Semantic Segmentation-assisted Scene Completion for LiDAR Point Clouds

Xuemeng Yang[1], Hao Zou[1], Xin Kong[1], Tianxin Huang[1], Yong Liu[1,*], Wanlong Li[2], Feng Wen[2], and Hongbo Zhang[2]

***Abstract*— Outdoor scene completion is a challenging issue in 3D scene understanding, which plays an important role in intelligent robotics and autonomous driving. Due to the sparsity of LiDAR acquisition, it is far more complex for 3D scene completion and semantic segmentation. Since semantic features can provide constraints and semantic priors for completion tasks, the relationship between them is worth exploring. Therefore, we propose an end-to-end semantic segmentation-assisted scene completion network, including a 2D completion branch and a 3D semantic segmentation branch. Specifically, the network takes a raw point cloud as input, and merges the features from the segmentation branch into the completion branch hierarchically to provide semantic information. By adopting BEV representation and 3D sparse convolution, we can benefit from the lower operand while maintaining effective expression. Besides, the decoder of the segmentation branch is used as an auxiliary, which can be discarded in the inference stage to save computational consumption. Extensive experiments demonstrate that our method achieves competitive performance on SemanticKITTI dataset with low latency. Code and models will be released at https://github.com/jokester-zzz/SSA-SC.**

I. Introduction

Scene completion plays an important role in autonomous driving, which is a fundamental block of 3D scene understanding. In autonomous driving scenarios, LiDAR is the most commonly used 3D sensor. However, the point cloud collected by a LiDAR is inherently sparse due to its acquisition method, and can only collect data on object surface, which makes it more difficult for machines to infer and understand the scene than humans. Therefore, semantic scene completion task has been proposed to complete the entire scene from limited information, and segment the semantics. Fig. 1 is an illustration of outdoor point cloud scene completion.

Compared with scene completion of depth image, the point cloud scene completion contains much more input data, which will bring greater challenges. Furthermore, since the memory will grows cubically with the increasing with the input voxel resolution, the 3D convolutional neural network also requires noticeable overhead costs [1]. It was not until the emergence of sparse convolution [2] that this situation was alleviated. However, submanifold sparse convolution always maintains the sparsity of voxels, it is hard to be directly applied to the task of scene completion. So Zhang et al. [3] insert a "dense" deconvolution layer in the Sparse Convolutional Network (SCN) to generate new voxels. LMSCNet [4] tackles this problem by using a lightweight 2D convolutions followed by a 3D segmentation head block.

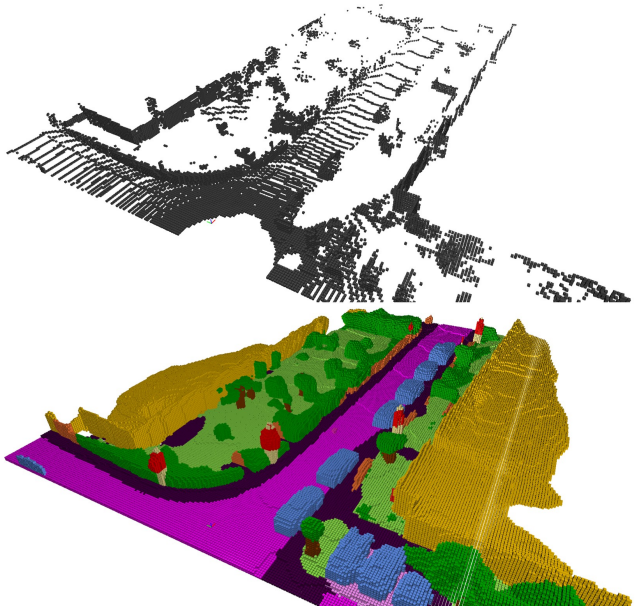

Fig. 1: Semantic scene completion on SemanticKITTI dataset. The input is the sparse raw point cloud and the output is the complete semantic scene predicted by our approach.

Inspired by [5] which adopts a completion network after semantic segmentation, we explore the relationship between semantic segmentation and scene completion tasks. For scene completion, semantic features are critical in recovering the complete shape of the specific objects and the entire scene. It's easy to infer the whole shape from an incomplete object by identifying it's category, which means the semantics essentially provide the constraints and priori for completion. Based on this insight, we believe that semantic segmentation tasks and scene completion tasks are inseparable, and semantic segmentation can provide potential semantic information for the scene completion task.

In this paper, we try to explore the combination of semantic segmentation and scene completion, and benefit from both the 2D and 3D convolutional neural network. The whole network consists of two parts including a 2D completion branch and an assistant 3D segmentation branch. Since 3D dense convolution consumes too much resources, and 3D sparse convolution is difficult to generate new

[1]The authors are with the Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, 310027, China. (Yong Liu* is the corresponding author, email: yongliu@iipc.zju.edu.cn)

[2]The authors are with Huawei Noah's Ark Lab, Beijing, China.

voxels, we use a 2D network on Bird's Eye View (BEV) to complete the scene. It is efficient and easy to diffuse features with 2D convolution. In addition, we introduce the features of the semantic segmentation branch as an auxiliary. We assume features from the semantic segmentation branch as source that can continuously deliver semantic features to the completion branch with combining the advantages of 2D and 3D networks from multi-view fusion. The main contributions of this paper are three-fold:

- We propose a novel semantic segmentation-assisted scene completion network that leverages the complementary information between BEV map and 3D voxels from multi-views.
- The proposed network can produce reasonable completion results with low latency and memory cost for outdoor 3D scenes by the aid of auxiliary semantic segmentation branch.
- Experiments on SemanticKITTI dataset show that our approach achieves the state-of-the-art performance. We rank the 3^{rd} in public semantic scene completion benchmark[1] and outperform all the published work in terms of completion metric (IoU).

