

FRNet: Frustum-Range Networks for Scalable LiDAR Segmentation

Xiang Xu, Lingdong Kong, *Student Member, IEEE*, Hui Shuai, and Qingshan Liu, *Senior Member, IEEE*

Abstract—LiDAR segmentation has become a crucial component of advanced autonomous driving systems. Recent range-view LiDAR segmentation approaches show promise for real-time processing. However, they inevitably suffer from corrupted contextual information and rely heavily on post-processing techniques for prediction refinement. In this work, we propose FRNet, a simple yet powerful method aimed at restoring the contextual information of range image pixels using corresponding frustum LiDAR points. First, a frustum feature encoder module is used to extract per-point features within the frustum region, which preserves scene consistency and is critical for point-level predictions. Next, a frustum-point fusion module is introduced to update per-point features hierarchically, enabling each point to extract more surrounding information through the frustum features. Finally, a head fusion module is used to fuse features at different levels for final semantic predictions. Extensive experiments conducted on four popular LiDAR segmentation benchmarks under various task setups demonstrate the superiority of FRNet. Notably, FRNet achieves 73.3% and 82.5% mIoU scores on the testing sets of SemanticKITTI and nuScenes. While achieving competitive performance, FRNet operates 5 times faster than state-of-the-art approaches. Such high efficiency opens up new possibilities for more scalable LiDAR segmentation. The code has been made publicly available at <https://github.com/Xiangxu-0103/FRNet>.

Index Terms—LiDAR Semantic Segmentation; Autonomous Driving; Real-Time Processing; Frustum-Range Representation

I. INTRODUCTION

LiDAR segmentation, a crucial and indispensable component in modern autonomous driving, robotics, and other safety-critical applications, has witnessed substantial advancements recently [1]–[4]. The principal challenge lies in devising a *scalable* LiDAR segmentation system that strikes a delicate balance between efficiency and accuracy, especially in resource-constrained operational scenarios [5], [6]. Here, “scalable” refers to the system’s ability to maintain high performance while managing increased data loads without compromising real-time processing capabilities.

Existing LiDAR segmentation methods adopt various data representation perspectives, each with inherent trade-offs in terms of accuracy and computational efficiency. Point-based methods [7]–[12] manipulate the original point cloud to preserve spatial granularity but entail computationally expensive

X. Xu is with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China.

L. Kong is with the School of Computing, Department of Computer Science, National University of Singapore, Singapore.

H. Shuai and Q. Liu are with the School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing, China.

The corresponding author is Qingshan Liu: qsliu@njupt.edu.cn.

This work was supported in part by the Natural Science Foundation of China under Grants U21B2044 and U24B20155, and in part by the Jiangsu Province Science and Technology Project under Grants BA2022026 and BK20243051.

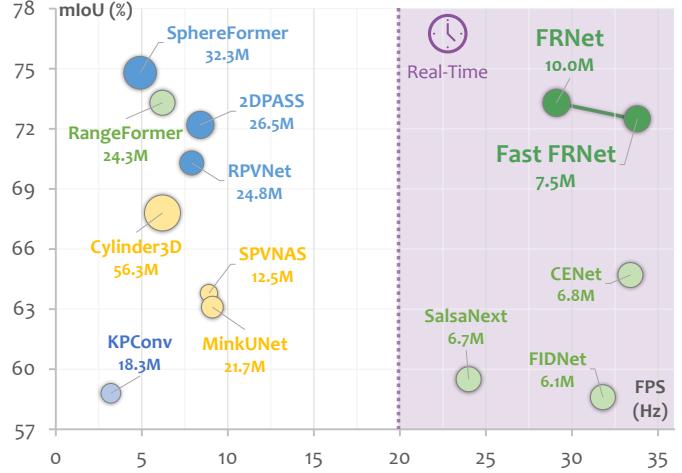


Fig. 1: **A study on the scalability** of state-of-the-art LiDAR segmentation models on the SemanticKITTI [32] leaderboard. The size of the circular representation corresponds to the number of model parameters. FRNet achieves competitive performance with current state-of-the-art models while still maintaining satisfactory efficiency for real-time processing.

neighborhood searches to construct local structures. This demand results in significant computational overhead and limits their capability to deal with large-scale point clouds [10], [13], [14]. Sparse-voxel-based methods [15]–[19] transform scattered LiDAR points into regular voxel grids and leverage popular sparse convolutions to extract features. However, this process also demands heavy computations, especially for applications that require high voxel resolutions [15], [16], [20]. Multi-view methods [21]–[24] extract features from various representations to enhance prediction accuracy [25]–[27]. While this strategy can improve performance, it aggregates the computation required for all representations. Albeit these representation methods achieved satisfactory performance, their computation overhead hinders them from real-time online applications in real life [24], [28]–[31].

Recently, pseudo-image-based methods [33]–[40] have appeared as a simple yet efficient intermediate representation for LiDAR processing, such as bird’s-eye-view [37]–[39] and range-view [33]–[36], which are more computationally tractable. These representations enable the direct application of popular 2D segmentation approaches [41]–[43] to the pseudo-images, offering a promising solution for real-time LiDAR segmentation. However, the 3D-to-2D projection inevitably introduces corrupted neighborhood contextual information, which poses limitations on the further development of pseudo-image approaches [44]. The detailed analysis of this problem

can be further summarized into two aspects as follows.

