Point Cloud Domain Adaptation via Masked Local 3D Structure Prediction

Hanxue Liang[1], Hehe Fan[2], Zhiwen Fan[1], Yi Wang[1],
Tianlong Chen[1], Yu Cheng[3], Zhangyang Wang[1]

[1] The University of Texas at Austin
[2] National University of Singapore
[3] Microsoft Research Redmond

Abstract. The superiority of deep learning based point cloud representations relies on large-scale labeled datasets, while the annotation of point clouds is notoriously expensive. One of the most effective solutions is to transfer the knowledge from existing labeled source data to unlabeled target data. However, domain bias typically hinders knowledge transfer and leads to accuracy degradation. In this paper, we propose a Masked Local Structure Prediction (MLSP) method to encode target data. Along with the supervised learning on the source domain, our method enables models to embed source and target data in a shared feature space. Specifically, we predict masked local structure via estimating point cardinality, position and normal. Our design philosophies lie in: 1) Point cardinality reflects basic structures (*e.g.*, line, edge and plane) that are invariant to specific domains. 2) Predicting point positions in masked areas generalizes learned representations so that they are robust to incompletion-caused domain bias. 3) Point normal is generated by neighbors and thus robust to noise across domains. We conduct experiments on shape classification and semantic segmentation with different transfer permutations and the results demonstrate the effectiveness of our method. Code is available at `https://github.com/VITA-Group/MLSP`.

Keywords: Point cloud representation learning, unsupervised domain adaptation, shape classification, semantic segmentation.

1 Introduction

Point cloud representation learning based on deep neural networks constitute the recent achievements in 3D vision [28,29,42,19]. However, most of them are conducted under supervised learning and therefore require a large amount of annotated data. The expensive labeling cost limits the scalability to more unseen environments. To alleviate this problem, we can transfer the knowledge from existing labeled source data to unseen unlabeled target data. However, due to different point scales, object styles, LiDAR viewpoints, incompletion, sensor noise, *etc*, models often suffer the problem of domain bias, leading to poor accuracy. Although part of the biases can be addressed by data preprocessing, *e.g.*,

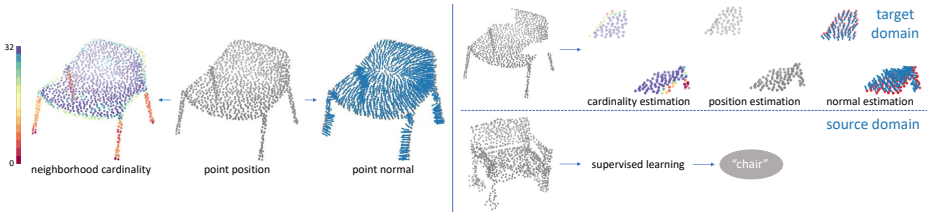

Fig. 1: Illustration of the proposed method. To encode unlabeled target data, we predict masked local structure via estimating point cardinality, position and normal. Point cardinality reflects basic structures such as line, edge and plane, which are invariant to specific domains. Predicting the positions of missing points enables the network to infer structure from partial observation, thus learning a robust representation for incompletion-caused domain bias. Point normal is generated by a few neighbors and thus robust to different individual noisy points across domains. Along with the supervised learning on the source domain, source and target data are embedded into a shared feature space.

unifying point scales and normalizing object sizes, the other biases have to be reduced via well-designed learning approaches.

To alleviate the problem of domain bias, the first solution is to employ adversarial learning [12,41] to directly learn unbiased representations. Specifically, a discriminator is trained to judge whether the learned representations are from the target domain or the source domain, whereas the model is trained to confuse the discriminator [30]. The second solution is to directly align source and target representations in a shared feature space via assigning pseudo labels to target data [10]. The key to this solution is how to achieve pseudo labels and avoid adding noise [47,7]. The third solution is to design self-supervised learning tasks to learn the internal structure of unlabeled target data [1,47,36,7]. Along with the supervised learning on labeled source data, self-supervised tasks enable models to embed source and target data in a shared feature space. Note that, the adversarial learning and pseudo label solutions are independent of point cloud modality and can be borrowed from general learning methods. Therefore, in this work we emphasize exploiting point cloud characters for self-supervised learning tasks design, and make a minor effort in the pseudo label solution.

In this paper, we propose a Masked Local Structure Prediction (MLSP) method for point cloud domain adaptation. Different from most existing methods [1,47], which mainly focus on designing multiple effective SSL tasks (*e.g.*, predicting the angle between two point clouds and the location of distorted area [47]), our method focuses on exploiting domain-invariant features or attributes. Specifically, we mask a random local area of the input point cloud and then ask models to predict the area structure by estimating the neighborhood cardinalities (the number of neighboring points within a predefined radius), positions and normals of the missing points, respectively. First, we find that point cardinality can reflect basic structures of local areas, *e.g.*, line, edge and plane.

As shown in Fig. 1, points at the chair seat have larger cardinalities than points at the legs. Predicting neighborhood cardinality enables models to learn the primitive structure of target objects, which is invariant to different domains. Second, incomplete point clouds are usually encountered in sim-to-real adaptation. Therefore, we follow [1] to predict the positions of missing points to generalize learned representations so that they are invariant to different region missing scenarios between source and target domains. Third, different from image pixels, point clouds are usually not smooth with noisy points, to different degrees in different domains. This hinders models to learn accurate features across different domains[31]. To mitigate this problem, we integrate a normal prediction task in our framework, because point normal is generated by a few neighbors and can be robust to noise. In addition to MLSP, we develop a self-paced learning [17,14,10] variant that leverages prediction probability entropy to select reliable pseudo-labeled samples. The motivation is that a target sample with small entropy of prediction probability is discriminative and its pseudo label is most probably correct.

