3D Point Cloud Completion with Geometric-Aware Adversarial Augmentation

Mengxi Wu
NYU Multimedia and Visual Computing Lab
NYU Tandon School of Engineering
New York University, New York, USA
Email: mw4355@nyu.edu

Hao Huang, Yi Fang†
NYU Multimedia and Visual Computing Lab
NYUAD Center for Artificial Intelligence and Robotics
New York University, Abu Dhabi, UAE
Email: {hh1811, yfang}@nyu.edu

***Abstract*—With the popularity of 3D sensors in self-driving and other robotics applications, extensive research has focused on designing novel neural network architectures for accurate 3D point cloud completion. However, unlike point cloud classification and reconstruction, the role of adversarial samples in 3D point cloud completion has seldom been explored. In this work, we demonstrate that adversarial samples can benefit neural networks on 3D point cloud completion tasks. We propose a novel approach to craft adversarial samples that improve the performance of models on both clean and adversarial inputs. In contrast to the Projected Gradient Descent (PGD) attack, our method generates adversarial samples that keep the geometric features in clean samples and contain few outliers. In particular, we use minimum absolute curvature directions to constrain the adversarial perturbations for each input point. The gradient components in the minimum absolute curvature directions are taken as adversarial perturbations. In addition, we adopt attack strength accumulation and auxiliary Batch Normalization layers to speed up the training process and alleviate the distribution mismatch between clean and adversarial samples. Experimental results demonstrate that training with the adversarial samples crafted by our method under the geometric-aware constraint effectively enhances the performance of the Point Completion Network (PCN) on the ShapeNet dataset.**

***Index Terms*—Adversarial Machine Learning, 3D Point Cloud Completion**

I. Introduction

Modern deep learning models are vulnerable to adversarial examples [1]. Adversarial examples are often treated as threats to the safety-critical application of neural networks. For example, in point cloud classification and reconstruction, adversarial samples can mislead the model to predict the wrong classes or reconstruct different types of objects. Thus, it has triggered a large amount of research focusing on attack/defense techniques in these two fields. However, very few efforts have been made to investigate how the adversarial samples affect the 3D point cloud completion.

In this paper, we investigate adversarial samples and adversarial training in 3D point completion tasks. Xie et al. [2] demonstrate that training with adversarial samples does help image recognition. Thus, rather than focusing on defending the adversarial samples, we pay attention to using adversarial features to boost the performance of the model on 3D shape completion. However, not all adversarial features are beneficial. Outliers are parts of adversarial features but are noise that may distort the original shapes of the point clouds. Training with adversarial samples that have many outliers will lead neural networks to learn the noise features instead of the true geometric properties of the objects that may benefit the performance. Widely applied attack methods, including Fast Gradient Sign Method (FGSM) [3] and PGD [4] usually generate adversarial samples that contain many outliers. Thus, developing a method to craft strong adversarial samples with few outliers becomes essential.

To overcome the problem caused by outliers, various approaches [5], [6], [7] have attempted to make adversarial perturbations imperceptible. These methods constrain adversarial point clouds to be close to clean ones under specific distance metrics. More recently, in [8], the authors point out that sharp inconsistencies exist between the human perception of 2D images and that of 3D shapes. They suggest that adversarial samples should satisfy the surface properties of the clean ones. Technically, they make the magnitudes of local curvatures between adversarial and clean point clouds close to each other. Inspired by their work, we also take the consistency of surface properties into account. Different from [8], which brutally adopts a curvature loss defined on each point to enforce the local consistency between adversarial samples and clean samples, we use the direction of absolute minimum curvature on each point to maintain the surface properties when generating adversarial samples. Our approach integrates a gradient descent approach with 3D shape geometry, which has not been explored yet. As the curvature of clean shapes can be pre-computed once and for all, our approach can be more computation-efficient than [8] which needs to compute the curvature loss at each training iteration.

To be specific, at each point on the surface, along with different directions, we can obtain different curvatures. The gradient can be decomposed in different curvature directions. We notice that moving a point along the direction of minimum absolute curvature will most likely introduce imperceptible changes. As expected, our method produces adversarial point clouds without introducing noticeable shape modifications or outliers. We name this attack **PMCD attack**, abbreviated for **p**rojection on **m**inimum absolute **c**urvature **d**irection attack.