II. Related Work

In this section, we introduce the development of point cloud segmentation and completion.

A. 3D semantic segmentation

Semantic segmentation on point clouds can be classified into three types: projection-based, voxel-based and point-based.

Projection-based works usually project raw point clouds onto various 2D planes. SqueezeSegs [6]–[8] and RangeNet++ [9] project the point cloud onto the spherical image. SalsaNet [10] organizes point clouds into BEV feature maps. PolarNet [11] proposes a polar BEV representation, achieving a balanced grid distribution. They all use 2D semantic segmentation networks as the backbone. Voxel-based methods split the raw point clouds into tightly arranged voxels and usually apply 3D convolutions to abstract features. Due to the sparsity, 3D sparse convolution [2] is widely used and 3D UNet-like [12] is often adopted as the backbone network for semantic segmentation. Cylinder3D [13] and Cylinder3D++ [14] use a cylindrical partition on the point cloud and design an Asymmetric Residual Block and a Dimension-decomposition based Context Modeling based on 3D sparse convolution, which can significantly reduce the computational cost. Zhang et al. [15] propose a deep fusion network architecture with a unique voxel-based "mini-PointNet" point cloud representation, which fully integrates the features of points and voxels. Some researchers continue the PointNets [16], [17] approach, processing on the original point cloud without intermediate representation. They aim to segment the point cloud by directly extracting features from 3D points hierarchically to capture the local or global context [18], or design a new convolution method [19], [20].

Among all these methods, the voxel-based method can be best combined with the scene completion task, as it is convenient to indicate whether the space is occupied or not.

B. 3D scene completion

Given a single frame point cloud or depth map as input, the scene completion work should predict $C + 1$ labels for each voxel in the 3D space, indicating whether it is occupied and what the semantic category is. The earliest scene completion work is based on depth image proposed by Song et al. [1]. They formulate an end-to-end 3D ConvNet model (SSCNet) for the volumetric scene completion and semantic labeling with a dilation-based 3D context module. In order to complete the scene, the entire network is composed of dense 3D convolution, which consumes a lot of resources. Most of the subsequent works are based on 3D convolution networks. ScanComplete [21] introduces a coarse-to-fine inference strategy to produce high-resolution output. SGC [3] benefits from 3D sparse convolution and adopts Spatial Group Convolutions for efficient processing at the cost of small performance degradation. SATNet [22] decomposes semantic scene completion tasks into 2D semantic segmentation and 3D scene completion, which are connected by a 2D-3D reprojection layer.

Until the emergence of the SemanticKITTI dataset [23], researchers turn their attention to the semantic scene completion of outdoor point clouds. LMSCNet [4] designs a lightweight network with a mix of 2D and 3D convolutions, using a 2D UNet backbone followed by a 3D segmentation head. In the case of using only occupied voxels as input, they achieve acceptable results. Rist et al. [24] produce a continuous scene representation instead of voxelization. This method can predict any position in the scene without the need for spatial discretization. S3CNet [25] proposes a multi-view fusion method that performs semantic completion of the scene in 2D and 3D respectively, and finally uses a dynamic voxel fusion to merge the results. Besides, they leverage LiDAR-based flipped truncated symbol distance function (fTSDF [1]) calculated from the spherical range image and point-wise normal vectors as spatial encoding. JS3C-Net [5] adds a SSCNet after semantic segmentation network for completion, and proposes a Point-Voxel Interaction (PVI) module for refinement. However, it cannot perform in real-time due to the cascade architecture.

In this paper, we propose an end-to-end network that completely uses the network to extract features without manually designed features. We design a 2D scene completion network assisted by a semantic segmentation branch, whose decoder part is discarded in the inference stage to achieve a fast speed while reserving a helpful encoder with 3D sparse convolution layers.