Firstly, a fundamental concern with pseudo-image methods is the inadvertent exclusion of certain 3D points during the conversion to 2D projections [45]. This leads to an incomplete representation where only the points that are projected are involved in subsequent convolutional operations. Such a scenario disregards the contextual information of the unprojected points, causing the pseudo-images to lose crucial high-fidelity 3D geometric context [46]. Furthermore, the omitted points, which still necessitate 3D label predictions, lack associated feature information, thereby necessitating post-processing to infer these labels. Unfortunately, current post-processing methods struggle to incorporate local features of these omitted points effectively, capping the potential advancements in pseudo-image approaches.

Secondly, the challenge with pseudo-images is exacerbated by the significant presence of empty pixels due to the inherent sparsity of LiDAR data. For instance, in the nuScenes dataset, nearly 60% of range image pixels are found to be void [36], [47]. This vacancy not only skews the scene representation but also compromises the quality of semantic segmentation. As the network processes deeper layers, these sparse pseudo-images can become riddled with noise, obstructing the extraction of meaningful patterns. In essence, although pseudo-image methods excel in terms of real-time processing capabilities, their reduced dimensional perspective can overlook crucial 3D details, leading to degraded scalability.

Observing the above issues, in this work, we propose a Frustum-Range Network (FRNet) for scalable LiDAR segmentation, which incorporates points into the range image, achieving a superior balance between efficiency and accuracy. FRNet consists of three main components. Firstly, a Frustum Feature Encoder (FFE) is utilized to group all points with the same frustum region into corresponding range-view pixels in relation to the range image using multiple multi-layer perceptrons (MLPs). This allows for the preservation of all points and the prediction of semantic labels for these points in an end-to-end manner. Subsequently, the point features are pooled to represent the frustum region and formatted into a 2D representation, which is then subjected to traditional convolutions. Secondly, a Frustum-Point (FP) fusion module is employed to efficiently update the hierarchical features of each point during each convolutional stage. This module includes frustum-to-point fusion to update per-point features and point-to-frustum fusion to enhance frustum features. As a result, all points extract larger local features based on the frustum region. Finally, a Fusion Head (FH) module is designed to leverage features from different levels to generate individual features for each point, facilitating end-to-end prediction without the need for post-processing techniques. As shown in Fig. 1, the proposed FRNet achieves great improvement among range-view methods while still maintaining high efficiency.

Data augmentation plays an important role in training robust and generalizable models. Inspired by the success of mixing strategies [48]–[50], we introduce a fine-grained mixing method dubbed FrustumMix, which operates based on frustum units. Specifically, we randomly divide point cloud scenes into several frustum regions along inclination and azimuth

directions and swap the corresponding regions from different scenes, with the neighborhood of each frustum region coming from the other scene. FrustumMix provides a straightforward and lightweight method to adapt FRNet by focusing on the frustum region. Furthermore, to address the issue of empty pixels in the 2D representation, we propose RangeInterpolation to reconstruct the semantic surface based on surrounding point information. It first transforms the LiDAR point cloud into a range image, as done in previous works [33]–[36]. For empty pixels in the range image, we then aggregate surrounding range information within a pre-defined window using an average pooling operation to generate a new point. RangeInterpolation is able to assist in creating a more compact 2D representation with fewer empty pixels, enabling convolutions to learn more semantically coherent information.

The key contributions of this work are summarized as:

- We introduce a novel Frustum-Range representation for scalable LiDAR segmentation. In our framework, the point-level geometric information is integrated into the pseudo-image representation, which not only retains the complete geometric structure of the point cloud, but also takes advantage of the efficiency of the 2D network.
- We propose two 3D data augmentation techniques to assist in training robust and generalizable models. FrustumMix generates more complex scenes by mixing two different scans, while RangeInterpolation reconstructs the semantic surfaces based on surrounding range information. These techniques help generate a compact 2D representation and learn more context-aware features.
- Extensive experiments across four prevailing LiDAR segmentation benchmarks demonstrate our superiority. FRNet achieves 73.3% and 82.5% mIoU scores, respectively, on SemanticKITTI and nuScenes, while still maintaining promising scalability for real-time LiDAR segmentation.

II. RELATED WORK

A. Scalable LiDAR Segmentation

Although point-view [7], [8], [10]–[12], sparse-voxel-view [15]–[18], and multi-view [21]–[24], [51], [52] methods have achieved significant success in LiDAR segmentation, they often suffer from high computational costs and are unable to achieve real-time processing. Recently, various methods have been proposed to project LiDAR points into range images along inclination and azimuth directions to balance the efficiency and accuracy of segmentation [53]. SqueezeSeg [33] and SqueezeSegV2 [54] use the lightweight model SqueezeNet [55] to retain information. SqueezeSegV3 [56] introduces a Spatially-Adaptive Convolution module to adjust kernel weights based on the locations of range images. RangeNet++ [34] integrates DarkNet into SqueezeSeg [33] and proposes an efficient KNN post-processing for segmentation. RangeViT [35] and RangeFormer [36] introduce transformer blocks to extract both local and global information from the range image. However, these methods only involve projected points and fail to extract contextual information for discarded points. To address this problem, WaffleIron [11]

proposes to mix point and image features with residual connections. Although achieving promising performance, it requires downsampling points based on voxel grid and space neighborhood search for point feature embedding, which hinders it from real-time processing and end-to-end predictions over the entire point cloud. In this work, we propose an efficient integration of point and image features that preserves both the 3D geometric information from raw point features and the efficiency of 2D convolutions, eliminating the need for pre- or post-processing techniques and yielding superior scalability for real-time LiDAR segmentation.

