{\partial X'_a}. \tag{19}$$

Where $\lambda * \mathcal{D}_c(X, X'_a)$ is an Lagrange multipier to restrict the attached points to move around surface of point cloud objects. In our experiments $\lambda = 0.001$ empirically. We use Eq. 11 to only update the attached points without changing the original point cloud. This iteration is repeated until exceeding the adversarial budgets. The proposition below further provides a theoretical guarantee for our Point-Attach Method given any point cloud datasets:

**Proposition 2.** *Given any point cloud dataset* $\mathbb{X}$, $\exists \epsilon \in \mathbb{R}$, $\exists N_a \in \mathbb{N}$, $\forall X \in \mathbb{X}$ *s.t.* $\arg\max f(X) = c^*(X)$, $\exists X_a = T_{PA}(X) : \mathcal{D}_C(X, X_a) < \epsilon$, $\Delta N_{X,X_a} \leq N_a$, *and* $\arg\max f(X_a) \neq c^*(X)$.

## 4. Defense Methods

### 4.1. Principle and Notation

Given a test sample $X \in \mathbb{R}^{N\times c}$, unknown whether it is an adversarial sample $X \in \mathbb{B}$ or a natural sample $X \in \mathbb{X}$, our defense methods are supposed to detect $X \in \mathbb{B}$ and reject them.

Observed that the outputs of adversarial point clouds are less stable than the natural ones facing small perturbation, we assume the adversarial samples exist in a narrow and very structured sub-space in a high-dimension input space; in other words, by perturbing inputs nondirectionally, we expect the attack to be less aggressive with help of the intrinsic robustness of the attacked models.

To this regard, we propose several perturbation methods, which are related to *input transformation* [10] and *stochastic activation* [6] methodology on natural images. However,

for small perturbation, the predictive class does not change in most cases; besides, a perturbation-only defense may be not safe enough since the "obfuscated gradients give a false sense of security" [2]. Instead of utilizing the perturbation as a direct defense, we argue that statistics of the outputs provide rich information to detect the adversarial samples.

Our defense framework follows a *perturbation-measurement* principle, which applies perturbation methods $\mathcal{P}(\cdot)$ on $X$ multiple times before measuring particular statistics of outputs, and detects the adversarial samples by thresholding the statistics. We define an $M$-times pertubation input set of $X$ as,

$$X'_m = \{X'_1, X'_2 ... X'_i ... X'_M \,|\, X'_i = \mathcal{P}_i(X)\}. \tag{20}$$

then an $M$-times perturbation output set is defined as,

$$O'_m = \{O'_1, O'_2 ... O'_i ... O'_M \,|\, O'_i = f(X'_i)\}, \tag{21}$$

where $f$ denotes a PC-Net (PointNet [23] in our experiments). As will be shown later, the distributions of $O'_m$ are very different between natural samples and adversarial samples. We compute certain statistics over $O'_m$, to capture the difference, and report several metrics as [18] to evaluate the adversarial sample detection performance.

- **AUROC** is the Area Under the Receiver Operating Characteristic curve [7], a.k.a. AUC, which serves as an evaluation method widely used in binary classification. Note AUROC is threshold-free and insensitive to class imbalance. In our setting, AUROC measures the separability of the adversarial samples and the natural samples. As we use imperfect PC-Nets in the defense, we consider two situations for fair evaluation: 1) $All$: the AUROC between adversarial samples and **all** natural samples in the test set, and 2) $Correct$: the AUROC between adversarial samples and only the **correctly classified** natural samples in the test set.

- **Defense Detection Rate (DDR)** (@ $t\%$), measures the detection sensitivity at a threshold where the specificity is $1-t\%$, appearing as a single point on the ROC curve. In other words, it measures how many adversarial samples are detected when $t\%$ of natural samples is incorrectly rejected. Note we define the adversarial samples as the positive class.

In the following sections, we instantiate the framework into several perturbations and measurements, to adapt to different attack scenarios.

## 4.2. Perturbation Methods

**Gaussian Noising.** As common practices [30, 9] for robustness test in machine learning, we first add Gaussian noises $\rho$ to $X$, named *Gaussian Noising Method* $\mathcal{G}_i^\sigma(\cdot)$.

$$\mathcal{P}_i(X) = \mathcal{G}_i^\sigma(X) \triangleq X + \rho_i, \quad s.t.\ \rho_i \sim \{N(0, \sigma^2)\}_{N\times c}. \tag{22}$$

The noises are i.i.d. sampled from a Gaussian distribution $N(0, \sigma^2)$. Adding undirectional Gaussian noise to $X$ helps the attacked models to escape from the narrow adversarial sub-space, which enables an effective follow-up detection.

**Quantification.** Motivated by the fact that adversarial perturbations are by definition small in magnitude, we define a *Quantification Method* $\mathcal{Q}_i^\mu(\cdot)$ to convert the inputs into low numerical precision with multiple quantification levels,

$$\mathcal{P}_i(X) = \mathcal{Q}_i^\mu(X) \triangleq \lfloor X \times \frac{M}{\mu \times i} \rfloor \times (\frac{\mu \times i}{M}), \tag{23}$$

where $\mu$ defines as a max quantification level. For $i = 1, ..., M$, quantification level ranges from $\mu/M$ to $\mu$. With larger quantification level, $X'$ appears more distorting. PC-Nets are robust to natural samples with a degree of distortion, while we observe that adversarial samples are vulnerable to quantification, resulting in chaotic distributions of outputs from the classifier and distinguishable statistics.

**Random Sampling.** The above perturbations are defined on the Euclidean space, we then define a perturbation on the number of points changed $\Delta N$. For $X$ contains $N$ points, we randomly sample $n$ ($n < N$) points $s_i$ from $X$ without replacement, named *Random Sampling Method* $\mathcal{S}_i^n(X)$.

$$\mathcal{P}_i(X) = \mathcal{S}_i^n(X) = \{\mathbb{1}_x x \,|\, x \in X,\ \mathbb{1}_x \sim Ber(0.5)\}, \tag{24}$$

where $\mathbb{1}_x$ is sampled from $Bernoulli(0.5)$ distribution to indicate the existence of point $x$ in the post-sampled set. As expected, the sampling perturbation is very effective against the point-attach attacks (Section 5.2).

