Supplementary Materials for CAP: Robust Point Cloud Classification via Semantic and Structural Modeling

1. Details of the Robustness Evaluation

We first present the basic concepts in extreme value theory as follows. Common distributions such as Gaussian, Poisson and Exponential are light-tailed distributions, i.e.,

$$\int_0^{\infty} e^{tX} dF(X) < \infty, \forall t > 0 \tag{1}$$

where $F(X)$ is the cumulative density function. Such property makes it easy to model the data around the mean value, but hard to characterize the tail distribution of real-world data [20, 25]. To address the issue, the extreme value theory proposes to study the maximum of observations. Generally speaking, suppose n random variables $X_1, \cdots, X_n$ are i.i.d sampled from an arbitrary distribution $F(X)$, then the distribution of the maximum will degenerate to 0 because,

$$\lim_{n\to\infty} P(\max(X_1,\cdots,X_n) \le x) = \lim_{n\to\infty} F^n(x) = 0 \tag{2}$$

However, previous works find that, if we conduct a linear transformation on $M_n = \max(x_1, \cdots, x_n)$, the distribution of the maximum will not degenerate: [[4, 5]] If there exists sequences $(a_n)_{n\in\mathbb{Z}^+}, (\delta_n)_{n\in\mathbb{Z}^+}$ such that $\forall n \in \mathbb{Z}^+, a_n > 0$ and

$$\lim_{n\to\infty} P\Big(\frac{M_n - \delta_n}{a_n} \le x\Big) = G(x) \neq 0 \tag{3}$$

Then the class of $G(x)$ must be of the following form

$$G_\gamma(x) = \begin{cases} \exp\big[-(1+\gamma x)^{-1/\gamma}\big], 1+\gamma x > 0, \gamma \neq 0 \\ \exp\big[-e^{-x}\big], \gamma = 0 \end{cases} \tag{4}$$

, where γ is called the *extreme value index*. Besides, there exists a positive function a such that

$$\lim_{t\to x^*} P\Big(\frac{X-t}{a(t)} > x | X > t\Big) = H_\gamma(x), \quad \text{s.t. } X > t \tag{5}$$

, where $x^* = \sup\{x : F(x) < 1\}$, $a(t)$ is a real-valued function and $H_\gamma(x) = \log G(x)$.

In the theorem above, $G_\gamma(x)$ is conventionally called the class of extreme distributions, which describes the limiting probability of maximum of parent distribution $F(x)$. In consideration of the correlation between parent distribution $F(x)$ and limiting distribution $G_\gamma(x)$ [5], this theorem can be regarded as the law of large numbers for maximum to some extent [11].

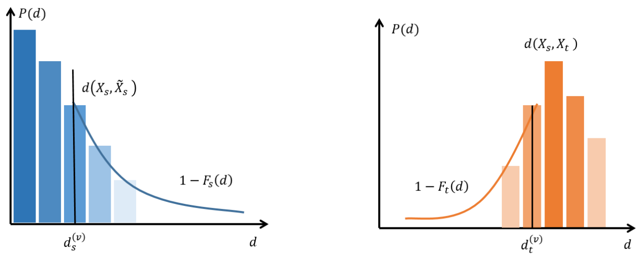

Figure 1. The tail distribution of $d(X_s, \tilde{X}_s)$ and $d(X_s, X_t)$.

Eq. 5 describes the distribution over the threshold, which could be used to characterize the tail distribution in our problem. Specifically, for the sampled $d(X_s, \tilde{X}_s)$, we set the threshold as the v^{th}-largest value, i.e., $d_s^{(v)}$. As such,

$$P(d > d_s^{(v)}) \approx \frac{v}{V} \tag{6}$$

where V is the sampling size. Then the conditional probability could be represented by,

$$P\Big(\frac{d - d_s^{(v)}}{\beta(d_s^{(v)})} \mid d > d_s^{(v)}\Big) = \frac{P((d - d_s^{(v)})/\beta(d_s^{(v)}))}{P(d > d_s^{(v)})} = H_{\gamma_s}(d) \tag{7}$$

where $\beta(d_s^{(v)})$ and γ_s are the scale and shape parameters respectively. After the transformation on d we obtain,

$$1 - P_s(d) \approx \frac{v}{V} \cdot \Big[1 + \gamma_s \cdot \frac{d - d_s^{(v)}}{\beta_s}\Big]^{-1/\gamma_s}, \quad d > d_s^{(v)} \tag{8}$$

where $\beta_s = \beta(d_s^{(v)})$ is the scale parameter and $1 - P_s(d)$ stands for $P(d_s > d)$. Similarly, for the sampled $d(X_s, X_t)$ we could obtain,

$$1 - P_t(d) \approx \frac{v}{V} \cdot \Big[1 + \gamma_t \cdot \frac{d_t^{(v)} - d}{\beta_t}\Big]^{-1/\gamma_t}, \quad d < d_t^{(v)} \tag{9}$$

where $1 - P_t(d)$ stands for $P(d_t < d)$, $d_t^{(v)}$ is the v^{th}-smallest distance, β_t and γ_t are the scale and shape parameters, respectively. Next, we demonstrate the inference of parameters in the distributions. For the shape parameters, we randomly sample several $d(X_s, \tilde{X}_s)$ and $d(X_s, X_t)$ with different s and t. Then we leverage the Pickands Estimator to estimate the shape parameters [9],

$$\gamma_s = \frac{1}{\ln 2} \log \frac{d_s^{(v)} - d_s^{(2v)}}{d_s^{(2v)} - d_s^{(4v)}} \tag{10}$$

where $d_s^{(2v)}$ is the $2v^{th}$-largest value in $d(X_s, \tilde{X}_s)$. We find that the inferred γ_s and γ_t for different s, t are close to 0.05. Therefore, we set $\gamma_s = \gamma_t = 0.05$ for all distributions in practice.