We conduct extensive experiments including shape classification on the PointDA dataset [30] and semantic segmentation on the PointSegDA dataset [1]. Results demonstrate the effectiveness of our method. Our main contributions are three-fold:

- ⋆ We propose a novel Masked Local Structure Prediction (MLSP) method for point cloud domain adaption, which exploits three types of local attributes to encode unlabeled target data.
- ⋆ We propose a new point cloud attribute, *i.e.*, neighborhood cardinality, which is able to reflect the basic or primitive structure of point clouds.
- ⋆ We achieve the new state-of-the-art accuracy of unsupervised domain adaption on shape classification and segmentation benchmarks.

2 Related Work

Point Cloud Representation Learning Point clouds, which use a set of points with 3D coordinates to specify object positions, are the most straightforward way to preserve 3D spatial information and are very closed to a number of 3D environment understanding applications (*e.g.*, autonomous driving, indoor scene parsing). Point cloud object-level classification and point-level segmentation are two of the fundamental tasks for point cloud processing. Recently, a number of deep neural networks have been proposed to address the two problems [28,29,42,19,46,39,9]. For example, PointNet [28] pioneeringly proposes the first deep neural networks to directly deal with raw point clouds. The successor PointNet++ [29] is based on PointNet and is enhanced to extract both local and global geometric information in a hierarchical way. PointCNN [19] proposes a novel convolution on point cloud to aggregate features in local and equipped with a bottom-up network structure. Recently, PointTransformer [46] adopts the self-attention mechanism to point cloud processing alone with an encoder-decoder

structure and achieves state-of-the-art performance in several point cloud benchmarks. Although straightforward to use, point clouds are difficult to be annotated because it requires a huge amount of labor work, especially for point-level labeling. Therefore, it is necessary and urgent for us to develop an effective method for point cloud-based unsupervised domain adaptation to mitigate the domain gap between labeled data in the source domain and unlabeled data in another domain.

Unsupervised Domain Adaptation Unsupervised domain adaptation (UDA) methods for 2D tasks mainly focus on reducing the discrepancy across different domains. For example, UDA for image classification can be roughly classified into two categories: 1) Minimize the domain discrepancy of a proxy. Methods in [23,5,26,44,32,15,20] measure such discrepancy using the domain distribution statistics. 2) Align feature distributions in an adversarial manner. Works in [11,27,41,12,11,3,18,34,33] either play minimax games at domain level or category level. UDA has also been applied to point cloud processing in works [1,30], which also faces the challenge of gaps in semantic level and domain-agostic feature encoding from local geometries of the point cloud. In the above works, [30] proposes a node module with an adaptive receptive field to model the discriminative local structures and minimize MMD loss to align features in different domains. [1] adopts a self-supervised manner to learn an informative representation with local geometries. The work [24] proposes a multi-level consistency network for 3D detection domain adaptation and enjoys the benefits of the detector-agnostic feature. GAST [47] aims to learn a domain-shared representation of semantic categories by proposing two self-supervised geometric learning tasks as feature regularization.

Self-Supervised Learning on Point Clouds Self-supervised learning (SSL) aims to leverage the raw input as supervision signals by a pre-defined rule or task. SSL is able to learn the representation which benefits downstream tasks. A comprehensive summary of existing methods in SSL can be found in [22]. Several recent works [13,21,35,38,45] studied using SSL framework for learning rich representations of point cloud. [1] studied point cloud reconstruction for DA on point clouds. [13] adopted three tasks including clustering, prediction and reconstruction from noisy input. The work [35] proposed to generate new point cloud by splitting a shape into voxels and then shuffling them. The task is defined as finding the voxel assignment to reconstruct the original point cloud. [38] proposed a task to predict the next point in a space-filling sequence which further boosts the performance. [37] proposed normal prediction for point cloud and aims at multi-task geometric learning network to improve semantic analysis. [45] splitted the point cloud into two parts and proposed to learn a classifier to determine which part it comes from.

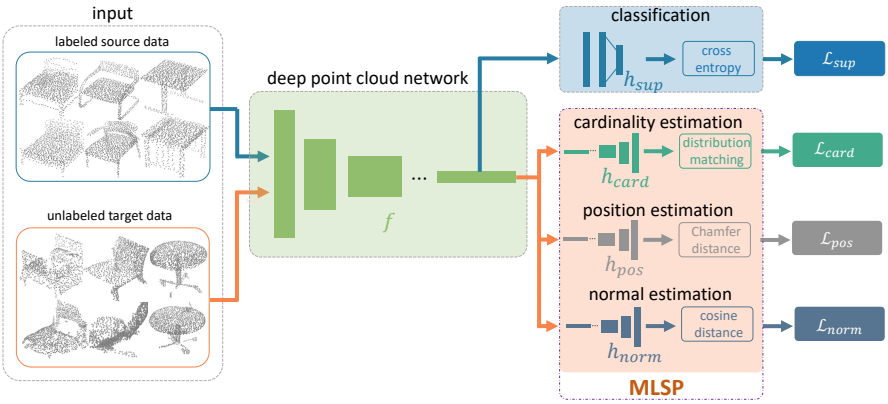

Fig. 2: Overview of the proposed framework for unsupervised domain adaptation on point clouds. The supervised pathway takes as input the point clouds from the source domain and calculates the cross-entropy loss with ground-truth labels. The self-supervised pathway takes point clouds from target domain and calculates the self-supervised loss with the proposed masked local structure prediction, including cardinality, position and normal.

3 Method

In this section, we present our proposed Masked Local Structure Prediction (MLSP) for point cloud domain adaptation. We first formulate the general setup of unsupervised point cloud domain adaptation in Sec. 3.1. Then, we introduce our solution to domain adaptation and describe the proposed MLSP scheme in Sec. 3.2. In Sec. 3.3, we design entropy-based self-paced learning to select reliable pseudo-labeled target data for global representation alignment. We conclude our framework with the overall loss function during our training process in Sec. 3.4. For clarity, we describe MLSP in the context of a classification task, but the same principle applies to the segmentation task as well.

