

Efficient Semantic Scene Completion Network with Spatial Group Convolution

Jiahui Zhang^{*1}, Hao Zhao^{*2}, Anbang Yao^{✉3}, Yurong Chen³, Li Zhang², and
Hongen Liao^{✉1}

¹ Department of Biomedical Engineering, Tsinghua University

² Department of Electronic Engineering, Tsinghua University

³ Intel Labs China

{jiahui-z15@mails., zhao-h13@mails., chinazhangli@mail., liao@}tsinghua.edu.cn
{anbang.yao, yurong.chen}@intel.com

Abstract. We introduce Spatial Group Convolution (SGC) for accelerating the computation of 3D dense prediction tasks. SGC is orthogonal to group convolution, which works on spatial dimensions rather than feature channel dimension. It divides input voxels into different groups, then conducts 3D sparse convolution on these separated groups. As only valid voxels are considered when performing convolution, computation can be significantly reduced with a slight loss of accuracy. The proposed operations are validated on semantic scene completion task, which aims to predict a complete 3D volume with semantic labels from a single depth image. With SGC, we further present an efficient 3D sparse convolutional network, which harnesses a multiscale architecture and a coarse-to-fine prediction strategy. Evaluations are conducted on the SUNCG dataset, achieving state-of-the-art performance and fast speed.

Keywords: Spatial Group Convolution, Sparse Convolutional Network, Efficient Neural Network, Semantic Scene Completion

1 Introduction

3D shape processing has attracted increased attention recently, because large scale 3D datasets and deep learning based methods open new opportunities for understanding and synthesizing 3D data, such as segmentation and shape completion. These 3D dense prediction tasks are quite useful for many applications. For example, robots need semantic information to understand the world, while knowing complete scene geometry can help them to grasp objects [43] and avoid obstacles. However, it is not a trivial task to adopt 3D Convolutional Neural Network (CNN) by just adding one dimension to 2D CNN. Dense 3D CNN methods [45,30] face the problem of cubic growth of computational and memory requirements with the increase of voxel resolution.

But meanwhile, we observe that 3D data has some attractive characteristics, which inspire us to build efficient 3D CNN blocks. Firstly, intrinsic sparsity in

* indicates interns at Intel Labs China. ✉ indicates corresponding authors.

3D data. Most of the voxels in a dense 3D grid are empty. Non-trivial voxels usually exist near the boundaries of objects. This property has been explored in several recent works [37,44,9,11]. Secondly, redundancy in 3D voxels. Dense 3D voxels are usually redundant, discarding a large portion of voxels (e.g. 70%) randomly does not prevent humans from reasoning the overall semantic information, as shown in Fig. 1. Thirdly, different subsets of original dense voxels contain complementary information. It is hard to recognize objects with small size and complex geometry when giving only partial voxels. These properties motivate us to design computation-efficient 3D CNNs for dense prediction tasks. We adopt Sparse Convolutional Network (SCN) [11]⁴ to exploit the intrinsic sparsity of 3D data, which encodes sparse 3D data with Hash Table and presents sparse convolution design. These designs can avoid unnecessary memory or computation cost on empty voxels. However, the computation is still intensive when the resolution is high or input is not so sparse. For example, the complexity of the baseline SCN used in this paper is about 80 GFLOPs while only outputting 1/64 sized predictions. Our work takes advantage of SCN and steps further by encouraging higher sparsity in feature maps. We propose SGC to exploit the redundancy of 3D voxels, which partitions features into different groups and makes voxels sparser. Then we conduct sparse convolution on each group. Because only valid voxels are considered in sparse convolution rather than all voxels in a regular grid, and only partial voxels exist in each group after partition, the computation of networks with SGC can be significantly reduced compared to previous SCN. Besides, in order to utilize the complementary information of different groups, results of different groups after certain SGC operations are gathered for further processing.

Network acceleration methods in 2D CNN such as weight pruning, quantization, and Group Convolution (GC) design [20,50] can also be used, but these methods have not been well explored in 3D CNNs for now. Different from these methods, **SGC speeds up 3D CNNs from another perspective by encouraging sparsity in feature maps**. Though recently there are works [6,13] exploiting sparsity in feature maps, they are not suitable for dense prediction tasks because some voxels need to be predicted are deactivated in the network. **Our method is orthogonal to Group Convolution**, which is an operation widely used in recent CNN architectures [24,46,3]. SGC is defined on spatial dimensions while GC is defined on channel dimension. Besides, because voxels in different groups are similar, weights are shared between different groups in SGC, which is not the case in GC.

We validate our method on semantic scene completion as test case to show its effectiveness on 3D dense prediction tasks. This task not only aims to predict semantic labels, but also needs to output complete structure which is different from the input. We introduce a novel SCN architecture that is applicable to scenarios where output has a different structure with input. Dense deconvolution layer and Abstract Module are designed to generate voxels which are absent in input and remove trivial voxels respectively. Multiscale encoder-decoder ar-

⁴ or called Submanifold Sparse Convolutional Network in [11].

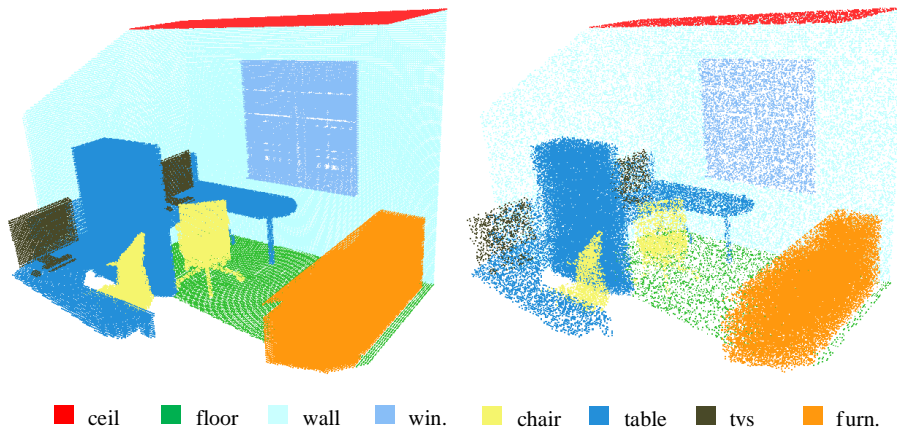


Fig. 1. A 3D scene image from the SUNCG dataset. Left is the ground truth image. Right is a sampled image with only 30% voxels reserved. Giving only partial voxels does not prevent humans in reasoning the overall semantic information, but it imposes a challenge to recognize small objects such as chair’s leg. (**Best viewed in color**)