As for the scale parameters β_s and β_t, which mostly influence the fitted tail distribution in related literature [11], we estimate them by,

$$\begin{aligned} \beta_s &= \frac{1}{2} d_s^{(v)} \Theta_s (1 - \frac{\Theta_s^2}{\hat{\Theta}_s})^{-1} \\ \Theta_s &= \frac{1}{v} \sum_{\tau=0}^{v-1} \log d_s^{(\tau)} - \log d_s^{(v)} \\ \hat{\Theta} &= \frac{1}{v} \sum_{\tau=0}^{v-1} [d_s^{(\tau)} - \log d_s^{(v)}]^2 \end{aligned} \tag{11}$$

Finally, for the cross point of $1 - F_s(d)$ and $1 - F_t(d)$, since $\gamma_s = \gamma_t = \gamma$, we have,

$$\begin{aligned} & \Big[1 + \gamma \cdot \frac{d - d_s^{(v)}}{\beta_s}\Big]^{-1/\gamma} = \Big[1 + \gamma \cdot \frac{d_t^{(v)} - d}{\beta_t}\Big]^{-1/\gamma} \\ \Leftrightarrow \quad & \frac{d - d_s^{(v)}}{\beta_s} = \frac{d_t^{(v)} - d}{\beta_t} \\ \Leftrightarrow \quad & d_{\text{cross}} = \frac{\beta_t d_s^{(v)} + \beta_s d_t^{(v)}}{\beta_s + \beta_t} \end{aligned} \tag{12}$$

Then we could leverage the d_{cross} to calculate the $\rho(s, t)$. Note that if $d_s^{(v)} > d_t^{(v)}$ we set $\rho(s, t) = 1$ since there always exists $d_t < d_s$ in this case.

In summary, we develop a theoretical framework to certify the robustness under certain constraints. PointGuard [16] is a work similar to ours, which also develops a certified robust point cloud classification model. However, owing to the randomized smoothing technique, it is nontrivial to apply the framework to various kinds of classification models. Another work similar to ours is the certified robustness of a certain attack [17], i.e., the transformation attack. In comparison, the proposed CAP, with the aid of the attention-based pooling module and the dynamic contrastive learning paradigm, could be applied to various classification models and defend against multiple attack strategies. Recent work also proposes to leverage several semi-supervised tasks to improve the robustness of the point cloud classifier [24]. Nevertheless, our work achieves the same goal without introducing any new tasks or datasets.

Algorithm 1 The workflow of the proposed CAP.

```
1:  for each sample $X \in \mathcal{M}^s$ do
2:      Randomly select $X_s$ with the same label as $X$, and $X_t$ with other labels.
3:      Compute the classification loss $L_{\text{cls}}$ with $X$ and $X_t$.
4:      Compute the contrastive loss $L_{\text{manifold}}$ with $X, X_s$ and $X_t$.
5:      Update the model with the loss $L = L_{\text{cls}} + \lambda L_{\text{manifold}}$
6:      Repeat until convergence.
7:  for each label pair $(s, t)$ do
8:      for $v = 1, \cdots, V$ do
9:          Randomly sample a $X_s$ from label $s$
10:         Randomly sample $\tilde{X}_s$ under the constraints $\|\tilde{X}_s - X_s\|_\infty \leq \delta$
11:         Randomly sample $X_t$ from label $t$
12:         Calculate the distances $d(X_s, \tilde{X}^\delta)$ and $d(X_s, X_t)$ for $V$ times
13:     Estimate the parameters of the distribution $1 - P_s(d)$ and $1 - P_t(d)$
14:     Find the interaction between two distributions and compute the probability $\rho(s, t)$
```

2. Detailed Implementation

2.1. The Proposed CAP Defense Framework

We further present the details of applying our proposed CAP for training the three classification models. For the PointNet/PointNet++ [21, 22], we first leverage the original set abstraction module, which is built on the multi-layer perceptron (MLP), farthest point sampling (FPS) and neighborhood aggregation, to obtain the latent representations of the selected key points. After several set abstraction modules, instead of performing the max-pooling on all key points features, we use the proposed attention-based feature pooling module to obtain the global representation of a whole point cloud, which is later fed to the fully-connected classification layer. Similarly, for the PointCNN [12], we simply replace the original max-pooling layer with the proposed attention module. Finally, for the DGCNN [27], where there is no key point selection in the original design, i.e., the model outputs the latent representations of all points, we apply the random key point selection (RS) and max pooling on neighbors' features. Finally, we apply the proposed attention module before the classification layer.

