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

Joonhyung Park
KAIST
deepjoon@kaist.ac.kr

Hyunjin Seo
KAIST
bella72@kaist.ac.kr

Eunho Yang
KAIST, AITRICS
eunhoy@kaist.ac.kr

Abstract

*Understanding point clouds captured from the real-world is challenging due to shifts in data distribution caused by varying object scales, sensor angles, and self-occlusion. Prior works have addressed this issue by combining recent learning principles such as self-supervised learning, self-training, and adversarial training, which leads to significant computational overhead. Toward succinct yet powerful domain adaptation for point clouds, we revisit the unique challenges of point cloud data under domain shift scenarios and discover the importance of the global geometry of source data and trends of target pseudo-labels biased to the source label distribution. Motivated by our observations, we propose an adapter-guided domain adaptation method, **PC-Adapter**, that preserves the global shape information of the source domain using an attention-based adapter, while learning the local characteristics of the target domain via another adapter equipped with graph convolution. Additionally, we propose a novel pseudo-labeling strategy resilient to the classifier bias by adjusting confidence scores using their class-wise confidence distributions to consider relative confidences. Our method demonstrates superiority over baselines on various domain shift settings in benchmark datasets - PointDA, GraspNetPC, and PointSegDA.*

1. Introduction

3D vision has gained considerable attention as an immense amount of 3D data is collected by LiDAR sensors or depth-sensing cameras in real-world applications such as autonomous vehicles, surveillance systems, and drones. In recent years, deep neural networks [15, 16, 22, 28, 17] have become a de facto approach for understanding unordered 3D point sets (*i.e.* point cloud data). However, for the case of point clouds in the wild, deep models inevitably encounter *distributional shift* - also known as *domain shift* - due to the different object scales, point density, angle of

sensor, object occlusion, *etc*. This problem makes the models perform poorly outside the laboratory.

Unsupervised domain adaptation (UDA) is a representative solution to tackle the domain shift problem, which targets to transfer the knowledge from a labeled source domain to an unlabeled target domain [24, 8, 10]. UDA methods generally endeavor to reduce the discrepancy of two domains in representation space, so that a classifier trained on the label-rich source domain generalizes well on the target domain. Such standard UDA strategies would be attractive options even for point cloud data, since collecting synthetic point clouds can be simply done by sampling points from the 3D CAD models whereas annotating the enormous real 3D data is cumbersome and time-consuming. Hence, if a UDA method can successfully transfer the knowledge from synthetic point clouds (source domain) to real point clouds (target domain), it will enable the models to significantly boost performances over a broad range of 3D vision tasks.

Therefore, numerous studies considering the innate characteristic of point cloud data have recently been proposed for domain adaptation [18, 1, 29, 5, 20, 12, 19]. Most of them *jointly* exploit components such as self-supervised learning, adversarial training, data augmentation, and self-training to squeeze the best adaptation performances. Especially, recent works [29, 5, 12] focus on designing at least two self-supervised learning tasks by deforming point clouds; for example, reconstructing the deformed parts or predicting the extent of specific distortion. Although these methods have shown successful results, due to their excessive computations and memory access, these frameworks are difficult to deploy in real-world systems where we do not have sufficient computing resources. These situations raise questions regarding the race of performance-oriented design and call for a pivot towards a succinct yet effective strategy toward a practical domain adaptation framework.

Toward this, we reconsider the individual efficacy of different types of knowledge for domain adaptation, especially focusing on which knowledge of the source benefits to be transferred, and suggest two design philosophies for a point cloud-specific domain adaptation framework. We first com-

pare the value of knowledge in terms of global geometry and local structure to determine which knowledge would be more beneficial under domain shift scenarios. Drawing on our observations, we argue the necessity of transferring the shape information of the source domain during the adaptation phase. Secondly, we observe that pseudo-labels are highly dependent on the label distribution of the source. To obtain unbiased pseudo-labels for target data, we suggest that the label bias induced from the knowledge of source label distribution should be considered in selecting pseudo-labels as there is no guarantee that the label distribution of the source domain would match that of the target domain.

Inspired by the prior insights, we introduce **P**oint **C**loud Adapter (*PC-Adapter*), an adapter-based domain adaptation framework that can efficiently learn feature transformation to adapt local spatial structures of the target domain while preserving the knowledge of global geometry acquired from the source domain. Our method exploits two adapters, a global shape-aware adapter and a locality-aware adapter, both of which consist of one simple block. Our global shape-aware adapter learns implicit shape information of the source domain using a proposed relative positional encoding and then transfers this knowledge to target point clouds. To preserve the source geometry information during the adaptation process, the parameters are weakly updated when training target point clouds. Meanwhile, the locality-aware adapter actively learns the target domain-specific local structures using graph neural network operations, which is *only* updated by target point clouds. In other words, the local structure adapter solely provides room for feature transformation to adapt to the target domain. After passing the entire path of the model including adapters, target point clouds are trained via pseudo-labels. To avoid pseudo-labels being selected relying on the label distribution of the source domain, we propose a simple yet novel pseudo-label correction strategy. Our approach rectifies pseudo-labels based on their percentile ranks from approximated class-wise confidence distributions, taking into account their relative confidences.

In summary, our contribution is threefold:

- We present design philosophies to promote succinct yet effective domain adaptation on point cloud data, concentrated on which knowledge from the source domain has to be transferred.

- We propose a tailored framework for these design principles that preserves global shape knowledge of the source via an attention-based adapter while providing room for target-specific feature transformation through a graph convolution-based adapter. We also devise a reliable pseudo-labeling method that adjusts class-wise confidence with the guidance of its percentile on confidence distribution. Note that our method does not resort to any extra sub-tasks or adversarial training.

- Extensive experiments show that our *PC-Adapter* brings significant performance gain over self-training and adversarial domain alignment baselines, and even achieves state-of-the-art performances when combined with auxiliary self-supervised learning tasks.

