Point Cloud Pre-training with Diffusion Models

Supplementary Material

1. Proof

Calculating the probability distribution $q(X^{t-1}|X^t)$ for the reverse process is hard. However, given X^0, the posterior of the forward diffusion process can be calculated using the following equation:

$$q(X^{t-1}|X^t, X^0)=N(X^{t-1}; \tilde{\mu}_t(X^t, X^0), \tilde{\beta}_t I), \tag{1}$$

$$\tilde{\mu}_t(X^t, X^0)=\frac{1}{\sqrt{\alpha_t}}(X^t-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon),\ \tilde{\beta}_t=\frac{\beta_t(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}. \tag{2}$$

According to Eq. (6) in the main text, the variational lower bound can be divided into three parts:

$$\begin{aligned} &E_q[\underbrace{-logp_\theta(X^0|X^1,c)}_{L_0}+\underbrace{D_{KL}(q(X^T|X^0)||p(X^T))}_{L_T} \\ &+\sum_{t=2}^{T}\underbrace{D_{KL}(q(X^{t-1}|X^t,X^0)||p_\theta(X^{t-1}|X^t,c))}_{L_{t-1}}. \end{aligned} \tag{3}$$

L_T is a constant without parameters that can be ignored. To compute the parameterization of L_{t-1}, following [4], we set the mean $\mu_\theta(X^t, t, c)$ of $p_\theta(X^{t-1}|X^t, c)$ to:

$$\mu_\theta(X^t, c, t)=\frac{1}{\sqrt{\alpha_t}}(X^t-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(X^t, c, t)). \tag{4}$$

We can calculate L_{t-1}:

$$L_{t-1}=\mathbb{E}_q\left[\frac{1}{2\sigma_t^2}||\tilde{\mu}_t(X^t,X^0)-\mu_\theta(X^t,c,t)||^2\right]+C, \tag{5}$$

where C is a parameter-free constant that can be disregarded. By substituting Eq. (2) and Eq. (4) into L_{t-1}:

$$\begin{aligned} L_{t-1} &= \mathbb{E}_{t,X^t,c,\epsilon}\left[\frac{1}{2\sigma_t^2}||\frac{1}{\sqrt{\alpha_t}}(X^t-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon) \right. \\ &\qquad \left. -\frac{1}{\sqrt{\alpha_t}}(X^t-\frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(X^t,c,t))||^2\right] \\ &= \mathbb{E}_{t,X^t,c,\epsilon}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}t)}||\epsilon-\epsilon_\theta(X^t,c,t)||^2\right] \\ &= \mathbb{E}_{t,X^0,c,\epsilon}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}t)}||\epsilon-\epsilon_\theta(\sqrt{\bar{\alpha}_t}X^0+\sqrt{1-\bar{\alpha}_t}\epsilon,c,t)||^2\right], \end{aligned} \tag{6}$$

where $\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}t)}$ is a constant that is unrelated to the loss, and following [4], we can further simplify the training loss:

$$L(\theta)=\mathbb{E}_{t,X^0,c,\epsilon}\left[\|\epsilon-\epsilon_\theta(\sqrt{\bar{\alpha}_t}X^0+\sqrt{1-\bar{\alpha}_t}\epsilon,c,t)\|^2\right]. \tag{7}$$

2. Implementation Details

All experiments are conducted on the RTX 3090 GPU. We describe the details of pre-training and fine-tuning on various tasks.

Pre-training. During pre-training, we adopt the AdamW optimizer with a weight decay of 0.05 and a learning rate of 0.001. We apply the cosine decay schedule to adjust the learning rate. Random scaling and translation are used for data augmentation. Our model is pre-trained for 300 epochs with a batchsize of 128. The T for the diffusion process is set to 2000, and β_t linearly increases from 1e-4 to 1e-2.

Object classification. We use a three-layer MLP with dropout as the classification head. During the fine-tuning process, we sample 2048 points for each point cloud, divide them into 128 point patches. We set the learning rate to 5e-4, and fine-tune for 300 epochs.

Object detection. Unlike MaskPoint [5], which is pre-trained on ScanNet-Medium and loads the weights of both the SA layer and the encoder during fine-tuning. We only load the weights of the transformer encoder pre-trained on ShapeNet [2] during fine-tuning. Following Maskpoint, we set the learning rate to 5e-4 and use the AdamW optimizer with a weight decay of 0.1. The batch size is set to 8.

Indoor semantic segmentation. For a fair comparison, we put all pre-trained transformer encoders within the same codebase and freeze them while fine-tuning the decoder and semantic segmentation head. Due to limited computing resources, we set the batch size to 4 during fine-tuning. The remaining settings followed those used for training PointNeXt [6] from scratch in the original paper.