B. LiDAR Data Augmentation

Recent works have explored various data augmentation techniques for point clouds [57], [58], which have shown promise in improving performance for indoor scene understanding but exhibit limited generalization to outdoor scenes. Mix3D [50] creates new training samples by combining two out-of-context scenes, which incurs higher memory costs. PolarMix [49] proposes to mix two LiDAR scenes based on azimuth angles and crop instances from one scene to another, while LaserMix [48] suggests exchanging labeled scans with unlabeled scans along the inclination direction for efficient semi-supervised learning. RangeFormer [36] presents a lightweight augmentation method that operates only on projected points in range images, disregarding occluded points. In this work, we introduce augmentation techniques aimed at training a robust and scalable model for LiDAR segmentation, including FrustumMix and RangeInterpolation.

C. LiDAR Post-Processing

Prior works have proposed various post-processing techniques to reconstruct the semantic labels of occluded points based on 2D results. SqueezeSeg [33] employs the traditional conditional random field (CRF) as a recurrent neural network (RNN). RangeNet++ [34] introduces a fast K-Nearest-Neighbor (KNN) search to aggregate final results across the entire point cloud. FIDNet [59] uses a Nearest Label Assignment (NLA) method for label smoothing. RangeFormer [36] divides full point clouds into sub-clouds and infers each subset in a supervised manner. However, these methods heavily rely on 2D results and fail to extract point features over the entire point cloud in a supervised manner. In contrast, KPRNet [60] and RangeViT [35] incorporate a KPConv [7] block to extract point features in an end-to-end manner. Nevertheless, this approach requires substantial computational resources to search neighborhoods in 3D space, which hinders them from real-time processing. In this work, we partition points into frustum regions and extract local features of points in a 2D manner, avoiding time-consuming neighbor searches. Based on the learned point features, FRNet predicts results over the entire point cloud in an end-to-end manner with high efficiency.

III. FRNET: A SCALABLE LiDAR SEGMENTOR

In this section, we first conduct a pilot study on the most popular range-view LiDAR segmentation methods, wherein

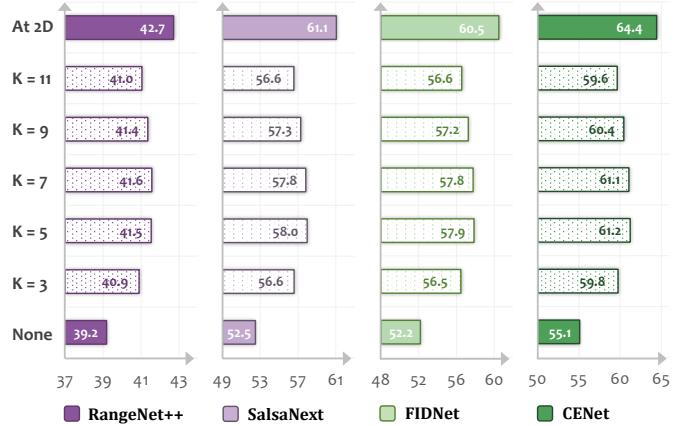


Fig. 2: **Pilot study** on the performance degradation of post-processing in existing range-view methods [34], [39], [59], [61] on the val set of SemanticKITTI [32]. We choose various K values as hyperparameters in KNN post-processing. Compared to their performance at 2D (*i.e.*, the range image), a severe drop in performance occurs with different K values.

we observe a significant impact of KNN post-processing on performance (Sec. III-A). We then elaborate on the technical details of frustum-range architecture (Sec. III-B) and frustum-range operators (Sec. III-C). The overall pipeline of the proposed FRNet framework is depicted in Fig. 3.

A. Pilot Study

LiDAR segmentation aims to assign semantic labels to each point in the point cloud [62]. A common approach is to operate directly on the LiDAR points without any pre-processing, following the paradigm established by PointNet++ [63]. However, point-based operations often require extensive computations and can yield sub-optimal performance due to the sparsity of outdoor scenes. To address this problem, many recent works have proposed projecting LiDAR points onto a regular intermediate representation, known as range images, based on spherical coordinates to enhance efficiency. Specifically, given a point cloud $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^N$ with N points, where $\mathbf{p}_i \in \mathbb{R}^{3+L}$ includes the coordinate (x_i, y_i, z_i) and additional L -dimensional features (intensity, elongation, *etc.*), we map the point to its corresponding 2D position (u_i, v_i) in the range image via the following transformation:

$$\begin{pmatrix} u_i \\ v_i \end{pmatrix} = \begin{pmatrix} \frac{1}{2}[1 - \arctan(y_i, x_i)\pi^{-1}]W \\ [1 - (\arcsin(z_i d_i^{-1}) + \phi_{\text{down}})\phi^{-1}]H \end{pmatrix}, \quad (1)$$

where $d_i = \sqrt{x_i^2 + y_i^2 + z_i^2}$ is the depth of the point; $\phi = |\phi_{\text{up}}| + |\phi_{\text{down}}|$ represents the vertical field-of-view (FOV) of the sensor, with ϕ_{up} and ϕ_{down} being the inclination angles in the upward and downward directions, respectively; H and W are the height and width of the range image. However, such projection often results in the loss of occluded points and requires post-processing techniques to reconstruct the entire semantic information [36].

