A. Proof of Theorem 4.2

We prove the theorem by constructing a base point cloud classifier f' and an adversarial point cloud P'. Suppose $P_1, P_2, \cdots, P_m$ are the m sub-point clouds for the point cloud P. We let f' predict label y for P_j, where $j = 1, 2, \cdots, M_y(P)$, and predict label l ($l \neq y$) for $M_l(P)$ sub-point clouds among the remaining $m - M_y(P)$ sub-point clouds. When an attacker can arbitrarily add (or delete or modify) at most $t'(P) + 1$ points to P, we construct the following P'. For point addition (or deletion) attacks, we can find P' such that a point is added (or deleted) to P_j, where $j = 1, 2, \cdots, t'(P) + 1$. For simplicity, we use P'_j to denote the corresponding sub-point cloud by adding (or deleting) a point to P_j. For point modification/perturbation attacks, we can find P' such that a point is deleted from $P_{2 \cdot j - 1}$ and a point is added to $P_{2 \cdot j}$, where $j = 1, 2, \cdots, t'(P) + 1$. For simplicity, we use $P'_{2 \cdot j - 1}$ and $P'_{2 \cdot j}$ to denote the corresponding sub-point clouds. Suppose $l' = \text{argmax}_{l \neq y}(M_l(P) + \mathbb{I}(y > l))$. We let f' predict label l' for sub-point clouds P'_j, where $j = 1, 2, \cdots, \tau \cdot (t'(P) + 1)$ and τ is 1 (or 1 or 2 or 2) for point addition (or deletion or modification or perturbation) attacks. Given the constructed f' and P', we have $M_y(P') = M_y(P) - \tau \cdot (t'(P) + 1)$ and $M_{l'}(P') = M_{l'}(P) + \tau \cdot (t'(P) + 1)$. Then, we have the following:

$$
\begin{aligned}
& M_{l'}(P') + \mathbb{I}(y > l') && (3)\\
=& M_{l'}(P) + \tau \cdot (t'(P) + 1) + \mathbb{I}(y > l') && (4)\\
=& M_{l'}(P) + (2 \cdot \tau - \tau) \cdot (t'(P) + 1) + \mathbb{I}(y > l') && (5)\\
>& M_{l'}(P) + 2 \cdot \tau \cdot \frac{M_y(P) - (M_{l'}(P) + \mathbb{I}(y > l'))}{2 \cdot \tau} \\
& - \tau \cdot (t'(P) + 1) + \mathbb{I}(y > l') && (6)\\
=& M_y(P) - \tau \cdot (t'(P) + 1) && (7)\\
=& M_y(P'). && (8)
\end{aligned}
$$

We have Equation (6) from Equation (5) based on the fact that $\lfloor x \rfloor + 1 > x$, where $\lfloor \cdot \rfloor$ is floor function and x is an arbitrary non-negative real number. Therefore, the ensemble point cloud classifier h' built upon f' predicts label l' instead of y for the constructed point cloud P'.

B. Loss Term Proposed by Fan et al. [8]

Fan et al. [8] proposed the following loss term $L_p(\mathcal{D}_u, \mathcal{C})$ (Chamfer Distance):

$$
\begin{aligned}
L_p(\mathcal{D}_u, \mathcal{C}) = \frac{1}{|\mathcal{D}_u|} \sum_{(P_s, P_p) \in \mathcal{D}_u} [\frac{1}{|\mathcal{C}(P_s)|} \sum_{\mathbf{e}_s \in \mathcal{C}(P_s)} \min_{\mathbf{e}_p \in P_p} \|\mathbf{e}_s - \mathbf{e}_p\|_2 \\
+ \frac{1}{|P_p|} \sum_{\mathbf{e}_p \in P_p} \min_{\mathbf{e}_s \in \mathcal{C}(P_s)} \|\mathbf{e}_s - \mathbf{e}_p\|_2], \qquad (9)
\end{aligned}
$$

where $\mathcal{C}(P_s)$ is the completed point cloud outputted by $\mathcal{C}$ for the sub-point cloud P_s, and $\mathbf{e}_s$ (or $\mathbf{e}_p$) is a point in $\mathcal{C}(P_s)$ (or P_p).