2. Related Work

2.1. Domain Adaptation on Point Clouds

UDA has been recently investigated on 3D point clouds to bridge the gap between domains of different natures. PointDAN [18] proposed a point cloud-specialized UDA framework via multi-scale feature alignment. DefRec [1] improved the performance by leveraging a self-supervised learning task of reconstructing deformed object regions along with a mixup strategy. GAST [29] learned a domain-shared point cloud representation by integrating geometry-aware auxiliary tasks of angle/location prediction, and further encouraged the training through the pseudo-labeling process. GLRV [5] devised different self-supervised learning methods to learn global and local point cloud structures in-depth combined with adversarial training and voting-based pseudo-labeling strategy. Shen et al. [20] learned unsigned distance fields of point clouds in a self-supervised manner to enhance the understanding of implicit shape geometry shared across domains. MLSP [12] incorporated prediction tasks of estimating local attributes of masked regions to alleviate domain bias for target point clouds. However, most of these works generally leverage cumbersome tasks that are infeasible in real-world systems. Different from existing studies, we propose an efficient domain adaptation framework for point clouds without requiring additional tasks with excessive computation.

2.2. Adapter

Recently, adapter modules have been proposed as a parameter-efficient method to fine-tune large-scale pre-trained models for downstream tasks [9, 7, 21, 3]. The adapter modules are inserted between layers of a pretrained network, which consist of only a few trainable parameters (*e.g.* two linear layers with nonlinear activation). This adapter-based transfer has shown successful performances on various downstream tasks by enabling the adapter module to learn the task-specific feature transformations. Inspired by the success of adapter-tuning, we exploit the concept of adapter to address domain shift problems in point cloud data. Compared to an adapter in the transfer learning field, our proposed adapter is specially designed to learn the structural information of 3D objects.

Original Input | Farthest Point Sampled Input (25%) | Original Input | Farthest Point Sampled Input (25%)
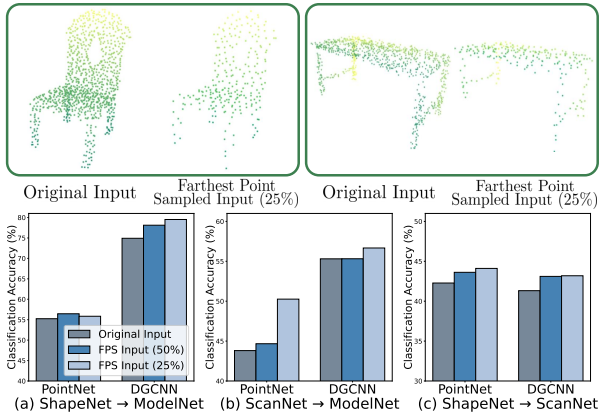

Figure 1. Classification accuracy of varying input point ratios on two point cloud encoding architectures. We gradually reduce the number of given point cloud points by 50% and 25% using FPS, preserving only the global anatomy of inputs. Domain adaptations solely based on global structures show superior performance against the original input in multiple domain shift scenarios.

## 3. Design Philosophies for Domain Adaptation on Point Cloud

Before devising the domain adaptation method, we would like to suggest key philosophies that need to be considered for domain adaptation on point clouds. Especially, we mainly focus on which knowledge from the source domain has to be *preserved* and which information should be *transformed* adaptively towards the target domain.

3.1. Problem Formulation and Notation

In this work, we tackle unsupervised domain adaptation (UDA) for 3D point cloud data. Let $\boldsymbol{X}_k \in \mathbb{R}^{m \times 3}$ be the point cloud data represented as an unordered set of 3D coordinates $\boldsymbol{X}_k = \{\mathbf{p}_1, \ldots \mathbf{p}_m\}$, where m is the number of points and $\mathbf{p}_i \in \mathbb{R}^3$ is a coordinate of i-th point. The objective of UDA is to transfer the knowledge from the annotated source domain $\mathcal{S} = \{(\boldsymbol{X}_k^{\mathrm{src}}, y_k^{\mathrm{src}})\}_{k=1}^{n_s}$ to unlabeled target domain $\mathcal{T} = \{\boldsymbol{X}_k^{\mathrm{trgt}}\}_{k=1}^{n_t}$ where n_s and n_t denote the number of point clouds in the source and target domains, respectively. In UDA settings, two domains are assumed to have different data distributions $\mathcal{D}_{\mathrm{source}}$ and $\mathcal{D}_{\mathrm{target}}$ but with a shared label space $\mathcal{Y} = \{1, \ldots, c\}$. The core concept of the UDA is to learn a model that projects inputs from different domains onto a shared representation space by reducing the discrepancy of representations across domains.

3.2. Efficacy of Global Geometry Knowledge

We begin our discussion by showing the importance of leveraging global geometry knowledge where local spatial structures are usually discarded in domain-shift scenarios. Given the causes of distributional shifts (*e.g.*
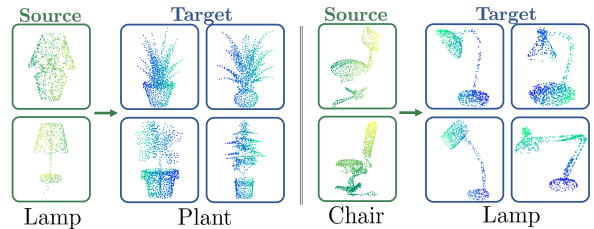
(a) Target *plant* objects misclassified as *lamp* in ShapeNet to ModelNet domain shift scenario. (b) Target *lamp* objects misclassified as *chair* in ScanNet to ShapeNet domain shift scenario.

Figure 2. **Target** point cloud samples that Point Transformer mispredicts as the ground-truth classes of **source** samples on two domain shift scenarios (a) and (b). Regarding (a), the encoder perceives the vague outlines of target plants that resemble those of the source lamps, thereby mispredicting the target objects without comprehending the precise structure (leaves, for instance). A similar tendency is observed in (b), where round seats and curved outlines of source chairs are analogous to target lamps, resulting in the encoder misclassifying target lamps as chairs.

self-occlusion, sensor noise, point density), there generally exists a large characteristic discrepancy appearing locally across domains. Therefore, it is important to *selectively* transfer the pertinent knowledge to the distribution-shifted domain rather than ignorantly transferring all types of knowledge from the source domain. To identify this 'versatile' knowledge in domain shift scenarios, we compare the efficacy of global geometry information with that of local spatial information on domain-shifted point clouds.

Toward this, we design an experiment to train models with data that only contains information about global architecture by eliminating the local attributes of point clouds using Farthest Point Sampling (FPS). Two representative models - PointNet [15] and DGCNN [26] - are trained with farthest point sampled data from labeled source datasets and tested on the unlabeled target dataset without applying any domain adaptation methods. In Figure 1, we can somehow surprisingly confirm that results with eliminating some points (lightening the knowledge by eliminating local properties) show superior, at least comparable, object classification accuracy compared to the original input under multiple domain shifts. These results imply that the knowledge of global geometry remains helpful when the domain shifts, whereas the local characteristics of objects do not.

