Improving Adversarial Robustness of 3D Point Cloud Classification Models

Guanlin Li[1,2], Guowen Xu[1,*], Han Qiu[3], Ruan He[4], Jiwei Li[5,6], and Tianwei Zhang[1]

[1] Nanyang Technological University, [2] S-Lab, NTU,
[3] Tsinghua University, [4] Tencent, [5] Shannon.AI, [6] Zhejiang University,
{guanlin001,guowen.xu, tianwei.zhang}@ntu.edu.sg,
qiuhan@tsinghua.edu.cn, ruanhe@tencent.com,
jiwei_li@shannonai.com
[*] Corresponding author

Abstract. 3D point cloud classification models based on deep neural networks were proven to be vulnerable to adversarial examples, with a quantity of novel attack techniques proposed by researchers recently. It is of paramount importance to preserve the robustness of 3D models under adversarial environments, considering their broad application in safety- and security-critical tasks. Unfortunately, existing defenses are not general enough to satisfactorily mitigate all types of attacks. In this paper, we design two innovative methodologies to improve the adversarial robustness of 3D point cloud classification models. (1) We introduce CCN, a novel point cloud architecture which can smooth and disrupt the adversarial perturbations. (2) We propose AMS, a novel *data augmentation* strategy to adaptively balance the model usability and robustness. Extensive evaluations indicate the integration of the two techniques provides much more robustness than existing defense solutions for 3D classification models. Our code can be found in https://github.com/GuanlinLee/CCNAMS.

1 Introduction

A point cloud is a popular representation of 3D objects and shapes. It consists of a set of data points with x, y and z coordinates to describe the external surface of an object. Interpreting point cloud data becomes important in many scenarios, *e.g.*, robotics [12], manufacturing [2], construction [19], *etc*. Recently, researchers designed new models based on Deep Neural Networks (DNNs) (*e.g.*, PointNet [21], DGCNN [27]) for 3D object classification, which achieve remarkable breakthrough over traditional methods.

Unfortunately, DNNs are well known to be vulnerable during training stage [31,32] and inference stage [5]. In the inference stage, DNNs are easy to be attacked by Adversarial Examples (AEs) [25], where imperceptible perturbations on a normal sample can mislead the model to make wrong predictions. Over the years, a plethora of attacks were designed to efficiently generate AEs [9,20].New techniques were further proposed to attack point cloud models [15,36,30]. Such vulnerabilities can significantly threaten the safe- and security-critical applications based on point cloud models.

Past works have extensively explored methods of defending 2D models against AEs. In contrast, how to enhance the robustness of 3D models is relatively less studied. The unique features of point cloud data and models increase the difficulty of model protection: (1) point clouds usually have irregular formats determined by the sensors for data collection; (2) adversaries have more choices to perform the attacks (*e.g.*, adding or removing points) in addition to changing the coordinate values; (3) 3D point clouds have a larger perturbation space than the 2D image space, resulting in more qualified AEs. These features make existing solutions less effective: they are not general enough to cover different types of adversarial attacks [15,37,17], or can be easily bypassed by adaptive attacks [18,24]. Hence, it is urgent but challenging to have a general and comprehensive defense mechanism.

In this paper, we propose new solutions to effectively defend point cloud classification models against AEs in two aspects. First, we design Context-Consistency dynamic graph Network (`CCN`), a new 3D network structure with higher adversarial robustness. It is able to dilute the noise in the adversarial samples, and make them closer to the clean samples in the feature space. Second, we introduce a new data augmentation strategy, named adaptive augmentation with Adversarial and Mix-up Samples (`AMS`). Researchers have proposed to train 3D point cloud models with adversarial examples [15] or mix-up sampling [6,34]. However, these methods cannot achieve comprehensive protection due to the variety of techniques in crafting AEs. Hence, we propose to augment the training set with different types of adversarial examples and mix-up samples. Simply incorporating all these data samples could easily affect the model accuracy over clean samples or overfit some specific attack. To balance the trade-off between model usability and robustness, we dynamically monitor the model's behaviors during training, and adaptively select the samples that can best improve the model performance. Compared to prior solutions that mainly focus on specific attacks, our solutions can achieve the *best adversarial robustness trade-off* among all types of attacks.

To assess the adversarial robustness of our two methodologies, we leverage the mutual information theory to theoretically explain the effectiveness of the proposed network architecture and training strategy. We also perform comprehensive evaluations over two commonly-used 3D point cloud datasets (ModelNet10 and ModelNet40) against four state-of-the-art white-box attacks and one black-box attack. Experimental results show that each solution exhibits advantages compared to the baselines with the same configurations. The integration of `CCN` and `AMS` outperforms existing solutions by about 8% on average adversarial accuracy.

2 Background and Related Works

2.1 Point Cloud Models

A point cloud is formally defined as an unordered set of points $x = \{x_i\}_{i=1}^{N}$, where $x_i \in \mathbb{R}^3$ is a 3D point with (*x, y, z*) coordinates, and N is the number of points. A point cloud classification model is thus a parameterized function $f_\theta : \mathcal{X} \mapsto \mathcal{Y}$ that predicts the corresponding label from a point cloud. Researchers have proposed different deep learning algorithms and neural networks to realize this classification task. We describe three common models. (1) PointNet [21]: this network consists of single variable-functions, a max pooling layer, and a function of the max pooled features to handle unordered points with arbitrary dimensions. It converts the point cloud data to feature vectors with fixed lengths, and then learns the labels. (2) PointNet++ [22]: this is a hierarchical neural network, which recursively applies PointNet over partitioned point sets to learn the local structures. Both of PointNet and PointNet++ adopt the coordinates of the points to produce the features. (3) DGCNN [27]: this Dynamic Graph Convolutional Neural Network integrates a new module EdgeConv to point cloud models. This module captures the local geometric structures by constructing a local graph and learning the embeddings for the edges. Then the integrated model can learn to semantically group the points for more accurate classification. Different from PointNet and PointNet++, DGCNN considers the neighbors of the points and adopts high-order features, *i.e.*, distances between adjacent points, to predict the labels. As a result, it gives higher robustness than the other two models. We also validate this conclusion in Section 5.3.

2.2 Adversarial Attacks against Point Clouds

The concept of adversarial examples was first proposed in [25], where the adversary tries to identify the imperceptible perturbation with the minimal scale to mislead the 2D image model. Then this attack was extended to the 3D point clouds with more techniques. Generally, these attacks can be classified into the following three categories:
Point perturbing. Similar to 2D image attacks, the adversary can slightly perturb the coordinates of certain critical points to fool the 3D model. Conventional approaches in 2D image tasks can be applied to 3D point clouds as well. For instance, Xiang et al. [30] adopted the C&W technique [5] to identify the optimal perturbing scale. Liu et al. [15] adopted the FGSM method [25] with various perturbation constraints to craft adversarial point clouds.
Point adding. The adversary can inject a small set of new points into the clean point cloud to attack the model. Xiang et al. [30] designed an initialize-and-shift approach to calculate the added points with their positions. Zhang et al. [35] proposed a point-wise gradient method to generate the optimal locations for point attachment.
Point dropping. The adversary can also remove some points from the original set to alter the model output. Zheng et al. [36] constructed the saliency map to identify the critical points and then drop them for attacks. A similar idea was also proposed in [35].

2.3 Adversarial Defenses for Point Clouds

A couple of approaches were proposed to defeat adversarial attacks against point clouds. They can be briefly summarized with the following categories.
Denoising point clouds. The basic idea is to cleanse the point cloud data and possibly remove the adversarial perturbations. For instance, Zhou et al. [37] designed a new structure DUP-Net, with the SOR operation to drop outliers in the input samples. However, it is only effective for point perturbing attacks, but fails to thwart point adding or dropping attacks. Dong et al. [8] designed a self-robust network with the self-attention mechanism to remove adversarial local features. These defense methods have also been defeated by new adaptive attacks [18,24].
Training robust models. Liu et al. [15] explored how to train a 3D point cloud model with adversarial examples generated by PGD. They concluded this strategy can beat

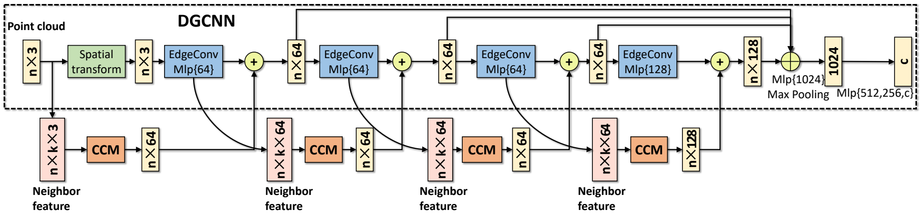


Fig. 1: Overview of CCN.

SOR and salient point removal approaches under certain attacks. Unfortunately, simple adversarial training based on PGD is not robust to cover all types of attacks, which will be demonstrated in our evaluation. Sun et al. [24] proposed a sorting-based parametric pooling operation to overcome the frangibility of default-used fixed pooling operations in point cloud models. Mix-up is a popular technique to augment training data with linear interpolations of feature vectors and labels to defeat 2D adversarial images [33]. This idea was then extended to the point cloud scenario, based on which researchers designed PointMixUP [6], and PointCutMix [34]. Our adaptive augmentation can outperform these purely mix-up strategies from the evaluation.