3.1 Overview

We follow the conventional unsupervised domain adaptation (UDA) framework [30] for point cloud representation learning, which aims at transferring the knowledge from a labeled source domain $\mathcal{S} = \{(\boldsymbol{P}_i^{(s)}, y_i^{(s)})\}_{i=1}^{n^{(s)}}$ to an unlabeled target domain $\mathcal{T} = \{\boldsymbol{P}_i^{(t)}\}_{i=1}^{n^{(t)}}$, where $n^{(s)}$ and $n^{(t)}$ denote the numbers of source and target point clouds, respectively. $\boldsymbol{P}_i^{(s)}$ and $\boldsymbol{P}_i^{(t)}$ denote two point clouds from source and target domain, respectively. $y_i^{(s)} \in \mathcal{Y} = \{1, \cdots, L\}$ is the label for source point cloud $\boldsymbol{P}_i^{(s)}$.

We employ a common approach to tackle this learning setup for UDA, which is to learn a shared feature encoder f and trained on two tasks: (1) a supervised

task over source domain $\mathcal{S}$, and (2) a self-supervised task that can be trained over both source domain $\mathcal{S}$ and target domain $\mathcal{T}$. In this work, we propose the MLSP method which includes three distinct components: cardinality, position and normal prediction. An overall pipeline is illustrated in Fig. 2. During training, the supervised task and MLSP task will be trained in an alternating fashion. Specifically, in the supervised task flow, the labeled source samples will be processed by shared encoder f and a classification head h_{sup} to produce the prediction result. A supervision loss will be applied to it. In MLSP flow, after the shared encoder, the unlabeled source/target samples will be passed separately through h_{card}, h_{pos} and h_{norm} to predict the cardinality, position and normal of missing points. Different losses are applied to each prediction.

3.2 Masked Local Structure Prediction

The main idea of the proposed domain adaptation scheme is to learn an unbiased feature across both source and target domains. To this end, we enforce models to predict **cardinality**, **position** and **normal** of masked points so that they can encode the internal structure of unlabed target data. To be more specific, point cardinality reflects the primitive structure of local areas, *e.g.*, line, edge and plane, which is invariant to different domains. Missing position prediction enables models to infer point cloud structure from partial observation. In this way, models can learn robust representation against incompletion-caused domain bias. Point normal is generated from a few neighboring points and thus robust to noisy points. By predicting point normal, our models can learn features mitigating the noisy point distortion across different domains.

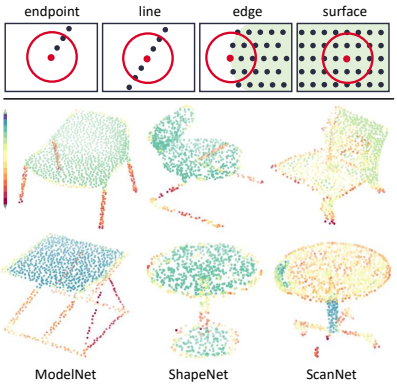

Fig. 3: **Top**: Visualization of point cardinality. It reflects the basis geometry structure, *e.g.*, the cardinalities of points on lines are smaller than that of those on edges and larger than those on endpoints. **Bottom**: Cardinality visualization in three point cloud domains. Cardinalities of different regions or parts show a strong pattern. Therefore, we can use the cardinality attribute to reflect basic local structures, *e.g.*, endpoint, line, edge and surface. This representation is invariant across different domains and classes.

We refer to $\boldsymbol{p} \in \mathbb{R}^3$ as an individual point and denote $\boldsymbol{P}_{\mathcal{M}}$ as the point cloud after masking $\boldsymbol{P}$. For each masked point $\boldsymbol{p} \in \boldsymbol{P} \backslash \boldsymbol{P}_{\mathcal{M}}$, we denote prediction of cardinality as $f_{card}(\boldsymbol{p}) = h_{card} \circ f(\boldsymbol{P}_{\mathcal{M}})$, position as $f_{pos}(\boldsymbol{p}) = h_{pos} \circ f(\boldsymbol{P}_{\mathcal{M}})$, and normal vector as $f_{norm}(\boldsymbol{p}) = h_{norm} \circ f(\boldsymbol{P}_{\mathcal{M}})$, respectively. Following [1], we employ the voxel

partition-based method to generate masked point set $\boldsymbol{P}_{\mathcal{M}}$, which uniformly samples one of the spatial partition of $3 \times 3 \times 3$ cubes and remove all the points in the sampled cube. The masked points are replaced with new points sampled from a Gaussian distribution around the mask center.

Neighborhood Cardinality Prediction Given a sampled point cloud, neighborhood cardinality indicates the basis geometry structures (*e.g.*, line, edge and plane) around each point. As shown in Fig. 3, for points lie on lines (*e.g.*, chair legs), the number of neighbors within a given radius is generally smaller than that of points on planes (*e.g.*, tabletop). The basic geometry structures are shared across different domains and object categories. This observation motivates us to predict point cardinalities to learn domain-invariant features across different domains. More formally, we define neighborhood cardinality as follows:

$$\mathcal{N}(\boldsymbol{p}, r) = \{\boldsymbol{q} \in \boldsymbol{P} \mid \|\boldsymbol{p} - \boldsymbol{q}\| \leq r,\ \boldsymbol{q} \neq \boldsymbol{p}\}, \quad \mathcal{C}(\boldsymbol{p}, r) = |\mathcal{N}(\boldsymbol{p}, r)|, \tag{1}$$

where r is the radius of the neighborhood. As point cardinality only reflects a few basic structures, representing these discrete values in a regression manner is not necessary. Therefore, we formulate the cardinalty prediction task as a classification problem. Specifically, we first experimentally choose the maximum cardinality value of a dataset, denoted by C_{max}. Then we choose the number of cardinality bins K and obtain cardinality interval $c_0 = C_{max}/K$. We consider K as a hyperparameter in our method. From here, we can convert $C(\boldsymbol{p}, r)$ into a groudtruth class label $c = \left\lfloor \frac{C(\boldsymbol{p},r)}{c_0} \right\rfloor$. For each point $\boldsymbol{p}$, the cardinality head will predict its cardinality class. To calculate the cardinality classification loss, we convert the one-hot classification labels into a two-points probability vector representation which is justified in Fig. 4. Specifically, for each point $\boldsymbol{p}$, we convert its cardinality $C(\boldsymbol{p}, r)$ into a probability vector $\boldsymbol{\lambda}$ with two non-zero values:

