EPMF: Efficient Perception-Aware Multi-Sensor Fusion for 3D Semantic Segmentation

Mingkui Tan, *Member, IEEE*, Zhuangwei Zhuang, *Graduate Student Member, IEEE*, Sitao Chen, Rong Li, Kui Jia, *Member, IEEE*, Qicheng Wang, and Yuanqing Li, *Fellow, IEEE*

***Abstract*—We study multi-sensor fusion for 3D semantic segmentation that is important to scene understanding for many applications, such as autonomous driving and robotics. Existing fusion-based methods, however, may not achieve promising performance due to the vast difference between the two modalities. In this work, we investigate a collaborative fusion scheme called perception-aware multi-sensor fusion (PMF) to effectively exploit perceptual information from two modalities, namely, appearance information from RGB images and spatio-depth information from point clouds. To this end, we project point clouds to the camera coordinate using perspective projection, and process both inputs from LiDAR and cameras in 2D space while preventing the information loss of RGB images. Then, we propose a two-stream network to extract features from the two modalities, separately. The extracted features are fused by effective residual-based fusion modules. Moreover, we introduce additional perception-aware losses to measure the perceptual difference between the two modalities. Last, we propose an improved version of PMF, i.e., EPMF, which is more efficient and effective by optimizing data pre-processing and network architecture under perspective projection. Specifically, we propose cross-modal alignment and cropping to obtain tight inputs and reduce unnecessary computational costs. We then explore more efficient contextual modules under perspective projection and fuse the LiDAR features into the camera stream to boost the performance of the two-stream network. Extensive experiments on benchmark data sets show the superiority of our method. For example, on nuScenes test set, our EPMF outperforms the state-of-the-art method, i.e., RangeFormer, by 0.9% in mIoU.**

***Index Terms*—3D semantic segmentation, autonomous driving, deep neural networks, multi-sensor fusion, scene understanding.**

Manuscript received 14 August 2023; revised 16 February 2024; accepted 9 May 2024. Date of publication 29 May 2024; date of current version 5 November 2024. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 62072190, in part by the Key-Area Research and Development Program of Guangdong Province under Grant 2018B010107001, and in part by Guangdong Introducing Innovative and Enterpreneurial Teams under Grant 2017ZT07X183. Recommended for acceptance by P. Arbelaez. *(Mingkui Tan and Zhuangwei Zhuang contributed equally to this work.) (Corresponding author: Mingkui Tan.)*

Mingkui Tan and Zhuangwei Zhuang are with the School of Software Engineering, South China University of Technology, Guangzhou, Guangdong 510641, China, and also with Pazhou Laboratory, Guangzhou 510335, China (e-mail: mingkuitan@scut.edu.cn; z.zhuangwei@mail.scut.edu.cn).

Sitao Chen and Rong Li are with the School of Software Engineering, South China University of Technology, Guangzhou, Guangdong 510641, China (e-mail: mechenst@mail.scut.edu.cn; selirong@mail.scut.edu.cn).

Kui Jia is with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou, Guangdong 510641, China (e-mail: kuijia@scut.edu.cn).

Qicheng Wang is with the Department of Mathematics, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, and also with Minieye, Shenzhen, Guangdong 518063, China (e-mail: wangqicheng@minieye.cc).

Yuanqing Li is with the Pazhou Laboratory, Guangzhou 510335, China (e-mail: auyqli@scut.edu.cn).

Our source code is available at https://github.com/ICEORY/PMF.

Digital Object Identifier 10.1109/TPAMI.2024.3402232

I. Introduction

SEMANTIC scene understanding is a fundamental task for many applications, such as autonomous driving and robotics [18], [44], [58], [59]. Specifically, in the scenes of autonomous driving, it provides fine-grained environmental information for high-level motion planning and improves the safety of autonomous cars [3], [20]. One of the important tasks in semantic scene understanding is semantic segmentation, which assigns a class label to each data point in the input data, and helps autonomous cars to better understand the environment.

According to the sensors used by semantic segmentation methods, recent studies can be divided into three categories: camera-only methods [2], [9], [10], [45], [73], LiDAR-only methods [1], [15], [31], [66], [77] and multi-sensor fusion methods [37], [47], [49], [64], [74]. Camera-only methods have achieved great progress with the help of a massive amount of open-access data sets [6], [14], [16]. Since images obtained by a camera are rich in appearance information (*e.g.*, texture and color), camera-only methods can provide fine-grained and accurate semantic segmentation results. However, as passive sensors, cameras are susceptible to changes in lighting conditions and are thus unreliable [61].[1] To address this problem, researchers conduct semantic segmentation on point clouds from LiDAR. Compared with camera-only approaches, LiDAR-only methods are more robust in different light conditions, as LiDAR provides reliable and accurate spatio-depth information on the physical world. Unfortunately, LiDAR-only semantic segmentation is challenging due to the sparse and irregular distribution of point clouds. In addition, point clouds lack texture and color information, resulting in high classification error in the fine-grained segmentation task of LiDAR-only methods. A straightforward solution for addressing both drawbacks of camera-only and LiDAR-only methods is to fuse the multimodal data from both sensors, *i.e.*, multi-sensor fusion methods. Nevertheless, due to the large domain gap between RGB cameras and LiDAR, multi-sensor fusion is still a nontrivial task.

In multi-sensor fusion methods, fusing multimodal data from different sensors is an important problem. Existing fusion-based methods [47], [64] mainly lift the dense 2D image features to

[1] See Section IV-G for more details.

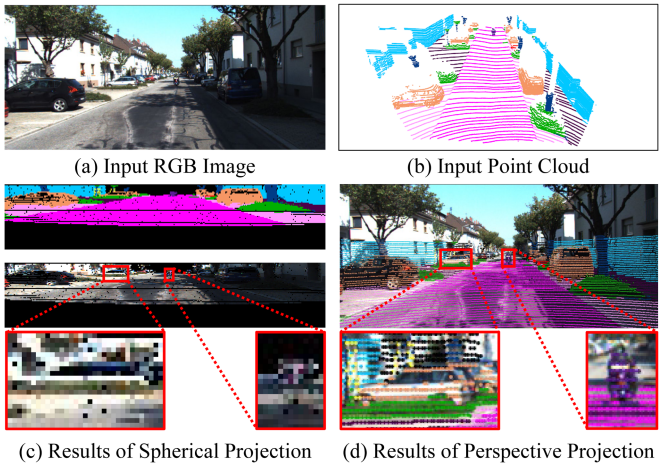


Fig. 1. Comparisons of spherical projection [50], [66] and perspective projection. With spherical projection, most of the appearance information from RGB images is lost. Instead, we preserve the information of images with perspective projection. To distinguish different classes, we colorize the point clouds using semantic labels from SemanticKITTI.

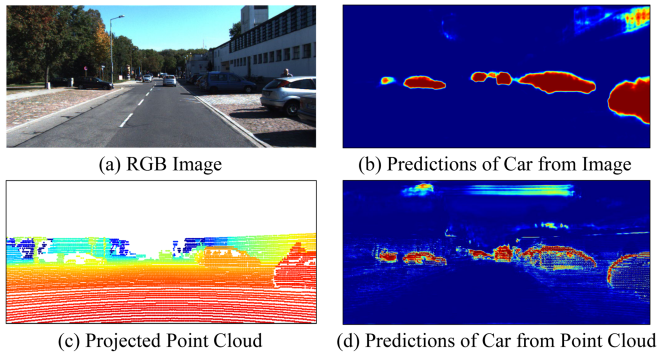


Fig. 2. Comparisons of the predictions from images and point clouds. Deep neural networks capture different perceptual information from RGB images and point clouds. Red indicates predictions with higher scores.

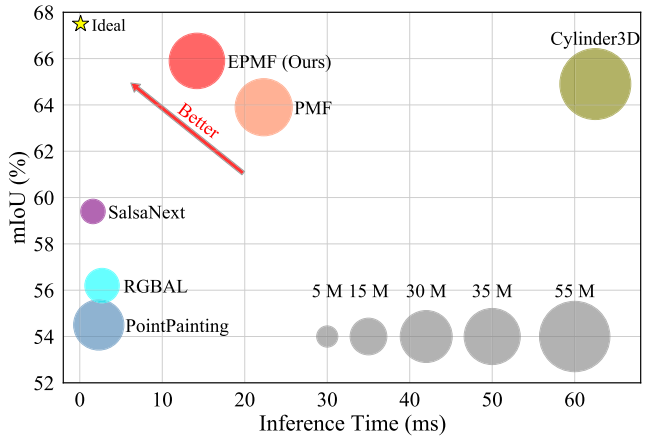


Fig. 3. Comparisons of efficiency and performance of different methods on SemanticKITTI-FV.

the 3D LiDAR coordinates using spherical projection [50] and conduct feature fusion in the sparse LiDAR domain. However, these methods suffer from a critical limitation: as the point clouds are very sparse, most of the appearance information from the RGB images is missing after un-projecting it to the LiDAR coordinates. For example, as shown in Fig. 1(c), the car and motorcycle in the image become distorted with spherical projection. As a result, existing fusion-based methods have difficulty capturing the appearance information from the projected RGB images.

In this paper, we aim to exploit an effective multi-sensor fusion method. Unlike existing methods [47], [64], we assume and highlight that the perceptual information from both RGB images and point clouds, *i.e.*, appearance information from images and spatio-depth information from point clouds, is important in fusion-based semantic segmentation. Based on this intuition, we propose a perception-aware multi-sensor fusion (PMF) scheme that conducts a collaborative fusion of perceptual information from two modalities of data in three aspects. **First**, we propose perspective projection to project the point clouds to the camera coordinate system to obtain additional spatio-depth information for RGB images. **Second**, we propose a two-stream network (TSNet) that contains a camera stream and a LiDAR stream to extract perceptual features from multi-modal sensors separately. Considering that the information from images is unreliable in an outdoor environment, we fuse the image features to the LiDAR stream by effective residual-based fusion (RF) modules, which are designed to learn the complementary features of the original LiDAR modules. **Third**, we propose perception-aware losses to measure the vast perceptual difference between the two data modalities and boost the fusion of different perceptual information. Specifically, as shown in Fig. 2, the perceptual features captured by the camera stream and LiDAR stream are different. Therefore, we use the predictions with higher confidence to supervise those with lower confidence. Since model efficiency is also an essential factor for real-world applications, we further exploit the efficiency of PMF and propose an improved version, *i.e.*, EPMF.

Our contributions are summarized as follows. First, we propose a perception-aware multi-sensor fusion (PMF) scheme to effectively fuse the perceptual information from RGB images and point clouds. Second, by fusing the spatio-depth information from point clouds and appearance information from RGB images, PMF is able to address segmentation with undesired light conditions and sparse point clouds. More critically, PMF is robust in adversarial samples of RGB images by integrating the information from point clouds. Third, we introduce perception-aware losses into the network and force the network to capture the perceptual information from two different-modality sensors. As demonstrated in Fig. 3, on top of PMF, we further propose EPMF, which reduces the model complexity of PMF while improving the model performance by a large margin. The extensive experiments on three benchmark data sets including SemanticKITTI-FV [3], nuScenes [7], and A2D2 [23], demonstrate the superior performance of our method. For example, on nuScenes test set, our EPMF outperforms the state-of-the-art methods, *i.e.*, SphereFormer [38] and RangeFormer [36]

by 1.1% and 0.9% in mIoU, respectively, without additional fine-tuning or test-time augmentation.

This paper extends our prior version [82] from following aspects. 1) We propose cross-modal alignment and cropping (CAC) to address the miss-alignment issue of point clouds and RGB images. 2) We explore the impact of the different resolutions of point clouds and improve the efficiency of our method without performance degradation. 3) We adopt the proposed EPMF on more benchmark data sets and show the superior performance of our method on extremely sparse point clouds. 4) We provide more ablation studies to investigate the effectiveness of our method.

II. Related Work

In this section, we revisit the existing literature on 2D and 3D semantic segmentation, *i.e.*, camera-only methods, LiDAR-only methods and multi-sensor fusion methods.

Camera-only methods Camera-only semantic segmentation aims to predict the pixel-wise labels of 2D images. FCN [45] is a fundamental work in semantic segmentation, which proposes an end-to-end fully convolutional architecture based on image classification networks. In addition to FCN, recent works have achieved significant improvements via exploring multi-scale information [9], [40], [78], dilated convolution [10], [48], [65], and attention mechanisms [32], [73]. More recently, transformer-based methods have been proposed for robust and accurate segmentation [11], [68]. Specifically, SegFormer [68] designs a positional-encoding-free and hierarchical transformer encoder that generates both high-resolution fine features and low-resolution coarse features. Moreover, it proposes a lightweight All-MLP decoder to aggregate the multiscale features for robust semantic segmentation. Mask2Former [11] proposes a universal architecture to address any image segmentation tasks, including panoptic, instance, or semantic segmentation. Although these methods show strong robustness to the corruptions and perturbations in autonomous driving scenes, their performance under poor lighting conditions (*i.e.*, at night-time) is unsatisfactory compared to LiDAR-based methods.