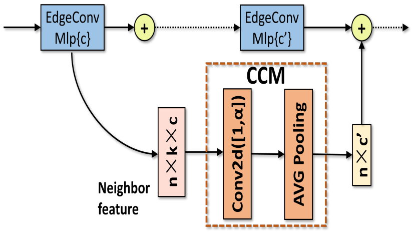
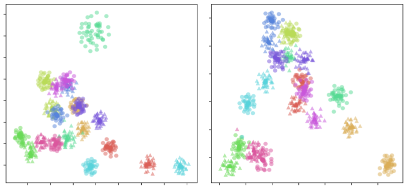


Fig. 2: The structure of CCM.



(a) DGCNN (b) CCN

Fig. 3: Feature map visualization.

Certified defenses. A couple of works designed certified defenses to defeat adversarial attacks in a theoretical way. For instance, Liu et al. [16] used a downsampling method to give an upper bound of the number of perturbed points. However, this method is time-consuming and needs clean inputs as guides, which is not practical. Lorenz et al. [17] studied the robustness of a model with transformations (e.g., rotating, shearing). They only considered the FGSM attack while ignoring other techniques.

3 Methodologies

In this section, we present two methodologies to protect the point cloud models against adversarial attacks: a new model structure and training strategy. Each method can enhance the model's adversarial robustness from a different perspective, and their combination serves as an effective defense solution.

3.1 Context-Consistency Dynamic Graph Network

From the aspect of model architecture, we design Context-Consistency dynamic graph Network (`CCN`), a new 3D model structure for robustness enhancement. The core insight behind our architecture is to decrease feature distances between clean and noisy samples with an adaptive denoising mechanism. Fig. 1 shows the structure overview. It is mainly built from the DGCNN model with the same spatial transform and EdgeConv layers. We choose DGCNN because it exhibits higher robustness than PointNet and PointNet++, due to the adoption of relation features, i.e., distances between points.

The key component of `CCN` is a lightweight Context-Consistency Module (CCM), which is inserted at many layers (Fig. 1). This module is responsible for collecting the features of point cloud data and diluting the potential adversarial noise, which can move the features of adversarial samples closer to that of clean ones. Since the adversarial noise in the feature space increases significantly at deeper layers (observed in Fig. 4), CCM extracts the neighbor features of every point (i.e., coordinates of each point's neighbors) before the next EdgeConv layer, which is relatively easier for denoising. The output of CCM will be combined with the output of the next EdgeConv layer.

Fig. 2 shows the detailed structure of CCM, which consists of a convolutional layer and a pooling layer. Specifically, (1) the 2D convolutional layer is used to simulate the function of the Edge layer to reduce the noise from model parameters. It has a receptive field size (*i.e.*, kernel size) of $[1, \alpha]$ to process the context information (*i.e.*, coordinates) in the neighbors (the closest points generated by the KNN [27]) of each point. It calculates new coordinates for the neighbors in the scope of α, which automatically learns to smooth the noise in the neighbor features. The sliding window in this convolutional layer can handle all the continuous scopes in the neighbor feature. In this way, the feature distance between adversarial and clean samples can be minimized. (2) The average pooling layer is following the convolutional layer to reduce the redundancy features. This function chooses the proper elements in the features based on their values. It can prevent noise accumulation during the model's forward propagation process by averaging elements in the features. (3) A residual connection transfers the adaptively selected context information to the output of the next EdgeConv layer. With such operation, CCM can keep the features between different layers (*i.e.*, contexts extracted from inputs) consistent. As a result, it can prevent the adversarial noise in the features from growing quickly at deeper layers.

It is worth noting that the receptive field size α in CCM can impact the model robustness when the order of inputs changes. This is because in a point cloud, the correlation between neighbor points is less tight than the correlation between neighbor pixels in a 2D image. Visiting too many points with a big receptive field can make the noise unacceptably large. On the other hand, using a small receptive field to visit very few points can make the information from points useless to calculate the correct coordinates. Currently, there are no theoretical guidelines for determining this hyperparameter, and we figure out this optimal value empirically in Section 5.1.

To demonstrate the effects of CCM, we use the t-SNE method to visualize the feature map of DGCNN and our `CCN` with the ModelNet40 dataset, as shown in Fig. 3. We randomly select 10 classes, and each class contains 50 point clouds (Visualization results for all the 40 classes can be found in the supplementary material). Different classes are represented with different colors. We use circles and triangles to denote the clean and perturbed point clouds, respectively. From Fig. 3a, we can see that in DGCNN, some perturbed data are far from the clean data in the same class, or even overlapped with data from other classes. This implies misclassification for those data. In contrast, for `CCN` (Fig. 3b), the perturbed and clean data in the same class are much closer, and there is less overlap among different classes. This indicates that CCM can effectively remove the noise, making the perturbed and clean data much closer in the feature space.

3.2 Adaptive Augmentation with Adversarial and Mix-up Samples

In addition to `CCN`, we also introduce a novel data augmentation strategy to enhance the robustness of a point cloud model. In the conventional 2D image tasks, there are generally two types of training strategies for defeating adversarial examples. Unfortunately, they cannot achieve satisfactory performance when extended to 3D point cloud models. The first strategy is adversarial training, which augments the training set with adversarial examples crafted by the PGD technique. However, there are essentially various types of methods to generate adversarial point clouds with distinct features. Adversarial training with one type of AEs cannot provide comprehensive protection for other types of attacks [15], while simply incorporating all these sorts of AEs can significantly harm the model accuracy for clean samples. The second strategy is to mix up clean samples with different labels for model training [33]. This strategy is applied to the point cloud classification [6,34], which have limited robustness improvement.

Our adaptive augmentation strategy (`AMS`) considers the adversarial examples (of different types), mix-up samples as well as clean samples for model training. However, it is challenging to decide the type and quantity of samples to be used before the training task, as the training process is dynamic and relatively random. To overcome this challenge, `AMS` adaptively selects the desired samples in each epoch based on the current model. This dynamic selection can efficiently balance the model robustness and accuracy over clean samples for the complex 3D point cloud classification tasks.

Our training algorithm is shown in Algorithm 1. At every training epoch, for each batch (X, Y) from the training set Q, we first generate three types of batches from each sample in the batch[1]: (1) X_{drop} is a batch of AEs with the point dropping technique using the function $\texttt{AE-Gen}^{\texttt{drop}}$; (2) X_{perturb} is a batch of AEs with the point perturbing technique using the function $\texttt{AE-Gen}^{\texttt{perturb}}$; (3) X_{mix} is a batch of mix-up samples with the corresponding mix-up labels Y_{mix} using the function `MS-Gen`. Second, we compute the accuracy of clean and AE batches from the current model, as acc_x, acc_{drop} and acc_{perturb}, respectively. We compute $acc_{\text{min}} = \min(acc_{\text{drop}}, acc_{\text{perturb}})$, and compare it with the weighted mean accuracy of the clean batches $acc_{\text{avg}} = T * \text{mean}(acc)$, where acc is a collection of clean accuracy acc_x at the current training epoch. If acc_{min} is higher than acc_{avg}, then this model is regarded as robust enough to defend against different types of AEs. So we perform *mix-up augmentation* to improve the model's generalization and utility, *i.e.*, training the model with the clean batch (X, Y) and mix-up batch $(X_{\text{mix}}, Y_{\text{mix}})$. Otherwise, we perform *adversarial augmentation* to improve

[1] We do not consider point adding as the generation complexity is extremely high. Experiments show the incorporation of the other two AEs can defeat the point adding AEs as well.

Algorithm 1: Adaptive Augmentation with Adversarial and Mix-up Samples.

```
Input  : $Q$: point cloud training set
Output: $M$: robust point cloud model
Initialize($M$);
foreach training epoch do
    $acc = []$;
    foreach batch $(X, Y) \sim Q$ do
        $X_{\text{perturb}} = \texttt{AE-GEN}^{\texttt{perturb}}(X, Y, M)$;
        $X_{\text{drop}} = \texttt{AE-GEN}^{\texttt{drop}}(X, Y, M)$;
        $(X_{\text{mix}}, Y_{\text{mix}}) = \texttt{MS-GEN}(X, Y)$;
        calculate accuracy $acc_x$, $acc_{\text{perturb}}$ and $acc_{\text{drop}}$ for $X$, $X_{\text{perturb}}$ and $X_{\text{drop}}$;
        $acc.\text{append}(acc_x)$, $acc_{\text{min}} = \min(acc_{\text{perturb}}, acc_{\text{drop}})$;
        if $acc_{\text{min}} > T * \text{mean}(acc)$ then
            train $M$ with $(X, Y)$ and $(X_{\text{mix}}, Y_{\text{mix}})$;
        else
            train $M$ with $(X_{\text{perturb}}, Y)$, $(X_{\text{drop}}, Y)$ and $(X, Y)$;
return $M$
```

the model's robustness, *i.e.*, training it with the clean batch (X, Y) and two types of adversarial batches (X_{drop}, Y), (X_{perturb}, Y).

In practice, we implement `MS-Gen` with the PointCutMix approach [34]. We adopt the Saliency Map Attack [36] to craft AEs by dropping points for $\texttt{AE-Gen}^{\texttt{drop}}$. For $\texttt{AE-Gen}^{\texttt{perturb}}$, we utilize the 3D L_∞-BIM technique [14], which is a basic version of L_∞-PGD [30][2]. Besides, we calculate the averaged accuracy of acc_x for clean samples to avoid overfitting of T on a specific model and make the algorithm better generalize to other models. The optimal value of T will be empirically determined in Section 5.2.