$$\boldsymbol{\lambda}_i = \begin{cases} 1 - \frac{C(\boldsymbol{p},r) - c * c_0}{c_0} & i = c, \\ \frac{C(\boldsymbol{p},r) - c * c_0}{c_0} & i = c + 1, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Such a representation includes richer distribution information than a one-hot vector. Cardinality prediction for point $\boldsymbol{p}$ will be $\hat{\boldsymbol{\lambda}} = h_{card} \circ f(\boldsymbol{P}_{\mathcal{M}})$ and we employ KL divergence as the measurement to minimize the gap between prediction weights $\hat{\boldsymbol{\lambda}}$ and "ground-truth" weights $\boldsymbol{\lambda}$, leading to the following cardinality loss:

$$\mathcal{L}_{card}(\boldsymbol{P}_{\mathcal{M}}, f_{card}) = \sum_{\boldsymbol{p} \in \boldsymbol{P} \setminus \boldsymbol{P}_{\mathcal{M}}} \sum_{i=1}^{K} \boldsymbol{\lambda}_i \log(\hat{\boldsymbol{\lambda}}_i). \tag{3}$$

Point Position Prediction To recover the missing or masked region, point coordinate information is necessary. Even though non-smoothness and noisiness appear in point cloud input, reconstructed points can not deviate too much from the given ground-truth points. Similar to [1], we utilize the Chamfer distance

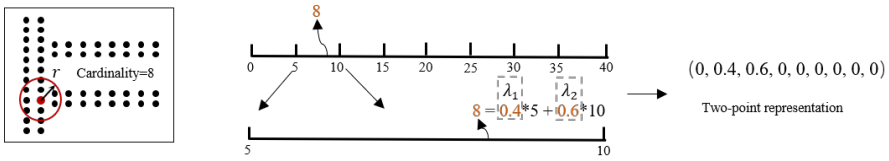

Fig. 4: The proposed two-point representations for the neighborhood cardinality. It is represented by two adjacent bins instead of one-hot vectors

[6] between the predicted point cloud coordinates and ground-truth point coordinates at the masked region. The loss of position prediction has the following form:

$$\mathcal{L}_{pos}(\boldsymbol{P}_{\mathcal{M}}, f_{pos}) = \sum_{\boldsymbol{p}\in\boldsymbol{P}\setminus\boldsymbol{P}_{\mathcal{M}}} \min_{\boldsymbol{q}\in f_{pos}(\boldsymbol{P}_{\mathcal{M}})} \|\boldsymbol{p}-\boldsymbol{q}\|^2 + \sum_{\boldsymbol{q}\in f_{pos}(\boldsymbol{P}_{\mathcal{M}})} \min_{\boldsymbol{p}\in\boldsymbol{P}\setminus\boldsymbol{P}_{\mathcal{M}}} \|\boldsymbol{q}-\boldsymbol{p}\|^2. \quad (4)$$

Point Normal Prediction Normal vector estimation of a point cloud is relevant to maintaining local geometric features. It is generated from a few neighboring points and thus robust to noisy points. Motivated by this observation, we integrate normal estimation as a self-supervised task into our method and expect that models can learn features mitigating the noisy point distortion across different domains. The ground-truth normal vector at the point $\boldsymbol{p}$ is obtained via least-square fit based on all the points in the original point cloud in the neighboring region:

$$\boldsymbol{n}(\boldsymbol{p}) = \underset{\boldsymbol{n}\in\mathbb{R}^3}{\operatorname{argmin}} \sum_{\boldsymbol{q}\in\mathcal{N}(\boldsymbol{p})} (\boldsymbol{n}\cdot(\boldsymbol{q}-\boldsymbol{p}))^2, \quad (5)$$

where $\hat{n}(\boldsymbol{p}) = f_{norm}(\boldsymbol{p})$ is the predicted normal vector of masked point $\boldsymbol{p}$. As shown in Fig 5, the normal is robust against instrumental noise from the point cloud data acquisition. To obtain the appropriate neighbouring point set $\mathcal{N}(\boldsymbol{p})$ of $\boldsymbol{p}$, we test with using both neighbouring points defined with a radius r of point $\boldsymbol{p}$ as $\mathcal{N}(\boldsymbol{p}) = \mathcal{N}(\boldsymbol{p}, r)$ in Equ. 1 and nearest neighbors search, where $\mathcal{N}(\boldsymbol{p}) = kNN(\boldsymbol{p}, k)$. The orientation of normal vector is defined towards the center of the object. We use the cosine-similarity metric to measure the distance between predicted normal and the estimated "true" normal, leading to the following self-supervised loss:

$$\mathcal{L}_{norm}(\boldsymbol{P}_{\mathcal{M}}, f_{norm}) = \sum_{\boldsymbol{p}\in\boldsymbol{P}\setminus\boldsymbol{P}_{\mathcal{M}}} \frac{\boldsymbol{n}(\boldsymbol{p})\cdot\hat{\boldsymbol{n}}(\boldsymbol{p})}{\|\boldsymbol{n}(\boldsymbol{p})\|\|\hat{\boldsymbol{n}}(\boldsymbol{p})\|}. \quad (6)$$

At last, we define the total loss of MLSP for a given masked point cloud $\boldsymbol{P}_{\mathcal{M}}$ as:

$$\mathcal{L}_{pred}(\boldsymbol{P}_{\mathcal{M}}) = \alpha_1\mathcal{L}_{card}(\boldsymbol{P}_{\mathcal{M}}, f_{card}) + \alpha_2\mathcal{L}_{pos}(\boldsymbol{P}_{\mathcal{M}}, f_{pos}) + \alpha_3\mathcal{L}_{norm}(\boldsymbol{P}_{\mathcal{M}}, f_{norm}) \quad (7)$$

where $\alpha_1, \alpha_2, \alpha_3$ are hyperparemters that control the weights of masked point cloud attribute predictions.