We conducted a pilot study on various range-view methods, including RangeNet++ [34], SalsaNext [64], FIDNet [59], and CENet [61]. We compared their performances on the original

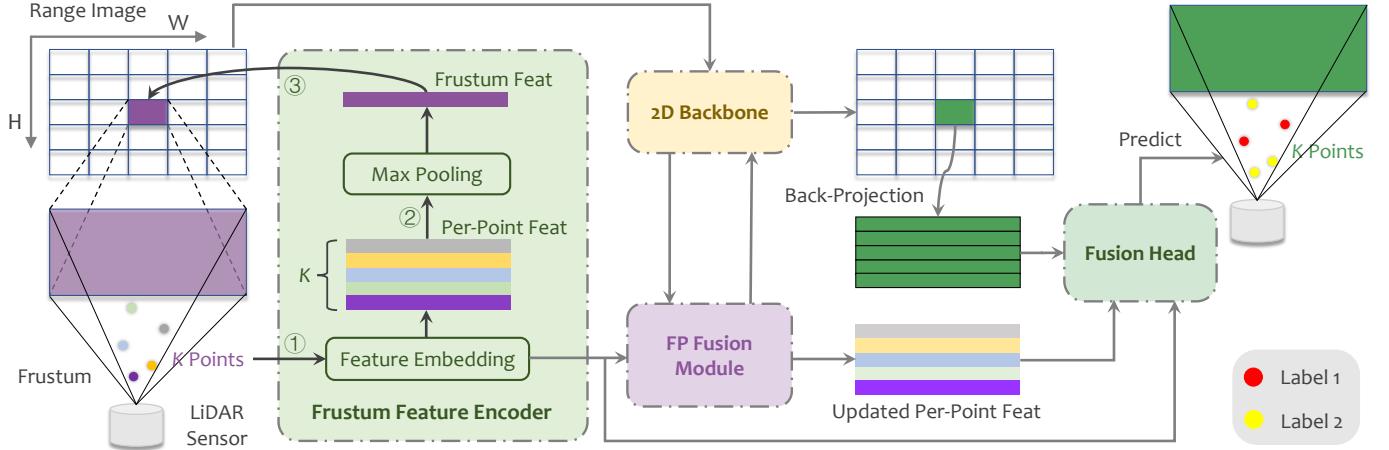


Fig. 3: Architecture overview. The proposed FRNet comprises three main components: 1) *Frustum Feature Encoder* is used to embed per-point features within the frustum region. 2) *Frustum-Point (FP) Fusion Module* updates per-point features hierarchically at each stage of the 2D backbone. 3) *Fusion Head* fuses different levels of features to predict final results.

points (*i.e.*, after the 2D-to-3D projection) using the widely-used KNN post-processing with different nearest neighbor parameters, as well as their performance on the 2D range images. As illustrated in Fig. 2, the post-processing procedures are often unsupervised, and the choice of hyperparameters can significantly impact the final performance. To address this issue, we propose a scalable Frustum-Range Network (FRNet) capable of directly predicting semantic labels for each point while maintaining high efficiency.

B. Frustum-Range Architecture

Problem Definition. Given a point cloud \mathcal{P} acquired by the LiDAR sensor, the objective of FRNet is to employ a feed-forward network \mathcal{G} to predict semantic labels $\hat{\mathcal{Y}}$ for each individual point as follows:

$$\hat{\mathcal{Y}} = \mathcal{G}(\mathcal{P}, \theta), \quad (2)$$

where θ represents the learnable parameters within the network. As depicted in Fig. 3, FRNet consists of three key components: 1) a frustum feature encoder responsible for per-point feature extraction; 2) a frustum-point fusion module that efficiently fuses hierarchical point features and frustum features; 3) a fusion head module that integrates point features at different levels for accurate semantic prediction.

Operator Representation. Let $\text{Flat}(\cdot) : \mathbb{R}^{N \times C} \rightarrow \mathbb{R}^{H \times W \times C}$ denotes the function that projects point features onto the frustum image plane with resolution (H, W) . Within each frustum region, we apply a max-pooling function to obtain the frustum features. Let $\text{MLP}(\cdot) : \mathbb{R}^{N \times C} \rightarrow \mathbb{R}^{N \times D}$ represents the MLPs that take C -dimensional point features as input and output D -dimensional point features. Let $\text{Inflat}(\cdot) : \mathbb{R}^{H \times W \times C} \rightarrow \mathbb{R}^{N \times C}$ be the operation that back-projects frustum features to point features. Notably, points falling into the same frustum region share the same features under this operation.

