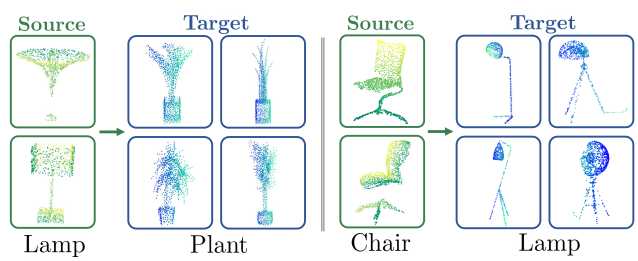
Supplementary Material for PC-Adapter: Topology-Aware Adapter for Efficient Domain Adaption on Point Clouds with Rectified Pseudo-label

A. Additional Experimental Results

A.1. Further Discovery of Remark 3.2

We provide additional examples that Point Transformer [11] misclassifies target domain objects as ground-truth classes of source domain objects in Figure 1. The figure illustrates that the existing encoding architecture fails to perceive the implicit shape of target objects in its entirety, as described in Remark 3.2. Specifically, the encoder exhibits poor prediction ability on target objects due to its *partial* focus on similar outlines (*i.e.* cylinders in lamps and stems in plants) as shown in Figure 1 (a) and the presence of multiple legs in Figure 1 (b).



(a) Target *plant* objects misclassified as *lamp* in ShapeNet to ModelNet domain shift scenario. (b) Target *lamp* objects misclassified as *chair* in ScanNet to ShapeNet domain shift scenario.

Figure 1. **Target** point cloud samples that Point Transformer mispredicts as the ground-truth classes of **source** samples on two domain shift scenarios - (a) ShapeNet→ModelNet and (b) ScanNet→ShapeNet.

A.2. Learning Rate Adjustment for Different Training Paths

As we describe in Section 3.2 of our main paper, we reduce the learning rate by a factor of ρ for the parameters of the shared components (Φ and Ψ_g) to preserve the implicit shape knowledge of the source domain while training on target point clouds. To demonstrate the effectiveness of this strategy, we compare domain adaptation results of ours with and *without* weakly updating parameters of shared parts for the target path (Table 1). The results show that maintaining an identical learning rate for both the source and target training paths significantly degrades the adaptation performance. These results highlight the importance of altering the learning rate on source and target paths, which is effective for *preserving* the source geometry information.

Table 1. Ablation study for learning rate adjustment strategy on two domain shift settings in PointDA-10, averaged over three repetitions (± SEM).

Method	S→M	M→S*
	Acc.	Acc.
PC-Adapter ($\rho = 1.0$)	74.8 ± 0.4	53.6 ± 0.5
PC-Adapter ($\rho = \frac{1}{2.5}$)	76.6 ± 0.3	53.9 ± 0.2
PC-Adapter ($\rho = \frac{1}{5.0}$)	**77.5** ± 0.2	**58.2** ± 0.4

A.3. Qualitative Analysis of Relative Positional Encoding

In Figure 2, we qualitatively analyze target samples to complement the efficacy of the proposed *relative* positional encoding σ. The target objects in the figure are ones that *PC-Adapter* equipped with conventional point cloud positional encoding [11] mispredicts whereas *PC-Adapter* with relative positional encoding correctly classifies.

B. Proof: Parameter Estimation for Beta Distribution

Before proving the parameter estimation for beta distribution, we first introduce the method-of-moments estimation below.

Lemma 1 (Method of Moments). Let $x = \{x_1, \dots, x_n\}$ be a set of independent and identically distributed realizations (samples) from random variable $\boldsymbol{X}$. We define the probability distribution $p(x|\theta)$ parameterized by unknown parameters $\theta = \{\theta_1, \dots, \theta_k\}$. Then the unknown parameters are estimated by matching the moments as follows.

Let us derive first k sample moments $\{\hat{\mu}_i(\boldsymbol{X})\}_{i=1}^{k}$ for constant b_k as

$$\hat{\mu}_1(\boldsymbol{X}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - b_1)^1, \; \dots, \; \hat{\mu}_k(\boldsymbol{X}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - b_k)^k. \tag{1}$$

Then we express the first k moments of $\boldsymbol{X}$ in terms of θ:

$$\mu_1(\boldsymbol{X}) = f_1(\theta_1, \ldots, \theta_k), \; \ldots \; , \mu_k(\boldsymbol{X}) = f_k(\theta_1, \ldots, \theta_k). \tag{2}$$

By solving the following system of k equations,

$$\begin{cases} \hat{\mu}_1(\boldsymbol{X}) = f_1(\theta_1, \ldots, \theta_k) \\ \vdots \\ \hat{\mu}_k(\boldsymbol{X}) = f_k(\theta_1, \ldots, \theta_k), \end{cases} \tag{3}$$

the estimated parameters $\hat{\theta}_1, \ldots, \hat{\theta}_k$ can be derived.