Outdoor semantic segmentation. During fine-tuning, we load the backbone MinkUNet pre-trained on ShapeNet. And fine-tune the entire network while following the same settings used for training MinkowskiNet [3] from scratch.

Object detection of CAGroup3D with pre-training. We load the weights of the backbone BiResNet, which is pre-trained on ShapeNet using our method. Then, we fine-tune the entire CAGroup3D [8] model using the same settings as those used for training CAGroup3D from scratch. Note that, we utilize the official codebase of CAGroup3D and consider the best-reproduced results as the baseline for comparison.

3. Additional results

Outdoor semantic segmentation results. As shown in Tab. 1, We report the mean IoU(%) and the IoU(%) on SemanticKITTI [1] for all semantic classes for different methods. Our method improves mean IoU and IoU for multiple categories compared to the variant trained from scratch.

Table 1. **Semantic segmentation results on SemanticKITTI val set.** We report the mean IoU(%) and the IoU(%) for all semantic classes.

Methods	mIoU	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic-sign
Cylinder3D [10]	66.1	96.9	54.4	75.9	81.0	67.0	79.3	92.4	0.1	94.6	47.9	82.2	0.1	90.3	57.0	85.9	66.9	69.2	63.6	50.6
SPVCNN [7]	68.5	97.9	59.8	81.1	79.8	80.8	80.0	92.0	0.6	94.2	50.4	81.7	0.6	90.9	63.5	88.0	69.7	74.1	65.8	51.5
RPVNet [9]	68.9	97.9	42.8	87.6	91.2	83.5	78.3	90.2	0.7	95.2	57.1	83.1	0.2	91.0	63.2	87.3	71.4	72.0	64.9	51.5
MinkowskiNet [3]	70.2	97.4	56.1	**84.9**	84.0	79.1	**81.9**	91.4	24.0	94.0	52.2	81.3	0.2	**92.0**	**67.2**	88.4	68.6	74.8	**65.5**	**50.6**
MinkowskiNet+PointDif	**71.3**	**97.5**	**58.8**	84.6	**92.8**	**80.6**	81.4	**92.3**	**30.3**	**94.1**	**56.0**	**81.7**	**0.2**	91.4	65.4	**88.5**	**69.1**	**75.2**	65.0	50.5

Table 2. **Object detection results of CAGroup3D with and without pre-training.** We report the Overall and different category results at $AP_{25}(\%)$ and $AP_{50}(\%)$.

Methods	Metric	Overall	cabinet	bed	chair	sofa	table	door	window	bookshelf	picture	counter	desk	curtain	refrigerator	showercurtrain	toilet	sink	bathtub	garbagebin
CAGroup3D [8]	AP_{25}	73.20	**54.39**	85.78	**95.70**	**91.95**	69.67	67.87	**60.84**	63.71	38.70	73.62	82.12	66.96	**58.32**	75.80	**99.97**	77.85	87.74	66.61
CAGroup3D+PointDif	AP_{25}	**74.14**	53.71	**87.85**	95.46	89.73	**73.01**	**69.36**	59.72	**65.22**	**41.65**	**75.07**	**82.66**	**67.10**	56.27	**79.22**	99.91	**82.27**	**89.55**	**66.69**
CAGroup3D [8]	AP_{50}	60.84	**39.01**	81.51	90.24	**82.75**	65.89	53.47	**36.39**	55.82	25.13	**42.01**	**66.19**	**49.33**	**53.16**	57.73	**96.52**	**53.80**	**86.75**	**59.35**
CAGroup3D+PointDif	AP_{50}	**61.31**	38.47	**82.46**	**91.03**	82.23	**67.09**	**53.88**	34.72	**56.80**	**31.34**	40.02	65.49	48.19	51.40	**70.57**	96.37	52.60	82.33	58.53

Table 3. **Recurrent uniform sampling.** '#Point Clouds' represents the number of unique point clouds in a batch, and '#t' represents the number of time steps t sampled for each point cloud. We report the mean IoU(%) and mean Accuracy(%) on S3DIS.

#Point Clouds	#t	Intervals	Effective Batchsize	mIoU	mAcc
128	2	2	256	69.52	75.46
128	**4**	**4**	512	**70.02**	**77.05**
128	8	8	1024	69.49	76.50

The experimental results also demonstrate that our method performs well on outdoor datasets.

Object detection results of CAGroup3D with and without pre-training. We report the Overall and different category results at $AP_{25}(\%)$ and $AP_{50}(\%)$. From Tab. 2, we observe that pre-training with our method leads to better performance than training CAGroup3D from scratch. Therefore, our pre-training framework can be flexibly applied to various backbones to improve performance.