Frustum Feature Encoder. Accurate prediction of semantic labels for each point requires the extraction of individual features. Before converting the point cloud into a 2D frustum representation, the frustum feature encoder plays a pivotal role in extracting per-point features through a series of MLPs,

which is crucial for accurate label prediction. The frustum feature encoder follows three steps. ① We divide the point cloud into H and W frustum regions along the vertical and horizontal directions, respectively. According to Eq. (1), for each point p_i in the point cloud \mathcal{P} , we calculate its frustum position (u_i, v_i) . Points sharing the same (u, v) coordinates are grouped into the same frustum region. This results in a set of M frustum regions, defined as $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_M\}$, where $M = H \times W$ and $\mathcal{P}_i \in \mathbb{R}^{N_i \times (3+L)}$ consists of N_i points within the i -th frustum region. ② To explicitly embed the frustum structure, we leverage an average pooling function within each frustum region to obtain the cluster points $\tilde{\mathcal{P}} = \{\tilde{\mathcal{P}}_1, \tilde{\mathcal{P}}_2, \dots, \tilde{\mathcal{P}}_M\}$, where $\tilde{\mathcal{P}}_i \in \mathbb{R}^{1 \times (3+L)}$. Subsequently, we construct a frustum graph between the point cloud \mathcal{P} and the cluster points $\tilde{\mathcal{P}}$ within each frustum region. The per-point features are embedded as follows:

$$\mathcal{F}_p^0 = \text{MLP}([\mathcal{P}; \mathcal{P} - \tilde{\mathcal{P}}]), \quad (3)$$

where $[\cdot ; \cdot]$ denotes feature concatenation. A max-pooling function is then applied within the frustum region to yield the initialized frustum feature. ③ The frustum feature is transformed into a 2D frustum representation based on its frustum position (u, v) : $\mathcal{F}_f^0 = \text{Flat}(\mathcal{F}_p^0)$. Both \mathcal{F}_p^0 and \mathcal{F}_f^0 serve as inputs to our backbone for the efficient hierarchical updating of point and frustum features.

Frustum-Point Fusion Module. Building on prior works [36], [59], [61], we utilize a 2D backbone composed of multiple convolution blocks to efficiently extract hierarchical frustum features. Each stage block is accompanied by a frustum-point fusion module, facilitating the incremental update of point and frustum features. As illustrated in Fig. 4, the frustum-point fusion module consists of two essential components: 1) a frustum-to-point fusion that enables the hierarchical update of per-point features, and 2) a point-to-frustum fusion that fuses individual features of each point within its corresponding frustum region. The i -th frustum-point fusion module takes \mathcal{F}_p^{i-1} and \mathcal{F}_f^{i-1} as inputs, where \mathcal{F}_f^{i-1} is the output of the 2D convolution block that takes \mathcal{F}_p^{i-1} as input.

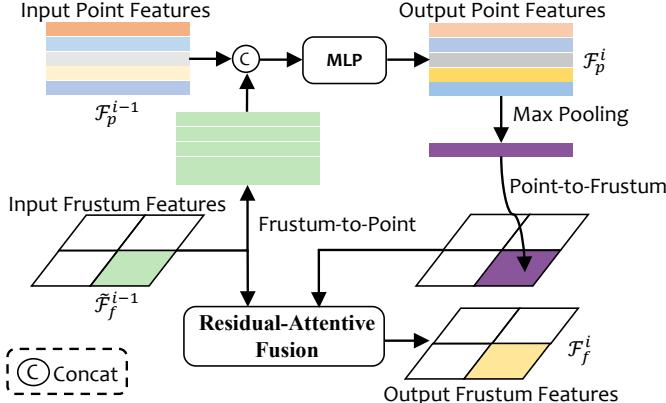


Fig. 4: **Frustum-point fusion module** comprises two steps: 1) A Frustum-to-Point fusion to update per-point features. 2) A Point-to-Frustum fusion to update frustum features.

To efficiently extract context information for the points, we employ a frustum-to-point fusion block to integrate local features from the frustum region into the points. First, $\tilde{\mathcal{F}}_f^{i-1}$ is back-projected to the point-level features according to the projection index. Subsequently, the back-projected features are concatenated with \mathcal{F}_p^{i-1} and passed through an MLP to update the per-point features, avoiding high computational costs from neighborhood searches:

$$\mathcal{F}_p^i = \text{MLP}([\text{Inflat}(\tilde{\mathcal{F}}_f^{i-1}); \mathcal{F}_p^{i-1}]) . \quad (4)$$

As discussed above, the 2D representation is non-compact due to the sparsity of the point cloud. The convolution block inevitably introduces noisy information for empty frustums, leading to compact feature representations. To preserve the sparsity attributes of the 2D representation, we design a point-to-frustum fusion module to fuse the sparse and dense frustum features. Specifically, the updated point features \mathcal{F}_p^i are first pooled into the corresponding frustum region to yield a non-compact 2D representation. Subsequently, the non-compact frustum feature is concatenated with the compact frustum feature $\tilde{\mathcal{F}}_f^{i-1}$ and passed through a simple convolutional layer to reduce the number of channels:

$$\mathcal{F}_{\text{fuse}}^i = \text{Conv}([\text{Flat}(\mathcal{F}_p^i); \tilde{\mathcal{F}}_f^{i-1}]) , \quad (5)$$

where $\text{Conv}(\cdot)$ represents a convolution layer with batch normalization and an activation function. Finally, the fused features are utilized to further enhance the frustum features in a residual-attentive manner:

$$\mathcal{F}_f^i = \tilde{\mathcal{F}}_f^{i-1} + \sigma(h(\mathcal{F}_{\text{fuse}}^i, \theta)) \odot \mathcal{F}_{\text{fuse}}^i , \quad (6)$$

where σ represents the sigmoid function, $h(\cdot)$ is a linear function with trainable parameters θ , and \odot indicates element-wise multiplication.