† Corresponding author

Additionally, we apply auxiliary batch normalization [2] to alleviate distribution mismatch and adopt attack strength accumulation [9] to reduce the time consumption for adversarial samples generation. Experiments show that models trained with the adversarial samples crafted by our method achieve more improvement compared with other existing methods.

II. Related Work

A. 3D Point Cloud Completion

Traditional methods based on voxel grids or distance fields to represent 3D objects [10], [11], [12] achieve outstanding performances in shape completion [10], [11], [13]. However, compared to voxel grids or distance fields, 3D point cloud representations need less memory consumption and have a strong ability to represent fine-grained details. Point clouds are unordered point sets, so the convolution operation is no longer applicable. For point cloud completion, PCN [14] is the first learning-based architecture. It contains an encoder that summarizes the geometric information in the input point cloud. A folding-based operation that maps the 2D points onto a 3D surface is adopted in the decoder to approximate a smooth surface representing a shape's local geometry. After PCN, many other methods [15], [16], [17] are proposed to achieve higher resolution. In more recent work, PoinTr [18] reformulates point cloud completion as a set-to-set translation. It achieves state-of-art results by employing a transformer encoder-decoder architecture.

In most works on 3D point cloud completion tasks, Chamfer Distance is used as training loss and evaluation metrics. Given two point sets P and P', the Chamfer distance between P and P' can be computed as

$$CD = \frac{1}{n'} \sum_{p' \in P'} \min_{p \in P} \|p - p'\|_2 + \frac{1}{n} \sum_{p \in P} \min_{p' \in P'} \|p' - p\|_2 \quad (1)$$

where n' represents the number of points in P' and n represents the number of points in P.

B. Benefit of Adversarial Perturbations

Many previous works illustrate that adversarial examples can provide additional features to neural networks. For example, in 2D image classification, images are represented by pixel values. Adding noise to pixel values may not affect the image visually but will be very different from the original image at the pixel level. Thus, it can bring new learnable features to neural networks. Similarly, each point in point clouds is represented by coordinates. Adding noise will make noticeable changes at the coordinate level. In [19], the authors show that models trained with adversarial samples are more robust to high-frequency noise. Zhang et al. [20] further illustrate those feature representations that are learned through adding adversarial samples to the training set are less sensitive to distortions and more focused on geometric and shape information.

C. Adversarial Training

In adversarial training, adversarial attacks are taken as data augmentation. Models trained with adversarial samples can achieve considerable robustness. Adversarial training can be formulated as a min-max optimization problem [21]:

$$\min_{f \in \mathcal{H}} \mathbb{E}_{(x,y)\sim\mathcal{D}}[\max_{\delta \in \mathcal{S}} L(f(x+\delta), y)] \quad (2)$$

where $\mathcal{H}$ is the parameters space, $\mathcal{D}$ is the training dataset distribution, L is a loss function, f is the neural network, and $\mathcal{S}$ is the permitted perturbation space [9].

Projected Gradient Descent (PGD) [4] provides an approximate solution for this problem. It takes multi-step projected gradient descent on a negative loss function. Then, it projects adversarial perturbations outside the allowed perturbation space S back to S:

$$x_{adv} = \Pi_{x+S}(x + \alpha \text{sign}(\nabla_x L(f(x), y))) \quad (3)$$

α is the attack step size. Π is the projection function. Our proposed adversarial samples generation method is inspired by the PGD attack and fuses it with 3D shape geometry.

III. Methodology

Our geometric-aware adversarial training algorithm includes three major techniques:

- A novel geometric-aware attack method (*i.e.*, PMCD attack) is proposed to generate adversarial samples with few outliers and maintain the surface cleanness of clean samples.
- We reuse adversarial perturbations from the previous epoch to achieve attack strength accumulation across epochs [9].
- The model is trained with both adversarial and clean samples, which are normalized by different Batch Normalization layers separately for training efficiency [2].

The pipeline of the training method is illustrated in Figure 1, and the details are presented in Algorithms 1 and 2.

A. PMCD attack

Our novel attack method, projection on the minimum absolute curvature attack (PMCD attack) is formally presented in Algorithm 1. Its most notable feature is constraining the adversarial perturbations with geometric properties of clean samples. It projects the gradient of an input point on the minimum absolute curvature direction at that point. The projection is used as the adversarial perturbation. The adversarial samples generated by this method maintain the geometric features of the clean samples and contain very few outliers.