## 4.3. Measurement Methods

Given a perturbation output set $O'_m$, for each $O'_i \in O'_m$,

$$O'_i = \{o'_{i1}, o'_{i2} ... o'_{ij} ... o'_{iN_c}\}, \quad s.t.\ o'_{ij} \in [0, 1], \tag{25}$$

where $o'_{ij}$ denotes the confidence score of class $j$, and $N_c$ is the number of output classes.

**Set-Indiv Variance Measurement.** As observed that the adversarial samples destabilize the model outputs, we first consider to measure the diversity of confidences on the output classes, instead of using an entropy-based measurement, we find $variance$ is more distinguishable in practice, thereby a *Set-Indiv Variance Measurement* $\mathrm{SIV}(\cdot)$ is defined as

$$\mathrm{SIV}(O'_m) = \frac{1}{N_c} \sum_{k=1}^{N_c} \mathrm{Var}_{i\in 1,2,...M}(o'_{ik}). \tag{26}$$

The variance of each class's confidence set is computed before averaged to SIV measurement, which is empirically the most effective in most cases (Section 5.2).

**Max Confidence-Based Measurement.** Inspired by prior works on out-of-distribution detection [11, 18], we propose to use the max confidence score to detect adversarial samples. In our *perturbation-measurement* framework, we use the max confidence scores statistically.

We define a *Confidence Average Measurement* $\mathrm{CoA}(\cdot)$,

$$\mathrm{CoA}(O'_m) = \frac{1}{M}\sum_{i=1}^{M}(\max_{j\in 1,2,\dots N_c}(o'_{ij})), \tag{27}$$

which measures the average of max confidence scores.

Besides, we also measure the variance of max confidence scores, by designing *Confidence Variance Measurement* $\mathrm{CoV}(\cdot)$,

$$\mathrm{CoV}(O'_m) = \mathrm{Var}_{i\in 1,2\dots M}(\max_{j\in 1,2\dots N_c}(o'_{ij})). \tag{28}$$

With the following two assumptions, we provide a theoretical guarantee for the effectiveness of Max Confidence-based Measurement.

**Assumption 1.** *PC-Nets are local continuous convex or concave functions around natural samples.*

**Assumption 2.** *The proportion of adversarial samples in the local area around natural samples is small enough,* i.e.*, given natural sample $X_n$,*

$$\exists\delta>0, \textit{define}\ \mathbb{D}_\delta=\{X\,|\,\mathcal{D}_C(X,X_n)<\delta\},$$
$$\mathbb{B}_\delta=\{X_a\ \textit{is adversarial sample}\,|\,\mathcal{D}_C(X_a,X_n)<\delta\}\subset\mathbb{D},$$
$$\exists\epsilon\ll 1:\forall X, P(X\in\mathbb{B}_\delta)/P(X\in\mathbb{D}_\delta)<\epsilon.$$

**Proposition 1.** *Given any sample $X$, it can always be detected whether it is an adversarial sample or a natural sample by Max Confidence-Based Measurement (Section 4.3) ($CoV$ for Convex functions and $CoA$ for Concave functions), with Gaussian Noising or Quantification perturbation (Section 4.2).*

## 5. Results

In this section, we evaluate the attack performances of our methods on a simple yet effective PC-Net, PointNet (with T-Net) [23]. At the same time, we conduct defense experiments on these adversarial point clouds with positive results. All results are reported on ModelNet40 dataset [33] of 40-category CAD models if not specified otherwise. We use the official split with 9,843 samples for training, and 2,468 samples for test / attack / defense. 1,024 points are uniformly sampled from the mesh surfaces as in PointNet.

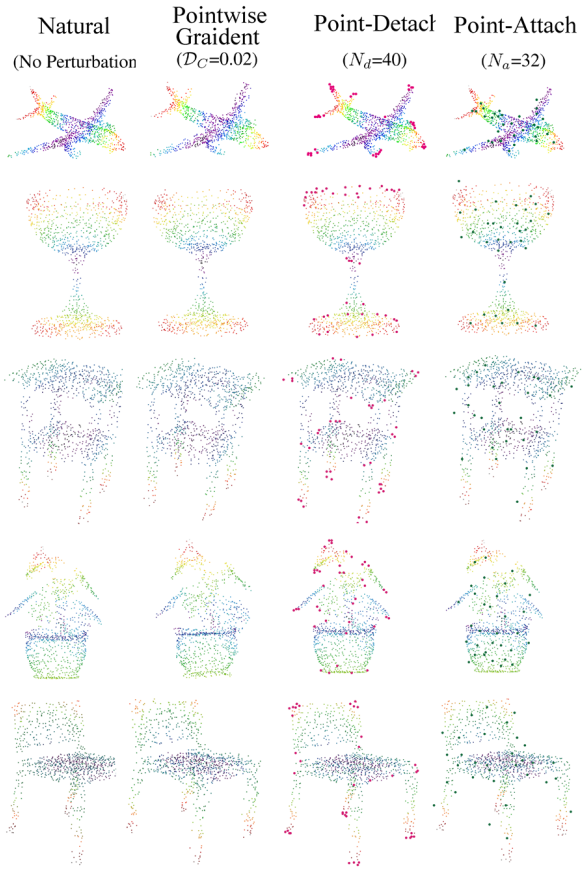


Figure 2. **Natural point clouds and adversarial point clouds produced by three attack methods.** Pointwise Gradient method point clouds are generated in $\mathcal{D}_c = 0.02$. The 40 red points in Point-Detach method are points that have been detached. The 32 green points in Point-Attach method are attached points.