Recurrent uniform sampling. Keeping the number of unique point clouds in a batch constant, we conduct experiments with 2 and 8 intervals divisions. The results are shown in Tab. 3, our strategy of dividing the 4 intervals and uniform sampling time step t is optimal.

Masking strategy. We report the experimental results for downstream classification and semantic segmentation tasks with different masking strategies. The strategy of block masking involves masking adjacent point patches. From Tab. 4, we observe that random masking performs better than block masking under the same masking ratio (0.8).

Table 4. **Masking strategy.** "Random" refers to Random masking and "Block" refers to Block masking, We report the Overall Accuracy(%) on ScanObjectNN OBJ-BG subset and the mean IoU(%) on S3DIS.

Masking Strategy	Mask Ratio	OBJ-BG	mIoU
Block	0.8	91.91	69.47
Random	0.8	**93.29**	**70.02**

4. Additional Visualization.

S3DIS semantic segmentation visualizations. We provide a qualitative comparison of results for S3DIS semantic segmentation. As shown in Fig. 1, the predictions of our method are closer to the ground truth and less incorrectly segmented than training PointNeXt from scratch and Point-MAE.

5. limitation

Our pre-training method has demonstrated outstanding performance on various 3D real datasets, but its performance is slightly worse on synthetic datasets. We suspect that this is due to the inability of synthetic datasets to fully simulate the complexity of real-world objects, such as the presence of more noise and occlusion in real datasets. Furthermore, the synthetic datasets are relatively simple, and the performance on the synthetic datasets is currently saturated, with only slight improvements from other pre-training methods. Therefore, it is insufficient to demonstrate the performance advantage of the algorithm on the synthetic datasets. In

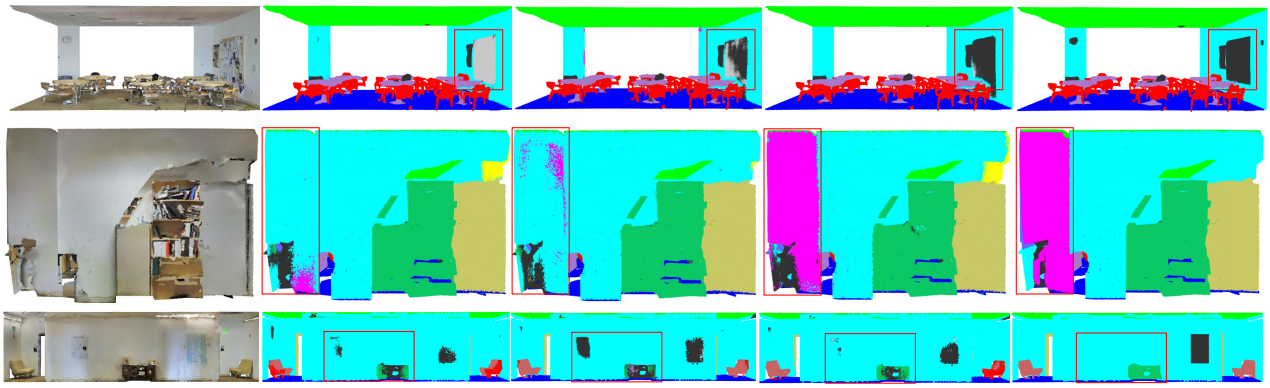

Figure 1. **Qualitative comparison on S3DIS semantic segmentation.** The first column shows the original point cloud input, followed by columns 2-4, which display the segmentation results of PointNeXt, Point-MAE, and our method. The fifth column shows the ground truth.

the future, we will continue exploring and fully exploiting diffusion models' beneficial impact on point cloud pre-training. We also hope that our work will inspire more research on pre-training with diffusion models, contributing to the advancement of the field.

References

[1] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9297–9307, 2019. 1

[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. 1

[3] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3075–3084, 2019. 1, 2

[4] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 1

[5] Haotian Liu, Mu Cai, and Yong Jae Lee. Masked discrimination for self-supervised learning on point clouds. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*, pages 657–675. Springer, 2022. 1

[6] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Advances in Neural Information Processing Systems*, 35:23192–23204, 2022. 1

[7] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII*, pages 685–702. Springer, 2020. 2

[8] Haiyang Wang, Shaocong Dong, Shaoshuai Shi, Aoxue Li, Jianan Li, Zhenguo Li, Liwei Wang, et al. Cagroup3d: Class-aware grouping for 3d object detection on point clouds. *Advances in Neural Information Processing Systems*, 35: 29975–29988, 2022. 1, 2

[9] Jianyun Xu, Ruixiang Zhang, Jian Dou, Yushi Zhu, Jie Sun, and Shiliang Pu. Rpvnet: A deep and efficient range-point-voxel fusion network for lidar point cloud segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16024–16033, 2021. 2

[10] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9939–9948, 2021. 2