PRNet: Point-Range Fusion Network for Real-Time LiDAR Semantic Segmentation

Xiaoyan Li[1,3*], **Gang Zhang**[2*†], **Tao Jiang**[3], **Xufen Cai**[4] and **Zhenhua Wang**[2]
[1]University of Chinese Academy of Sciences
[2]Damo Academy, Alibaba Group
[3]Institute of Computing Technology, Chinese Academy of Sciences
[4]Beijing School
xiaoyan.li@vipl.ict.ac.cn, zhanggang11021136@gmail.com, wave.leaf27@gmail.com
caixufen0402@outlook.com, zhwang.me@gmail.com

Abstract

Accurate and real-time LiDAR semantic segmentation is necessary for advanced autonomous driving systems. To guarantee a fast inference speed, previous methods utilize the highly optimized 2D convolutions to extract features on the range view (RV), which is the most compact representation of the LiDAR point clouds. However, these methods often suffer from lower accuracy for two reasons: 1) the information loss during the projection from 3D points to the RV, 2) the semantic ambiguity when 3D points labels are assigned according to the RV predictions. In this work, we introduce an end-to-end point-range fusion network (PRNet) that extracts semantic features mainly on the RV and iteratively fuses the RV features back to the 3D points for the final prediction. Besides, a novel range view projection (RVP) operation is designed to alleviate the information loss during the projection to the RV, and a point-range convolution (PRConv) is proposed to automatically mitigate the semantic ambiguity during transmitting features from the RV back to 3D points. Experiments on the SemanticKITTI and nuScenes benchmarks demonstrate that the PRNet pushes the range-based methods to a new state-of-the-art, and achieves a better speed-accuracy trade-off.

1 Introduction

LiDAR semantic segmentation provides a crucial point-level perception of the surrounding environments for applications, *e.g.* autonomous driving and moving robots. Such applications require the LiDAR semantic segmentation model to run in real-time to support timely downstream object detection and planning. However, the LiDAR point clouds come in an unstructured and sparse format, which brings challenges for efficient and effective processing. Different data representations (the 3D point, voxel, and range view) have been used to accomplish this task, which further results in differences in efficiency and accuracy as shown in Figure 1.

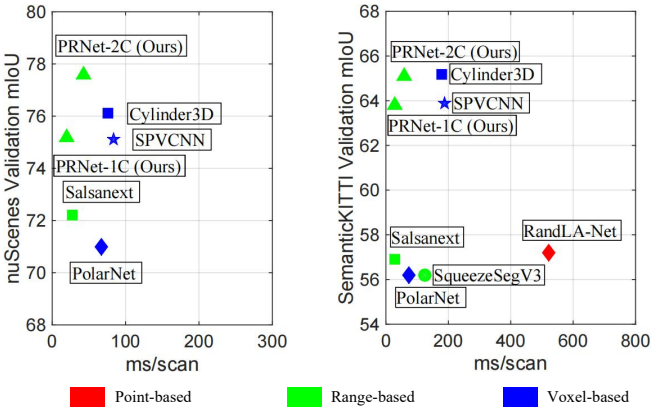

Figure 1: mIoU vs. runtime on the validation set of both SemanticKITTI and nuScenes. All methods are trained and evaluated based on the official code. The experiments are conducted with PyTorch FP32 on NVIDIA RTX 2080Ti GPU. Note that PRNet-2C has twice as many RV feature channels as PRNet-1C. The definition of mean Intersection over Union (mIoU) is given in the Section 4.1.

The point-based methods [Qi *et al.*, 2017a; Qi *et al.*, 2017b; Thomas *et al.*, 2019; Hu *et al.*, 2020] directly consume the unstructured 3D point clouds, which are the most complete sources of 3D information. However, as shown in Figure 1, point-based methods run slowly, since it has to adopt inefficient components for searching neighbors and aggregating features within unstructured points.

The latter voxel-based methods [Choy *et al.*, 2019; Tang *et al.*, 2020; Cheng *et al.*, 2021; Zhang *et al.*, 2020; Zhu *et al.*, 2021] discretize the point clouds into voxel cells and utilize the 2D or 3D convolution networks for feature extraction. These methods achieve relatively higher performance than point-based methods, but their efficiency (speed) still acts as a significant bottleneck for real-world applications.

The range-based methods adopt the most compact representation of the LiDAR point clouds, namely the RV representation, to ensure their efficiency. The previous range-based methods [Cortinhal *et al.*, 2020; Milioto *et al.*, 2019; Xu *et al.*, 2020; Razani *et al.*, 2021] first project 3D points

*Equal contribution
†Corresponding author

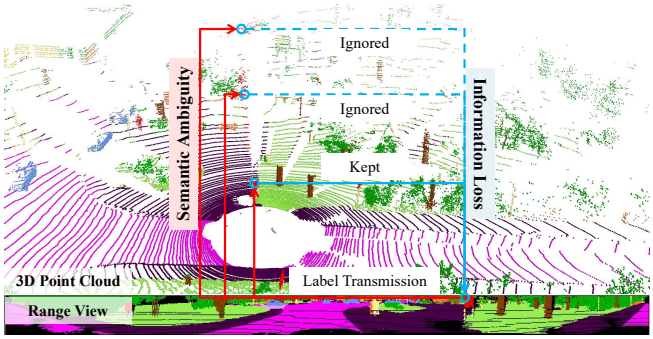

Figure 2: The information loss during the projection from 3D points to the RV: for the points of *traffic-sign* and *vegetation* in the same RV pixel, existing range-based methods only keep the nearest *traffic-sign* point. The semantic ambiguity from the RV predictions back to 3D points labels: the occluded *vegetation* points are assigned by the label of *traffic-sign*. The results are acquired by the official SalsaNext with kNN as post-processing.