We leverage a three-layered key point selection $(512, 256, 64)$ for all models. The hidden size of the features before the classification layer is 512. Other hyperparameters used in the key point selection (e.g., FPS and RS), neighborhood aggregation (e.g., max pooling and convolutional layer) and feature extraction (e.g., MLP, CNN, GCN) are set to be the same as in the original works. The

Table 1. Dataset description.

Dataset	Train size	Test size	# of classes
ModelNet40 [29]	9,843	2,468	40
ShapeNet [2]	35,708	15,419	55

learning rate for training is set as 0.001. For each batch, the number of X_s and X_t are 16 and 32 respectively. The λ in the whole loss is 0.5. The ϵ in the contrastive learning loss is 1.0. Furthermore, we leverage the SOR to filter outlier points for our framework. We also show the performance without the SOR in the ablation study. For the robustness evaluation, we consider the $\|\tilde{X} - X\|_\infty \leq \delta$ constraint when we generate the $\tilde{X}_s$. Specifically, for each point, we randomly add noises smaller than δ on each dimension, where $\delta = 0.05$ is used in this work. The main reason for measuring the L_∞ constraint is that it covers the largest perturbation budget in comparison to other constraints. For instance, the L_0 constraint limits the number of points to be perturbed, while the L_1 constraint considers the absolute value of a perturbation, which can be bounded by the L_∞ constraint. As such, the L_∞ constraint can better help us estimate the maximal radius under different attacks. For the adding or deleting operation, we randomly delete 5% points, copy some points and add noises with L_∞ constraints on these new points. The workflow of the training and robust evaluation of our CAP framework is shown in Alg. 1.

2.2. Experimental Setting

Dataset. In this paper, we run all our experiments on two datasets, i.e., ModelNet40 [29] and ShapeNet [2]. The ModelNet40 dataset consists of $12,311$ CAD models with 40 human-labeled object classes. We use the official split with $9,843$ training samples and $2,468$ testing samples. For the ShapeNet dataset, the original training set contains $35,708$ samples, while the test set contains $15,419$ samples, with a total number of classes of 55. For clarity, the description of the datasets is summarized in Table 1. For both datasets, we uniformly sample 2048 points for each point cloud and normalize them into a unit sphere. We apply no augmentation methods during our model training.

All the attacks and defenses are performed on the test set of both datasets. Specifically, for time efficiency, we perform GeoA3 [28] attack only on a subset with 800 samples consisting of 20 samples from each of the 40 classes from ModelNet40. Note that in the original work, the authors sampled a subset of 250 pairs of samples and target labels in their experiments as well. On the other hand, we subsample the test set of ShapeNet to obtain a set of $2,732$ samples with at most 60 samples from each class for performing all attacks. We believe that our sampling strategy can provide us with a subset of samples representative enough.

Attack Methods. We briefly summarize the attack methods involved in our work. Nine attack methods are considered in this paper, including Minimal [10], Smooth [18], IFGM [14], PGD [19], Gen3D-Add [30], Gen3D-Pert [30], KNN [26], GeoA3 [28], and ShapeInvariant (SI) [8]. Specifically, Minimal proposes to exploit the L_1 constraint to approximate the L_0 constraint, which tends to reduce the number of perturbed points. IFGM exploits normalized gradients to generate perturbations instead of the product of the sign of gradients and a perturbation clip threshold originally used in FGSM [6]. PGD simply applies the iterative gradient sign method to generate perturbation. Smooth attempts to obtain more robust gradients by adding random noises to a point cloud before generating perturbations based on IFGM and averaging them. Gen3D-Pert and Gen3D-Add stand for the perturbation and adding attacks proposed in the original work, where Gen3D-Pert imposes small perturbations on all points of a point cloud under the constraint of L_2 distance, while Gen3D-Add imposes the perturbations on a copy of a separate subset of points from the input point cloud, considered as the added points, under the constraint of Chamfer distance. KNN is another perturbation attack with an additional k-NN distance constraint other than Chamfer distance and additional clip operations based on L_∞ norms and normal vectors. GeoA3 takes Chamfer distance, Hausdorff distance and local curvature loss into account when performing a perturbation attack. SI uses specific designs to restrict the direction of perturbations on points. The optimization of the perturbations concerning all the attacks above is based on the C&W attack [1] except for IFGM and Smooth.

We provide the hyper-parameter settings for the attacks as follows for potential reproduction needs. For all attacks, we first apply the settings from the original works if provided and then tune them for better attack performance on a small validation set. For Minimal, we set the attack step size as 0.02, the number of iterations as 20 for ModelNet40 and 45 for ShapeNet, and the perturbation clip threshold ϵ as 0.5. For Smooth, we set the attack step size as 0.02, the number of iterations as 100, the clip threshold as 0.5, and the random noise is sampled from a Uniform distribution $U_{[0,0.1]}$, the number of iterations for searching robust gradients is 15. For IFGM, we set the attack step size as 0.02, the number of iterations as 100, and the clip threshold ϵ as 0.5. For Gen3D-Pert, we set the attack step size as 0.001, the number of iterations as 200 and the distance loss weight as 0.1. For Gen3D-Add, we set the number of points added as 256, the attack step size as 0.001, the number of iterations as 400 and the distance loss weight as 400.0. For KNN, we set the attack step size as 0.001, the number of iterations as 400, the κ used in the C&W attack as 5.0, the Chamfer distance weight as 10.0, the k-NN distance weight as 5.0,

