Supplementary Material: Interpolation-Aware Padding for 3D Sparse Convolutional Neural Networks

Yu-Qi Yang[1,2] Peng-Shuai Wang[2] Yang Liu[2]
[1]Tsinghua University [2]Microsoft Research Asia
yangyq18@mails.tsinghua.edu.cn {penwan,yangliu}@microsoft.com

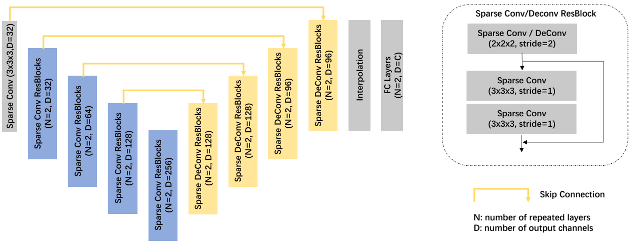
Sparse Conv (3x3x3,D=32)
Sparse Conv ResBlocks (N=2, D=32)
Sparse Conv ResBlocks (N=2, D=64)
Sparse Conv ResBlocks (N=2, D=128)
Sparse Conv ResBlocks (N=2, D=256)
Sparse DeConv ResBlocks (N=2, D=128)
Sparse DeConv ResBlocks (N=2, D=128)
Sparse DeConv ResBlocks (N=2, D=96)
Sparse DeConv ResBlocks (N=2, D=96)
Interpolation
FC Layers (N=2, D=C)
Sparse Conv/Deconv ResBlock
Sparse Conv / DeConv (2x2x2, stride=2)
Sparse Conv (3x3x3, stride=1)
Sparse Conv (3x3x3, stride=1)
Skip Connection
N: number of repeated layers
D: number of output channels

Figure 1: Network structure for PartNet Segmentation and Semantic KITTI. The numbers of repeated resblocks are [2,2,2,2,2,2,2,2].

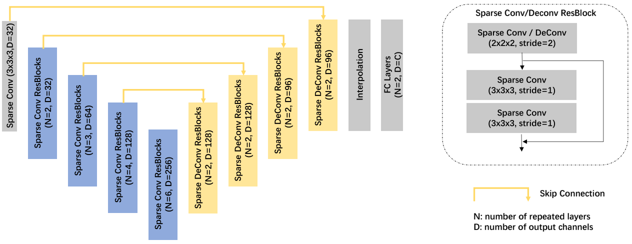
Sparse Conv (3x3x3,D=32)
Sparse Conv ResBlocks (N=2, D=32)
Sparse Conv ResBlocks (N=3, D=64)
Sparse Conv ResBlocks (N=4, D=128)
Sparse Conv ResBlocks (N=6, D=256)
Sparse DeConv ResBlocks (N=2, D=128)
Sparse DeConv ResBlocks (N=2, D=128)
Sparse DeConv ResBlocks (N=2, D=96)
Sparse DeConv ResBlocks (N=2, D=96)
Interpolation
FC Layers (N=2, D=C)
Sparse Conv/Deconv ResBlock
Sparse Conv / DeConv (2x2x2, stride=2)
Sparse Conv (3x3x3, stride=1)
Sparse Conv (3x3x3, stride=1)
Skip Connection
N: number of repeated layers
D: number of output channels

Figure 2: Network structure for Scannet segmentation. The numbers of repeated resblocks are [2,3,4,6,2,2,2,2].

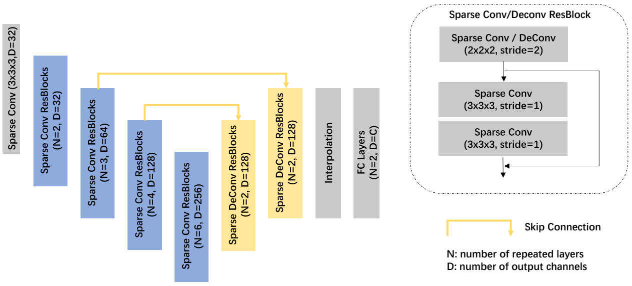
Sparse Conv (3x3x3,D=32)
Sparse Conv ResBlocks (N=2, D=32)
Sparse Conv ResBlocks (N=3, D=64)
Sparse Conv ResBlocks (N=4, D=128)
Sparse Conv ResBlocks (N=6, D=256)
Sparse DeConv ResBlocks (N=2, D=128)
Sparse DeConv ResBlocks (N=2, D=128)
Interpolation
FC Layers (N=2, D=C)
Sparse Conv/Deconv ResBlock
Sparse Conv / DeConv (2x2x2, stride=2)
Sparse Conv (3x3x3, stride=1)
Sparse Conv (3x3x3, stride=1)
Skip Connection
N: number of repeated layers
D: number of output channels

Figure 3: Network structure for Scannet segmentation when $s_{out} = 8\,\mathrm{cm}$. The numbers of repeated resblocks are [2,3,4,6,2,2].