Then the following question naturally arises: *How to transfer the global anatomy information from the source domain to the target domain?* A naive approach for this distillation is leveraging a shared encoder and training it on both source and target domains. However, as existing point cloud encoding architectures [15, 16, 26, 28, 13] are not specifically designed for capturing the global anatomy of the objects, they could often fail to identify this shape information. To confirm our claim, we select the Point Transformer [28], one of the representative point cloud encoding architectures, as a shared encoder and test this model under
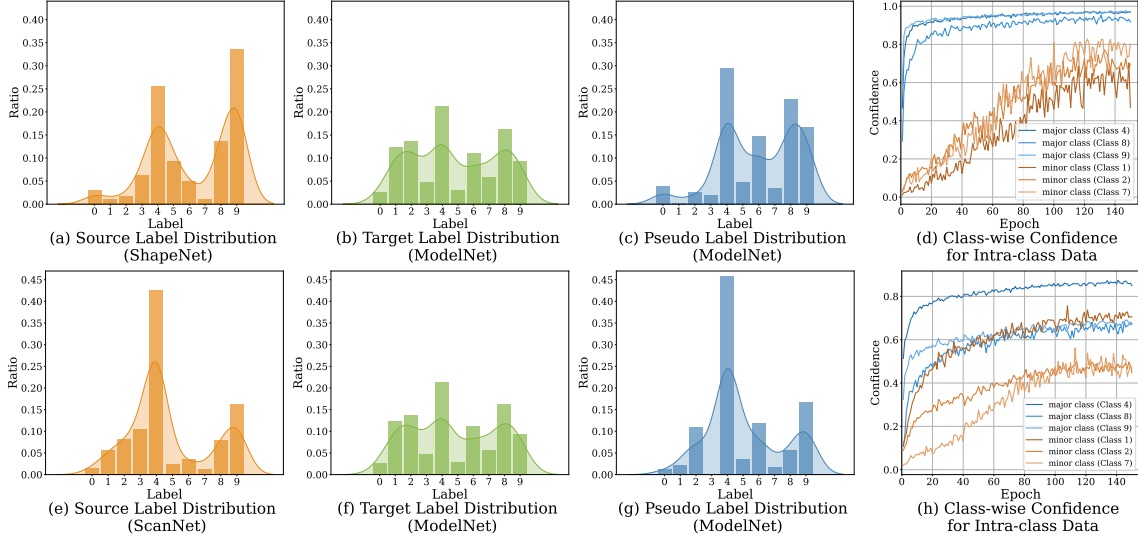
Figure 3. Label distributions (a-c and e-h) and class-wise confidence trends on source training data (d and h). The first row corresponds to ShapeNet to ModelNet domain shift scenario, whereas the below indicates the domain shift of ScanNet to ModelNet. According to the above figures, target pseudo-label distributions (c and g) tends to follow the source label distributions where source objects of minority classes show low confidence, injecting bias into the shared classifier and thereby rarely yielding pseudo-labels for the very classes.

two domain shift settings (ShapeNet-10→ModelNet-10 and ScanNet-10→ShapeNet-10). Then we visualize representative target objects that the encoder misclassifies in Figure 2. According to the figure, we discover that Point Transformer tends to predict target objects depending solely on the *specific* part, without full consideration. From these observations, we could claim the following remark as the design principle of the domain adaptation framework:

Remark 3.1. *Knowledge of global geometry is potentially conducive to domains with distributional shifts, whereas local features of point clouds need to be thoroughly adapted toward the target domain. To transfer the geometry information across domains, a sophisticated encoding module capturing objects in the entirety is necessary for adaptation.*

3.3. Analysis of Pseudo-label Distribution

Regarding another key ingredient of UDA for point clouds, pseudo-labeling, we investigate how the model trained on source labels affects the distribution of pseudo-labels obtained for target point clouds. Most recent domain adaptation works on point cloud data including [5, 29, 12, 20] involve a *self-training* technique, which utilizes pseudo-labels of target point clouds to train the model. Pseudo-labels are typically determined by the prediction scores of the classifier and are adopted only when the maximum confidence score is above a certain threshold γ. However, this strategy has a risk that the classifier could recklessly predict target point clouds as major classes of the source domain. To verify this, we explore pseudo-label distributions target training point clouds (Figure 3). DGCNN models are

trained under two domain shift settings (ShapeNet → ModelNet and ScanNet → ModelNet) using basic self-training where pseudo-labels for target data are determined by the classes of maximum confidence scores given a fixed threshold ($\gamma = 0.9$). As shown in Figure 3 (c) and (g), pseudo-labels indeed follow the (ground-truth) label distribution of the source training data (Figure 3 (a) and (e)), and pseudo-labels are rarely selected for the minority classes at the source domain. Imposing such an inductive bias of source label distribution onto the arbitrary target domains is problematic as they could have far different label distributions.

