Supplementary Material for Progressive Growing of Points with Tree-structured Generators

Hyeontae Son[1]
sonhyuntae@snu.ac.kr
Young Min Kim[2]
youngmin.kim@snu.ac.kr

[1] NAVER LABS
Seongnam, South Korea

[2] Department of ECE
Seoul National University
Seoul, South Korea

This document is the supplementary material of the paper Progressive Growing of Points with Tree-structured Generators. Section 1 contains evaluation metrics of the auto-encoding, completion, and generation of the point clouds. Section 2 contains the training details to apply the progressive growing to the MRTDecoder [2].

1 Evaluation Metrics

Reconstruction. Same evaluation metrics were used for auto-encoding and completion of the point clouds. We used Chamfer Distance (CD), Earth Mover's Distance (EMD) and F-score [5, 6, 7] to evaluate the quality of the reconstructed point cloud P_{out} compared to the ground truth P_{gt}. The Chamfer Distance is defined in Equation 1 of the main paper.

- **EMD** is defined as:

$$\mathrm{EMD}(P_{out}, P_{gt}) = \min_{\phi: P_{out} \to P_{gt}} \frac{1}{|P_{out}|} \sum_{x \in P_{out}} \|x - \phi(x)\|_2 \tag{1}$$

  We used the implementation of [4] for EMD approximation with the same test setting.

- **F-score** is proposed to supplement the weakness of CD and EMD of being heavily influenced by outliers. It is defined as:

$$\text{F-score(d)} = \frac{2 \cdot \mathcal{P}(d) \cdot \mathcal{R}(d)}{\mathcal{P}(d) + \mathcal{R}(d)} \tag{2}$$

  with

$$\mathcal{P}(d) = \frac{1}{|P_{out}|} \sum_{p \in P_{out}} \mathcal{I}[\min_{p' \in P_{gt}} \|p - p'\| < d] \tag{3}$$

$$\mathcal{R}(d) = \frac{1}{|P_{gt}|} \sum_{p' \in P_{gt}} \mathcal{I}[\min_{p \in P_{out}} \|p - p'\| < d] \tag{4}$$

where $\mathcal{P}(d)$ and $\mathcal{R}(d)$ means the precision and recall for a threshold distance d, and $\mathcal{I}[\cdot]$ represents an indicator function. We report F-scores with three different thresholds which are 1%, 2%, and 3% of the average length of edges in the tight axis-aligned bounding box of the ground truth point cloud.

Generation. Let S_{out} be the set of generated point clouds and S_{gt} be the set of ground truth point clouds in the test split. Following the prior convention, we generate S_{out} to be $|S_{out}| = |S_{gt}|$ [3] and then we evaluate the quality of S_{out} with four metrics [3, 8]; JSD, COV, MMD and 1-NNA.

- **JSD** is the Jenshen-Shannon Divergence between two marginal distributions which is defined as:

$$\text{JSD}(S_{out}, S_{gt}) = \frac{1}{2}KL(S_{out}\|M) + \frac{1}{2}KL(S_{gt}\|M) \tag{5}$$

where $M = \frac{1}{2}(S_{out} + S_{gt})$ and KL is the KL-divergence. Each marginal distribution is calculated by assigning the points into the canonical 28^3 voxel grids following [1].

- **COV** is the fraction of the S_{gt} that were matched to point cloud in S_{out}, which is calculated as:

$$\text{COV}(S_{out}, S_{gt}) = \frac{|\{\text{argmin}_{Y \in S_{gt}} D(X,Y) | X \in S_{out}\}|}{|S_{gt}|} \tag{6}$$

where $D(\cdot,\cdot)$ can be either the CD or EMD.

- **MMD** is the average of matched distances to the nearest point cloud in S_{out} for each point cloud in S_{gt}, which is calculated as:

$$\text{MMD}(S_{out}, S_{gt}) = \frac{1}{|S_{gt}|}\sum_{Y \in S_{gt}} \min_{X \in S_{out}} D(X,Y) \tag{7}$$

where $D(\cdot,\cdot)$ can be also either the CD or EMD.