Assume the confidence distribution for each class t given source training dataset $\mathcal{S}_{\text{train}}$ follows a beta distribution, $p(c_t|\mathcal{S}_{\text{train}}) \approx \texttt{Beta}(\hat{\alpha}_t, \hat{\beta}_t)$. For each class t, we compute the sample mean of confidences $\bar{c}_t$ as $\bar{c}_t = \frac{1}{|\mathcal{S}^t_{\text{train}}|}\sum_{i \in \mathcal{S}^t_{\text{train}}} c_{i,t}$, and sample variance $\bar{v}_t$ as $\bar{v}_t = \frac{1}{|\mathcal{S}^t_{\text{train}}|-1}\sum_{i \in \mathcal{S}^t_{\text{train}}} (c_{i,t} - \bar{c}_t)^2$, where $\mathcal{S}^t_{\text{train}}$ denotes the indices of samples belonging to class t, and $c_{i,t}$ is the confidence score for class t on the i-th sample. Then, two unknown parameters - $\hat{\alpha}_t$ and $\hat{\beta}_t$ - are estimated as:

$$\hat{\alpha}_t = \bar{c}_t\Big(\frac{\bar{c}_t(1-\bar{c}_t)}{\bar{v}_t} - 1\Big), \; \hat{\beta}_t = (1-\bar{c}_t)\Big(\frac{\bar{c}_t(1-\bar{c}_t)}{\bar{v}_t} - 1\Big). \tag{4}$$

Proof. Mean and variance of random variable $\boldsymbol{X}$ which follows beta distribution $\texttt{Beta}(\hat{\alpha}_t, \hat{\beta}_t)$ are given by

$$\mathrm{E}(\boldsymbol{X}) = \frac{\hat{\alpha}_t}{\hat{\alpha}_t + \hat{\beta}_t}, \mathrm{Var}(\boldsymbol{X}) = \frac{\hat{\alpha}_t\hat{\beta}_t}{(\hat{\alpha}_t + \hat{\beta}_t)^2(\hat{\alpha}_t + \hat{\beta}_t + 1)}. \tag{5}$$

Using Lemma 1, we acquire the following equations by matching the moments:

$$\bar{c}_t = \frac{\hat{\alpha}_t}{\hat{\alpha}_t + \hat{\beta}_t} \tag{6}$$

$$\bar{v}_t = \frac{\hat{\alpha}_t\hat{\beta}_t}{(\hat{\alpha}_t + \hat{\beta}_t)^2(\hat{\alpha}_t + \hat{\beta}_t + 1)} \tag{7}$$

From the Equation 6,

$$\begin{aligned} \bar{c}_t(\hat{\alpha}_t + \hat{\beta}_t) &= \hat{\alpha}_t \\ \bar{c}_t\hat{\beta}_t &= \hat{\alpha}_t - \bar{c}_t\hat{\alpha}_t \\ \hat{\beta}_t &= \hat{\alpha}_t\Big(\frac{1}{\bar{c}_t} - 1\Big). \end{aligned} \tag{8}$$

Plugging Equation 8 into Equation 7, we have:

$$\begin{aligned} \bar{v}_t &= \frac{\hat{\alpha}_t^2(\frac{1}{\bar{c}_t} - 1)}{(\frac{\hat{\alpha}_t}{\bar{c}_t})^2(\frac{\hat{\alpha}_t}{\bar{c}_t} + 1)} \\ &= \frac{(\frac{1}{\bar{c}_t} - 1)}{(\frac{1}{\bar{c}_t})^2(\frac{\hat{\alpha}_t}{\bar{c}_t} + 1)} \\ &= \frac{\bar{c}_t - \bar{c}_t^2}{\frac{\hat{\alpha}_t}{\bar{c}_t} + 1} \\ \bar{v}_t\Big(\frac{\hat{\alpha}_t}{\bar{c}_t} + 1\Big) &= \bar{c}_t - \bar{c}_t^2 \\ \frac{\hat{\alpha}_t}{\bar{c}_t} + 1 &= \frac{\bar{c}_t - \bar{c}_t^2}{\bar{v}_t} \\ \hat{\alpha}_t &= \bar{c}_t\Big(\frac{\bar{c}_t(1-\bar{c}_t)}{\bar{v}_t} - 1\Big). \end{aligned} \tag{9}$$