While previous works have attempted to generate accurate pseudo-labels on domain-shifted data, these methods are insufficient to alleviate this problem as they merely focus on the reliability of pseudo-labels. Specifically, [29] increasingly selects pseudo-labels of target samples with low confidences (*i.e.* harder samples) to train a model in an easy-to-hard learning manner, and [5] uses voting to select pseudo-labels based on the labels of nearest neighbors in the representation space to ensure the consistency of pseudo-labels. Although these methods can filter out noisy pseudo-labels to some extent by focusing on the *magnitude* of maximum, they are not enough to prevent pseudo-labels from following the source label distribution as they cannot change the *order of prediction scores* between classes. The reason for modifying the order of prediction scores can be found in our following observation. In Figure 3 (d) and (h), we measure the class-wise confidence on the source training data, computed by averaging the confidences on intra-class data. Despite confidences only for intra-class samples, we confirm that the model predicts the samples from the mi-
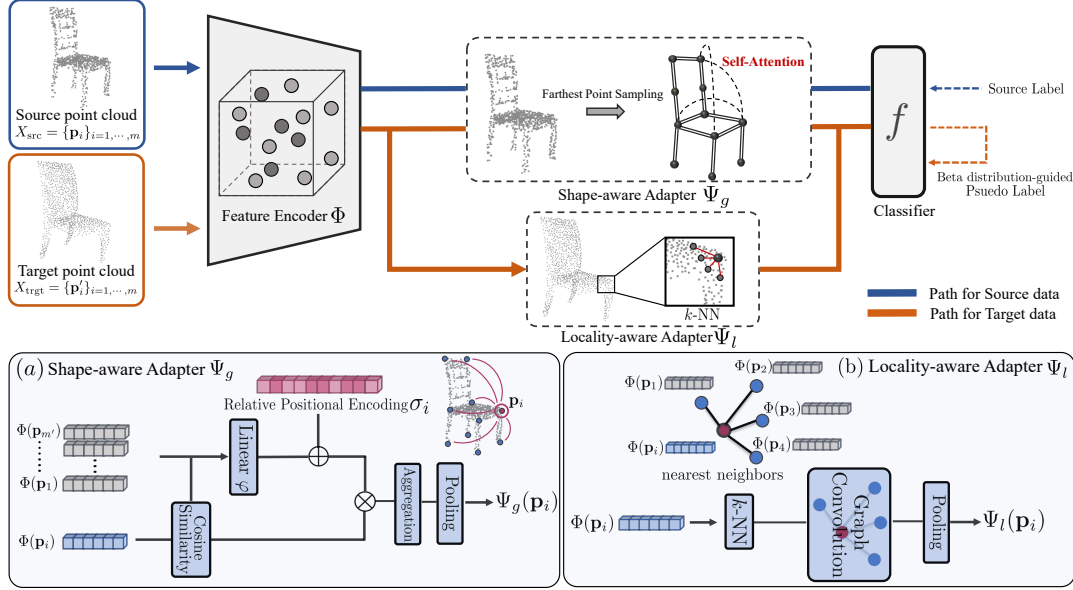
Figure 4. Illustration of proposed framework *PC-Adapter*. Our framework provides distinct training paths for different domains to conduct effective domain adaptation. To begin with, the *PC-Adapter* feeds X_{src} to shared Φ and learns the global anatomy of the source domain from Ψ_g that utilizes relative positional encoding σ with farthest sampled points and $\Phi(X_{\text{src}})$. Meanwhile, the pathway for X_{trgt} includes all key components Φ, Ψ_g, and Ψ_l. Following Φ, the proposed framework perceives global structures of X_{trgt} from Ψ_g trained by the source domain. In conjunction with global knowledge adjustment, our *PC-Adapter* learns local properties of X_{trgt} via Ψ_l to compensate for the characteristic discrepancy resulting from the domain shift. Finally, the adapter outputs are combined for each domain and transmitted to the shared classifier f. In addition, *PC-Adapter* leverages unbiased pseudo-labels for target point clouds with the guidance of beta distribution.

nor classes with low confidences during the training phase though they actually belong to corresponding minor classes. Hence, before generating pseudo-labels, rectifying the confidences considering the classifier bias is highly required to obtain unbiased pseudo-labels. Our second design principle for domain adaptation on point cloud data is as follows:

Remark 3.2. *Pseudo-labels for target point clouds are readily biased toward the source label distribution. As classifier bias leads to varying scales of intra-class prediction probabilities for major/minor class samples, this phenomenon should be reflected in pseudo-label generation.*

4. Proposed Method: PC-Adapter

Building upon the design principles outlined in Section 3, we introduce *PC-Adapter*, an effective domain adaptation framework for 3D point clouds. Our method is composed of two key components. (1) To *selectively* distill the object information from the source emphasized in Remark 3.1, we employ two adapters (Ψ_g and Ψ_l) inserted between the encoder Φ and classifier f (Section 4.1). The intuition behind leveraging the adapter module for domain adaptation is that it can serve as a mediator, facilitating the transfer of versatile shape knowledge across domains via shape-aware adapter Ψ_g while intensifying the adaptation to local characteristics of the target domain through target-exclusive locality-aware adapter Ψ_l. (2) Our *PC-Adapter*

generates qualified pseudo-labels for target point clouds that are resistant to classifier bias, as motivated in Remark 3.2. Prior to pseudo-label selection, our method adjusts the prediction score in advance by referring to the relative percentile on its confidence distribution (Section 4.2). The overall pipeline of *PC-Adapter* is presented in Figure 4 and the training algorithm in Appendix C. We describe the details of each component in the following subsections.

4.1. Adapter-based Domain Adaptation

Shape-aware adapter We now present the shape-aware adapter Ψ_g to effectively encode the knowledge of global structure implied in both the source and target domains. To identify the shape of objects, we design our adapter to compute self-attention [25] between the farthest points of an object (Figure 4). For a given 3D point cloud X_k, a set of farthest points $\tilde{X}_k = \{\mathbf{p}_i\}_{i=1}^{m'}$ are sampled from X_k using FPS algorithm. Then self-attention operations are conducted among the farthest points following attention weights w_{ij}, computed as cosine similarity between outputs of feature encoder Φ:

$$\Psi_g(\mathbf{p}_i) = \sum_{\mathbf{p}_j \in \tilde{X}_k \setminus \mathbf{p}_i} w_{ij}(\varphi(\Phi(\mathbf{p}_j)) + \sigma_{ij}),$$

$$w_{ij} = \frac{\Phi(\mathbf{p}_i) \cdot \Phi(\mathbf{p}_j)}{\|\Phi(\mathbf{p}_i)\|\|\Phi(\mathbf{p}_j)\|},$$

(1)

where φ denotes a linear projection layer, and σ_{ij} represents a position encoding. Note here that the shape-aware adapter Ψ_g consists of only *one* self-attention layer.

As we have argued in Remark 3.2, properly capturing shape information of the source and conveying this knowledge to the target domain facilities the interpretation of domain-shifted objects. To achieve this, we propose a novel relative positional encoding that identifies the global structure by considering both the relative point location and the ratio information of the objects. Our positional encoding utilizes the relative distances from $\mathbf{p}_i$ to the remaining FPS points guided by the maximum distance concerning $\mathbf{p}_i$, $d_i^* = \max_{\mathbf{p}_j \in \tilde{X}_k}(\|\mathbf{p}_i - \mathbf{p_j}\|)$. Specifically, for a given point $\mathbf{p}_i \in \tilde{X}_k$, the relative distance $\hat{d}_{ij}$ with another FPS point $\mathbf{p}_j$ is derived by subtracting the maximum distance in terms of $\mathbf{p}_i$, denoted as d_i^*, to the euclidean distance between $\mathbf{p}_i$ and $\mathbf{p}_j$, $d_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\|$. Then, we obtain positional encoding σ_{ij} by normalizing $\hat{d}_{ij}$ over other remaining relative distances $\{\hat{d}_{in}|\mathbf{p}_n \in \tilde{X}_k\}$ to sum to 1 as follows:

$$\sigma_{ij} = \frac{\hat{d}_{ij}}{\sum_{n=1}^{m'} \hat{d}_{in}} = \frac{d_i^* - d_{ij}}{\sum_{n=1}^{m'}(d_i^* - d_{in})}. \quad (2)$$

The above formulation allows us to yield $\mathbf{p}_i$-specific positional encoding values. As a result, our relative positional encoding approach using d_i^* contemplates the object structure from diverse perspectives, enabling Ψ_g to encode implicit global geometry in a more sophisticated fashion.

