Supplemental Material for SeedFormer: Patch Seeds based Point Cloud Completion with Upsample Transformer

Haoran Zhou[1], Yun Cao[2], Wenqing Chu[2], Junwei Zhu[2], Lu Tong[1], Ying Tai[2], and Chengjie Wang[2]

[1]Nanjing University [2]Youtu Lab, Tencent

A Implementation Details

In this section, we provide additional implementation details of the proposed SeedFormer.

Encoder. We use point transformer [8] (SA) and set abstraction [1] (PT) layers to extract features from an input point cloud. The point cloud (with 2,048 points) is downsampled using FPS in each layer of set abstraction. Detailed network architecture is as follows: $SA(C = 128, N = 512) \rightarrow PT(C = 128) \rightarrow SA(C = 256, N = N_p) \rightarrow PT(C = C_p)$. The outputs are $N_p = 128$ patch centers $\mathcal{P}_p$ as well as the corresponding patch features $\mathcal{F}_p$ with $C_p = 256$ channels.

Seed Generator. The inputs to seed generator are the obtained patch centers $\mathcal{P}_p$ and correponding patch features $\mathcal{F}_p$. The implementation of Upsample Transformer (Eq. 1 in the main paper) is slightly different from that in an upsample layer. Since there are no skipped features, we use $\mathcal{F}_p$, applied by linear projections, for both keys and queries in the transformer. Also, the positional encoding of each point is calculated without incorporating seed features. Then, after we obtain seed features $\mathcal{F}$, we concatenate them with a pooled feature from $\mathcal{F}_p$ which is used for encoding global contexts, and apply shared MLPs to produce the seed coordinates $\mathcal{S}$.

Experimental settings. For PCN dataset (with 16,384 points in ground-truth), we set the upsampling rate as $r_1 = 1, r_2 = 4, r_3 = 8$ where the first upsample layer is used to refine the input coarse point cloud $\mathcal{P}_0$. For ShapeNet-34/55 (8,192 points), we set $r_1 = 1, r_2 = 4, r_3 = 4$. Each input point cloud from all datasets contains 2,048 points. For those point clouds with less than 2,048 points, we randomly duplicate input points; for those with more than 2,048 points, we pick a subset as input.

B Ablation Studies

We give more details about ablation networks used in Sec. 4.5 in the main paper. Following the same idea of point generation by local aggregation, we propose two

Table 1: Complexity analysis on PCN dataset evaluated as the number of parameters (Params) and theoretical computational cost (FLOPs). We also report the average CDs of all categories as references.

Methods	Params	FLOPs	CD-Avg
FoldingNet [5]	**2.41M**	27.65G	14.31
PCN [7]	6.84M	14.69G	9.64
GRNet [4]	76.71M	25.88G	8.83
SnowflakeNet [3]	19.32M	10.32G	7.21
point-wise attention	3.12M	**7.87G**	6.85
SeedFormer	3.20M	29.61G	**6.74**

optional generator designs which are both more effective than previous methods. Compared with the default Upsample Transformer, these options can also achieve decent results (Tab. 6 in the main paper) while being more efficient. Similarly, given the features $\{f_i^l\}_{i=1}^{N_l}$ of the input point cloud $\mathcal{P}_l$, we first apply graph convolutions [2] to generate new point features by aggregating local neighborhood information:

$$h_{im} = \max_{j \in \mathcal{N}(i)} \alpha_m(f_j^l). \tag{1}$$

Here, α_m is a feature mapping function (MLPs) and $m = 1, 2, ..., r_l$ corresponds to each of the r_l generation processes in this layer. The output features are $\mathcal{H}_l = \{h_{im} | i = 1, 2, ..., N_l; m = 1, 2, ..., r_l\} \in \mathbb{R}^{r_l N_l \times C}$ with a upsampling rate of r_l. Then, we obtain the new point cloud by learning displacement offsets from the produced features using shared MLPs.