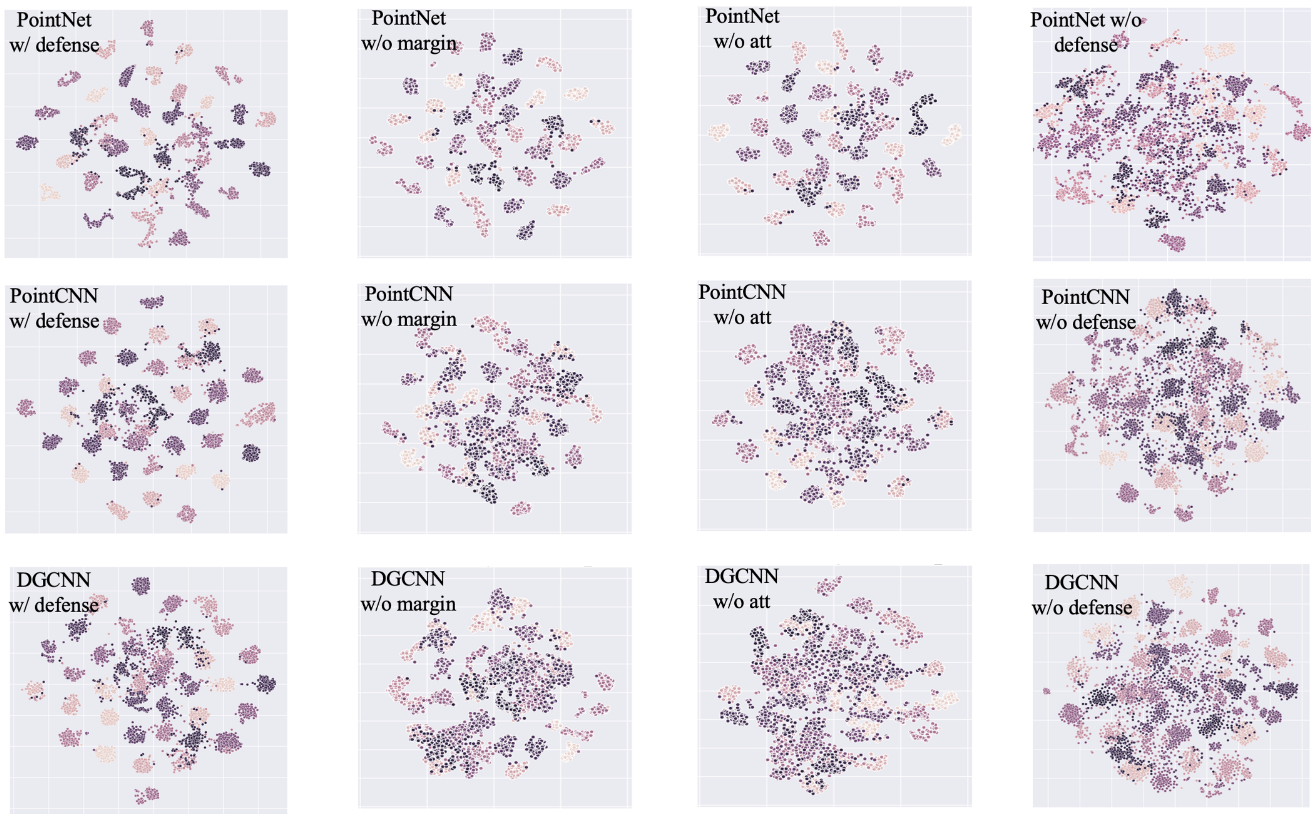

Figure 2. The visualization of global representations of three classification models on ModelNet40.

and the clip threshold is 0.8. For GeoA3, we set the max binary search steps of attack step size as 10, the number of iterations as 300, and the weights of Chamfer distance, Hausdorff distance and curvature loss are 10.0, 0.1, 1.0, respectively. All the attacks are performed with the settings above unless otherwise specified.

In fact, we have considered other attack methods, such as AdvPC [7] and Stick [15], as well. We didn't include the results of these attacks under the consideration of providing comparisons between models under unsatisfied attack effectiveness given our targeted attack setting, which is further detailed in the following paragraphs.

Defense Methods. We consider two recovery-based methods, i.e., SOR [23] and DUP-Net [32], as our defense baselines. SOR computes the k-NN distance for each point in a point cloud and removes those points with distances larger than $\mu + \alpha \cdot \sigma$, where μ and σ denote the mean and standard deviation of the distances. DUP-Net further utilizes an upsampling network [31] to enhance the visual quality of a point cloud after SOR. For the hyperparameters, we set k as 2 and α as 1.1 as described in the original work. We also consider four adversarial training-based methods: Vanilla adversarial training based on FGSM (AT) and PGD (AT-PGD) [18], Ensemble adversarial training (EAT) and PAGN [13]. For AT, we combined adversarial examples generated by the IFGM/PGD attack with benign samples for finetuning the vanilla classification model. As for Ensemble AT, the Gen3D-Pert attack is also leveraged for generating samples for training. Specifically, for PointNet, we take Gather-vector Guidance (GvG) [3] into consideration as well due to its special design of predicting gather vectors to detect adversarial examples.