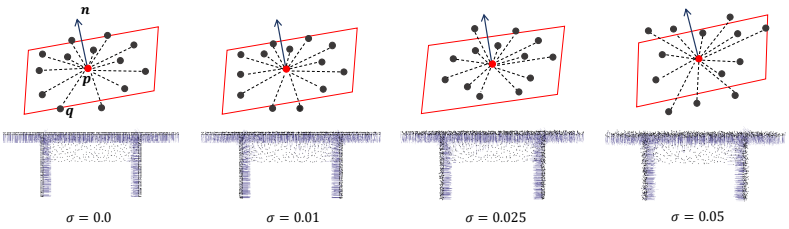

Fig. 5: Normal regression visualization under different scales of Gaussian noise. Top row: our normal regression scheme fits a least-square plane using neighboring points to obtain the pseudo label of "true" normal $\boldsymbol{n}$ at target domain $\mathcal{T}$. As the noise increases, even though the point coordinates are shifting, the estimation of normal shall be consistent. Bottom row: An example point cloud in the PointDA-10 dataset. The estimated normals (blue arrows) are mostly robust even when point position are shifted under large-scale noise ($\sigma = 0.05$). Value of standard derivations σ shown below are set under the normalized point cloud scaling.

3.3 Entropy-guided Self-paced Global Representation Alignment

In addition to the MLSP, we also exploit pseudo labels to boost the domain adaptation accuracy by aligning source and target global representations. The key to this approach is how to assign correct pseudo labels and avoid introducing noise. To this end, we follow [47] to employ self-paced learning [17,14,10,8] to select reliable pseudo-labeled target samples. We first assign the corresponding class of the maximum prediction probability as the pseudo label of each target sample. Then, we only use the target samples whose prediction probability entropies are small enough to train the model with their pseudo labels. We denote $\boldsymbol{y}_i^{(t)}$ as the prediction probability vector of the i-th target sample. We first predict the pseudo label as $\hat{y}_i^{(t)} = \arg\max_l \boldsymbol{y}_{i,l}^{(t)}$. Then, we select the target samples as follows,

$$v_i = \begin{cases} 1, & \sum_{l=1}^{L} -\boldsymbol{y}_{i,l}^{(t)} \log \boldsymbol{y}_{i,l}^{(t)} \leq \gamma, \\ 0, & \text{otherwise}, \end{cases} \tag{8}$$

where $v_i = 1$ indicates the i-th target sample is selected.

The value of the threshold γ controls how confident the predicted class labeling is. Our motivation is that small entropy means the sample is discriminative and can be easily recognized. Therefore, pseudo labels with small prediction probability entropies are more likely correct. Compared to [47], which selects pseudo labels whose probabilities are large enough, entropy is robust and reflects the "entire discriminativeness" of target samples. Note that, although with a fixed γ, as models become stronger and domain bias reduces during training, more and more target samples are selected. As the number of selected target samples increases, models are improved in return. In this way, the accuracy progressively increases.

3.4 Overall Loss

The overall loss function is the sum of supervision loss under $\mathcal{S}$ and linear combination of three estimation tasks as well as the self-paced learning loss under $\mathcal{T}$:

$$\mathcal{L}(\mathcal{S},\mathcal{T},\mathcal{M}) = \sum_{i=1}^{n^{(s)}} \mathcal{L}_{sup}(\boldsymbol{P}_i^{(s)}, y_i^{(s)}) + \sum_{i=1}^{n^{(t)}} \mathcal{L}_{pred}(\boldsymbol{P}_{i,\mathcal{M}}^{(t)}) + \beta \sum_{i=1}^{n^{(t)}} v_i \mathcal{L}_{sup}(\boldsymbol{P}_i^{(t)}, \hat{y}_i^{(t)}), \tag{9}$$

where β controls the importance of the self-paced loss. Both β and $\alpha_1, \alpha_2, \alpha_3$ in Equ. 7 are selected empirically and more details can be found in supplementary. The classification loss $\mathcal{L}_{sup}(\boldsymbol{P}_i^{(t)}, \tilde{y}_i^{(t)})$ under the target is the sum of all the samples that are selected and assigned pseudo labels.

4 Experiments

We evaluate the proposed method under the standard protocol of unsupervised domain adaptation on the task of point cloud data classification and segmentation. We conduct experiments on PointDA-10 and PointSegDA datasets.

4.1 Datasets

PointDA-10[30] contains three widely-used datasets: ModelNet[43], ShapeNet[2] and ScanNet[4]. All three datasets share the same ten categories (bed, table, sofa, chair, etc.). ModelNet($\mathbf{M}$) contains 4183 train samples and 856 test samples and ShapeNet($\mathbf{S}$) contains 17,378 train samples and 2492 test samples. Both ModelNet and ShapeNet are sampled from 3D CAD models. ScanNet($\mathbf{S}^*$) is a more challenging dataset, which contains 6110 train and 1769 test samples. Samples from this dataset are scanned and reconstructed from real-world indoor scenes. The objects often lose some parts and get occluded by surroundings. We follow the data preparation procedure used in [1]. Specifically, all object point clouds in all datasets are aligned along the direction of gravity, while arbitrary rotations along the z-axis are allowed. Each sample is down-sampled to 1024 points and normalized within a unit ball. A typical 80%/20% data split for training and validation on both source and target domains is employed.

PointSegDA is based on a dataset of meshes of human models proposed by [1] and consists of four subsets: ADOBE, FAUST, MIT, and SCAPE. They share eight classes of human body parts (hand, head, feet, etc.) with difference in point distribution, pose and human shapes. PointSegDA differs from PointDA-10 in the type of the domain shifts and the actual shapes representing deformable objects. Thus, PointSegDA allows us to evaluate the proposed method in a fundamentally different setup. For data processing, we follow the data processing in [1] and aligned the shapes with the positive Z-axis and scaled them to the unit cube. Each sample contains 2048 points sampled from mesh vertices and downsampled by farthest point sampling.

4.2 Implementation details

Network architecture We adopt DGCNN [42] with the same configurations as in the official implementation for feature extractor backbone and supervised task head. As for SSL heads, they share the same input as global feature vector concatenated to point-wise feature from the backbone network. Please refer to the supplement for more details about network architecture.