to the RV, and then perform the 2D semantic segmentation on the RV with the well-optimized 2D convolution networks, and finally transmit the 2D RV labels back to 3D points. Despite their efficiency, their performance is much poorer than the voxel-based counterparts. We analyze two major reasons in Figure 2 and propose to solve these issues as the following.

First, the information loss during the projection from 3D points to the RV is nonnegligible. As shown in Figure 2, multiple LiDAR points may be projected to the same RV pixel, but existing methods only keep the nearest point. For example, SalsaNext [Cortinhal *et al.*, 2020] drops a significant proportion of 22% points on the SemanticKITTI dataset. To better encode the RV features, a novel range view projection (RVP) operation is proposed to aggregate features of the points in the same RV pixel through max-pooling. Moreover, the features of the RV and points are fused for the final per-point prediction to further minimize the information loss.

Second, the semantic ambiguity results in the wrong predictions, when the RV labels are transmitted back to the 3D points. The RV squeezes the distance r dimension and the 3D points far away in the 3D space may correspond to adjacent or even the same pixel in the RV space, which implies that these 3D points would be probably assigned to the same label. To ameliorate this problem, existing methods adopt k-Nearest-Neighbor (kNN) as post-processing, voting for the final prediction. But sometimes the kNN also fails, as shown in Figure 2, the occluded *vegetation* regions are assigned by the label of *traffic-sign*. To address this issue fundamentally, we propose to transmit the RV features instead of labels back to 3D points and introduce a novel point-range convolution (PRConv), which transmits the RV features with dynamic per-point hyper-parameters and allows distinct 3D point features although they are corresponding to the same RV pixel.

In summary, we propose an end-to-end point-range fusion network (PRNet) for real-time LiDAR semantic segmentation. Different from the previous range-based methods that predict 2D RV semantic labels, the proposed method first projects the features from 3D points to the RV for the latter feature extraction and then fuses the RV features back to 3D points for the final per-point prediction. In this framework, the RVP operation aims to alleviate the information loss during the projection from 3D points to the RV, while the PRConv is proposed to mitigate the semantic ambiguity during the feature transmission from the RV back to 3D points. Experiments on the SemanticKITTI and nuScenes benchmarks show that the proposed PRNet outperforms existing range-based methods by a large margin. Compared with the top-ranking methods, the PRNet achieves a better speed-accuracy trade-off.

2 Related Work

Point-based Methods. The point-based methods directly consume raw point clouds without any quantization, but their performance and efficiency are relatively worse. The pioneering work, PointNet [Qi *et al.*, 2017a], adopts the shared multilayer perception (MLP) to extract per-point features and the global max-pooling to integrate global features. To learn richer local structures, many subsequent works have been introduced. PointNet++ [Qi *et al.*, 2017b] proposes the stacked set abstraction layers to learn hierarchical point features. KPConv [Thomas *et al.*, 2019] proposes a novel spatial kernel-based point convolution to extract local structure. However, since the farthest point sampling (FPS) widely used in these networks is inefficient in both computation and memory cost, these methods are limited to indoor scenes and cannot be directly extended to autonomous driving with a large number of 3D points. RandLA-Net [Hu *et al.*, 2020] replaces the FPS with random sampling to improve efficiency, but it is still far away from real-time processing.

Voxel-based Methods. The voxel-based methods quantize the 3D points into structured voxels, where the mature 2D or 3D convolution networks are applied. To reduce the computation and memory cost, sparse convolution [Choy *et al.*, 2019] is applied only on these non-empty voxels. Considering that the voxelization brings quantization errors, the following works [Tang *et al.*, 2020; Cheng *et al.*, 2021; Xu *et al.*, 2021] fuse the voxel features with the fine-grained point features. SPVNAS [Tang *et al.*, 2020] further adopts neural architecture search and achieves better results with lower computation cost. AF2S3Net [Cheng *et al.*, 2021] designs the attentive feature fusion module (AF2M) and adaptive feature selection module (AFSM) to efficiently extract local and global structures simultaneously. Recent RPVNet [Xu *et al.*, 2021] fuses the features from the points, voxel, and RV in a single framework to alleviate quantization errors, and achieves the best results on the SemanticKITTI and nuScenes benchmarks. Besides, different 3D space partition strategies are proposed. PolarNet [Zhang *et al.*, 2020] uses the polar grid representation and Cylinder3D [Zhu *et al.*, 2021] follows the cylindrical partition. Although the voxel-based methods dominate the LiDAR semantic segmentation benchmarks, they cannot run in real-time on mobile platforms.