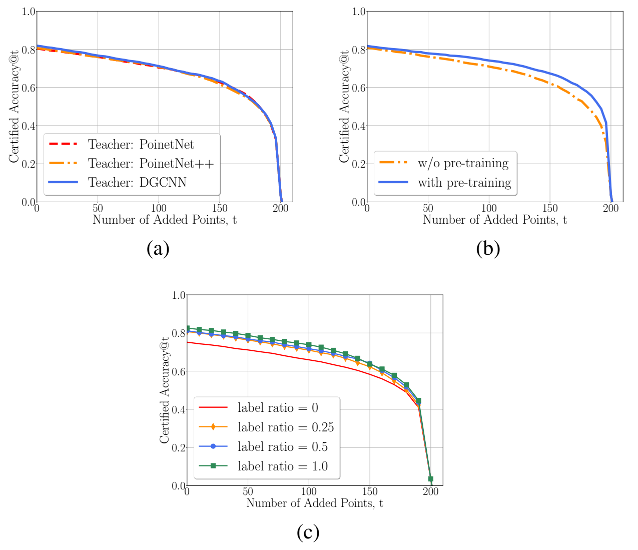


Figure 8. (a) Impact of the teacher model's architecture on the certified accuracy of PointCert in the black-box setting of Scenario III. Student model architecture is PointNet and dataset is ModelNet40. (b) Pre-trained PCN improves PointCert in Scenario III. (c) Impact of the fraction of a customer's labeled point clouds on PointCert in Scenario III.

C. Dataset Description

We adopt two publicly available benchmark datasets, namely ModelNet40 [42] and ScanObjectNN [36], in our evaluation. In particular, ModelNet40 contains 9,843 training point clouds and 2,468 testing point clouds. Each point cloud has 10,000 points on average and belongs to one of the 40 categories. In ScanObjectNN dataset, the number of training point clouds and the number of testing point clouds are respectively 2,319 and 583. The total number of classes in this dataset is 15. ScanObjectNN dataset has two variants, namely ScanObjectNN-OBJ_Only and ScanObjectNN-OBJ_BG. The difference is that the object in ScanObjectNN-OBJ_Only does not have background while the object in ScanObjectNN-OBJ_BG has. We use both variants. In accordance with ModelNet40, we keep at most 10,000 points in each point cloud in ScanObjectNN. On average, each point cloud has 9,594 and 9,774 points respectively for the two variants. Under the same setting, we compare with existing defenses and conduct our experiments using raw point clouds to simulate real-world attack scenarios. It is noted that our PointCert still outperforms previous defenses when the point clouds are all sub-sampled to reduce their sizes as previous defenses [23].

D. Details of Compared Methods.

We compare PointCert with undefended model, randomized smoothing [5], and PointGuard [23].

- **Undefended model:** Undefended model is a base point cloud classifier that is trained and tested in the standard way. It does not have certified robustness guarantees.

- **Randomized smoothing [5]:** Randomized smoothing builds a certifiably robust classifier via adding a zero-mean Gaussian noise with standard deviation σ to an input. In particular, given a testing point cloud, randomized smoothing constructs N noisy point clouds, each of which is constructed by adding random Gaussian noise to each dimension of each point of the point cloud. Then, randomized smoothing uses a point cloud classifier to predict the labels of the noisy point clouds and takes a majority vote among the predicted labels as the final predicted label of the point cloud. Randomized smoothing provably predicts the same label for a point cloud when the ℓ_2-norm of the adversarial perturbation added to its points is less than a threshold (called *certified radius*).

  When applied to point cloud classification, randomized smoothing can only derive certified radius against point modification attacks [23]. Moreover, we can transform certified radius to certified perturbation size via employing the relationship between ℓ_2-norm and ℓ_0-norm. In particular, suppose the points in a (adversarial) point cloud lie in a space (denoted as Ω). We assume the largest ℓ_2-norm distance between two arbitrary points in the space Ω is bounded by η. In other words, we have $\eta \geq \max_{\omega_1 \in \Omega, \omega_2 \in \Omega} \|\omega_1 - \omega_2\|_2$. Note that η could be different for different datasets. For instance, η is respectively $2\sqrt{3}$ and $\sqrt{15}$ on ModelNet40 and ScanObjectNN datasets. Given a certified radius γ (under ℓ_2-norm) obtained by randomized smoothing and the η, the certified perturbation size can be computed as $\lfloor \gamma^2/\eta^2 \rfloor$.