LiDAR-only methods To address the drawbacks of cameras, LiDAR is an important sensor on an autonomous car, as it is robust in more complex scenes. According to the preprocessing pipeline, existing methods for point clouds mainly contain two categories, including direct methods [31], [55], [56], [81] and projection-based methods [15], [66], [67], [69].

Direct methods perform semantic segmentation by processing the raw 3D point clouds directly. PointNet [55] is a pioneering work in this category that extracts point cloud features by multi-layer perception. A subsequent extension, *i.e.*, PointNet++ [56], further aggregates a multi-scale sampling mechanism to aggregate global and local features. However, these methods do not consider the varying sparsity of point clouds in outdoor scenes. Cylinder3D [81] addresses this issue by using 3D cylindrical partitions and asymmetrical 3D convolutional networks. SphereFormer [38] directly aggregates information from dense close points to sparse distant ones through radial window partitions and proposes dynamic feature selection to select local neighbor features or radial contextual features. However, direct methods have a high computational complexity, which limits their applicability in autonomous driving. PVKD [30] transfers both point-level and voxel-level hidden knowledge from a large LiDAR semantic segmentation model to a slim network to achieve model compression. WaffleIron [54] uses standard MLPs and dense 2D convolutions to build a 3D backbone for point cloud semantic segmentation, which does not rely on sparse 3D convolution. In addition, one can easily improve the efficiency of networks by existing neural architecture search [8], [26], [52] and model compression techniques [27], [43], [71].

Projection-based methods are more efficient because they convert 3D point clouds to a 2D grid. In projection-based methods, researchers focus on exploiting effective projection methods, such as spherical projection [50], [66] and bird's-eye projection [77]. Such 2D representations allow researchers to investigate efficient network architectures based on existing 2D convolutional networks [1], [15], [25]. AMVNet [42] utilizes the late fusion to combine the advantages of both the range view and bird's-eye view networks. RPVNet [70] proposes a range-point-voxel fusion network to synergize all the three view's representations to alleviate the above problem. RangeFormer [36] formulates the segmentation of range view grids as a seq2seq problem and adopts several standard Transformer blocks to capture the rich contextual information, then uses the MLP heads to decode the multi-scale features. Unlike uni-modal methods, we focus on fusing information from both the camera and LiDAR to achieve accurate and robust 3D semantic segmentation for autonomous driving.

Multi-sensor fusion methods To leverage the benefits of both camera and LiDAR, recent work has attempted to fuse information from two complementary sensors to improve the accuracy and robustness of the 3D semantic segmentation algorithm [37], [47], [49], [64]. RGBAL [47] converts RGB images to a polar-grid mapping representation and designs early and mid-level fusion strategies. PointPainting [64] obtains the segmentation results of images and projects them to the LiDAR space by using bird's-eye projection [77] or spherical projection [50]. The projected segmentation scores are concatenated with the original point cloud features to improve the performance of LiDAR networks. 2DPASS [72] enhances the representation learning of 3D semantic segmentation network by distilling multi-modal knowledge to single point cloud modality. In this way, 2DPASS can use LiDAR-only input in test-time. However, the model performance is unsatisfactory under the scene with sparser point clouds (*e.g.*, A2D2 with 16-beam LiDARs). In contrast, our EPMF can achieve promising performance by fusing 2D images and 3D point clouds during inference. Besides, unlike existing methods that perform feature fusion in the LiDAR domain, PMF [82] exploits a collaborative fusion of multimodal data in camera coordinates. In this work, we further extend PMF to improve its efficiency and performance.

III. Proposed Method

In this work, we propose an efficient perception-aware multi-sensor fusion (EPMF) scheme to perform an effective fusion of

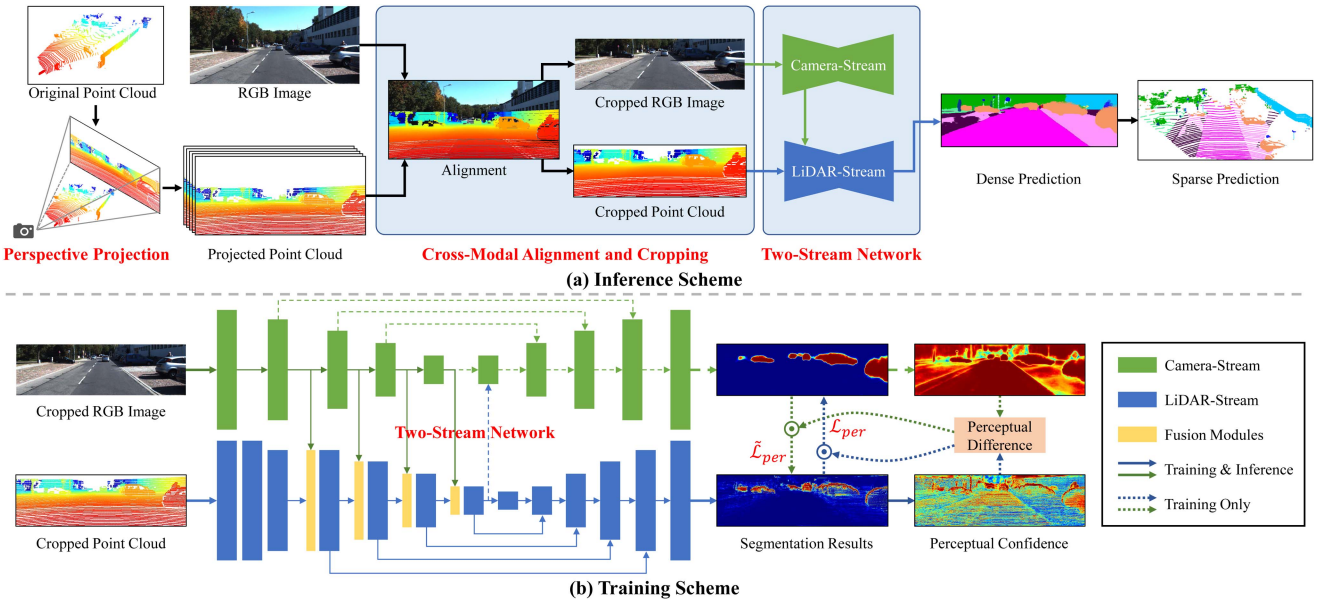


Fig. 4. Illustration of the training and inference schemes of EPMF. EPMF consists of three components: (1) perspective projection with cross-modal alignment and crop; (2) a two-stream network (TSNet) with feature fusion modules; and (3) perception-aware losses $\mathcal{L}_{per}, \widetilde{\mathcal{L}_{per}}$ w.r.t. the camera stream and the LiDAR stream. We first project the point clouds to the camera coordinate with perspective projection and learn the features from both the RGB images and point clouds using TSNet. The image features are fused into the LiDAR stream network by fusion modules. In the training procedure, we use perception-aware losses to help the network focus on the perceptual features of both images and point clouds. In the inference procedure, we apply dense-to-sparse mapping to obtain 3D segmentation results of point clouds.

the perceptual information from both RGB images and point clouds. Specifically, as shown in Fig. 4, EPMF contains three components: (1) perspective projection with cross-modal alignment and cropping; (2) a two-stream network (TSNet) with residual-based fusion modules; (3) perception-aware losses. The general scheme of EPMF is shown in Algorithm 1. We first project the point clouds to the camera coordinate system by using perspective projection. Then, we use a two-stream network that contains a camera stream and a LiDAR stream to extract perceptual features from the two modalities, separately. The features from the camera stream are fused into the LiDAR stream by residual-based fusion modules. Finally, we introduce perception-aware losses into the optimization of the network.

A. Pipeline of Data Pre-Processing

Existing methods [47], [64] mainly project images to the LiDAR coordinate system using spherical projection. However, due to the sparse nature of point clouds, most of the appearance information from the images is lost with spherical projection (see Fig. 1). To address this issue, we propose perspective projection to project the sparse point clouds to the camera coordinate system.

Formulation of perspective projection. Let $\{\mathbf{P}, \mathbf{X}, \mathbf{y}\}$ be one of the training samples from a given data set, where $\mathbf{P} \in \mathbb{R}^{4\times N}$ indicates a point cloud from LiDAR and N denotes the number of points. Each point $\mathbf{P}_i$ in point cloud $\mathbf{P}$ consists of 3D coordinates (x, y, z) and a reflectance value (r). Let $\mathbf{X} \in \mathbb{R}^{3\times H\times W}$ be an image from an RGB camera, where H and W represent the height and width of the image, respectively. $\mathbf{y} \in \mathbb{R}^N$ is the set of semantic labels for point cloud $\mathbf{P}$.

In perspective projection, we aim to project the point cloud $\mathbf{P}$ from LiDAR coordinate to the camera coordinate to obtain the 2D LiDAR features $\widetilde{\mathbf{X}} \in \mathbb{R}^{C\times H\times W}$. Here, C indicates the number of channels w.r.t. the projected point cloud. Following [19], we obtain $\mathbf{P}_i = (x, y, z, 1)^\top$ by appending a fourth column to $\mathbf{P}_i$ and compute the projected point $\widetilde{\mathbf{P}}_i = (\widetilde{x}, \widetilde{y}, \widetilde{z})^\top$ in the camera coordinates by

$$\widetilde{\mathbf{P}}_i = \mathbf{TRP}_i, \tag{1}$$

where $\mathbf{T} \in \mathbb{R}^{3\times 4}$ is the projection matrix from LiDAR coordinates to camera coordinates. $\mathbf{R} \in \mathbb{R}^{4\times 4}$ is expanded from the rectifying rotation matrix $\mathbf{R}^{(0)} \in \mathbb{R}^{3\times 3}$ by appending a fourth zero row and column and setting $\mathbf{R}(4, 4) = 1$. The calibration parameters $\mathbf{T}$ and $\mathbf{R}^{(0)}$ can be obtained by the approach in [21]. Subsequently, the corresponding pixel (h, w) in the projected image $\widetilde{\mathbf{X}}$ w.r.t. the point $\mathbf{P}_i$ is computed by $h = \widetilde{x}/\widetilde{z}$ and $w = \widetilde{y}/\widetilde{z}$.

Cross-modal alignment and cropping. As shown in Fig. 4(a), since we only focus on the segmentation of point clouds, directly projecting point clouds to the view of cameras leads to unnecessary computational costs. To address this issue, we introduce cross-modal alignment and cropping (CAC). First, we align the RGB image and the projected point clouds to find the overlap of the multi-modal inputs. Then, we crop both RGB images and projected point clouds to obtain compact inputs: For RGB images, we only keep the area that contains point clouds. For the projected point clouds, as the area outside the horizontal field of view (FOV) of the camera is covered by other cameras, we only keep the points within the horizontal FOV of the camera.

Algorithm 1: General Scheme of EPMF.

Input: Training data $\{\mathbf{P}, \mathbf{X}, \mathbf{y}\}$, TSNet with submodels $M, \widetilde{M}$, hyperparameters τ, λ, γ.

1: **while** *not convergent* **do**
2: Project the point clouds $\mathbf{P}$ by using perspective projection to obtain $\widetilde{\mathbf{X}}$.
3: Use $\{\widetilde{\mathbf{X}}, \mathbf{X}\}$ as the inputs of TSNet and compute the output probabilities $\{\widetilde{\mathbf{O}}, \mathbf{O}\}$ with (2).
4: Compute the perceptual confidence $\widetilde{\mathbf{C}}$ and $\mathbf{C}$.
5: Construct perception-aware losses to measure the perceptual difference with (7) and (9).
6: Update $\widetilde{M}$ and M by minimizing the objective in (17).
7: **end while**

In the case that the LiDAR sensor has a larger vertical FOV, we can keep the point clouds outside the image.[2]

After applying CAC, we compute the features of the projected point clouds. Because the point cloud is very sparse, each pixel in the projected $\widetilde{\mathbf{X}}$ may not have a corresponding point $\mathbf{p}$. Therefore, we first initialize all pixels in $\widetilde{\mathbf{X}}$ to 0. Following [15], [50], we compute 5-channel LiDAR features, *i.e.*, (d, x, y, z, r), for each pixel (h, w) in the projected 2D image $\widetilde{\mathbf{X}}$, where $d = \sqrt{x^2 + y^2 + z^2}$ represents the range value of each point. Thus, we set the number of channels C to 5 in this work.