| Pointwise Gradient Method | | | | | | |
|---|---|---|---|---|---|---|
| $\mathcal{D}_C$ | 0.01 | 0.02 | 0.03 | 0.04 | 0.05 | 0.06 |
| $\eta$ | 86.58 | 59.21 | 39.38 | 29.25 | 9.25 | 0.00 |
| Point-Detach Method | | | | | | |
| $N_d$ | 8 | 12 | 16 | 20 | 30 | 40 |
| $\eta$ | 87.54 | 86.53 | 85.21 | 83.89 | 81.21 | 78.89 |
| Point-Attach Method | | | | | | |
| $N_a$ | 16 | 16 | 16 | 32 | 32 | 32 |
| $\mathcal{D}_c$ | 0.1 | 0.2 | 0.3 | 0.3 | 0.4 | 0.5 |
| $\eta$ | 86.82 | 85.14 | 84.89 | 84.26 | 83.89 | 83.17 |

Table 1. **Attack performance on ModelNet40 dataset with three different methods**. Metric ($\eta$) is the accuracy (%) on classification after attacking.

### 5.1. Attack Performance

In order to verify the performance of our attack method, extensive experiments are conducted. The attack performance is shown in Table 1. Different attack parameters

| Attack Method | Param. | Perturb. | Measur. | AUC$_{All}$ | AUC$_{Correct}$ | DDR (@5%) | DDR (@10%) |
|---|---|---|---|---|---|---|---|
| Pointwise Gradient | $\mathcal{D}_C = 0.02$ | Quanti. (0.08) | SIV | 0.9168 | 0.9381 | 61.34 | 80.78 |
| Point-Detach | $N_d = 20$ | GN. (0.012) | SIV | 0.9229 | 0.9517 | 65.69 | 88.73 |
| Point-Detach | $N_d = 40$ | GN. (0.012) | SIV | 0.9011 | 0.9343 | 54.26 | 77.56 |
| Point-Attach | $N_a = 32$ | RS. (1000) | SIV | 0.9729 | 0.9816 | 93.15 | 95.21 |

Table 2. **Defense performance with best perturbation-measurement combination toward three attack methods.** The defense strategies detect most ($> 75\%$) adversarial samples. The Defense Detection Rates (DDR) are evaluated with $Correct$ natural samples.

correspond to different attack intensities. By applying our Pointwise Gradient method, the accuracy of PointNet could be reduced to $0\%$, which means that all natural samples are attacked. Although Point-Detach method does not achieve comparable performance in terms of accuracy reduction, the adversarial samples generated by this method are generally unnoticeable and can easily deceive human's eyes. In Point-Attach method, we utilize different parameters. Better performance is achieved with more attached points and larger Chamfer distance.

Figure 2 shows the contrast between natural samples and adversarial samples, which is visually indiscernible. In Point-Detach method, the detached points are the critical points, which are often located on the edges of the point clouds.

### 5.2. Defense Performance

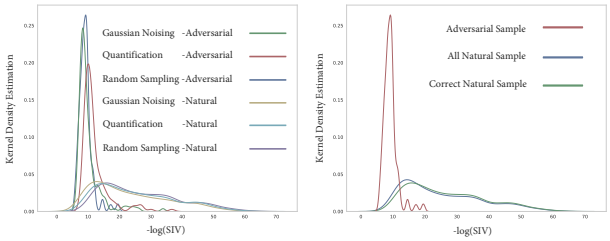


Figure 3. **Left**: Kernel density estimation (KDE) of $-log(SIV)$ towards 32-point Point-Attach method, which shows statistics distribution of adversarial samples and $Correct$ natural samples. **Right**: KDE contrast between adversarial samples, $All$ natural samples, and $Correct$ natural samples, which reveals the Defense Error II in Section 7.

Perturbation-measurement defense methods are utilized to detect the adversarial samples produced by three attack methods. AUROC and Defense Detection Rate (DDR) reflect the defense performance of different methods, which are shown in Table 2. Given the fact that our defense methods strongly rely on the robustness of PC-Nets, their performance drops with $N_d$ increasing in the defense of Point-Detach attack. When the critical points detached, the fluctuation of outputs decreases, resulting in a decrease of DDR. The statistics $CoA$ and $CoV$ perform slightly worse than $SIV$, their performance is shown in Appendix D.

| | 0.00 | 0.01 | 0.02 | 0.03 | 0.04 | 0.05 |
|---|---|---|---|---|---|---|
| PointNet | 89.20 | 86.58 | 59.21 | 39.38 | 29.25 | 9.25 |
| PointNet++ | 90.43 | 80.50 | 52.55 | 10.43 | 1.05 | 0.00 |
| DGCNN | 91.51 | 87.48 | 82.40 | 75.52 | 63.63 | 49.59 |

Table 3. The accuracy (%) of different PC-Nets and $\mathcal{D}_C$. It shows the robustness of PC-Nets, in which DGCNN performs the best robustness among the three PC-Nets.

Two kinds of natural samples are evaluated: $All$ natural samples which belong to the whole test dataset and $Correct$ natural samples which are point clouds that can be correctly classified by PC-Nets. We will give a further discussion in Section 7 about these two natural samples.

The kernel density estimation (KDE) of $-log(SIV)$ is shown in Figure 3, which reflects the distribution of measured statistics. We apply $-log(\cdot)$ operation for visual convenience. The adversarial samples can be detected by choosing different thresholds.

## 6. Transferability of Adversarial Samples

### 6.1. Between Various PC-Nets

Some other PC-Nets such as PointNet++ by Qi [25] and Dynamic Graph CNN by Wang [32] are designed based on PointNet [23]. We evaluate the attack performance and transferability of these PC-Nets.

**Attack Performance.** Pointwise Gradient method is universal to attack PC-Nets. Thus, we apply it to evaluate the robustness of them. The accuracy of PC-Nets after attacking is shown in Table 3, which reflect their robustness.

We observe that DGCNN has better robustness than that of PointNet and PointNet++. Since Pointwise Gradient method is directional, the dynamic edge convolution operation in DGCNN disturbs the attacking direction and contributes to high robustness.