chitecture and coarse-to-fine prediction strategy are used for final predictions. We evaluate our network on the SUNCG dataset [40] and achieve state-of-the-art results. Our SGC operation can reduce about 3/4 of the computation while losing only 0.7% and 1.2% in terms of Intersection over Union (IoU) for scene completion and semantic scene completion compared to networks without SGC.

Our main contributions are as follows:

- We propose SGC by exploiting sparsity in features for 3D dense prediction tasks, which can significantly reduce computation with slight loss of accuracy.
- We present a novel end-to-end sparse convolutional network design to generate unknown structures for 3D semantic scene completion.
- We achieve state-of-the-art results on the SUNCG dataset, reaching an IoU of 84.5% for scene completion and 70.5% for semantic segmentation.

2 Related works

2.1 3D Deep Learning

The success of deep learning in 2D computer vision areas has inspired researchers to employ CNN in 3D tasks, such as object recognition [45,30,32], shape completion [45,40,4], and segmentation [40,1]. However, the cubic growth in data size impedes building wider and deeper networks because of memory and computation restrictions. Recently, several works attempt to solve this problem by utilizing the intrinsic sparsity of 3D data. FPNN [28] used learned field probes to sample 3D data at a small set of positions, then fed features into fully connected layers. Graham et al. [9,11] proposed Hash Table based sparse convolutional

networks and solved the “submanifold dilation” problem by forcing to keep the same sparsity level throughout the network. OctNet [37] and O-CNN [44] used Octree-based 3D CNN for 3D shape analysis. SBNNet [35] performed convolution on blockwise decomposition of the structured sparsity patterns. Apart from these methods based on volumetric representation, PointNet [31] is a seminal work building deep neural networks directly on point clouds. PointNet++ [33] and Kd-Networks [23] further employed hierarchical architectures to capture local structures of point clouds.

Our main difference with these architectures is the introduction of SGC, which encourages higher sparsity in features and makes networks more efficient.

2.2 Computation-efficient Networks

Most previous computation-efficient networks focus on reducing model size to accelerate inference, such as pruning weight connections [25,14] and quantizing weights [8]. Another line of works uses GC to reduce the computation, such as MobileNet [20] and ShuffleNets [50]. GC separates features to different groups along channel dimension and performs convolution on each group parallelly. Besides, Graham [9] used smaller filters on different lattices to decrease the computation.

However, there are seldom works designing computation-efficient networks by exploiting higher sparsity in feature maps for 3D dense prediction tasks. Vote3deep [6] encouraged sparsity in feature maps using L_1 regularization. ILA-SCNN [13] used adaptive rectified linear unit to control the sparsity of features. But these methods are not suitable for dense prediction tasks, because some desired voxels are deactivated in the network and cannot be recovered. Besides, Li et al. [26] also exploited sparsity and reduced the computation of 2D segmentation task with cascaded networks, and only hard pixels are handled by deeper sub-models.

Different with these methods, we create groups along the spatial dimensions and make voxels in each group sparser. Computation of convolution can be largely reduced because only partial valid voxels are used in each computation.

2.3 3D Semantic Segmentation and Shape Completion

3D semantic segmentation [48,2,29,34] and Shape Completion [45,7,4,15] are both active areas in computer vision. 3D segmentation gives semantic labels to observed voxels, while shape completion completes missing voxels. SSCNet [40] combined these two tasks together and showed that segmentation and completion can benefit from each other. In order to generate high resolution 3D structure, various methods had been explored, such as long short-term memorized [15], coarse-to-fine strategy [5], 3D generative adversarial network [47], and inverse discrete cosine transform [22]. Recently, segmentation and completion are both benefited from these advanced 3D deep learning methods described in section 2.1. Different methods have been presented in the 3D segmentation challenge [49], such as SCN, Pd-Network, densely connected PointNet, and Point CNN [27].

For 3D completion tasks, advanced Octree-based CNN methods [36,41,17] were also used for generating high resolution 3D outputs. Our network architecture shares some similarities with [36,41], while the main difference is that we focus on efficient model design in this paper.

3 Method

In this section, we firstly give a brief introduction to previous SCN architecture [11], and then introduce SGC for computation-efficient 3D dense prediction tasks. Thirdly, a novel sparse convolutional network architecture which can predict unknown structures will be presented for semantic scene completion. Finally, details about training and networks will be given.

3.1 Sparse Convolutional Network

Previous dense 3D convolution is neither computational nor memory efficient because of the usage of dense 3D grid for representation. Another problem is that traditional “dense” convolution has the “dilation” problem [11] which will destroy the sparsity of 3D feature maps. For example, after a $3 \times 3 \times 3$ convolution, surrounding 26 voxels will be filled in. SCN addressed these problems by only storing non-empty voxels in 3D feature maps using Hash Table. **Only non-empty voxels are considered in sparse convolutional network.** Besides, **it forces to keep sparsity at the same level throughout the network** when performing convolution, which means the activation pattern of next layer is the same as the previous layer. These designs can largely decrease computation and memory requirements, enabling the usage of deeper 3D CNNs.

However, there is still intensive computation in 3D sparse CNN as mentioned above. Thus reducing the computation of 3D sparse CNN is necessary for real-time applications. Another problem of previous SCN is that it cannot be directly used for scene completion task. Because completion needs to output a complete structure which is different from the input, while previous SCN can only output predictions with the same structure as input. We introduce a novel sparse convolutional network to predict unknown structures.