[1] https://competitions.codalab.org/competitions/22037#results

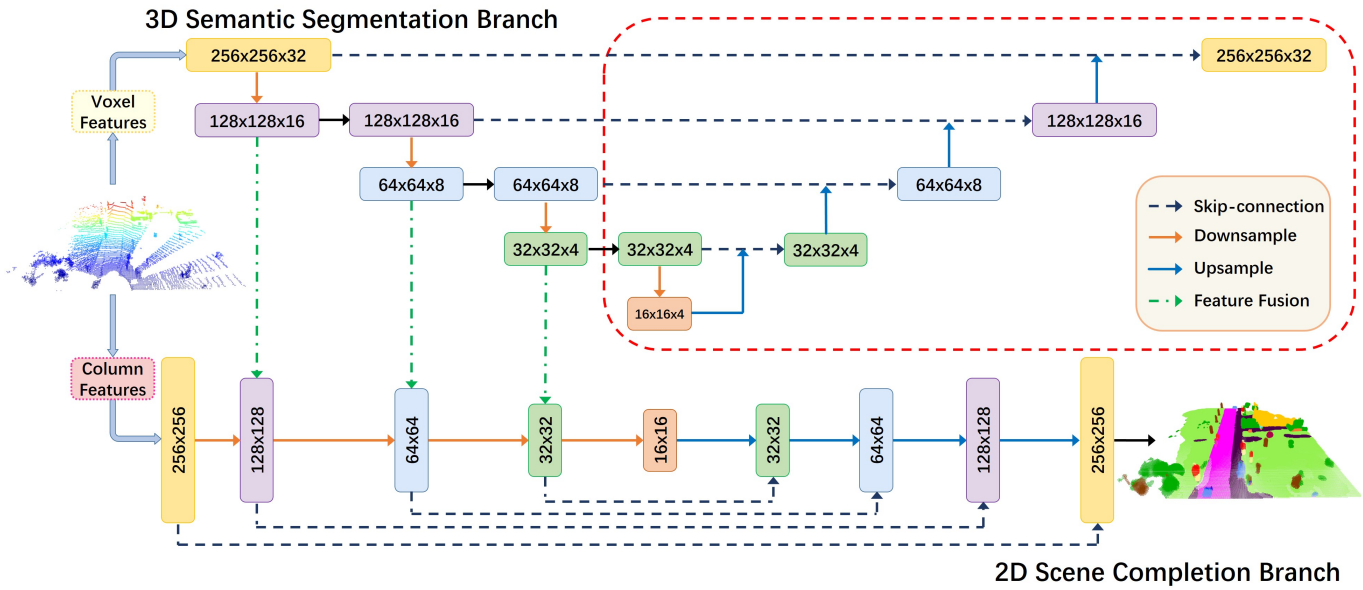

Fig. 2: The network structure of the proposed method. The upper part of the figure is an auxiliary 3D semantic segmentation branch, and the lower part is a 2D completion branch. Both branches follow the UNet structure and carry out four downsamplings. We merge the first three downsampling results from the semantic segmentation branch into the completion branch at the same level to supplement semantic information. The same color rectangles represent the same downsampling stage, and the numbers in the rectangles represent the feature resolution. In the inference stage, the part in the red dashed line can be discarded to further save memory. (Best viewed in color.)

III. Methodology

The problem we solve is to infer the entire scene from a single frame point cloud. In this work, we combine 2D and 3D convolutional networks, as illustrated in Fig. 2. We use a 2D network for scene completion since it is far less complex and a 3D segmentation network as an auxiliary branch for the completion branch. The input of the network is a point cloud P with points features f_p, which represents the coordinates of points and their corresponding point-wise information. The output is a label of $C+1$ categories for each voxel, where C is the number of semantic categories, indicating whether a voxel is free or occupied by a semantic category.

A. Scene Completion Branch

Since the 3D completion network often needs "dense" convolution for dilation, it consumes more resources. In contrast, 2D networks are more lightweight and convenient in diffusing features. Therefore, we adopt a 2D encoder-decoder architecture as the backbone. In order to adapt to the scene completion task, we carry out the Cartesian voxelization instead of the commonly used spherical projection in segmentation. Given the point cloud $P \in \mathbb{R}^{N\times 3}$ in the range of $[R_x, R_y, R_z]$, we voxelize the irregular points into voxels with a resolution of $L \times W \times H$.

Here, we adopt a top-down view to generate a BEV feature map with the size of $L \times W$. Since the BEV map is very similar to image, 2D CNN can be directly applied on it. As with some detection tasks [26], we fuse the features of each column along the z-axis. Specifically, we use a simple MLP to learn point features, followed by a max-pooling layer in each column to get the preliminary column feature. After a feature dimension reduction layer, we obtain the column feature.

$$f_{x,y} = \mathcal{A}_1 \left\{ \underset{p\in V_{x,y}}{\mathrm{MAX}} \left(\mathcal{M}(f_p) \right) \right\}. \tag{1}$$

$\mathcal{A}_1$ is a dimension reduction layer including a simple Linear layer followed by a ReLU function. $\mathcal{M}$ represents an MLP that only contains fully connected layers, batch normalization and ReLU. $V_{x,y}$ denotes the $(x,y)^{th}$ column. f_p represents the feature of point p and can be defined flexibly. In our implementation, the point feature f_p is defined as:

$$f_p = (\Delta x, \Delta y, \Delta z, x, y, z, r), \tag{2}$$

$(\Delta x, \Delta y, \Delta z)$ is the coordinate difference between the point p and the center of the voxel it locates while r denotes the reflection intensity. So far, we have obtained a BEV feature of size $C_f \times L \times W$ that can represent the entire scene, where C_f is the feature dimension. Then, we input the obtained BEV features into the 2D completion branch.

