Front.Comput.Sci.

RESEARCH ARTICLE

A Learnable Self-supervised Task for Unsupervised Domain Adaptation on Point Cloud Classification and Segmentation

Shaolei Liu [1,2,*], Xiaoyuan Luo [1,2,*], Kexue Fu [1,2], Manning Wang [1,2,†], and Zhijian Song [1,2,†]

[1] Shanghai Key Laboratory of Medical Image Computing and Computer Assisted Intervention
[2] Digital Medical Research Center, School of Basic Medical Science, Fudan University, Shanghai 200032, China

1 More Method Details

1.1 Self-Supervised Task Using A Learnable Transformation

Many self-supervised tasks on point clouds rely on hand-crafted transformations of the point clouds, and different transformations will lead to different features being learned by the encoder. Transformations explored in literature include adding noise [1], removing a part of the point cloud [1], shuffling the subblocks of a point cloud [2], and so on. However, hand-crafted transformations are limited and nonflexible, and this may restrict the effectiveness of the trained encoder. In this section, we propose a learnable transformation φ based on deep neural networks.

We construct a transformation network φ_ω parameterized by ω to transform the original point cloud x into x'. The auxiliary network f_{aux} is applied to reconstruct the original point cloud from x', and the reconstructed point cloud is denoted as $\hat{x}$:

$$x' = \varphi_\omega(x) \tag{1}$$

$$\hat{x} = f_{h\_aux}\left(f_{enc}\left(x'\right)\right) \tag{2}$$

The objective of the transformation network φ_ω is to maximize the chamfer distance between the original point cloud and the transformed point cloud, and the objective of the auxiliary reconstruction network, which consists of the shared encoder f_{enc} and the head f_{h_aux}, is to minimize the chamfer distance between the reconstructed point cloud and the original point cloud. So, the optimization objective of the auxiliary self-supervised task in formula (2) becomes:

$$\min \lambda_1 CD(\hat{x}, x) - \lambda_2 CD\left(x', x\right) \tag{3}$$

where λ_1 and λ_2 are hyper-parameters, and CD represents the chamfer distance between two point clouds. Training the transformation network φ_ω and the reconstruction network $f_{enc} \circ f_{h\_aux}$ in such an end-to-end way would enable φ_ω to learn a nonlinear and continuous transformation. Meanwhile, the encoder would also learn more robust features to reconstruct the original point cloud from

Received month dd, yyyy; accepted month dd, yyyy

E-mail: S. Liu:slliu@fudan.edu.cn; Z. Song:zjsong@fudan.edu.cn

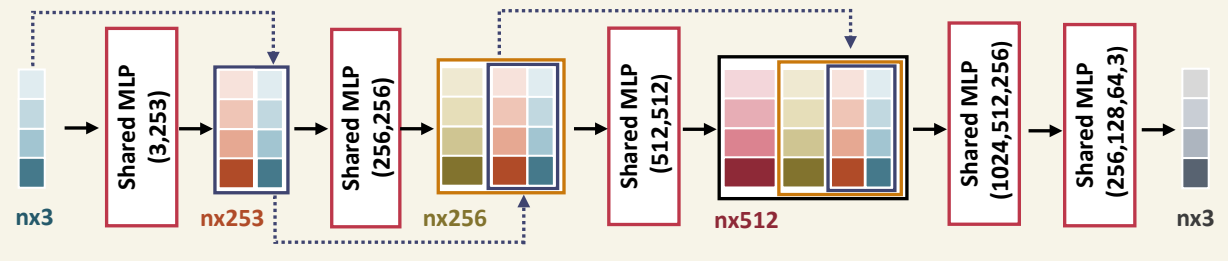

Fig. 1 Point cloud transformation network. The blocks with different colors represent point cloud features, the dashed lines represent skip connections, and the values in the shared MLP module represent the numbers of neurons. Note that the batch normalization layer is applied after each layer of shared MLP, and ReLU is used as the activation function.

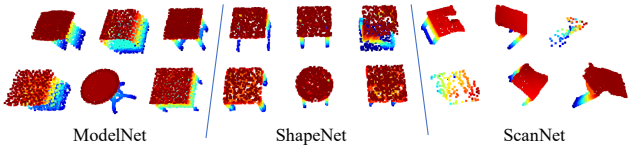

Fig. 2 Examples of point clouds in the Point-DA dataset.

the transformed one, thereby improving the transferability of the learned features and domain adaptability of the encoder.