3.2 Spatial Group Convolution

This section introduce SGC which can significantly reduce the computation of 3D dense prediction tasks. Our design makes use of those three properties of 3D data described in section 1 (see Fig. 2). We partition voxels uniformly into different groups, then conduct 3D sparse convolution on each group. Weights are shared among different groups because these groups are similar. Features of different groups are gathered later in order to utilize the complementary information of different groups.

In the implementation of SGC, we partition features along the spatial dimensions and then stack different groups along the batch dimension. For one sparse

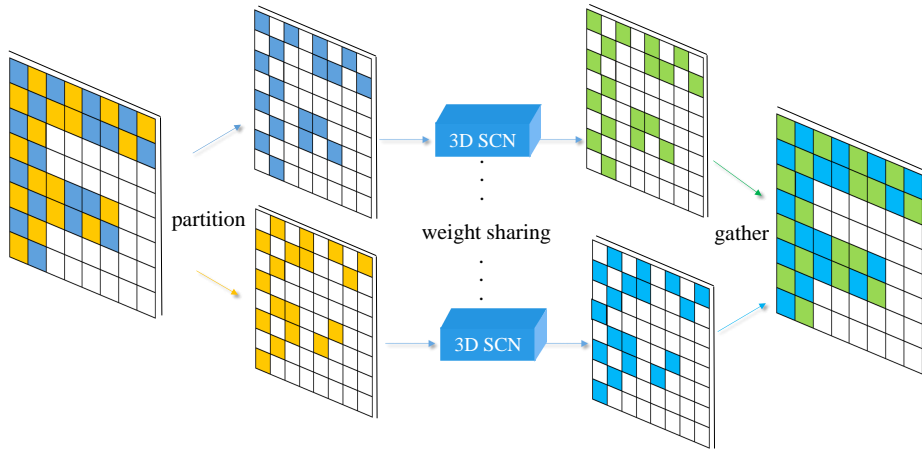


Fig. 2. Illustration of SGC. Feature maps are partitioned uniformly into different groups along the spatial dimensions (only two groups are shown here). 3D CNNs are conducted on different groups and give the final dense prediction for all voxels. Weights are shared between different groups.

feature map whose size is $B \times D \times H \times W \times C$ (*batchsize* \times *depth* \times *height* \times *width* \times *channel*), after the partition operation, it becomes $(G \times B) \times D \times H \times W \times C$, where G is the group number. Note that because we use Hash Table based representation, only non-empty voxels are stored. So this operation does not require extra memory. In each convolution computation, only part of original non-empty voxels in its receptive field participate in the calculation, and the number of valid voxels in each group is about $1/G$ of the original non-empty voxels after partition. The final computation cost is thus about $1/G = \frac{N \times \frac{1}{G} \times k^3}{N \times k^3}$ of original convolution when ignoring the bias computation, where N is the total number of valid voxels, and k is the filter size. SGC can readily replace plain 3D sparse convolution in existing CNNs.

Obviously, partition strategy plays an important role in SGC. Here we present two different partition strategies:

- Random partition method. Voxels of feature maps are partitioned into different groups randomly and uniformly.
- Partition with a fixed pattern. Random partition expects convolutional filters to be invariant to all possible patterns of activation, which may be hard for CNN to learn. We propose to partition input voxels with a fixed regular pattern for all input voxels throughout training and testing. For example, we can partition voxels by the following formulation:

$$i = \text{mod}(ax + by + cz, G) \quad (1)$$

where i is the group index, (x, y, z) is the position of the voxel, G is the total group number, mod is the modulus operation, (a, b, c) controls the distribu-

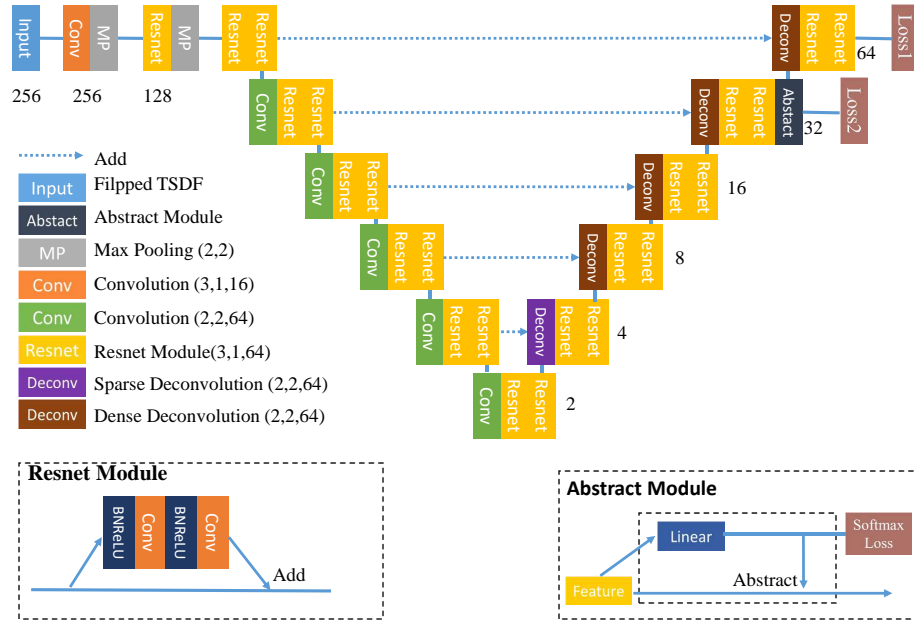


Fig. 3. Network architecture for semantic scene completion. Taking flipped TSDF as input, the network predicts occupancy and object labels in 1/4 size. **The resolution of each layer is marked nearby.** Parameters of each layer are shown in the order of (filter size, stride, output channel). Dense deconvolution layers can generate new voxels. The abstract module can abstract non-trivial voxels to high resolution according to the prediction in low resolution. **(Best viewed in color)**

tion of different groups. This strategy can also partition voxels uniformly but in a fixed pattern manner. Different (a, b, c) and G give different patterns.