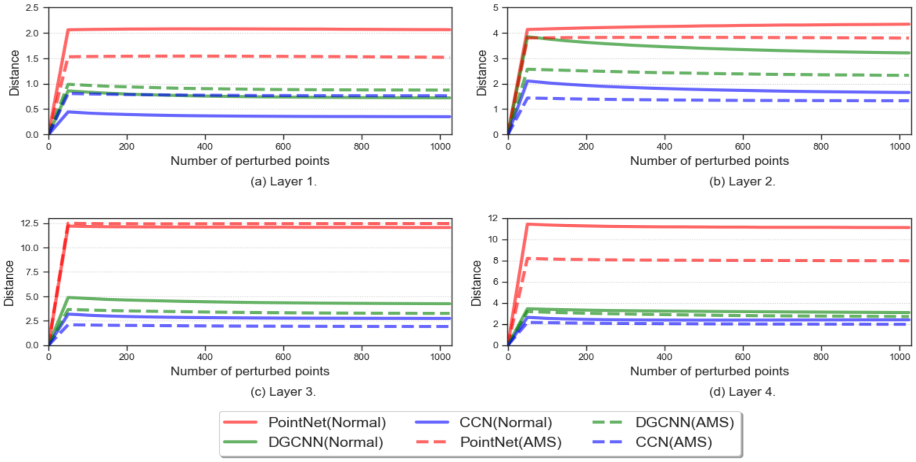
For training complexity, compared to the pure adversarial training strategies, we need to generate two types of AEs for each clean sample. To keep the same training cost, we craft each AE with half the number of iterations. Our experiments in Section 5.2 indicate that `AMS` can help the model obtain higher robustness than conventional adversarial training methods under the same computational complexity constraint.

4 Explaining the Effectiveness of Our Methodologies

In this section, we perform an in-depth analysis to understand why our proposed solutions can improve the model robustness. Past works have developed frameworks to study the vulnerability of adversarial examples for 2D image models based on the mutual information theory [11,38]. Inspired by those frameworks, we aim to disclose the factors that can affect the robustness of point cloud models.

Specifically, we apply mutual information to calculate the correlation between the features of perturbed and clean point clouds. A high correlation indicates the feature context of noisy data is more consistent with that of clean data, and the model is more

[2] We do not use L_∞-PGD because when we randomly project the point cloud to an initialization position, the model has a high chance to give a wrong prediction initially, and the adversary will obtain less useful information than starting from the original position.



(a) Layer 1.

(b) Layer 2.

(c) Layer 3.

(d) Layer 4.

Fig. 4: Cosine distance of features between clean and perturbed samples at different layers. The perturbation is generated from a Gaussian distribution ($\mu = 0$, $\sigma = 0.05$).

robust to predict correct labels from noisy samples. However, it is computationally infeasible to directly calculate such mutual information, due to the high dimensions of the input space and feature space. Alternatively, we can estimate the mutual information with a substitute measurement, *i.e.*, the k-Measurement M_k. This measurement is based on the cosine distance, which can represent both the direction and magnitude of a distance in a high dimension at the same time. Formally, we have the following definition for k-Measurement:

Definition 1 (k-Measurement) *Let f be a function that maps a point cloud to the feature space: $\{X_i | X_i \in R^d, i \in [N]\} \mapsto \{Y_i | Y_i \in R^D, i \in [N]\}$, where d is the dimension of the point coordinate in point clouds, D is the dimension of f's outputs for each point and N is the number of points in a point cloud. Consider a clean point cloud $S = \{X_i | X_i = (x_i, y_i, z_i), i \in [N]\}$. S_k is a perturbed point cloud with k different points compared with S,* i.e., *$S_k = \{X_{j_i} | X_{j_i} = (x_{j_i}, y_{j_i}, z_{j_i}) \in S, j_i \in [N-k]\} \cup \{X_{h_i} + \epsilon_{h_i} | X_{h_i} = (x_{h_i}, y_{h_i}, z_{h_i}) \in S, \epsilon_{h_i} = (\epsilon_{0,h_i}, \epsilon_{1,h_i}, \epsilon_{2,h_i}), h_i \in [k]\}$. Then the k-Measurement M_k for f, S and S_k are defined as:*

$$f(S) = \frac{\sum_{i=1}^{N} f(S)_i}{N}, f(S_k) = \frac{\sum_{i=1}^{N} f(S_k)_i}{N}$$
$$M_k(f, S, S_k) = 1 - \frac{f(S) \cdot f(S_k)}{\|f(S)\| \|f(S_k)\|}$$

We introduce a general theorem to prove that under the same value of k, a small $M_k(f, S, S_k)$ implies a large mutual information $I(S_K, f(S))$. The proof can be found in the supplementary material.

Theorem 1 *Let f be a function that maps a point cloud to the feature space, and Q be the distribution of clean point clouds. S is sampled from Q. $Q^k(S, \epsilon)$ is the distribution of noisy point clouds, in which each element S_k is perturbed from S with an additional* *noise ϵ, and the difference of numbers of points between S and S_k is smaller than a constant k,* i.e.*, $-k \leq |S_k| - |S| \leq k$. Then for every $S \sim Q$ and $S_k \sim Q^k(S, \epsilon)$, the mutual information $I(S_k, f(S))$ has a lower bound, which is negatively correlated with the k-measurement $M_k(f, S, S_k)$.*

Theorem 1 can help us establish a connection between $M_k(f, S, S_k)$ and model robustness. A small k-measurement $M_k(f, S, S_k)$ could increase the mutual information value $I(S_k, f(S))$. According to the observation in [38], with a larger $I(S_k, f(S))$, the corresponding point cloud model is more robust, as the confidence of correctly predicting $f(S)$ from the perturbed sample S_k is higher. Therefore, *a small k-measurement $M_k(f, S, S_k)$ indicates a more robust point cloud model.*

With the above conclusion, we now explain the mechanisms of our proposed strategies. For the network architecture, as described in Section 3.1, the introduction of CCM is to increase the similarity of clean and perturbed samples in the feature space, which can lead to a small $M_k(f, S, S_k)$. For model training, our AMS adopts both AEs and mix-up samples for data augmentation. According to [1], training with AEs can be regarded as the process of feature purification, which can purify the non-robust direction in the features and build a tight connection between features and correct labels, i.e., increasing the mutual information between features and labels and decreasing M_k. From [33], mix-up samples can provide a generic vicinal distribution, and sampling from such a distribution can generate virtual feature-target vector pairs to force the model to minimize the Empirical Risk, which equals to minimizing M_k. We adaptively select different kinds of samples based on the model behaviors, which can take advantage of both methods and further reduce the k-measurement.

We also empirically verify the effectiveness of CCN and AMS in reducing the k-measurement. We consider three network architectures[3]: PointNet has five convolutional layers, with the first four used for feature extraction; both DGCNN and CCN have four EdgeConv layers for feature extraction. We compute the cosine distances between the features of clean and perturbed point clouds at these four layers[4]. Fig. 4 compares the differences of different architectures and training strategies versus the number of perturbed points in clean point clouds. We have the following observations. (1) CCN and models trained with AMS always give smaller distances, indicating their efficacy in increasing the mutual information and enhancing the robustness. (2) For each layer, the distance decreases as the number of perturbed points increases. This is because when more points are perturbed in the point cloud, its distribution is closer to some augmented clean samples during the model training, leading to relatively smaller distance. (3) The distance increases from layer 1 to layer 3, indicating the noise is amplified at deeper layers. Therefore, we adopt the neighbor features from the previous EdgeConv instead of the current one in CCM.

5 Evaluations

We perform extensive experiments to evaluate our solutions. Below, we describe the detailed experimental setup.

[3] The results for PointNet++ can be found in the supplementary material

[4] For CCN, we choose $\alpha = 4$, which is identified in Section 5.1.

Network	**Clean**	**Adversarial Examples**					
Structure	**Sample**	SMA−40	APP	AIC	AIH	**AAUA**	**LAUA**
PointNet	90.83	63.11	**82.55**	76.14	**69.56**	72.84	63.11
DGCNN	91.88	79.91	74.88	76.01	68.47	74.82	**68.47**
CCN (α=20)	92.71	**82.04**	80.21	73.78	66.44	75.62	66.44
CCN (α=16)	92.53	80.80	78.21	73.70	64.41	74.28	64.41
CCN (α=12)	**92.74**	80.64	79.55	75.04	66.44	75.42	66.44
CCN (α=8)	92.05	81.66	78.81	71.14	63.92	73.88	63.92
CCN (α=4)	92.25	81.17	79.46	**76.46**	68.30	**76.35**	68.30
CCN (α=1)	92.37	80.60	78.45	74.88	66.60	75.13	66.60

Table 1: Results for different architectures and hyperparameters (%).

Datasets and Models. We perform comprehensive experiments to validate the effectiveness of CCN, AMS, and their combination. We mainly consider the PointNet and DGCNN models. The evaluation results for PointNet++ give the same conclusion, and can be found in the supplementary material. We adopt the ModelNet40 dataset [29], which contains 12,311 CAD objects from 40 different classes. These objects are split into a training set of 9,843 samples and a test set of 2,468 samples. For the training process, all the models are trained for 250 epochs with a learning rate of 0.001 and the Adam optimizer [13]. The size of an input point cloud is 1,024 * 3, *i.e.*, there are 1,024 points in each point cloud with three coordinates. We also perform evaluations on ModelNet10, a subset of ModelNet40, and the results can be found in the supplementary material. Note that there are also some more realistic point cloud datasets (e.g., ScanNet [7], ScanObjectNN [26]). We do not consider them as currently there are no works evaluating the attacks and defenses over them, and the feasibility of attacking these datasets is unknown. We will consider this as future work.