The accuracy of PointNet++ decreases with attack intensity increasing, concluding that PointNet++ model appears worse robustness than PointNet and DGCNN.

| | PointNet | PointNet++ | DGCNN |
|---|---|---|---|
| PointNet | 100.0 / 0.00 | 94.63 / 94.21 | 88.57 / 87.01 |
| PointNet++ | 89.79 / 83.01 | 100.0 / 0.00 | 84.42 / 81.56 |
| DGCNN | 92.63 / 87.53 | 95.89 / 94.63 | 100.0 / 0.00 |

Table 4. Adversarial samples transferability evaluation of different PC-Nets. The nets in the left column generate adversarial sample and the nets in the first row evaluate the transferred adversarial samples. The first number is accuracy (%) of the samples before attacking and the second number is transferred accuracy (%).

**Transferability.** To evaluate transferability between these PC-Nets, we transfer adversarial samples generated by one net with Pointwise Gradient method to another evaluating PC-Net. The evaluation results are shown in Table 4, in which the transferred accuracy means the test accuracy of generated adversarial samples on evaluating PC-Net.

The evaluation results indicate that the attack transferability of different PC-Nets is poor, which means that Pointwise Gradient method is not suited for black-box adversarial attack in PC-Nets cases.

### 6.2. Between PC-Nets and CNNs

To fully investigate the transferability of adversarial samples generated by our methods, we explore the transfer robustness between PC-Nets and CNNs on MNIST dataset. Data conversion details between point clouds and MNIST images are provided in Appendix B.

**Transferring from PC-Nets to CNNs.** We apply Pointwise Gradient method to attack PC-Net (PointNet) which classifies MNIST. The attack performance $\eta$ is 21%. After converting the adversarial point clouds to images and transfering them to MNIST CNN model, the accuracy of the CNN model drops from 99.1% to 60.8%. The left part in Figure 4 shows samples generated by PC-Nets.

**Transferring from CNNs to PC-Nets.** Different attack thresholds $\epsilon$ are set on the MNIST CNN model. We use PointNet model to classify these samples, which shows notable robustness: PointNet still reaches $98.9\%$ accuracy in the worst case. The attack and transferring performance is shown in the right part of Figure 4.

## 7. Discussion

**Robustness on Rotation** We find that rotation does not produce adversarial samples to PointNet effectively, since the PointNet **is sensitive to rotation**. Note that rotation data augmentation has been applied during training the PointNet with T-Net [23] mimicking spatial transformers [12].

In our experiment, rotation angles in x-axis and y-axis are limited within 0.3 rad. We attempt to attack PointNet

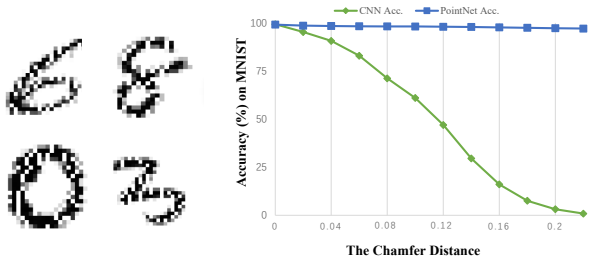


Figure 4. **Left**: The converted 2D images from MNIST samples attacked by PC-Nets. **Right**: Transferring performance from CNN models to PC-Nets. Attack CNN models on MNIST dataset to obtain some adversarial samples and use PC-Nets to classify these samples. The result indicates that PC-Nets is robust to the adversarial samples transferred from CNN models.

using a gradient-based approach on the rotation matrix, see Appendix C for the details of rotation attack. The attack method drops the classification accuracy from $89.2\%$ to $76.01\%$. However, randomly rotating natural point clouds within a same degree drops the accuracy to $77.53\%$, with NO significant difference. The reason for which we argue is that the weak robustness on rotation provides non-informative gradients ("stochastic gradients") to the gradient based attack. For AI safety, PC-Nets should also be designed with theoritically guaranteed rotation-invariance [3, 4]. We leave this for further exploration.

**Two Types of Defense Errors.** We characterize two types of defense errors with *perturbation-measurement* framework. **Defense Error I** is the error caused by the defense framework. As illustrated in Figure 3 Left, there are overlaps between the distributions of adversarial samples and natural samples. To reduce the Defense Error I, advanced perturbations and measurements with fine-tuned hyper-parameters should be explored.

As we explore the intrinsic robustness of the attack model, we suffer from its weakness by design, resulting in **Defense Error II**. Refer to Figure 3 Right, the distribution of $Correct$ and $All$ natural samples is sightly different. We observe that the defense performance with $Correct$ natural samples better than that with $All$ natural samples, which means the misclassified natural samples by PC-Nets are more likely to be mistakenly regarded as adversarial samples. We can choose $Correct$ samples in defense methods to less count the Defense Error II. Theoretically, a better-performing model comes with lower Defense Error II.

**Limitations and Further Work.** Although we have conducted experiments on transferability, all these three methods are based on white-box attack. It is a further work for black-box PC-Nets adversarial attacks and defenses.

We only apply our Point-Detach attack method on PointNet model due to the utilization of critical points in PointNet whom many other PC-Nets share the same structure with, it can be further explored on Point-Detach attack method on other PC-Nets.

Since our methods utilize the PC-Nets' own robustness to detect adversarial samples, our defense to the Point-Detach attack methods does not perform well with large $N_d$. The robustness of PC-Nets decreases as the critical points detached, which affects our measured statistics and worsen the defense performance.

## References


[1] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas. Learning representations and generative models for 3d point clouds. 2018.

[2] A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

[3] T. Cohen and M. Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999, 2016.

[4] T. S. Cohen, M. Geiger, J. Köhler, and M. Welling. Spherical cnns. *arXiv preprint arXiv:1801.10130*, 2018.

[5] H. Deng, T. Birdal, and S. Ilic. Ppfnet: Global context aware local features for robust 3d point matching. *Computer Vision and Pattern Recognition (CVPR). IEEE*, 1, 2018.

