Point Transformer — Supplementary Material

Hengshuang Zhao[1,2] Li Jiang[3] Jiaya Jia[3] Philip Torr[1] Vladlen Koltun[4]
[1]University of Oxford [2]The University of Hong Kong
[3]The Chinese University of Hong Kong [4]Intel Labs

More detailed results. In Table 1, we present the detailed comparison of the semantic segmentation results on the S3DIS dataset [1], under the 6-fold cross-validation setting. We get the highest mIoU as 73.5%, outperforming previous approaches (e.g., RandLA-Net [2] and KPConv [7]) by a large margin. For most of the categories (like wall, column, table, etc.), our method gets the best accuracy. We will release all the implementation details and trained models to the community soon.

More visual results. Figure 1 and Figure 2 show more qualitative results produced by Point Transformer on the S3DIS [1] and ModelNet40 [8] datasets, respectively. From Figure 1, we can see that our Point Transformer model is able to parse complex 3D scenes with many objects, as well as the "stuff" (like wall, floor, etc.) surrounding them. From Figure 2, we can see that the retrieved shapes are very similar to the input query. This demonstrates the effectiveness and generality of Point Transformer.

Method	OA	mAcc	mIoU	ceiling	floor	wall	beam	column	window	door	table	chair	sofa	bookcase	board	clutter
PointNet [6]	78.5	66.2	47.6	88.0	88.7	69.3	42.4	23.1	47.5	51.6	54.1	42.0	9.6	38.2	29.4	35.2
RSNet [3]	–	66.5	56.5	92.5	92.8	78.6	32.8	34.4	51.6	68.1	60.1	59.7	50.2	16.4	44.9	52.0
SPGraph [4]	85.5	73.0	62.1	89.9	95.1	76.4	62.8	47.1	55.3	68.4	73.5	69.2	63.2	45.9	8.7	52.9
PAT [9]	–	76.5	64.3	93.0	98.4	73.5	58.5	38.9	77.4	67.7	62.7	67.3	30.6	59.6	66.6	41.4
PointCNN [5]	88.1	75.6	65.4	94.8	97.3	75.8	63.3	51.7	58.4	57.2	71.6	69.1	39.1	61.2	52.2	58.6
PointWeb [11]	87.3	76.2	66.7	93.5	94.2	80.8	52.4	41.3	64.9	68.1	71.4	67.1	50.3	62.7	62.2	58.5
ShellNet [10]	87.1	–	66.8	90.2	93.6	79.9	60.4	44.1	64.9	52.9	71.6	84.7	53.8	64.6	48.6	59.4
RandLA-Net [7]	88.0	82.0	70.0	93.1	96.1	80.6	62.4	48.0	64.4	69.4	69.4	76.4	60.0	64.2	65.9	60.1
KPConv [7]	–	79.1	70.6	93.6	92.4	83.1	63.9	54.3	66.1	76.6	64.0	57.8	74.9	69.3	61.3	60.3
PointTransformer	**90.2**	**81.9**	**73.5**	94.3	97.5	84.7	55.6	58.1	66.1	78.2	77.6	74.1	67.3	71.2	65.7	64.8

Table 1. Semantic segmentation results on S3DIS, evaluated with 6-fold cross-validation.

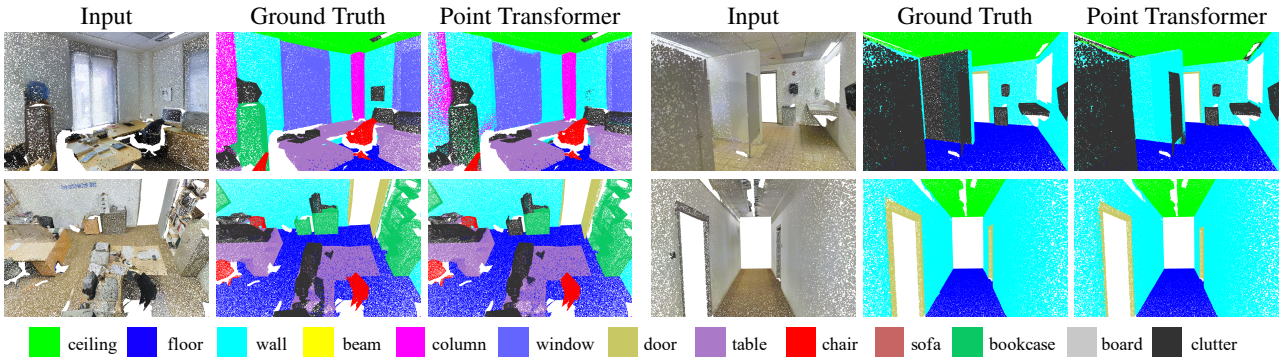

Figure 1. Visualization of semantic segmentation results on S3DIS.

Figure 2. Visualization of shape retrieval results on ModelNet40. The leftmost column shows the input query and the other columns show the retrieved models.

Inference time and memory. We test the inference time and memory consumption of Point Transformer on one Quadro RTX 6000, with different size of input point clouds. The inference time and memory consumption are 44ms/86ms/222ms/719ms and 1702M/2064M/2800M/4266M for 10k/20k/40k/80k input points respectively and they can be further reduced with optimized implementation.

kNN efficiency. For kNN, when constructing local point cloud regions, previous methods like KPConv [7] and RandLA-Net [2] use precomputed kNN indices, which limits the flexibility of the overall framework. In our architecture, we implement a high-efficiency solution for kNN using the heap sort algorithm. We test the running time of our efficient implementation on one Quadro RTX 6000; the results are listed in Table 2. We also test some naive implementations, the running time is 56ms/228ms when given 10k/20k points, which is much slower than ours. Moreover, naive implementations run out of memory when given larger point clouds.

#pts	k=8	k=16	k=32	k=64	k=128	k=256
10k	2	2	5	10	17	21
20k	3	5	8	23	43	49
40k	8	12	26	71	127	144
80k	23	37	82	198	356	399
100k	32	46	99	248	445	494
200k	104	125	225	545	992	1091
500k	639	695	867	1589	2865	3143
1m	2496	2648	2949	4087	6362	6878

Table 2. High-efficiency kNN implementation with heap sort algorithm. The leftmost column stands for the number of points and the topmost row specifies the number of nearest neighbors. The reported running time is in milliseconds.

References

[1] Iro Armeni, Ozan Sener, Amir R. Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3D semantic parsing of large-scale indoor spaces. In *CVPR*, 2016. 1

[2] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *CVPR*, 2020. 1, 2

[3] Qiangui Huang, Weiyue Wang, and Ulrich Neumann. Recurrent slice networks for 3d segmentation of point clouds. In *CVPR*, 2018. 1

[4] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *CVPR*, 2018. 1

[5] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on $\mathcal{X}$-transformed points. In *NIPS*, 2018. 1

[6] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 1

[7] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, 2019. 1, 2

[8] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015. 1

[9] Jiancheng Yang, Qiang Zhang, Bingbing Ni, Linguo Li, Jinxian Liu, Mengdie Zhou, and Qi Tian. Modeling point clouds with self-attention and gumbel subset sampling. In *CVPR*, 2019. 1

[10] Zhiyuan Zhang, Binh-Son Hua, and Sai-Kit Yeung. Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics. In *ICCV*, 2019. 1

[11] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. PointWeb: Enhancing local neighborhood features for point cloud processing. In *CVPR*, 2019. 1