Another option applies transformer structures in point generation. Different from an Upsample Transformer, we adopt point-wise attention which is more efficient in the computations. Similar to Eq. 3 in the main paper, we compute self-attention weights by:

$$\hat{a}_{ijm} = \alpha_m(\beta(q_i^l) - \gamma(k_j^l) + \delta), j \in \mathcal{N}(i). \tag{2}$$

The difference is that $\alpha_m : \mathbb{R}^C \to \mathbb{R}$ outputs one-dimensional weights which are assigned to different points in the neighborhood. Then, we obtain the generated point features by combining weights with duplicated values (Eq. 5 in the main paper). To sum up, the point-wise attention is faster (Sec. C) while it can still outperform state-of-the-art methods.

C Complexity Analysis

We provide a detailed complexity analysis of our method. Tab. 1 reports the theoretical computational cost (FLOPs) and number of parameters for different models. The scores are measured on PCN dataset (16,384 points) with a batch

size of 1. We also provide the overall Chamfer Distances as references. We can see that the model size of SeedFormer is very favorable compared to previous methods. This is owing to our designs of seed generator and upsample layers which process each point with a shared generator. On the other hand, since Upsample Transformer is required to aggregate local points in each generation, the computational cost of SeedFormer is relatively high but it is comparable with GRNet [4] and FoldingNet [5]. We also provide a more efficient version of SeedFormer using a faster transformer structure (point-wise attention) as discussed in Sec. 4.5 of the main manuscript. This model achieves a decent result (6.85) with lower computational cost. In all, our model can offer a pleasant trade-off between cost and performance.

D Additional Experimental Results

Qualitative results on PCN dataset. In Fig. 1, we provide more visual results compared with PCN [7], GRNet [4] and SnowflakeNet [3]. All the results are obtained from their released pretrained models. We can see that SeedFormer performs clearly better than previous methods.

More results on ShapeNet-55/34. We report complete results of our method on ShapeNet-55 in Tab. 3 and results of novel categories on ShapeNet-34 in Tab. 2. The models are tested under three difficulty levels: simple, moderate and hard. For novel objects of ShapeNet-34, we can see that SeedFormer achieves best scores on all categories.

References

1. 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)
2. 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)
3. Xiang, P., Wen, X., Liu, Y.S., Cao, Y.P., Wan, P., Zheng, W., Han, Z.: Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5499–5509 (2021)
4. Xie, H., Yao, H., Zhou, S., Mao, J., Zhang, S., Sun, W.: Grnet: Gridding residual network for dense point cloud completion. In: European Conference on Computer Vision. pp. 365–381. Springer (2020)
5. Yang, Y., Feng, C., Shen, Y., Tian, D.: Foldingnet: Point cloud auto-encoder via deep grid deformation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 206–215 (2018)
6. Yu, X., Rao, Y., Wang, Z., Liu, Z., Lu, J., Zhou, J.: Pointr: Diverse point cloud completion with geometry-aware transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12498–12507 (2021)

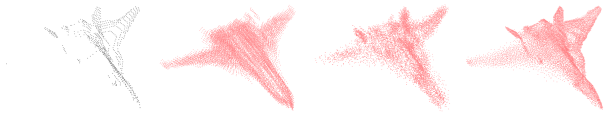
Table 2: Detailed results for novel 21 categories on ShapeNet-34 dataset. *S:*, *M:* and *H:* stand for the simple, moderate and hard difficulty levels.