While we take these methods to compare the defense performance of our models, we would like to point out that our framework is used for training a robust classification model, which is compatible with all the existing recovery-based defense methods, e.g., one can first utilize recovery methods to restore an adversarial example before inputting it into our model to further boost the robustness.

More Details. We first describe the design of random targeted attack in the main body in detail. For each sample, we randomly assign a label other than the ground truth label as the target label and maximize the model's prediction of this label. For attack methods that are originally designed for performing untargeted attacks, e.g., IFGM and Minimal, we alter their classification loss to a targeted one, i.e., from

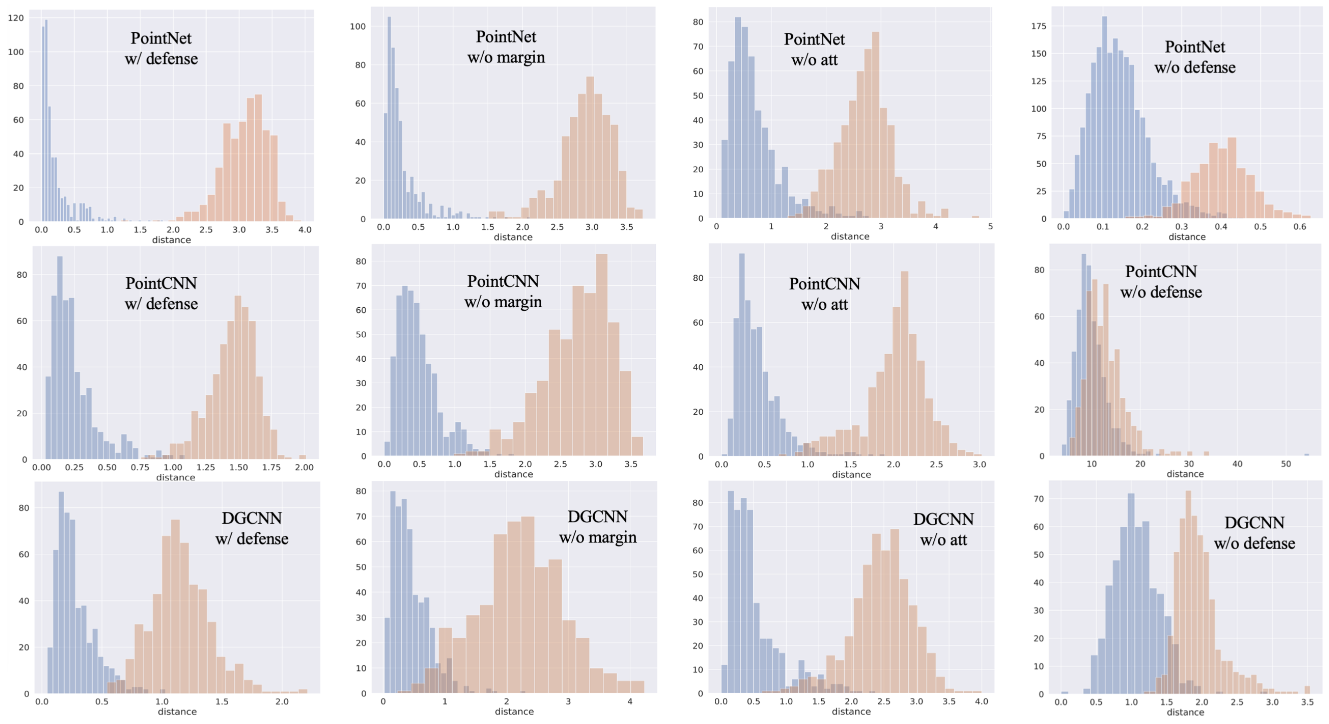

Figure 3. The distances between *chair* and *table* for three classification models on ModelNet40.

maximizing the loss against the ground truth label to minimizing the loss against the target label. Further, we may follow the design in the C&W-based attack [1] and use the margin logit loss.

Under such a design of targeted attacks, the computation of the attack success rate (ASR) is defined as,

$$\mathrm{ASR} = \frac{1}{N_{\text{test}}} \sum_{\tilde{X}} I(f_\theta(\tilde{X}), \tilde{y}) \tag{13}$$

where $\tilde{y}$ is the target label for each $\tilde{X}$ in the test set, N_{test} is the size of the testing set and $I(\cdot, \cdot)$ is the indicator function, where $I = 1$ if the model predicts the target label $\tilde{y}$ and otherwise 0.

As for the untargeted attack in the appendix, the attack is set to aim at misleading the prediction of the model without any designated label, i.e., causing the model to predict any label other than the ground truth one. Therefore, we utilize classification accuracy to measure the effectiveness of an attack. Specifically, we take IFGM, Minimal and Gen3D-Pert as representative attacks for the corresponding experiments.

For the implementation of the classification models and baseline methods, we directly apply those with released PyTorch implementations. For those without source code released or with TensorFlow implementations only, we implement them according to their work. We trained all the models ourselves including those used for classification and those for defense baselines, e.g. DUP-Net [32].

