2019 International Conference on Robotics and Automation (ICRA)
Palais des congres de Montreal, Montreal, Canada, May 20-24, 2019

Hierarchical Depthwise Graph Convolutional Neural Network for 3D Semantic Segmentation of Point Clouds

Zhidong Liang[1], Ming Yang[2], Liuyuan Deng[2], Chunxiang Wang[2] and Bing Wang[2]

***Abstract*— This paper proposes a hierarchical depthwise graph convolutional neural network (HDGCN) for point cloud semantic segmentation. The main chanllenge for learning on point clouds is to capture local structures or relationships. Graph convolution has the strong ability to extract local shape information from neighbors. Inspired by depthwise convolution, we propose a depthwise graph convolution which requires less memory consumption compared with the previous graph convolution. While depthwise graph convolution aggregates features channel-wisely, pointwise convolution is used to learn features across different channels. A customized block called DGConv is specially designed for local feature extraction based on depthwise graph convolution and pointwise convolution. The DGConv block can extract features from points and transfer features to neighbors while being invariant to different point orders. HDGCN is constructed by a series of DGConv blocks using a hierarchical structure which can extract both local and global features of point clouds. Experiments show that HDGCN achieves the state-of-the-art performance in the indoor dataset S3DIS and the outdoor dataset Paris-Lille-3D.**

I. INTRODUCTION

The LIDAR scanner is one of the most important sensors in the autonomous driving system. It produces point clouds which can measure the surrounding objects accurately. Point clouds can directly provide rich geometric information which can be utilized in localization. Meanwhile, semantic understanding based on point clouds is also essential especially for high-level autonomous driving system. It does not only tell where the object is but also classifies what the object is which is meaningful for localization, decision and planing modules.

Among tasks related to 3D point cloud understanding, semantic segmentation is challenging as it needs to assign each point a semantic label (shown in Fig. 1). It is the basic for generating 3D semantic map which can provide prior semantic understanding for autonomous vehicles. In this work, we focus on 3D point cloud semantic segmentation.

Previous work proposed several approaches to extract semantic information from point clouds. One simple idea [1] [2] [3] is first to project 3D point clouds to a collections of 2D images and then utilize the experience and the structure of the 2D CNN. Such projection loses 3D geometric information and is not suitable for tasks such as semantic segmentation. Voxelizing point clouds into a regular grid [4] [5] [6] is another approach to preprocess the raw point clouds. However, high memory demand of 3D grids limits its spatial resolution which affects the classification accuracy. Recently, PointNet [7] directly takes raw point clouds as input without extra projection or voxelization. PointNet++ [8] proposed a hierarchical structure to capture local features based on PointNet. PointCNN [9] uses $\mathcal{X}$-transformation to permute local points into canonical order so that local shape information can be considered more precisely. However, it is not easy for $\mathcal{X}$-transformation to learn an accurate permutation matrix. Graph convolutional neural network [10] [11] [12] [13] is widely used in local relationship modeling and shape analysis. Compared with spectral graph CNN [11], spatial graph CNN [12] [13] is more suitable for point cloud semantic segmenation because of better flexibility and generalizaion ability. However, most of the previous graph CNNs applied in point cloud semantic segmentation keep all points over the whole network which cannot extract hierarchical features. Also, some of them are not designed specially for point clouds which cannot extract local features satisfactorily. Additionally, the memory demand of the spatial graph CNN in [12] is huge and cannot support the deep feature channel.

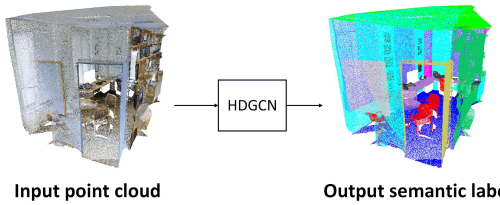

Fig. 1. The input of our network is point clouds with xyz coordinates and other attributes such as RGB or intensity. The output is the semantic label for each point.

In this paper, we propose a hierarchical depthwise graph convolutional neural network (HDGCN) for point cloud semantic segmentation. We propose a block called DGConv for point clouds which is effective on extracting local features. The graph convolution is an important part of our DGConv block. To reduce the memory consumption, we propose a depthwise graph convolution followed by a pointwise convolution to replace the previous graph convolution. Inspired by the success of the hierarchical structure in image semantic segmentation and point cloud semantic segmentation, we

*This work was supported by National Natural Science Foundation of China (U1764264/61873165), Shanghai Automotive Industry Science and Technology Development Foundation (1733/1807), International Chair on automated driving of ground vehicle. Ming Yang is the corresponding author.