Locality-aware adapter By incorporating the shape-aware adapter, the global shape knowledge would be effectively shared between source and target domains. Nevertheless, for successful domain adaptation, it is indispensable to introduce an exclusive module that can learn target-specialized local characteristics. To this end, we design a locality-aware adapter Ψ_l, which is founded on Graph Convolution layer [11] to aggregate regional information (Figure 4 (b)). For a given point $\mathbf{p}_i$ sampled by FPS, k-nearest neighbors (k-NN) graph is constructed with nearest local neighbors among full input points, denoted as $\mathcal{N}(i)$. Then, the Graph Convolution operation is performed on the k-NN graph to update the representation of $\mathbf{p}_i$ via adjacent point embeddings. This operation is formulated as follows:

$$\Psi_l(\mathbf{p}_i) = \Theta^\mathsf{T} \sum_{\mathbf{p}_j \in \mathcal{N}(i) \cup \{\mathbf{p}_i\}} \frac{e_{j,i}}{\deg(\mathbf{p}_j)\deg(\mathbf{p}_i)} \Phi(\mathbf{p}_j), \quad (3)$$

where Θ is a matrix of filter parameters, e_{ji} is an edge weight of edge $\{\mathbf{p_j}, \mathbf{p_i}\}$, and $\deg(\mathbf{p}_i)$ is a degree normalization constant computed as $\deg(\mathbf{p}_i) = 1 + \sum_{\mathbf{p}_j \in \mathcal{N}(i)} e_{j,i}$. In this work, we consider edge feature e_{ji} as 1. We would like to emphasize that locality-aware adapter Ψ_l passes only the target data, generating representation

vectors that are added to the existing model path via a residual connection. The detailed training procedures of ours are described at the following paragraph.

Training procedure We here describe the training procedure of *PC-Adapter* using a single point cloud instance per each domain for brevity, though the model is trained by mini-batches. To promote effective adaptation, we provide *different* training paths for the source and target domains. For each iteration, we pass source data $X^{\text{src}} = \{\mathbf{p}_i\}_{i=1}^{m}$ to the shared encoder Φ, and the encoder output $\{\Phi(\mathbf{p}_i)\}_{i=1}^{m}$ with its coordinate information is fed to the shape-aware adapter Ψ_g. Then, Ψ_g learns the global shape of the source domain by performing self-attention between representative points. The representation obtained from Ψ_g is combined with the original encoder output, $\texttt{Combine}(\{\Psi_g(\mathbf{p}_i)\}_{i=1}^{m'}, \{\Phi(\mathbf{p}_i)\}_{i=1}^{m})$, and then transmitted to the classifier f. Subsequently, our method makes the target data $X^{\text{trgt}} = \{\mathbf{p}'_i\}_{i=1}^{m}$ forward through all components, *i.e.* Φ, Ψ_g, and Ψ_l. Following the shared Ψ_g, *PC-Adapter* identifies the global anatomy of X^{trgt} by receiving learned global knowledge of the source. In conjunction with global structure adaptation, our *PC-Adapter* learns local properties via locality-aware adapter Ψ_l to compensate for the characteristic discrepancy between two domains. Lastly, the outputs of two adapters are combined with encoder output as $\texttt{Combine}(\{\Psi_g(\mathbf{p}_i)\}_{i=1}^{m'}, \{\Psi_l(\mathbf{p}_i)\}_{i=1}^{m'}, \{\Phi(\mathbf{p}_i)\}_{i=1}^{m'})$, and they are fed into the classifier f. During training target point clouds, we lower the learning rate for components where the source data have passed (*i.e.* Φ and Ψ_g) to preserve the implicit shape knowledge of the source domain. The efficacy of weakly updating these components is confirmed in Appendix A. We select the $\texttt{Combine}$ operation among summation and averaging.

4.2. Distribution-guided Unbiased Pseudo-label

There remains a challenge of how to annotate target point clouds with pseudo-labels which are readily biased towards the source label distribution. With this goal in mind, we devise a bias-resilient pseudo-labeling method that identifies the classifier bias in advance and adjusts model confidences before selecting pseudo-labels. We start by approximating the confidence distribution for each class t given source training dataset $\mathcal{S}_{\text{train}}$ as a skewed beta distribution, $p(c_t|\mathcal{S}_{\text{train}}) \approx \texttt{Beta}(\hat{\alpha}_t, \hat{\beta}_t)$. We justify this modeling for the following two reasons: (1) the average prediction confidence for intra-class data of each class is significantly different, which strongly correlates with the source label distribution (Figure 3 (d) and (h)). (2) the skewness of confidence distribution is determined by the degree of a majority of the class. In other words, for major classes, confidence distributions are likely to be left-tailed as samples of major classes are frequently encountered during training, while for minor

Table 1. Classification accuracy of proposed *PC-Adapter* against other baselines on PointDA-10, averaged over three repetitions ($\pm$ SEM). *SSL*, *Adv.*, and *ST* denote self-supervised learning, adversarial learning, and self-training (include data augmentation), respectively.