- **1-NNA** is proposed to complement the three metrics above. For each point cloud $X \in S_{out} \bigcup S_{gt}$, let $S_{-X} = S_{out} \bigcup S_{gt} - \{X\}$ and N_X be the nearest point cloud of X in S_{-X}. Then 1-NNA is defined as:

$$\text{1-NNA}(S_{out}, S_{gt}) = \frac{\sum_{X \in S_{out}} \mathcal{I}[N_X \in S_{out}] + \sum_{Y \in S_{gt}} \mathcal{I}[N_Y \in S_{gt}]}{|S_{out}| + |S_{gt}|} \tag{8}$$

2 Progressive Growing for MRTDecoder

MRTDecoder [2] is a multi-resolution version of tree networks. Namely, the output of each layer is not a single tensor, but a list of tensors with different resolutions. Thus, it may be confused to interpret MRTDecoder network as a single tree structure and to generate intermediate point clouds as shown in the Figure 1 in the main paper. This section is for clarifying how to apply the progressive growing to the MRTDecoder.

There can be two possible choices to apply the progressive growing method to MRTDecoder. In other words, there are two methods for creating nodes in each layer from the

Table 1: Auto-encoding results.

Category	Model	CD(↓)	EMD(↓)	F-score(↑)		
				@1%	@2%	@3%
Airplane	MRTDec(PG, ver1)	8.088	1.425	0.337	0.780	0.913
	MRTDec(PG, ver2)	**7.777**	**1.410**	**0.345**	**0.798**	**0.928**
Chair	MRTDec(PG, ver1)	15.009	3.482	0.0811	0.382	0.684
	MRTDec(PG, ver2)	**14.188**	**3.286**	**0.0918**	**0.413**	**0.716**
Car	MRTDec(PG, ver1)	14.059	3.243	0.0647	0.328	0.643
	MRTDec(PG, ver2)	**13.770**	**2.500**	**0.0659**	**0.345**	**0.668**

Table 2: Point cloud completion results.

Dataset	Model	CD(↓)	EMD(↓)	F-Score(↑)		
				@1%	@2%	@3%
PCN	MRTDec(PG, ver1)	10.624	2.477	0.285	0.658	0.826
	MRTDec(PG, ver2)	**10.401**	**2.447**	**0.295**	**0.670**	**0.834**
TopNet	MRTDec(PG, ver1)	23.125	3.893	0.0662	0.313	0.564
	MRTDec(PG, ver2)	**22.830**	**3.770**	**0.0711**	**0.319**	**0.574**

multi-resolution features to generate intermediate point clouds; One is aggregating the feature vectors in the list, and the other is just regarding the most high-resolution feature vectors as the intermediate feature nodes. Aggregating the feature vectors of the first method is implemented as two steps; upsampling the low-resolution features to the same number of as the most high-resolution feature vectors and channel-wise concatenation of them.

We implemented two methods mentioned above, and reported experimental results in Table 1 and 2. We denoted the aggregation method as ver1 and the other method as ver2. We easily found that quantitative results show that ver1 performs worse than ver2 in all metrics. It seems the aggregation of feature vectors in each layer rather confuses the whole training process. In the main paper, we reported the results of the ver2 training method to apply the progressive growing to MRTDecoder.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J. Guibas. Learning Representations and Generative Models for 3D Point Clouds. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, pages 40–49, 2018.

[2] Matheus Gadelha, Rui Wang, and Subhransu Maji. Multiresolution Tree Networks for 3D Point Cloud Processing. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII*.

[3] Le Hui, Rui Xu, Jin Xie, Jianjun Qian, and Jian Yang. Progressive Point Cloud Deconvolution Generation Network. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XV*.

[4] Minghua Liu, Lu Sheng, Sheng Yang, Jing Shao, and Shi-Min Hu. Morphing and Sampling Network for Dense Point Cloud Completion. In *The Thirty-Fourth AAAI Confer-*

ence on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020.

[5] Maxim Tatarchenko, Stephan R. Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What Do Single-view 3D Reconstruction Networks Learn? 2019.

[6] Huikai Wu and Kaiqi Huang. Point Cloud Super Resolution with Adversarial Residual Graph Networks. In *31st British Machine Vision Conference 2020, BMVC 2020, Virtual Event, UK, September 7-10, 2020*.

[7] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Jiageng Mao, Shengping Zhang, and Wenxiu Sun. GRNet: Gridding Residual Network for Dense Point Cloud Completion. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part IX*.

[8] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge J. Belongie, and Bharath Hariharan. PointFlow: 3D Point Cloud Generation With Continuous Normalizing Flows. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*.