Fusion Head Module. FRNet aims to assign a category to each point in the point cloud, which requires point-level features for end-to-end label prediction. Although the output point features from the last layer of the frustum-point fusion module can yield promising performance, we observe this is not the optimal solution. Features from different levels have different context-aware information. Lower-level features are geometric while higher-level features are semantically rich. Leveraging

these different level features can enhance performance. Drawing inspiration from FIDNet [59] and CENet [61], we first combine all point and frustum features from each stage of the backbone and fuse them to reduce channel dimension:

$$\mathcal{F}_p^{\text{out}} = \text{MLP}([\mathcal{F}_p^1; \dots; \mathcal{F}_p^V]) , \quad (7)$$

$$\mathcal{F}_f^{\text{out}} = \text{Conv}([\mathcal{F}_f^1; \dots; \mathcal{F}_f^V]) , \quad (8)$$

where V is the number of stages in the backbone. Fusing backbone features with different receptive fields helps obtain more context-aware and semantic information. Additionally, point features $\mathcal{F}_p^{\text{out}}$ contains local information from neighborhood frustum regions, while frustum features $\mathcal{F}_f^{\text{out}}$ contains global information within the corresponding frustum region. Meanwhile, \mathcal{F}_p^0 from the frustum feature encoder module constructs graph structures within the frustum region and between neighborhood frustum regions. Progressively fusing these features helps extract features from local to global, and from geometric to semantic, resulting in more accurate predictions. To fuse $\mathcal{F}_f^{\text{out}}$ and $\mathcal{F}_p^{\text{out}}$, we first back-project the frustum features to point-level and use an MLP layer to ensure the same channels as $\mathcal{F}_p^{\text{out}}$. Then, an addition operation is applied to fuse the two features. The same operation is utilized to fuse the new features with \mathcal{F}_p^0 . The overall operation can be formulated as:

$$\mathcal{F}_{\text{logit}} = \text{MLP}(\text{MLP}(\text{Inflat}(\mathcal{F}_f^{\text{out}})) + \mathcal{F}_p^{\text{out}}) + \mathcal{F}_p^0 . \quad (9)$$

Here, $\mathcal{F}_{\text{logit}}$ is used to generate the final semantic scores with a linear head for the point over the entire point cloud.

Optimization. In the conventional setup, a point-level loss function is commonly employed to supervise the final semantic scores. However, in this work, we propose optimizing frustum features with additional auxiliary heads, utilizing the frustum-level loss function. This approach requires generating pseudo frustum labels. Typically, a frustum region encompasses multiple points, which may not all belong to the same category. To address this, we adopt a statistical approach inspired by sparse-voxel methods [16]. We generate a pseudo label for each frustum by first counting the frequency of each category existing in the given frustum region, and then choosing the category with the highest frequency as the pseudo label for the frustum. Finally, the frustum-level features are supervised using the generated pseudo labels. The overall loss function can be formulated as follows:

$$\mathcal{L} = \mathcal{L}_p + \lambda \mathcal{L}_f , \quad (10)$$

where \mathcal{L}_p and \mathcal{L}_f represent the point-level and frustum-level loss functions, respectively, and λ is a hyperparameter that controls the weight of frustum supervision. \mathcal{L}_p is calculated by cross-entropy loss, while \mathcal{L}_f consists of cross-entropy loss, Lovász-Softmax loss [65], and boundary loss [66].

C. Frustum-Range Operators

Data augmentation enables the model to learn more robust representations. Common augmentations in LiDAR segmentation typically operate from a global perspective, including rotation, flipping, scaling, jittering, etc. In this work, we

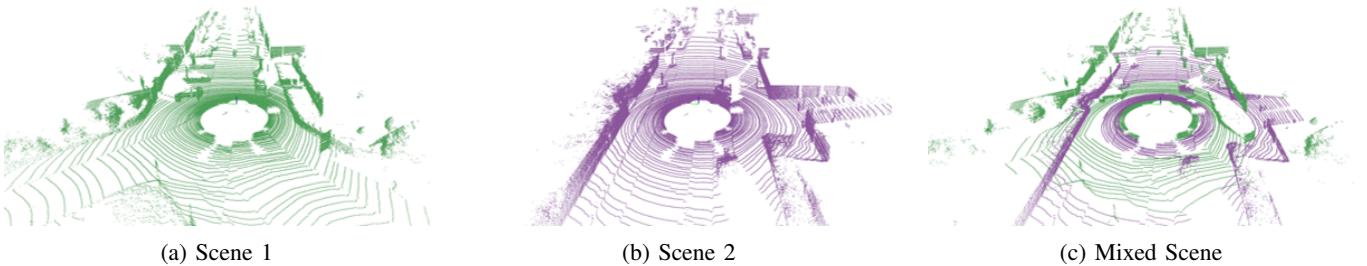


Fig. 5: **FrustumMix illustration.** (a) and (b) show the original two LiDAR scenes. (c) presents the mixed scenes generated by the FrustumMix strategy, where scene 1 is colored green and scene 2 is colored purple.