- **PointGuard [23]:** PointGuard is the state-of-the-art certified defense against adversarial point clouds. Roughly speaking, given a testing point cloud, PointGuard first creates N subsampled point clouds, each of which is obtained by randomly subsampling k (a parameter in PointGuard) points from the given point cloud. Then, PointGuard uses a base point cloud classifier to predict labels for those subsampled point clouds. Finally, PointGuard counts the number of subsampled point clouds whose predicted labels are l ($l = 1, 2, \cdots, c$). The label with the largest count is viewed as the predicted label for the given point cloud. PointGuard provably predicts the same label for a point cloud when the number of arbitrarily added, deleted, and/or modified points is less than a threshold, which is the certified perturbation size.

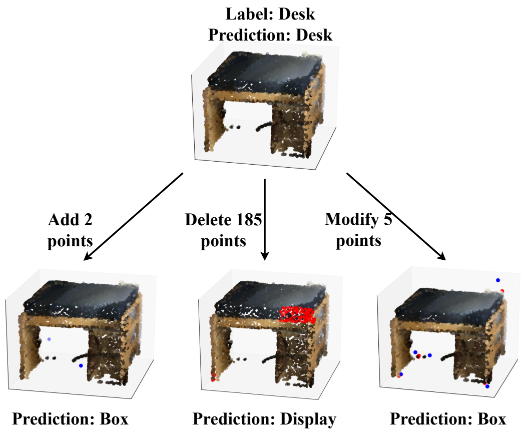


Figure 9. Illustration of *point addition attack* (left), *point deletion attack* (middle), and *point modification attack* (right). Blue (or red) points are added (or deleted) by an attacker.

E. Details of Our Empirical Attacks.

In our experiments, we adopt the strong point addition, modification, and perturbation attacks developed by [43] and point deletion attack developed by [40] to attack an undefended model. Roughly speaking, Xiang et al. [43] formulated point addition (or modification or perturbation) attack as an optimization problem, i.e., adversarial points can be crafted by minimizing a loss function using gradient descent. Wicker et al. [40] developed an algorithm to identify a set of critical points in a point cloud whose removal would make a point cloud classifier predict an incorrect label for the point cloud.

Since there are no existing adversarial point cloud attacks tailored to randomized smoothing, PointGuard, and PointCert, we generalize existing attacks to them to compute Empirical Accuracy@t:

For **generalized point addition attack**, we iteratively add t points using the attack in [43]. In particular, in the ith iteration ($i = 1, 2, \cdots, t$), we generate multiple noisy point clouds with Gaussian noise (or subsampled point clouds or sub-point clouds) from the testing point cloud with $i-1$ adversarially added points in randomized smoothing (or PointGuard or PointCert). Then, we find a point such that the average loss (i.e., cross-entropy loss) of the base point cloud classifier on the noisy point clouds (or subampled point clouds or sub-point clouds) is maximized when the point is added to them. We use gradient descent to find the point. Moreover, to consider a powerful attack, we do not restrict the dimension values of the point.

For **generalized point deletion attack**, we first generate multiple noisy point clouds (or subsampled point clouds or sub-point clouds) from a point cloud and then use the point deletion attack in [40] to identify a set of critical points for each of them. Finally, we count the number of times for

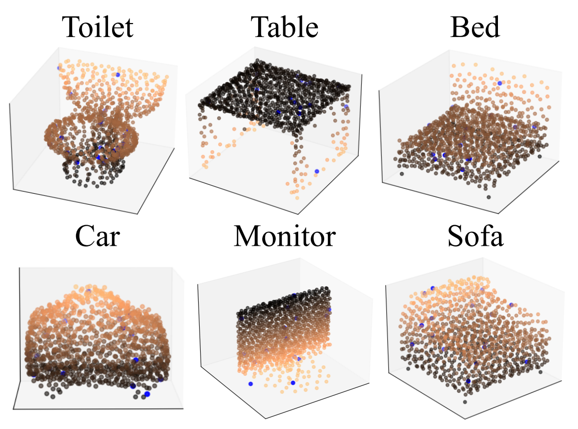


Figure 10. More visual illustrations of point completion with sub-point clouds. Each point cloud is reconstructed by a very small number of points (at most 32 points), colored in blue. Motivated by this observation, we utilize point cloud completion to build robust classifiers on the customer side in Scenario III.