Training procedure During training, we alternate between a batch of source samples and a batch of target samples in all methods. We run each configuration with 3 different random seeds for 150 epochs and use source-validation-based early stopping. We use a fixed batch size of 32 for PointDA-10 and a batch size of 16 for PointSegDA. We adopt ADAM [16] optimizer with learning rate 0.001 and weight decay 0.00005. A cosine annealing learning rate scheduler implemented via PyTorch is assigned in training. β is selected to be 1 and γ is selected over $\{1.516, 1.549, 1.551, 1.628\}$ and we empirically set it to be 1.549. Please refer to the supplement for more details about training.

4.3 Classification Results on PointDA-10 Dataset

We compare our proposed method with a list of recent state-of-the-art point-based DA methods including Domain Adversarial Neural Network (DANN) [11], Point Domain Adaptation Network (PointDAN)[30], Reconstruction Space Network (RS) [35] , Deformation Reconstruction Network with Point Cloud Mixup (DefRec+PCM) [1] and Geometry-aware self-training (GAST) [47]. [1] demonstrates the effectiveness of Mixup training on source domain and we adopt this setup in our PointDA-10 experiments. The supervised model is trained on labeled target data and the baseline model is trained only with labeled source samples. All comparative methods take the same training protocol and the best models are selected according to source-validation based early stopping. Results are presented in Table 1.

By only adopting a self-supervised learning strategy, our approach outperforms all competing domain adaptation methods by a significant margin. Particularly, it improves the average accuracy by relative 2.2% and 3.5% over the state-of-the-art DefRec+PCM [1] and Geometry-aware self-training (GAST) [47]. Considering that our model uses the same backbone DGCNN as the baseline, this performance gain should be attributed to the design of the local feature prediction algorithm, which helps to learn a more adaptive representation across domains. More importantly, in comparison with [1], MLSP achieves superior performance on almost all six adaptation tasks. This demonstrates the effectiveness of our designed local point structure. Particularly, we improve with a large margin on M $\rightarrow$ S* (55.4% *vs.* 51.8%), S*$\rightarrow$M (78.2% *vs.* 73.7%) and S* $\rightarrow$ S (76.1% *vs.* 71.1%) tasks.

After adopting the pseudo label strategy, MLSP can outperform all previous domain adaptation methods. Using entropy-based global alignment(EGA) strategy, we further boost our model's performance to a high level of average accuracy

74.0%. The superior of MLSP+EGA over MLSP+SPST demonstrates the effectiveness of our entropy-guided pseudo label selection design. It is noteworthy that we are able to reach a remarkable performance on two synthetic-to-real M $\rightarrow$ S* and S $\rightarrow$ S* tasks, with mean accuracy 59.1% and 57.6% respectively.

Table 1: Comparative evaluation in classification accuracy (%) averaged over 3 seeds (± SEM) on the PointDA-10 dataset. *BS* indicate baseline method, *PS* means Pseudo Label.

Methods	SSL	PS	M→S	M→S*	S→M	S→S*	S* →M	S* →S	Avg.
Supervised			93.9±0.2	78.4±0.6	96.2±0.1	78.4±0.6	96.2±0.1	93.9±0.2	89.5±0.3
BS (w/o adap.)			83.3±0.7	43.8±2.3	75.5±1.8	42.5±1.4	63.8±3.9	64.2±0.8	62.2±1.8
DANN [11]			74.8±2.8	42.1±0.6	57.5±0.4	50.9±1.0	43.7±2.9	71.6±1.0	56.8±1.5
PointDAN [30]			83.9±0.3	44.8±1.4	63.3±1.1	45.7±0.7	43.6±2.0	56.4±1.5	56.3±1.2
RS [35]	✓		79.9±0.8	46.7±4.8	75.2±2.0	51.4±3.9	71.8±2.3	71.2±2.8	66.0±1.6
DefRec+PCM [1]	✓		81.7±0.6	51.8±0.3	78.6±0.7	54.5±0.3	73.7±1.6	71.1±1.4	68.6±0.8
GAST [47]	✓		83.9±0.2	56.7±0.3	76.4±0.2	55.0±0.2	73.4±0.3	72.2±0.2	69.5±0.2
Ours	✓		83.7±0.4	55.4±1.8	77.1±0.9	55.6±0.7	78.2±1.5	76.1±0.5	71.0±0.8
GAST+SPST [47]	✓	✓	74.8±0.1	**59.8±0.2**	80.8±0.6	56.7±0.2	81.1±0.8	74.9±0.5	73.0±0.4
Ours+SPST	✓	✓	85.7±0.6	59.4±1.3	82.3±0.9	57.3±0.7	**82.2±0.5**	76.4±0.5	73.8±1.0
Ours+EGA	✓	✓	**86.2±0.8**	59.1±0.9	**83.5±0.4**	**57.6±0.6**	81.2±0.4	**76.4±0.3**	**74.0±0.5**

4.4 Segmentation Results on PointSegDA Dataset

We evaluate the generalization ability of MLSP beyond classification tasks on PointSegDA dataset. We compare against several methods including unsupervised baseline, RS [35], Adapt-SegMap [40] and DefRec+PCM [1] on the mean Intersection over Union (IoU). As is shown in Table 2, our model achieves the highest accuracy compared with all previous adaptation methods, which demonstrates that MLSP can generalize well on segmentation task. Particularly, it demonstrates the best performance on most adaptations.

4.5 Ablation Study and Analysis

In this section, we conduct ablation study experiments to analyze the effectiveness of different components of the model. Experiments are conducted on PointDA-10 dataset.

Effect of neighborhood cardinality, position and normal prediction In this part, we separately evaluate the performance of neighborhood cardinality, position and normal prediction. To get a unified measure of performance, we tuned the loss weight between SSL loss and source domain supervised loss, so that the best performance can be reached for each prediction component. More details can be found in supplementary and the result is shown in Table 3. Evidently, all three prediction components can improve the domain adaptation

Table 2: Point Cloud Segmentation Performance(mean IoU.) on PointSegDA dataset, averaged over three runs (± SEM).