1. Detailed Segmentation Results on PartNet

The part IOUs of all categories are listed in Tab. 1. The results shown in the table are the average metrics and mean deviations of three rounds. On each round, all the networks are initialized with the same parameters.

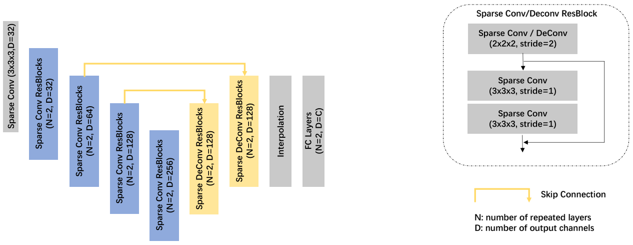
Sparse Conv (3x3x3,D=32)
Sparse Conv ResBlocks (N=2, D=32)
Sparse Conv ResBlocks (N=2, D=64)
Sparse Conv ResBlocks (N=2, D=128)
Sparse Conv ResBlocks (N=2, D=256)
Sparse DeConv ResBlocks (N=2, D=128)
Sparse DeConv ResBlocks (N=2, D=128)
Interpolation
FC Layers (N=2, D=C)
Sparse Conv/Deconv ResBlock
Sparse Conv / DeConv (2x2x2, stride=2)
Sparse Conv (3x3x3, stride=1)
Sparse Conv (3x3x3, stride=1)
Skip Connection
N: number of repeated layers
D: number of output channels

Figure 4: Network structure for Semantic KITTI when $s_{out} = 20\,\mathrm{cm}$. It is also the network for Scannet detection, $s_{out} = 8\,\mathrm{cm}$. The numbers of repeated resblocks are [2,2,2,2,2,2].

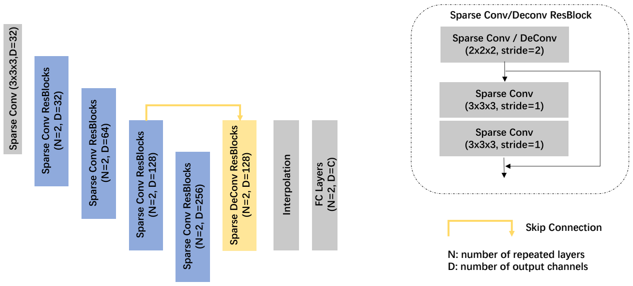
Sparse Conv (3x3x3,D=32)
Sparse Conv ResBlocks (N=2, D=32)
Sparse Conv ResBlocks (N=2, D=64)
Sparse Conv ResBlocks (N=2, D=128)
Sparse Conv ResBlocks (N=2, D=256)
Sparse DeConv ResBlocks (N=2, D=128)
Interpolation
FC Layers (N=2, D=C)
Sparse Conv/Deconv ResBlock
Sparse Conv / DeConv (2x2x2, stride=2)
Sparse Conv (3x3x3, stride=1)
Sparse Conv (3x3x3, stride=1)
Skip Connection
N: number of repeated layers
D: number of output channels

Figure 5: Network structure for Semantic KITTI when $s_{out} = 40\,\mathrm{cm}$. The numbers of repeated resblocks are [2,2,2,2,2].

2. 3D Object Detection with a Stronger Baseline

H3DNet [5] is one of the state-of-the-art methods on 3D object detection of ScanNet. It uses the PointNet++[3] as the backbone to extract point features and predicts geometric primitives with these features, then generates object proposals. The results of H3DNet when using one PointNet++ is reported in Tab. 2. After replacing the PointNet++ by a sparse-voxel-based U-Net with our interpolation-aware sparse padding and the trilinear interpolation, we can achieve 47.1mAP@0.5, and the performance gain compared with H3DNet is 3.4. We also observe that compared with our own results based on VoteNet (see the 3rd row in Tab. 2), the performance is also greatly improved, which indicates that

Group	Pad.	Interp.	$\mathrm{S_{out}}$	mIOU	Chair	Lamp	Stora	Table
(1)	Zero	Near	64^3	40.5 ± 0.2	46.2 ± 0.2	24.2 ± 0.3	53.7 ± 0.3	37.8 ± 0.4
	Octree	Near	64^3	40.0 ± 0.0	46.3 ± 0.2	23.7 ± 0.1	53.4 ± 0.2	36.7 ± 0.2
	Ring	Near	64^3	$\mathbf{40.9} \pm 0.0$	$\mathbf{47.3} \pm 0.2$	$\mathbf{24.4} \pm 0.1$	$\mathbf{53.8} \pm 0.1$	$\mathbf{38.2} \pm 0.1$
	Interp	Near	64^3	40.6 ± 0.1	46.9 ± 0.2	24.1 ± 0.2	53.4 ± 0.2	37.9 ± 0.5
(2)	Zero	Linear	64^3	41.5 ± 0.0	46.8 ± 0.3	28.0 ± 0.1	53.3 ± 0.5	37.9 ± 0.1
	Octree	Linear	64^3	41.4 ± 0.0	47.3 ± 0.3	27.2 ± 0.4	53.6 ± 0.5	37.6 ± 0.1
	Ring	Linear	64^3	$\mathbf{42.7} \pm 0.3$	$\mathbf{48.0} \pm 0.4$	$\mathbf{28.7} \pm 0.4$	$\mathbf{54.4} \pm 0.2$	$\mathbf{39.7} \pm 0.4$
	Interp	Linear	64^3	42.3 ± 0.3	47.6 ± 0.2	28.6 ± 0.5	54.3 ± 0.1	38.8 ± 0.5
(3)	Zero	Near	32^3	38.1 ± 0.2	45.1 ± 0.2	23.0 ± 0.4	47.9 ± 0.1	36.3 ± 0.2
	Interp	Linear	32^3	$\mathbf{40.1} \pm 0.1$	46.2 ± 0.1	27.3 ± 0.3	49.7 ± 0.5	37.1 ± 0.2