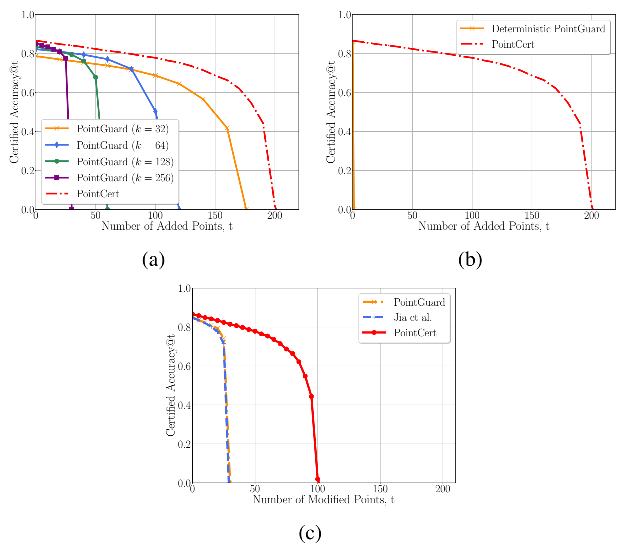


Figure 11. (a) Comparing PointCert and PointGuard with different k. (b) Comparing PointCert with deterministic PointGuard. (c) Comparing PointCert and Jia et al. [15].

each point being identified as a critical point and delete the t points with the largest counts from the point cloud.

Our **generalized point modification (or perturbation) attack** against randomized smoothing, PointGuard, and PointCert is a combination of our point addition and deletion attacks. Specifically, we first use our point deletion attack to delete t points in a testing point cloud and then use our point addition attack to add t points to the point cloud.

F. Additional Experiments

F.1. Comparing PointCert and PointGuard with Different k

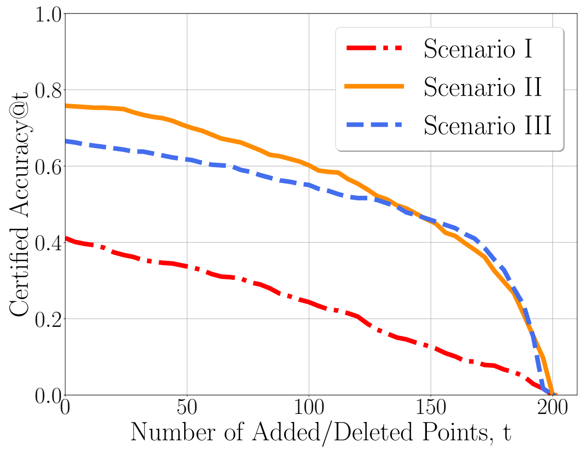
Certified defenses have accuracy-robustness trade-offs, which are controlled by their parameters (e.g., m in PointCert and k in PointGuard). By default, we use PointGuard with $k = 256$ such that PointGuard and PointCert have similar accuracy under no attacks for fair comparison of robustness. We also compare PointCert ($m = 400$) and PointGuard with different k, where the same k is used for both training and inference. Figure 11(a) shows the results when PointGuard uses different k. When k is very small, PointGuard can tolerate more perturbed points, but its certified accuracy under no attacks is much lower than PointCert. The reason is that PointGuard estimates probability bounds using a Monte-Carlo algorithm when computing certified robustness.