1) Benefits of Minimum Absolute Curvature Direction: We observe that, in the direction of the curvature with minimum absolute value, the surface may bend most slightly compared to other directions. In other words, the surface is closest to the tangent plane in this direction. If we add perturbations to the point along this direction on the tangent plane, the point will still be close to the original surface. Thus, we choose to move

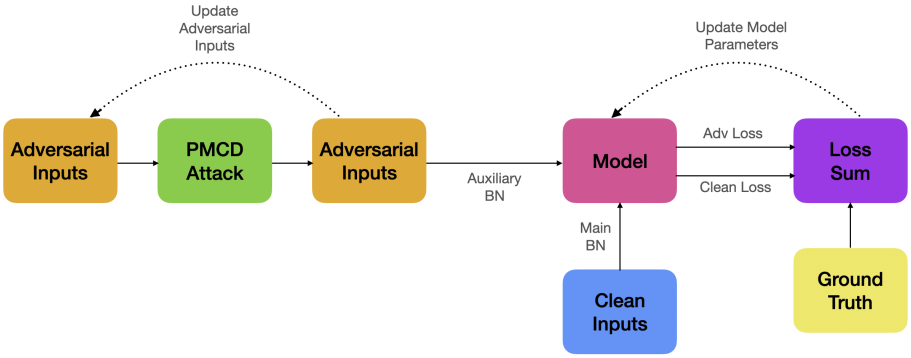

Fig. 1. The Pipeline of Geometric-Aware Training Algorithm. Initially, adversarial inputs are the same as clean inputs. The algorithm first loads the adversarial samples generated from the previous epoch and passes them to the PMCD attack to compute the adversarial samples in the current epoch. Then, the loss on clean samples is computed with main BN layers, and the loss on adversarial samples is computed with auxiliary BN layers. The parameters of the neural networks are optimized with respect to the sum of two losses.

Algorithm 1 PMCD attack

Input: Training sample x, Ground truth gt, minimum absolute curvature directions d_m
Output: Adversarial sample x_{adv}
Parameters: Movement multiplier β, Network parameter θ, Allowed perturbations space S

1: Initialize x_{adv} by cloning x
2: Compute loss $l = L(\theta, x_{adv}, gt)$
3: Compute gradients $g = \nabla_{x_{adv}} l$
4: Compute projections $p = \text{rowMulti}(g, d_m)$ [1]
5: Compute perturbations $\delta = \text{rowMulti}(p, d_m)$
6: $x_{adv} = \Pi_{x+S}(x_{adv} + \beta \cdot \delta)$
7: **return** x_{adv}

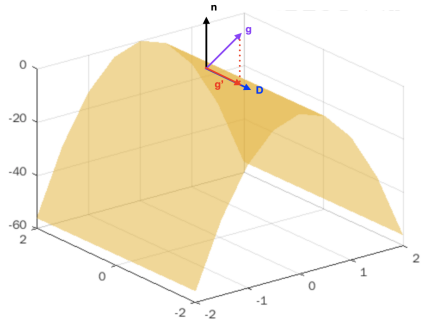

Fig. 2. PMCD attack. n is the normal vector of the tangent plane at point p. D is the minimum absolute curvature direction. g is the gradient of the p. g' is component of g in the direction of D. We use g' as the perturbations for p.

the point by adding gradient components in the minimum absolute curvature direction. Figure 2 illustrates our idea.

Using minimum absolute curvature direction can be computation-efficient. Wen et al. [8] demonstrate that magnitudes of local curvatures can also effectively describe the bending condition of a surface. However, directly using magnitude differences between adversarial and clean samples as a restriction or regularization term is not very computation-efficient. The magnitudes of the local curvatures need to be recomputed once new perturbations are added to the adversarial samples. For a training scheme that clean samples are fixed during the training process, minimum absolute curvature directions of clean samples can be pre-computed and saved before the training process begin. In this way, significant computation consumption can be reduced.