We can also obtain $\hat{\beta}_t$ using Equation 8 and Equation 9:

$$\begin{aligned} \hat{\beta}_t &= \hat{\alpha}_t\Big(\frac{1}{\bar{c}_t} - 1\Big) \\ &= \bar{c}_t\Big(\frac{\bar{c}_t(1-\bar{c}_t)}{\bar{v}_t} - 1\Big)\Big(\frac{1}{\bar{c}_t} - 1\Big) \\ &= (1-\bar{c}_t)\Big(\frac{\bar{c}_t(1-\bar{c}_t)}{\bar{v}_t} - 1\Big), \end{aligned} \tag{10}$$

which ends the proof. □

C. Training Algorithm of *PC-Adapter*

We provide detailed algorithm of *PC-Adapter* in Algorithm 1.

D. Detailed Experiment Setup

D.1. Dataset Statistics

PointDA-10 The PointDA-10 [7] dataset comprises objects from 10 shared classes that are collected in three datasets - ModelNet40 (**M**) [10], ShapeNet (**S**) [2], and ScanNet (**S***) [3]. As provided in Table 2, the ModelNet-10 contains 4,183 training and 856 test point clouds, while ShapeNet-10 consists of 17,378 training and 2,492 test samples. Point clouds of both datasets are acquired by sampling 2,048 points from the surface of 3D CAD models (*i.e.* synthetic datasets). Compared to these datasets, ScanNet-10 includes 6,110 training and 1,769 point clouds with 2048 points each, which are scanned and reconstructed from real-world scenes. The point clouds in ScanNet-10 usually exhibit partial views of objects due to the occlusion (by adjacent objects or self-occlusion) and sensor noises.

We carefully follow the data preparation and data split procedures adopted in previous studies [7, 1, 12, 8]. The object point clouds from all datasets are oriented to align with

Algorithm 1 Overall training procedure of *PC-Adapter*.

Input: Source data $\mathcal{S} = \{(\boldsymbol{X}_k^{\text{src}} = \{\mathbf{p}_i\}_{i=1}^m, y_k^{\text{src}})\}_{k=1}^{n_s}$, target data $\mathcal{T} = \{(\boldsymbol{X}_k^{\text{trgt}}\} = \{\mathbf{p}'_i\}_{i=1}^m)\}_{k=1}^{n_t}$, feature encoder Φ, shape-aware adapter Ψ_g, locality-aware adapter Ψ_l, classifier f, correction intensity r_0.