B. Architecture Design of EPMF

As images and point clouds are different-modality data, it is difficult to handle both types of information from the two modalities by using a single network [37]. Motivated by [17], [60], we propose a two-stream network (TSNet) that contains a camera stream and a LiDAR stream to process the features from camera and LiDAR, separately, as illustrated in Fig. 4. In this way, we can use the network architectures designed for images and point clouds as the backbones of each stream in TSNet.

Formulation of Two Stream Network. Let $\widetilde{M}$ and M be the LiDAR stream and the camera stream in TSNet, respectively. Let $\widetilde{\mathbf{O}} \in \mathbb{R}^{S \times H \times W}$ and $\mathbf{O} \in \mathbb{R}^{S \times H \times W}$ be the output probabilities w.r.t. each network, where S indicates the number of semantic classes. The outputs of TSNet are computed by

$$\begin{cases} \mathbf{O} = M(\mathbf{X}), \\ \widetilde{\mathbf{O}} = \widetilde{M}\left(\widetilde{\mathbf{X}}\right). \end{cases} \tag{2}$$

Design of Residual-based Fusion Module. Since the features of images contain many details of objects, we then introduce a residual-based fusion module, as illustrated in Fig. 5, to fuse the image features to the LiDAR stream.[3] Let $\{\mathbf{F}_l \in \mathbb{R}^{C_l \times H_l \times W_l}\}_{l=1}^{L}$ be a set of image features from the camera stream, where l indicates the layer in which we obtain the features. C_l indicates the number of channels of the l-th layer in the camera stream. H_l and W_l indicate the height and width of the feature maps from the l-th layer, respectively. Let $\{\widetilde{\mathbf{F}}_l \in \mathbb{R}^{\widetilde{C}_l \times H_l \times W_l}\}_{l=1}^{L}$ be the features from the LiDAR stream, where $\widetilde{C}_l$ indicates the number of channels of the l-th layer in the LiDAR stream. To obtain the fused features, we first concatenate the features from each network and use a convolutional layer to reduce the number of channels of the fused features. The fused features $\mathbf{F}_l^{fuse} \in \mathbb{R}^{\widetilde{C}_l \times H_l \times W_l}$ are computed by

$$\mathbf{F}_l^{fuse} = f_l\left(\left[\widetilde{\mathbf{F}}_l; \mathbf{F}_l\right]\right), \tag{3}$$

where $[\cdot;\cdot]$ indicates the concatenation operation. $f_l(\cdot)$ is the convolution operation w.r.t. the l-th fusion module.

Considering that the camera is easily affected by different lighting and weather conditions, the information from RGB images is not reliable in an outdoor environment. We use the fused features as the complement of the original LiDAR features and design the fusion module based on the residual structure [28]. Incorporating with the attention module [5], the output features $\mathbf{F}_l^{out} \in \mathbb{R}^{\widetilde{C}_l \times H_l \times W_l}$ of the fusion module are computed by

$$\mathbf{F}_l^{out} = \widetilde{\mathbf{F}}_l + \sigma\left(g_l\left(\mathbf{F}_l^{fuse}\right)\right) \odot \mathbf{F}_l^{fuse}, \tag{4}$$

where $\sigma(x) = 1/(1 + e^{-x})$ indicates sigmoid function. $g_l(\cdot)$ indicates convolution operation in the attention module w.r.t. the l-th fusion module. $\odot$ indicates element-wise multiplication operation.

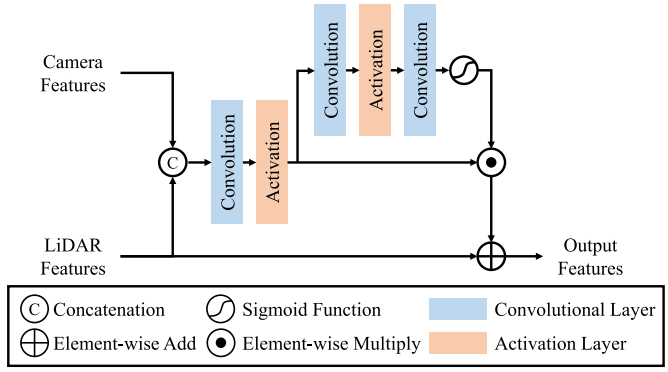


Fig. 5. Illustration of the residual-based fusion (RF) module. RF fuses features from both the camera and LiDAR to generate the complementary information of the original LiDAR features.

C. Construction of Perception-Aware Loss

The construction of perception-aware loss is very important in our method. As demonstrated in Fig. 2, because the point clouds are very sparse, the LiDAR stream network learns only the local features of points while ignoring the shape of objects. In contrast, the camera stream can easily capture the shape and texture of objects from dense images. In other words, the perceptual features captured by the camera stream and LiDAR stream are different. With this intuition, we introduce a perception-aware loss to make the fusion network focus on the perceptual features from the camera and LiDAR.

To measure the perceptual confidence of the predictions w.r.t. the LiDAR stream, we first compute the entropy map $\widetilde{\mathbf{E}} \in$

[2] More discussions of CAC can be found in Seciton V-C

[3] We discuss different designs of fusion modules in Section II of the supplementary material of [82]

$\mathbb{R}^{H\times W}$ by

$$\widetilde{\mathbf{E}}_{h,w} = -\frac{1}{\log S}\sum_{s=1}^{S}\widetilde{\mathbf{O}}_{s,h,w}\log\left(\widetilde{\mathbf{O}}_{s,h,w}\right). \tag{5}$$

Following [57], we use $\log S$ to normalize the entropy to (0,1]. Then, the perceptual confidence map $\widetilde{\mathbf{C}}$ w.r.t. the LiDAR stream is computed by $\widetilde{\mathbf{C}} = \mathbf{1} - \widetilde{\mathbf{E}}$. For the camera stream, the confidence map is computed by $\mathbf{C} = \mathbf{1} - \mathbf{E}$.

Note that not all information from the camera stream is useful. For example, the camera stream is confident inside objects but may make mistakes at the boundary of the objects. In addition, the predictions with lower confidence scores are more likely to be wrong. Incorporating a confidence threshold, we measure the importance of perceptual information from the camera stream by

$$\widetilde{\mathbf{\Omega}}_{h,w} = \begin{cases}\max\left(\mathbf{C}_{h,w} - \widetilde{\mathbf{C}}_{h,w}, 0\right), & \text{if } \mathbf{C}_{h,w} > \tau,\\ 0, & \text{otherwise.}\end{cases} \tag{6}$$

Here τ indicates the confidence threshold.

Inspired by [29], [33], [76], to learn the perceptual information from the camera stream, we construct the perception-aware loss w.r.t. the LiDAR stream by

$$\widetilde{\mathcal{L}}_{per} = \frac{1}{Q}\sum_{h=1}^{H}\sum_{w=1}^{W}\widetilde{\mathbf{\Omega}}_{h,w}D_{KL}\left(\widetilde{\mathbf{O}}_{:,h,w}||\mathbf{O}_{:,h,w}\right), \tag{7}$$

where $Q = H\cdot W$ and $D_{KL}(\cdot||\cdot)$ indicates the Kullback-Leibler divergence [29].

For the camera stream, the importance of information from LiDAR stream is computed by

$$\mathbf{\Omega}_{h,w} = \begin{cases}\max\left(\widetilde{\mathbf{C}}_{h,w} - \mathbf{C}_{h,w}, 0\right), & \text{if } \widetilde{\mathbf{C}}_{h,w} > \tau,\\ 0, & \text{otherwise.}\end{cases} \tag{8}$$

The perception-aware loss w.r.t. the camera stream is

$$\mathcal{L}_{per} = \frac{1}{Q}\sum_{h=1}^{H}\sum_{w=1}^{W}\mathbf{\Omega}_{h,w}D_{KL}\left(\mathbf{O}_{:,h,w}||\widetilde{\mathbf{O}}_{:,h,w}\right). \tag{9}$$

D. Objective Functions

In addition to the perception-aware loss, we also use multi-class focal loss [41] and Lovász-softmax loss [4], which are commonly used in existing segmentation work [15], [81], to train the two stream network.

Let $\mathbf{Y}\in\mathbb{R}^{H\times W}$ be the projected labels in the camera coordinates. H and W indicate the height and width, respectively. For each point $\mathbf{P}_i$, we project the 3D coordinates (x, y, z) to the pixel (h, w) in the camera coordinate system by using perspective projection. Then, we initialize all pixels in $\mathbf{Y}$ by 0 and compute the projected labels in $\mathbf{Y}$ by

$$\mathbf{Y}_{h,w} := \mathbf{y}_i. \tag{10}$$

The multi-class focal loss w.r.t. the LiDAR stream is defined as

$$\widetilde{\mathcal{L}}_{foc} = \frac{1}{K}\sum_{s=1}^{S}\sum_{h=1}^{H}\sum_{w=1}^{W}\alpha_s\mathbb{1}\{\mathbf{Y}_{h,w}=s\}FL\left(\widetilde{\mathbf{O}}_{s,h,w}\right), \tag{11}$$

where $FL(p) = -(1-p)^2\log(p)$ denotes the focal-loss function. α_s denotes the weights w.r.t. the s-th class. $K = \sum_{s=1}^{S}\sum_{h=1}^{H}\sum_{w=1}^{W}\mathbb{1}\{\mathbf{Y}_{h,w}=s\}$ indicates the number of available labels. $\mathbb{1}\{\cdot\}$ indicates the indicator function. Then, the multi-class focal loss w.r.t. the camera stream is

$$\mathcal{L}_{foc} = \frac{1}{K}\sum_{s=1}^{S}\sum_{h=1}^{H}\sum_{w=1}^{W}\mathbb{1}\{\mathbf{Y}_{h,w}=s\}FL(\mathbf{O}_{s,h,w}), \tag{12}$$

The Lovász-softmax loss w.r.t. the LiDAR stream is

$$\widetilde{\mathcal{L}}_{lov} = \frac{1}{S}\sum_{s=1}^{S}\overline{\Delta_{J_s}}\left(\widetilde{\mathbf{m}}(s)\right), \tag{13}$$

where

$$\widetilde{\mathbf{m}}_i(s) = \begin{cases}1-\widetilde{\mathbf{O}}_{s,h,w} & \text{if } s=\mathbf{Y}_{h,w},\\ \widetilde{\mathbf{O}}_{s,h,w} & \text{otherwise.}\end{cases} \tag{14}$$

$\overline{\Delta_{J_s}}$ indicates the Lovász extension of the Jaccard index for class s. Here, (h, w) is obtained from the 3D coordinates (x, y, z) of $\mathbf{P}_i$ by using perspective projection. $\widetilde{\mathbf{m}}(s)\in[0,1]^N$ indicates the vector of errors. The Lovász-softmax loss w.r.t. the camera stream is defined as

$$\mathcal{L}_{lov} = \frac{1}{S}\sum_{s=1}^{S}\overline{\Delta_{J_s}}(\mathbf{m}(s)), \tag{15}$$

where

$$\mathbf{m}_i(s) = \begin{cases}1-\mathbf{O}_{s,h,w} & \text{if } s=\mathbf{Y}_{h,w},\\ \mathbf{O}_{s,h,w} & \text{otherwise.}\end{cases} \tag{16}$$

By considering the objective functions of both LiDAR stream and camera stream, we formulate the objective function of the proposed two-stream network as

$$\mathcal{L} = \widetilde{\mathcal{L}}_{foc} + \mathcal{L}_{foc} + \widetilde{\lambda}\widetilde{\mathcal{L}}_{lov} + \lambda\mathcal{L}_{lov} + \widetilde{\gamma}\widetilde{\mathcal{L}}_{per} + \gamma\mathcal{L}_{per}, \tag{17}$$

where $\lambda, \widetilde{\lambda}, \gamma$ and $\widetilde{\gamma}$ indicate the hyper-parameters that balance different losses.