Attacks. We find most of previous works only focus on the point perturbing attack. In contrast, we also consider point adding, point dropping and black-box adversarial attacks. We test five state-of-the-art adversarial attacks (four white-box and one black-box). All of them are implemented as untargeted attacks. Specifically,

- SMA$-k$ [36] is a point dropping attack which drops $5\times k$ points in k iterations based on the saliency map.
- APP [30] is a point perturbing attack which shifts points with 10 binary searches and 100 iterations for each search to craft AEs. Note that the $GeoA^3$ attack [28] can be regarded as APP with an external curve loss to attack models equipped with SOR. Since our CCN and AMS do not adopt SOR, evaluations on APP and $GeoA^3$ will be the same. So we do not specifically consider $GeoA^3$.
- AIC [30] is a point adding attack which conducts 10 binary searches and 100 iterations for each search to add 512 points to the point cloud. Chamfer distance is adopted to measure the point locations.
- AIH [30] is similar as AIC with the Hausdorff distance.
- BIM$-k$[5] is a point perturbing attack using the L_∞ basic iterative method: each sample is generated with k iterations, $\epsilon = 0.03$ and step size $= 0.0005$. We do not adopt the PGD attack as the adversarial point cloud will get disrupted at the beginning

[5] Results can be found in supplementary materials.

of sample generation. Different from 2D image attacks, a little perturbation in 3D point cloud can change the shape of the original object significantly. We find when $\epsilon = 0.03$ under L_∞-norm, the point clouds are difficult for humans to recognize, so we do not use the PGD attack to avoid the disruption of point clouds at the start.

- `AdvPC` [10] is a state-of-the-art black-box attack with higher transferability than others. We follow the same hyperparameters in the original paper, and use a larger number of iterations (500) to improve its performance.

It is also worth noting that there are some physical attacks against point cloud models (e.g., attacking the LiDAR sensor in autonomous driving [4,3]). Those attacks are very different from our focus with physical constraints. The defenses against them are beyond the scope of this paper.

Baselines. We select a couple of baseline methods to compare with our solution. (1) For the ablation study of `CCN`, we choose the conventional PointNet and DGCNN as the baselines. (2) For the ablation study of `AMS`, we compare it with normal training, adversarial training and mix-up training. For adversarial training, we consider two strategies: AT-BIM trains the model using the 3D L_∞-BIM point perturbing technique [14], with the configurations of 20 iterations, $\epsilon = 0.02$ and step $= 0.005$; AT-SMA trains the model using the point dropping technique [36], with the configurations of 20 iterations and 5 points dropped in each iteration. For mix-up training, we select PointCutMix-K [34], as it achieves the highest robustness in the white-box scenario. (3) For evaluating the integration of the two techniques, we consider the following state-of-the-art solutions: adversarial training (AT-BIM and AT-SMA); mix-up training (PointCutMix-R and PointCutMix-R [34]), SRS [35], SOR with the configuration of k=2 and α=1.1 [23] and DUP-Net [37]. Since these solutions target different phases in the model pipeline, some of them can be integrated to further enhance the model robustness, which we will consider as well in our evaluations.

Metrics. We measure the model accuracy over clean samples and different types of adversarial examples for its usability and robustness, respectively. For adversarial robustness, (1) **AAUA** measures the Average Accuracy Under Attacks in our consideration; (2) **LAUA** measures the Lowest Accuracy Under Attacks, which is the worst situation. Formally, we consider n different attacks. For each attack i, we measure the model's accuracy over the generated AEs as acc_i. The two metrics can be calculated as:

$$\textbf{AAUA} = \frac{\sum_{i=1}^{n} acc_i}{n}, \quad \textbf{LAUA} = \min\{acc_i\}, i \in [n]$$

5.1 Ablation Study of `CCN`

As discussed in Section 3.1, the size α of the receptive field in CCM can affect the model's robustness against different types of attacks. We first perform ablation studies on the hyperparameter α. We compare the performance of our `CCN` for different α values with PointNet and DGCNN under four white-box attacks. Each model is trained with PointCutMix-K, which gives the best results compared to other training strategies except our `AMS`. Table 1 presents the results. First, we observe that PointNet has the best robustness against the point perturbing attack (`APP`) and adding attack (`AIH`), as

Training	Clean	Adversarial Examples					
Strategy	Sample	SMA−40	APP	AIC	AIH	**AAUA**	**LAUA**
Normal	88.76	41.88	55.64	49.68	43.43	47.66	41.88
PointCutMix-K	**90.83**	63.11	82.55	76.14	69.56	72.84	63.11
AT-BIM	88.23	45.41	85.39	84.98	86.36	75.54	45.41
AT-SMA	87.38	**67.37**	79.79	75.73	74.92	74.45	**67.37**
AMS (T=0.7)	88.64	51.30	86.69	85.31	85.96	77.32	51.30
AMS (T=0.5)	89.45	48.99	87.01	**86.49**	**87.26**	**77.44**	48.99
AMS (T=0.3)	89.65	46.02	**87.30**	86.00	86.77	76.52	46.02
AMS (T=0.1)	89.20	42.98	80.24	79.87	81.01	71.03	42.98

Table 2: Results for different training methods and hyperparameters with PointNet (%).

Network	Training	Clean	Adversarial Examples					
Structure	Strategy	Sample	SMA-40	APP	AIC	AIH	**AAUA**	**LAUA**
CCN	Normal	90.87	67.94	57.47	61.04	53.37	59.96	53.37
	AT-SMA	90.75	**84.17**	74.51	70.05	64.98	73.43	64.98
	AT-BIM	90.05	67.37	88.80	83.77	79.75	79.92	67.37
	PointCutMix-K	92.25	81.17	79.46	76.46	68.30	76.35	68.30
	AMS	**92.41**	77.72	**90.50**	**86.09**	**84.05**	**84.74**	**77.72**

Table 3: Results for different training strategies with CCN (%).

it only uses individual points to generate features, avoiding the noise accumulation. However, it has very bad performance for the point dropping attack (SMA−40). Second, our CCN provides more satisfactory accuracy for both clean and adversarial examples. The accuracy values for different AEs change with the hyperparameters, and $\alpha = 4$ can give the best trade-off considering all the point adding, dropping and perturbing attacks. For **AAUA**, it is 3.51% higher than PointNet, and 1.53% higher than DGCNN. For **LAUA**, it is 5.19% higher than PointNet, and only 0.17% lower than DGCNN. **We conclude that CCN is a more robust architecture than PointNet and DGCNN when we comprehensively consider all the types of attacks.**

5.2 Ablation Study of AMS

Next, we focus on the evaluation of our adaptive augmentation strategy. One important hyperparameter in AMS is T, which determines the kind of batch samples for training. We perform an ablation study to select the optimal T value. We use the PointNet model, which is simple and easy to obtain the results. We generate X_{drop} using the Saliency Map Attack (10 iterations, 10 points dropped in each iteration) and X_{perturb} using the 3D L_∞-BIM attack (10 iterations, $\epsilon = 0.02$ and step $= 0.005$). $(X_{\text{mix}}, Y_{\text{mix}})$ are generated by PointCutMix-K. Four white-box attacks are used for evaluation. Table 2 presents the accuracy of models trained with different strategies.

From Table 2, we observe that PointCutMix-K can achieve high accuracy over clean samples and AEs generated from SMA−40. However, it behaves much worse under the other three attacks. For AMS, the value of T can affect the model accuracy over different types of samples. With $T = 0.5$, the model has the highest robustness against AIC and AIH attacks. Although **LAUA** in this configuration is lower than PointCutMix-K and

Defense Solutions	**Clean**	**Adversarial Examples**					
	Sample	SMA−40	APP	AIC	AIH	**AAUA**	**LAUA**
PointNet + Normal	88.76	41.88	55.64	49.68	43.43	47.66	41.88
PointNet + PointCutMix-K	90.83	63.11	82.55	76.14	69.56	72.84	63.11
PointNet + AT-BIM	88.23	45.41	85.39	84.98	86.36	75.54	45.41
PointNet + AT-SMA	87.38	67.37	79.79	75.73	74.92	74.45	67.37
DGCNN + Normal	91.03	65.87	46.10	54.06	48.78	53.70	46.10
DGCNN + PointCutMix-R	90.91	72.65	71.63	62.26	56.53	65.77	56.53
DGCNN + PointCutMix-K	91.88	79.91	74.88	76.01	68.47	74.82	68.47
DGCNN + AT-BIM	91.27	66.68	89.98	81.37	76.99	78.76	66.68
DGCNN + AT-SMA	91.80	**84.66**	72.00	71.75	64.25	73.17	64.25
DGCNN + SOR + Normal	91.00	66.00	86.83	51.82	54.38	64.76	51.82
DGCNN + SOR + AT-BIM	91.77	65.52	84.97	58.59	58.27	66.84	58.27
DGCNN + SOR + AT-SMA	91.05	80.59	86.91	59.85	60.25	71.90	59.85
SRS*	83.00	35.10	64.70	59.50	58.80	54.53	35.10
DUP-Net*	86.30	43.70	84.50	61.40	62.70	63.08	43.70
PointNet + AMS	89.45	48.99	87.01	**86.49**	**87.26**	77.44	48.99
DGCNN + AMS	92.21	75.41	**90.83**	85.47	83.93	83.91	75.41
CCN + AMS	**92.41**	77.72	90.50	86.09	84.05	**84.74**	**77.72**

Table 4: Results for different solutions under the white-box attacks (%). *Data of SRS and DUP-Net are adopted from [37].