All the experiments are conducted on a machine with a 20-core CPU, 96 GBs of memory and 6 NVIDIA 2080Ti GPUs.

3. Additional Experimental Results

3.1. Comprehensive Results

We first present the results of the untargeted attacks on ModelNet40. The classification accuracy under attacks and defenses for three models are reported in Table 2. As we can see from the results, the IFGM attack has such strong attack effectiveness that the vanilla PointNet model has only 1.6% classification accuracy. While the baseline defense methods can help relieve part of the attack effectiveness, our CAP helps the model surpass both the recovery-based and the adversarial training baseline methods. However, the robustness of DGCNN is worse than PointNet and PointCNN. We infer the main reason to be that we directly add a naive downsampling module on the original model, i.e., the random sampling and max pooling. We believe that the results would be much better if we could find a more effective downsampling module. Furthermore, the accuracy still drops even if we adopt the proposed CAP, where the root reason lies on that there exists labels that share similar se-

Table 2. Classification accuracy of ModelNet40 adversarial examples generated by untargeted attacks on PointNet, DGCNN and PointCNN with various defense methods.

	PointNet							DGCNN							PointCNN						
	Vanilla	SOR	DUP-Net	AT	EAT	PAGN	CAP	Vanilla	SOR	DUP-Net	AT	EAT	PAGN	CAP	Vanilla	SOR	DUP-Net	AT	EAT	PAGN	CAP
IFGM	1.6%	23.6%	24.8%	15.1%	9.6%	24.1%	25.5%	0.0%	11.9%	32.1%	22.7%	28.7%	10.5%	2.0%	40.8%	59.9%	54.2%	47.2%	32.7%	63.2%	64.2%
Minimal	32.5%	63.0%	61.4%	59.6%	51.9%	61.8%	41.7%	12.6%	37.9%	35.9%	59.0%	61.9%	41.0%	44.6%	66.6%	67.8%	40.3%	57.7%	48.3%	68.8%	70.0%
Gen3D-Pert	64.4%	63.4%	63.0%	48.1%	54.1%	63.5%	65.4%	34.6%	35.2%	37.1%	28.6%	32.9%	34.9%	58.5%	48.9%	51.2%	37.1%	42.3%	30.3%	45.4%	66.9%

Table 3. Recovery rate of ModelNet40 adversarial examples generated by targeted attacks on PointNet, DGCNN and PointCNN with various defense methods.

	PointNet						DGCNN					PointCNN				
	Vanilla	SOR	DUP-Net	AT-PGD	GvG	CAP	Vanilla	SOR	DUP-Net	AT-PGD	CAP	Vanilla	SOR	DUP-Net	AT-PGD	CAP
Minimal	60.3%	60.3%	46.9%	70.3%	64.3%	42.3%	56.4%	54.9%	54.9%	85.0%	60.3%	81.8%	78.1%	77.4%	75.7%	79.6%
Smooth	28.3%	54.9%	54.9%	38.4%	31.2%	50.0%	2.5%	33.8%	33.8%	31.8%	29.3%	72.9%	68.8%	68.8%	67.6%	70.8%
IFGM	18.0%	18.0%	56.6%	22.5%	23.6%	49.3%	0.0%	71.6%	71.6%	3.8%	62.8%	68.2%	72.2%	72.2%	62.7%	76.5%
PGD	7.5%	51.8%	51.8%	18.9%	9.3%	68.4%	8.4%	21.5%	21.5%	31.8%	81.8%	60.7%	59.0%	59.0%	78.6%	82.9%
Gen3D-Add	10.9%	0.9%	39.0%	18.1%	16.5%	57.4%	24.6%	0.5%	36.8%	73.8%	52.1%	80.6%	1.2%	76.1%	78.6%	77.4%
Gen3D-Pert	7.7%	59.8%	58.1%	14.3%	9.1%	69.4%	2.3%	43.5%	43.5%	9.4%	67.7%	29.2%	27.7%	12.7%	61.8%	68.1%
KNN	5.5%	40.6%	41.1%	0.2%	1.3%	47.7%	0.5%	32.1%	32.1%	1.9%	53.5%	8.7%	0.4%	13.2%	44.4%	52.4%
ShapeInvariant	10.6%	40.6%	34.5%	18.9%	64.3%	68.4%	11.5%	5.8%	54.9%	31.8%	60.3%	13.0%	59.0%	72.2%	62.7%	52.4%
Avg.	18.6%	40.9%	47.9%	25.2%	27.5%	56.6%	13.3%	33.0%	43.6%	33.7%	58.5%	51.9%	45.8%	56.4%	66.5%	70.0%

mantic information as we have discussed, e.g., the flower pot and the vase, which opens a door for the attacker to successfully create untargeted adversarial examples in this problem.

To help understand how the proposed CAP enhances the classifier's robustness against various attacks, we further report the recovery rate under various targeted attacks of three models on ModelNet40 in Table 3. Specifically, the recovery rate represents the classification accuracy rather than the attack success rate for targeted attacks, which could demonstrate the actual recovery ability of a defense method. From the table, we can conclude that our proposed CAP shows consistent superiority in comparison to other defense baselines.