E. Pipeline of Post-Processing

With the proposed perception-aware losses, PMF generates dense segmentation results with information from RGB images and point clouds. We then obtain the sparse prediction from the dense results. Let $\widetilde{\mathbf{O}}\in\mathbb{R}^{S\times H\times W}$ be the output probabilities of the LiDAR stream. S indicates the number of classes. H and W indicate the height and width of the predictions, respectively. Let $\widehat{\mathbf{Y}}\in\mathbb{R}^{H\times W}$ be the dense predictions from the LiDAR stream. Then, the dense predictions are computed by

$$\widehat{\mathbf{Y}}_{h,w} = \arg\max_{s}\widetilde{\mathbf{O}}_{s,h,w}. \tag{18}$$

Let $\widehat{\mathbf{y}}\in\mathbb{R}^N$ be the sparse predictions of point cloud $\mathbf{P}$. As shown in Fig. 6, for each point $\mathbf{P}_i$, we first project the 3D

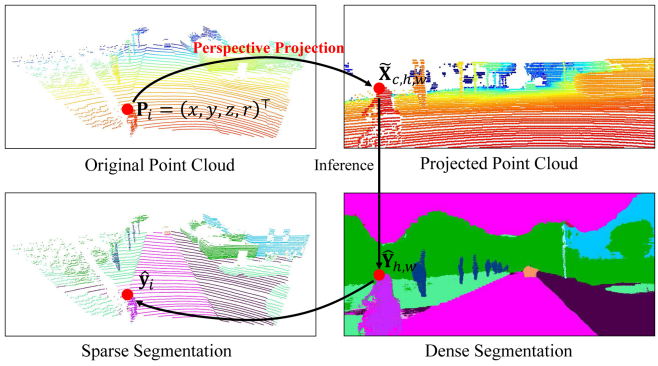


Fig. 6. Illustration of the pipeline to obtain the sparse segmentation from the dense prediction results. $\widehat{\mathbf{X}}$ indicates the projected point cloud. $\widehat{\mathbf{Y}}$ and $\widehat{\mathbf{y}}$ indicate the dense predictions and sparse predictions, respectively. For each point $\mathbf{P}_i$, we first compute the corresponding pixel (h, w) in the camera coordinate system by perspective projection. Second, we get the dense segmentation $\widehat{\mathbf{Y}}$ from the prediction results of PMF. Last, we obtain the corresponding sparse prediction $\widehat{\mathbf{y}}_i$ w.r.t. the point $\mathbf{P}_i$ from the dense segmentation $\widehat{\mathbf{Y}}_{h,w}$.

coordinates (x, y, z) to the camera coordinate system by using perspective projection and compute the corresponding pixel (h, w) in the projected image. Then the semantic prediction $\widehat{\mathbf{y}}_i$ w.r.t. the point $\mathbf{P}_i$ is computed by

$$\widehat{\mathbf{y}}_i := \widehat{\mathbf{Y}}_{h,w}. \tag{19}$$

Note that for the point cloud with multi-camera views, *e.g.*, nuScenes, there are overlaps between different camera views. To address this issue, we conduct inference for each camera view and merge the results by assigning the predictions with the highest confidence scores to the points in the overlaps of different views.

F. Techniques to Improve Efficiency and Effectiveness

On top of the two-stream network designed in [82], we further explore the techniques to improve the efficiency and effectiveness of the fusion network.[4]

Dropping decoder of camera stream. By introducing the perception-aware loss, the knowledge of the camera stream is distilled into the LiDAR stream. Therefore, we only use the predictions of LiDAR stream while dropping the results of the camera stream during inference. In this sense, we can also drop the decoder of the camera stream to speed up inference.

Improved contextual module. In [15], Cortinhal et al. have designed the contextual module that improves the ability of SalsaNet [1] for comprehending global contextual information. However, the proposed contextual module is explored under spherical projection and may be ineffective with perspective projection. In our experiments, we find that the high resolution of the projected point cloud feature is unnecessary due to the sparse nature of point clouds. Therefore, we insert extra down-sampling operation into the contextual module of LiDAR stream to reduce the resolution of point cloud features and improve the efficiency of LiDAR stream. Moreover, to reduce the impact of the sparse issue of projected point clouds, we replace the convolutional layers in the contextual modules of LiDAR stream with sparse invariant convolutional layers [63].

Fusing high-level LiDAR features. In the two-stream network, the camera stream generates predictions with RGB images only, which, however, results in the unsatisfactory segmentation performance of the camera stream and may limit in performance of the fusion network. To address this issue, we further fuse the high-level features from the last stage of the LiDAR stream backbone into the camera stream by a concatenate operation. In this way, we improve the performance of the camera stream without introducing extra computational costs and thus boost the effectiveness of the perception-aware loss.

[4]We study the effect of the proposed techniques in Section V-C

TABLE I
COMPARISONS OF THE NUMBER OF POINTS OF SEMANTICKITTI AND SEMANTICKITTI-FV

Dataset splits	SemanticKITTI	SemanticKTTI-FV	Percentage
Training set	2.35×10^9	3.78×10^8	16.03%
Validation set	4.99×10^8	8.00×10^7	16.07%

IV. EXPERIMENTS

In this section, we first compare EPMF with the state-of-the-art methods on the benchmark data sets. Then, we provide distance-based evaluation and qualitative results of our methods. Last, we conduct experiments to evaluate the efficiency of our method.

We organize the experiments as follows. 1) We introduce the benchmark data sets and evaluation metrics in Section IV-A. 2) We provide the implementation details of our method in Section IV-B. 3) We evaluate the performance of our method on several benchmark data sets in Section IV-C. 4) We investigate the performance of our method under different distances in Section IV-D. 5) We discuss the efficiency of the proposed method in Section IV-E. 6) We show the qualitative results in Section IV-F. 7) We study the robustness of the proposed method on adversarial samples in Section IV-G.

A. Data Sets and Evaluation Metrics

1) Data Sets: We empirically evaluate our method on several benchmark data sets, including SemanticKITTI-FV [3], nuScenes [7], and A2D2 [23].

SemanticKITTI is a large-scale data set based on the KITTI Odometry Benchmark [20], providing 43,000 scans with point-wise semantic annotation, where 21,000 scans (sequence 00-10) are available for training and validation. The data set has 19 semantic classes for the evaluation of semantic benchmarks. The point clouds are collected using a Velodyne HDL-64E sensor, which has 64 beams vertically. Since SemanticKITTI provides only the images of the front-view camera, we project the point clouds to a perspective view and keep only the available points on the images to build a subset of SemanticKITTI, namely, SemanticKITTI-FV. The comparisons between SemanticKITTI and SemanticKITTI-FV are shown in Tables I and II. SemanticKITTI-FV has only 15.93% data for training compared with the full SemanticKITTI data set.

TABLE II
COMPARISONS OF THE NUMBER OF VALID LABELS OF SEMANTICKITTI AND SEMANTICKITTI-FV

Dataset splits	SemanticKITTI	SemanticKTTI-FV	Percentage
Training set	2.28×10^9	3.63×10^8	15.93%
Validation set	4.77×10^8	7.57×10^7	15.88%

nuScenes contains 1,000 driving scenes with different weather and light conditions. The scenes are split into 28,130 training frames and 6,019 validation frames, which are collected with a Velodyne HDL-32E sensor. Unlike SemanticKITTI, which provides only the images of the front-view camera, nuScenes has 6 cameras for different views of LiDAR.

A2D2 provides 38-class semantic segmentation images and point cloud labels for 41,277 non-sequential frames, which are collected with six cameras and five Velodyne VLP-16 sensors. Following [77], we split 22,408 scans for training, 2,274 for validation, and 13,264 for testing, respectively. Similar to nuScenes, the sensor suite of A2D2 provides full 360° coverage. However, the camera and LiDAR sensor orientations are optimized manually to minimize the blind spot around the vehicle and maximize camera and LiDAR field of view overlap, which makes A2D2 lack horizontal scan lines.

2) Evaluation Metrics: To evaluate the performance of our method, we use the mean Intersection over Union (mIoU) as the evaluation metric following the official guidance in [3], [7]. To class s, the respective intersection over union $\mathrm{IoU_s}$ is defined by

$$\mathrm{IoU_s} = \frac{|\mathcal{P}_\mathrm{s} \cap \mathcal{G}_\mathrm{s}|}{|\mathcal{P}_\mathrm{s} \cup \mathcal{G}_\mathrm{s}|}, \tag{20}$$

where $\mathcal{P}_s$ is the set of point with a class prediction s, $\mathcal{G}_s$ is the set of label for class s, and $|\cdot|$ represents the cardinality of the set. Then, the mIoU is formulated as $\mathrm{mIoU} = \frac{1}{\mathrm{S}} \sum_{\mathrm{s}=1}^{\mathrm{S}} \mathrm{IoU_s}$.

B. Implementation Details

We implement the proposed method in PyTorch [53], and use ResNet-34 [28] and SalsaNext [15] as the backbones of the camera stream and LiDAR stream, respectively. Because we process the point clouds in the camera coordinates, we incorporate ASPP [9] into the LiDAR stream network to adjust the receptive field adaptively. To leverage the benefits of existing image classification models, we initialize the parameters of ResNet-34 with the pre-trained ImageNet models from [53]. We also adopt hybrid optimization methods [75] to train the networks w.r.t. different modalities, *i.e.*, SGD with Nesterov [51] for the camera stream and Adam [35] for the LiDAR stream. We train the networks for 50 epochs with a batch size of 8 on SemanticKITTI. On nuScenes and A2D2, the batch size is 24 with 150 epochs for training as the point clouds of these data sets are more sparse. The learning rate starts at 0.001 and decays to 0 with a cosine policy [46]. We tune the hyper-parameters of the objective function in (17) using the weighting strategy proposed in [34]. We set the confidence threshold τ to 0.7 following [82].[5] To prevent over-fitting, a series of data augmentation strategies are used, including random horizontal flipping, random scaling, color jitter, 2D random rotation, and random cropping.

[5]The ablation study of τ is in the supplementary materials of [82].

C. Comparisons on Benchmark Data Sets

1) Results on SemanticKITTI-FV: To evaluate our method on SemanticKITTI-FV, we compare EPMF with several state-of-the-art LiDAR-only methods including SalsaNext [15], Cylinder3D [81], 2DPASS [72], etc. Following [15], [33], [81], we use sequence 08 for validation. The remaining sequences (00-07 and 09-10) are used as the training set. We evaluate the release models of the state-of-the-art LiDAR-only methods on our data set. Because SPVNAS [62] did not release its best model, we report the result of the best-released model (with 65G MACs). In addition, we re-implement two fusion-based methods, *i.e.*, RGBAL [47] and PointPainting [64] on our data set. To further understand the effectiveness of our proposed fusion strategies, we construct a *uni-modal baseline*, which uses the same network architecture of LiDAR stream of the proposed two-stream network. Note that we do not use test time augmentation (TTA) during evaluation as this technique is time-consuming and cannot be adopted to real applications on the car.

From Table III, EPMF outperfoms the uni-modal baseline by 7.3% in mIoU. Compared with PMF, our EPMF also achieves 2.0% improvements in mIoU. However, EPMF performs slightly worse than the pre-trained model of 2DPASS [72] on SemanticKITTI-FV. Note that our EPMF is trained on SemanticKITTI-FV with only 16.03% points of SemanticKITTI. For fair comparisons, we also train 2DPASS using the officially released code and evaluate the model on SemanticKITTI-FV. In this case, our EPMF outperforms 2DPASS by 4.1% in mIoU.

2) Results on Nuscenes: Following [81], to evaluate our method on more complex scenes, we compare EPMF with the state-of-the-art methods on the nuScenes LiDAR-seg validation set. The experimental results are shown in Table IV. Note that the point clouds of nuScenes are sparser than those of SemanticKITTI (35 k points/frame versus 125 k points/frame). Thus, it is more challenging for 3D segmentation tasks. In this case, EPMF achieves the best performance on nuScenes validation set. Specifically, EPMF outperforms the best LiDAR-only method, *i.e.*, SphereFormer, by 2.2% in mIoU. Compared with PMF, our EPMF achieves 3.7% improvements in mIoU.