F.2. Comparing PointCert with Deterministic PointGuard

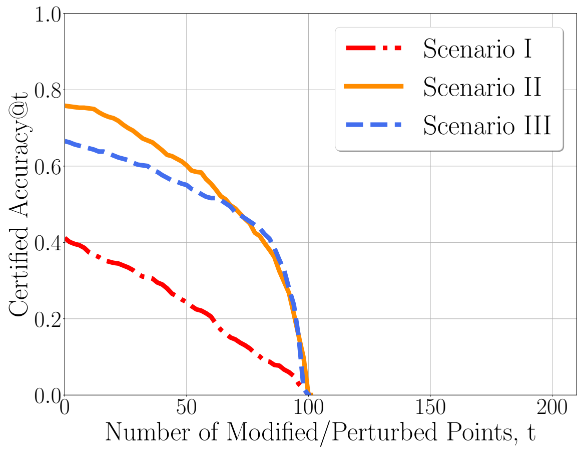
We can make PointGuard deterministic via fixing the seed in the random number generator. Moreover, we can extend our techniques for PointCert to derive tight certified robustness guarantees of such deterministic PointGuard. However, the certified accuracy of such deterministic PointGuard is low, as shown in Figure 11(b). This is because a single newly added adversarial point can influence all subsampled point clouds in the worst-case for PointGuard. However, a single added adversarial point can only influence one sub-point cloud for PointCert because the sub-point clouds are disjoint. Note that, when the guarantees are probabilistic, PointGuard achieves larger certified accuracy because the probability that the worst-case happens is small and can be tolerated within the error probability.

F.3. Comparing PointCert with Jia et al. [15]

Jia et al. [15] develops almost tight l_0-norm certified robustness of top-k predictions for image classification. Given a testing image I, their method creates different ablated inputs via retaining b randomly selected pixels of I and setting the remaining pixels to a special value. Then, they feed the ablated inputs to a base classifier and count the probabilities that the base classifier outputs for each label. In this way, they build a smoothed classifier that outputs top-k predictions with l_0-norm certified robustness. When extending their method to point cloud classification, we set $k = 1$ and create ablated point clouds via retaining b randomly selected points while setting the remaining points to a special value. From Figure 11(c), we observe that Jia et al. achieves similar certified accuracy with PointGuard, which is lower than PointCert. The reason is that these two methods both use randomly selected/subsampled points for certification. We note that Jia et al. is only applicable to point modification attacks.

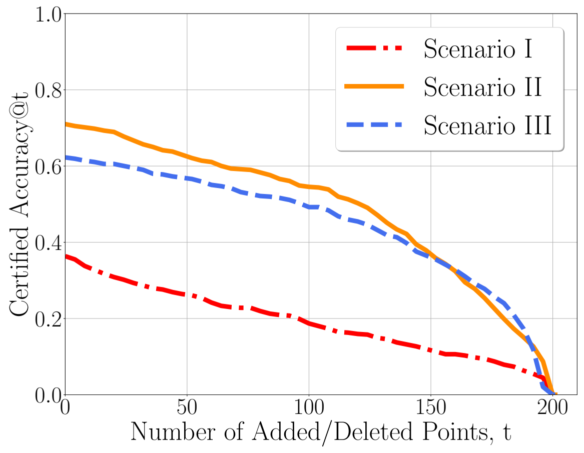


(a) Point addition/deletion attacks

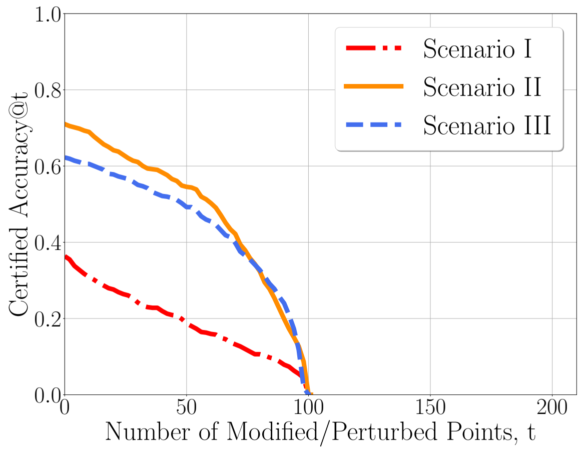


(b) Point modification/perturbation attacks

Figure 12. Comparing the certified accuracy of PointCert in the three application scenarios under different attacks. The dataset is ScanObjectNN-OBJ_Only.