[1]Zhidong Liang is with Research Institute of Robotics, Shanghai Jiao Tong University, Shanghai, 200240, China.

[2]Ming Yang, Liuyuan Deng, Chunxiang Wang and Bing Wang are with Department of Automation, Shanghai Jiao Tong University, Key Laboratory of System Control and Information Processing, Ministry of Education of China, Shanghai, 200240, China(phone: +86-21-34204553; e-mail: MingYang@sjtu.edu.cn).

combine the hierarchical structure and the DGConv block to extract both local and global features of point clouds hierarchically.

The key contributions of our work are as follows:

- We propose a depthwise spatial graph convolution followed by 1×1 convolution as the pointwise convolution to replace the spatial graph convolution in [12]. Based on this, we design the DGConv block which is effective for local feature extraction.
- We utilize the hierarchical structure combined with the DGConv block to extract local and global features hierarchically.
- Experiments show that HDGCN achieves state-of-the-art performance on Stanford Large-Scale 3D Indoor Spaces Dataset (S3DIS) and Paris-Lille-3D benchmark.

II. RELATED WORK

A. Deep Learning on Point Clouds

Volumetric Representation. Volumetric representation is to convert point clouds into voxelized grids. 3D CNN is commonly applied over voxels. [4] combines 3D FCN and CRF as an end-to-end framework for per-voxel classification. However, the performance of voxel-based method is limited by the resolution of the voxels. [5] [6] leverage the octree structure to exploit sparsity property of 3D data and enable 3D CNN to achieve a higher resolution.

Multi-view Representation. Multi-view based methods project 3D data into multiple 2D image views and apply standard 2D CNN. [1] first proposed a multi-view CNN for 3D shape recognition. [2] fuses projected 2D images with the camera image for 3D object detection. [3] picks various camera positions and generates virtual images with RGB texture and depth texture for point cloud semantic segmentation. Compared with volumetric representation, view-based methods are much more efficient. However, geometric information loss caused by projection makes it difficult to achieve dominating performance on tasks such as point-wise semantic segmentation.

Point Clouds. [7] is a pioneer using point-based method to process 3D data. [8] considers hierarchical features of point sets. [14] projects unordered points into an ordered sequence and then leverages recurrent neural network to aggregate local features. [9] proposed $\mathcal{X}$-Conv to permute points into canonical order and weight input features. [15] partitions point clouds into geometrically homogeneous elements called superpoints and applies PointNet for superpoint embedding.

Graph CNN. Spectral graph CNN [10] [11] and spatial graph CNN [13] [12] are two main representations of deep learning on graphs. Spectral graph CNN is defined in the spectral domain based on Fourier analysis. [11] proposed a first-order approximation of spectral graph convolution to avoid the Laplcacian eigendecomposition for more efficient implementation. However, most spectral methods can only be applied to the fixed graph so they are not suitable for point clouds. Several spatial graph CNN were proposed to process point clouds in the spatial domain. [12] proposed edge convolution to extract shape features from local neighborhood graph. [13] proposed EdgeConv to incorporate local neighborhood information.

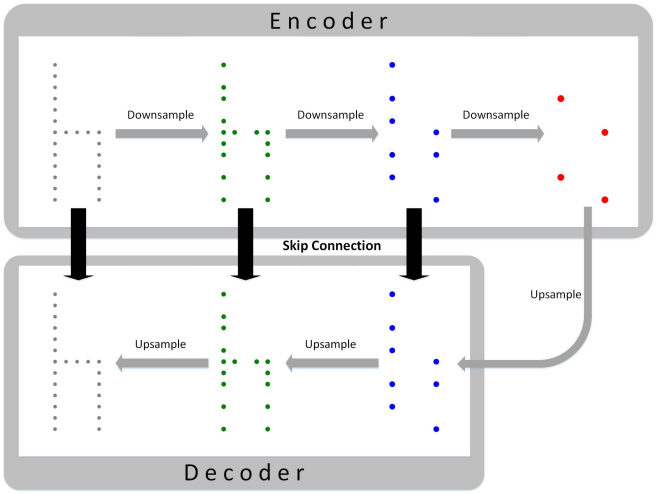

Fig. 2. Illustration of the change of the point set during the whole network. The input is simliar to a chair. Farthest point sampling (FPS) is used to downsample the point set from the last layer. When upsampling, it gets the point set from the corresponding layer in encoder by skip connection.

B. Depthwise Convolution

[16] first developed depthwise separable convolutions to accelerate convergence of AlexNet. [17] proposed an efficient model called Xception inspired by depthwise separable convolution. The idea of depthwise convolution is widely used to reduce the model size and increase the model efficiency.