Method	SSL	Adv.	ST	M→S	M→S*	S→M	S→S*	S*→M	S*→S	Avg.
Supervised				93.9 $_{\pm0.2}$	78.4 $_{\pm0.6}$	96.2 $_{\pm0.1}$	78.4 $_{\pm0.6}$	96.2 $_{\pm0.1}$	93.9 $_{\pm0.2}$	89.5
DANN [6]		✓		74.8 $_{\pm2.8}$	42.1 $_{\pm0.6}$	57.5 $_{\pm0.4}$	50.9 $_{\pm1.0}$	43.7 $_{\pm2.9}$	71.6 $_{\pm1.0}$	56.8
PointDAN [18]		✓		**83.9** $_{\pm0.3}$	44.8 $_{\pm1.4}$	63.3 $_{\pm1.1}$	45.7 $_{\pm0.7}$	43.6 $_{\pm2.0}$	56.4 $_{\pm1.5}$	56.3
DefRec+PCM [1]	✓		✓	83.7 $_{\pm0.6}$	42.6 $_{\pm0.9}$	71.4 $_{\pm1.5}$	46.1 $_{\pm1.7}$	71.5 $_{\pm1.0}$	74.6 $_{\pm0.5}$	65.0
GAST [29]	✓		✓	83.7 $_{\pm0.2}$	54.9 $_{\pm0.9}$	71.1 $_{\pm0.2}$	53.6 $_{\pm0.2}$	48.4 $_{\pm3.3}$	58.7 $_{\pm0.1}$	61.7
GLRV [5]	✓	✓	✓	78.4 $_{\pm0.6}$	53.6 $_{\pm0.1}$	64.4 $_{\pm3.8}$	49.1 $_{\pm2.8}$	59.4 $_{\pm2.5}$	71.2 $_{\pm3.7}$	62.7
PC-Adapter			✓	83.3 $_{\pm0.3}$	**58.2** $_{\pm0.4}$	**77.5** $_{\pm0.2}$	**53.7** $_{\pm0.2}$	**73.7** $_{\pm0.5}$	**75.4** $_{\pm0.4}$	**70.3**

Table 2. Classification accuracy of the *PC-Adapter* compared to baselines on GraspNetPC-10, averaged over three repetitions ($\pm$ SEM). Syn., Kin., and RS. refer to the synthetic, kinetic, and realsense domains, respectively.

Method	SSL	Adv.	ST	Syn.→Kin.	Syn.→RS.	Kin.→RS.	RS.→Kin.	Avg.
Supervised				97.2 $_{\pm0.8}$	95.6 $_{\pm0.4}$	95.6 $_{\pm0.3}$	97.2 $_{\pm0.4}$	96.4
DANN [6]		✓		78.6 $_{\pm0.3}$	70.3 $_{\pm0.5}$	46.1 $_{\pm2.2}$	67.9 $_{\pm0.3}$	65.7
PointDAN [18]		✓		77.0 $_{\pm0.2}$	72.5 $_{\pm0.3}$	65.9 $_{\pm1.2}$	82.3 $_{\pm0.5}$	74.4
RS [19]	✓			67.3 $_{\pm0.4}$	58.6 $_{\pm0.8}$	55.7 $_{\pm1.5}$	69.6 $_{\pm0.4}$	62.8
DefRec+PCM [1]	✓		✓	80.7 $_{\pm0.1}$	70.5 $_{\pm0.4}$	65.1 $_{\pm0.3}$	77.7 $_{\pm1.2}$	73.5
GAST [29]	✓		✓	81.3 $_{\pm1.8}$	72.3 $_{\pm0.8}$	61.3 $_{\pm0.9}$	80.1 $_{\pm0.3}$	73.8
GLRV [5]	✓	✓	✓	92.2 $_{\pm2.1}$	74.4 $_{\pm0.8}$	70.2 $_{\pm2.4}$	86.7 $_{\pm0.3}$	73.8
ImplicitPCDA [20]	✓		✓	94.6 $_{\pm0.4}$	80.5 $_{\pm0.2}$	76.8 $_{\pm0.4}$	85.9 $_{\pm0.3}$	84.4
***PC-Adapter*+ImplicitPCDA**	✓		✓	**97.5** $_{\pm0.3}$	72.5 $_{\pm0.2}$	**82.5** $_{\pm0.7}$	**88.6** $_{\pm1.7}$	**85.3**

classes, the confidence distributions are more likely to be right-tailed. Let us denote the indices of training samples belonging to class t as $\mathcal{S}_{\text{train}}^t$, and the confidence on class t for the i-th sample as $c_{i,t}$. Then, for each class t, two unknown parameters - $\hat{\alpha}_t$ and $\hat{\beta}_t$ - can be estimated using sample mean of confidences, $\bar{c}_t = \frac{1}{|\mathcal{S}_{\text{train}}^t|}\sum_{i\in\mathcal{S}_{\text{train}}^t} c_{i,t}$, and sample variance, $\bar{v}_t = \frac{1}{|\mathcal{S}_{\text{train}}^t|-1}\sum_{i\in\mathcal{S}_{\text{train}}^t}(c_{i,t}-\bar{c}_t)^2$, based on Method of Moments [14]:

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

The detailed proof is provided at Appendix B.

Armed with approximated class-wise confidence distributions, our method adjusts the confidence score of each class by reflecting its percentile rank within its own confidence distribution, prior to assigning pseudo-labels. By considering the *relative* confidence of each class within its distribution, our method encourages the selection of minority classes as pseudo-labels, thereby allowing the model to obtain unbiased pseudo-labels without requiring access to the label distribution of the target domain. Specifically, our pseudo-labeling method modifies the confidence score $c_{i,t}$ of each class t such that it increases proportionally with the percentile rank r_i as follows: $\tilde{c}_{i,t} = c_{i,t} \cdot \big(\frac{1}{1-r_i+r_0}\big)$ to remove the effect of bias in the source label distribution, where r_i is derived from percentile point function of beta distribution (*i.e.* inverse cumulative density function) and r_0 is a hyperparameter that controls the intensity of ad-

justment. Lastly, the pseudo-label $\hat{y}_i^{\text{trgt}}$ for the target point cloud $\boldsymbol{X}_i^{\text{trgt}}$ is annotated as $\arg\max_t \tilde{c}_{i,t}$. Note that our pseudo-labeling strategy requires no additional model forward/backward pass as distributions are computed based on the model predictions acquired during the training phase.

While our distribution-guided pseudo-labeling mitigates the selection of biased pseudo-labels for target data, in extreme cases (*e.g.* when the source domain exhibits high class imbalance), our confidence rectification might not succeed in altering the order of prediction scores due to significant gaps between intra-class prediction scores. To address this challenge, we incorporate the regularization loss term $\mathcal{L}_{\text{centroid}}$ for each training batch, enforcing the orthogonalization of averaged representations for each class.