	PCN [7]			GRNet [4]			PoinTr [6]			SeedFormer		
	S.	M.	H.	S.	M.	H.	S.	M.	H.	S.	M.	H.
bag	2.48	2.46	3.94	1.47	1.88	3.45	0.96	1.34	2.08	**0.49**	**0.82**	**1.45**
basket	2.79	2.51	4.78	1.78	1.94	4.18	1.04	1.40	2.90	**0.60**	**0.85**	**1.98**
birdhouse	3.53	3.47	5.31	1.89	2.34	5.16	1.22	1.79	3.45	**0.72**	**1.19**	**2.31**
bowl	2.66	2.35	3.97	1.77	1.97	3.90	1.05	1.32	2.40	**0.60**	**0.77**	**1.50**
camera	4.84	5.30	8.03	2.31	3.38	7.20	1.63	2.67	4.97	**0.89**	**1.77**	**3.75**
can	1.95	1.89	5.21	1.53	1.80	3.08	0.80	1.17	2.85	**0.56**	**0.89**	**1.57**
cap	7.21	7.14	10.94	3.29	4.87	13.02	1.40	2.74	8.35	**0.50**	**1.34**	**5.19**
keyboard	1.07	1.00	1.23	0.73	0.77	1.11	0.43	0.45	0.63	**0.32**	**0.41**	**0.60**
dishwasher	2.45	2.09	3.53	1.79	1.70	3.27	0.93	1.05	2.04	**0.63**	**0.78**	**1.44**
earphone	7.88	6.59	16.53	4.29	4.16	10.30	2.03	5.10	10.69	**1.18**	**2.78**	**6.71**
helmet	6.15	6.41	9.16	3.06	4.38	10.27	1.86	3.30	6.96	**1.10**	**2.27**	**4.78**
mailbox	2.74	2.68	4.31	1.52	1.90	4.33	1.03	1.47	3.34	**0.56**	**0.99**	**2.06**
microphone	4.36	4.65	8.46	2.29	3.23	8.41	1.25	2.27	5.47	**0.80**	**1.61**	**4.21**
microwaves	2.59	2.35	4.47	1.74	1.81	3.82	1.01	1.18	2.14	**0.64**	**0.83**	**1.69**
pillow	2.09	2.16	3.54	1.43	1.69	3.43	0.92	1.24	2.39	**0.43**	**0.66**	**1.45**
printer	3.28	3.60	5.56	1.82	2.41	5.09	1.18	1.76	3.10	**0.69**	**1.25**	**2.33**
remote	0.95	1.08	1.58	0.82	1.02	1.29	0.44	0.58	0.78	**0.27**	**0.42**	**0.61**
rocket	1.39	1.22	2.01	0.97	0.79	1.60	0.39	0.72	1.39	**0.28**	**0.51**	**1.02**
skateboard	1.97	1.78	2.45	0.93	1.07	1.83	0.52	0.80	1.31	**0.35**	**0.56**	**0.92**
tower	2.37	2.40	4.35	1.35	1.80	3.85	0.82	1.35	2.48	**0.51**	**0.92**	**1.87**
washer	2.77	2.52	4.64	1.83	1.97	5.28	1.04	1.39	2.73	**0.61**	**0.87**	**1.94**
mean	3.22	3.13	5.43	1.84	2.23	4.95	1.05	1.67	3.45	**0.61**	**1.07**	**2.35**

7. Yuan, W., Khot, T., Held, D., Mertz, C., Hebert, M.: Pcn: Point completion network. In: 2018 International Conference on 3D Vision (3DV). pp. 728–737. IEEE (2018)
8. 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)

Table 3: Detailed results on ShapeNet-55 dataset. *S:*, *M:* and *H:* stand for the simple, moderate and hard difficulty levels.