The transformation network is composed of multiple MLP layers with shared parameters as shown in Fig. 3. Sharing parameters across all points ensures that neighboring points in the input point cloud get similar outputs after transformation so that the learned transformation is continuous and the global semantic information of the point cloud is maintained. Instead of directly outputting the coordinates of the points, we output the displacement of each point through the point cloud transformation network and control the scale of displacement by a shift scale hyperparameter α as follows:

$$x' = \alpha \cdot \varphi_\omega(x) + x \tag{4}$$

1.2 Multi-Region Transformation

Experiments show that transforming the whole point cloud will make it very difficult to reconstruct the original point cloud, which may be due to the loss of most local and contextual information during transformation. To avoid this issue, we introduce a multi-region local point cloud transformation strategy based on our proposed learnable transformation network.

As shown in Figure 4 (c) , we employ a local transformation for the point cloud, so that the encoder can learn features from both the transformed and the untransformed regions. For one point cloud x, multiple transformation networks φ_ω with shared parameters are used to transform parts of the point cloud at different random locations to obtain several different transformed point clouds of the same object, and then the same reconstruction network $f_{enc} \circ f_{h\_aux}$ is applied to reconstruct the original point cloud from the transformed ones. This strategy facilitates the encoder to extract features from different parts of the objects for the reconstruction task.

2 More Visualization Results

In Fig. 3, we visualize the segmentation results while the source domain is faust and the target domain are adobe, mit and scape respectively. The first rows in each subfigure are point clouds with ground truth label and the second rows

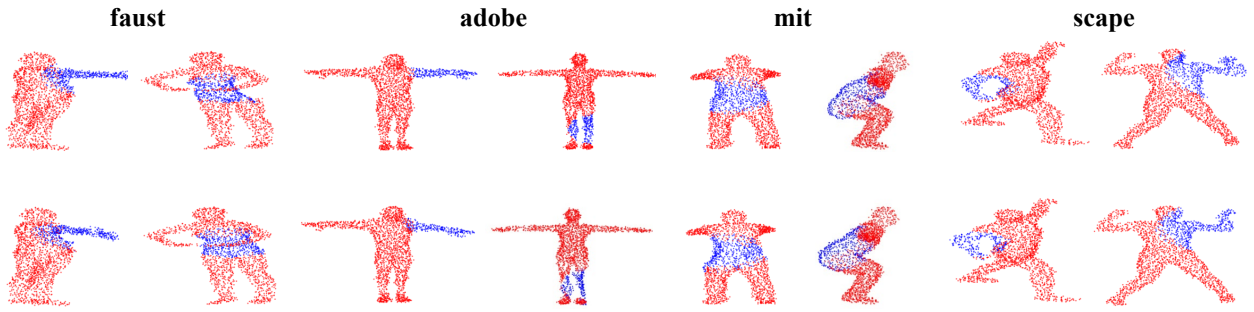

Fig. 3 The visualization of segmentation results while the source domain is faust and the target domain are adobe, mit and scape respectively. The first rows are point clouds with ground truth label and the second rows are point clouds predictions. (a) segmentation result on source domain faust; (b) segmentation result on target domain adobe;(c) segmentation result on mit; (d) segmentation result on scape.

are point cloud predictions. In Fig. 4, we show more visual examples for the learned destructed and reconstructed point clouds in the segmentation scenario. Furthermore, We visualize several examples of original point clouds and the corresponding point clouds transformed by the transformation network φ_ω when $m = 512$ and $\alpha = 0.05$ in Figure 5. We can see that the point clouds undergo continuous nonlinear local transformations, which retain the local semantic information and generate new point clouds with richer shapes than before. Compared with the transformation in [1] that directly removes part of the point cloud and replaces it with random points, we do not destroy the overall distribution of the point cloud, but rather construct more complicated and meaningful samples with richer shapes through the learnable transformation, forcing the encoder to learn robust features for DA.