1: **for** $e = 1 \dots E$ **do**
2: **for** $(\{\mathbf{p}_i\}_{i=1}^m, y_k^{\text{src}}), \{\mathbf{p}'_i\}_{i=1}^m$ in $(\mathcal{S}, \mathcal{T})$ **do**
3: Obtain source encoder output $\{\Phi(\mathbf{p}_i)\}_{i=1}^m$
4: Sample farthest points $\tilde{\boldsymbol{X}}_k^{\text{src}} = \{\mathbf{p}_i\}_{i=1}^{m'}$
5: $\{\Psi_g(\mathbf{p}_i)\}_{i=1}^{m'} \leftarrow \sum_{\mathbf{p}_j \in \tilde{\boldsymbol{X}}_k^{\text{src}} \setminus \mathbf{p}_i} w_{ij}(\varphi(\Phi(\mathbf{p}_j)) + \sigma_{ij})$ ▷ Encoding by shape-aware adapter
6: Train f on $\texttt{Combine}\big(\{\Psi_g(\mathbf{p}_i)\}_{i=1}^{m'}, \{\Phi(\mathbf{p}_i)\}_{i=1}^m\big)$ with label y_k^{src}
7:
8: **for** $t = 1 \dots c$ **do**
9: $\bar{c}_t \leftarrow \frac{1}{|\mathcal{S}_{\text{train}}^t|} \sum_{i \in \mathcal{S}_{\text{train}}^t} c_{i,t}, \;\; \bar{v}_t \leftarrow \frac{1}{|\mathcal{S}_{\text{train}}^t|-1} \sum_{i \in \mathcal{S}_{\text{train}}^t} (c_{i,t} - \bar{c}_t)^2$
10: $\hat{\alpha}_t \leftarrow \bar{c}_t \big(\frac{\bar{c}_t(1-\bar{c}_t)}{\bar{v}_t} - 1\big)$
11: $\hat{\beta}_t \leftarrow (1 - \bar{c}_t)\big(\frac{\bar{c}_t(1-\bar{c}_t)}{\bar{v}_t} - 1\big)$
12: Compute r_i from percentile point function of $\texttt{Beta}(\hat{\alpha}_t, \hat{\beta}_t)$
13: $\tilde{c}_{i,t} \leftarrow c_{i,t} \cdot \big(\frac{1}{1-r_i+r_0}\big)$ ▷ Adjusting confidence score for each class
14: **end for**
15: $\hat{y}_k^{\text{trgt}} \leftarrow \underset{t}{\operatorname{argmax}}\, \tilde{c}_{i,t}$
16:
17: Repeat line 3 - 5 on $\{(\mathbf{p}'_i)\}_{i=1}^m$
18: $\{\Psi_l(\mathbf{p}'_i)\}_{i=1}^{m'} \leftarrow \mathbf{\Theta}^\top \sum_{\mathbf{p}_j \in \mathcal{N}(i) \cup \{\mathbf{p}_i\}} \frac{e_{j,i}}{\texttt{deg}(\mathbf{p}_j)\texttt{deg}(\mathbf{p}_i)} \Phi(\mathbf{p}_j)$ ▷ Encoding by locality-aware adapter
19: Train f on $\texttt{Combine}\big(\{\Psi_g(\mathbf{p}_i)\}_{i=1}^{m'}, \{\Psi_l(\mathbf{p}_i)\}_{i=1}^{m'}, \{\Phi(\mathbf{p}_i)\}_{i=1}^{m'}\big)$ with label $\hat{y}_k^{\text{trgt}}$
20: **end for**
21: **end for**

Table 2. Data statistics. The number of samples for each class in PointDA-10.

Dataset	**Domain**		Bathtub	Bed	Bookshelf	Cabinet	Chair	Lamp	Monitor	Plant	Sofa	Table	**Total**
ModelNet-10	Synthetic	Train	106	515	572	200	889	124	465	240	680	392	4,183
		Test	50	100	100	86	100	20	100	100	100	100	856
ShapeNet-10	Synthetic	Train	599	167	310	1,076	4,612	1,620	762	158	2,198	5,876	17,378
		Test	85	23	50	126	662	232	112	30	330	842	2,492
ScanNet-10	Real	Train	98	329	464	650	2,578	161	210	88	495	1,037	6,110
		Test	26	85	146	149	801	41	61	25	134	301	1,769

the direction of gravity, while arbitrary rotations along the z-axis are tolerated. Then, point clouds are normalized to fit within a unit cube. Input point clouds with 2,048 points are down-sampled to 1,024 points.

GraspNetPC-10 The point clouds in GraspNetPC-10 [8, 5] are collected through raw depth scans on both real-world and synthetic scenes using two different cameras, Kinect2 and Intel Realsense. Specifically, the raw depth scans are projected to 3D space and object segmentation masks are applied to extract the object point clouds from the scenes. Unlike samples in PointDA-10, point clouds of this dataset are not aligned along with the vertical direction. For real scenes, the Kinect domain includes 10,973 training and 2,560 test samples, while the Realsense domain contains 10,698 training and 2,560 test point clouds. For synthetic scenes, the Kinect domain has 10,887 training samples, and the Realsense domain contains 10,542 training point clouds. Detailed statistics are provided in Table 3. Point clouds captured by the two different cameras are often affected by different types of noises, and different degrees of structural distortions. We follow the data preparation procedure of [8].

PointSegDA The PointSegDA dataset [1] originates from a 3D mesh-structured human model dataset [6], featuring four distinct subsets. These subsets encompass a total of eight human body part classes, exhibiting variations in terms of point distribution, pose, and scanned individuals. The process involves generating point cloud data by sam-

Table 3. Data statistics. The number of samples for each class in GraspNetPC-10. Syn. and Real denote synthetic scenes and real scenes, respectively.