In addition, we further provide the results on nuScenes test set. Different from the results on the nuScenes validation set, existing methods always use additional techniques, including test-time augmentation or fine-tuning with class re-sampling that improves performance on rare classes, to pursue better results on the leaderboard. Unlike existing methods, we directly evaluate our model on the test set without additional techniques. For fair comparisons, we also evaluate the performance of the released models of SphereFormer without TTA or fine-tuning. As shown in Table V, our EPMF outperforms SphereFormer [38] by 1.1% in mIoU under the same evaluation settings. Compared with PMF, our EPMF consistently achieves 3.7% improvements in mIoU on nuScenes test set. These results are consistent with our expectations. Since EPMF incorporates RGB images, our fusion

TABLE III
COMPARISONS ON SEMANTICKITTI-FV VALIDATION SET

Method	Modality	Car	Bicycle	Motorcycle	Truck	Other-vehicle	Person	Bicyclist	Motorcyclist	Road	Parking	Sidewalk	Other-ground	Building	Fence	Vegetation	Trunk	Terrain	Pole	Traffic-sign	mIoU (%)
RandLANet [31]	L	92.0	8.0	12.8	74.8	46.7	52.3	46.0	0.0	93.4	32.7	73.4	0.1	84.0	43.5	83.7	57.3	73.1	48.0	27.3	50.0
RangeNet++ [50]	L	89.4	26.5	48.4	33.9	26.7	54.8	69.4	0.0	92.9	37.0	69.9	0.0	83.4	51.0	83.3	54.0	68.1	49.8	34.0	51.2
SequeezeSegV2 [67]	L	82.7	15.1	22.7	25.6	26.9	22.9	44.5	0.0	92.7	39.7	70.7	0.1	71.6	37.0	74.6	35.8	68.1	21.8	22.2	40.8
SequeezeSegV3 [69]	L	87.1	34.3	48.6	47.5	47.1	58.1	53.8	0.0	95.3	43.1	78.2	0.3	78.9	53.2	82.3	55.5	70.4	46.3	33.2	53.3
SalsaNext [15]	L	90.5	44.6	49.6	86.3	54.6	74.0	81.4	0.0	93.4	40.6	69.1	0.0	84.6	53.0	83.6	64.3	64.2	54.4	39.8	59.4
MinkowskiNet [13]	L	95.0	23.9	50.4	55.3	45.9	65.6	82.2	0.0	94.3	43.7	76.4	0.0	87.9	57.6	87.4	67.7	71.5	63.5	43.6	58.5
SPVNAS [62]	L	96.5	44.8	63.1	59.9	64.3	72.0	86.0	0.0	93.9	42.4	75.9	0.0	88.8	59.1	88.0	67.5	73.0	63.5	44.3	62.3
Cylinder3D [81]	L	96.4	61.5	78.2	66.3	69.8	80.8	93.3	0.0	94.9	41.5	78.0	1.4	87.5	50.0	86.7	72.2	68.8	63.0	42.1	64.9
2DPASS [72]	L+C	96.6	60.8	71.6	82.0	77.8	78.2	92.0	0.2	93.9	44.4	75.7	3.1	89.5	60.9	88.7	72.6	74.0	61.5	45.5	**66.8**
2DPASS[l][72]	L+C	93.6	54.9	71.6	81.6	43.5	71.5	82.2	0.2	94.0	33.8	75.1	0.2	88.7	57.3	88.5	67.1	75.1	56.0	40.2	61.8
PointPainting* [64]	L+C	94.7	17.7	35.0	28.8	55.0	59.4	63.6	0.0	95.3	39.9	77.6	0.4	87.5	55.1	87.7	67.0	72.9	61.8	36.5	54.5
RGBAL* [47]	L+C	87.3	36.1	26.4	64.6	54.6	58.1	72.7	0.0	95.1	45.6	77.5	0.8	78.9	53.4	84.3	61.7	72.9	56.1	41.5	56.2
PMF [82]	L+C	95.4	47.8	62.9	68.4	75.2	78.9	71.6	0.0	96.4	43.5	80.5	0.1	88.7	60.1	88.6	72.7	75.3	65.5	43.0	63.9
Uni-modal baseline	L	93.2	35.9	33.9	78.2	54.4	63.1	71.9	0.0	95.0	42.4	80.0	0.7	86.0	53.7	86.1	63.5	73.7	59.4	41.9	58.6
EPMF (Ours)	L+C	95.4	52.5	66.3	82.4	80.5	77.2	85.1	0.0	95.8	48.1	80.5	0.8	89.2	66.1	86.0	70.0	68.9	62.6	43.6	65.9

[l]Model trained under our settings.
L indicates LiDAR-only methods. L+C indicates fusion-based methods. * Indicates the results based on our implementation. The bold numbers indicate the best results.

TABLE IV
COMPARISONS ON THE NUSCENES VALIDATION SET

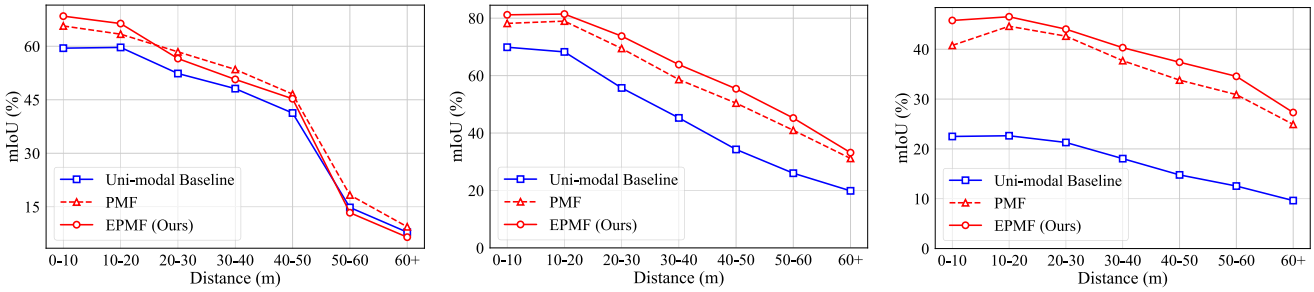
Method	Modality	Barrier	Bicycle	Bus	Car	Construction	Motorcycle	Pedestrian	Traffic-cone	Trailer	Truck	Driveable	Other-flat	Sidewalk	Terrain	Manmade	Vegetation	mIoU (%)
RangeNet++ [50]	L	66.0	21.3	77.2	80.9	30.2	66.8	69.6	52.1	54.2	72.3	94.1	66.6	63.5	70.1	83.1	79.8	65.5
PolarNet [77]	L	74.7	28.2	85.3	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	71.0
Salsanext [15]	L	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	72.2
SVASeg [79]	L	73.1	44.5	88.4	86.6	48.2	80.5	77.7	65.6	57.5	82.1	96.5	70.5	74.7	74.6	87.3	86.9	74.7
PVKD [30]	L	76.2	40.0	90.2	94.0	50.9	77.4	78.8	64.7	62.0	84.1	96.6	71.4	76.4	76.3	90.3	86.9	76.0
AMVNet [42]	L	79.8	32.4	82.2	86.4	62.5	81.9	75.3	72.3	83.5	65.1	97.4	67.0	78.8	74.6	90.8	87.9	76.1
Cylinder3D [81]	L	76.4	40.3	91.3	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	76.1
RPVNet [70]	L	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	77.6
SDSeg3D [39]	L	77.5	49.4	93.9	92.5	54.9	86.7	80.1	67.8	65.7	86.0	96.4	74.0	74.9	74.5	86.0	82.8	77.7
SDSeg3D† [39]	L	78.2	52.8	94.5	93.1	54.5	88.1	82.2	69.4	67.3	86.6	96.4	74.5	75.2	75.3	87.1	84.1	78.7
WaffleIron [54]	L	78.7	51.3	93.6	88.2	47.2	86.5	81.7	68.9	69.3	83.1	96.9	74.3	75.6	74.2	87.2	85.2	77.6
WaffleIron† [54]	L	79.8	53.8	94.3	87.6	49.6	89.1	83.8	70.6	72.7	84.9	97.1	75.8	76.5	75.9	87.8	86.3	79.1
RangeFormer [36]	L	78.0	45.2	94.0	92.9	58.7	83.9	77.9	69.1	63.7	85.6	96.7	74.5	75.1	75.3	89.1	87.5	78.1
SphereFormer [38]	L	77.7	43.8	94.5	93.1	52.4	86.9	81.2	65.4	73.4	85.3	97.0	73.4	75.4	75.0	91.0	89.2	78.4
SphereFormer† [38]	L	78.7	46.7	95.2	93.7	54.0	88.9	81.1	68.0	74.2	86.2	97.2	74.3	76.3	75.8	91.4	89.7	79.5
2DPASS [72]	L+C	74.4	44.3	93.6	92.0	54.0	79.7	78.9	57.2	72.5	85.7	96.2	72.7	74.1	74.5	87.5	85.4	76.4
2DPASS† [72]	L+C	77.0	50.4	95.9	94.2	56.0	86.0	81.9	64.4	76.9	88.6	96.8	75.4	76.7	76.4	88.9	86.6	79.5
2D3DNet [22]	L+C	78.3	55.1	95.4	87.7	59.4	79.3	80.7	70.2	68.2	86.6	96.1	74.9	75.7	75.1	91.4	89.9	79.0
PMF [82]	L+C	74.1	46.6	89.8	92.1	57.0	77.7	80.9	70.9	64.6	82.9	95.5	73.3	73.6	74.8	89.4	87.7	76.9
PMF-R50 [82]	L+C	74.9	55.4	91.0	93.0	60.5	80.3	83.2	73.6	67.2	84.5	95.9	75.1	74.6	75.5	90.3	89.0	79.0
Uni-modal baseline	L	74.9	19.2	73.5	89.3	36.9	63.1	68.2	52.0	64.5	74.4	96.8	73.0	75.6	75.1	87.8	86.0	69.4
EPMF (Ours)	L+C	79.7	55.8	96.0	92.4	65.6	86.4	80.9	74.3	68.1	87.0	97.0	75.6	76.2	76.3	90.2	88.1	**80.6**

L indicates LiDAR-only methods. L+C indicates fusion-based methods. †Indicates the results with test-time augmentation. The bold numbers indicate the best results.

TABLE V
COMPARISONS ON THE NUSCENES TEST SET

Method	Modality	Barrier	Bicycle	Bus	Car	Construction	Motorcycle	Pedestrian	Traffic-cone	Trailer	Truck	Driveable	Other-flat	Sidewalk	Terrain	Manmade	Vegetation	mIoU (%)
PolarNet [77]	L	72.2	16.8	77.0	86.5	51.1	69.7	64.8	54.1	69.7	63.4	96.6	67.1	77.7	72.1	87.1	84.4	69.4
AMVNet [42]	L	79.8	32.4	82.2	86.4	62.5	81.9	75.3	72.3	83.5	65.1	97.4	67.0	78.8	74.6	90.8	87.9	76.1
Cylinder3D* [81]	L	82.8	29.8	84.3	89.4	63.0	79.3	77.2	73.4	84.6	69.1	97.7	70.2	80.3	75.5	90.4	87.6	77.2
SPVNAS* [62]	L	80.0	30.0	91.9	90.8	64.7	79.0	75.6	70.9	81.0	74.6	97.4	69.2	80.0	76.1	89.3	87.1	77.4
AF2S3Net* [12]	L	78.9	52.2	89.9	84.2	77.4	74.3	77.3	72.0	83.9	73.8	97.1	66.5	77.5	74.0	87.7	86.8	78.3
RangeFormer [36]	L	83.9	46.1	89.4	89.2	70.3	83.3	75.4	72.5	81.4	71.1	95.6	68.5	77.3	73.4	89.3	86.9	78.3
RangeFormer† [36]	L	85.6	47.4	91.2	90.9	70.7	84.7	77.1	74.1	83.2	72.6	97.5	70.7	79.2	75.4	91.3	88.9	80.1
SphereFormer [38]	L	81.5	39.7	93.4	87.5	66.4	75.7	77.2	70.6	85.6	73.6	97.6	64.8	79.8	75.0	92.2	89.0	78.1
SphereFormer†‡ [38]	L	83.3	39.2	94.7	92.5	77.5	84.2	84.4	79.1	88.4	78.3	97.9	69.0	81.5	77.2	93.4	90.2	**81.9**
2DPASS†‡ [72]	L+C	81.7	55.3	92.0	91.8	73.3	86.5	78.5	72.5	84.7	75.5	97.6	69.1	79.9	75.5	90.2	88.0	80.8
2D3DNet‡ [22]	L+C	83.0	59.4	88.0	85.1	63.7	84.4	82.0	76.0	84.8	71.9	96.9	67.4	79.8	76.0	92.1	89.2	80.0
PMF [82]	L+C	80.1	35.7	79.7	86.0	62.4	76.3	76.9	73.6	78.5	66.9	97.1	65.3	77.6	74.4	89.5	87.7	75.5
PMF-R50 [82]	L+C	82.1	40.3	80.9	86.4	63.7	79.2	79.8	75.9	81.2	67.1	97.3	67.7	78.1	74.5	89.9	88.5	77.0
EPMF (Ours)	L+C	76.9	39.8	90.3	87.8	72.0	86.4	79.6	76.6	84.1	74.9	97.7	66.4	79.5	76.4	91.1	87.9	79.2

L indicates LiDAR-only methods. L+C indicates fusion-based methods. * Represents the results reported in the leader board of nuScenes. † Indicates the results with test-time augmentation. ‡ Denotes the results with additional fine-tuning with class re-sampling before the leaderboard submission. The bold numbers indicate the best results.