Range-based Methods. The range-based methods are more attractive because they are fast and easy-deployed by utilizing highly optimized 2D convolutions. These methods first project 3D points to the RV space and then perform semantic segmentation on the RV. RangeNet++ [Milioto *et*

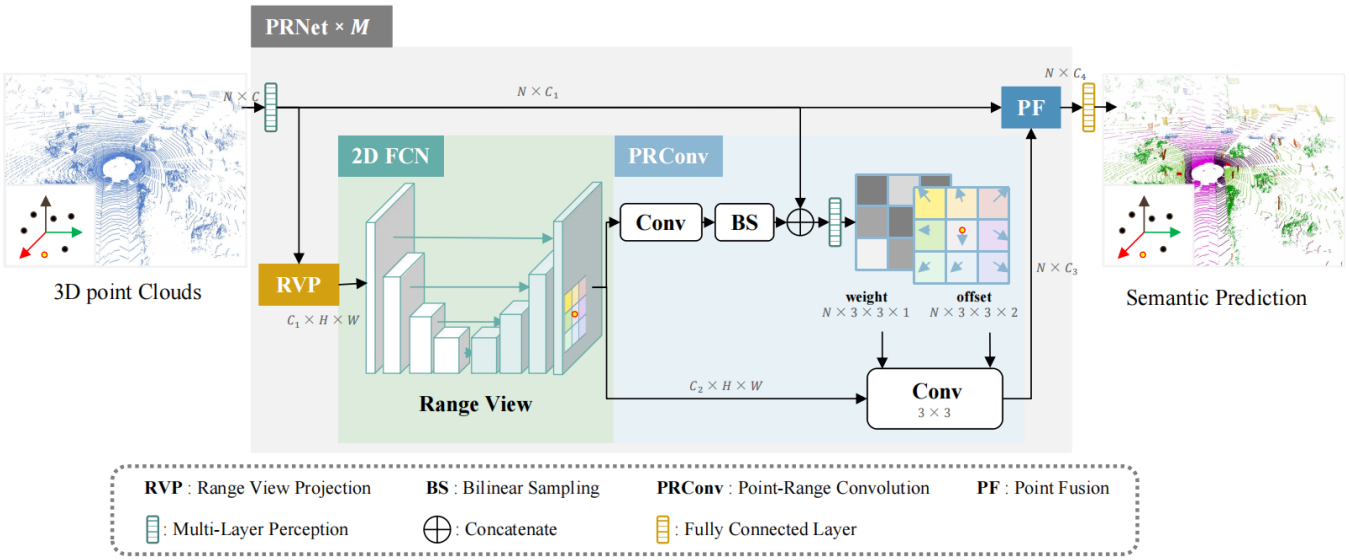

Figure 3: The point-range fusion network (PRNet). It takes the 3D point features as input and extracts semantic features mainly on the RV. The output point features are acquired by fusing the RV features back to the 3D points. The PRNet can be stacked for multiple times to increase its representation capacity.

al., 2019] proposes an accelerated kNN as post-processing to deal with the semantic ambiguity from the RV predictions back to 3D points labels. SqueezeSegV3 [Xu *et al.*, 2020] proposes the spatially adaptive convolution to apply different convolution parameters for different locations on the RV. SalsaNext [Cortinhal *et al.*, 2020] designs a novel encoder-decoder network and adopts Lovász-Softmax loss that can directly optimize the mean Intersection over Union (mIoU) metric. More recently, Lite-HDSeg [Razani *et al.*, 2021] designs lightweight harmonic dense convolutions for both efficiency and effectiveness.

3 Methodology

To achieve accurate and real-time LiDAR semantic segmentation, the method needs to extract semantic features efficiently while keeping the LiDAR point clouds information as much as possible. Inspired by the recent range-based methods that extract features at a fast speed and the point-based methods that can preserve point information, we propose a novel end-to-end point-range fusion network (PRNet). The proposed method extracts semantic features mainly on the RV, and then fuses the features with 3D point features for enhancement, as shown in Figure 3. The PRNet consists of four steps: 1) the range view projection (RVP) projects the input point features to the RV feature maps; 2) a 2D FCN is applied to the 2D RV feature maps to efficiently extract semantic features; 3) the point-range convolution (PRConv) transmits features from the RV back to 3D points; 4) the point fusion (PF) module fuses the features from the RV and the 3D points to ensure complete information. Additionally, the PRNet can be stacked M times to get stronger features. Finally, the features are fed to a single fully connected layer for the final per-point semantic prediction.

In the following subsections, we first illustrate the four components of the proposed PRNet in Section 3.1, 3.2, 3.3 and 3.4. Then, the loss function is shown in Section 3.5.