We present the visualization of the global representations for all models trained w/ and w/o our CAP in Fig. 2. From the figure, we can see that the vanilla models learn entangled representations, which explains why they are vulnerable to adversarial attacks in our experiments as well. After training with CAP, the features of samples with different labels are separated, i.e., the model learns disentangled manifolds. To further validate this, we present the distances of $d(X, X_s)$ and $d(X_s, X_t)$ in Fig. 3, where the s and t are *chair* and *table*, respectively. The distribution of distances shows that the proposed contrastive learning loss and the attention module could separate objects with similar labels well.

3.2. Ablation Study

We conduct the following ablation study to inspect the actual performance gain of our proposed attention-based feature pooling module and contrastive loss qualitatively and quantitatively. We remove L_{margin} and the attention feature pooling module from CAP respectively to train two versions of models denoted as w/o margin and w/o att. Similar to Appendix 3.1, we visualize the global representations extracted by these models with t-SNE, and the distances between samples with label *chair* and *table* for three classification models in Fig. 2 and Fig. 3, respectively. The distance distributions of models trained without the attention module indicate that there still exist some small overlaps between the manifolds of the chair and the table even though we conduct the standard contrastive learning. We infer the main reason is that, since the two kinds of objects often share similar parts, it is difficult to learn disentangled representations if we simply leverage max-pooling on all points. On the other side, we observe similar results when disabling the margin loss in the contrastive learning. These qualitative results show the effectiveness of the proposed attention module and the margin loss. In addition, we point out that models trained without these two modules can still learn better-disentangled manifolds compared with the vanilla models.

For quantitative comparisons, we perform the Gen3D-Pert and KNN attacks against three classification models on ModelNet40 and compare their ASRs with the corresponding full CAP w/ and w/o attention module, margin loss and SOR preprocessing in Table 4. As shown by the results, the attention module and margin loss do not influence the averaged ASRs a lot, which may be explained by the somehow disentangled manifolds learned. To successfully attack them, an adversary needs to find the samples in the overlapped area of two manifolds and conduct the optimal attack as mentioned in Sec. 4.1. However, current attack methods only manipulate the pre-defined samples instead of finding the most effective ones, which may not be able to be within the overlapped area with large probabilities. As such, it is still difficult to move the samples across the decision bound-

Table 4. Attack success rate of ModelNet40 adversarial examples under Gen3D-Pert and KNN attacks on three classification models w/ our full CAP, w/o SOR preprocessing (w/o SOR), w/o attention module (w/o att), and w/o L_{margin} (w/o margin).

	PointNet				DGCNN				PointCNN			
	CAP	w/o SOR	w/o att	w/o margin	CAP	w/o SOR	w/o att	w/o margin	CAP	w/o SOR	w/o att	w/o margin
Gen3D-Pert	1.5%	3.9%	1.5%	1.5%	3.3%	4.2%	1.9%	2.9%	1.8%	2.1%	1.9%	1.9%
KNN	4.0%	7.4%	4.6%	3.2%	6.4%	7.3%	4.1%	6.4%	3.3%	4.1%	6.0%	4.5%
Avg.	2.7%	5.7%	3.1%	2.3%	4.8%	5.7%	3.0%	4.6%	2.5%	3.1%	4.0%	3.2%

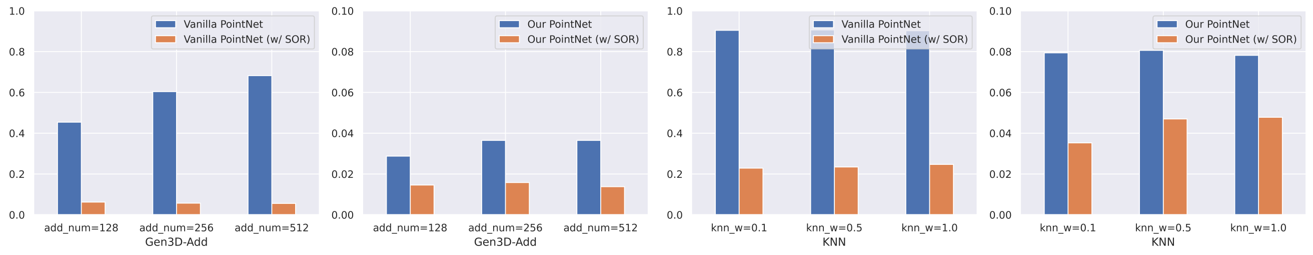

Figure 4. Attack success rate of ModelNet40 adversarial examples under Gen3D-Add (left two) and KNN (right two) attacks with various hyper-parameter settings on vanilla PointNet and our PointNet w/o and w/ SOR preprocessing.

ary. On the other side, the SOR module does not contribute a large part to the robustness of models with CAP. For instance, the ASR of the PointCNN only drops from 3.1% to 2.5% after using the SOR module. Nevertheless, it could help models with CAP to filter outlier points in some cases. Besides, one advantage of our framework is that we could leverage the existing recovery-based defense methods such as SOR and DUP-Net to further improve the robustness of the classification models.