3 Experiment

3.1 Point Cloud Classification UDA

Dataset. We evaluated our method on the PointDA-10 dataset [3] specifically designed for point cloud DA; this dataset contains 10 shared classes from three widely used point cloud datasets: ModelNet [4], ShapeNet [5] and ScanNet [6], as shown in Figure 2. The point clouds in ModelNet and ShapeNet were sampled from 3D CAD models, while point clouds in ScanNet were sampled from scanned and reconstructed real-world indoor scenes. As in [1], we rotate the ScanNet and ShapeNet models 90 around their Y axis, so that in all three datasets, the upward direction of each model corresponds to the positive direction of Z axis. There are obvious domain gaps between the three datasets, as shown in Figure 2. The point clouds in both ModelNet and ShapeNet are complete surface points of 3D objects, but the objects in the two sets have different shapes and styles, while the point clouds in ScanNet are partial surface points with noise. UDA experiments were performed by choosing one of them as the source domain and another as the target domain, and this resulted in six UDA scenarios, where the performance was measured in terms of classification accuracy on the target domain. We split the official training set with 80% for training and 20% for validation, and used the official test sets for testing.

Optimization Details. Our model was trained on a single NVIDIA V100 GPU based on the deep learning library PyTorch [7]. The ADAM [8] optimizer with a learning rate of 0.001 and a weight decay of 0.0005 was applied under a cosine annealing learning rate scheduler. Early-stop mechanism was applied on validation dataset to avoid over-fitting. The batch size and number of epochs were set as 16 and 150, respectively. During training, we ap-