3.3 Sparse Convolutional Networks for Semantic Scene Completion

This section will present a novel SCN architecture for semantic scene completion. Previous SCN [10] keeps the sparsity unchanged to avoid “submanifold dilation” problem. The output of previous SCN has the same known structure as input. This design restricts its application in shape completion, RGB-D fusion and etc., which aim to predict unknown structures. In order to generate unknown voxels for semantic scene completion task, we have to break this restriction.

Here we use multiscale encoder-decoder architecture [38]. As shown in Fig. 3, encoder modules are constituted of sparse convolutions described in section 3.1. While in decoder modules, we implement a “dense” deconvolution layer to generate new voxels. More specifically, after a “dense” up-sampling deconvolution layer, each voxel in low resolution will generate $2 \times 2 \times 2$ voxels in high resolution. The sparsity changes rather than keeping the same as the layers in encoder modules. New voxels can be generated in this process.

Applying this module repeatedly in each scale can generate all missing structures but it will soon destroy the sparsity of 3D feature maps just as the “sub-manifold dilation” problem. So we introduce Abstract Module similar to [36,41], which abstracts a coarse structure and removes unnecessary voxels in low resolution. Details will be refined in high resolution. The Abstract Module contains a $1 \times 1 \times 1$ convolution layer and a softmax layer, these layers give a prediction in this scale and provide guiding information for abstracting. Only voxels with non-empty labels and their surrounding voxels are abstracted. Abstracting these surrounding empty voxels within a distance of k could provide fine details. $k = 1$ works well in our practice. We apply the abstract module in resolution higher than 32 because removing voxels in early stages may hurt the performance. Since our setting exploits resolution 64 for output, one Abstract module is enough.

Voxel-wise softmax loss is used in the two scales which give a prediction:

$$L_i = \frac{1}{\sum_j w_j} \sum_j w_j L_{sm}(p_j, y_j), \quad (2)$$

where $i \in \{0, 1\}$ means resolution scale as shown in Fig. 3, L_{sm} is softmax loss, y_j is ground truth label of voxel j , p_j is the predicted possibility, and $w_j \in \{0, 1\}$ is the weight of this voxel. The final loss is a summation of all losses as follows:

$$L = \sum_i \alpha_i L_i, \quad (3)$$

where α_i is the weight for each scale. We found $\alpha_i = 1$ works well.

3.4 Implementation Details

Dataset. We train and evaluate our network on the SUNCG dataset, which is a manually created large-scale synthetic scene dataset [40]. It contains 139368 valid pairs of depth map and complete labels for training, and 470 pairs for testing. Depth maps are converted to volumes with a size of $240 \times 144 \times 240$. The ground truth labels are 12-class volumes with $1/4$ size of input volume.

Network Details. The detailed network architecture is illustrated in Fig. 3. For volumetric data encoding, we use flipped Truncated Signed Distance Function (fTSDF), which can enhance performance because it eliminates strong gradients in empty space [40]. The input size of our network is 256^3 , and we put the original fTSDF volume in the middle of input volume. The input volume is down-sampled twice using Max-pooling layer. Then a U-Net architecture follows, which contains six resolution scales, from 64^3 to 2^3 . Features from encoding stages and decoding stages are summed, and zeros are filled at missing locations. The network uses pre-activation Resnet block in encoding and decoding modules [18,19], and each block has two $3 \times 3 \times 3$ convolutions. Down-sampling and up-sampling are implemented by convolution layers with stride 2 and kernel size 2. SGC is used in resolution scales not less than 32^3 , which account for most of the computation. Partition operation is performed again once the resolution scale changes, which can help information flow across each other group.

The weight of each voxel is computed by randomly sampling empty and non-empty voxels at a ratio of 1 : 2 [40]. All non-empty voxels are positive examples. For negative examples, we mainly consider empty voxels around the surface as hard examples, which can be determined by the TSDF value of GT labels ($|TSDF| < 1$). The ratio of hard negative and easy negative examples is 9:1.

Training Policy. Networks are trained using stochastic gradient descent with a momentum of 0.9. The initial learning rate is 0.1, and L2 weight decay is 1e-4. We train our network for 10 epochs with a batch size of 4, and decay learning rate by a factor of $\exp(-0.5)$ in each epoch. In order to reduce training time, we randomly select 40000 samples in each epoch, and the total training time is about 5 days with a GTX TitanX GPU and two Intel E5-2650 CPUs.

4 Evaluation

In this section, we evaluate our network on the standard SUNCG test dataset. Both semantic scene completion results and scene completion results are given. Voxel-level IoU evaluation metric is used. Semantic scene completion results are evaluated on both observed and unobserved voxels, and completion results are evaluated on unobserved voxels. Table 1 and Table 2 show the quantitative results of our network without or with SGC. Fig. 4 shows the qualitative comparison with previous work. We also give results on real-world noisy NYU dataset [39].

4.1 Comparison to SSCNet

Table 1 shows the result of our baseline network without SGC (group number is 1). We outperform the previous SSCNet by a significant margin, having an improvement of 24.1% in semantic scene completion and 11.0% in scene completion, and achieving state-of-the-art results. Our network exceeds SSCNet in almost all classes, especially in small and hard categories such as chair, tvs and objects. We attribute this improvement to the novel SCN architecture that enables the usage of several advanced deep learning techniques such as deeper networks (15-layer vs 57-layer), multiscale network architecture (3 resolution scales vs 8 resolution scales), batch normalization layer [21] and stacked Resnet style blocks. Fig. 4 shows the visualization results of semantic scene completion from a single depth image. Obviously our baseline network produces visually better results compared to SSCNet, especially around the object boundaries.