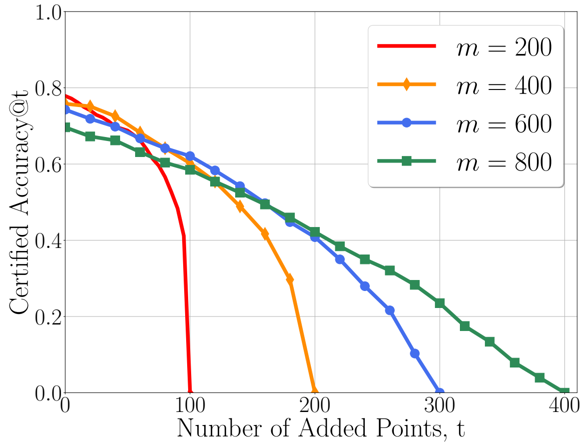


(a) Point addition/deletion attacks

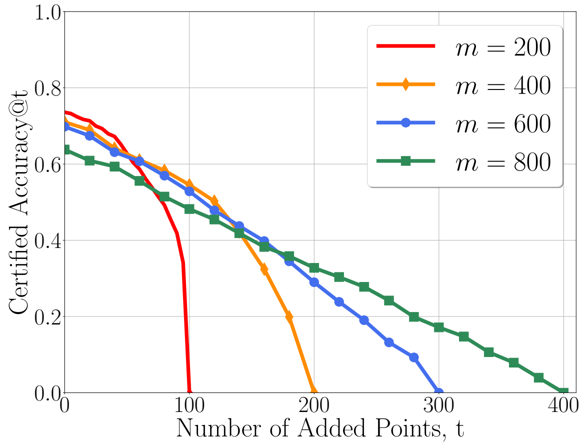


(b) Point modification/perturbation attacks

Figure 13. Comparing the certified accuracy of PointCert in the three application scenarios under different attacks. The dataset is ScanObjectNN-OBJ_BG.

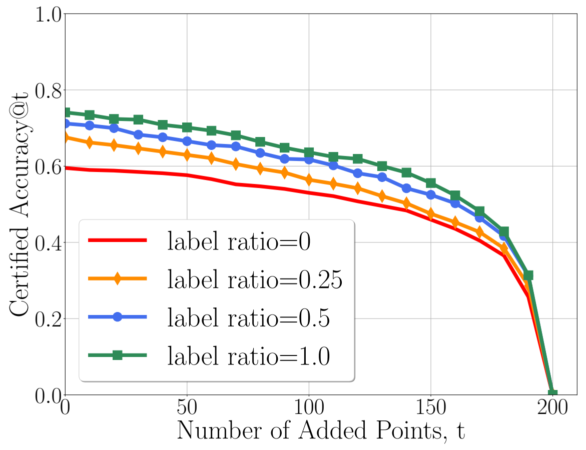


(a) ScanObjectNN-OBJ_Only

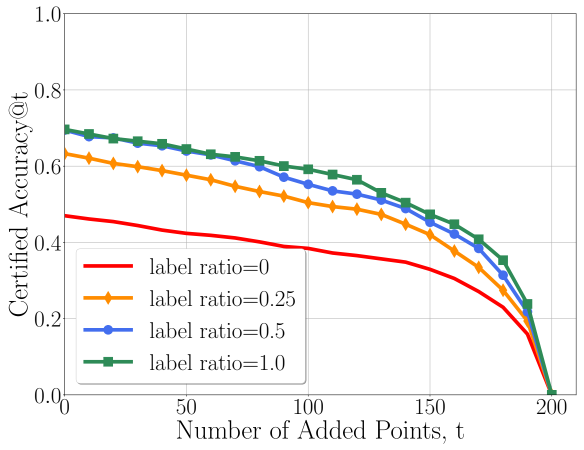


(b) ScanObjectNN-OBJ_BG

Figure 14. Impact of m on the certified accuracy of PointCert in Scenario II.



(a) ScanObjectNN-OBJ_Only



(b) ScanObjectNN-OBJ_BG

Figure 15. Impact of the fraction of a customer's labeled point clouds on the certified accuracy of PointCert in Scenario III.