III. METHOD DESCRIPTION

We first review the hierarchical structure applied to point clouds in [8] and then propose our depthwise graph convolution. Afterward, we show the design of the DGConv block and the hierarchical depthwise graph convolutional neural network (HDGCN).

A. Hierarchical Structure

For point cloud semantic segmentation, we choose the encode-decode structure for hierarchical feature extraction. Given n_0 points, farthest point sampling (FPS) can be used to downsample the input point set to n_1 points($n_1 < n_0$) which is a subset of the input. Meanwhile, features from n_0 points should be aggregated to n_1 points using some algorithm. In this paper, we use the depthwise graph convolution which will be described later.

The whole architecture of the encode-decode structure is shown in Fig. 2. FPS is used to iteratively downsample points which generated by the last layer. The encoder in Fig. 2 consists of three times of downsampling. After downsampling, the remaining points extract more general shape information and have a larger receptive field which is beneficial for semantic segmentation. After encoding, the point number needs to be upsampled to the same as input for

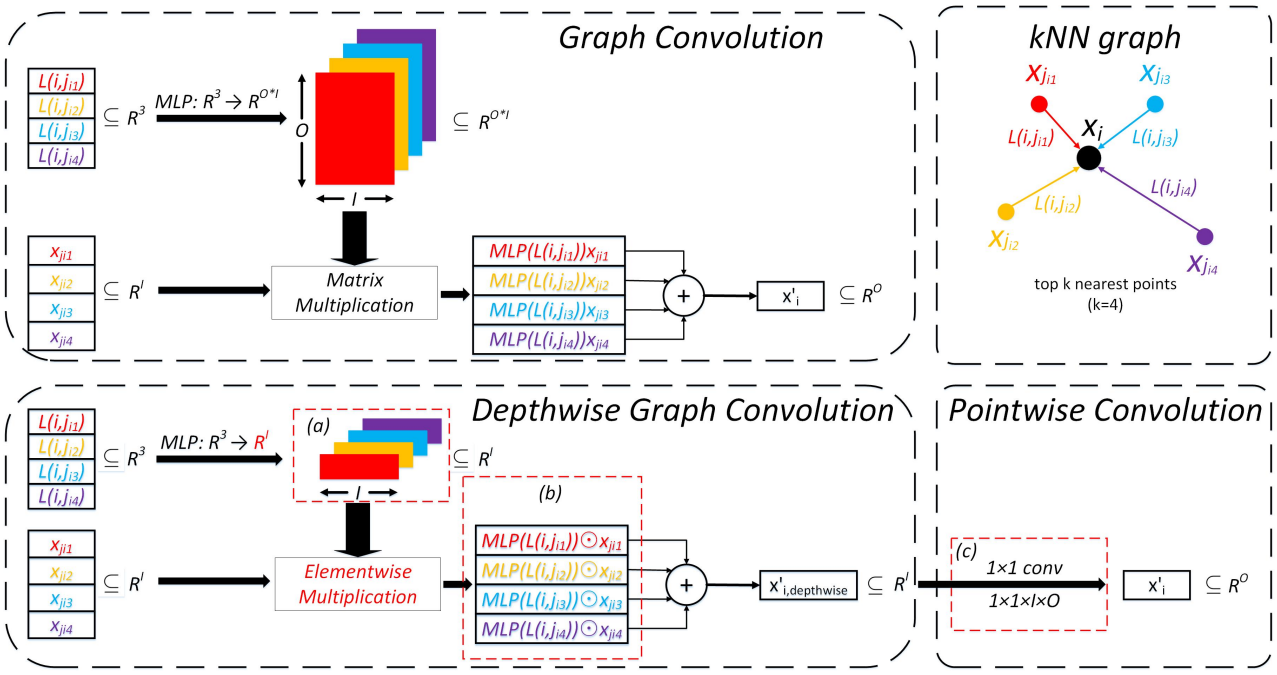

Fig. 3. Illustration of the graph convolution. The top is the graph convolution in [12]. The bottom is our proposed depthwise graph convolution. The graph convolution is based on kNN graph and here we choose $k = 4$. Our proposed depthwise graph convolution has three special designs. In rectangle (a), we generate filter $\subseteq R^{1\times I}$ which need less memory consumption. In rectangle (b), we use the elementwise multiplication instead of the matrix multiplication. In rectangle (c), we use a 1×1 convolution to extract features across different channels. Here "MLP" stands for multi-layer perceptron.

pointwise labeling. When decoding, skip connection is used to obtain point coordinates and feature information from the corresponding layer in the encoder.