Unlike LMSCNet [4] using a 3D segmentation head, we directly output a tensor of size $L \times W$, which encodes the prediction of each voxel along the height of that location. By reshaping the output tensor, the prediction of each voxel in the scene can be acquired. In order to fully diffuse the features, we use a series of 2D convolution layers to expand the receptive field. The entire network is downsampled four times, and each time the resolution size is reduced by 2 shown in Fig.2. Skip connections with concatenation are used at the same time. However, aggregating features with a 2D

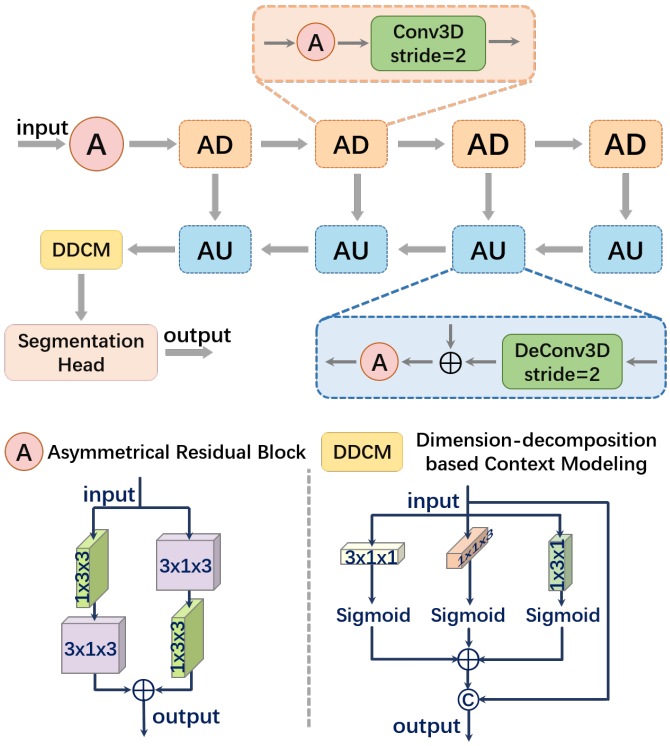

Fig. 3: The structure of the assistant semantic segmentation network, where AD stands for asymmetrical downsample block and AU stands for asymmetrical upsample block.

network inevitably lacks some information along the z-axis, so we supplement additional features with the aid of the semantic segmentation branch.

B. Semantic Segmentation Branch

To supplement the height information lost in the 2D convolution, we introduce a 3D sparse convolutional branch. The sparse convolution can aggregate the voxel features effectively and continuously provide semantic features for the completion branch hierarchically. These voxel features from the segmentation branch help the completion task in extending voxels and predicting their semantic categories.

Similar to the operations in *Section III-A*, the input of the branch is aggregated voxel features. Based on the Cartesian voxelization, the point-wise features obtained from the same MLP in Eq.1 are reassigned to obtain voxel features. With a max pooling performed in each voxel and another dimension reduction layer $\mathcal{A}_2$ (a Linear layer followed by a ReLU), we obtain the input of the 3D branch.

$$f_{x,y,z} = \mathcal{A}_2 \left\{ \underset{p \in V_{x,y,z}}{\text{MAX}} \left(\mathcal{M}(f_p) \right) \right\}, \tag{3}$$

where $V_{x,y,z}$ denotes the $(x, y, z)^{th}$ voxel. Reusing the same MLP but only adopting different dimension reduction layers can save some resource consumption. Now, we obtain a feature of size $C_f \times L \times W \times H$ that represents the entire scene, where C_f is the feature dimension and is consistent with Section III-A. Then, we input the obtained features into the 3D segmentation branch.

To better supplement features for the completion branch with each corresponding layer, we choose the same encoder-decoder structure in this semantic segmentation branch modified from the 3D UNet in Cylinder3D [13], which is implemented with 3D sparse convolution. As shown in Fig. 3, both the asymmetrical downsample and upsample blocks in the network contain Asymmetrical Residual Block. The usage of the convolution with a $3 \times 1 \times 3$ kernel followed by a $1 \times 3 \times 3$ kernel is equivalent to sliding a two-layer network with a conventional $3 \times 3 \times 3$ kernel, but can save 33% of memory consumption. The asymmetrical downsample block also halves the resolution size each time, so that the tensors output from the same level of the 3D and 2D network have the same size in terms of length and width. Dimension-decomposition based Context Modeling Block divides high-level context information into low-level features in three dimensions (length, width, height), and aggregates all three low-level activations to obtain features representing a complete context. We use the most frequently appeared semantic label in the raw occupied voxel as supervision.

We draw semantic features from the encoder part and input them into the completion branch. Specifically, we stack the 3D features along the z-axis and reduce the stacked feature dimension so that it can have the same dimension $(C_s \times L \times W)$ as the 2D feature of the same resolution. Then, we concatenate it with the completion feature in the C_s dimension, which can provide features with information along the z-axis for the completion task. It is worth noting that we only use features from the encoder part and the decoder part can be discarded to further reduce the burden of GPU memory and calculation in the inference stage.