(a) Comparisons on SemanticKITTI-FV (b) Comparisons on nuScenes (c) Comparisons on A2D2

Fig. 7. Distance-based evaluation on SemanticKITTI-FV, nuScenes and A2D2. As the distance increases, the point cloud becomes sparser.

TABLE VI
COMPARISONS ON A2D2 TEST SET - PART1

Method	Car	Bicycle	Pedestrian	Truck	Small-vehi	Traffic-signal	Traffic-signl	Utility-vehi	Sidebars	Bumper	Curbstone	Solid line	Irrelevant signs	Road blocks	Tractor	Non-drivable	Zebra crossing	Obstacles	Poles	mIoU (%)
SqueezeSeg [66]	9.7	0.0	0.0	15.8	0.0	0.7	64.4	0.0	0.4	0.0	2.2	15.6	0.5	15.9	0.0	0.0	0.0	0.0	0.3	8.9
SqueezeSegV2 [67]	15.4	0.2	8.6	63.8	0.0	16.8	61.7	0.6	0.1	0.0	14.8	24.7	12.7	33.2	0.0	5.8	0.0	0.2	5.2	16.4
DarkNet53 [3]	15.2	0.8	6.1	68.5	0.0	15.5	63.8	0.4	0.3	0.0	17.3	23.8	13.3	35.6	0.0	6.3	0.0	3.9	7.6	17.2
PolarNet [77]	23.8	10.1	18.2	69.7	9.6	49.1	58.5	0.0	11.3	0.0	28.3	37.6	24.8	42.8	0.0	14.8	0.0	8.0	11.0	23.9
Cylinder3D [80]	32.6	6.0	14.9	74.7	20.9	51.2	65.9	3.6	6.8	0.0	33.2	42.8	24.1	40.7	0.1	16.7	0.0	10.8	5.4	24.2
2DPASS [72]	26.2	1.1	16.9	78.7	12.4	39.0	66.2	2.8	4.8	0.0	19.9	20.0	18.8	31.4	0.0	6.0	0.0	9.9	3.4	18.9
PMF [82]	84.5	36.0	47.5	89.4	38.1	75.0	82.8	4.6	64.9	0.0	54.8	72.8	41.2	77.1	0.0	27.1	0.0	30.8	29.6	41.8
Uni-modal baseline	42.3	5.6	13.7	75.2	0.6	32.2	65.6	0.2	13.3	0.0	30.6	50.5	12.3	53.9	0.1	6.9	0.0	6.1	9.2	21.8
EPMF(Ours)	79.4	42.3	49.4	90.7	79.8	78.4	84.4	11.8	69.1	0.0	56.8	73.9	47.6	78.3	0.0	26.0	0.0	35.5	33.0	**45.0**

The bold numbers indicate the best results.

TABLE VII
COMPARISONS ON A2D2 TEST SET - PART2

Method	RD restricted area	Animals	Grid structure	Signal corpus	Drivable cobbleston	Electronic traffic	Slow drive area	Nature object	Parking area	Sidewalk	Ego car	Painted driv instr	Traffic guide obj	Dashed line	RD normal street	Sky	Buildings	Blurred area	Rain dirt	mIoU (%)
SqueezeSeg [66]	0.0	0.0	0.0	0.0	0.0	0.0	0.0	64.5	0.0	13.7	0.0	0.0	0.1	0.2	77.7	10.4	27.7	0.0	0.0	8.9
SqueezeSegV2 [67]	29.5	0.0	10.3	5.5	2.7	0.0	1.9	76.4	3.8	29.2	0.0	6.4	12.4	17.1	85.8	12.1	50.9	0.0	0.0	16.4
DarkNet53 [3]	38.7	0.0	10.8	4.4	3.3	0.0	0.0	77.9	3.1	31.5	0.0	9.4	7.3	15.7	86.4	12.9	55.2	0.0	0.0	17.2
PolarNet [77]	55.6	0.0	14.8	11.9	7.0	0.0	4.4	81.6	12.8	42.5	0.0	12.7	11.5	31.8	90.3	9.2	57.0	0.0	0.0	23.9
Cylinder3D [80]	57.7	0.0	20.5	10.6	10.4	1.5	2.5	82.9	9.1	47.4	0.0	19.1	8.0	33.8	89.7	13.0	63.5	0.0	0.0	24.2
2DPASS [72]	37.0	0.0	13.8	6.7	5.0	0.0	0.3	74.8	5.9	35.1	0.0	8.6	10.0	11.9	85.6	6.9	57.5	0.0	0.0	18.9
PMF [82]	65.4	0.0	46.6	22.6	26.3	0.0	3.7	91.7	17.8	48.6	0.0	48.9	64.1	69.2	94.1	53.5	78.4	0.0	0.0	41.8
Uni-modal baseline	45.7	0.0	14.0	2.7	9.1	0.0	0.1	74.6	3.1	38.1	0.0	12.3	11.5	35.5	90.2	14.2	59.0	0.0	0.0	21.8
EPMF(Ours)	70.3	0.0	52.1	33.3	29.4	0.0	1.5	92.4	15.3	57.4	0.0	55.0	66.3	70.0	94.5	54.2	80.5	0.0	0.0	**45.0**

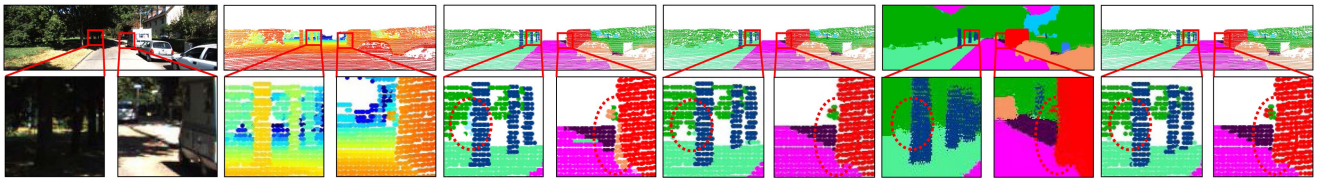
The bold numbers indicate the best results.

strategy is capable of addressing such challenging segmentation under extremely sparse point clouds.

3) Results on A2D2: To further evaluate the performance of our method on more sparse and irregular point clouds, we conduct experiments on A2D2 and compare our EPMF to existing methods. For a fair comparison, we select the model with the best validation performance and report the evaluation results on the test set. Note that both Cylinder3D and 2DPASS did not report the results on A2D2 test set, we also train Cylinder3D and 2DPASS using the officially released code and report the results on the test set. As shown in Tables VI and VII, both our PMF and EPMF outperform the LiDAR-only method by a large margin. Specifically, our EPMF outperforms Cylinder3D by 20.8% in mIoU. Compared to uni-modal baseline, EPMF achieves 23.2% improvements in mIoU, which indicates the effectiveness of our proposed fusion strategies. Moreover, our EPMF outperforms PMF consistently, with 3.2% improvement in mIoU.

D. Distance-Based Evaluation

In 3D LiDAR perception, the point cloud becomes sparser with the increase of perception distance. As long-range perception is important to the safety of autonomous cars, we further conduct a distance evaluation on the benchmark datasets and investigate the performance of our method under different distances. As shown in Fig. 7, both PMF and EPMF outperform the uni-modal baseline by a large margin under different distances on nuScenes and A2D2, which indicates that our fusion strategy can incorporate the information from RGB images effectively.



(a) Input Images (b) Input Point Clouds (c) Cylinder3D (d) PMF (Ours) (e) PMF-dense (Ours) (f) Ground-truth

Fig. 8. Qualitative results on SemanticKITTI-FV. The red dashed circle indicates the difference between the results of PMF and the baseline.

TABLE VIII
COMPARISONS OF THE NUMBER OF CLASSES AT DIFFERENT DISTANCES

Dataset	Distance (m)						
	0-10	10-20	20-30	30-40	40-50	50-60	60+
SemanticKITTI-FV	18	19	19	19	19	11	8
nuScenes	16	16	16	16	16	16	16
A2D2	33	35	35	34	34	33	34

TABLE IX
PERCENTAGE OF POINT CLOUD DISTRIBUTION (%) AT DIFFERENT DISTANCES

Dataset	Distance (m)						
	0-10	10-20	20-30	30-40	40-50	50-60	60+
SemanticKITTI-FV	34.7	41.5	12.9	5.2	2.7	1.5	1.5
nuScenes	70.8	15.7	6.3	3.2	1.8	1.0	1.2
A2D2	27.4	36.9	19.5	9.4	4.1	1.6	1.1

We also notice that EPMF cannot consistently outperform PMF or the uni-modal baseline on SemanticKITTI-FV. We argue that this phenomenon is mainly caused by the following reasons. First, we compare the number of classes at different distances w.r.t. the validation set of SemanticKITTI-FV and nuScenes, as well as the test set of A2D2. From Table VIII, the number of classes on SemanticKITTI-FV decreases at long distances. Since it is difficult for our method to beat the baseline on all the semantic classes of SemanticKITTI-FV, EPMF performs worse than the uni-modal baseline when the distance is larger than 50 m on SemanticKITTI-FV, which covers only 3% of point clouds (See Table IX). Second, to make a trade-off between efficiency and effectiveness, we insert a down-sampling operation into the contextual module of LiDAR stream to improve its efficiency. As the point clouds of SemanticKITTI are dense, reducing the resolution of point cloud features also leads to higher classification errors at long distances. Nevertheless, EPMF achieves better performance when the distance is less than 20 m which covers 76.2% of the points. Overall, our EPMF outperforms PMF by 2.0% in mIoU with 2.06× acceleration on SemanticKITTI-FV.

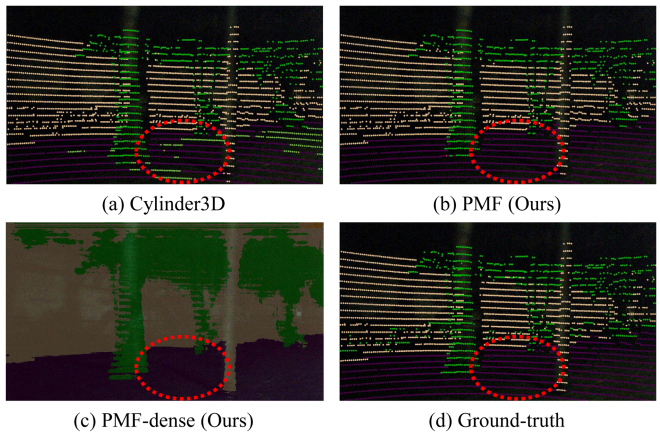
E. Efficiency Analysis

In this section, we evaluate the efficiency of EPMF on GeForce RTX 3090. Note that we consider the efficiency of PMF in two aspects. First, since predictions of the camera stream are fused into the LiDAR stream, we remove the decoder of the camera stream to speed up the inference. Second, both our PMF and EPMF are built on 2D convolutions and can be easily optimized by existing inference toolkits, *e.g.*, TensorRT. In contrast, Cylinder3D is built on 3D sparse convolutions [24] and is difficult to be accelerated by TensorRT. We report the inference time of different models optimized by TensorRT in Table X.

TABLE X
INFERENCE TIME OF DIFFERENT METHODS ON SEMANTICKITTI USING TENSORRT

Method	#FLOPs	#Params.	Inference time	mIoU
PointPainting [64]	51.0 G	28.1 M	2.3 ms	54.5%
RGBAL [47]	55.0 G	13.2 M	2.7 ms	56.2%
SalsaNext [15]	31.4 G	6.7 M	1.6 ms	59.4%
Cylinder3D [81]	-	55.9 M	62.5 ms	64.9%
PMF [82]	854.7 G	36.3 M	22.3 ms	63.9%
EPMF (Ours)	418.0 G	34.2 M	14.2 ms	**65.9%**

"-" Indicates the results that are not available. For a fair comparison, Cylinder3D is accelerated by sparse convolution.



(a) Cylinder3D (b) PMF (Ours) (c) PMF-dense (Ours) (d) Ground-truth

Fig. 9. Qualitative results on nuScenes. We use the corresponding images (night) as the background of both the predictions and labels. We highlight the difference between the results of PMF and the baseline with the red dashed circle.

From the results, our PMF achieves the best performance on nuScenes and is 2.8× faster than Cylinder3D (22.3 ms versus 62.5 ms) with fewer parameters. While compared to PMF, our EPMF achieves 1.6× acceleration (14.2 ms versus 22.3 ms) with 2.0% improvements in mIoU.