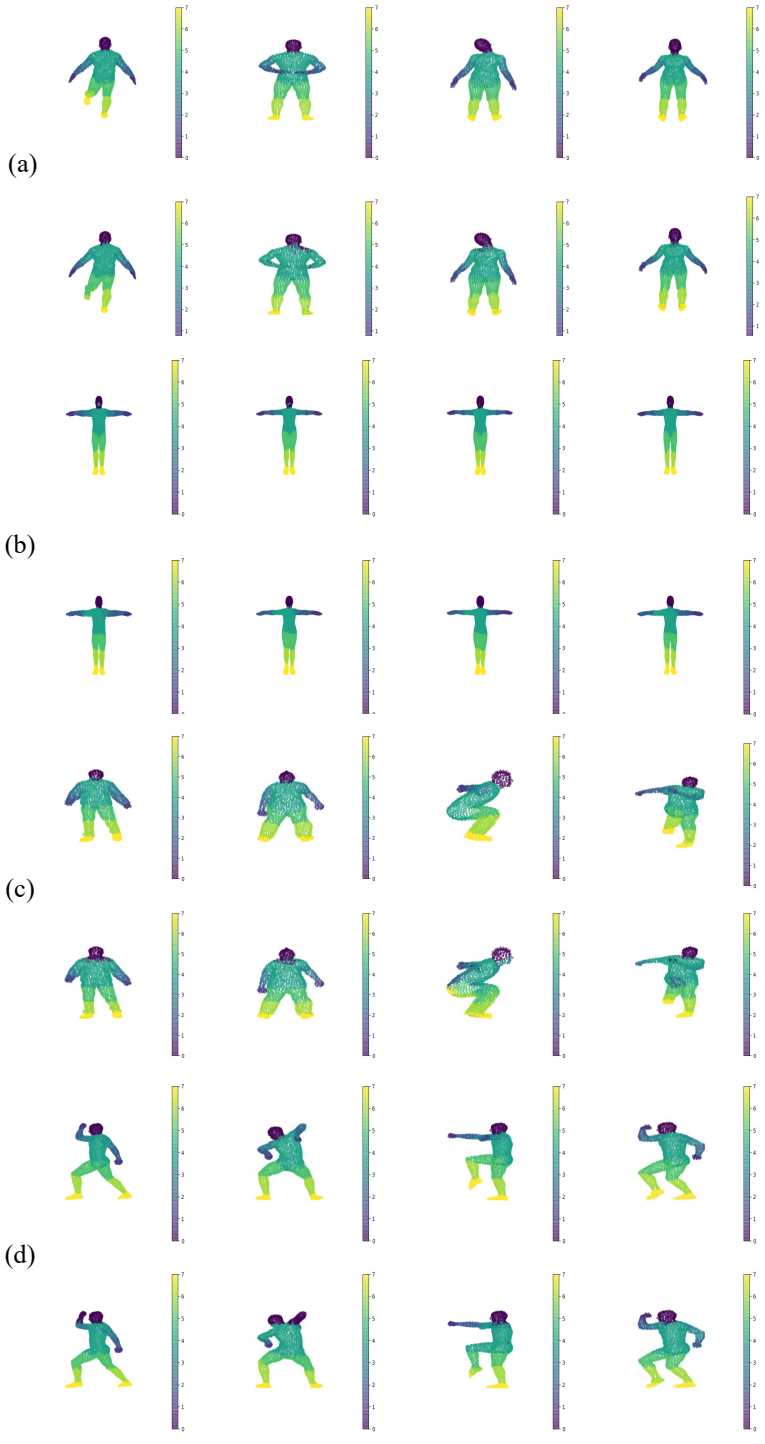

Fig. 4 For each dataset, two samples are visualized, the first row are raw point clouds and the second row are transformed point clouds, the points in transformed area marked in blue.

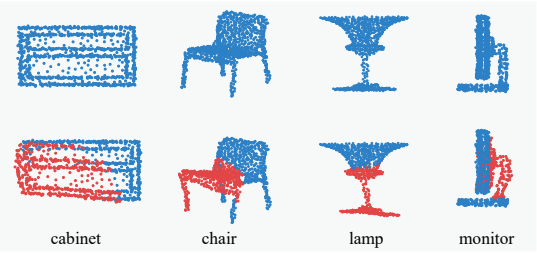

Fig. 5 Examples of original point clouds (upper) and their deformed counterparts (lower) by the learnable transformation. Red points correspond to the deformed parts.

plied rotations about the Z axis and random jittering with standard deviation and clip parameters of 0.01 and 0.02, respectively, for data augmentation. The model with the highest classification accuracy on the source domain validation dataset was preserved to evaluate the performance of that model on the target domain test dataset. We applied a grid search over the shift scale α {0.01, 0.05, 0.1, 0.2} and the number of nearest points k {128, 256, 512, 1024}. In the training of the destruction-reconstruction task, λ_1 and λ_2 are two hyper-parameters determining the degree of destruction and reconstruction. We experimented on the following six combinations of λ_1 and λ_2 {(10, 1), (5, 1), (1, 1), (1, 5), (1, 10), (1, 25)}, and chose the best one. In the multi-region point cloud transformation strategy, we constructed two transformation networks with shared parameters to transform the 512 nearest neighbors of a randomly selected point with a shift scale α of 0.05. Since the point cloud transformation network is easier to train than the reconstruction network, we set λ_1 and λ_2 as 1 and 10, respectively.

Architecture Details. Any point cloud encoding network can be used as the encoder in our method, and we adopted PointNet and DGCNN as the encoder f_{enc} for feature extraction. In PointNet, five 1D convolution layers with kernel size 1 and output size [64, 64, 64, 128, 1024] respectively are utilized to extract point-level features, and two joint transform networks are employed to align input points in the first layer and point features in the third layer. Both transform nets have three 1D convolution layers of size [64, 128, 1024] followed by a max-pooling layer and they have three fully connect layer with neuron size [512, 256, 9] and [512, 256, 4096], respectively, after max-pooling. Finally, a max-pooling layer is applied to aggregate global information from all input points. In DGCNN, we applied the official EdgeConv implementation [9]. Four EdgeConv layers of output size [64, 64, 128, 256] respectively were applied, and a 1D convolution layer with a size of 1024 and max-pooling were used to generate global feature vector. Furthermore, a point cloud transform block was used in the input layer to align an input point set to a canonical space.

The classification head network f_{h_main} was implemented using three fully connected layers with output size [512, 256, 10] respectively (where 10 is the number of classes), and the 1024 dimensional features extracted by the encoder were input to the classification network. Furthermore, a dropout layer with a probability of 0.5 was introduced to the first two layers.

The reconstruction head network f_{h_aux} was implemented by applying four 1D convolution layer with kernel size 1, and the output size of these layers was set as [256, 256, 128, 3], respectively. We employed batch normalization after every convolution layer and a Leaky ReLU with a slope of 0.2.

In the multi-region point cloud transformation strategy, we constructed two transformation networks with shared parameters to transform the 512 nearest neighbors of a randomly selected point with a shift scale α of 0.05. Since the point cloud transformation network is easier to train than the reconstruction network, we set λ_1 and λ_2 as 1 and 10, respectively.