3.1 Range View Projection

In the range-based methods, the 3D point features are needed to be projected to the 2D RV feature maps. Unlike the previous methods that directly ignore the occluded points, the RVP is proposed to aggregate all point features projected to the same RV pixels by max-pooling. In detail, it first transforms the k^{th} 3D point from the cartesian space $\boldsymbol{p}_k^{3D} = (x_k, y_k, z_k)$ to the spherical space $\boldsymbol{p}_k^{sph} = (r_k, \theta_k, \phi_k)$ by applying,

$$\begin{pmatrix} r_k \\ \theta_k \\ \phi_k \end{pmatrix} = \begin{pmatrix} \sqrt{{x_k}^2 + {y_k}^2 + {z_k}^2} \\ \arcsin\left(\frac{z_k}{\sqrt{{x_k}^2+{y_k}^2+{z_k}^2}}\right) \\ \arctan\left(y_k, x_k\right) \end{pmatrix}, \tag{1}$$

where r_k, θ_k, ϕ_k denote the distance, zenith and azimuth angle respectively. Then, it acquires the corresponding 2D RV coordinates $\boldsymbol{p}_k^{RV} = (u_k, v_k)$ by discretizing θ_k and ϕ_k and ignoring r_k, as the following,

$$\begin{pmatrix} u_k \\ v_k \end{pmatrix} = \begin{pmatrix} \frac{1}{2}[1 - \phi_k \pi^{-1}]W \\ [1 - (\theta_k + f_{up})f^{-1}]H \end{pmatrix}, \tag{2}$$

where $f = f_{up} + f_{down}$ is the LiDAR vertical field-of-view. W, H are the predefined width and height of the RV feature maps.

After grabbing the coordinates on the RV, the 3D features can be transposed to the corresponding 2D position. However, there may be more than one points that fall in the same RV pixel (h, w), and an aggregation manner needs to be determined. The traditional range-based methods only select

the feature from the nearest 3D point, which leads to the above-mentioned information loss. To tackle this problem, the RVP gathers the features $\boldsymbol{\mathcal{F}}^{3D}_{k,c}$ of the 3D points that fall in the pixel (h, w) by max-pooling to form the RV features $\boldsymbol{\mathcal{F}}^{RV}_{h,w,c}$,

$$\boldsymbol{\mathcal{F}}^{RV}_{h,w,c} = \max_{\forall k \text{ s.t. } \lfloor u_k \rfloor = h, \lfloor v_k \rfloor = w} \boldsymbol{\mathcal{F}}^{3D}_{k,c}. \tag{3}$$

Theoretically, the complete 3D points information can be retained within the RV features $\boldsymbol{\mathcal{F}}^{RV}$ by passing information from different points through different feature channels.

3.2 2D Fully Convolutional Network

The 2D FCN is applied to the RV features for semantic feature extraction. Its architecture follows the conventional encoder and decoder structure. The encoder network has multiple down-sampling stages and does not apply down-sampling along the height dimension. The decoder network conducts up-sampling while fusing high-level and low-level feature maps. The detailed architecture is shown in the supplementary material.

3.3 Point-Range Convolution

To avoid semantic ambiguity, unlike the previous range-based methods that directly predict semantic labels on the RV, the proposed method transmits the RV features back to the 3D points and then performs per-point prediction. Considering that different 3D points may correspond to the same 2D pixel, the transmission module must be able to produce dynamic features according to its 3D and 2D positions to avoid ambiguity. Therefore, the PRConv is proposed to fulfill this function by first predicting a set of offsets and weights, and then doing a 3×3 convolution on the displaced positions according to the predicted offsets and weights. Intuitively, as shown in Figure 2, to aggregate features for the occluded *vegetation* points, the kernel position should be adjusted to the nearby non-occluded *vegetation* regions, and the kernel amplitude should be reduced if the corresponding kernel position falls on the *traffic-sign*.

Formally, for the k^{th} 3D point, its features $\boldsymbol{\mathcal{F}}^{3D}_{k}$ are acquired by aggregating the RV features $\boldsymbol{\mathcal{F}}^{RV}$ within a 3×3 window as the following,

$$\boldsymbol{\mathcal{F}}^{3D}_{k} = \sum_{n=1}^{9} \boldsymbol{w}_n \cdot \boldsymbol{\mathcal{F}}^{RV}(\boldsymbol{p}^{RV}_{k} + \boldsymbol{p}_n + \Delta\boldsymbol{p}_{k,n}) \cdot \Delta m_{k,n}, \tag{4}$$

where $\boldsymbol{p}_n \in \{(-1,-1),(-1,0),...,(1,1)\}$ defines a 3×3 window, and $\boldsymbol{w}_n$ denotes the convolution parameters applied on the n^{th} location. To enable the PRConv to alleviate the semantic ambiguity, the kernel offset $\Delta\boldsymbol{p}_{k,n}$ is used to adjust kernel position, and the kernel weight $\Delta m_{k,n}$ is introduced for modulating the kernel amplitude.

As illustrated in Figure 3, both $\Delta\boldsymbol{p}_{k,n}$ and $\Delta m_{k,n}$ are predicted dynamically according to features from both the RV and the 3D points. Specifically, they are estimated following a three-step manner: 1) a 2D convolution layer is applied on the 2D feature maps $\boldsymbol{\mathcal{F}}^{RV}$ to get $\boldsymbol{\mathcal{F}}^{RV'}$; 2) the 3D features $\boldsymbol{\mathcal{F}}^{3D\leftarrow RV'}_{k}$ for the k^{th} point is obtained by bilinear sampling among the 2×2 2D neighbours of the corresponding RV position $\boldsymbol{p}^{RV}_{k}$ on the feature map $\boldsymbol{\mathcal{F}}^{RV'}$; 3) a MLP takes a concatenation of the original 3D point features $\boldsymbol{\mathcal{F}}^{3D}_{k}$ and the interpolated features $\boldsymbol{\mathcal{F}}^{3D\leftarrow RV'}$ as its input and outputs 27-channel ($3 \times 3 \times 3$) vector per 3D point, where the first 18 channels denote the kernel offset $\Delta\boldsymbol{p}_{k,n}$ and the remaining 9 channels are activated by a sigmoid function to obtain the kernel weight $\Delta m_{k,n}$.