2) Minimum Absolute Curvature Direction Estimation: Since along the minimum absolute curvature, the variation of directions of normal vectors is the smallest, we use normal variation to estimate minimum absolute curvature. Previous works [22], [8] use normal variation to estimate principal directions. Principal directions are directions of minimum curvature and maximum curvature. Thus, we first apply their method to compute principal directions and then use the direction of the minimum curvature as the estimation of the minimum absolute curvature direction. The Visualization Toolkit (VTK) [23] does not support principal directions, while The Point Cloud Library (PCL) [24] does support principal directions but is less computationally efficient than the method we offer as follows.

Following [22], [8], let P represents the surface of an input point cloud. For every point $p \in P$, the unit normal vector n_p of the surface at p is estimated first. We find the closest point $p' \in P$ by $p' = \text{argmin}_{q \in P} \|q - p\|_2^2$. We can obtain a local neighborhood $N_p \subset P$ for each p. The number of points in N_p is represented by k_p. We use $k_p = 20$. A positive semidefinite covariance matrix C_p can be constructed by

$$C_p = \sum_{q \in N_p} (p - q) \otimes (p - q) \tag{4}$$

where $\otimes$ denotes the outer product operation. The estimated n_p is the eigenvector corresponding to the smallest eigenvalue

[1] rowMulti denotes row-wise dot product operation.

of C_p. Thus, we can obtain the estimated n_p by applying eigendecomposition to C_p.

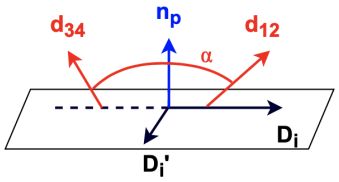

Fig. 3. Normal Variation Calculation. n_p is the normal vector of the tangent plane. D_i and D_i' are two orthogonal directions. d_{12} and d_{34} lie on the plane determined by n_p and D_i. α is the angle between d_{34} and d_{12}.

For every point $p \in P$, we collect the normal vectors of its k_n-nearest neighbors. We set k_n to be 10. This set of normal vectors is represented by M_p. All the directions that start from point p on the tangent plane make up a circle. We randomly pick an orthogonal pair of them to begin. We denote this pair as (D_0, D_0'). We generate other $k_d - 1$ pairs by rotating (D_0, D_0') on the tangent plane. In this way, we obtain k_d pairs of directions on the tangent plane. These k_d pair of directions are equally spaced. k_d is set to be 18 in the experiment. For each pair, we compute the difference between their normal variations. In direction D_i, we find two normal vectors $n_1, n_2 \in M_p$, that have the smallest angles with D_i. We calculate d_{12}, the direction of the intersection line of the plane determined by n_1 and n_2, and the plane determined by D_i and n_p.

$$d_{12} = (n_1 \times n_2) \times D_i' \tag{5}$$

where $\times$ denotes the cross product. Then, we use two other vectors $n_3, n_4 \in M_p$, that have the largest angles with D_i to calculate d_{34} in the same way. The angle between d_{12} and d_{34} is α_i, as shown in Figure 3. α_i represents the normal variation along with D_i. We repeat the way to calculate α_i to compute the normal variation of the D_i', β_i. Then, we pick (D_i, D_i') as the principal directions when $|\beta_i - \alpha_i|$ is the largest. If α_i is smaller than β_i, we set D_i to be the minimum absolute curvature direction; otherwise, we set D_i' to be the minimum absolute curvature direction.

B. Attack Strength Accumulation

As shown in [9], PGD-based adversarial training requires approximately 100x more time to make the model converge than the regular training does. The key insight of [9] is the transferability of models from neighboring epochs. The adversarial samples in the previously neighboring epochs will still be adversarial in the current epoch. They conclude that the attack strength can be accumulated across neighboring epochs [9]. Inspired by their results, we use attack strength accumulation in our training algorithm to let the PMCD attack achieve high attack strength efficiently. However, model parameters tend to vary drastically between epochs that are far from each other. Hence, the adversarial perturbations in epochs that are too far away from the current epoch will be less useful. To reduce this impact, we let the accumulation restart from the beginning periodically.