Random Cropping. ModelNet and ShapeNet contain point clouds of complete surfaces of objects, while ScanNet contains scanned and reconstructed real-world point clouds, which are often incomplete surfaces of objects. Due to the shape differences between them, we applied a random cropping strategy as in [10] for data augmentation when ScanNet was chosen as the target domain, where the

Table 1 The classification accuracy (%) on the PointDA-10 dataset with PointNet as the encoder.

Method	ModelNet to ShapeNet	ModelNet to ScanNet	ShapeNet to ModelNet	ShapeNet to ScanNet	ScanNet to ModelNet	ScanNet to ShapeNet	Avg
w/o Adapt	80.2	43.1	**75.8**	40.7	63.2	67.2	61.7
Rotate	81.6	48.2	64.6	49.0	48.0	63.0	59.1
PointDAN	80.2	45.3	71.2	46.9	59.8	66.2	61.6
DefRec	80.0	46.0	68.5	41.7	63.0	68.2	61.2
DefRec+PCM	81.1	50.3	54.3	52.8	54.0	**69.0**	60.3
Resort	81.6	49.7	73.6	41.9	65.9	68.1	63.5
Ours	**82.5**	**52.7**	73.8	**53.8**	**67.4**	**69.0**	**66.5**

Table 2 The classification accuracy (%) on the PointDA-10 dataset with DGCNN as the encoder.

Method	ModelNet to ShapeNet	ModelNet to ScanNet	ShapeNet to ModelNet	ShapeNet to ScanNet	ScanNet to ModelNet	ScanNet to ShapeNet	Avg
w/o Adapt	81.7	42.9	72.2	44.2	67.3	65.1	62.2
Rotate	83.0	51.6	72.5	41.0	67.1	70.3	64.3
PointDAN	83.9	44.8	63.3	45.7	43.6	56.4	56.3
RS	81.5	35.2	71.9	39.8	61.0	63.6	58.8
DefRec	83.3	46.6	79.8	49.9	70.7	64.4	65.8
DefRec+PCM	81.7	51.8	78.6	54.5	73.7	71.1	68.6
GAST	84.8	**59.8**	80.8	56.7	**81.8**	74.9	73.0
Ours	**85.8**	56.3	**81.7**	**59.8**	77.9	**79.7**	**73.5**

point clouds of the source domain were randomly cropped by a plane with a random direction, and 73.5% of the point cloud was retained.

Classification Results. We compared our proposed method with other competitive point cloud UDA methods, including PointDAN [3], DefRec [1], Resort [2], RS [11] and GAST [12]. In addition, we compared with another two baselines methods. The first baseline is to train the main task network with the source domain data and then directly use it on the target domain without adaptation. The second baseline utilizes the same architecture as our proposed method but uses rotation prediction as the self-supervised task, and it is denoted as Rotate. In addition, these UDA methods are encoder-independent, so we compare the results with either PointNet or DGCNN as the encoder to evaluate the generality of the comparing UDA methods. The results with PointNet and DGCNN as the encoder are shown in Table 1 and Table 2, respectively. We reproduced PointDAN's results. Because the code of Resort [2] is not publicly available, we only use the results reported in the original paper with PointNet as the encoder. The results of DefRec and RS are from [1], and PCM refers to an extension with data augmentation adopted in [1].

Our proposed method achieves the highest average classification accuracy with both encoders. In Table 1 with PointNet as the encoder, our method achieves the highest accuracy on five out of six UDA scenarios, and w/o Adapt achieves the highest accuracy on another scenario. In Table 2 with DGCNN as the encoder, our method achieves the highest classification accuracy on four out of six UDA scenarios, while GAST achieves the highest accuracy on two scenarios. With PointNet and DGCNN as the encoder, the average accuracy of our method is 4.8% and 11.3% higher than the baseline without adaptation, which indicates the efficiency of the proposed adaptation method. By comparing the results of the proposed method and Rotate, we can see that the proposed destruction-reconstruction self-supervised task is more effective than the rotation prediction task. By comparing the results of Table 1 and Table 2, we can find that our proposed method is more agnostic to different encoders.