C. Loss

In the experiment, both lovasz loss [27] and cross-entropy loss are used for the two branches. Lovasz loss is a method for directly optimizing the mean intersection-over-union (mIoU) metric, which is defined as:

$$Loss_{lovasz} = \frac{1}{|C|} \sum_{c \in C} J(e(c)), \tag{4}$$

where J is the lovasz extension of IoU and denotes a piece-wise linear function with a global minimum, and $e(c)$ is the vector of errors for class c. Cross-entropy loss is widely used and optimizes the accuracy:

$$Loss_{CE} = -\sum_{i} y_i \log \hat{y}_i. \tag{5}$$

$\hat{y}_i$ and y_i are the corresponding predicted and ground truth probability. The total loss is:

$$Loss_{all} = \sigma_1 Loss_{seg} + \sigma_2 Loss_{com}. \tag{6}$$

In our work, we set $\sigma_1 = \sigma_2 = 0.5$, and the loss for two branches are:

$$\begin{aligned} Loss_{seg} &= Loss_{lovasz} + Loss_{CE}, \\ Loss_{com} &= Loss_{lovasz} + Loss_{CE}. \end{aligned} \tag{7}$$

In the ablation study, we verify the effectiveness of lovasz loss in the completion branch.

TABLE I: Comparison of published methods on the official SemanticKITTI [23] benchmark. Our network surpasses all the published methods in terms of completion metrics (IoU), and ranks 3^{rd} on the semantic segmentation metrics (mIoU). (* originate from [4]. The last column of data comes from their paper, the data in brackets are reproduced on our device.)

Approach	IoU	road	sidewalk	parking	other-ground	building	car	truck	bicycle	motorcycle	other-vehicles	vegetation	trunk	terrain	person	bicyclist	motorcyclist	fence	pole	traffic-sign	mIoU	FPS
*SSCNet [1]	29.8	27.6	17.0	15.6	6.0	20.9	10.4	1.8	0.0	0.0	0.1	25.8	11.9	18.2	0.0	0.0	0.0	14.4	7.9	3.7	9.5	**56.90**
*SSCNet-full [1]	50.0	51.2	30.8	27.1	6.4	34.5	24.3	1.2	0.5	0.8	4.3	35.3	18.2	29.0	0.3	0.3	0.0	19.9	13.1	6.7	16.1	45.94
*TS3D [28]	29.8	28.0	17.0	15.7	4.9	23.2	10.7	2.4	0.0	0.0	0.2	24.7	12.5	18.3	0.0	0.1	0.0	13.2	7.0	3.5	9.5	9.79
*TS3D+DNet [23]	25.0	27.5	18.5	18.9	6.6	22.1	8.0	2.2	0.1	0.0	4.0	19.5	12.9	20.2	2.3	0.6	0.0	15.8	7.6	7.0	10.2	8.72
*TS3D+DNet+SATNet [23]	50.6	62.2	31.6	23.3	6.5	34.1	30.7	4.9	0.0	0.0	0.1	40.1	21.9	33.1	0.0	0.0	0.0	24.1	16.9	6.9	17.7	1.27
LMSCNet [4]	55.3	64.0	33.1	24.9	3.2	38.7	29.5	2.5	0.0	0.0	0.1	40.5	19.0	30.8	0.0	0.0	0.0	20.5	15.7	0.5	17.0	21.28 (8.51)
LMSCNet-singlescale [4]	56.7	64.8	34.7	29.0	4.6	38.1	30.9	1.5	0.0	0.0	0.8	41.3	19.9	32.1	0.0	0.0	0.0	21.3	15.0	0.8	17.6	-
Local-DIFs [24]	57.7	67.9	42.9	**40.1**	11.4	40.4	34.8	4.4	3.6	2.4	4.8	42.2	26.5	39.1	2.5	1.1	0.0	29.0	21.3	17.5	22.7	-
JS3C-Net [5]	56.6	64.7	39.9	34.9	**14.1**	39.4	33.3	**7.2**	14.4	8.8	12.7	43.1	19.6	40.5	8.0	5.1	0.4	30.4	18.9	15.9	23.8	1.73 (1.20)
S3CNet [25]	45.6	42.0	22.5	17.0	7.9	**52.2**	31.2	6.7	**41.5**	**45.0**	**16.1**	39.5	**34.0**	21.2	**45.9**	**35.8**	**16.0**	**31.3**	**31.0**	**24.3**	**29.5**	1.82
Ours	**58.8**	**72.2**	**43.7**	37.4	10.9	43.6	**36.5**	5.7	13.9	4.6	7.4	**43.5**	25.6	**41.8**	4.4	2.6	0.7	30.7	14.5	6.9	23.5	20.04

TABLE II: Ablation experiment on SemanticKITTI dataset. The results are conducted on the validation set.