[6] G. S. Dhillon, K. Azizzadenesheli, Z. C. Lipton, J. Bernstein, J. Kossaifi, A. Khanna, and A. Anandkumar. Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*, 2018.

[7] T. Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[8] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. volume abs/1412.6572, 2014.

[9] S. Gu and L. Rigazio. Towards deep neural network architectures robust to adversarial examples. *CoRR*, abs/1412.5068, 2014.

[10] C. Guo, M. Rana, M. Cisse, and L. van der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

[11] D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016.

[12] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015.

[13] F. Järemo Lawin, M. Danelljan, F. S. Khan, P.-E. Forssen, and M. Felsberg. Density adaptive point set registration. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[14] M. Jiang, Y. Wu, and C. Lu. Pointsift: A sift-like network module for 3d point cloud semantic segmentation. volume abs/1807.00652, 2018.

[15] H. Kato, Y. Ushiku, and T. Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3907–3916, 2018.

[16] A. Kurakin, I. J. Goodfellow, and S. Bengio. Adversarial examples in the physical world. volume abs/1607.02533, 2016.

[17] Y. Li, R. Bu, M. Sun, and B. Chen. Pointcnn. volume abs/1801.07791, 2018.

[18] S. Liang, Y. Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017.

[19] X. Ma, B. Li, Y. Wang, S. M. Erfani, S. N. R. Wijewickrema, M. E. Houle, G. Schoenebeck, D. Song, and J. Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. *CoRR*, abs/1801.02613, 2018.

[20] T. Miyato, S. Maeda, M. Koyama, and S. Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *CoRR*, abs/1704.03976, 2017.

[21] S. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2574–2582, 2016.

[22] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. *arXiv preprint arXiv:1711.08488*, 2017.

[23] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 77–85, 2017.

[24] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2016.

[25] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5105–5114, 2017.

[26] H. Su, S. Maji, E. Kalogerakis, and E. 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.

[27] J. Su, D. V. Vargas, and K. Sakurai. One pixel attack for fooling deep neural networks. volume abs/1710.08864, 2017.

[28] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. volume abs/1312.6199, 2013.

[29] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

[30] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008.

[31] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. *arXiv preprint arXiv:1804.01654*, 2018.

[32] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018.

[33] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. 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.

[34] C. Xiang, C. R. Qi, and B. Li. Generating 3d adversarial point clouds. *arXiv preprint arXiv:1809.07016*, 2018.

[35] X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In *Advances in Neural Information Processing Systems*, pages 1696–1704, 2016.

[36] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. In *Advances in Neural Information Processing Systems*, pages 3391–3401, 2017.

[37] Y. Zhou and O. Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. *arXiv preprint arXiv:1711.06396*, 2017.

# Appendix

## A. Proofs

### A.1. On Pointwise-Gradient Attack

**Proposition 1.** *Given any point cloud dataset* $\mathbb{X}$, $\exists \epsilon \in \mathbb{R}$, $\forall X \in \mathbb{X}$ *s.t.* $\arg\max f(X) = c^*(X)$, $\exists X_a = T_{PG}(X) : \mathcal{D}_C(X, X_a) < \epsilon$ *and* $\arg\max f(X_a) \neq c^*(X)$.

*Proof.* Ultimately, for any sample $X$ with label $t$, we could pointwisely change it into another sample $X'$ with label $t'$.

In this way, $\epsilon_X = \mathcal{D}_C(X, X')$. For all samples in the dataset, $\epsilon = \max_{X \in \mathbb{X}}(\epsilon_X)$.

□

### A.2. On Point-Attach Attack

**Proposition 2.** *Given any point cloud dataset* $\mathbb{X}$, $\exists \epsilon \in \mathbb{R}$, $\exists N_a \in \mathbb{N}$, $\exists X \in \mathbb{X}$ *s.t.* $\arg\max f(X) = c^*(X)$, $\exists X_a = T_{PA}(X) : \mathcal{D}_C(X, X_a) < \epsilon$, $\Delta N_{X,X_a} \leq N_a$, *and* $\arg\max f(X_a) \neq c^*(X)$.

*Proof.* By contradiction.

Assume there is not such sample in the dataset. We choose two samples $X_1$ and $X_2$ with different labels, then attach one to another as a new sample $X'$. If the prediction label is the same as the $X_1$, then we successfully attack $X_2$ by attaching $X_1$. Otherwise, we successfully attack $X_1$ by attaching $X_2$. □

### A.3. On Max Confidence-Based Measurement

**Assumption 1.** *PC-Nets are local continuous convex or concave functions around natural samples.*

**Assumption 2.** *The proportion of adversarial samples in the local area around natural samples is small enough,* i.e.*, given natural sample* $X_n$*,*

$$\exists \delta > 0, \textit{define } \mathbb{D}_\delta = \{X \mid \mathcal{D}_C(X, X_n) < \delta\},$$
$$\mathbb{B}_\delta = \{X_a \textit{ is adversarial sample} \mid \mathcal{D}_C(X_a, X_n) < \delta\} \subset \mathbb{D},$$
$$\exists \epsilon \ll 1 : \forall X, P(X \in \mathbb{B}_\delta)/P(X \in \mathbb{D}_\delta) < \epsilon.$$

**Proposition 3.** *Given any sample* $X$*, it can always be detected whether it is an adversarial sample or a natural sample by Max Confidence-Based Measurement (Section 4.3) (*$CoV$ *for Convex functions and* $CoA$ *for Concave functions), with Gaussian Noising or Quantification perturbation (Section 4.2).*

*Proof.* We prove this proposition in two separate propositions (Proposition 4 and Proposition 5). □

**Proposition 4** (Concave)**.** $CoA$ *is an effective index to detect whether an given sample X is an adversarial sample or a natural sample if PC-Nets are local continuous concave functions.*

*Proof.* Without losing generality, we choose a dataset containing with only 2 labels: negative and positive, the network output value is limited between 0 and 1. Specifically, if it is more than 0.5, it denotes positive class, otherwise, it denotes negative class. Similar to Gaussian Noising, Quantization is equavalent to add perturbation using uniform distribution. For simplicity, we take uniform distribution as example.