	PCN [7]			GRNet [4]			PoinTr [6]			SeedFormer		
	S.	M.	H.	S.	M.	H.	S.	M.	H.	S.	M.	H.
airplane	0.90	0.89	1.32	0.87	0.87	1.27	0.27	0.38	0.69	**0.23**	**0.35**	**0.61**
trash-bin	2.16	2.18	5.15	1.69	2.01	3.48	0.80	1.15	2.15	**0.73**	**1.08**	**1.94**
bag	2.11	2.04	4.44	1.41	1.70	2.97	0.53	0.74	1.51	**0.43**	**0.67**	**1.28**
basket	2.21	2.10	4.55	1.65	1.84	3.15	0.73	0.88	1.82	**0.65**	**0.83**	**1.54**
bathtub	2.11	2.09	3.94	1.46	1.73	2.73	0.64	0.94	1.68	**0.52**	**0.82**	**1.45**
bed	2.86	3.07	5.54	1.64	2.03	3.70	0.76	1.10	2.26	**0.63**	**0.91**	**1.89**
bench	1.31	1.24	2.14	1.03	1.09	1.71	0.38	0.52	0.94	**0.32**	**0.42**	**0.84**
birdhouse	3.29	3.53	6.69	1.87	2.40	4.71	0.98	1.49	3.13	**0.76**	**1.30**	**2.46**
bookshelf	2.70	2.70	4.61	1.42	1.71	2.78	0.71	1.06	1.93	**0.57**	**0.84**	**1.57**
bottle	1.25	1.43	4.61	1.05	1.44	2.67	0.37	0.74	1.50	**0.31**	**0.63**	**1.21**
bowl	2.05	1.83	3.66	1.60	1.77	2.99	0.68	0.78	1.44	**0.56**	**0.65**	**1.18**
bus	1.20	1.14	2.08	1.06	1.16	1.48	0.42	0.55	0.79	**0.42**	**0.55**	**0.73**
cabinet	1.60	1.49	3.47	1.27	1.41	2.09	**0.55**	**0.66**	1.16	0.57	0.69	**1.05**
camera	4.05	4.54	8.27	2.14	3.15	6.09	1.10	2.03	4.34	**0.83**	**1.68**	**3.45**
can	2.02	2.28	6.48	1.58	2.11	3.81	0.68	1.19	2.14	**0.58**	**1.03**	**1.79**
cap	1.82	1.76	4.20	1.17	1.37	3.05	0.46	0.62	1.64	**0.33**	**0.45**	**1.18**
car	1.48	1.47	2.60	1.29	1.48	2.14	**0.64**	0.86	1.25	0.65	**0.86**	**1.17**
cellphone	0.80	0.79	1.71	0.82	0.91	1.18	0.32	**0.39**	0.60	**0.31**	0.40	**0.54**
chair	1.70	1.81	3.34	1.24	1.56	2.73	0.49	0.74	1.63	**0.41**	**0.65**	**1.38**
clock	2.10	2.01	3.98	1.46	1.66	2.67	0.62	0.84	1.65	**0.53**	**0.74**	**1.35**
keyboard	0.82	0.82	1.04	0.74	0.81	1.09	0.30	0.39	**0.45**	**0.28**	**0.36**	**0.45**
dishwasher	1.93	1.66	4.39	1.43	1.59	2.53	**0.55**	0.69	1.42	0.56	**0.69**	**1.30**
display	1.56	1.66	3.26	1.13	1.38	2.29	0.48	0.67	1.33	**0.39**	**0.59**	**1.10**
earphone	3.13	2.94	7.56	1.78	2.18	5.33	0.81	1.38	3.78	**0.64**	**1.04**	**2.75**
faucet	3.21	3.48	7.52	1.81	2.32	4.91	0.71	1.42	3.49	**0.55**	**1.15**	**2.63**
filecabinet	2.02	1.97	4.14	1.46	1.71	2.89	**0.63**	**0.84**	1.69	**0.63**	**0.84**	**1.49**
guitar	0.42	0.38	1.23	0.44	0.48	0.76	0.14	0.21	0.42	**0.13**	**0.19**	**0.32**
helmet	3.76	4.18	7.53	2.33	3.18	6.03	0.99	1.93	4.22	**0.79**	**1.52**	**3.61**