3.4 Point Fusion

The point fusion (PF) module fuses the features from the RV and the 3D points to enhance per-point features. For efficiency, it consists of a feature concatenation and two MLP layers. The output point features can serve as the input of the next stacked PRNet or the input of the prediction header. If more than one PRNets are stacked, we impose supervision on each to facilitate the training process. For inference, the output of the last PRNet is used for the final prediction.

3.5 Loss Function

The amounts of data for different categories are highly unbalanced in the LiDAR semantic segmentation dataset (*e.g.* SemanticKITTI, nuScenes). For example, the proportions of *road*, *building*, and *car* are hundreds times than those of *motorcyclist* and *traffic-sign*. Therefore, we apply the weighted cross entropy (WCE) $\mathcal{L}_{wce}$ that emphasizes the rare categories as the following,

$$\begin{aligned} \alpha_c &= \frac{1}{F_c + 0.001} \\ \mathcal{L}_{wce} &= -\frac{1}{N}\sum_{n=1}^{N}\sum_{c=1}^{C} \alpha_c y_n^c \log(\hat{y}_n^c), \end{aligned} \tag{5}$$

where y_n^c ($y_n^c \in \{0,1\}$) and $\hat{y}_n^c$ ($\hat{y}_n^c \in [0,1]$) are the ground-truth and the predicted probability of the c^{th} class on the n^{th} point. F_c is the frequency, and α_c is the weight of the c^{th} class. We also adopt the Lovász-Softmax loss [Berman *et al.*, 2018] $\mathcal{L}_{ls}$ to directly optimize the mean Intersection over Union (mIoU) metric. According to [Aksoy *et al.*, 2020; Cortinhal *et al.*, 2020], it can improve the mIoU performance of the LiDAR semantic segmentation task. The total loss $\mathcal{L}_{total}$ is the sum of the two loss terms and is defined as

$$\mathcal{L}_{total} = \mathcal{L}_{wce} + 3\mathcal{L}_{ls}. \tag{6}$$

4 Experiments

We evaluate the effectiveness and efficiency of the proposed PRNet on the public nuScenes [Caesar *et al.*, 2020] and SemanticKITTI [Behley *et al.*, 2019] single-scan LiDAR semantic segmentation benchmarks.

4.1 Experimental Setup

Datasets. nuScenes for the LiDAR semantic segmentation is a newly released benchmark with 1,000 scenes collected in Boston and Singapore. Each LiDAR scan is collected by a

Methods	mIoU	Runtime(ms)	barrier	bicycle	bus	car	construction	motorcycle	pedestrian	traffic-cone	trailer	truck	driveable	other_flat	sidewalk	terrain	manmade	vegetation
RangeNet++ [Milioto *et al.*, 2019]	65.5	78.6	66.0	21.3	77.2	80.9	30.2	66.8	69.6	52.1	54.2	72.3	94.1	66.6	63.5	70.1	83.1	79.8
SalsaNext [Cortinhal *et al.*, 2020]	72.2	26.2	74.8	34.1	85.9	88.4	42.2	72.4	72.2	63.1	61.3	76.5	96.0	70.8	71.2	71.5	86.7	84.4
PolarNet [Zhang *et al.*, 2020]	71.0	67.5	74.7	28.2	85.3	90.9	35.1	77.5	71.3	58.8	57.4	76.1	96.5	71.1	74.7	74.0	87.3	85.7
AMVNet [Liong *et al.*, 2020]	77.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Cylinder3D [Zhu *et al.*, 2021]	76.1	75.7	76.4	40.3	91.2	**93.8**	51.3	78.0	78.9	64.9	62.1	84.4	96.8	71.6	76.4	75.4	90.5	87.4
RPVNet [Xu *et al.*, 2021]	77.6	-	**78.2**	**43.4**	**92.7**	93.2	49.0	**85.7**	80.5	**66.0**	66.9	84.0	96.9	73.5	75.9	76.0	90.6	88.9
PRNet-1C [ours]	75.3	**20**	76.9	38.6	90.9	92.1	44.8	79.8	76.6	62.5	60.1	79.8	97.0	74.4	76.2	75.7	89.5	87.9
PRNet-2C [ours]	**78.0**	43	78.0	40.9	92.5	93.4	**54.1**	85.4	**80.6**	63.2	**69.8**	**84.7**	**97.3**	**75.1**	**77.7**	**76.1**	**91.0**	**89.3**

Table 1: Class-wise and mean IoU of the proposed PRNet and its competitors on the nuScenes validation set. Runtime measurements are taken on a single NVIDIA RTX 2080Ti GPU.