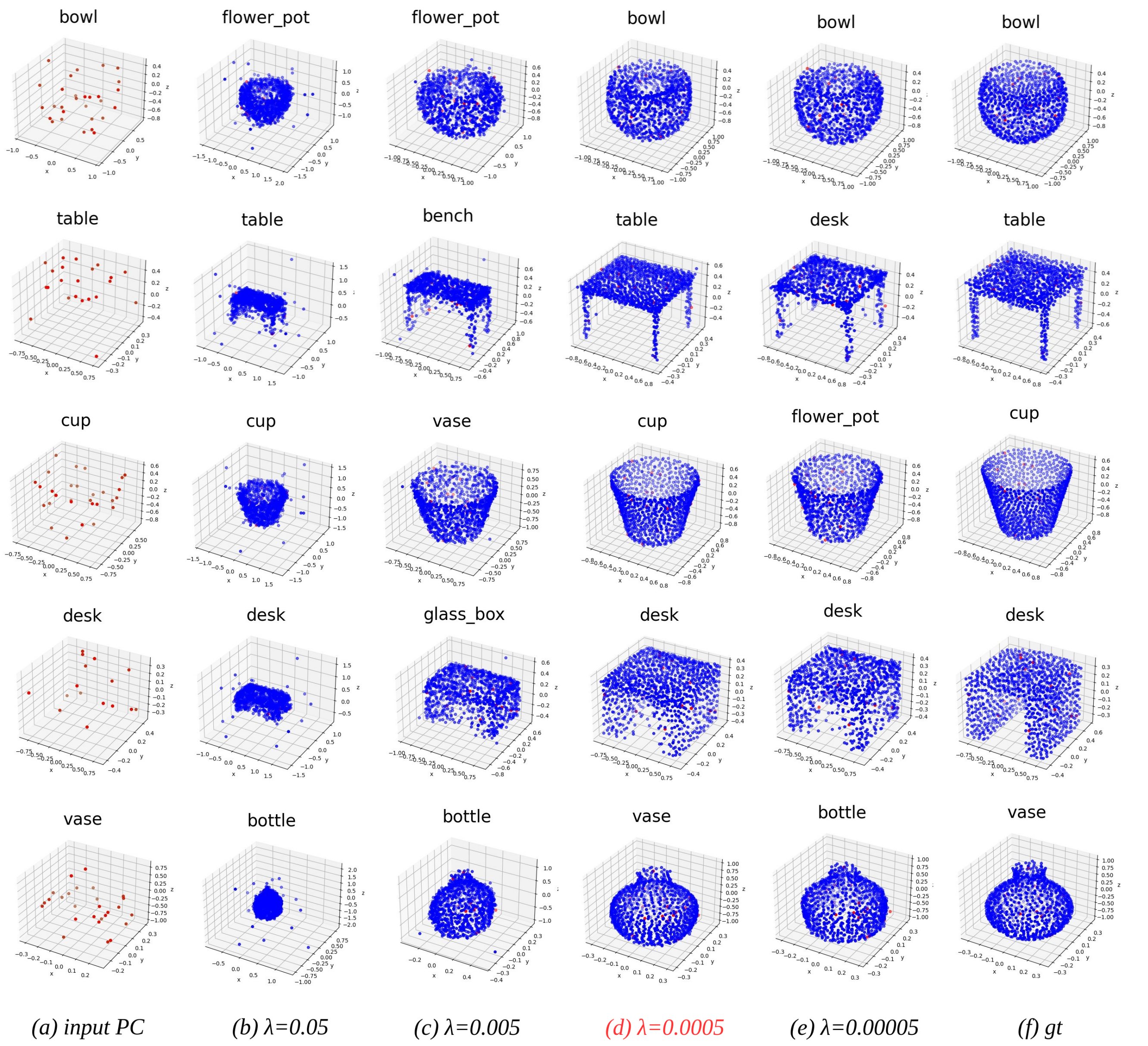


Figure 16. Examples of completed point clouds outputted by PCN when λ varies. PC stands for point cloud and gt stands for ground truth. The word above a point cloud is its label predicted by the base point cloud classifier. The dataset is ModelNet40. The highlighted $\lambda = 0.0005$ is used in our experiments.