AT-SMA (due to the bad performance in SMA−40), the average accuracy **AAUA** is still the highest. This validates the advantage of AMS, and we will adopt $T = 0.5$ for the following experiments. Table 3 shows the similar comparisons of training strategies with the CCN architecture. AMS gives much higher **AAUA** and **LAUA** than others.

5.3 End-to-End Evaluations and Comparisons

After identifying the optimal hyperparameters, we comprehensively compare our two methodologies and their integration with existing works of different network architectures (PointNet, DGCNN, DGCNN with SOR, SRS and DUP-Net) and training strategies (Normal, PointCutMix-R, PointCutMix-K, AT-BIM, AT-SMA).

Table 4 summarizes the comparison results for white-box attacks. There can be a lot of combinations with these solutions. Since PointNet has the least robustness among these architectures, we mainly compare the DGCNN architecture. First, we observe that our solution achieves the highest accuracy over clean samples. Second, for adversarial attacks, our solution also gives the best result for APP and AIC attacks. For SMA−40, our solution is worse than DGCNN+AT-SMA; for AIH, our solution is slightly worse than PointNet + AT-BIM. Nevertheless, it still gives the highest **AAUA** and **LAUA**, due to its comprehensive robustness. Furthermore, PointNet and DGCNN trained with our AMS can outperform other defense solutions when using the same model.

We further evaluate our methodologies against a black-box attack (AdvPC). The adversary crafts AEs from a different source model, and then leverages the transferability to attack the target victim model. We consider two constraints to generate AEs for testing. The results are shown in Table 5. We observe that for the source model of DGCNN with $\epsilon = 0.18$, the integration (CCN + AMS) is slightly worse than DGCNN + AT-BIM. For the source model of DGCNN with $\epsilon = 0.45$, our solution is slightly worse

Target Model	**Defense**	**Source Model**					
		PointNet		DGCNN		CCN	
		$\epsilon = 0.18$	$\epsilon = 0.45$	$\epsilon = 0.18$	$\epsilon = 0.45$	$\epsilon = 0.18$	$\epsilon = 0.45$
PointNet	Normal	84.50	84.50	86.27	86.27	86.98	85.56
	Normal + SRS	81.61	82.48	82.99	84.32	83.88	85.65
	Normal + SOR	65.46	65.90	67.64	67.90	68.08	69.78
	AT-BIM	86.11	84.70	87.88	87.88	85.76	86.82
	AT-BIM + SRS	80.52	83.93	83.05	84.55	81.17	84.26
	AT-BIM + SOR	74.36	74.26	73.38	73.67	72.06	74.11
	AMS	86.59	85.51	88.02	86.95	88.02	88.02
DGCNN	Normal	89.21	89.21	87.02	86.30	83.38	85.57
	Normal + SRS	61.88	60.97	63.72	59.61	62.97	58.24
	Normal + SOR	39.13	34.58	33.67	35.95	33.67	35.95
	AT-BIM	89.08	90.54	**89.81**	89.08	88.71	89.08
	AT-BIM + SRS	63.32	63.78	57.36	63.43	63.43	63.25
	AT-BIM + SOR	38.54	38.54	37.63	40.38	39.15	40.62
	AMS	88.89	89.26	89.63	89.63	90.00	**90.73**
CCN	Normal	89.42	88.33	89.05	88.33	85.42	85.42
	Normal + SRS	70.42	65.88	64.52	66.79	64.97	67.24
	Normal + SOR	41.80	43.16	41.80	40.95	42.71	42.25
	AT-BIM	88.61	88.61	88.61	88.25	88.25	88.97
	AT-BIM + SRS	64.53	64.39	60.78	63.04	63.94	68.85
	AT-BIM + SOR	47.25	46.38	45.48	46.83	43.22	45.90
	AMS	**90.93**	**90.56**	89.45	**90.19**	**90.56**	90.19

Table 5: Results for different solutions under the black-box attacks (%).

than DGCNN + AMS. For the rest of cases, it gives the highest accuracy. This indicates the effectiveness of our proposed solution under the black-box attack.

We compare our methodologies with more baselines, model architectures and attack configurations. The results can be found in the supplementary material. All the results confirm that our proposed CCN trained with AMS has the best robustness against different types of AEs.

6 Conclusion

Numerous research works have been done to increase our understanding about the inherent features of adversarial examples and model robustness in 2D image tasks. However, studies of adversarial defenses in the point cloud domain are still at an early stage. We advance this research direction with two contributions. For network architecture, we propose CCN, which can denoise the adversarial point clouds and smooth the perturbations in the feature space. For model training, we propose AMS, which can adaptively select clean, mix-up or adversarial samples to balance the model utility and robustness. Comprehensive evaluations show that our solution outperforms a variety of baselines under different types of white-box and black-box attacks.

Acknowledgement. This work is supported under the RIE2020 Industry Alignment Fund–Industry Collaboration Projects (IAF-ICP) Funding Initiative, as well as cash and in-kind contributions from the industry partner(s). It is also supported in part by Singapore Ministry of Education (MOE) AcRF Tier 2 MOE-T2EP20121-0006 and AcRF Tier 1 RS02/19.

References

1. Allen-Zhu, Z., Li, Y.: Feature purification: How adversarial training performs robust deep learning. CoRR **abs/2005.10190** (2020)
2. Arsalan Soltani, A., Huang, H., Wu, J., Kulkarni, T.D., Tenenbaum, J.B.: Synthesizing 3d shapes via modeling multi-view depth maps and silhouettes with deep generative networks. In: Proc. of the CVPR. pp. 1511–1519 (2017)
3. Cao, Y., Wang, N., Xiao, C., Yang, D., Fang, J., Yang, R., Chen, Q.A., Liu, M., Li, B.: Invisible for both camera and lidar: Security of multi-sensor fusion based perception in autonomous driving under physical-world attacks. In: 2021 IEEE Symposium on Security and Privacy (SP). pp. 176–194. IEEE (2021)
4. Cao, Y., Xiao, C., Cyr, B., Zhou, Y., Park, W., Rampazzi, S., Chen, Q.A., Fu, K., Mao, Z.M.: Adversarial sensor attack on lidar-based perception in autonomous driving. In: Proceedings of the 2019 ACM SIGSAC conference on computer and communications security. pp. 2267–2281 (2019)
5. Carlini, N., Wagner, D.: Towards Evaluating the Robustness of Neural Networks. In: Proc. of the S&P. pp. 39–57 (2017)
6. Chen, Y., Hu, V.T., Gavves, E., Mensink, T., Mettes, P., Yang, P., Snoek, C.G.M.: Point-Mixup: Augmentation for Point Clouds. In: Proc. of the ECCV. pp. 330–345 (2020)
7. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5828–5839 (2017)
8. Dong, X., Chen, D., Zhou, H., Hua, G., Zhang, W., Yu, N.: Self-robust 3d point recognition via gather-vector guidance. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11513–11521. IEEE (2020)
9. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: Proc. of the ICLR (2015)
10. Hamdi, A., Rojas, S., Thabet, A.K., Ghanem, B.: Advpc: Transferable adversarial perturbations on 3d point clouds. In: Proc. of the ECCV. vol. 12357, pp. 241–257 (2020)
11. Hjelm, R.D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A., Bengio, Y.: Learning deep representations by mutual information estimation and maximization. arXiv preprint arXiv:1808.06670 (2018)
12. Kim, P., Chen, J., Cho, Y.K.: Slam-driven robotic mapping and registration of 3d point clouds. Automation in Construction **89**, 38–48 (2018)
13. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Proc. of the ICLR (2015)
14. Kurakin, A., Goodfellow, I.J., Bengio, S.: Adversarial examples in the physical world. In: Proc. of the ICLR (Workshop) (2017)
15. Liu, D., Yu, R., Su, H.: Extending Adversarial Attacks and Defenses to Deep 3D Point Cloud Classifiers. In: Proc. of the ICIP (2019)
16. Liu, H., Jia, J., Gong, N.Z.: Pointguard: Provably robust 3d point cloud classification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6186–6195 (2021)
17. Lorenz, T., Ruoss, A., Balunović, M., Singh, G., Vechev, M.: Robustness certification for point cloud models. arXiv preprint arXiv:2103.16652 (2021)
18. Ma, C., Meng, W., Wu, B., Xu, S., Zhang, X.: Efficient Joint Gradient Based Attack Against SOR Defense for 3D Point Cloud Classification. In: Proc. of the MM. pp. 1819–1827 (2020)
19. Macher, H., Landes, T., Grussenmeyer, P.: From point clouds to building information models: 3d semi-automatic reconstruction of indoors of existing buildings. Applied Sciences **7**(10), 1030 (2017)

20. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards Deep Learning Models Resistant to Adversarial Attacks. In: Proc. of the ICLR (2018)
21. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In: Proc. of the CVPR (2017)
22. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Proc. of the NIPS. pp. 5099–5108 (2017)
23. Rusu, R.B., Marton, Z.C., Blodow, N., Dolha, M.E., Beetz, M.: Towards 3D Point cloud based object maps for household environments. Robotics Auton. Syst. **56**(11), 927–941 (2008)
24. Sun, J., Koenig, K., Cao, Y., Chen, Q.A., Mao, Z.M.: On adversarial robustness of 3d point cloud classification under adaptive attacks. arXiv preprint arXiv:2011.11922 (2020)
25. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing Properties of Neural Networks. In: Proc. of the ICLR (2014)
26. Uy, M.A., Pham, Q.H., Hua, B.S., Nguyen, T., Yeung, S.K.: Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1588–1597 (2019)
27. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic Graph CNN for Learning on Point Clouds. ACM Trans. Graph. **38**(5), 146:1–146:12 (2019)
28. Wen, Y., Lin, J., Chen, K., Jia, K.: Geometry-aware Generation of Adversarial and Cooperative Point Clouds. CoRR **abs/1912.11171** (2019)
29. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3D ShapeNets: A deep representation for volumetric shapes. In: Proc. of the CVPR (2015)
30. Xiang, C., Qi, C.R., Li, B.: Generating 3D Adversarial Point Clouds. In: Proc. of the CVPR (2019)
31. Xu, G., Li, H., Liu, S., Yang, K., Lin, X.: Verifynet: Secure and verifiable federated learning. IEEE Transactions on Information Forensics and Security **15**, 911–926 (2020)
32. Xu, G., Li, H., Zhang, Y., Xu, S., Ning, J., Deng, R.H.: Privacy-preserving federated deep learning with irregular users. IEEE Transactions on Dependable and Secure Computing **19**(2), 1364–1381 (2022)
33. Zhang, H., Cissé, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond Empirical Risk Minimization. In: Proc. of the ICLR (2018)
34. Zhang, J., Chen, L., Ouyang, B., Liu, B., Zhu, J., Chen, Y., Meng, Y., Wu, D.: PointCutMix: Regularization Strategy for Point Cloud Classification. CoRR **abs/2101.01461** (2021)
35. Zhang, Q., Yang, J., Fang, R., Ni, B., Liu, J., Tian, Q.: Adversarial attack and defense on point sets. CoRR **abs/1902.10899** (2019)
36. Zheng, T., Chen, C., Yuan, J., Li, B., Ren, K.: PointCloud Saliency Maps. In: Proc. of the ICCV (2019)
37. Zhou, H., Chen, K., Zhang, W., Fang, H., Zhou, W., Yu, N.: DUP-Net: Denoiser and Upsampler Network for 3D Adversarial Point Clouds Defense. In: Proc. of the ICCV (2019)
38. Zhu, S., Zhang, X., Evans, D.: Learning Adversarially Robust Representations via Worst-Case Mutual Information Maximization. In: Proc. of the ICML. pp. 11609–11618 (2020)

Supplementary Materials of Improving Adversarial Robustness of 3D Point Cloud Classification Models

Guanlin Li[1,2], Guowen Xu[1,*], Han Qiu[3], Ruan He[4], Jiwei Li[5,6], and Tianwei Zhang[1]

[1] Nanyang Technological University, [2] S-Lab, NTU,
[3] Tsinghua University, [4] Tencent, [5] Shannon.AI, [6] Zhejiang University,
{guanlin001,guowen.xu, tianwei.zhang}@ntu.edu.sg,
qiuhan@tsinghua.edu.cn, ruanhe@tencent.com,
jiwei_li@shannonai.com
[*] Corresponding author

1 Mutual Information Maximization

Theorem 1 *Let f be a function that maps a point cloud to the feature space, and Q be the distribution of clean point clouds. S is sampled from Q. $Q^k(S,\epsilon)$ is the distribution of noisy point clouds, in which each element S_k is perturbed from S with an additional noise ϵ, and the difference of numbers of points between S and S_k is smaller than a constant k,* i.e.*, $-k \leq |S_k| - |S| \leq k$. Then for every $S \sim Q$ and $S_k \sim Q^k(S,\epsilon)$, the mutual information $I(S_k, f(S))$ has a lower bound, which is negatively correlated with the k-measurement $M_k(f,S,S_k)$.*

Proof. According to the definition of mutual information, we have the following equation:

$$I(S_k, f(S)) = H(f(S)) - H(f(S)|S_k) =$$
$$-\sum_{1}^{|Q|} \frac{1}{|Q|} \log f(S) - H(f(S)|S_k).$$

We use $B(S,\epsilon)$ to denote a hyper-sphere whose center is S and radius is $\|\epsilon\|$. So the second term of the above expression can be rewritten as follows:

$$-H(f(S)|S_k) = \sum_{k} \sum_{S_k \sim Q^k(S,\epsilon)} Pr[f(S), S_k] \log Pr[f(S)|S_k]$$
$$\geq \sum_{k} \int_{\epsilon} \int_{S_k \sim B(S,\epsilon)} \frac{1}{M_k(f,S,S_k)} \log \frac{|Q|}{M_k(f,S,S_k)}.$$

The mutual information can be further derived as follows:

$$I(S_k, f(S)) \geq -\sum_{1}^{|Q|} \frac{1}{|Q|} \log f(S) +$$
$$\sum_{k} \int_{\epsilon} \int_{S_k \sim B(S,\epsilon)} \frac{1}{M_k(f,S,S_k)} \log \frac{|Q|}{M_k(f,S,S_k)}.$$

This means the lower bound of $I(S_k, f(S))$ is increased when $M_k(f,S,S_k)$ is smaller.

Works & Venue	Type	Models	Datasets	Attacks	Attack Type
ICCV'19 [13]	Defense	PointNet, PointNet++, DGCNN	ModelNet40	AIC, AIH, APP, SMA	P, A, D
ICCV'19 [12]	Attack	PointNet, PointNet++, DGCNN	3DMNIST, ModelNet40	SMA	D
ICIP'19 [3]	Defense & Attack	PointNet, PointNet++	ModelNet40	FGSM, I-FGSM, JSMA	P
CVPR'19 [9]	Attack	PointNet, PointNet++, DGCNN	ModelNet40	AIC, AIH, APP	P, A
MM'20 [5]	Attack	PointNet, PointNet++, DGCNN	ModelNet40	FGSM, PGD, CW	P
AAAI'20 [7]	Attack	PointNet++	ModelNet40	kNN	P
CVPR'20 [1]	Defense	PointNet, PointNet++	ModelNet40	FGSM, I-FGSM, PGD, MI-PGD	P
ECCV'20 [2]	Attack	PointNet, PointNet++, DGCNN	ModelNet40	AdvPC	P
ICCV'21 [4]	Defense	PointNet	ModelNet40	FGSM, PGD	P
TPAMI [8]	Attack	PointNet, PointNet++, DGCNN	ModelNet40	$GeoA^3$	P
ArXiv [11]	Defense & Attack	PointNet	ModelNet40	PG, PD, PA	P, A, D
ArXiv [6]	Defense & Attack	PointNet++, GvG-P, DUP-Net	ModelNet40	FGSM, BIM, MIM	P
ArXiv [10]	Defense	PointNet, PointNet++, DGCNN, RS-CNN	ModelNet10, ModelNet40	APP, kNN, SMA	P, D, B
Ours	Defense	PointNet, PointNet++, DGCNN, DUP-Net	ModelNet10, ModelNet40	AIC, AIH, APP, SMA, BIM, AdvPC	P, A, D, B

Table 1: Summary of evaluations in prior works and this paper. For attack type: P - point perturbing; A - point adding; D - point dropping; B - blackbox attack

2 Comparison With Previous Works

Table 1 summarizes the models, datasets and attacks adopted in our experiments, as well as comparisons with prior works. We claim that our evaluations are more comprehensive than existing studies about point cloud robustness.

3 PointNet++ under attacks

Network Structure	Training Strategy	Clean	Attacks					
			SMA−40	APP	AIC	AIH	**AAUA**	**LAUA**
PointNet++ (MSG)	Normal	89.77	62.18	0.00	0.00	0.00	15.55	0.00
	PointCutMix-R	92.57	85.92	0.00	0.16	2.15	22.06	0.00
	AT-BIM	88.47	64.00	0.00	0.00	0.00	16.00	0.00
PointNet++ (SSG)	Normal	89.85	56.45	0.00	0.00	0.00	14.11	0.00
	PointCutMix-R	92.78	86.57	0.00	1.83	2.88	22.82	0.00
	AT-BIM	89.53	71.51	0.00	0.00	0.00	17.88	0.00

Table 1: Accuracy of PointNet++ under untargeted attacks (%). All results are running results.

We further compare the accuracy of two types of PointNet++ under attacks. The results are shown in Table 1. For both types of PointNet++, training models with mix-up samples can significantly improve the accuracy under the dropping point attack, as mix-up samples can be seen as clean points dropped a lot of original points. However, PointNet++ cannot defend against adding perturbation attacks and adding additional points attacks. As we analyzed before, PointNet++ uses each point and its neighbors sampled based on distances coordinates to generate local features directly. When sampling neighbors on perturbed point clouds or point clouds with additional points, PointNet++

will use more noisy points to generate local features causing noise accumulating. Comparing with previous works, we find when using targeted attacks to attack PointNet++, the accuracy under attacks are significantly higher than results in Table 1. It is easy to understand that untargeted attacks are more powerful, and PointNet++ does not always predict adversarial examples as labels the adversary wants. For an adversary who wants the model to give wrong labels instead of specific labels, attacking PointNet++ is uncomplicated. Since the structure of PointNet++ is fragile under attacks, we do not apply our AMS on it.