Methods	FAUST to ADOBE	FAUST to MIT	FAUST to SCAPE	MIT to ADOBE	MIT to FAUST	MIT to SCAPE	ADOBE to FAUST	ADOBE to MIT	ADOBE to SCAPE	SCAPE to ADOBE	SCAPE to FAUST	SCAPE to MIT	Avg.
Supervised-T	84.0±1.8	84.0±1.8	84.0±1.8	81.8±0.3	81.8±0.3	81.8±0.3	80.9±7.2	80.9±7.2	80.9±7.2	82.4±1.2	82.4±1.2	82.4±1.2	82.3±2.6
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	65.5±0.5	53.6±1.6
Adapt-SegMap [40]	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±2.6
RS [35]	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±1.1
DefRec [1]	79.7±0.3	61.8±0.1	**67.4±1.0**	67.1±1.0	40.1±1.4	**72.6±0.5**	42.5±0.3	28.9±1.5	32.2±1.2	66.4±0.9	72.2±1.2	66.2±0.9	58.1±0.9
Ours	**80.9±0.4**	60.0±0.2	65.5±0.5	**67.3±0.3**	40.4±0.6	70.8±1.0	**45.4±1.0**	31.1±0.8	**38.4±0.5**	**74.8±1.0**	72.5±0.5	**66.6±0.9**	**59.5±0.6**

Table 3: Ablation study of MLSP prediction tasks on PointDA-10 dataset

Cardinality	Position	Normal	M→S	M→S*	S→M	S→S*	S* →M	S* →S	Avg.
✓			83.0	54.3	74.0	53.5	71.9	75.6	68.7
	✓		82.1	52.3	76.2	53.7	75.1	72.4	68.6
		✓	83.5	49.4	74.9	53.4	75.5	72.4	68.2
✓		✓	83.6	52.6	74.8	52.7	74.5	75.6	69.0
	✓	✓	83.1	56.0	77.8	55.7	76.4	72.2	70.2
✓	✓		82.5	54.9	76.6	55.5	76.8	77.3	70.6
✓	✓	✓	83.7	55.4	77.1	55.6	78.2	76.1	71.0

performance by a good margin. Among them, cardinality prediction can provide the highest average accuracy of 68.7%. By combining these three components together, we are able to further improve the performance, which demonstrates the compatibility between these components. The combination of position estimation and normal estimation provides the best accuracy in the sim-to-real scenario.

Effect of class number of cardinality For cardinality estimation, we consider the number of cardinality classes as a hyper-parameter and analyze its influence on the domain adaptation performance. As is shown in Table 3 in supplement, 8-class reaches the highest performance accuracy on PointDA-10. When we further increase the class number, we notice a performance drop. This might be explained that when we over-classify the cardinality, features of some similar geometric structure with different cardinalities will be encourage to be separative.

Effect of loss for cardinality estimation In this part, we compare our proposed cardinality estimation loss with cross-entropy loss. Experiments demonstrate the superiority of our proposed loss over directly using cross-entropy loss. And the former achieves better accuracy on four tasks and reaches a higher average accuracy.

Effect of number of neighboring points for normal estimation To validate that point normal attribute can help mitigate the distortion caused by noisy points, we change the number of neighboring points to calculate each point's ground truth normal, and compare their performance in PointDA-10 benchmark. As shown in Fig.1 in supplement, we notice that improving the number

(a) w/o DA: S^* (b) w/o DA: $S^* \rightarrow M$ (c) MLSP:S^* (d) MLSP: $S^* \rightarrow M$

Fig. 6: The t-SNE visualization of learnt feature vectors from model f under source (Scannet) and target (ModelNet) domain. First two images are generated without domain adaptation. Different colors reveals different classes.

Table 4: Cardinality prediction on PointDA-10 dataset with different loss.

Loss Function	M→S	M→S*	S→M	S→S*	S* →M	S* →S	Average
CE	83.4	55.6	73.6	52.6	69.9	71.9	67.8
Ours	83.0	54.3	74.0	53.5	71.9	75.6	68.7

of neighboring points considered for normal estimation will help the transfer learning process. While when we further increase it after 15 points, we witness a small performance drop. This can be explained that if we use too many neighboring points, the smoothing effect is too strong which can affect learning detail structure.

Feature Visualization We utilize t-SNE [25] to visualize the feature distribution of the 1024-dimension latent codes in Fig. 6. Without domain adaptation, features of different classes in the target domain are mixed up. After adaptation. the distribution of the features in the target domain demonstrates clear clusters.

5 Conclusions

Our work focuses on designing an algorithm to solve the object point cloud domain adaptation problems. Starting from a self-supervised framework, we propose MLSP method that is beneficial to point cloud DA goal, which includes cardinally, position and normal estimation. We validate the effectiveness of these three point features in mitigating the domain bias problems. And our method achieves state-of-art performance on point cloud shape classification and semantic segmentation benchmarks.

References

1. Achituve, I., Maron, H., Chechik, G.: Self-supervised learning for domain adaptation on point clouds. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 123–133 (2021)
2. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
3. Chen, X., Wang, S., Long, M., Wang, J.: Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation. In: International conference on machine learning. pp. 1081–1090. PMLR (2019)
4. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5828–5839 (2017)
5. Deng, Z., Luo, Y., Zhu, J.: Cluster alignment with a teacher for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9944–9953 (2019)
6. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 605–613 (2017)
7. Fan, H., Chang, X., Zhang, W., Cheng, Y., Sun, Y., Kankanhalli, M.: 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. pp. 6377–6386 (2022)
8. Fan, H., Liu, P., Xu, M., Yang, Y.: Unsupervised visual representation learning via dual-level progressive similar instance selection. IEEE Transactions on Cybernetic (2021). https://doi.org/10.1109/TCYB.2021.3054978
9. Fan, H., Yu, X., Ding, Y., Yang, Y., Kankanhalli, M.: Pstnet: Point spatio-temporal convolution on point cloud sequences. In: International Conference on Learning Representations (2020)
10. Fan, H., Zheng, L., Yan, C., Yang, Y.: Unsupervised person re-identification: Clustering and fine-tuning. ACM Trans. Multim. Comput. Commun. Appl. **14**(4), 83:1–83:18 (2018). https://doi.org/10.1145/3243316
11. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V.: Domain-adversarial training of neural networks. The journal of machine learning research **17**(1), 2096–2030 (2016)
12. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)
13. Hassani, K., Haley, M.: Unsupervised multi-task feature learning on point clouds. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8160–8171 (2019)
14. Jiang, L., Meng, D., Yu, S., Lan, Z., Shan, S., Hauptmann, A.G.: Self-paced learning with diversity. In: Advances in Neural Information Processing Systems. pp. 2078–2086 (2014)
15. Kang, G., Jiang, L., Yang, Y., Hauptmann, A.G.: Contrastive adaptation network for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4893–4902 (2019)
16. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