2D	3D	Com CE / lvz	Seg CE / lvz	IoU	Precision	Recall	mIoU	road	sidewalk	parking	other-ground	building	car	truck	bicycle	motorcycle	other-vehicles	vegetation	trunk	terrain	person	bicyclist	motorcyclist	fence	pole	traffic-sign
✓	×	✓ / ×	× / ×	57.44	82.15	65.64	20.90	72.79	43.36	24.98	2.26	39.92	41.77	17.91	0.00	0.00	4.98	41.15	15.02	**49.68**	0.00	0.00	0.00	14.01	25.89	3.37
✓	×	✓ / ✓	× / ×	56.43	79.17	66.28	22.17	72.13	43.55	**31.09**	2.10	39.87	45.22	21.35	6.50	4.55	13.90	40.36	17.55	48.27	3.76	0.00	0.00	13.21	14.09	3.72
✓	✓	✓ / ×	✓ / ✓	58.30	**82.78**	66.34	22.75	72.94	43.93	23.94	2.71	40.70	44.48	19.28	4.07	2.83	9.66	**42.46**	21.24	47.34	2.04	0.72	0.00	**17.39**	**28.02**	8.48
✓	✓	✓ / ×	× / ×	**58.61**	79.64	68.94	22.88	**73.31**	43.50	23.33	2.67	40.05	44.91	25.57	3.18	3.93	9.62	41.78	18.15	49.22	1.99	0.33	0.00	15.06	27.95	**10.10**
✓	✓	✓ / ✓	× / ×	57.90	80.35	67.45	23.96	73.11	43.70	24.27	2.85	41.09	46.74	29.54	7.93	**8.09**	**19.84**	41.62	21.86	49.77	5.85	1.34	0.00	14.14	18.24	5.26
✓	✓	✓ / ✓	✓ / ✓	58.25	78.49	**69.31**	**24.54**	72.81	**44.31**	21.09	**4.10**	**41.48**	**46.97**	**39.65**	**9.18**	7.41	19.10	41.86	**21.98**	49.45	**6.32**	**3.17**	0.00	15.20	17.78	4.40

IV. Experiments

A. Dataset and Metrics

Dataset. We test our method on the public semantic scene completion benchmark SemanticKITTI [23]. SemanticKITTI is a large-scale LiDAR point cloud dataset with a point-wise annotation collected by a single Velodyne HDL-64E laser scanner. The ground truth semantic labels of the scene completion task are composed of multiple consecutive point cloud frames with annotations. Following the official protocol, we select a single frame of the raw point cloud in the range of $[0 \sim 51.2m, -25.6 \sim 25.6m, -2 \sim 4.4m]$ as input, and divide it by 0.2m to obtain voxels with a resolution of $256 \times 256 \times 32$. The output is the completed scene in the same area. The dataset contains 22 sequences with 19 semantic categories for training and testing. We use Sequences 0-7, 9-10 (3834 scans) for training, Sequence 8 (815 scans) for validation, and Sequences 11-21 (3901 scans) for testing.

Metrics. We follow the regulations set by Song et al. [1] to calculate IoU for scene completion, representing the completion of the scene (not involving semantics), and mIoU for semantic scene completion to measure the semantic segmentation performance over 19 classes of a completed scene. The metrics for semantic segmentation measurement mIoU is defined as:

$$mIoU = \frac{1}{C} \sum_{c=1}^{C} \frac{TP_c}{TP_c + FP_c + FN_c} \tag{8}$$

where TP_c, FP_c, FN_c denote the number of true positive, false positive, and false negative predictions for class c respectively, and C denotes the number of classes.

B. Implementation Details

We augment the point cloud during training by randomly x-y flipping. We adopt the Adam optimizer [29] with a learning rate of 0.001 ($\beta_1 = 0.9$, $\beta_2 = 0.999$) for training, and each epoch is reduced by 2%. All experiments are conducted on a single Nvidia GTX 1080 Ti with 11GB memory with batch size 2.

C. Quantitative Results and Analysis

In this experiment, we submit the results of our work to the official evaluation server. The results of our method and the state-of-the-art methods are shown in Table. I. It is worth noting that our method surpasses all previously published works in terms of completion metrics (IoU). By the time of submission, our method ranks 2^{nd} in completion metrics (IoU) and 3^{rd} in semantic completion metrics (mIoU) on the SemanticKITTI benchmark. In addition, we report the