Assume $x_1$ is the quantized data for a negative adversarial sample $X_a$ and $x_2$ is the quantized data of the corresponding positive natural sample $X$. Besides, we apply a same perturbation to $x_1$ and $x_2$, $\mathcal{D}_C(x_1, X_a) \in [a_1, b_1]$, $\mathcal{D}_C(x_2, X) \in [a_2, b_2]$, note $b_1 - a_1 = b_2 - a_2$ (for the "same perturbation"). Thus, according to the definition of $CoA$, we have:

$$CoA(x_1) = \frac{1}{b_1 - a_1} \int_{a_1}^{b_1} [1 - f(x)]dx,$$

$$CoA(x_2) = \frac{1}{b_2 - a_2} \int_{a_2}^{b_2} f(x)dx.$$

Define $f$ is the PC-Net, $f'$ is its derivative. Since $f$ is continuous, $\exists c : f(c) = 0.5$ when the point set varies from $x_2(f(x_2) > 0.5)$ to $x_1(f(x_1) < 0.5)$. Besides, since $f$ is a concave function via the distance, we have the following equations:

$$f(x_1) > 0.5 - |f'(c)| * \mathcal{D}_C(x_1, c),$$

$$f(x_2) > 0.5 + |f'(c)| * \mathcal{D}_C(c, x_2).$$

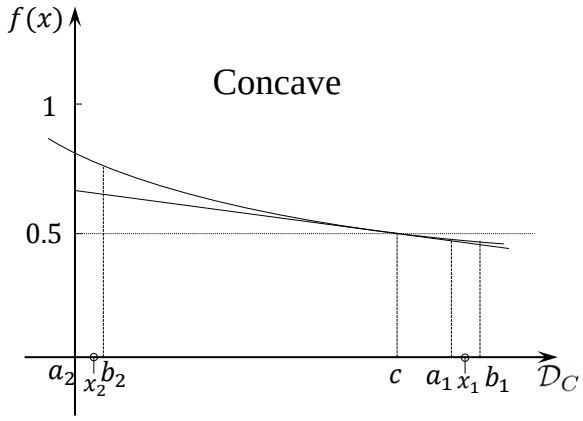


Figure 5. Illustration for Proposition 4. The tangent at $f(c) = 0.5$ is shown in the figure.

According to our assumption, adversarial samples are much less than natural samples, as illustrated in Figure 5:

$$\mathcal{D}_C(c, x_2) > \mathcal{D}_C(x_1, c).$$

Thereby,

$$f(x_2) > 1 - f(x_1)$$
$$\Rightarrow \frac{1}{b_2 - a_2} \int_{a_2}^{b_2} f(x)dx > \frac{1}{b_1 - a_1} \int_{a_1}^{b_1} [1 - f(x)]dx,$$

which means $CoA(x_2) > CoA(x_1)$, thus $CoA$ is effective to detect adversarial samples. □

To prove the next proposition, we provide following lemma:

**Lemma.** $\forall x$, $\forall f(x)$, $\forall c$: $\int_a^b [f(x) - \frac{1}{b-a} \int_a^b f(x)dx]^2 dx \leqslant \int_a^b (f(x) - c)^2 dx$

*Proof.*

$$
\begin{aligned}
&\int_a^b [f(x) - \frac{1}{b-a}\int_a^b f(x)dx]^2 dx \\
&= \int_a^b (f(x)-c)^2 dx + 2(c - \frac{1}{b-a}\int_a^b f(x)dx)\times \\
&\quad \int_a^b (f(x)-c)dx + (c - \frac{1}{b-a}\int_a^b f(x)dx)^2 (b-a) \\
&= \int_a^b (f(x)-c)^2 dx + \frac{2}{a-b}[\int_a^b (f(x)-c)dx]^2 \\
&\quad + \frac{1}{b-a}[\int_a^b (f(x)-c)dx]^2 \\
&\leqslant \int_a^b (f(x)-c)^2 dx
\end{aligned}
$$

□

**Proposition 5** (Convex)**.** *$CoV$ is an effective index to detect whether an given sample X is an adversarial sample or a natural sample if PC-Nets are local continuous convex functions.*

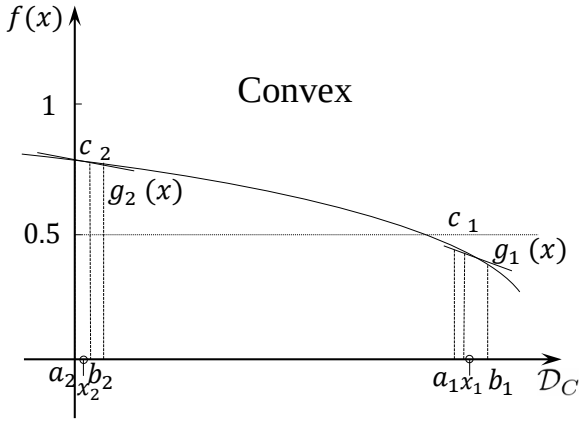


Figure 6. Illustration for the Proposition 5. The two straight lines are parameterized by $g_1(x) = f'(a_1)x + l_1, s.t.\, g_1(c_1) = f(c_1)$, $g_2(x) = f'(b_2)x + l_2, s.t.\, f(c_2) = g(c_2)$.

*Proof.* We use the same notations in the Proposition 5.