Domain		$\mathbf{L}_0$	$\mathbf{L}_1$	$\mathbf{L}_2$	$\mathbf{L}_3$	$\mathbf{L}_4$	$\mathbf{L}_5$	$\mathbf{L}_6$	$\mathbf{L}_7$	$\mathbf{L}_8$	$\mathbf{L}_9$	**Total**
Kinect (Syn.)	Train	1,024	1,280	1,273	1,024	1,020	1,278	934	1,006	1,024	1,024	10,887
Realsense (Syn.)	Train	972	1,280	1,280	1,024	1,024	1,280	895	792	980	1,015	10,542
Kinect (Real)	Train	1,024	1,280	1,273	1,024	1,015	1,272	1,019	1,024	1,018	1,024	10,973
	Test	256	256	256	256	256	256	256	256	256	256	2,560
Realsense (Real)	Train	968	1,280	1,280	1,024	1,020	1,279	1,020	841	971	1,015	10,698
	Test	256	256	256	256	256	256	256	256	256	256	2,560

pling 2048 points from the 3D mesh data. Subsequently, the sampled points are aligned along the z-axis and normalized to fit within a unit cube. Corresponding point labels are assigned based on polygon labels. Concise data statistics are presented in Table 4. We adhere to the data preprocessing and data split rules proposed in [1].

Table 4. Data statistics. The number of samples for each subset in PointSegDA.

Domain	Train	Validation	Test	**Total**
FAUST	70	10	20	100
MIT	118	17	34	169
ADOBE	29	4	8	41
SCAPE	50	7	14	71

D.2. Implementation Details

In this subsection, we describe the implementation details and hyperparameters of our method. For our experiments, we set the farthest point sampling (FPS) ratio in to 0.1 for adapter modules, and the number of nearest neighbors k to 5 for k-NN graph in locality-aware adapter Ψ_l. For our distribution-guided pseudo labeling, we tune the correction intensity, r_0, from the $\{0.1, 10, 15, 20, 30, 40, 45\}$. As in previous works [4, 12], we employ the (fixed) threshold, γ, to filter out noisy pseudo labels that have low confidence scores. We search optimal γ among the range $[0.7, 0.92]$. Since our correction strategy modifies the confidences by a ratio from $\frac{1}{r_0+1}$ to $\frac{1}{r_0}$, we multiply the average scale of modification to threshold γ, $\gamma * \frac{1}{2}(\frac{1}{r_0+1} + \frac{1}{r_0})$, to account for the altered scale of confidence scores. The loss coefficient for the regularization loss $\mathcal{L}_{\text{centroid}}$ is tuned from the set $\{$1, 0.1, 0.001$\}$. In part segmentation experiments, we do not use distribution-guided pseudo labels and select a threshold γ from the options $\{$0.98, 0.99$\}$.

D.3. Evaluation Protocol

In our experiments, we use DGCNN [9] as the architecture for the feature encoder Φ, in line with previous works [4, 8]. We follow the evaluation protocol outlined in [4, 1] for the PointDA-10 dataset and that in [8] for the GraspNetPC-10 dataset. For point cloud classification experiments, we adopt an Adam optimizer with an initial learning rate 0.001, and weight decay is set to 0.00005. The cosine annealing is employed for the learning rate scheduler and the learning rate weakening factor, ρ, is set to 0.2 for target domain training, except for the ShapeNet to ScanNet setting, where it is set to 0.01. We train the models for 150 epochs on PointDA-10 and 120 epochs on GraspNetPC-10, and the best model is selected using source validation accuracy. For the PointSegDA dataset, we follow the evaluation settings of [1]. PCM data augmentation is employed in scenarios where it is utilized in [1]. In part segmentation experiments, we search the learning rate decay factor ρ within the set $\{1, \frac{1}{3}, \frac{1}{5}\}$.

References

[1] Idan Achituve, Haggai Maron, and Gal Chechik. Self-supervised learning for domain adaptation on point clouds. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 123–133, 2021. 2, 3, 4

[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. 2

[3] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. 2

[4] Hehe Fan, Xiaojun Chang, Wanyue Zhang, Yi Cheng, Ying Sun, and Mohan Kankanhalli. Self-supervised global-local structure modeling for point cloud domain adaptation with reliable voted pseudo labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6377–6386, 2022. 4

[5] Hao-Shu Fang, Chenxi Wang, Minghao Gou, and Cewu Lu. Graspnet-1billion: A large-scale benchmark for general object grasping. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11444–11453, 2020. 3

[6] Haggai Maron, Meirav Galun, Noam Aigerman, Miri Trope, Nadav Dym, Ersin Yumer, Vladimir G Kim, and Yaron Lipman. Convolutional neural networks on surfaces via seamless toric covers. *ACM Trans. Graph.*, 36(4):71–1, 2017. 3