5. Experiments

We conduct a series of experiments to verify our *PC-Adapter* in various domain shift scenarios. In Section 5.2 and 5.3, we evaluate the adaptation performance on two shape classification benchmarks and one part segmentation benchmark dataset, comparing it with recent unsupervised domain adaptation methods for the point cloud. To further demonstrate the effectiveness of our framework, we test our method under data-scarce conditions and analyze its time complexity (Section 5.4). Lastly, we perform an ablation study to examine the contribution of each sub-component of *PC-Adapter* (Section 5.5).

Table 3. Part segmentation results on the PointSegDA dataset. Mean per-class IoU (mIoU) for our method and other baselines are reported, averaged over three repetitions ($\pm$ SEM).

Method	FAUST→ ADOBE	FAUST→ MIT	FAUST→ SCAPE	MIT→ ADOBE	MIT→ FAUST	MIT→ SCAPE	ADOBE→ FAUST	ADOBE→ MIT	ADOBE→ SCAPE	SCAPE→ ADOBE	SCAPE→ FAUST	SCAPE→ MIT	Avg.
Supervised	80.9 ±7.2	81.8 ±0.3	82.4 ±1.2	80.9 ±7.2	84.0 ±1.8	82.4 ±1.2	84.0 ±1.8	81.8 ±0.3	82.4 ±1.2	80.9 ±7.2	84.0 ±1.8	81.8 ±0.3	82.3
Unsupervised	78.5 ±0.4	60.9 ±0.6	66.5 ±0.6	26.6 ±3.5	33.6 ±1.3	69.9 ±1.2	38.5 ±2.2	31.2 ±1.4	30.0 ±3.6	74.1 ±1.0	68.4 ±2.4	64.5 ±0.5	53.6
Adapt-SegMap [23]	70.5 ±3.4	60.1 ±0.6	65.3 ±1.3	49.1 ±9.7	54.0 ±0.5	62.8 ±7.6	44.2 ±1.7	35.4 ±0.3	35.1 ±1.4	70.1 ±2.5	67.7 ±1.4	63.8 ±1.2	56.5
RS [19]	78.7 ±0.5	60.7 ±0.4	66.9 ±0.4	59.6 ±5.0	38.4 ±2.1	70.4 ±1.0	44.0 ±0.6	30.4 ±0.5	36.6 ±0.8	70.7 ±0.8	73.0 ±1.5	65.3 ±0.1	57.9
DefRec+PCM [1]	79.7 ±0.3	61.8 ±0.1	67.4 ±1.0	67.1 ±1.0	48.6 ±2.4	72.6 ±0.5	46.9 ±1.0	33.2 ±0.3	37.6 ±0.1	66.4 ±0.9	72.2 ±1.2	66.2 ±0.9	60.0
PC-Adapter	78.8 ±0.5	61.5 ±0.1	**69.2** ±0.3	**68.9** ±0.2	**62.1** ±0.3	**72.8** ±0.1	**49.0** ±0.1	**39.6** ±0.1	**43.1** ±0.1	**74.4** ±0.7	**75.7** ±0.3	**66.4** ±0.1	**63.5**

5.1. Experimental Settings

Datasets Two shape classification datasets, PointDA-10 [18] and GraspNetPC-10 [20] are adopted for our experiments. PointDA-10 is constructed by extracting the samples in 10 shared classes from three datasets - ModelNet40 (**M**) [27], ShapeNet (**S**) [2], and ScanNet (**S***) [4]. Using these subsets, we conduct experiments on 6 different domain adaptation settings. To further verify our method on synthetic-to-real and real-to-real adaptation scenarios, we also conduct experiments on GraspNetPC-10 [20]. GraspNetPC-10 is curated via raw depth scans on real-world and also synthetic scenes which consist of 3D CAD models of objects using two different cameras, Kinect2 and Intel Realsense. Following [20], we conduct experiments on 4 domain adaptation scenarios, where all domains share the same 10 classes. Among them, scenarios whose source domains correspond to Syn. represent *synthetic*-to-real shifts, whereas the rest indicate real-to-real shifts where sensor noise differs. For 3D part segmentation tasks, we utilize PointSegDA dataset [1] in our experiments, which comprises 4 different subsets: FAUST, MIT, ADOBE, and SCAPE. These subsets encompass 8 classes of human body parts varying in point distribution, pose, and scanned individuals. We validate our framework on 12 different domain shift conditions, following the experimental settings of the [1]. Detailed statistics are provided in Appendix D.

Implementation detail In our experiments, we set the farthest point sampling (FPS) ratio to 0.1, and k to 5 for k-NN graph in locality-aware adapter Ψ_l. We adopt DGCNN [26] as the feature encoder, consistent with prior works [29, 5, 20]. We follow the evaluation protocol of [5, 29] for PointDA-10 dataset and [20] for GraspNetPC-10 dataset. For part segmentation experiments, we adhere to the experimental setup of [1]. Detailed experimental settings and more implementation details of ours are provided in Appendix D.

5.2. Results on Shape Classification

In Table 1, we report the averaged classification accuracy of *PC-Adapter* and baselines on 6 domain adaptation settings in PointDA-10. We first compare with UDA base-lines that do not leverage auxiliary tasks (*i.e.* self-supervised learning tasks) as our *PC-Adapter* does. These baselines perform domain adaptation using such as adversarial domain alignment [18, 5, 6], self-training [29, 5], and mixup augmentation [1]. For this comparison, we exclude the self-supervised tasks from the methods that originally use them. As shown in Table 1, our framework shows significant superiority over the baselines in most settings. The *PC-Adapter* enhances the adaptation performance by an average of **5.3%** compared to the best baseline results. For GraspNetPC-10, the *PC-Adapter* exhibits state-of-the-art performances on three settings when combined with existing self-supervised learning tasks of ImplicitPCDA [20] (Table 2). In particular, *PC-Adapter* solely excels the performance over 80% on Kin.→ RS, yielding the improvement by **5.7%** compared to the second highest result. From these results, we believe that our method can seamlessly integrate with existing self-supervised tasks and potentially exhibit superior performance, despite being designed for succinct adaptation-oriented purposes.