4.2 Spatial Group Convolution Evaluation

This section describes the results of networks with SGC (see Table 2). We conduct experiments on 2,3,4,6 groups with different partition strategies. The efficiency is evaluated with FLOPs, i.e., the number of floating-point multiplication-adds of the whole network. As shown in Table 2, SGC can reduce about $(G-1)/G$ of the whole computation. Experiments show that 3D sparse CNNs can be sparsity-invariant to some extent, because accuracy only drops about 0.5% when

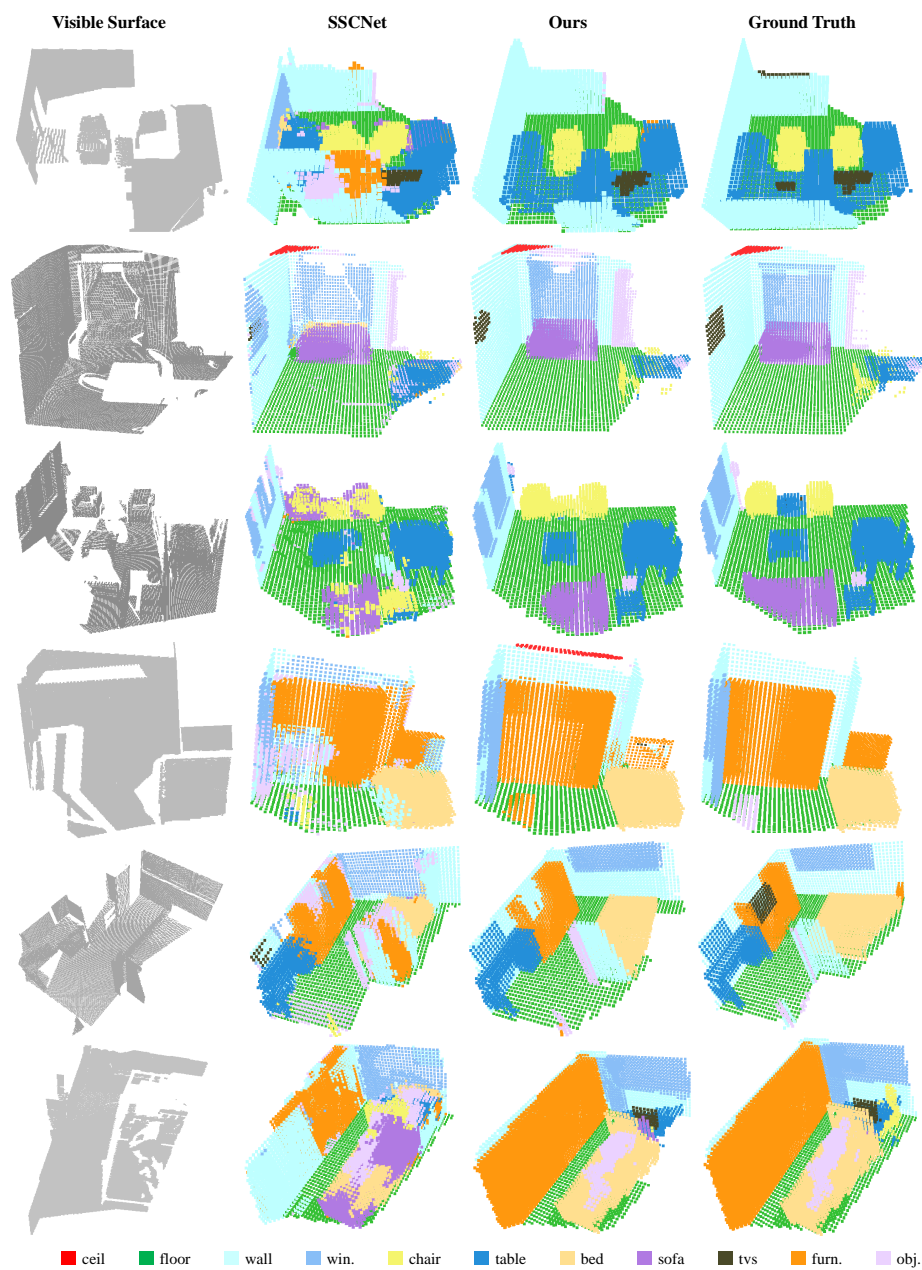


Fig. 4. Qualitative results of our network and SSCNet. We achieve obviously much better results, such as predictions around object boundaries.

Table 1. Quantitative results of our network and SSCNet on the SUNCG dataset. Scene completion IoU is measured on unobserved voxels, and all non-empty classes are treated as one category. Semantic scene completion IoU is measured on both observed and unobserved voxels. Overall, our method outperforms SSCNet by a large margin. Better results of each category are bold.

Method	scene completion			semantic scene completion											
	prec.	recall	IoU	ceil.	floor	wall	win.	chair	bed	sofa	table	tv	furn.	objs.	avg.
SSCNet [40]	76.3	95.2	73.5	96.3	84.9	56.8	28.2	21.3	56.0	52.7	33.7	10.9	44.3	25.4	46.4
Our	92.6	90.4	84.5	96.6	83.7	74.9	59.0	55.1	83.3	78.0	61.5	47.4	73.5	62.9	70.5

Table 2. Quantitative IoU (%) results of networks using SGC with random partition strategy or fixed pattern partition strategy. Both accuracy and FLOPs are given. For fixed pattern partition method, flexible parameters (a,b,c) are also given and we select the best results in our experiments. Best trade-off is bolded.

Group No.	Method	scene completion	semantic scene completion	FLOPs/G
1(Baseline)		84.5	70.5	79
2	Random	83.9	69.9	42
	Pattern(1,1,1)	84.0	69.6	39
3	Random	82.6	67.6	29
	Pattern(1,1,1)	84.0	69.6	27
4	Random	83.1	67.6	23
	Pattern(1,2,3)	83.8	69.3	22
6	Random	82.3	66.6	17
	Pattern(1,2,1)	82.6	66.9	16

dividing voxels into two groups even randomly, while only about 50% voxels preserved in this case. Increasing group number will reduce more computation at the cost of a little drop of performance. Compared to random partition method, fixed pattern partition strategy can give better performance yet requires less computation. For example, 1.7% IoU enhancement for semantic completion can be achieved using fixed pattern partition method when dividing voxels into four groups. Overall, SGC can significantly reduce the computation while maintaining accuracy, achieving a drop of only 0.7% and 1.2% in terms of IoU for scene completion and semantic completion task while using only 27.8% computation.