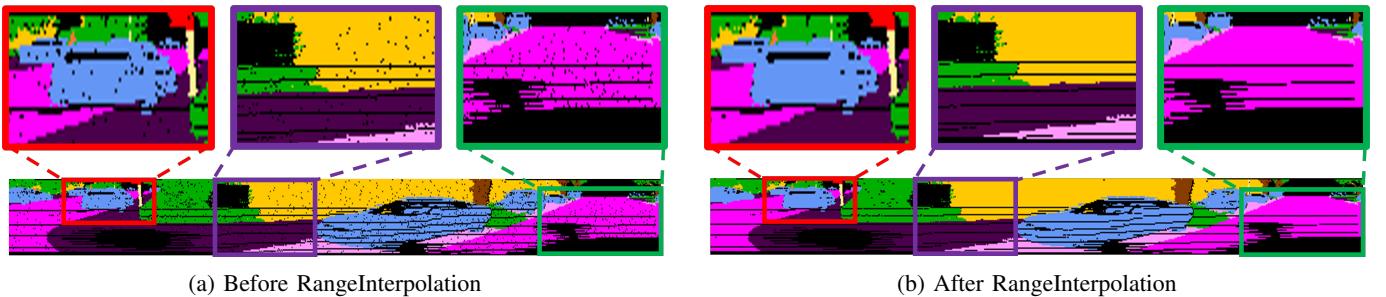


Fig. 6: **RangeInterpolation** illustration. (a) shows the raw range image with numerous empty pixels. (b) illustrates the range image after applying RangeInterpolation, resulting in a smoother and more compact representation.

introduce two novel augmentations based on our proposed frustum-range view representation.

FrustumMix. Previous mixing strategies [48], [49] perform point-level swapping between two LiDAR scans. While these methods achieve notable improvements in various LiDAR representations, they are not well aligned with the frustum representation, potentially causing two distinct scenes to mix within the same frustum region. This misalignment disrupts both the semantic coherence and geometric structure within the frustum region. To this end, we propose FrustumMix, a fine-grained extension of LaserMix [48] that aligns better with the frustum representation. Specifically, given two LiDAR point clouds \mathcal{P}_1 and \mathcal{P}_2 , we randomly split the point clouds into M non-overlapping frustum regions along the inclination or azimuth direction, denoted as $\mathcal{P}_1 = \{\mathcal{A}_1^1, \dots, \mathcal{A}_1^M\}$ and $\mathcal{P}_2 = \{\mathcal{A}_2^1, \dots, \mathcal{A}_2^M\}$. We then generate a new LiDAR scan $\hat{\mathcal{P}}$ by alternately swapping frustum regions from \mathcal{P}_1 and \mathcal{P}_2 , as follows: $\hat{\mathcal{P}} = \mathcal{A}_1^1 \cup \mathcal{A}_2^2 \cup \mathcal{A}_1^3 \cup \mathcal{A}_2^4 \cup \dots$. This approach ensures that mixing occurs at the level of frustum units, preserving semantic coherence and structural details entirely within each frustum region, thereby maintaining alignment with the frustum representation. Fig. 5 provides an illustration of FrustumMix. By enhancing context-awareness between frustums while preserving intra-frustum structural invariance, FrustumMix achieves improved segmentation performance and better representation learning.

RangeInterpolation. Previous range-view methods [36], [59], [61] often encounter a large number of empty pixels due to the sparsity of LiDAR points, resulting in noisy information. To mitigate this problem, we introduce RangeInterpolation, a technique that utilizes the range image to reconstruct surfaces in the LiDAR scan. Specifically, the point cloud \mathcal{P} is first projected onto the range image $\mathcal{R} \in \mathbb{R}^{H \times W \times (3+L)}$. For empty

pixel positions (u, v) in the range image \mathcal{R} , we establish a window of size $m \times n$ centered at (u, v) , where m and n are odd. We aim to interpolate the point falling into (u, v) using the surrounding range information within the pre-defined window. Considering that not all pixels in the window contain valid values, we aggregate information from non-empty pixels within the window for point interpolation using an average operation. During the training procedure, the semantic label of the interpolated point is determined via threshold-guided voting within the window. Specifically, we identify the most frequent category within the window and its frequency. If the frequency is below a threshold, we classify the interpolated point as located at the boundary and assign the label as ignored, which will not contribute to loss calculation. Otherwise, the label is determined by the category with the highest frequency. As shown in Fig. 6, RangeInterpolation produces coherent semantic information in the range image, resulting in a more compact representation.

IV. EXPERIMENTS

In this section, we demonstrate the scalability and robustness of the proposed method. We first introduce the datasets, metrics, and detailed implementations we used. Subsequently, we show that FRNet achieves an optimal trade-off between efficiency and accuracy when compared to state-of-the-art methods across various task setups. Finally, we conduct a series of ablation studies to analyze each component in FRNet.

A. Experimental Settings

Datasets. We conduct comprehensive experiments on four popular LiDAR segmentation datasets. *SemanticKITTI* [32] is a large-scale outdoor dataset that comprises 22 sequences collected from various scenes in Karlsruhe, Germany, using a

TABLE I: **Supervised LiDAR segmentation** results on the official SemanticKITTI [32], nuScenes [47], ScribbleKITTI [67], and SemanticPOSS [68] benchmarks. \dagger denotes models that use pretrained weights. **FPS** denotes frame rate per second (s). All **mIoU** and **mAcc** scores are given in percentage (%). The **best** and **second best** scores for models from each representation group are highlighted in **bold** and underline.