B. Depthwise Graph Convolution

Unlike images, it is not easy to define the convolution operation over point clouds. Graph convolution is proposed to handle unordered data and has strong ability to extract local shape information. We briefly introduce the spatial graph convolution proposed in [12] and then describe our depthwise graph convolution. We point that our description of the method is a little different from that in [12] but they are actually the same.

Assume an I-dimensional point cloud with n points denoted by $X = \{x_1, ..., x_n\} \subseteq R^I$. Let us consider an directed graph $G = (V, E)$ composed of a set of vertice with $V = \{1, ..., n\}$ and $E \subseteq V \times V$ is constructed according to the k-nearest neighbour (kNN) rule. For a specific point x_i, it searches the k nearest points $\left\{x_{j_{i_1}}, ..., x_{j_{i_k}}\right\}$ and connects with them to generate the edge $L(i, j_{i_m})(m = 1, ..., k)$. For simplicity, we define $L(i, j_{i_m})$ equal to $p_{j_{i_m}} - p_i$. $p_i \subseteq R^3$ is the 3D coordinate of the i_{th} point.

We symbolize the input as $X = \{x_1, ..., x_n\} \subseteq R^I$ and the output as $X' = \{x'_1, ..., x'_n\} \subseteq R^O$. x_i represents the input feature of the i_{th} point. x'_i represents the output feature of the i_{th} point. The spatial graph convolution (shown in the top of Fig. 3) can be formalized as follows:

$$x'_i = \sum_{m=1}^{k} MLP(L(i, j_{i_m}); \theta) x_{j_{i_m}} \tag{1}$$

MLP: $R^3 \mapsto R^{O\times I}$ is an instantiation of the filter-generating network in [12]. The input of MLP is the edge attributes $L(i, j_{im})$ while the output is the filter to the feature $x_{j_{im}}$.

The output dimension of MLP in equation (1) is $O \times I$. For n input points with k neighbors, the memory demand is $n \times k \times O \times I$ which is extremely huge when I and O are large. As the dimension of the output is much higher than that of other variables in MLP, we only consider the output while ignoring the memory consumption of other variables. The memory consumption of the output of equation (1) is $n \times k \times O$. High memory demand limits the extraction of high-level semantic information. In this paper, we propose a depthwise spatial graph convolution to reduce the memory consumption.

Rethinking the spatial graph convolution above, it can be divided into two stages. First, graph convolution is used to extract local features channel-wisely which is called depthwise graph convolution (shown in the bottom of Fig. 3). This operation can be formalized as follows:

$$x'_{i,depthwise} = \sum_{m=1}^{k} MLP(L(i, j_{im}); \theta) \odot x_{j_{im}} \tag{2}$$

In equation (2), we define the MLP: $R^3 \mapsto R^I$ instead of $R^3 \mapsto R^{O\times I}$ in equation (1). Correspondingly, the elementwise multiplication is defined between the output of MLP and the feature instead of matrix multiplication in equation (1). The ouput $x'_{i,depthwise} \subseteq R^I$ is the same dimension as x_i. Its function is to obtain the local information from surrounding points.