Algorithm 2 Geometric-Aware Adversarial Training

Input: Clean training set X with ground truth, Minimum absolute curvature directions d_m for clean samples, The number of epochs to reset perturbation r
Output: Network parameter θ

Initialize θ
Initialize adversarial training set X_{adv} by cloning X
for $epoch = 1, ..., N$ **do**
  for each training step **do**
    Sample a mini-batch $x^b \subset X$, $x^b_{adv} \subset X_{adv}$ with ground truth gt^b and d^b_m
    if $epoch \ \% \ r == 0$ **then**
      $x^b_{adv} = x^b$
    end if
    Compute x^b_{adv}=PMCD$(\theta, x^b_{adv}, gt^b, d^b_m)$ with the auxiliary BNs
    Compute loss $L(\theta, x^b, gt^b)$ with the main BNs
    Compute loss $L(\theta, x^b_{adv}, gt^b)$ with the auxiliary BNs
    $\theta = \text{argmin}_\theta \ L(\theta, x^b, gt^b) + L(\theta, x^b_{adv}, gt^b)$
  end for
end for
return θ

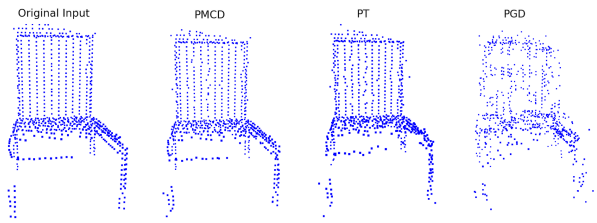

Fig. 4. Adversarial samples generated by different attack methods. The category we choose is chair. We use 15 iterations to generate adversarial samples. We can see the samples generated by PGD or PT contain a lot of outliers. However, the sample generated by PMCD is near without outliers.

C. Auxiliary Batch Normalization

Adding adversarial samples to the clean training set may lead to performance degradation on clean samples. Very limited amount of research in the point cloud processing community address this issue. Here we refer to a study for image classification. In [2], the authors demonstrate that adversarial images and clean images have different underlying distributions. Maintaining the same Batch Normalization layers for adversarial images and clean images will lead to inaccurate statistics estimation. Inaccurate estimations will cause performance degradation. Therefore, we adapt their auxiliary Batch Normalization method to avoid performance degradation caused by this factor.

TABLE I

SEEN CATEGORIES. THESE EIGHT CATEGORIES ARE INCLUDED IN THE TRAINING SET. PMCD PERFORMS THE BEST AMONG ALL THE METHODS. IT OUTPERFORMS BASELINE WITH A CONSIDERABLE DECREASE OF AROUND 4.8% IN CHAMFER DISTANCE.

Method	Avg	Chair	Table	Sofa	Cabinet	Lamp	Car	Airplane	Vessel
Baseline [14]	0.01166	0.01292	0.01135	0.01356	0.01304	0.01472	0.00984	**0.00625**	0.01120
Random	0.01446	0.01517	0.01415	0.01618	0.01591	0.01767	0.01076	0.00856	0.01728
PGD [4]	0.01144	0.01298	0.01117	0.01360	0.01206	0.01433	0.00982	0.00655	0.01101
PGD w/o	0.01537	0.01774	0.01537	0.01718	0.01761	0.01933	0.01276	0.01007	0.01557
PT [25]	0.01128	0.01307	0.01080	**0.01352**	0.01208	0.01368	0.00982	0.00642	0.01089
PMCD	**0.01110**	**0.01253**	**0.01072**	0.01353	**0.01183**	**0.01365**	**0.00957**	0.00638	**0.01061**

TABLE II

UNSEEN CATEGORIES. THE FIRST FOUR CATEGORIES HAVE SIMILAR SHAPES TO THE CATEGORIES THAT ARE INCLUDED IN THE TRAINING SET. PMCD PERFORMS BEST AMONG ALL THE METHODS. COMPARED TO BASELINE, THE CHAMFER DISTANCE DECREASES BY 1.3%. THE LAST FOUR CATEGORIES ARE NOT SIMILAR TO CATEGORIES THAT ARE INCLUDED IN THE TRAINING SET. COMPARED WITH THE RESULTS IN TABLE 1, ALL THE METHODS PERFORM WORSE IN THESE FOUR CATEGORIES, AND BASELINE OUTPERFORMS PMCD.