5.3. Results on Part Segmentation

While *PC-Adapter* is primarily designed for point cloud classification tasks, our framework holds the potential for extension into various other 3D understanding tasks. To demonstrate the versatility of our approach, we perform experiments on the part segmentation tasks across 12 domain adaptation settings using the PointSegDA dataset. Unlike shape classification tasks, the crux of part segmentation lies in grasping local characteristics within each segment, enabling accurate predictions for individual points. In light of this, for part segmentation tasks, we only employ the locality-aware adapter Ψ_l in the domain adaptation process, facilitating the transfer of knowledge concerning local characteristics from the source domain. When training target points, we simply annotate target points using pseudo-labels with a fixed threshold γ, while also reducing the learning rate for the feature encoder Φ and the locality-aware adapter Ψ_l, similar to that in shape classification tasks. In Table 3, our method achieves the best segmentation performances in 10 out of 12 domain shift settings, exhibiting an average mIoU enhancement of **3.5%**.
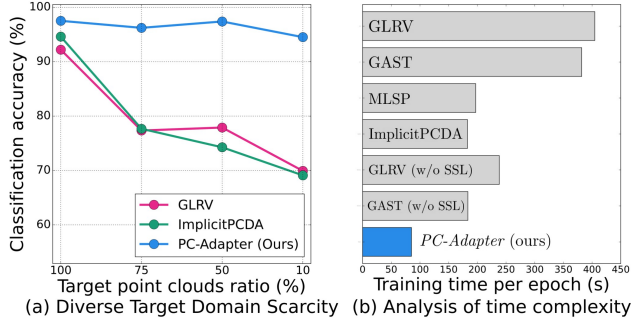
Figure 5. (a) Classification accuracy under diverse target domain scarcity on Syn.→Kin. (b) Time complexity analysis of the *PC-Adapter* against baselines.

5.4. Further Analysis

Data scarcity In practical scenarios, it is common to encounter situations where the target point clouds obtained from the real-world exhibit deficiencies when compared to synthetic source point clouds. To verify our method under this condition, we conduct an experiment by decreasing the number of target data to 75%, 50%, and 10% at the synthetic-to-real domain shift setting in GraspNetPC-10. In Figure 5 (a), the *PC-Adapter* consistently shows the highest adaptation performances over other baselines across multiple sparsities. These results imply that, unlike baselines, our method can efficiently learn target-specific features using only a few target data is given.

Time Complexity To test the cost efficiency of our method, we compare the computation cost of *PC-Adapter* with that of baselines. Training time per epoch is measured over 10 repetitions using an RTX 3090 GPU in the ModelNet→ShapeNet setting. As shown in Figure 5 (b), our method achieves **3.43x** faster training on average compared to the self-supervised learning works, and even faster than self-training (GAST *w/o ssl*) and adversarial training (GLRV *w/o ssl*) methods. This result demonstrates the advantage of deploying *PC-Adapter* for real-world usage.

5.5. Ablation Study

Step-wise Evaluation In Table 4, we conduct a step-wise evaluation to verify the significance of each component in *PC-Adapter*: (1) Domain adaptation using only a shape-aware adapter Ψ_g (the first row), (2) Leveraging both adapters with basic pseudo-labeling (the second row), and (3) Replacing our relative positional encoding σ with conventional point cloud positional encoding method [28, 13], formulated as $\sigma_{ij} = \theta(\mathbf{p}_i - \mathbf{p}_j)$, (the third row). The results clearly show that each module consistently improves performance under domain shift scenarios.

Table 4. Ablation study of *PC-Adapter* on three domain shift settings in PointDA-10.

Modules				S→M	M→S	S*→M
Ψ_g	Ψ_l	Beta PL	Relative σ	Acc.	Acc.	Acc.
✓			✓	72.3	81.4	67.4
✓	✓		✓	75.7	82.7	68.2
✓	✓	✓		74.2	81.5	70.3
✓	✓	✓	✓	**77.5**	**83.3**	**73.7**

Analysis of distribution-guided pseudo-labels We quantitatively analyze the contribution of our pseudo-labeling method in comparison to maximum confidence-based pseudo-labels generation in Table 5. Our beta distribution-guided pseudo-labeling consistently brings improvement in the balanced accuracy compared to the conventional pseudo-labeling approach, showing robustness to the label bias induced from the source domain.

Table 5. Comparison of balanced accuracy (bAcc.) according to pseudo labeling strategies on GraspNetPC-10.

PL Methods	Syn.→Kin.	Syn.→RS.	Kin.→RS.	RS.→Kin.
Maximum confidence PL	96.41	67.93	79.49	87.93
Beta distribution-guided PL (Ours)	**97.66**	**71.99**	**83.52**	**90.31**

6. Conclusion

We scrutinized the essential design principles for domain adaption on point clouds, focusing on which information should be transferred during adaptation. Inspired by this, we proposed an efficient adapter-based domain adaptation framework, *PC-Adapter*, that learns target-specific feature transformations while selectively preserving pertinent knowledge from the source. *PC-Adpater* exhibits fast yet successful domain adaptation abilities under both normal and data-scarce conditions through extensive experiments.

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. 1, 2, 7, 8

[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. 8

[3] Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision transformer adapter for dense predictions. In *The Eleventh International Conference on Learning Representations*, 2023. 2

[4] 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. 8

[5] 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. 1, 2, 4, 7, 8

[6] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016. 7, 8

[7] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*, 2022. 2

[8] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998. Pmlr, 2018. 1

[9] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019. 2

[10] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4893–4902, 2019. 1

[11] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. 6

[12] Hanxue Liang, Hehe Fan, Zhiwen Fan, Yi Wang, Tianlong Chen, Yu Cheng, and Zhangyang Wang. Point cloud domain adaptation via masked local 3d structure prediction. In

[13] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8895–8904, 2019. 3, 9

[14] Karl Pearson. Method of moments and method of maximum likelihood. *Biometrika*, 28(1/2):34–59, 1936. 7

[15] 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. 1, 3

[16] 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. 1, 3

[17] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Abed Al Kader Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *arXiv preprint arXiv:2206.04670*, 2022. 1

[18] 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. 1, 2, 7, 8

[19] Jonathan Sauder and Bjarne Sievers. Self-supervised deep learning on point clouds by reconstructing space. *Advances in Neural Information Processing Systems*, 32, 2019. 1, 7, 8

[20] 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. 1, 2, 4, 7, 8

[21] Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5227–5237, June 2022. 2

[22] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6411–6420, 2019. 1

[23] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7472–7481, 2018. 8

[24] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017. 1

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia

European Conference on Computer Vision, pages 156–172. Springer, 2022. 1, 2, 4

Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 5

[26] 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. 3, 8

[27] 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. 8

[28] 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, 3, 9

[29] 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. 1, 2, 4, 7, 8