17. Kumar, M.P., Packer, B., Koller, D.: Self-paced learning for latent variable models. In: Advances in Neural Information Processing Systems. pp. 1189–1197 (2010)
18. Lee, C.Y., Batra, T., Baig, M.H., Ulbricht, D.: Sliced wasserstein discrepancy for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10285–10295 (2019)
19. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: Pointcnn: Convolution on x-transformed points. Advances in neural information processing systems **31** (2018)
20. Liang, H., Zhang, Q., Dai, P., Lu, J.: Boosting the generalization capability in cross-domain few-shot learning via noise-enhanced supervised autoencoder. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9424–9434 (2021)
21. Liang, H., Jiang, C., Feng, D., Chen, X., Xu, H., Liang, X., Zhang, W., Li, Z., Van Gool, L.: Exploring geometry-aware contrast and clustering harmonization for self-supervised 3d object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3293–3302 (2021)
22. Liu, X., Zhang, F., Hou, Z., Mian, L., Wang, Z., Zhang, J., Tang, J.: Self-supervised learning: Generative or contrastive. IEEE Transactions on Knowledge and Data Engineering (2021)
23. Long, M., Cao, Y., Cao, Z., Wang, J., Jordan, M.I.: Transferable representation learning with deep adaptation networks. IEEE transactions on pattern analysis and machine intelligence **41**(12), 3071–3085 (2018)
24. Luo, Z., Cai, Z., Zhou, C., Zhang, G., Zhao, H., Yi, S., Lu, S., Li, H., Zhang, S., Liu, Z.: Unsupervised domain adaptive 3d detection with multi-level consistency. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8866–8875 (2021)
25. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(11) (2008)
26. Pan, Y., Yao, T., Li, Y., Wang, Y., Ngo, C.W., Mei, T.: Transferrable prototypical networks for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2239–2247 (2019)
27. Pinheiro, P.O.: Unsupervised domain adaptation with similarity learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8004–8013 (2018)
28. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 652–660 (2017)
29. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in neural information processing systems **30** (2017)
30. Qin, C., You, H., Wang, L., Kuo, C.C.J., Fu, Y.: Pointdan: A multi-scale 3d domain adaption network for point cloud representation. Advances in Neural Information Processing Systems **32** (2019)
31. Rakotosaona, M.J., La Barbera, V., Guerrero, P., Mitra, N.J., Ovsjanikov, M.: Pointcleannet: Learning to denoise and remove outliers from dense point clouds. In: Computer Graphics Forum. vol. 39, pp. 185–203. Wiley Online Library (2020)
32. Rozantsev, A., Salzmann, M., Fua, P.: Beyond sharing weights for deep domain adaptation. IEEE transactions on pattern analysis and machine intelligence **41**(4), 801–814 (2018)
33. Saito, K., Ushiku, Y., Harada, T., Saenko, K.: Adversarial dropout regularization. arXiv preprint arXiv:1711.01575 (2017)

34. Saito, K., Watanabe, K., Ushiku, Y., Harada, T.: Maximum classifier discrepancy for unsupervised domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3723–3732 (2018)
35. Sauder, J., Sievers, B.: Self-supervised deep learning on point clouds by reconstructing space. Advances in Neural Information Processing Systems **32** (2019)
36. Shen, Y., Yang, Y., Yan, M., Wang, H., Zheng, Y., Guibas, L.J.: Domain adaptation on point clouds via geometry-aware implicits. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7223–7232 (2022)
37. Tang, L., Chen, K., Wu, C., Hong, Y., Jia, K., Yang, Z.X.: Improving semantic analysis on point clouds via auxiliary supervision of local geometric priors. IEEE Transactions on Cybernetics (2020)
38. Thabet, A., Alwassel, H., Ghanem, B.: Mortonnet: Self-supervised learning of local features in 3d point clouds. arXiv preprint arXiv:1904.00230 (2019)
39. Thomas, H., Qi, C.R., Deschaud, J.E., Marcotegui, B., Goulette, F., Guibas, L.J.: Kpconv: Flexible and deformable convolution for point clouds. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 6411–6420 (2019)
40. Tsai, Y.H., Hung, W.C., Schulter, S., Sohn, K., Yang, M.H., Chandraker, M.: Learning to adapt structured output space for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7472–7481 (2018)
41. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7167–7176 (2017)
42. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. Acm Transactions On Graphics (tog) **38**(5), 1–12 (2019)
43. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1912–1920 (2015)
44. Xie, S., Zheng, Z., Chen, L., Chen, C.: Learning semantic representations for unsupervised domain adaptation. In: International conference on machine learning. pp. 5423–5432. PMLR (2018)
45. Zhang, L., Zhu, Z.: Unsupervised feature learning for point cloud by contrasting and clustering with graph convolutional neural network. arXiv preprint arXiv:1904.12359 (2019)
46. Zhao, H., Jiang, L., Jia, J., Torr, P.H., Koltun, V.: Point transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 16259–16268 (2021)
47. Zou, L., Tang, H., Chen, K., Jia, K.: Geometry-aware self-training for unsupervised domain adaptation on object point clouds. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6403–6412 (2021)