Method	Avg	Bed	Bench	Bookshelf	Bus	Avg	Guitar	Motorbike	Pistol	Skateboard
Baseline [14]	0.01611	**0.02292**	0.01267	0.01744	0.01142	**0.01408**	**0.01242**	**0.01483**	**0.01610**	0.01298
Random	0.01913	0.02565	0.01386	0.01972	0.01729	0.01886	0.01707	0.01470	0.02147	0.02220
PGD [4]	0.01600	0.02348	0.01266	0.01642	0.01143	0.01558	0.01748	0.01493	0.01582	0.01410
PGD w/o	0.01975	0.02574	0.01639	0.01892	0.01796	0.01766	0.01491	0.01444	0.01708	0.02431
PT [25]	0.01608	0.02406	0.01240	0.01632	0.01153	0.01523	0.01547	0.01529	0.01699	0.01318
PMCD	**0.01590**	0.02387	**0.01246**	**0.01630**	**0.01095**	0.01456	0.01424	0.01516	0.01621	**0.01262**

TABLE III

SEEN CATEGORIES (WITH ATTACK). PMCD PERFORMS BETTER THAN BASELINE EXCEPT FOR TABLE, SOFA, AND AIRPLANE.

Method	Avg	Chair	Table	Sofa	Cabinet	Lamp	Car	Airplane	Vessel
Baseline [14]	0.02417	**0.02619**	0.02732	**0.02449**	0.02585	0.02999	0.01858	**0.01905**	0.02190
PMCD	**0.02355**	0.02713	**0.02520**	0.02523	**0.02349**	**0.02849**	**0.01809**	0.01919	**0.02162**

TABLE IV

UNSEEN CATEGORIES (WITH ATTACK). FOR THE FIRST FOUR CATEGORIES THAT ARE SIMILAR TO CATEGORIES IN THE TRAINING SET, PMCD OUTPERFORMS BASELINE IN ALL CATEGORIES. FOR THE LAST FOUR CATEGORIES THAT ARE DISSIMILAR TO THE CATEGORIES IN THE TRAINING SET, BASELINE PERFORMS BETTER THAN PMCD IN ALL CATEGORIES EXCEPT MOTORBIKE.

Method	Avg	Bed	Bench	Bookshelf	Bus	Avg	Guitar	Motorbike	Pistol	Skateboard
Baseline [14]	0.03057	0.03957	0.02852	0.03007	0.02413	**0.02880**	**0.02526**	0.03250	**0.03112**	**0.02633**
PMCD	**0.02841**	**0.03727**	**0.02692**	**0.02548**	**0.02395**	0.02911	0.02573	**0.03075**	0.03263	0.02736

IV. EXPERIMENTS

We conduct our experiment with the PCN [14]. To prove the effectiveness of our proposed method, we compare our PMCD with the following approaches:

- **Baseline**. We train a PCN without adding adversarial samples to the dataset. This PCN is treated as the Baseline of our experiments.
- **PGD** [4]. A PCN model is trained with samples generated by the PGD method and clean samples. PGD can generate strong adversarial perturbations based on the gradients. However, it does not use geometric features to constrain adversarial perturbations.
- **PGD w/o**. To demonstrate the auxiliary batch normalization is effective, we train a PCN with adversarial samples generated by PGD and clean samples, while the auxiliary batch normalization is not applied.
- **Random**. To show the effect of adversarial perturbations, we generate the new training samples by adding random noise drawn from a normal distribution with 0.05 as the standard deviation. We train a PCN with both the noisy samples and the clean samples.
- **PT**. PT (projection on the tangent plane) is proposed by Liu et al. [25]. It directly projects the gradients of a point on the tangent plane at that point. Unlike our PMCD attack, PT does not specify the directions of the gradient components on the tangent plane.

We evaluate models on both clean samples and adversarial samples. On clean samples, we compare our novel PMCD attack with Random, PGD, and PT to learn the effectiveness of adding adversarial samples and constraining adversarial perturbations with minimum absolute curvature direction. Besides, We compare PGD with PGD w/o to illustrate that auxiliary BN plays a significant role in the performance of the model. On adversarial samples, we only compare PMCD against Baseline. Since our main purpose for proposing PMCD is to enhance the model's performance on clean samples, we evaluate the