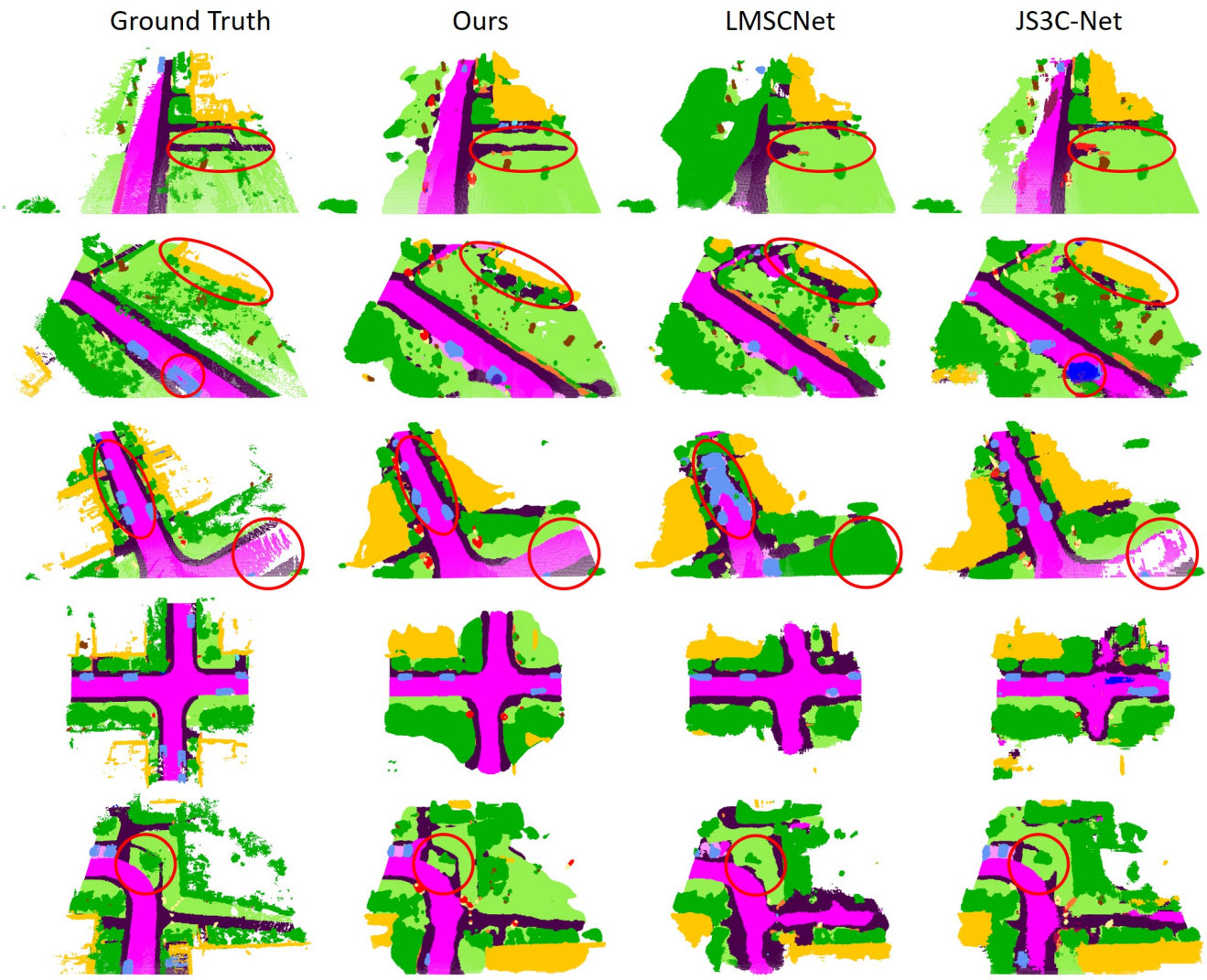

Fig. 4: Comparison of qualitative results with other recent works. Experiments conducted on SemanticKITTI validation set.

single scan inference latency on the entire validation split. Our network can reach a speed of 20.04 FPS with 2629 MB GPU memory when batch size is 1, which is much faster than other methods with comparable performance. In Table. I, the numbers in brackets are measured on our device with batch size 1 using the official code.

Compared with the recently published voxel-based method LMSCNet [4], we have better results due to the integration of point cloud information. We have advantages on both IoU and mIoU, especially for mIoU, we have increased by 38.2%. Compared with other integrated point cloud methods, we still have some advantages such as inference time. JS3C-Net [5] performs semantic segmentation on the point cloud before scene completion. Although it has obtained good mIoU results, it cannot achieve real-time results in inference time. S3CNet [25] has achieved outstanding results in semantic scene completion, thanks to the geometric-aware loss which makes it have a very good performance on small objects such as *bicycle* and *motorcycle*. Nevertheless, it is not satisfied in terms of completion performance and inference time. Our method has better performance on "plane" categories, such as *road* and *sidewalk*. It is guessed that the 2D completion network has a better feature extraction effect on these categories and can better expand these features to the surroundings voxels.

D. Qualitative Results

We use the pretrained model and the official code of LMSCNet[2] and JS3C-Net[3], and visualize the results in comparison with our results on SemanticKITTI validation set in Fig. 4. It can be seen that we do have obvious advantages in predicting the "plane" categories, which also verifies the metrics in Tab. I. For some difficult samples, the results obtained by other methods are somewhat distorted, but our results can still get recognizable scenes. In addition, in terms

[2]https://github.com/cv-rits/LMSCNet
[3]https://github.com/yanx27/JS3C-Net

of the completion performance, we can also make up a relatively complete scene better than other methods.

E. Ablation Study

In this part, we perform ablation studies on each part of the network to verify the effectiveness of the proposed method. The experiment results are shown in Tab. II. All experiments are trained on the training set, and the results are evaluated on the validation set.

The input for all experiments is point clouds. It can be seen that the baseline of our 2D UNet has already achieved good performance, especially for the "plane" categories which indicate that the 2D network is indeed good at feature diffusion. The addition of lovasz loss on this basis has further improved segmentation performance, because it can directly optimize mIoU, though it is not helpful for the completion metrics. Adding an assistant 3D branch on the baseline (no matter supervised by both lovasz loss and cross entropy loss or not) the network retains the ability to segment "plane" categories and also has a better effect on the segmentation of small objects. At the same time, the 3D structure helps to improve the performance of the completion. Finally, the semantic segmentation branch and lovasz loss are added on the baseline together, and the best results are obtained. Compared with the method without segmentation loss supervision, the proposed method obtains better performance because the semantic segmentation branch can provide semantic features instead of only a 3D structure. Furthermore, lovasz loss is also used to optimize the segmentation performance, they can promote each other with the semantic feature extraction and propagation. The lovasz loss allows the network to further improve the effect of small objects and accelerate the convergence, but at the same time, it seems to cause a decline on the completion metrics.