3.3. Hyper-parameters

To further evaluate the robustness of models trained with our framework. We adjust the hyper-parameters of the attack methods to inspect the ASRs under different attack effectiveness. Specifically, we take Gen3D-Add and KNN attacks for representing normal attacks and shape-invariant attacks, respectively. For Gen3D-Add, we adjust the number of added points to be $128, 256, 512$. For KNN, we set the weight of the k-NN distance loss to be $0.1, 0.5, 1.0$. The ASRs of ModelNet40 on PointNet w/ and w/o our framework under these attacks are demonstrated in Fig. 4.

We analyze the results of both attacks one by one. On one hand, for Gen3D-Add attacks, the ASR w/o SOR defense rises as the number of added points increases, while the ASR w/ SOR slightly declines, which indicates that the more perturbations introduced, the easier it is to be filtered by SOR defense. On the other hand, for KNN attacks, the ASR w/o SOR defense plateaus when raising the weight of k-NN distance loss, while the ASR w/ SOR slightly rises, which shows that for shape-invariant attacks, when the attack reaches its full effectiveness, the shape-related loss can enhance the stealthiness of the generated adversarial examples against defenses. Moreover, for both attacks, different hyper-parameter settings do not affect the performance of our models much. For instance, even for the KNN attack, the ASR only rises from 3.5% to 4.8% when the k-NN loss weight increases.

References

[1] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017. 3, 5

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

[3] Xiaoyi Dong, Dongdong Chen, Hang Zhou, Gang Hua, Weiming Zhang, and Nenghai Yu. Self-robust 3d point recognition via gather-vector guidance. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11513–11521. IEEE, 2020. 4

[4] R. A. Fisher and L. H. C. Tippett. Limiting forms of the frequency distribution of the largest or smallest member of a sample. *Mathematical Proceedings of the Cambridge Philosophical Society*, 24(2):180–190, 1928. 1

[5] Gnedenko. Sur la distribution limite du terme maximum d'une série aléatoire. *Annals of Mathematics*, 44(3):423–453, 1943. 1

[6] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2015. 3

[7] Abdullah Hamdi, Sara Rojas, Ali Thabet, and Bernard Ghanem. Advpc: Transferable adversarial perturbations on 3d point clouds. In *European Conference on Computer Vision*, pages 241–257. Springer, 2020. 4

[8] Qidong Huang, Xiaoyi Dong, Dongdong Chen, Hang Zhou, Weiming Zhang, and Nenghai Yu. Shape-invariant 3d adver-

sarial point clouds. *arXiv preprint arXiv:2203.04041*, 2022. 3

[9] James Pickands Iii. Statistical inference using extreme order statistics. *Annals of Statistics*, 3(1):119–131, 1975. 2

[10] Jaeyeon Kim, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Minimal adversarial examples for deep learning on 3d point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7797–7806, 2021. 3

[11] Samuel Kotz and Saralees Nadarajah. *Extreme value distributions. Theory and applications*. Prentice Hall,, 2000. 1, 2

[12] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems*, 31, 2018. 2

[13] Qi Liang, Qiang Li, Weizhi Nie, and An-An Liu. Pagn: perturbation adaption generation network for point cloud adversarial defense. *Multimedia Systems*, pages 1–9, 2022. 4

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

[15] Daniel Liu, Ronald Yu, and Hao Su. Adversarial shape perturbations on 3d point clouds. In *European Conference on Computer Vision*, pages 88–104. Springer, 2020. 4

[16] Hongbin Liu, Jinyuan Jia, and Neil Zhenqiang Gong. Pointguard: Provably robust 3d point cloud classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6186–6195, 2021. 2

[17] Tobias Lorenz, Anian Ruoss, Mislav Balunović, Gagandeep Singh, and Martin Vechev. Robustness certification for point cloud models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7608–7618, 2021. 2

[18] Chengcheng Ma, Weiliang Meng, Baoyuan Wu, Shibiao Xu, and Xiaopeng Zhang. Towards effective adversarial attack against 3d point cloud classification. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2021. 3, 4

[19] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 3

[20] T Okubo and N Narita. On the distribution of extreme winds expected in japan. *National Bureau of Standards Special Publication*, 560(1), 1980. 1

[21] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2

[22] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 2

[23] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Mihai Dolha, and Michael Beetz. Towards 3d point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941, 2008. 4

[24] Jiachen Sun, Yulong Cao, Christopher B Choy, Zhiding Yu, Anima Anandkumar, Zhuoqing Morley Mao, and Chaowei Xiao. Adversarially robust 3d point cloud recognition using self-supervisions. *Advances in Neural Information Processing Systems*, 34, 2021. 2

[25] Jonathan A Tawn. Estimating probabilities of extreme sea-levels. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 41(1):77–93, 1992. 1

[26] Tzungyu Tsai, Kaichen Yang, Tsung-Yi Ho, and Yier Jin. Robust adversarial objects against deep learning models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 954–962, 2020. 3

[27] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 2

[28] Yuxin Wen, Jiehong Lin, Ke Chen, CL Philip Chen, and Kui Jia. Geometry-aware generation of adversarial point clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 3

[29] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 3

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

[31] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2790–2799, 2018. 4

[32] Hang Zhou, Kejiang Chen, Weiming Zhang, Han Fang, Wenbo Zhou, and Nenghai Yu. Dup-net: Denoiser and upsampler network for 3d adversarial point clouds defense. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1961–1970, 2019. 4, 5