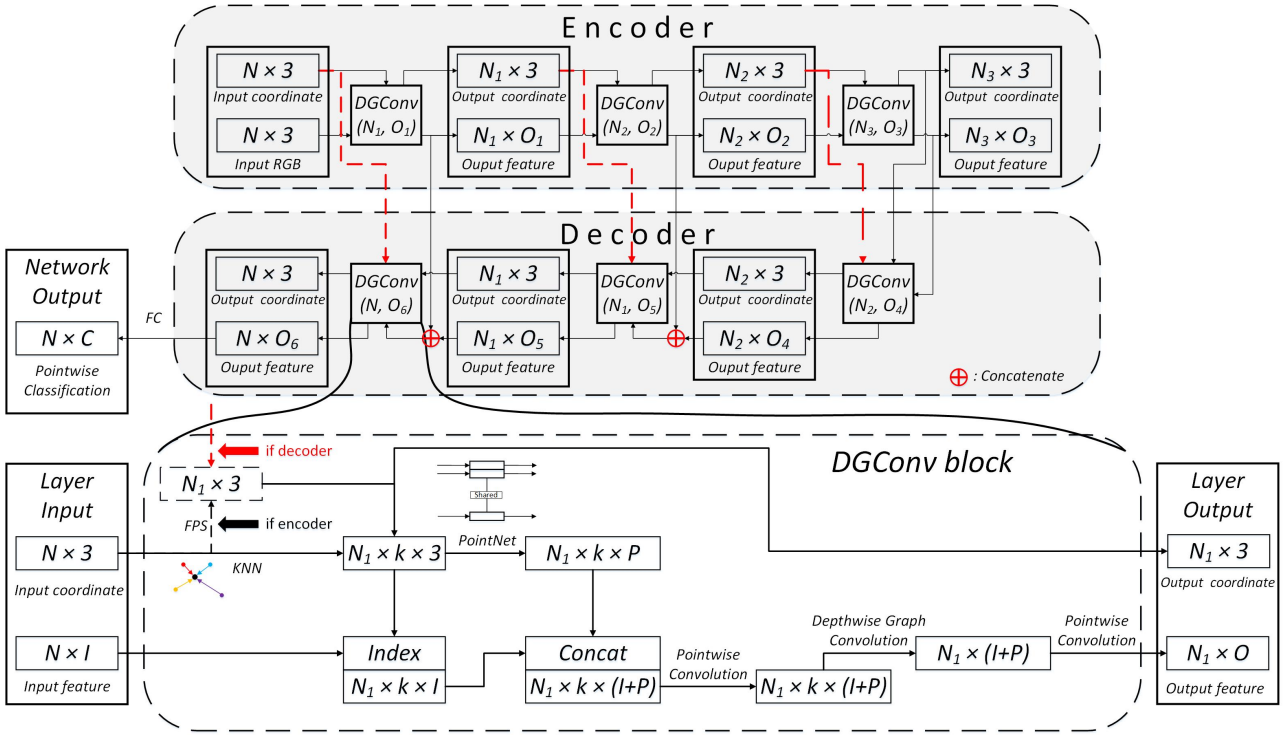

Fig. 4. Illustration of our hierarchical depthwise graph convolutional neural network. The whole architecture is composed by a series of customized blocks called DGConv based on depthwise graph convolution and pointwise convolution. DGConv(N_i, O_i) means that this DGConv block outputs N_i points with O_i feature channels. In DGConv block, $N_1 \times 3$ is generated by FPS only when encoding. If decoding, $N_1 \times 3$ is gotten by skip connction (shown in red line) instead of FPS. Finally, the network outputs $N \times C$ by fully connected layer where C is the number of class.

Second, pointwise convolution is used to extract features across different channels. PointNet [7] is a simple but effective pointwise network. MLP is the basic unit of PointNet. It can also be viewed as multiple 1×1 convolutions. Here we use 1×1 convolution as the pointwise convolution. It projects $x'_{i,depthwise} \subseteq R^I$ to $x'_i \subseteq R^O$ where I is the number of the input channel of the pointwise convolution and O is the number of the output channel.

By dividing the graph convolution into two stages, the memory consumption is greatly reduced. The output dimension of MLP in equation (2) is I. For n input points with k neighbors, the output dimensions of MLP and equation (2) are both $n \times k \times I$. For pointwise convolution, the memory demand of the kernel parameter is $1 \times 1 \times I \times O$ and the memory demand of the output is $n \times k \times O$. So the total memory demand of the depthwise graph convolution and the pointwise convolution is $n \times k \times (2I + O) + 1 \times 1 \times I \times O$. Compared with $(n \times k \times O \times I + n \times k \times O)$, the memory demand is much smaller especially when feature dimension O is large. We argue that we ignore some parameters and variables which have little influence when analyzing memory consumption for simplicity.

C. Hierarchical Depthwise Graph Convolutional Neural Network

Our hierarchical depthwise graph convolutional neural network is composed of a series of unit blocks called DGConv based on depthwise graph convolution and pointwise convolution. The structure of the DGConv block is shown in the bottom of Fig. 4. The input of the block contains two parts: the coordinate and the feature of input points. FPS is used to downsample the input points when encoding. If decoding, we need to upsample the input points instead of downsampling. Here we simply use the points from the corresponding layer in the encoder to achieve upsampling instead of using FPS. For each downsampled (or upsampled) point, it searches the k nearest points among the input points to construct the local directed graph. Then a small PointNet is used to extract information from the local graph. According to the constructed kNN graph, we can index the input features for each point in the graph. Features from PointNet and index are concatenated later. Before depthwise graph convolution, a pointwise convolution is applied to aggregate features from PointNet and indexed features. Then depthwise graph convolution and pointwise convolution are used to extract features respectively.

The whole network structure includes the encoder part and the decoder part shown in Fig. 4. When encoding, three DGConv blocks are used to iteratively downsample points and extract global and high-level semantic information. When decoding, skip connection is used to upsample points and concatenate features from the encoder. The DGConv block in the decoder is the same as that in the encoder except for the replacement of FPS. Finally, the network outputs the pointwise label.

Comparison with state-of-the-art methods. Point-

TABLE I

RESULTS ON S3DIS DATASET. WE PROVIDE BOTH RESULTS OF AREA 5 AND SIX-FOLDER CROSS VALIDATION.