V. CONCLUSIONS

In this paper, we propose a novel network for scene completion benefits from both 2D and 3D networks. The efficient 2D branch is used for completion. The 3D segmentation branch based on sparse convolution is to provide semantic features for the completion branch, bringing improvements in both completion and segmentation. The decoder of the segmentation branch can be discarded during inference, which will not increase too much computational burden. We also use lovasz loss to improve the network effect and accelerate convergence. Finally, our experiments show that the proposed model has real-time inference speed, and set a state-of-the-art performance on the SemanticKITTI dataset.

REFERENCES

[1] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic scene completion from a single depth image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1746–1754. 1, 2, 5

[2] B. Graham, M. Engelcke, and L. Van Der Maaten, "3d semantic segmentation with submanifold sparse convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9224–9232. 1, 2

[3] J. Zhang, H. Zhao, A. Yao, Y. Chen, L. Zhang, and H. Liao, "Efficient semantic scene completion network with spatial group convolution," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 733–749. 1, 2

[4] L. Roldão, R. de Charette, and A. Verroust-Blondet, "Lmscnet: Lightweight multiscale 3d semantic completion," in *3DV 2020-International Virtual Conference on 3D Vision*, 2020. 1, 2, 3, 5, 6

[5] X. Yan, J. Gao, J. Li, R. Zhang, Z. Li, R. Huang, and S. Cui, "Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion," *arXiv preprint arXiv:2012.03762*, 2020. 1, 2, 5, 6

[6] B. Wu, A. Wan, X. Yue, and K. Keutzer, "Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud," *ICRA*, 2018. 2

[7] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud," in *ICRA*, 2019. 2

[8] C. Xu, B. Wu, Z. Wang, W. Zhan, P. Vajda, K. Keutzer, and M. Tomizuka, "Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation," *arXiv preprint arXiv:2004.01803*, 2020. 2

[9] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet++: Fast and Accurate LiDAR Semantic Segmentation," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019. 2

[10] E. E. Aksoy, S. Baci, and S. Cavdar, "Salsanet: Fast road and vehicle segmentation in lidar point clouds for autonomous driving," in *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 926–932. 2

[11] Y. Zhang, Z. Zhou, P. David, X. Yue, Z. Xi, B. Gong, and H. Foroosh, "Polarnet: An improved grid representation for online lidar point clouds semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[12] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: learning dense volumetric segmentation from sparse annotation," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2016, pp. 424–432. 2

[13] H. Zhou, X. Zhu, X. Song, Y. Ma, Z. Wang, H. Li, and D. Lin, "Cylinder3d: An effective 3d framework for driving-scene lidar semantic segmentation," *arXiv preprint arXiv:2008.01550*, 2020. 2, 4

[14] X. Zhu, H. Zhou, T. Wang, F. Hong, Y. Ma, W. Li, H. Li, and D. Lin, "Cylindrical and asymmetrical 3d convolution networks for lidar segmentation," *arXiv preprint arXiv:2011.10033*, 2020. 2

[15] F. Zhang, J. Fang, B. Wah, and P. Torr, "Deep fusionnet for point cloud semantic segmentation," in *ECCV*, vol. 2, 2020, p. 6. 2

[16] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660. 2

[17] 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. 2

[18] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "Randla-net: Efficient semantic segmentation of large-scale point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 108–11 117. 2

[19] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "Spidercnn: Deep learning on point sets with parameterized convolutional filters," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 87–102. 2

[20] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "Kpconv: Flexible and deformable convolution for point clouds," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6411–6420. 2

[21] A. Dai, D. Ritchie, M. Bokeloh, S. Reed, J. Sturm, and M. Nießner, "Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4578–4587. 2

[22] S. Liu, Y. Hu, Y. Zeng, Q. Tang, B. Jin, Y. Han, and X. Li, "See and think: Disentangling semantic scene completion," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018, pp. 261–272. 2

[23] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "Semantickitti: A dataset for semantic scene understanding of lidar sequences," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 9297–9307. 2, 5

[24] C. B. Rist, D. Emmerichs, M. Enzweiler, and D. M. Gavrila, "Semantic scene completion using local deep implicit functions on lidar data," *arXiv preprint arXiv:2011.09141*, 2020. 2, 5

[25] R. Cheng, C. Agia, Y. Ren, X. Li, and L. Bingbing, "S3cnet: A sparse semantic scene completion network for lidar point clouds," *arXiv preprint arXiv:2012.09242*, 2020. 2, 5, 6

[26] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 697–12 705. 3

[27] M. Berman, A. Rannen Triki, and M. B. Blaschko, "The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4413–4421. 4

[28] M. Garbade, Y.-T. Chen, J. Sawatzky, and J. Gall, "Two stream 3d semantic scene completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0. 5

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 5