Velodyne HDL-32E 360° rotating LiDAR with 32 beams vertically. It splits 28,130 samples for training, 6,019 for validation, and 6,008 for testing. After merging similar categories and removing rare categories, 16 categories are used for the official evaluation.

SemanticKITTI contains 43,552 LiDAR scans from 22 sequences collected in Germany. Each LiDAR scan is collected by a Velodyne HDL-64E 360° rotating LiDAR with 64 beams vertically. The training set (19,130 scans) consists of sequences from 00 to 10 except 08, and the sequence 08 (4,071 scans) is used for validation. The rest sequences (20,351 scans) from 11 to 21 are only provided with LiDAR point clouds and are used for the online leaderboards. The dataset is annotated with 28 categories, but it merges categories with different motion states to acquire 19 valid categories for the single-scan LiDAR semantic segmentation benchmark.

Evaluation Metric. To evaluate the proposed PRNet and its competitors, we follow the official metric, namely mean Intersection over Union (mIoU), as the following,

$$mIoU = \frac{1}{C}\sum_{c=1}^{C}\frac{TP_c}{TP_c + FP_c + FN_c}, \tag{7}$$

where TP_c, FP_c, FN_c are the true positive, false positive and false negative of the c^{th} category, respectively. C is the total number of classes.

Network Setup. In the experiments, we adopt two stacked PRNets, and each has a similar network architecture but different parameters. The dimensions of the input point features for these two nets are 5, and 64, respectively. The first 5 feature channels contain x, y, z, $intensity$ and r. As shown in Figure 3, the first MLP layer of each PRNet outputs 64 feature channels. The details of the 2D FCN can be seen in the supplementary material. The two stacked PRNets output 64, 96 point feature channels, respectively. Finally, the segmentation results are acquired by applying an FC layer to the output of the last stacked PRNet. Note that PRNet-2C just doubles the RV feature channels of PRNet-1C.

For SemanticKITTI, the parameters of LiDAR vertical field-of-view are set as $f_{up} = 3°$ and $f_{down} = -25°$. The input size of the RV branch is 64×2048. For nuScenes, we adopt the same configuration except $f_{up} = 20°$ and $f_{down} = -40°$.

Training Details. All experiments are conducted with PyTorch FP32 on NVIDIA RTX 2080Ti GPU. The proposed PRNet is trained from scratch for 48 epochs with a batch size of 16 on 8 GPUs. Stochastic gradient descent (SGD) serves as the optimizer with a weight decay of 0.001, a momentum of 0.9, and an initial learning rate of 0.02, which is decayed by 0.1 every 10 epochs. Following the convention, the data augmentation strategies include random flipping along the x and y axes, random global scale sampled from $[0.95, 1.05]$, random rotation around the z axis, random Gaussian noise $\mathcal{N}(0, 0.02)$, and instance CutMix [Xu *et al.*, 2021].

4.2 Comparisons With the State-of-the-Arts

Results on nuScenes. We report the performance of the proposed PRNet on the newly released nuScenes validation set. As shown in Table 1, our PRNet-2C achieves the best performance among all competing methods, and even outperforms the most competitive RPVNet for most categories. In terms of running time, the PRNet-2C is much faster compared with all methods except the SalsaNext, but SalsaNext is slower and has poorer mIoU performance than the PRNet-1C.

Results on SemanticKITTI. The proposed PRNet is compared with the state-of-the-arts on the SemanticKITTI test set. As shown in Table 2, the methods are grouped as point-based, voxel-based, and range-based methods from top to bottom. We find that PRNet-2C outperforms all point-based and range-based methods by a large margin, and it can be comparable with the top-ranking voxel-based methods, while it runs much faster. Though the proposed PRNet-2C performs poorer compared with the top-ranking RPVNet on some hard categories, it runs 4 times faster with performance comparable with RPVNet on most categories.

Limitation Discussion. For the categories that have fewer training samples and are easily confused with other categories, the proposed method performs worse than the voxel-based RPVNet, *e.g. motorcyclist* that is easily confused with *bicyclist* and *motorcycle*. The reason is that the proposed range-based method mainly encodes context on the 2D RV, but this can be remedied by introducing images.

4.3 Ablation Studies

We make ablative analyses on the SemanticKITTI validation set to figure out the effectiveness of the proposed components.