jar	2.57	2.82	6.00	1.72	2.37	4.37	0.77	1.33	2.87	**0.63**	**1.13**	**2.36**
knife	0.94	0.62	1.37	0.72	0.66	0.96	0.20	0.33	0.56	**0.15**	**0.28**	**0.45**
lamp	3.10	3.45	7.02	1.68	2.43	5.17	0.64	1.40	3.58	**0.45**	**1.06**	**2.67**
laptop	0.75	0.79	1.59	0.83	0.87	1.28	**0.32**	**0.34**	0.60	**0.32**	0.37	**0.55**
loudspeaker	2.50	2.45	5.08	1.75	2.08	3.45	0.78	1.16	2.17	**0.67**	**1.01**	**1.80**
mailbox	1.66	1.74	5.18	1.15	1.59	3.42	0.39	0.78	2.56	**0.30**	**0.67**	**2.04**
microphone	3.44	3.90	8.52	2.09	2.76	5.70	0.70	1.66	4.48	**0.62**	**1.61**	**3.66**
microwaves	2.20	2.01	4.65	1.51	1.72	2.76	0.67	0.83	1.82	**0.63**	**0.79**	**1.47**
motorbike	2.03	2.01	3.13	1.38	1.52	2.26	0.75	1.10	1.92	**0.68**	**0.96**	**1.44**
mug	2.45	2.48	5.17	1.75	2.16	3.79	0.91	1.17	2.35	**0.79**	**1.03**	**2.06**
piano	2.64	2.74	4.83	1.53	1.82	3.21	0.76	1.06	2.23	**0.62**	**0.87**	**1.79**
pillow	1.85	1.81	3.68	1.42	1.67	3.04	0.61	0.82	1.56	**0.48**	**0.75**	**1.41**
pistol	1.25	1.17	2.65	1.11	1.06	1.76	0.43	0.66	1.30	**0.37**	**0.56**	**0.96**
flowerpot	3.32	3.39	6.04	2.02	2.48	4.19	1.01	1.51	2.77	**0.93**	**1.30**	**2.32**
printer	2.90	3.19	5.84	1.56	2.38	4.24	0.73	1.21	2.47	**0.58**	**1.11**	**2.13**
remote	0.99	0.97	2.04	0.89	1.05	1.29	0.36	0.53	0.71	**0.29**	**0.46**	**0.62**
rifle	0.98	0.80	1.31	0.83	0.77	1.16	0.30	0.45	0.79	**0.27**	**0.41**	**0.66**
rocket	1.05	1.04	1.87	0.78	0.92	1.44	0.23	0.48	0.99	**0.21**	**0.46**	**0.83**
skateboard	1.04	0.94	1.68	0.82	0.87	1.24	0.28	0.38	0.62	**0.23**	**0.32**	**0.62**
sofa	1.65	1.61	2.92	1.35	1.45	2.32	0.56	0.67	1.14	**0.50**	**0.62**	**1.02**
stove	2.07	2.02	4.72	1.46	1.72	3.22	0.63	0.92	1.73	**0.59**	**0.87**	**1.49**
table	1.56	1.50	3.36	1.15	1.33	2.33	0.46	0.64	1.31	**0.41**	**0.58**	**1.18**
telephone	0.80	0.80	1.67	0.81	0.89	1.18	**0.31**	**0.38**	0.59	**0.31**	0.39	**0.55**
tower	1.91	1.97	4.47	1.26	1.69	3.06	0.55	0.90	1.95	**0.47**	**0.84**	**1.65**
train	1.50	1.41	2.37	1.09	1.14	1.61	**0.50**	0.70	1.12	0.51	**0.66**	**1.01**
watercraft	1.46	1.39	2.40	1.09	1.12	1.65	0.41	0.62	1.07	**0.35**	**0.56**	**0.92**
washer	2.42	2.31	6.08	1.72	2.05	4.19	0.75	1.06	2.44	**0.64**	**0.91**	**2.04**
mean	1.96	1.98	4.09	1.35	1.63	2.86	0.58	0.88	1.80	**0.50**	**0.77**	**1.49**



(a) Input (b) PCN (c) GRNet (d) Snowflake (e) Ours (f) GT

Fig. 1: Visual comparisons on PCN dataset.