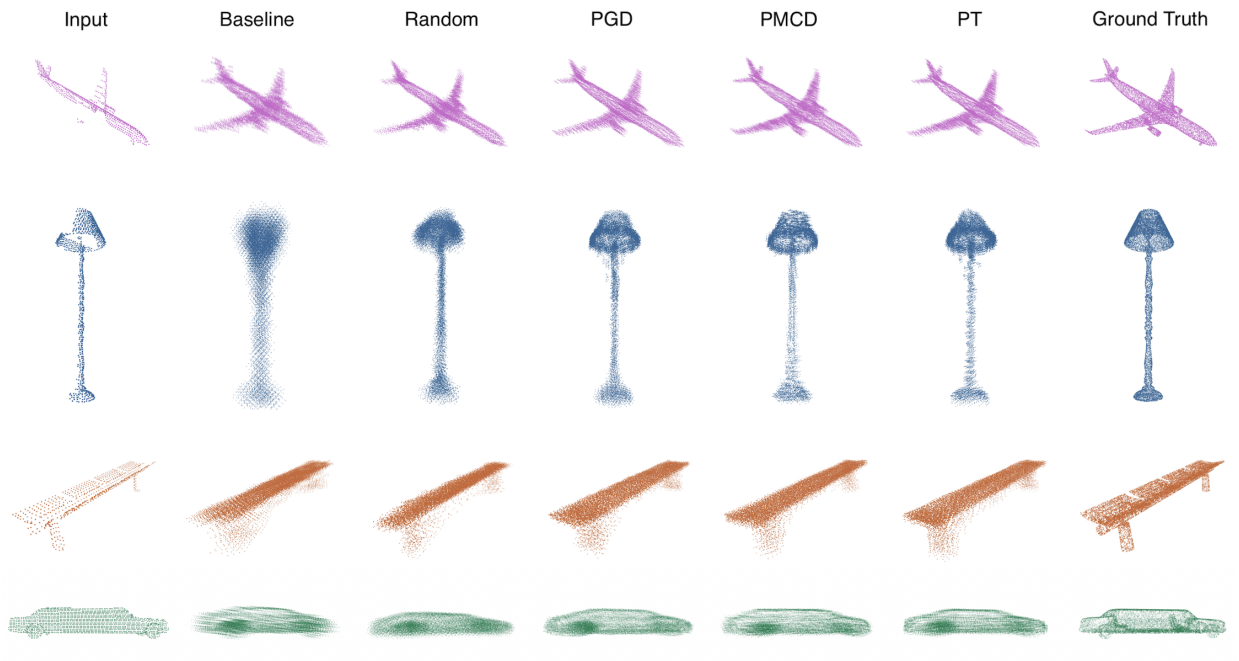

Fig. 5. Sample Visualization Results of each Method. These categories are airplane, lamp, bench, and car. The original inputs are selected from the test set. Compared with all other methods, our proposed PMCD helps the model complete with more clear outlines and details.

model trained with PMCD on adversarial samples to show that PMCD won't decrease the robustness of the model when tested on adversarial samples. The adversarial samples we used for testing are generated by PGD after five epochs without using auxiliary BN.

A. Training Schemes

We use the same dataset and split provided by PCN. The optimizer for training PCN models is Adam [26] optimizer. The models are trained for 30 epochs and a batch size of 64. For the Baseline, the initial learning rate is 0.001, while for adversarial training, the initial learning rate is set to be 0.0005. The learning rate is decayed by 0.7 every five epochs. We reset the accumulation of attack strength every 15 epochs.

B. Results

Figure 4 illustrates the visualization results for adversarial samples generated by different methods. Figure 5 provides the visualization of the results of each method on test samples. All results in the six tables are mean chamfer distance per point.

For seen and unseen similar categories, Baseline performs worse than methods that are trained with adversarial samples. This shows that adversarial perturbations can provide beneficial features to the models and boost their performances. PMCD outperforms Baseline with a decrease in chamfer distance. It also beats PT and PGD. PT and PGD lack sophisticated geometric constraints on adversarial perturbations. This reflects putting sophisticated geometric restrictions on adversarial perturbation is essential. PGD w/o performs worst among all the methods. This shows that, like in image classification, the auxiliary batch normalization method alleviates the negative effects caused by distribution mismatch between adversarial samples and clean samples. Besides, Random w/o performs second worse among all the methods. We can see that the simple random perturbations do not bring positive effects on the performance of PCN. Results for Table 3 and Table 4 demonstrate that our PMCD won't decrease the robustness of models too much while achieving a noticeable effect on clean samples.