Methods	mIoU	Runtime(ms)	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic-sign
Point-based Methods																					
PointNet [Qi *et al.*, 2017a]	14.6	-	46.3	1.3	0.3	0.1	0.8	0.2	0.2	0.0	61.6	15.8	35.7	1.4	41.4	12.9	31.0	4.6	17.6	2.4	3.7
PointNet++ [Qi *et al.*, 2017b]	20.1	-	53.7	1.9	0.2	0.9	0.2	0.9	1.0	0.0	72.0	18.7	41.8	5.6	62.3	16.9	46.5	13.8	30.0	6.0	8.9
RandLA-Net [Hu *et al.*, 2020]	53.9	521.8	94.2	26.0	25.8	40.1	38.9	49.2	48.2	7.2	90.7	60.3	73.7	20.4	86.9	56.3	81.4	61.3	66.8	49.2	47.7
KPConv [Thomas *et al.*, 2019]	58.8	-	96.0	30.2	42.5	33.4	44.3	61.5	61.6	11.8	88.8	61.3	72.7	31.6	90.5	64.2	84.8	69.2	69.1	56.4	47.4
Voxel-based Methods																					
PolarNet [Zhang *et al.*, 2020]	54.3	74.3	93.8	40.3	30.1	22.9	28.5	43.2	40.2	5.6	90.8	61.7	74.4	21.7	90.0	61.3	84.0	65.5	67.8	51.8	57.5
AMVNet [Liong *et al.*, 2020]	65.3	-	96.2	59.9	54.2	48.8	45.7	71.0	65.7	11.0	90.1	**71.0**	75.8	32.4	92.4	69.1	85.6	71.7	69.6	62.7	67.2
SPVCNN [Tang *et al.*, 2020]	63.8	187	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
SPVNAS [Tang *et al.*, 2020]	67.0	-	97.2	50.6	50.4	56.6	58.0	67.4	67.1	50.3	90.2	67.6	75.4	21.8	91.6	66.9	86.1	73.4	71.0	64.3	67.3
Cylinder3D [Zhu *et al.*, 2021]	67.8	178	97.1	67.6	64.0	**59.0**	58.6	73.9	67.9	36.0	91.4	65.1	75.5	32.3	91.0	66.5	85.4	71.8	68.5	62.6	65.6
DRINet [Ye *et al.*, 2021]	67.5	62	96.9	57.0	56.0	43.3	54.5	69.4	75.1	58.9	90.7	65.0	75.2	26.2	91.5	67.3	85.2	72.6	68.8	63.5	66.0
AF2S3Net [Cheng *et al.*, 2021]	69.7	-	94.5	65.4	**86.8**	39.2	41.1	**80.7**	**80.4**	**74.3**	91.3	68.8	72.5	**53.5**	87.9	63.2	70.2	68.5	53.7	61.5	**71.0**
RPVNet [Xu *et al.*, 2021]	**70.3**	168*	**97.6**	**68.4**	68.7	44.2	**61.1**	75.9	74.4	73.4	**93.4**	70.3	**80.7**	33.3	**93.5**	72.1	**86.5**	**75.1**	**71.7**	**64.8**	61.4
Ranged-based Methods																					
RangeNet++ [Milioto *et al.*, 2019]	52.2	82.3	91.4	25.7	34.4	25.7	23.0	38.3	38.8	4.8	91.8	65.0	75.2	27.8	87.4	58.6	80.5	55.1	64.6	47.9	55.9
SqueezeSegv3 [Xu *et al.*, 2020]	55.9	124.3	92.5	38.7	36.5	29.6	33.0	45.6	46.2	20.1	91.7	63.4	74.8	26.4	89.0	59.4	82.0	58.7	65.4	49.6	58.9
SalsaNext [Cortinhal *et al.*, 2020]	59.5	27.4	91.9	48.3	38.6	38.9	31.9	60.2	59.0	19.4	91.7	63.7	75.8	29.1	90.2	64.2	81.8	63.6	66.5	54.3	62.1
Lite-HDSeg [Razani *et al.*, 2021]	63.8	-	92.3	40.0	55.4	37.7	39.6	59.2	71.6	54.1	93.0	68.2	78.3	29.3	91.5	65.0	78.2	65.8	65.1	59.5	67.7
PRNet-1C [ours]	65.2	**30/24.3***	95.5	59.7	58.2	53.8	47.3	66.3	72.3	25.1	92.3	67.4	77.1	21.6	91.5	67.0	82.6	67.4	69.4	59.4	64.6
PRNet-2C [ours]	67.2	63/43.5*	95.7	62.1	60.0	56.3	48.4	71.0	75.0	28.6	92.4	66.3	75.9	21.8	92.9	**72.2**	84.4	70.4	68.6	63.5	**71.0**

Table 2: Class-wise and mean IoU of the proposed PRNet and its competitors on the SemanticKITTI test set. Runtime measurements are taken on a single NVIDIA RTX 2080Ti GPU, while $*$ means that it uses NVIDIA Tesla V100 GPU.

	Framework	Backbone	RVP	PRConv	PF	M	mIoU	RT(ms)
a	Baseline	SalsaNext				1	59.0	27.4
b	PRNet-1C	SalsaNext		✓		1	60.2	29.2
c		SalsaNext	✓	✓		1	61.2	29.6
d		SalsaNext	✓	✓	✓	1	62.1	30.8
e		Ours	✓	✓	✓	1	61.8	**14.6**
f		Ours	✓	✓	✓	2	**63.7**	30

Table 3: The proposed framework analysis on the SemanticKITTI validation set. M is the number of the stacked PRNets. RT is the abbreviation of running time.

Effects of the Framework. This analysis is very important to illustrate the insight of our motivation. It starts with the baseline model, SalsaNext [Cortinhal *et al.*, 2020], which is an open-sourced and top-ranking range-based method. As shown in Table 3, we progressively add the proposed components to the SalsaNext until it becomes the same as the proposed PRNet. It can be discovered that: 1) it achieves +1.2 mIoU gains when the post-processing kNN is replaced by the proposed PRConv (a, b); 2) the RVP outperforms the traditional range projection strategy that only keeps the nearest point within an RV pixel, by +1.0 mIoU (b, c). Based on the official code of SalsaNext, we find that only 78% of the whole 3D points are kept during projection to the RV, while the RVP keeps as much 3D points information as possible; 3) the feature fusion of the RV and the 3D points further improves the mIoU by +0.9 (c, d); 4) our 2D backbone is more efficient than that of SalsaNext (d, e, f); 5) two stacked PRNets perform better than a single one (e, f).