Table 3 shows the detailed semantic scene completion results of different categories using SGC. The accuracies of best and worst three categories compared to baseline network are marked in the table. It can be found that the IoUs of categories with small physical sizes such as chair, furniture, and objects drop more than categories with large size such as ceiling and floor. This may be caused by the fact that those small objects have fewer voxels. Dividing these voxels into different groups may lose important geometric information and makes it harder to distinguish these objects (see chair’s leg in Fig. 1). While large objects have surplus voxels, sparser voxels can still keep a rough structure. So, a possible future work to increase the accuracy of semantic scene completion is to adaptively

Table 3. Influence of SGC on each category. The numbers in third to sixth row mean IoU (%) drop when using SGC. The best or worst three are underlined or bolded.

Group No.	Method	ceil.	floor	wall	win.	chair	bed	sofa	table	tvs	furn.	objs.	avg.
Baseline		96.6	83.7	74.9	58.9	55.1	83.3	78.0	61.5	47.4	73.5	62.9	70.5
4	random	<u>0.4</u>	<u>-0.6</u>	<u>-2.0</u>	-2.9	-4.4	-3.0	-3.2	-4.1	-3.5	-4.4	-4.7	-2.9
	pattern	<u>0.2</u>	<u>-0.3</u>	<u>-0.8</u>	<u>0.7</u>	-3.7	-1.7	-1.5	-3.1	<u>1.1</u>	-1.9	-2.1	-1.2
6	random	<u>0.1</u>	<u>-0.7</u>	<u>-3.7</u>	<u>-2.8</u>	-5.7	-2.9	-3.5	-5.6	<u>-5.4</u>	-6.8	-6.0	-3.9
	pattern	<u>0.2</u>	<u>-0.4</u>	<u>-2.3</u>	<u>-3.8</u>	-5.5	<u>-1.8</u>	-3.7	-6.9	-4.0	-5.5	-5.8	-3.6

Table 4. Scene completion (IoU %) and semantic scene completion results (IoU %) on NYU dataset.

Method	scene completion			semantic scene completion											
	prec.	recall	IoU	ceil.	floor	wall	win.	chair	bed	sofa	table	tvs	furn.	objs.	avg.
SSCNet	57.0	94.5	55.1	15.1	94.7	24.4	0	12.6	32.1	35	13	7.8	27.1	10.1	24.7
Ours	71.9	71.9	56.2	17.5	75.4	25.8	6.7	15.3	53.8	42.4	11.2	0	33.4	11.8	26.7

handle large easy objects and small hard objects, sampling small objects with high density while sampling large objects with relatively low density.

We also tried sparsity invariant convolution [42] in random partition method, which normalizes convolution by a factor of valid voxels number, but it does not work in our task.

4.3 Evaluation on NYU dataset

NYU [39] contains 1449 depth maps captured by Kinect. Following SSCNet [40], we use Guo et al.’s algorithm [12] to generate ground truth annotations for semantic scene completion task. The object categories are mapped based on Handa et al. [16]. We trained the network described above from scratch on NYU dataset. The base of exponential learning rate decay is 0.12 and we trained it for 40 epochs using the whole dataset. Other hyperparameters are same as experiments on SUNCG. Table 4 shows that our network achieves an improvement of 2.0% in semantic scene completion and 1.1% in scene completion compared to SSCNet. Table 5 gives detail results on NYU dataset. It shows that SGC operation is still effective on real data. The fixed pattern partition method gives comparable or even better results than baseline network, and it is consistently better than the random partition method. Note that there exists a gap between the improvements on SUNCG and NYU. We attribute this gap to the fact that misalignment and incomplete annotations are common in the generated labels [12]. This may both mislead the training and evaluation procedures, and it may be unfavorable for our network considering the sparsity geometry representation.

Table 5. Results (IoU%) of networks with SGC using different partition strategies on NYU dataset. (SSC stands for semantic scene completion)

Group No.	1	2		3		4	
Method	baseline	random	pattern	random	pattern	random	pattern
SSC	26.4	24.1	26.5	23	26.7	22.6	25.9
Completion	55.7	53	54.8	52.2	56.2	52.6	55.1

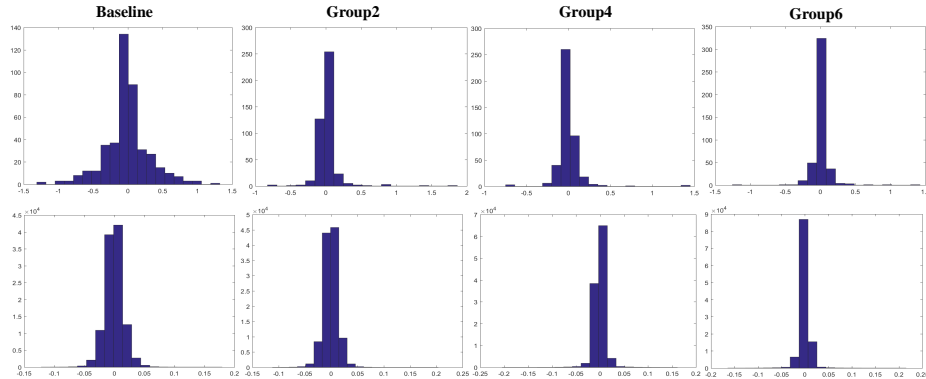


Fig. 5. Histograms of learned weight values of SCN and SGC with different groups. The first row shows the statistics of the first convolution layer, and the second row shows that of the last convolution layer. Filters of SGC have “sharper” histograms.