Method	mIOU	mAcc	ceiling	floor	wall	beam	column	window	door	chair	table	bookcase	sofa	board	clutter
Test on Area 5															
PointNet [7]	41.09	48.98	88.80	97.33	69.80	**0.05**	3.92	46.26	10.76	52.61	58.93	40.28	5.85	26.38	33.22
MS3_DVS [18]	46.32	57.93	79.03	88.07	53.55	0.00	20.47	29.01	37.29	68.84	63.72	47.44	**61.62**	16.50	36.64
SEGCloud [4]	48.92	57.35	90.06	96.05	69.86	0.00	18.37	38.35	23.12	75.89	70.40	58.42	40.88	12.96	41.60
RSNet [14]	51.93	59.42	93.34	**98.36**	79.18	0.00	15.75	45.37	50.10	65.52	67.87	22.45	52.45	41.02	43.64
SPGraph [15]	54.67	61.75	91.49	97.89	75.89	0.00	14.25	**51.34**	**52.29**	86.35	77.40	65.49	40.38	7.23	50.67
HDGCN(Ours)	**59.33**	**65.81**	**94.00**	98.26	**80.28**	0.00	**24.04**	48.24	33.82	**86.86**	**77.74**	**71.06**	46.25	**57.04**	**53.75**
Six-Folder Cross Validation															
PointNet [7]	47.6	66.2	88.0	88.7	69.3	42.4	23.1	47.5	51.6	42.0	54.1	38.2	9.6	29.4	35.2
DGCNN [13]	56.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
RSNet [14]	56.47	66.45	92.48	92.83	78.56	32.75	34.37	51.62	**68.11**	59.72	60.13	16.42	**50.22**	44.85	52.03
SPGraph [15]	54.06	64.45	92.17	95.00	71.91	33.46	15.03	46.53	60.92	65.05	69.45	56.82	38.21	6.86	51.29
PointCNN [9]	62.74	-	-	-	-	-	-	-	-	-	-	-	-	-	-
HDGCN(Ours)	**66.85**	**76.11**	**94.06**	**97.30**	**82.03**	**55.27**	**47.26**	**54.73**	66.64	**73.82**	**72.03**	**63.91**	44.07	**56.96**	**60.98**

TABLE II

RESULTS ON PARIS-LILLE-3D BENCHMARK

Method	mIOU	ground	building	pole	bollard	trash can	barrier	pedestrian	car	natural
RF_MSSF [19]	56.28	99.25	88.63	47.75	67.27	2.31	27.09	20.61	74.79	78.83
MS3_DVS [18]	66.89	99.03	**94.76**	52.4	38.13	**36.02**	**49.27**	**52.56**	**91.3**	88.58
HDGCN(Ours)	**68.30**	**99.37**	92.98	**67.72**	**75.69**	25.68	44.72	37.09	81.92	**89.56**

Net++ and PointCNN are two state-of-the-art methods on 3D semantic segmentation. PointNet++ is the first to propose the hierarchical structure for point clouds. It uses symmetric functions including MLP and pooling to extract local information. However, MLP and pooling are not powerful enough for complex shape analysis. PointCNN is an improvement based on PointNet++. It proposed $\mathcal{X}$-transformation to achieve the property of invariance. However, $\mathcal{X}$-transformation is a rough estimation for the point order. Our proposed depthwise graph convolution followed by pointwise convolution is better than $\mathcal{X}$-transformation on achieving invariance. Given the same point clouds with different orders, depthwise graph convolution can ensure the only result which is very important for local shape analysis. Experiments show the effectiveness of our method.

IV. EXPERIMENTS

We evaluate our HDGCN in two datasets, Stanford Large-Scale 3D Indoor Spaces (S3DIS) and Paris-Lille-3D. When preprocessing, we use a sliding window to divide each scene into several blocks with a fixed size. For each block, we sample a fixed number of points. Given the balance between the model size and the classification accuracy, we sample 2048 points for both indoor and outdoor datasets. As some blocks contain more than 2048 points, we generate more than one sample with 2048 points for each block to include as much as points in our data. We implement our network in Tensorflow1.9 with two NVIDIA GTX1080Ti GPUs. The training time is about 8 hours for both datasets.

A. Stanford Large-Scale 3D Indoor Spaces

Stanford Large-Scale 3D Indoor Spaces (S3DIS) [20] is one of the largest datasets with semantic labels of point clouds. It contains six areas with several different scenes. Most of the state-of-the-art approaches on semantic segmentation evaluate in this dataset. We use the same metrics including mean IOU, mean accuracy and per class IOU as previous work to evaluate our model.