Variants of the PRConv. We set the baseline as the bilinear sampling (BS) that acquires the point features by interpolating the four neighbors of the corresponding RV pixel. Afterwards, we add the kernel offset and the kernel weight

	Type	Offset	Weight	mIoU	Runtime(ms)
a	BS			60.7	**20**
b	PRConv			61.1	24.3
c		✓		62.9	26.6
d		✓	✓	**63.7**	30

Table 4: The PRConv analysis on the SemanticKITTI validation set. BS stands for bilinear sampling.

one by one to figure out the proposed PRConv. As shown in Table 4, the PRConv without the kernel offset and the kernel weight improves the +0.4 mIOU, compared with the baseline. Moreover, adding the kernel offset into PRConv leads to the biggest mIoU jump of +1.8. Finally, the complete PRConv equipped with the kernel offset and weight outperforms the baseline by +3.0 mIoU.

5 Conclusion

In this paper, we present an end-to-end PRNet for real-time LiDAR semantic segmentation, which leverages the strengths of the point-based and range-based methods. The PRNet extracts features mainly on the RV and produces semantic predictions in the 3D space by fused features originating from both the 3D space and the RV space. To minimize the information loss, we propose the RVP operation for the RV projection. To avoid semantic ambiguity, a novel PRConv is proposed to transmit the RV features back to the 3D points dynamically. Experimental results on the SemanticKITTI and nuScenes benchmarks demonstrate the effectiveness and superiority of the proposed components. The PRNet has resolved two long-plagued issues for the range-based methods and demonstrates the potential of the range-based methods for both effectiveness and efficiency. Future works lie in integrating image information to further improve its performance.

References

[Aksoy *et al.*, 2020] Eren Erdal Aksoy, Saimir Baci, and Selcuk Cavdar. Salsanet: Fast road and vehicle segmentation in lidar point clouds for autonomous driving. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 926–932. IEEE, 2020.

[Behley *et al.*, 2019] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9297–9307, 2019.

[Berman *et al.*, 2018] Maxim Berman, Amal Rannen Triki, and Matthew 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*, pages 4413–4421, 2018.

[Caesar *et al.*, 2020] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[Cheng *et al.*, 2021] Ran Cheng, Ryan Razani, Ehsan Taghavi, Enxu Li, and Bingbing Liu. 2-s3net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12547–12556, 2021.

[Choy *et al.*, 2019] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019.

[Cortinhal *et al.*, 2020] Tiago Cortinhal, George Tzelepis, and Eren Erdal Aksoy. Salsanext: Fast, uncertainty-aware semantic segmentation of lidar point clouds. In *International Symposium on Visual Computing*, pages 207–222. Springer, 2020.

[Hu *et al.*, 2020] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11108–11117, 2020.

[Liong *et al.*, 2020] Venice Erin Liong, Thi Ngoc Tho Nguyen, Sergi Widjaja, Dhananjai Sharma, and Zhuang Jie Chong. Amvnet: Assertion-based multi-view fusion network for lidar semantic segmentation. *arXiv preprint arXiv:2012.04934*, 2020.

[Milioto *et al.*, 2019] Andres Milioto, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. Rangenet++: Fast and accurate lidar semantic segmentation. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4213–4220. IEEE, 2019.

[Qi *et al.*, 2017a] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas 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*, pages 652–660, 2017.

[Qi *et al.*, 2017b] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017.

[Razani *et al.*, 2021] Ryan Razani, Ran Cheng, Ehsan Taghavi, and Liu Bingbing. Lite-hdseg: Lidar semantic segmentation using lite harmonic dense convolutions. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9550–9556. IEEE, 2021.

[Tang *et al.*, 2020] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *European Conference on Computer Vision*, pages 685–702. Springer, 2020.

[Thomas *et al.*, 2019] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6411–6420, 2019.

[Xu *et al.*, 2020] Chenfeng Xu, Bichen Wu, Zining Wang, Wei Zhan, Peter Vajda, Kurt Keutzer, and Masayoshi Tomizuka. Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation. In *European Conference on Computer Vision*, pages 1–19. Springer, 2020.

[Xu *et al.*, 2021] Jianyun Xu, Ruixiang Zhang, Jian Dou, Yushi Zhu, Jie Sun, and Shiliang Pu. Rpvnet: A deep and efficient range-point-voxel fusion network for lidar point cloud segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16024–16033, 2021.

[Ye *et al.*, 2021] Maosheng Ye, Shuangjie Xu, Tongyi Cao, and Qifeng Chen. Drinet: A dual-representation iterative learning network for point cloud segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7447–7456, 2021.

[Zhang *et al.*, 2020] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, Boqing Gong, and Hassan 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*, pages 9601–9610, 2020.

[Zhu *et al.*, 2021] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9939–9948, 2021.