However, the results on unseen and dissimilar categories reflect that model trained with PMCD does not generalize well on samples that are very dissimilar from the training ones. We also notice that in these categories, PGD and PT also perform worse than Baseline. Thus, we hypothesize that the performance degradation may be caused by overfitting. The model overfits the geometric features of the training data.

V. Conclusion

We introduce a novel approach to crafting adversarial samples. The key component is that we constrain the adversarial perturbations by selecting the gradient component in the minimum absolute curvature direction as the adversarial perturbations. Our approach effectively improves the deep shape generation model. For future study, we will explore methods such as zero-shot learning approaches to reduce the degradation of generalization ability on unseen categories that are dissimilar to the training set. We will develop new approaches to compute the minimum absolute curvature direction more precisely and efficiently. Another direction that catches our attention is to explore restricting adversarial perturbation with other geometric features of the input shapes.

REFERENCES

[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[2] C. Xie, M. Tan, B. Gong, J. Wang, A. L. Yuille, and Q. V. Le, "Adversarial examples improve image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 819–828.

[3] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[4] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," *arXiv preprint arXiv:1611.01236*, 2016.

[5] C. Xiang, C. R. Qi, and B. Li, "Generating 3d adversarial point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9136–9144.

[6] D. Liu, R. Yu, and H. Su, "Adversarial shape perturbations on 3d point clouds," in *European Conference on Computer Vision*. Springer, 2020, pp. 88–104.

[7] J. Yang, Q. Zhang, R. Fang, B. Ni, J. Liu, and Q. Tian, "Adversarial attack and defense on point sets," *arXiv preprint arXiv:1902.10899*, 2019.

[8] Y. Wen, J. Lin, K. Chen, C. P. Chen, and K. Jia, "Geometry-aware generation of adversarial point clouds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[9] H. Zheng, Z. Zhang, J. Gu, H. Lee, and A. Prakash, "Efficient adversarial training with transferable adversarial examples," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1181–1190.

[10] A. Dai, C. Ruizhongtai Qi, and M. Nießner, "Shape completion using 3d-encoder-predictor cnns and shape synthesis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5868–5877.

[11] X. Han, Z. Li, H. Huang, E. Kalogerakis, and Y. Yu, "High-resolution shape completion using deep neural networks for global structure and local geometry inference," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 85–93.

[12] D. Stutz and A. Geiger, "Learning 3d shape completion from laser scan data with weak supervision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1955–1964.

[13] 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*, 2015, pp. 1912–1920.

[14] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "Pcn: Point completion network," in *2018 International Conference on 3D Vision (3DV)*. IEEE, 2018, pp. 728–737.

[15] X. Wang, M. H. Ang Jr, and G. H. Lee, "Cascaded refinement network for point cloud completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 790–799.

[16] Z. Huang, Y. Yu, J. Xu, F. Ni, and X. Le, "Pf-net: Point fractal network for 3d point cloud completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7662–7670.

[17] M. Liu, L. Sheng, S. Yang, J. Shao, and S.-M. Hu, "Morphing and sampling network for dense point cloud completion," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 07, 2020, pp. 11 596–11 603.

[18] X. Yu, Y. Rao, Z. Wang, Z. Liu, J. Lu, and J. Zhou, "Pointr: Diverse point cloud completion with geometry-aware transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 498–12 507.

[19] D. Yin, R. Gontijo Lopes, J. Shlens, E. D. Cubuk, and J. Gilmer, "A fourier perspective on model robustness in computer vision," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[20] T. Zhang and Z. Zhu, "Interpreting adversarially trained convolutional neural networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7502–7511.

[21] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[22] C. He, Y. Zhang, and X. Li, "Principal directions estimation on point set surfaces," in *2013 Seventh International Conference on Image and Graphics*. IEEE, 2013, pp. 217–220.

[23] W. Schroeder, K. Martin, and B. Lorensen, "The visualization toolkit, 4th edn. kitware," *New York*, 2006.

[24] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China: IEEE, May 9-13 2011.

[25] D. Liu, R. Yu, and H. Su, "Extending adversarial attacks and defenses to deep 3d point cloud classifiers," in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 2279–2283.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.