The block size is $1.5m \times 1.5m$ when preprocessing. We

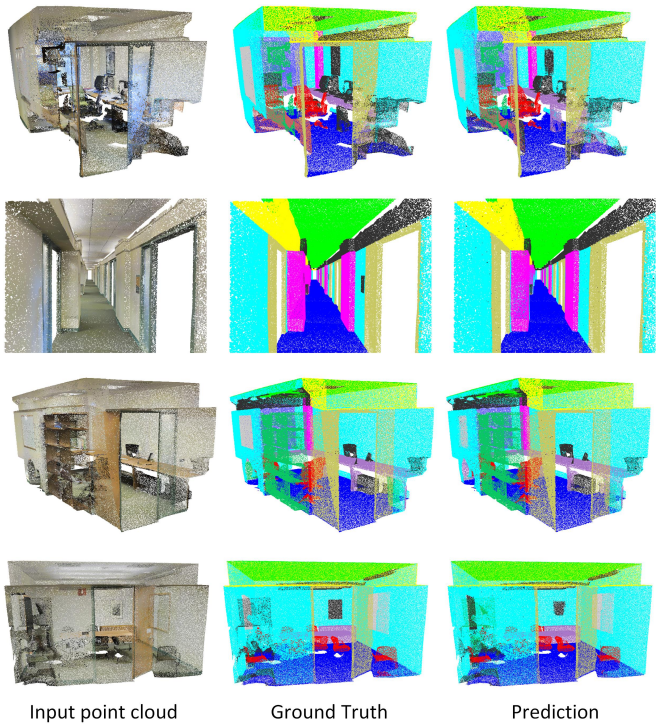

Fig. 5. Semantic segmentation results on S3DIS

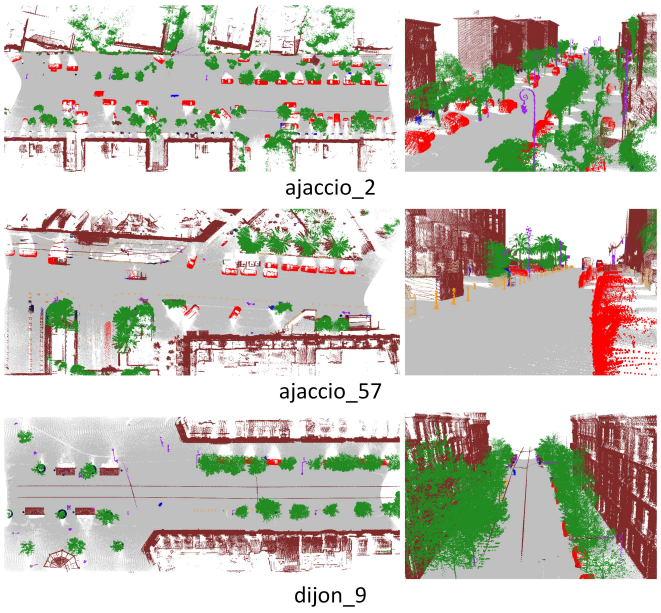

Fig. 6. Semantic segmentation results on Paris-Lille-3D benchmark. The benchmark provides three test scenes without ground truth, so we visualize the prediction of our network. The first column is the bird view of the scenes while the second column is a more detailed view.

use four downsampling DGConv blocks in the encoder and correspondingly four upsampling DGConv blocks in the decoder. The architecture is DGConv(2048,256)–DGConv(768,512)–DGConv(384,768)–DGConv(128,1024)–DGConv(128,1024)–DGConv(384,768)–DGConv(768,512)–DGConv(2048,256). Fully connected layers are used to get the final prediction. We choose the Adam optimizer with base learning rate 0.001 and batch size 16 (each GPU with 8). The learning rate declines by 20% every 5000 training steps.

We compare our approach with several state-of-the-art methods on semantic segmentation. The result is shown in TABLE I. Some previous methods test in Area 5 while others use the six-folder cross validation. For fair comparison, we provide both results of Area 5 and six-folder cross validation. Our method outperforms state-of-the-art methods in both strategies. Some test results are shown in Fig. 5. Categories with distinct features such as ceilings, floors and walls are predicted well. Results on chairs and tables are also satisfied in most scenes as they are separate from other objects. These improvements show that our method has the strong ability to capture local structures. DGCNN [13] is another graph CNN designed for point clouds. Our method outperforms it by a large margin. The experiment shows that our DGConv block has better ability to extract local features from point clouds. Also, DGCNN does not use a hierarchical structure which is very important for semantic segmentation.