We visualize several examples of original point clouds and the corresponding point clouds transformed by the transformation network φ_ω when $m = 512$ and $\alpha = 0.05$ in Figure 5. We can see that the point clouds undergo continuous nonlinear local transformations, which retain the local semantic information and generate new point clouds with richer shapes than before. Compared with the transformation in [1] that directly removes part of the point cloud and replaces it with random points, we do not destroy the overall distribution of the point cloud, but rather construct more complicated and meaningful samples with richer shapes through the learnable transformation, forcing the encoder to learn robust features for DA.

3.2 Learned Transformation as Data Augmentation

3.3 Point Cloud Segmentation UDA

Dataset and Implementation. We then evaluated our method on PointSegDA dataset proposed in [1] for point cloud segmentation UDA. The dataset contains human body point clouds collected from four datasets: ADOBE, FAUST, MIT and SCAPE. The point clouds in these datasets have different body poses and shapes, but they are all segmented into the same eight parts. UDA experiments were performed by choosing one of them as the source domain and another as the target domain, which resulted in 12 UDA scenarios. The performance was measured in terms of mean Intersection over Union (IoU) on the target domain. We split the training dataset with 80% for training and 20% for validation, and used the official test sets for testing. DGCNN was used as the encoder in this experiment and the segmentation head was implemented using four 1D convolutional layers with size [256, 256 ,128, 8].

Segmentation Results. We compare our method with the baseline without adaptation, RS [11] and DefRec [1]. The results are shown in Table 3. Our method achieved the highest average IoU and the highest IoUs in eight out of

Table 3 The mean IoU on PointSegDA dataset with DGCNN as the encoder.

Method	FAUST to MIT	FAUST to ADOBE	FAUST to SCAPE	MIT to FAUST	MIT to ADOBE	MIT to SCAPE	ADOBE to FAUST	ADOBE to MIT	ADOBE to SCAPE	SCAPE to FAUST	SCAPE to MIT	SCAPE to ADOBE	Avg
w/o Adapt	60.9	78.5	66.5	33.6	26.6	69.9	38.5	31.2	30.0	64.5	**74.1**	68.4	53.6
RS	60.7	78.7	66.9	38.4	59.6	70.4	44.0	30.4	36.6	65.3	70.7	**73.0**	57.9
DefRec	**61.8**	79.7	67.4	40.1	**67.1**	**72.6**	42.5	28.9	32.2	66.2	66.4	72.2	58.1
DefRec+PCM	60.9	78.8	63.6	48.6	48.1	70.1	46.9	33.2	37.6	62.6	66.3	66.5	56.9
Ours	**61.8**	**80.3**	**68.5**	**56.6**	60.8	67.8	**52.3**	**38.6**	**41**	**66.6**	67.4	68.0	**60.8**

Table 4 Ablation study

SSL	Multi	Crop	DGCNN	ModelNet to ShapeNet	ModelNet to ScanNet	ShapeNet to ModelNet	ShapeNet to ScanNet	ScanNet to ModelNet	ScanNet to ShapeNet	Avg
				80.2	43.1	75.8	40.7	63.2	67.2	61.7
√				82.6	48.8	74.0	44.7	63.1	67.9	63.5
√	√			82.5	47.8	73.8	46.2	67.4	69.0	64.5
√	√	√		82.5	52.7	73.8	53.8	67.4	69.0	66.5
√	√		√	82.8	46.7	**81.7**	51.3	72.9	71.7	67.9
√		√	√	80.6	52.2	72.1	50.5	63.1	68.4	64.5
√	√	√	√	**85.8**	56.3	**81.7**	**59.8**	77.9	**79.7**	**73.5**

12 UDA scenarios. Compared with the baseline without adaptation, we can observe that significant improvement is achieved in most scenarios after employing our domain adaption method.

3.4 Ablation Study

To verify the effects of different modules proposed in our method, we conduct a detailed ablation study, and the quantitative results are summarized in Table 4. SSL represents applying a self-supervised task to train the network, Multi represents applying the multi-region transformation strategy, Crop represents applying the random cropping data augmentation process when the target domain is ScanNet, and DGCNN represents using DGCNN instead of PointNet as the encoder. The first row shows the results obtained by the baseline model without adaptation.