4 Experiments of AMS

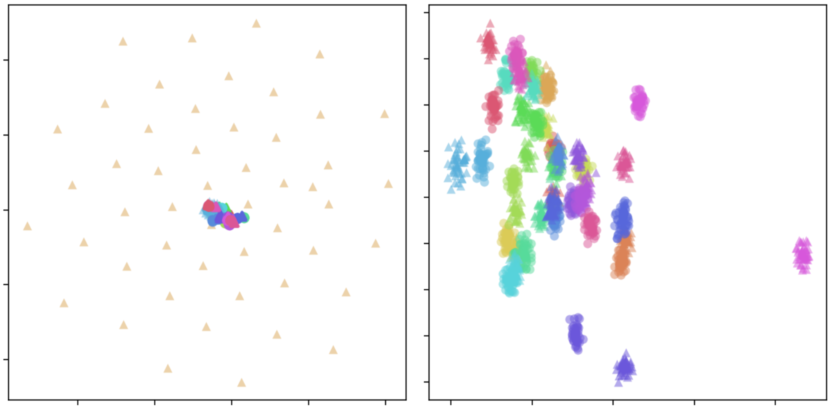
When comparing normally trained models and models trained with AT-BIM, we find that DGCNN achieves the highest clean accuracy. However, the DGCNN does not outperform our CCN under all four white-box attacks. On the other hand, the PointNet shows the worst performance. When comparing models trained with AT-BIM and our AMS, we can clearly notice that the AMS generalizes well to other model structures. Our CCN outperforms other baselines on clean accuracy and many white-box attacks. It achieves not only the highest **AAUA** but also the highest **LAUA**. It means that our CCN can work with the AMS together in harmony. In summary, for each architecture, AMS gives the best performance compared to Normal or AT-BIM training. To sum up, the integration of CCN and AMS is the most robust solution.

Network	**Training**	**Clean**	**Adversarial Examples**					
Structure	**Strategy**	**Sample**	SMA−40	APP	AIC	AIH	**AAUA**	**LAUA**
PointNet	Normal	88.76	41.88	55.64	49.68	43.43	47.66	41.88
	AT-BIM	88.23	45.41	85.39	84.98	86.36	75.54	45.41
	AMS	89.45	48.99	87.01	**86.49**	**87.26**	77.44	48.99
DGCNN	Normal	91.03	65.87	46.10	54.06	48.78	53.70	46.10
	AT-BIM	91.27	66.68	89.98	81.37	76.99	78.76	66.68
	AMS	**92.21**	75.41	**90.83**	85.47	83.93	83.91	75.41
CCN	Normal	90.87	67.94	57.47	61.04	53.37	59.96	53.37
	AT-BIM	90.05	67.37	88.80	83.77	79.75	79.92	67.37
	AMS	**92.41**	**77.72**	**90.50**	**86.09**	84.05	**84.74**	**77.72**

Table 2: Model accuracy for different solutions under the white-box attacks (%).

5 t-SNE Results Zoom Out

We plot all 40 classes (represented with different colors), and each class contains 50 point clouds from ModelNet40. Circles and triangles denote the clean and perturbed point clouds, respectively. From the Fig. 1, the DGCNN is not as robust as CCN. When we add perturbation to point clouds from a class, the features scatter in the feature space. However, our CCN will not be influenced by the perturbation significantly, which indicates that our CCMs in CCN can efficiently decrease the noise in the inputs.

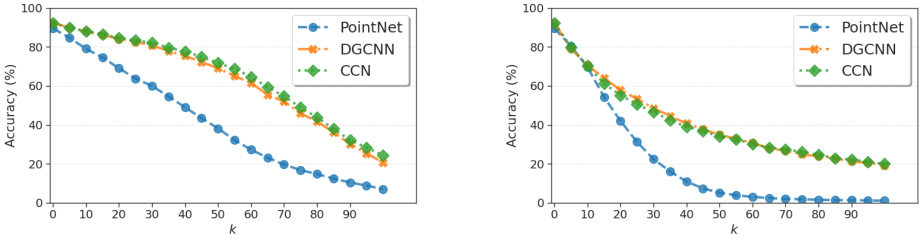


(a) DGCNN (b) CCN

Fig. 1: Feature map visualization.

6 Robustness under Different Attack Budgets

Furthermore, we show the accuracy of models trained with our AMS under SMA$-k$ and BIM$-k$ with different k in Fig. 2. When models are attacked by SMA$-k$, the PointNet is more fragile than other two models, resulting it has the lowest accuracy. The CCN outperforms the DGCNN with the k increasing becoming more clearly. When we attack models with BIM$-k$, we find that when the k is small, the accuracy of three models are very close. With the k increasing, the accuracy of the PointNet drops very quickly. As for CCN and DGCNN, the accuracy of DGCNN is higher than the accuracy of CCN at the start. However, when the k is higher than 60, the CCN starts to outperform the DGCNN. Both of them achieve higher accuracy than the PointNet. Overall, our CCN is the most robust one.



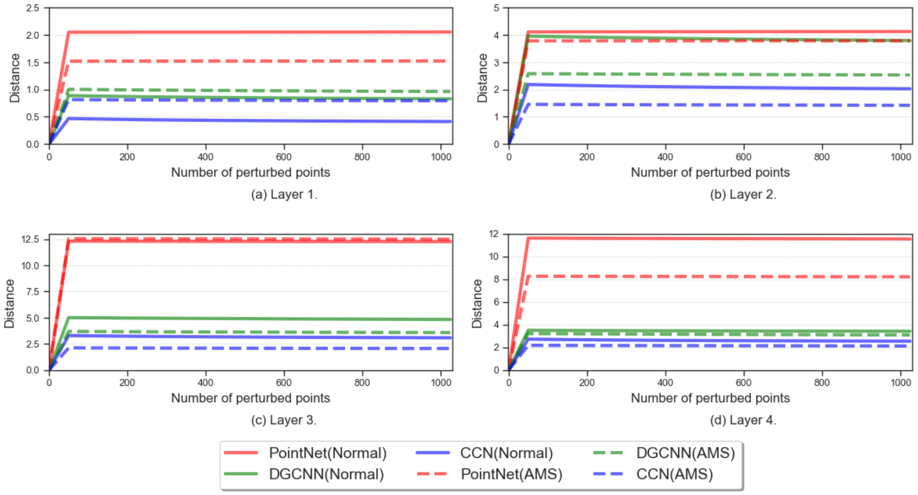
(a) Accuracy under SMA attack with different K

(b) Accuracy under BIM attack with different K

Fig. 2: Accuracy of models under SMA$-$K and BIM$-$K. All models are trained with our AMS.

7 Feature Distances under Different Perturbation Budgets

In Figs. 3 to 7, we compare feature distances under different perturbation strength. We adjust perturbation based on its variance. For each trail, we run 10 times and calculate the average distance. The results indicate that the scale of perturbation only has trivial influence of the feature distances. Our `CCN` can reduce the distances with the layer going deeper. Models trained with `AMS` can obtain smaller distance under all cases. Combining the above two phenomena, we can claim that our `CCN` and `AMS` can reduce feature distances and achieve higher mutual information. So both of them can improve model's robustness and be harmonious with our theoretical analysis.



(a) Layer 1.

(b) Layer 2.

(c) Layer 3.

(d) Layer 4.

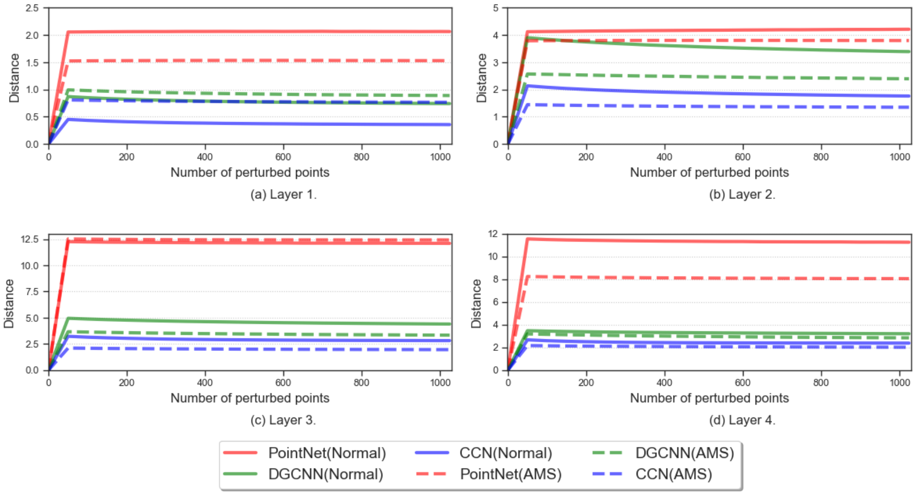
Fig. 3: Cosine distance of features between clean and perturbed samples from different layers. The perturbation is generated from a Gaussian distribution with $\mu = 0$ and $\sigma = 0.01$.

8 Model Size

Table 3 compares the model size and number of parameters for different point cloud models. We observe that `CCN` is slightly bigger than DGCNN due to the introduction of the CCM. Its size is still smaller than PointNet++. Nevertheless, `CCN` gives the best robustness among these models.

9 Comparing AT with multiple types of attacks.

We consider a stronger baseline method, "Multiple Types of Attacks" (MTA), where the robust model is trained with two types of AEs (BIM-20 attack and SMA-20 attack) together. Table 4 compares `AMS` with this MTA strategy. We have the following observations. (1) For the clean accuracy, `CCN` is better than MTA, as it uses the mix-up



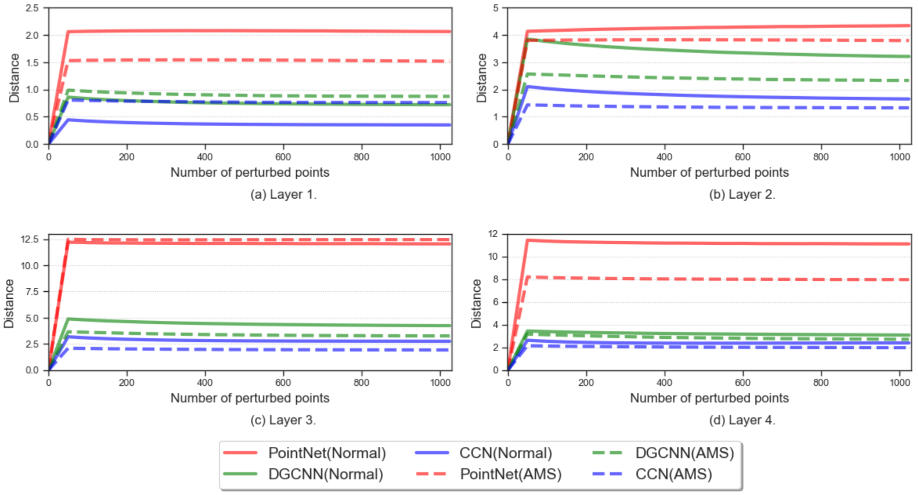
(a) Layer 1.