For adversarial sample: according to mean value theorem of integrals, $\exists c_1, f(c_1) = \frac{1}{b_1-a_1}\int_{a_1}^{b_1} f(x)dx$, we construct: $g_1(x) = f'(a_1)x + l_1, s.t. g_1(c_1) = f(c_1)$.

$$
\begin{aligned}
CoV(x_1) &= \int_{a_1}^{b_1} [f(x) - f(c_1)]^2 dx \\
&> \int_{a_1}^{b_1} [g(x) - f(c_1)]^2 dx \text{ (illustrated in Figure 6)} \\
&= \int_{a_1}^{b_1} [g(x) - g(c_1)]^2 dx \\
&\geq \int_{a_1}^{b_1} [g(x) - \frac{1}{b_1-a_1}\int_{a_1}^{b_1} g(x)dx]^2 dx \text{ (Lemma)} \\
&= \int_{a_1}^{b_1} [f'(a_1)x - \frac{f'(a_1)(a_1+b_1)}{2}]^2 dx \\
&= \frac{1}{12} f'(a_1)^2 (b_1-a_1)^3
\end{aligned}
$$

For natural sample: according to our assumption that $f(x)$ is continuous, we can choose $c_2$ satisfying $D_C(c_2, x_2) = \frac{a_2+b_2}{2}$ and construct $g_2(x) = f'(b_2)x + l_2, s.t. f(c_2) = g(c_2)$.

$$
\begin{aligned}
CoV(x_2) &= \int_{a_2}^{b_2} [f(x) - \frac{1}{b_2-a_2}\int_{b_2}^{b_2} f(x)dx]^2 dx \\
&\leq \int_{a_2}^{b_2} [f(x) - f(c_2)]^2 dx \text{ (Lemma)} \\
&< \int_{a_2}^{b_2} [g_2(x) - f(c_2)]^2 dx \text{ (illustrated in Figure 6)} \\
&= \int_{a_2}^{b_2} [g_2(x) - g(c_2)]^2 dx \\
&= \int_{a_2}^{b_2} [f'(b_2)x - \frac{f'(b_2)(a_2+b_2)}{2}]^2 dx \\
&= \frac{1}{12} f'(b_2)^2 (b_2-a_2)^3
\end{aligned}
$$

Since $a_1 > b_2$ and $f(x)$ is a convex function, $f'(a_1)^2 > f'(b_2)^2$. Besides, we have $b_2 - a_2 = b_1 - a_1$, which means $CoV(x_1) > CoV(x_2)$. Thereby, $CoV$ is effective to detect adversarial samples and natural samples. □

## B. Conversion between MNIST Images and Point Clouds

From images to point clouds, we convert each pixel position to $x$, $y$ axis and grayscale value to $z$ axis in point cloud $\{(x, y, z)\}$.

From (attacked) point clouds to images, we firstly abandon the points whose $x$, $y$ axis is less than 0 or more than 1, which are out of border; then, we convert $x$, $y$ axis to the corresponding pixel position in image with $\frac{1}{28}$ quantification, and $z$ axis to pixel value. If several points are converted to a overlapping pixel position in image, we adapt their average $z$ values. For the pixels without corresponding point, we set their values as zero.

## C. Details of Rotation Attack

In Section 7, we evaluate robustness of PointNet [23] on rotation. We use Euler angles (Figure 7) to parameterize rotation.

| AUROC | | Random Sampling | | | Quantification | | | Gaussian Noising | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | SIV | CoA | CoV | SIV | CoA | CoV | SIV | CoA | CoV |
| Pointwise Gradient | | | $n = 500$ | | | $\mu = 0.08$ | | | $\sigma = 0.020$ | |
| $\mathcal{D}_C = 0.02$ | All | 0.8806 | 0.8544 | 0.8749 | **0.9168** | 0.8639 | 0.9086 | 0.8839 | 0.8589 | 0.8816 |
| | Correct | 0.9132 | 0.9006 | 0.9054 | **0.9381** | 0.9125 | 0.9323 | 0.9207 | 0.9040 | 0.9192 |
| | | | $n = 500$ | | | $\mu = 0.12$ | | | $\sigma = 0.028$ | |
| $\mathcal{D}_C = 0.05$ | All | 0.8092 | 0.74283 | 0.80705 | **0.8352** | 0.75224 | 0.82039 | 0.78812 | 0.74675 | 0.78686 |
| | Correct | 0.84692 | 0.79698 | 0.84286 | **0.85932** | 0.80872 | 0.84934 | 0.82665 | 0.79681 | 0.82403 |
| Point-Detach | | | $n = 950$ | | | $\mu = 0.02$ | | | $\sigma = 0.012$ | |
| $N_d = 20$ | All | 0.9053 | 0.9083 | 0.9016 | 0.9154 | 0.9109 | 0.9097 | **0.9229** | 0.9139 | 0.9152 |
| | Correct | 0.9393 | 0.9439 | 0.9357 | 0.9454 | 0.9455 | 0.9407 | **0.9517** | 0.9475 | 0.9453 |
| $N_d = 40$ | All | 0.8705 | 0.8770 | 0.8681 | 0.8901 | 0.8827 | 0.8863 | **0.9011** | 0.8903 | 0.8969 |
| | Correct | 0.9130 | 0.9190 | 0.9102 | 0.9249 | 0.9231 | 0.9216 | **0.9343** | 0.9282 | 0.9304 |
| Point-Attach | | | $n = 1000$ | | | $\mu = 0.02$ | | | $\sigma = 0.012$ | |
| $N_a = 16$ | All | **0.9632** | 0.8275 | 0.9588 | 0.8058 | 0.8084 | 0.8031 | 0.8169 | 0.8113 | 0.8159 |
| | Correct | **0.9750** | 0.8712 | 0.9720 | 0.84656 | 0.8509 | 0.8442 | 0.8578 | 0.8539 | 0.8565 |
| $N_a = 32$ | All | **0.9729** | 0.8507 | 0.9686 | 0.8366 | 0.8329 | 0.8365 | 0.8516 | 0.8379 | 0.8512 |
| | Correct | **0.9816** | 0.8934 | 0.9785 | 0.8752 | 0.8752 | 0.8746 | 0.8890 | 0.8800 | 0.8878 |

Table 5. **Defense AUROC by using various perturbation-measurement combinations against adversarial samples produced by three attacking methods (Pointwise Gradient, Point-Detach and Point-Attach).**

$$M_\alpha = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$M_\beta = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta & \sin\beta \\ 0 & -\sin\beta & \cos\beta \end{bmatrix}$$

$$M_\gamma = \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$X_a = X\ M_\gamma\ M_\beta\ M_\alpha$$

$$\nabla\theta = \frac{\partial f^{(t)}(X_a)}{\partial\theta}, \theta = (\alpha, \beta, \gamma)$$

Following a gradient-based iteration, we update the $\theta$ adversarially. Each dimension of $\theta$ (*i.e.*, $\alpha$, $\beta$, and $\gamma$) is limited in 0.3 rad.