From Table 4, we can see that even when applying the proposed self-supervised task alone, our average classification accuracy can reach 63.5%, which is higher than those of the baseline and PointDAN [3]. In this situation, PointNet is the encoder, and the classification accuracy achieved by using the proposed self-supervised network is higher than that obtained using the deformation-based self-supervised task in DefRec [1] and on par with the results in Resort [2]. This result shows that our self-supervised task can indeed handle the domain distribution alignment more elaborately than the other methods, thus achieving better domain adaptation. After adopting the multi-region transformation strategy, the accuracy is further improved to 64.5%. In the case with ScanNet as the target domain, the random cropping augmentation for the source domain data reduces the domain gap between the source domain and the target domain, thus improving the accuracy of ModelNet to ScanNet and ShapeNet to ScanNet by 4.9% and 7.6%, respectively. In this way, the obtained average accuracy (66.5%) becomes the highest among those of all competi-

Table 5 The classification accuracy (%) on the PointDA-10 dataset

Method	ModelNet to ShapeNet	ModelNet to ScanNet	ShapeNet to ModelNet	ShapeNet to ScanNet	ScanNet to ModelNet	ScanNet to ShapeNet	Avg
w/o Adapt	80.2	43.1	**75.8**	40.7	63.2	67.2	61.7
DefRec	81.0	**44.9**	73.0	42.2	61.7	**68.8**	61.9
Ours	**81.2**	43.1	75.1	**43.2**	**66.7**	**68.8**	**63.0**

tive methods with PointNet as the encoder. Finally, when DGCNN is applied as the encoder, our method achieves the state-of-the-art average accuracy of 73.5%.

In addition, we also experimented on using Crop for the DefRec+PCM method, but its performance changes little. Concretely, the classification accuracy for the ModelNet to ScanNet decreased from 55.5% to 54.4%, and the classification accuracy for the ShapeNet to ScanNet increased from 53.4% to 53.7%.

In this work, we propose a learnable self-supervion task, in which a transformation is learned to non-linearly transform a part of a point cloud. Actually, besides being used in the domain adaptation framework, the learned transformation can also be used as a data augmentation strategy. In this section, we conduct the following experiment to study if it is an effective data augmentation strategy and how is its performance when compared to the domain adaptation strategy. In this experiment, PointNet is used as the encoder.

For each of the six domain adaptation scenarios, we do the following experiment. First, we learn a non-linear transformation by using the destruction-reconstruction self-supervised task on on the source domain data. Then, we train a classification network on the source domain data, in which the learned transformation is used as data augmentation. Finally, the learned classification network is used directly on classify the target domain point cloud. We also experiment on using the transformation in [2] as augmentation. The results are shown in Table 5. In most scenarios, our data augmentation method achieves equal or higher accuracy than [2] or without augmentation. This indicates that the transformation does improve the generalization ability of the trained model when it is used as data augmentation. However, when comparing with the results in Table 1, we can see that the domain adaptation strategy is better than the data augmentation strategy (Avg accuracy 66.5% vs. 63.0%).

We compared the training time per iteration under batch size 16 between DefRec and ours model. The results in Table 6 show that the time efficiency of our model is comparable with previous competitive method DefRec under different backbones.

Table 6 The comparison on time efficiency with the competitive method.

	PointNet	DGCNN
DefRec	0.3374	0.5101
Ours	0.4266	0.5253

References

1. Idan Achituve, Haggai Maron, and Gal Chechik. Self-supervised learning for domain adaptation on point clouds. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 123–133, 2021.
2. Antonio Alliegro, Davide Boscaini, and Tatiana Tommasi. Joint supervised and self-supervised learning for 3d real-world challenges. *arXiv preprint arXiv:2004.07392*, 2020.
3. Qin C, You H X, Wang L C, Kuo C J, and Fu Y. Pointdan: A multi-scale 3d domain adaption network for point cloud representation. In *Advances in Neural Information Processing Systems*, pages 7192–7203, 2019.

4. 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.
5. 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.
6. 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.
7. Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.
8. Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
9. 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.
10. Zi Jian Yew and Gim Hee Lee. Rpm-net: Robust point matching using learned features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11824–11833, 2020.
11. Jonathan Sauder and Bjarne Sievers. Self-supervised deep learning on point clouds by reconstructing space. In *Advances in Neural Information Processing Systems*, pages 12962–12972, 2019.
12. Longkun Zou, Hui Tang, Ke Chen, and Kui Jia. Geometry-aware self-training for unsupervised domain adaptation on object point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6403–6412, 2021.