[7] Can Qin, Haoxuan You, Lichen Wang, C-C Jay Kuo, and Yun Fu. Pointdan: A multi-scale 3d domain adaption network for point cloud representation. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[8] Yuefan Shen, Yanchao Yang, Mi Yan, He Wang, Youyi Zheng, and Leonidas J Guibas. Domain adaptation on point clouds via geometry-aware implicits. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7223–7232, 2022. 2, 3, 4

[9] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic

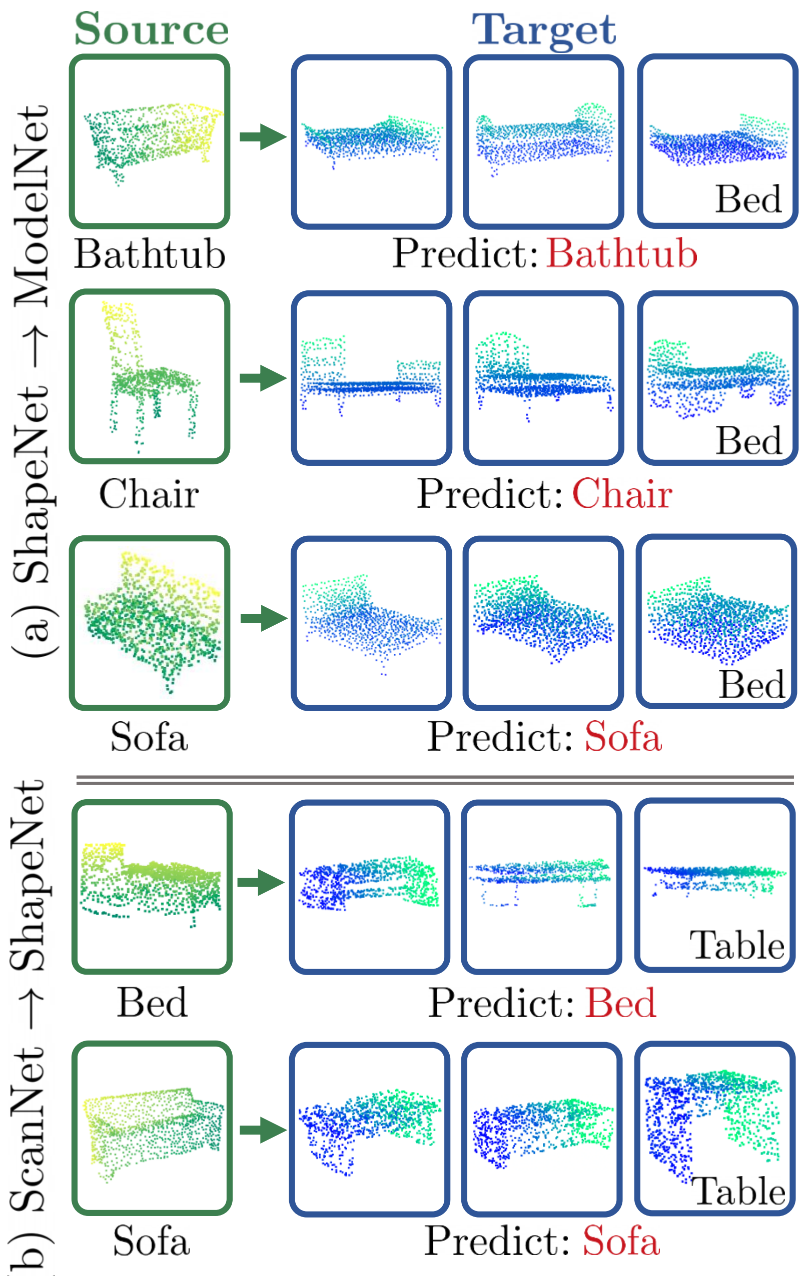


Figure 2. **Target** point cloud samples that proposed relative positional encoding *correctly* predicts while traditional point cloud positional encoding misclassifies as the ground-truth classes of **source** samples on two domain shift scenarios - (a) target *beds* mispredicted as bathtub, chair, and sofa, (b) target *tables* mispredicted as bed and sofa of the source. The results indicate that the major drawbacks of existing encoding architectures discussed in Section 3.2 and A.1 could be effectively mitigated by our proposed *relative* positional encoding.

graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 4

[10] 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. 2

[11] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16259–16268, 2021. 1

[12] Longkun Zou, Hui Tang, Ke Chen, and Kui Jia. Geometry-aware self-training for unsupervised domain adaptation on object point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6403–6412, 2021. 2, 4