(b) Layer 2.

(c) Layer 3.

(d) Layer 4.

Fig. 4: Cosine distance of features between clean and perturbed samples from different layers. The perturbation is generated from a Gaussian distribution with $\mu = 0$ and $\sigma = 0.03$.



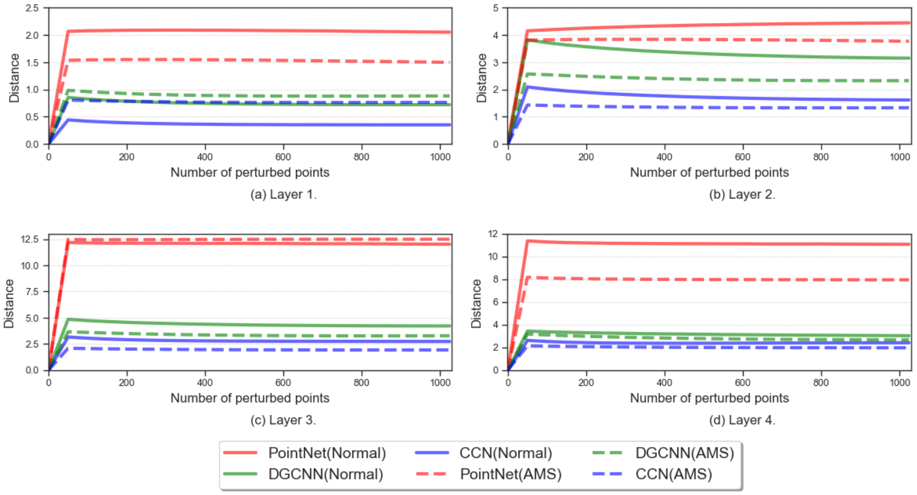
(a) Layer 1.

(b) Layer 2.

(c) Layer 3.

(d) Layer 4.

Fig. 5: Cosine distance of features between clean and perturbed samples from different layers. The perturbation is generated from a Gaussian distribution with $\mu = 0$ and $\sigma = 0.05$.



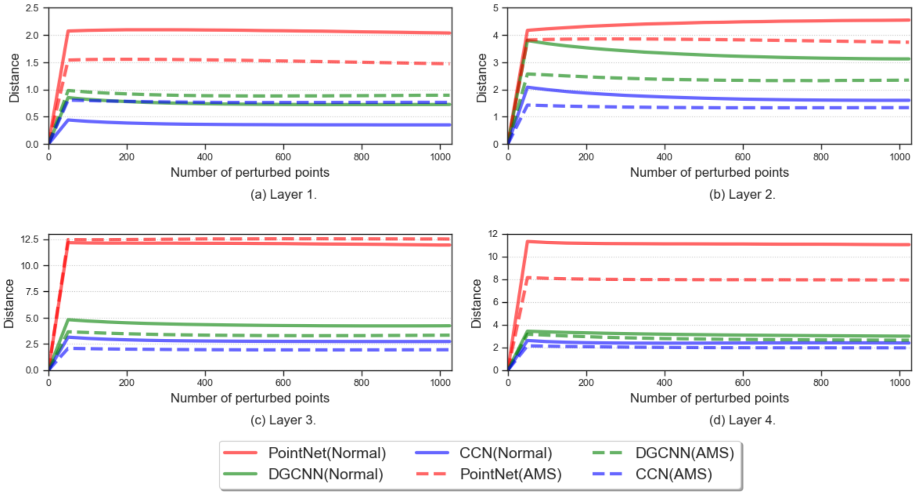
(a) Layer 1.

(b) Layer 2.

(c) Layer 3.

(d) Layer 4.

Fig. 6: Cosine distance of features between clean and perturbed samples from different layers. The perturbation is generated from a Gaussian distribution with $\mu = 0$ and $\sigma = 0.07$.



(a) Layer 1.

(b) Layer 2.

(c) Layer 3.

(d) Layer 4.

Fig. 7: Cosine distance of features between clean and perturbed samples from different layers. The perturbation is generated from a Gaussian distribution with $\mu = 0$ and $\sigma = 0.1$.

Model	Model size (MB)	# of Para (million).
PointNet	9.4	0.80
PointNet++	12.0	1.02
DGCNN	11.0	0.94
CCN	11.6	0.98

Table 3: The numbers of model parameters and model sizes for different point cloud models.

samples in the training process. (2) For the robustness, MTA only performs better than CCN for the SMA-20 attack, since it adopts this attack for adversarial training, and the robustness is overfitted on these samples, while AMS uses SMA-10. For other attacks, AMS outperforms MTA. AMS also gives the best **AAUA** and **LAUA**. This indicates AMS is still the better training strategy.

Model	**Strategy**	**Clean Sample**	**Adversarial Examples**					
			SMA-20	APP	AIC	AIH	**AAUA**	**LAUA**
PointNet	MTA	87.70	76.54	86.97	85.63	86.44	83.90	76.54
	AMS	89.45	69.03	87.01	**86.49**	**87.26**	82.46	69.03
DGCNN	MTA	90.79	**85.71**	87.95	82.35	76.95	83.24	76.95
	AMS	**92.21**	84.09	**90.83**	85.47	83.93	**88.58**	**83.93**

Table 4: Comparison of more baselines.

10 Results on ModelNet10

Defense Solutions	**Clean Sample**	**Adversarial Examples**					
		SMA-40	APP	AIC	AIH	**AAUA**	**LAUA**
PointNet + AMS	84.82	45.98	75.22	**73.33**	**69.53**	66.02	45.98
DGCNN + AMS	**93.64**	**81.36**	75.89	63.84	57.25	69.59	57.25
CCN + AMS	92.75	72.10	**78.01**	70.65	61.72	**70.62**	**61.72**

Table 5: Results on ModelNet10.

We verify the effectiveness of our CCN and AMS on ModelNet10 in Table 5. As ModelNet10 can be seen as a toy dataset, we do not fine-tune our training hyperparameters and only verify our proposed methods. The results verify that our methods can still work on other dataset. Because ModelNet10 is a small dataset, models heavily overfit the AEs and the robustness is not as high as on the ModelNet40.

References

1. Dong, X., Chen, D., Zhou, H., Hua, G., Zhang, W., Yu, N.: Self-robust 3d point recognition via gather-vector guidance. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11513–11521. IEEE (2020)

2. Hamdi, A., Rojas, S., Thabet, A.K., Ghanem, B.: Advpc: Transferable adversarial perturbations on 3d point clouds. In: Proc. of the ECCV. vol. 12357, pp. 241–257 (2020)
3. Liu, D., Yu, R., Su, H.: Extending Adversarial Attacks and Defenses to Deep 3D Point Cloud Classifiers. In: Proc. of the ICIP (2019)
4. Lorenz, T., Ruoss, A., Balunović, M., Singh, G., Vechev, M.: Robustness certification for point cloud models. arXiv preprint arXiv:2103.16652 (2021)
5. Ma, C., Meng, W., Wu, B., Xu, S., Zhang, X.: Efficient Joint Gradient Based Attack Against SOR Defense for 3D Point Cloud Classification. In: Proc. of the MM. pp. 1819–1827 (2020)
6. Sun, J., Koenig, K., Cao, Y., Chen, Q.A., Mao, Z.M.: On adversarial robustness of 3d point cloud classification under adaptive attacks. arXiv preprint arXiv:2011.11922 (2020)
7. Tsai, T., Yang, K., Ho, T.Y., Jin, Y.: Robust adversarial objects against deep learning models. In: Proc. of the AAAI. pp. 954–962 (2020)
8. Wen, Y., Lin, J., Chen, K., Jia, K.: Geometry-aware Generation of Adversarial and Cooperative Point Clouds. CoRR **abs/1912.11171** (2019)
9. Xiang, C., Qi, C.R., Li, B.: Generating 3D Adversarial Point Clouds. In: Proc. of the CVPR (2019)
10. Zhang, J., Chen, L., Ouyang, B., Liu, B., Zhu, J., Chen, Y., Meng, Y., Wu, D.: PointCutMix: Regularization Strategy for Point Cloud Classification. CoRR **abs/2101.01461** (2021)
11. Zhang, Q., Yang, J., Fang, R., Ni, B., Liu, J., Tian, Q.: Adversarial attack and defense on point sets. CoRR **abs/1902.10899** (2019)
12. Zheng, T., Chen, C., Yuan, J., Li, B., Ren, K.: PointCloud Saliency Maps. In: Proc. of the ICCV (2019)
13. Zhou, H., Chen, K., Zhang, W., Fang, H., Zhou, W., Yu, N.: DUP-Net: Denoiser and Upsampler Network for 3D Adversarial Point Clouds Defense. In: Proc. of the ICCV (2019)