## D. More Defense Results

More defense results are showed in Table 5. It provides defense AUROC with all nine Perturbation-Measurement combinations.

## E. Visualization on Pointwise-Gradient Attack

To illustrate the distortion of point cloud under Pointwise Gradient attack with different intensities, we visualize several point cloud samples in different attack levels. It is observed that certain point cloud samples become visibly distorted with the Chamfer distance larger than 0.02.

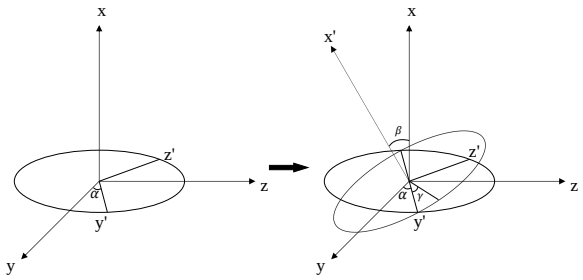


Figure 7. Rotation system using Euler angles $\alpha$, $\beta$, and $\gamma$. The rotation axes are $x$-axis (left), $y'$-axis (right) and $x'$-axis (right). The three angles are limited in 0.3 rad.

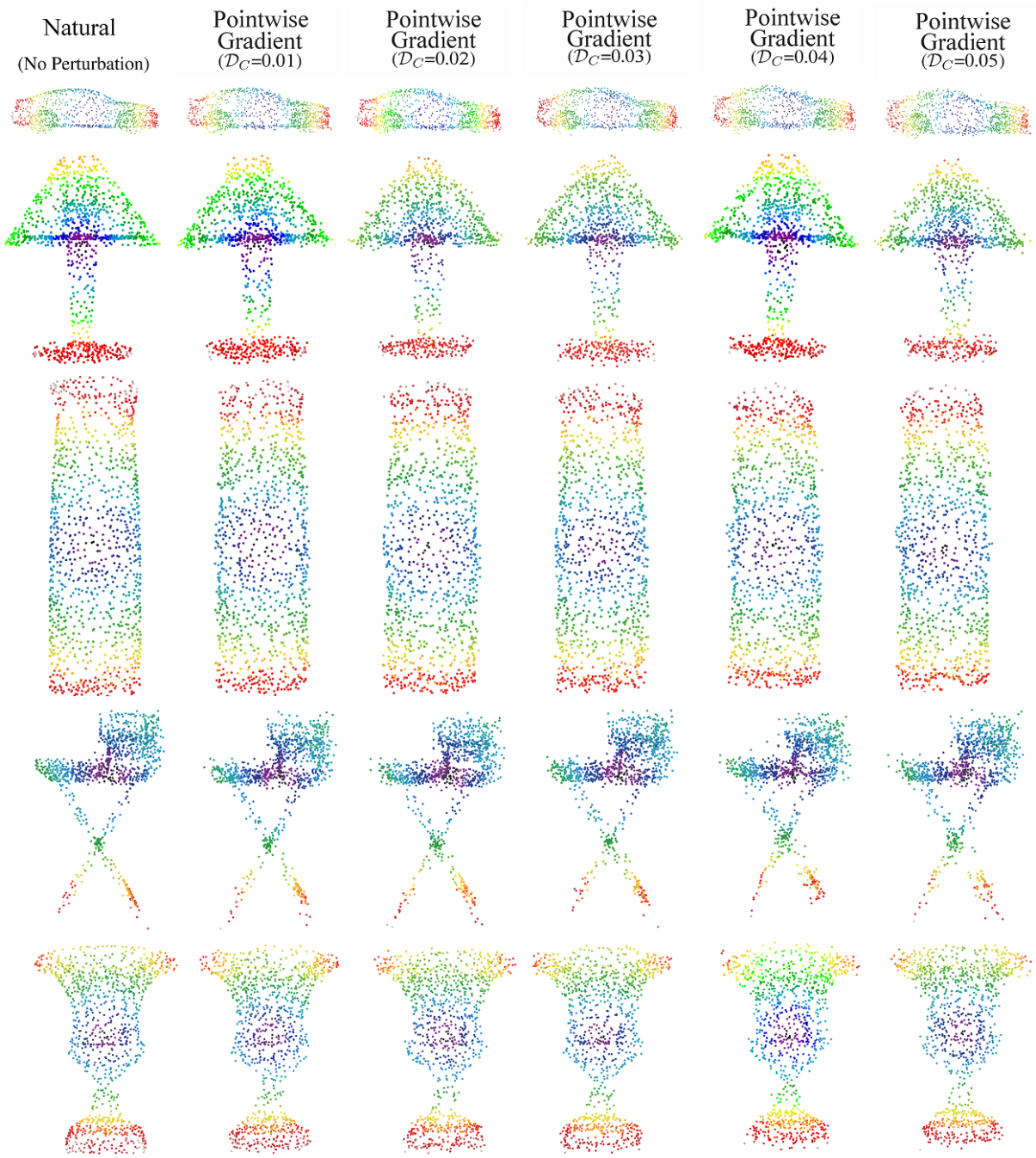


Figure 8. Natural samples and Pointwise Gradient attacked samples with Chamfer distance varies from 0.01 to 0.05. Certain samples become visibly distorted with the Chamfer distance larger than 0.02.