5 Discussion

5.1 What does Spatial Group Convolution learn?

In Fig. 5, we visualize the histograms of learned weight values of networks without or with SGC using random partition method. It can be observed that filters of SGC have “sharper” histograms while normal SCN filters have relative “flat” histograms, which means the values of SGC filters are pretty close. The histograms become “sharper” with the increase of group number. This may be caused by that filters of SGC need to be invariant to different sparsity patterns, so the values of filters at different locations had better be close to adapt to different sparsity patterns.

As for SGC with fixed pattern partition, we find it learned an irregular convolutional kernel. In Fig. 6a, we show a simple case in a 2D convolution which divides voxels into two groups. The valid convolutional kernel shape is always “X” because the sparsity pattern keeps the same when sliding the convolutional kernel. Fig. 6b shows the valid convolutional filter shapes used in Table 2.

5.2 Information flow among different groups

The SGC operation partitions voxels into different groups. During convolution, different groups are independent and have no information flow across each other.

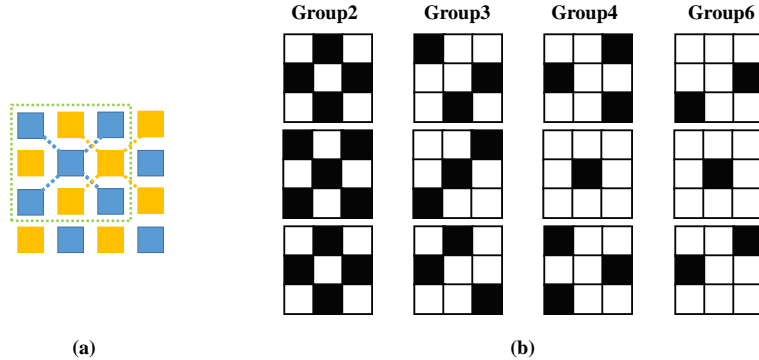


Fig. 6. Illustration of SGC with fixed pattern partition. (a) shows that for a 3×3 kernel, an “X” shape filter is learned when partitioning voxels into two groups. (b) shows the learned $3 \times 3 \times 3$ filters in Table 2. Filters are drawn by slice.

However, after SGC, the voxels are gathered and fed into down-sampling convolution or up-sampling deconvolution layers, in which information of different groups can communicate. Besides, we also explored more complicated methods to help information exchange among different groups. For example, Shuffled SGC, which is inspired by ShuffleNet [50]. ShuffleNet uses channel shuffle to help information flow across feature channels, while we shuffle the features across spatial dimensions which is implemented by using different partition patterns in the two convolution layers of Resnet block. But no obvious improvement is observed in our case.

6 Conclusions

The paper presents an efficient semantic scene completion network with Spatial Group Convolution. SGC partitions feature maps into different groups along the spatial dimensions and can significantly reduce the computation with slight loss of accuracy. Besides, we propose a novel end-to-end sparse convolutional network architecture for 3D semantic scene completion and set a new accuracy record on the SUNCG dataset.

Acknowledgment

This work was supported in part by National Key Research and Development Program of China (2017YFC0108000), National Natural Science Foundation of China (81427803, 81771940, 61132007, 61172125, 61601021, and U1533132), Beijing Municipal Natural Science Foundation (7172122, L172003), and Soochow-Tsinghua Innovation Project (2016SZ0206).

References

1. Ben-Shabat, Y., Lindenbaum, M., Fischer, A.: 3d point cloud classification and segmentation using 3d modified fisher vector representation for convolutional neural networks. arXiv preprint arXiv:1711.08241 (2017)
2. Chang, A., Dai, A., Funkhouser, T., Halber, M., Niebner, M., Savva, M., Song, S., Zeng, A., Zhang, Y.: Matterport3d: Learning from rgb-d data in indoor environments. In: 3DV. pp. 667–676. IEEE (2017)
3. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: CVPR. pp. 1251–1258 (2017)
4. Dai, A., Qi, C.R., Nießner, M.: Shape completion using 3d-encoder-predictor cnns and shape synthesis. In: CVPR. vol. 3 (2017)
5. Dai, A., Ritchie, D., Bokeloh, M., Reed, S., Sturm, J., Nießner, M.: Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans. In: CVPR. vol. 1, p. 2 (2018)
6. Engelcke, M., Rao, D., Wang, D.Z., Tong, C.H., Posner, I.: Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In: ICRA. pp. 1355–1361. IEEE (2017)
7. Firman, M., Mac Aodha, O., Julier, S., Brostow, G.J.: Structured prediction of unobserved voxels from a single depth image. In: CVPR. pp. 5431–5440 (2016)
8. Gong, Y., Liu, L., Yang, M., Bourdev, L.: Compressing deep convolutional networks using vector quantization. arXiv preprint arXiv:1412.6115 (2014)
9. Graham, B.: Sparse 3d convolutional neural networks. arXiv preprint arXiv:1505.02890 (2015)
10. Graham, B., Engelcke, M., van der Maaten, L.: 3d semantic segmentation with submanifold sparse convolutional networks. CVPR (2018)
11. Graham, B., van der Maaten, L.: Submanifold sparse convolutional networks. arXiv preprint arXiv:1706.01307 (2017)
12. Guo, R., Zou, C., Hoiem, D.: Predicting complete 3d models of indoor scenes. arXiv preprint arXiv:1504.02437 (2015)
13. Hackel, T., Usvyatsov, M., Galliani, S., Wegner, J.D., Schindler, K.: Inference, learning and attention mechanisms that exploit and preserve sparsity in convolutional networks. arXiv preprint arXiv:1801.10585 (2018)
14. Han, S., Pool, J., Tran, J., Dally, W.: Learning both weights and connections for efficient neural network. In: NIPS. pp. 1135–1143 (2015)
15. Han, X., Li, Z., Huang, H., Kalogerakis, E., Yu, Y.: High-resolution shape completion using deep neural networks for global structure and local geometry inference. In: CVPR. pp. 85–93 (2017)
16. Handa, A., Patraucean, V., Badrinarayanan, V., Stent, S., Cipolla, R.: Understanding real world indoor scenes with synthetic data. In: CVPR. pp. 4077–4085 (2016)
17. Häne, C., Tulsiani, S., Malik, J.: Hierarchical surface prediction for 3d object reconstruction. In: 3DV (2017)
18. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
19. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: ECCV. pp. 630–645. Springer (2016)
20. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. ArXiv e-prints (Apr 2017)

21. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML. pp. 448–456 (2015)
22. Johnston, A., Garg, R., Carneiro, G., Reid, I., vd Hengel, A.: Scaling cnns for high resolution volumetric reconstruction from a single image. In: ICCV Workshops (2017)
23. Klokov, R., Lempitsky, V.: Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In: ICCV. pp. 863–872. IEEE (2017)
24. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. pp. 1097–1105 (2012)
25. Le Cun, Y., Denker, J.S., Solla, S.A.: Optimal brain damage. In: NIPS. pp. 598–605. NIPS’89, MIT Press, Cambridge, MA, USA (1989)
26. Li, X., Liu, Z., Luo, P., Change Loy, C., Tang, X.: Not all pixels are equal: Difficulty-aware semantic segmentation via deep layer cascade. In: CVPR. pp. 3193–3202 (2017)
27. Li, Y., Bu, R., Sun, M., Chen, B.: PointCNN. ArXiv e-prints (Jan 2018)
28. Li, Y., Pirk, S., Su, H., Qi, C.R., Guibas, L.J.: Fpnn: Field probing neural networks for 3d data. In: NIPS. pp. 307–315 (2016)
29. Liu, F., Li, S., Zhang, L., Zhou, C., Ye, R., Wang, Y., Lu, J.: 3dcnn-dqn-rnn: A deep reinforcement learning framework for semantic parsing of large-scale 3d point clouds. In: CVPR. pp. 5678–5687 (2017)
30. Maturana, D., Scherer, S.: Voxnet: A 3d convolutional neural network for real-time object recognition. In: IROS. pp. 922–928. IEEE (2015)
31. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: CVPR. pp. 652–660 (2017)
32. Qi, C.R., Su, H., Nießner, M., Dai, A., Yan, M., Guibas, L.J.: Volumetric and multi-view cnns for object classification on 3d data. In: CVPR. pp. 5648–5656 (2016)
33. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: NIPS. pp. 5105–5114 (2017)
34. Qi, X., Liao, R., Jia, J., Fidler, S., Urtasun, R.: 3D Graph Neural Networks for RGBD Semantic Segmentation. In: ICCV (2017)
35. Ren, M., Pokrovsky, A., Yang, B., Urtasun, R.: Sbnnet: Sparse blocks network for fast inference. In: CVPR. pp. 8711–8720 (2018)
36. Riegler, G., Ulusoy, A.O., Bischof, H., Geiger, A.: Octnetfusion: Learning depth fusion from data. In: 3DV (2017)
37. Riegler, G., Ulusoy, A.O., Geiger, A.: Octnet: Learning deep 3d representations at high resolutions. In: CVPR. vol. 3 (2017)
38. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI. pp. 234–241. Springer (2015)
39. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: ECCV. pp. 746–760. Springer (2012)
40. Song, S., Yu, F., Zeng, A., Chang, A.X., Savva, M., Funkhouser, T.: Semantic scene completion from a single depth image. In: CVPR. pp. 190–198. IEEE (2017)
41. Tatarchenko, M., Dosovitskiy, A., Brox, T.: Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In: CVPR. pp. 2088–2096 (2017)
42. Uhrig, J., Schneider, N., Schneidre, L., Franke, U., Brox, T., Geiger, A.: Sparsity invariant cnns. In: 3DV (2017)
43. Varley, J., DeChant, C., Richardson, A., Ruales, J., Allen, P.: Shape completion enabled robotic grasping. In: IROS. pp. 2442–2447. IEEE (2017)

44. Wang, P.S., Liu, Y., Guo, Y.X., Sun, C.Y., Tong, X.: O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *TOG* **36**(4), 72 (2017)
45. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: *CVPR*. pp. 1912–1920 (2015)
46. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: *CVPR*. pp. 5987–5995. *IEEE* (2017)
47. Yang, B., Rosa, S., Markham, A., Trigoni, N., Wen, H.: 3d object dense reconstruction from a single depth view. *arXiv preprint arXiv:1802.00411* (2018)
48. Yi, L., Guibas, L., Hertzmann, A., Kim, V.G., Su, H., Yumer, E.: Learning hierarchical shape segmentation and labeling from online repositories. *TOG* **36**(4), 70 (2017)
49. Yi, L., Shao, L., Sava, M., Huang, H., Zhou, Y., Wang, Q., Graham, B., Engelcke, M., Klovov, R., Lempitsky, V., Gan, Y., Wang, P., Liu, K., Yu, F., Shui, P., Hu, B., Zhang, Y., Li, Y., Bu, R., Sun, M., Wu, W., Jeong, M., Choi, J., Kim, C., Geetchandra, A., Murthy, N., Ramu, B., Manda, B., Ramanathan, M., Kumar, G., Preetham, P., Srivastava, S., Bhugra, S., Lall, B., Haene, C., Tulsiani, S., Malik, J., Lafer, J., Jones, R., Li, S., Lu, J., Jin, S., Yu, J., Huang, Q., Kalogerakis, E., Savarese, S., Hanrahan, P., Funkhouser, T., Su, H., Guibas, L.: Large-Scale 3D Shape Reconstruction and Segmentation from ShapeNet Core55 (2017)
50. Zhang, X., Zhou, X., Lin, M., Sun, J.: Shufflenet: An extremely efficient convolutional neural network for mobile devices. *CVPR* (2018)