F. Qualitative Evaluation

To better understand the benefits of PMF, we visualize the predictions of PMF on the benchmark data sets. From Fig. 8, compared with Cylinder3D, PMF achieves better performance at the boundary of objects. For example, as shown in Fig. 8(d), the truck segmented by PMF has a more complete shape. More critically, PMF is robust in different lighting conditions. Specifically, as illustrated in Fig. 9, PMF outperforms the baselines on more challenging scenes (*e.g.*, night). In addition, as demonstrated in Figs. 8(e) and 9(c), PMF generates dense segmentation

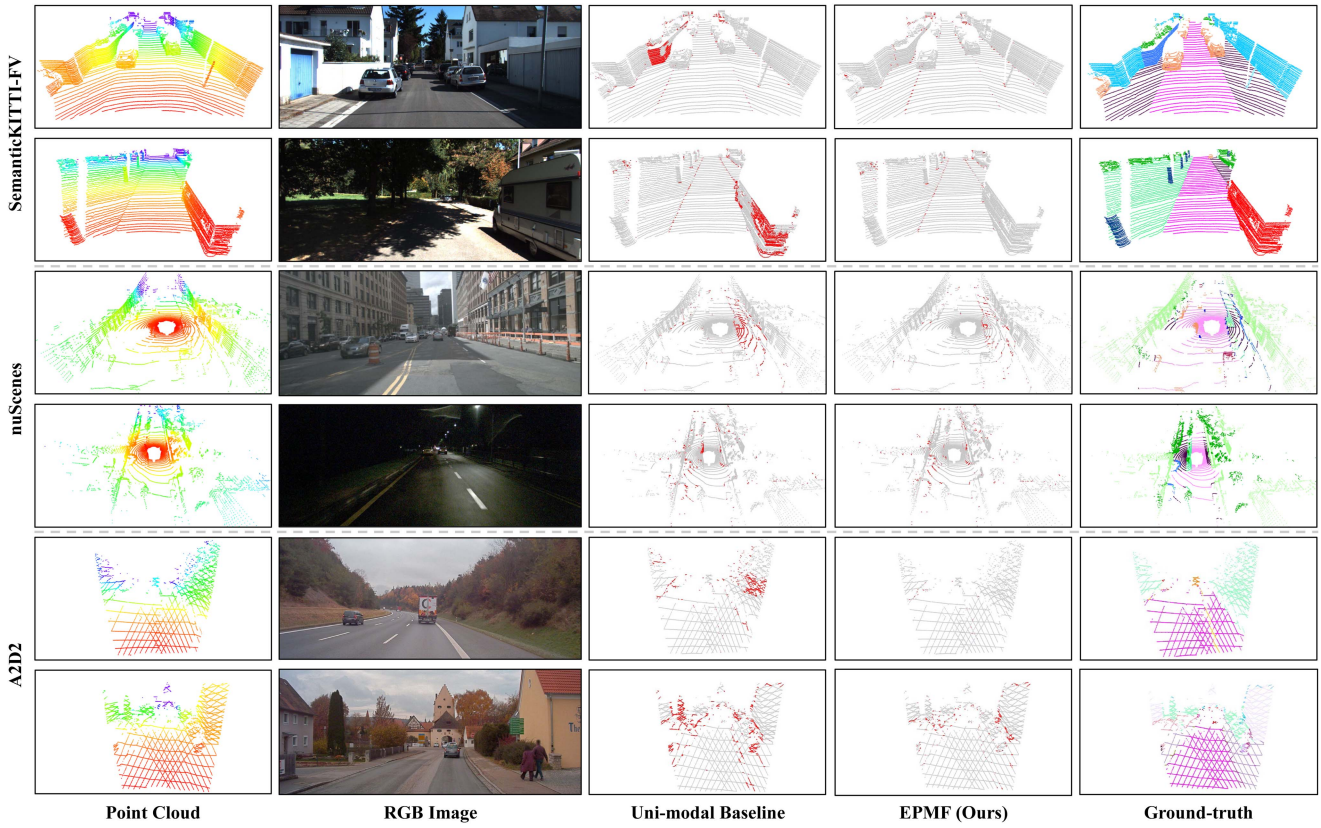


Fig. 10. Qualitative results on three benchmark data sets. We show the error maps of both our method and the uni-modal baseline, in which the red points indicate mispredictions. Zoom in for more details.

results that combine the benefits of both the camera and LiDAR, which are significantly different from existing LiDAR-only and fusion-based methods.

In Fig. 10, we show the error maps of both EPMF and the uni-modal baseline on three benchmark data sets under different scenes. From the results, our EPMF can fully utilize both information from point clouds and RGB images and thus performs well in some challenging scenes with extremely sparse point clouds or poor lighting conditions.

G. Robustness Analysis

To investigate the robustness of EPMF on adversarial samples, we first insert extra objects (*e.g.*, a traffic sign) into the images and keep the point clouds unchanged. In addition, we implement a camera-only method, *i.e.*, FCN [45], on SemanticKITTI as the baseline. Note that we do not use any adversarial training technique during training. As demonstrated in Fig. 11, the camera-only methods are easily affected by changes in the input images. In contrast, because EPMF integrates reliable point cloud information, the noise in the images is reduced during feature fusion and imposes only a slight effect on the model performance.

To investigate the performances of EPMF under different lighting conditions, we evaluate the performance of EPMF on nuScenes validation set at day/night time. We initialize

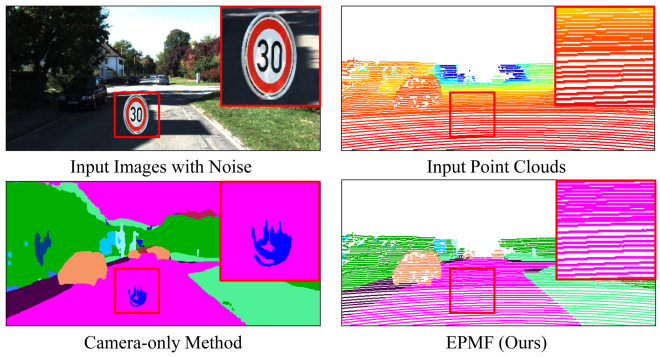


Fig. 11. Comparisons of EPMF and camera-only methods on adversarial samples. The camera-only methods use only RGB images as inputs, while PMF uses both images and point clouds as inputs. We highlight the inserted traffic sign with a red box.

SegFormer-B5 by the officially released weights on CityScape and train the model SegFormer-B5 with the camera or LiDAR as inputs. Note that some of the classes may not appear at night, we only report the mIoU of the available classes. From Table XI, as the camera only provides limited information at night, SegFormer with camera-only inputs performs worse than the LiDAR-only counterpart. By fusing both information from the camera and LiDAR, EPMF achieves better performance at night, with 15.1% improvements in mIoU compared to SegFormer-B5.

TABLE XI
COMPARISONS TO SEGFORMER ON NUSCENES AT DAY/NIGHT TIME

Method	Input	Day	Night
SegFormer-B5 [68]	C	67.6%	41.6%
SegFormer-B5 [68]	L	56.6%	46.9%
EPMF (Ours)	L+C	81.4%	62.0%

C and L indicate the model with camera or LiDAR as input, respectively. L+C indicates the model with both camera and LiDAR inputs.

TABLE XII
ABLATION STUDY FOR THE NETWORK COMPONENTS ON THE SEMANTICKITTI-FV VALIDATION SET

	Baseline	PP	ASPP	RF	PL	mIoU (%)
1	✓					57.2
2	✓	✓				57.6
3	✓	✓	✓			59.7
4	✓		✓	✓		55.8
5	✓	✓	✓	✓		61.7
6	✓	✓	✓	✓	✓	**63.9**

PP denotes perspective projection. RF denotes the residual-based fusion module. PL denotes perception-aware loss. The bold number is the best result.

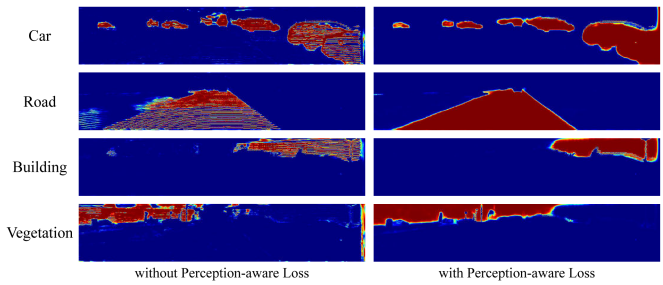


Fig. 12. Comparisons of the predictions w.r.t. the networks trained with and without perception-aware loss. **PL** denotes the perception-aware loss. Red indicates predictions with higher confidence scores. We only show the predictions of Car for the sake of clarity.

V. ABLATION STUDY

A. Effect of Network Components

We study the effect of the network components of PMF, *i.e.*, perspective projection, ASPP, residual-based fusion modules, and perception-aware loss. The experimental results are shown in Table XII. Since we use only the front-view point clouds of SemanticKITTI, we train SalsaNext as the baseline on our data set using the officially released code. Comparing the first and second lines in Table XII, perspective projection achieves only a 0.4% mIoU improvement over spherical projection with LiDAR-only input. In contrast, comparing the fourth and fifth lines, perspective projection brings a 5.9% mIoU improvement over spherical projection with multimodal data inputs. From the third and fifth lines, our fusion modules bring 2.0% mIoU improvement to the fusion network. Moreover, comparing the fifth and sixth lines, the perception-aware losses improve the performance of the network by 2.2% in mIoU.

B. Effect of Perception-Aware Loss

To investigate the effect of perception-aware loss, we visualize the predictions of the LiDAR stream networks with and without perception-aware loss in Fig. 12. From the results, perception-aware loss helps the LiDAR stream capture the perceptual information from the images. For example, the model trained with perception-aware loss learns the complete shape of cars, while the baseline model focuses only on the local features of points. As the perception-aware loss introduces the perceptual difference between the RGB images and the point clouds, it enables an effective fusion of the perceptual information from the data of both modalities. As a result, our PMF generates dense predictions that combine the benefits of both the images and point clouds.

TABLE XIII
ABLATION STUDY FOR THE PROPOSED IMPROVED TECHNIQUES ON SEMANTICKITTI-FV VALIDATION SET

Proposed strategies	mIoU	#FLOPs	#Params.
PMF [82]	63.9%	859.7 G	36.4 M
+dropping decoder of camera stream	63.9%	854.7 G	36.3 M
+cross-modal alignment and crop	64.4%	739.9 G	36.3 M
+improved contextual module	64.8%	418.0 G	34.2 M
+fusing high-level LiDAR feature	65.9%	418.0 G	34.2 M

TABLE XIV
MODEL PERFORMANCE WITH DIFFERENT IMAGE MASKED RATIOS ON SEMANTICKITTI-FV

Masked Ratio (%)	0	10	20	30	40	50
without fine-tuning	65.9	65.5	64.8	63.0	60.0	54.8
with fine-tuning	65.7	65.4	65.0	64.2	62.9	60.6

We report the mIoU (%) with/without fine-tuning.

C. Effect of the Improved Techniques

In this section, we explore the effectiveness of the proposed improved techniques on SemanticKITTI-FV. As shown in Table XIII, on top of PMF, dropping the decoder of the camera stream saves 5.0 GFLOPs in computation budgets without performance degradation. The proposed CAC further reduces 114.8 GFLOPs in computation costs by removing the useless area of RGB images. With the proposed improved contextual module, we achieve $2.04\times$ acceleration compared to PMF (418.0 GFLOPs versus 854.7 GFLOPs) with 0.9% improvement in mIoU. By fusing the high-level features of LiDAR stream into the camera stream, we further achieve 1.1% improvements in mIoU.

When conducting CAC, if the LiDAR has a larger vertical FOV than cameras, keeping point clouds outside the image also results in partial blank image inputs. To investigate the impact of these areas without image information, we partially mask the RGB image from bottom to top with ratios from 10% to 50%, and evaluate the model performance on SemanticKITTI-FV. As shown in Table XIV, masking 10% of RGB image only results in 0.4% degradation of mIoU. With the increasing of the masked ratio, the model performance suffers from significant degradation. Nevertheless, the impact of masked images can be mitigated by introducing the image mask into training or fine-tuning.

VI. CONCLUSION

In this work, we have proposed a perception-aware multi-sensor fusion scheme for 3D LiDAR semantic segmentation. Unlike existing methods that conduct feature fusion in the

LiDAR coordinate system, we project the point clouds to the camera coordinate system to enable a collaborative fusion of the perceptual features from the two modalities. By fusing complementary information from both cameras and LiDAR, PMF is robust in complex outdoor scenes with extremely sparse point clouds or poor lighting conditions. Moreover, we propose EPMF which improves the efficiency and effectiveness of PMF. Specifically, we introduce cross-modal alignment and cropping to reduce unnecessary computation. Besides, we also adjust the architecture of the fusion network by fusing high-level features into the camera stream and exploring the design of contextual modules under perspective projection. The experimental results on three benchmarks show the superiority of our method. In the future, we will extend EPMF to other challenging tasks, *e.g.*, object detection and semantic scene completion.