Table 1: Quality statistics of fine-grained part segmentation on four PartNet categories. *Pad.* is the sparse padding type, *Int.* is the sparse interpolation type, *mIoU* is the average part IoU

Network	Pad.	Int	mAP@0.25	mAP@0.5
VoteNet [2]	-	-	57.8 ± 0.6	34.7 ± 0.4
MinkNet [1]	Zero	Near	58.7 ± 0.5	37.9 ± 0.6
Ours	Interp	Linear	$\mathbf{60.7} \pm 0.8$	$\mathbf{41.4} \pm 0.6$
H3DNet [5]	-	-	64.0 ± 0.7	44.7 ± 0.8
MinkNet [1]	Zero	Near	63.6 ± 0.4	45.8 ± 0.4
Ours	Interp	Linear	$\mathbf{64.4} \pm 0.2$	$\mathbf{47.1} \pm 0.3$

Table 2: Quality statistics of instance detection on the ScanNet validation set.

with a stronger baseline, we can achieve get better results.

3. Network Structures

3.1. PartNet Segmentation

We use the U-Net structure with five levels of domain resolution. The finest grid resolution in the network is set to 64^3. The structure is shown in Fig. 1. The numbers of repeated resblocks are [2,2,2,2,2,2,2,2].

3.2. ScanNet Segmentation

For the ScanNet Segmentation task, We use the same U-Net structure in [1] with five levels of domain resolution. The finest grid resolution in the network is set to 2 cm. The structure is shown in Fig. 2. The numbers of repeated resblocks are [2,3,4,6,2,2,2,2]. When the s_{out} is set to 8 cm, we remove two high resolution layers in the decoder, and interpolate the feature from the 3rd level. The network structure is shown in Fig. 3.

3.3. Semantic KITTI Segmentation

For Semantic KITTI Segmentation, we use the same U-Net structure as [4]. It is same with the network shown in Fig.1. The finest grid resolution in the network is set to 5 cm. When the s_{out} is set to 20 cm or 40 cm, we also remove the corresponding layers in the decoder, the network structures are shown in Fig. 4 and Fig. 5.

3.4. ScanNet Detection

For the ScanNet detection task, we follow the pipeline of VoteNet [2], and only replace the PointNet++ with U-Net based on the sparse-convolution. The finest grid resolution in the network is set to 2 cm. And the s_{out} is fixed as 8 cm in this task. The U-Net structure is the same as the network shown in Fig. 5. The numbers of repeated resblocks are [2,2,2,2,2,2].

4. Advantage with 1-ring padding

Though the performance of the 1-ring padding is slightly better than the interpolation-aware padding on PartNet, its memory and computational cost are much higher, especially on large-scale datasets such as ScanNet and KITTI. The comparison of memory consumption with batch size as 1 is summarized in Table 3. In practice, we train the network with a batch size of at least 4 due to the requirement of Batch Normalization, otherwise, the performance may decrease greatly. And if the 1-ring padding is used, the network runs out of memory even on V100 GPU with 32GB memory on ScanNet and KITTI, despite its potential performance improvement. Our interpolation-aware padding is more practical and provides clear improvements over previous approaches.

Padding scheme	ScanNet	KITTI
1-ring padding	8.9G	12.9G
Interpolation-aware padding	6.7G	6.4G

Table 3: The memory cost comparison with batch size 1.

5. Source code

Our source code is available on our project homepage: *https://yukichiii.github.io/project/padding.html*, feel free to follow the instruction in ReadMe to use our padding scheme.

References

[1] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4D spatio-temporal ConvNets: Minkowski convolutional neural networks. In *CVPR*, 2019. 2

[2] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep Hough Voting for 3D Object Detection in Point Clouds. In *ICCV*, 2019. 2

[3] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 1

[4] 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 *ECCV*, 2020. 2

[5] Zaiwei Zhang, Bo Sun, Haitao Yang, and Qixing Huang. H3dnet: 3d object detection using hybrid geometric primitives. In *European Conference on Computer Vision*, pages 311–329. Springer, 2020. 1, 2