B. Paris-Lille-3D

Paris-Lille-3D [18] is a recent available benchmark[1] which provides semantic labels of point clouds. Different from S3DIS, this dataset only contains xyz coordinates and intensity information without RGB attributes. The dataset provides four scenes acquired by Mobile Laser Scanning (MLS) as the training set and three scenes as the test set.

As Paris-Lille-3D is an outdoor dataset, we set the block size with $5m \times 5m$ which is larger than the block size used in S3DIS. Except this, we use the same architecture and parameter setting as S3DIS.

In TABLE II, we compare our method with others in this benchmark. The bird view in Fig. 6 shows that our method classifies most points correctly. For classes such as ground, buildings, natural and cars, the accuracy of our method is high. Specially, our method has a great improvement on poles and bollards (small poles). We argue that the improvement comes from both the local feature extraction and the hierarchical structure. As shown in Fig. 2, the hierarchical structure provides both global and local shape inforamtion while depthwise graph convolution extracts local features in each layer.

C. Discussion

Effectiveness of our method. Our proposed DGConv block has the strong ability to aggregate features from neighbors and transfer features to neighbors. This is an important characteristic for tasks such as semantic segmentation which needs to incorporate contextual information to enhance the consistency. The hierarchical structure increases the receptive field while providing shape information of different scales in different layers. The combination of the DGConv block and the hierarchical structure is beneficial for both extracting local features and considering global information.

Future work. Postprocessing algorithm such as conditional random field (CRF) can be used to enhance the consistency. Also, CRF can be formulated as recurrent neural network (RNN) [21] to construct an end-to-end framework. Additionally, although our input point clouds can use RGB attributes as extra channels, we think that image-based CNN can extract more semantic information from color and texture information while point clouds can provide rich geometric information. Fusion between point clouds and images is needed for a more accurate semantic map. We believe that this is a valuable research as the camera and the LIDAR are both parts of the autonomous vehicle. We leave these improvements for future work.

V. CONCLUSIONS

In this paper, we propose the DGConv block which is effective for extracting and aggregating local information. Depthwise graph convolution is designed to reduce memory consumption and allow more efficient use of model parameters. We combine the hierarchical structure and the DGConv block to extract local and global features of point clouds. Experiments show that our network achieves state-of-the-art results on 3D semantic segmentation datasets. We believe that our method is useful for generating the 3D semantic map which is important for the autonomous driving.

[1] http://npm3d.fr/paris-lille-3d

REFERENCES

[1] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3d shape recognition," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 945–953.

[2] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *IEEE CVPR*, vol. 1, no. 2, 2017, p. 3.

[3] A. Boulch, B. Le Saux, and N. Audebert, "Unstructured point cloud semantic labeling using deep segmentation networks." in *3DOR*, 2017.

[4] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese, "Segcloud: Semantic segmentation of 3d point clouds," in *3D Vision (3DV), 2017 International Conference on*. IEEE, 2017, pp. 537–547.

[5] G. Riegler, A. O. Ulusoy, and A. Geiger, "Octnet: Learning deep 3d representations at high resolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 3, 2017.

[6] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong, "O-cnn: Octree-based convolutional neural networks for 3d shape analysis," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, p. 72, 2017.

[7] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, vol. 1, no. 2, p. 4, 2017.

[8] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, 2017, pp. 5099–5108.

[9] Y. Li, R. Bu, M. Sun, and B. Chen, "Pointcnn," *arXiv preprint arXiv:1801.07791*, 2018.

[10] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

[11] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[12] M. Simonovsky and N. Komodakis, "Dynamic edgeconditioned filters in convolutional neural networks on graphs," in *Proc. CVPR*, 2017.

[13] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *arXiv preprint arXiv:1801.07829*, 2018.

[14] Q. Huang, W. Wang, and U. Neumann, "Recurrent slice networks for 3d segmentation of point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2626–2635.

[15] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," *arXiv preprint arXiv:1711.09869*, 2017.

[16] V. Vanhoucke, "Learning visual representations at scale," *ICLR invited talk*, 2014.

[17] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *arXiv preprint*, pp. 1610–02 357, 2017.

[18] X. Roynard, J.-E. Deschaud, and F. Goulette, "Paris-lille-3d: A large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification," *The International Journal of Robotics Research*, p. 0278364918767506, 2017.

[19] H. Thomas, J.-E. Deschaud, B. Marcotegui, F. Goulette, and Y. L. Gall, "Semantic classification of 3d point clouds with multiscale spherical neighborhoods," *arXiv preprint arXiv:1808.00495*, 2018.

[20] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3d semantic parsing of large-scale indoor spaces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1534–1543.

[21] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr, "Conditional random fields as recurrent neural networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1529–1537.