REFERENCES

[1] E. E. Aksoy, S. Baci, and S. Cavdar, "SalsaNet: Fast road and vehicle segmentation in LiDAR point clouds for autonomous driving," in *Proc. IEEE Intell. Veh. Symp.*, 2020, pp. 926–932.

[2] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.

[3] J. Behley et al., "SemanticKITTI: A dataset for semantic scene understanding of LiDAR sequences," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 9297–9307.

[4] M. Berman, A. R. 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4413–4421.

[5] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOV4: Optimal speed and accuracy of object detection," pp. 1–17, 2020, *arXiv: 2004.10934*.

[6] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2008, pp. 44–57.

[7] H. Caesar et al., "nuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11621–11631.

[8] H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han, "Once-for-all: Train one network and specialize it for efficient deployment," in *Proc. Int. Conf. Learn. Representations*, 2020, pp. 1–15.

[9] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[10] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," pp. 1–14, 2017, *arXiv: 1706.05587*.

[11] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 1290–1299.

[12] R. Cheng, R. Razani, E. Taghavi, E. Li, and B. Liu, "2-S3Net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 12547–12556.

[13] C. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal convnets: Minkowski convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3075–3084.

[14] M. Cordts et al., "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3213–3223.

[15] T. Cortinhal, G. Tzelepis, and E. Erdal Aksoy, "SalsaNext: Fast, uncertainty-aware semantic segmentation of LiDAR point clouds," in *Proc. Int. Symp. Vis. Comput.*, Springer, 2020, pp. 207–222.

[16] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 743–761, Apr. 2012.

[17] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1933–1941.

[18] C. Gan, H. Zhao, P. Chen, D. Cox, and A. Torralba, "Self-supervised moving vehicle tracking with stereo sound," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 7053–7062.

[19] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[20] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.

[21] A. Geiger, F. Moosmann, Ö. Car, and B. Schuster, "Automatic camera and range sensor calibration using a single shot," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 3936–3943.

[22] K. Genova et al., "Learning 3D semantic segmentation with only 2D image supervision," in *Proc. Int. Conf. 3D Vis.*, 2021, pp. 361–372 .

[23] J. Geyer et al., "A2D2: Audi autonomous driving dataset," pp. 1–10, 2020, *arXiv:2004.06320*.

[24] B. Graham, M. Engelcke, and L. Van Der Maaten, "3D semantic segmentation with submanifold sparse convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9224–9232.

[25] Y. Guo et al., "NAT: Neural architecture transformer for accurate and compact architectures," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 737–748.

[26] Y. Guo et al., "Towards accurate and compact architectures via neural architecture transformer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 10, pp. 6501–6516, Oct. 2022.

[27] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," in *Proc. Int. Conf. Learn. Representations*, 2016, pp. 1–14.

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[29] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," pp. 1–9, 2015, *arXiv:1503.02531*.

[30] Y. Hou, X. Zhu, Y. Ma, C. C. Loy, and Y. Li, "Point-to-voxel knowledge distillation for LiDAR semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 8479–8488.

[31] Q. Hu et al., "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11108–11117.

[32] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "CCNet: Criss-cross attention for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 603–612.

[33] M. Jaritz, T.-H. Vu, R. D. Charette, E. Wirbel, and P. Pérez, "xMUDA: Cross-modal unsupervised domain adaptation for 3D semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 12605–12614.

[34] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7482–7491.

[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015, pp. 1–15.

[36] L. Kong et al., "Rethinking range view representation for LiDAR segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2023, pp. 228–240.

[37] G. Krispel, M. Opitz, G. Waltner, H. Possegger, and H. Bischof, "FuseSeg: LiDAR point cloud segmentation fusing multi-modal data," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2020, pp. 1874–1883.

[38] X. Lai, Y. Chen, F. Lu, J. Liu, and J. Jia, "Spherical transformer for LiDAR-based 3D recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 17545–17555.

[39] J. Li, H. Dai, and Y. Ding, "Self-distillation for robust LiDAR semantic segmentation in autonomous driving," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2022, pp. 659–676.

[40] G. Lin, C. Shen, A. Van DenHengel, and I. Reid, "Efficient piecewise training of deep structured models for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3194–3203.

[41] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988.

[42] V. E. Liong, T. N. T. Nguyen, S. Widjaja, D. Sharma, and Z. J. Chong, "AMVNet: Assertion-based multi-view fusion network for LiDAR semantic segmentation,", pp. 1–10, 2020, *arXiv: 2012.04934*.

[43] J. Liu et al., "Discrimination-aware network pruning for deep model compression," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 8, pp. 4035–4051, Aug. 2022.
[44] Z. Liu et al., "LPD-Net: 3D point cloud learning for large-scale place recognition and environment analysis," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 2831–2840.
[45] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3431–3440.
[46] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," in *Proc. Int. Conf. Learn. Representations*, 2017, pp. 1–16.
[47] K. E. Madawy, H. Rashed, A. E. Sallab, O. Nasr, H. Kamel, and S. Yogamani, "RGB and LiDAR fusion based 3D semantic segmentation for autonomous driving," in *Proc. IEEE Intell. Transp. Syst. Conf.*, 2019, pp. 7–12.
[48] S. Mehta, M. Rastegari, A. Caspi, L. Shapiro, and H. Hajishirzi, "ESPNet: Efficient spatial pyramid of dilated convolutions for semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 552–568.
[49] G. P. Meyer, J. Charland, D. Hegde, A. Laddha, and C. Vallespi-Gonzalez, "Sensor fusion for joint 3D object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2019, pp. 1–8.
[50] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet : Fast and accurate LiDAR semantic segmentation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 4213–4220.
[51] Y. E. Nesterov, "A method for solving the convex programming problem with convergence rate o (1/k2)," in *Proc. USSR Acad. Sci.*, vol. 269, pp. 543–547, 1983.
[52] S. Niu et al., "Disturbance-immune weight sharing for neural architecture search," *Neural Netw.*, vol. 144, pp. 553–564, 2021.
[53] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8026–8037.
[54] G. Puy, A. Boulch, and R. Marlet, "Using a waffle iron for automotive point cloud semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2023, pp. 3379–3389.
[55] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 652–660.
[56] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet : Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5099–5108.
[57] A. Rényi, "On measures of entropy and information," in *Proc. 4th Berkeley Symp. Math. Statist. Probability, vol. 1: Contributions Theory Statist.*, University of California Press, 1961, pp. 547–562.
[58] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3D point cloud based object maps for household environments," *Robot. Auton. Syst.*, vol. 56, no. 11, pp. 927–941, 2008.
[59] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized LiDAR odometry and mapping on variable terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 4758–4765.
[60] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 568–576.
[61] C. Sitawarin, A. N. Bhagoji, A. Mosenia, M. Chiang, and P. Mittal, "Darts: Deceiving autonomous cars with toxic signs,", pp. 1–18, 2018, *arXiv: 1802.06430*.
[62] H. Tang et al., "Searching efficient 3D architectures with sparse point-voxel convolution," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2020, pp. 685–702.
[63] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger, "Sparsity invariant CNNs," in *Proc. Int. Conf. 3D Vis.*, 2017, pp. 11–20.
[64] S. Vora, A. H. Lang, B. Helou, and O. Beijbom, "PointPainting: Sequential fusion for 3D object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4604–4612.
[65] P. Wang et al., "Understanding convolution for semantic segmentation," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2018, pp. 1451–1460.
[66] 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," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 1887–1893.
[67] 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 *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 4376–4382.
[68] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "SegFormer: Simple and efficient design for semantic segmentation with transformers," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 12077–12090.
[69] C. Xu et al., "SqueezeSegV3: Spatially-adaptive convolution for efficient point-cloud segmentation," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2020, pp. 1–19.
[70] J. Xu, R. Zhang, J. Dou, Y. Zhu, J. Sun, and S. Pu, "RPVNet: A deep and efficient range-point-voxel fusion network for LiDAR point cloud segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 16024–16033.
[71] S. Xu et al., "Generative low-bitwidth data free quantization," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2020, pp. 1–17.
[72] X. Yan et al., "2DPASS: 2D priors assisted semantic segmentation on LiDAR point clouds," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2022, pp. 677–695.
[73] Y. Yuan and J. Wang, "OCNet: Object context network for scene parsing,", pp. 1–22, 2018, *arXiv: 1809.00916*.
[74] F. Zhang, J. Fang, B. Wah, and P. Torr, "Deep fusionnet for point cloud semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 644–663.
[75] W. Zhang, Z. Wang, and C. C. Loy, "Exploring data augmentation for multi-modality 3D object detection,", pp. 1–12, 2020, *arXiv: 2012.12741*.
[76] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu, "Deep mutual learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4320–4328.
[77] Y. Zhang et al., "PolarNet: An improved grid representation for online LiDAR point clouds semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9601–9610.
[78] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2881–2890.
[79] L. Zhao, S. Xu, L. Liu, D. Ming, and W. Tao, "SVASeg: Sparse voxel-based attention for 3D LiDAR point cloud semantic segmentation," *Remote Sens.*, vol. 14, no. 18, 2022, Art. no. 4471.
[80] X. Zhu et al., "Cylindrical and asymmetrical 3D convolution networks for LiDAR-based perception," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 10, pp. 6807–6822, Oct. 2022.
[81] X. Zhu et al., "Cylindrical and asymmetrical 3D convolution networks for LiDAR segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 9939–9948.
[82] Z. Zhuang, R. Li, K. Jia, Q. Wang, Y. Li, and M. Tan, "Perception-aware multi-sensor fusion for 3D LiDAR semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 16280–16290.

Mingkui Tan (Member, IEEE) received the bachelor's degree in environmental science and engineering and the master's degree in control science and engineering from Hunan University, Changsha, China, in 2006 and 2009, respectively, and the PhD degree in computer science from Nanyang Technological University, Singapore, in 2014. He is currently a professor with the School of Software Engineering, South China University of Technology, Guangzhou, China. From 2014 to 2016, he worked as a senior research associate on computer vision with the School of Computer Science, University of Adelaide, Australia. His research interests include machine learning, sparse analysis, deep learning and large-scale optimization.

Zhuangwei Zhuang (Graduate Student Member, IEEE) received the bachelor's degree in automation and engineering and the master's degree in software engineering from the South China University of Technology, Guangzhou, China, in 2016 and 2018, respectively. He is currently working toward the PhD degree with the School of Software Engineering, South China University of Technology, and is currently working as an intern with RoboSense, Shenzhen, China. His research interests include model compression and 3D scene understanding for autonomous driving.

Sitao Chen received the bachelor's degree from the School of Mechanical and Automotive, South China University of Technology in Guangzhou, China, in 2023. He is currently working toward the master's degree with the School of Software Engineering, South China University of Technology. His research interests include 3D scene understanding for autonomous driving.

Rong Li received the bachelor's and master's degrees from the School of Software Engineering, South China University of Technology, China, in 2019 and 2022, respectively. She is currently an associate researcher with the Hong Kong University of Science and Technology (Guang Zhou). Her research interests include LiDAR perception.

Kui Jia (Member, IEEE) received the bachelor's degree in marine engineering from Northwestern Polytechnic University, Xi'an, China, in 2001, the master's degree in electrical and computer engineering from the National University of Singapore, in 2003, and the PhD degree in computer science from the Queen Mary University of London, U.K., in 2007. He is currently a professor with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou, China. His recent research focuses on theoretical deep learning and its applications in vision and robotic problems, including deep learning of 3D data and deep transfer learning.

Qicheng Wang received the bachelor's degree in electronic engineering from Tsinghua University, Beijing, in 2007. He is currently working toward the PhD degree with the Department of Mathematics, Hong Kong University of Science and Technology. His research interest lies in neural network and computer vision for autonomous driving.

Yuanqing Li (Fellow, IEEE) received the bachelor's degree in applied mathematics from Wuhan University, Wuhan, China, in 1988, the master's degree in applied mathematics from South China Normal University, Guangzhou, China, in 1994, and the PhD degree in control theory and applications from the South China University of Technology, Guangzhou, in 1997. He is currently a professor with the School of Automation and Engineering, South China University of Technology, Guangzhou, China. His research interests include blind signal processing, sparse representation, machine learning, brain-computer interface, and EEG and functional magnetic resonance imaging data analysis.