Method	Venue	Representation	Param	SemanticKITTI			nuScenes		ScribbleKITTI		SemanticPOSS	
				FPS \uparrow	Val \uparrow	Test \uparrow	Val \uparrow	Test \uparrow	mIoU \uparrow	mAcc \uparrow	mIoU \uparrow	mAcc \uparrow
MinkUNet [17]	CVPR'19	Sparse Voxel ●	21.7 M	<u>9.1</u>	62.8	63.7	75.8	-	55.0	-	53.1	68.3
SPVNAS [15]	ECCV'20	Sparse Voxel ●	21.8 M	8.9	<u>62.5</u>	66.4	74.4	-	<u>56.9</u>	-	48.4	61.5
Cylinder3D [16]	CVPR'21	Sparse Voxel ●	55.9 M	6.2	<u>65.9</u>	<u>67.8</u>	76.1	77.9	57.0	-	52.9	64.9
PVKD [69]	CVPR'22	Sparse Voxel ●	14.1 M	13.2	66.4	71.2	<u>76.0</u>	-	-	-	-	-
KPConv [7]	ICCV'19	Raw Points ●	18.3 M	3.2	61.3	58.8	-	-	-	-	-	-
RandLA-Net [8]	CVPR'20	Raw Points ●	1.2 M	1.9	<u>57.1</u>	53.9	-	-	-	-	-	-
PTv2 [70]	NeurIPS'22	Raw Points ●	12.8 M	4.7	<u>70.3</u>	<u>72.6</u>	<u>80.2</u>	<u>82.6</u>	-	-	-	-
WaffleIron [11]	ICCV'23	Raw Points ●	6.8 M	<u>7.4</u>	68.0	70.8	79.1	-	-	-	-	-
PTv3 [12]	CVPR'24	Raw Points ●	46.2 M	20.2	70.8	74.2	80.4	82.7	-	-	-	-
RPVNet [22]	ICCV'21	Multi-View ●	24.8 M	<u>7.9</u>	65.5	70.3	77.6	-	-	-	-	-
2DPASS [71]	ECCV'22	Multi-View ●	26.5 M	8.4	69.3	72.2	79.4	80.8	-	-	-	-
SphereFormer [72]	CVPR'23	Multi-View ●	32.3 M	4.9	67.8	74.8	<u>78.4</u>	81.9	-	-	-	-
UniSeg [24]	ICCV'23	Multi-View ●	147.6 M	6.9	<u>71.3</u>	<u>75.2</u>	-	<u>83.5</u>	-	-	-	-
TASeg [73]	CVPR'24	Multi-View ●	46.7 M	6.1	<u>72.7</u>	<u>76.5</u>	-	<u>84.6</u>	-	-	-	-
RangeNet++ [34]	IROS'19	Range View ●	50.4 M	12.0	50.3	52.2	65.5	-	44.6	57.8	30.9	-
FIDNet [59]	IROS'20	Range View ●	6.1 M	31.8	<u>58.9</u>	59.5	71.4	-	54.1	65.4	46.4	-
CENet [61]	ICME'22	Range View ●	6.8 M	<u>33.4</u>	62.6	64.7	73.3	-	55.7	66.8	50.3	-
RangeViT \dagger [35]	CVPR'23	Range View ●	23.7 M	10.0	60.7	64.0	75.2	-	53.6	66.5	-	-
RangeFormer \dagger [36]	ICCV'23	Range View ●	24.3 M	6.2	<u>67.6</u>	73.3	78.1	80.1	<u>63.0</u>	-	-	-
Fast-FRNet	Ours	Frustum-Range ●	7.5 M	33.8	67.1	<u>72.5</u>	<u>78.8</u>	<u>82.1</u>	62.4	<u>71.2</u>	52.5	67.1
FRNet	Ours	Frustum-Range ●	10.0 M	29.1	68.7	73.3	79.0	82.5	63.1	72.3	53.5	68.1

TABLE II: **Semi-supervised LiDAR segmentation** results on the SemanticKITTI [32], nuScenes [47], and ScribbleKITTI [67] benchmarks, under an annotation quota of 1%, 10%, 20%, and 50%, respectively. All **mIoU** scores are given in percentage (%). The **best** and **second best** scores for each data split are highlighted in **bold** and underline.

Method	Venue	Backbone	SemanticKITTI				nuScenes				ScribbleKITTI			
			1%	10%	20%	50%	1%	10%	20%	50%	1%	10%	20%	50%
<i>Sup.-only</i> LaserMix [48]	-	FIDNet [59]	36.2	52.2	55.9	57.2	38.3	57.5	62.7	67.6	33.1	47.7	49.9	52.5
	CVPR'23		43.4	58.8	59.4	61.4	49.5	68.2	70.6	73.0	38.3	54.4	55.6	58.7
<i>Sup.-only</i> CRB [67]	-		45.4	56.1	57.8	58.7	50.9	65.9	66.6	71.2	39.2	48.0	52.1	53.8
LaserMix [48]	CVPR'22		-	58.7	59.1	60.9	-	-	-	-	54.2	56.5	58.9	-
LiM3D [74]	CVPR'23		50.6	60.0	61.9	62.3	55.3	69.9	71.8	73.2	44.2	53.7	55.1	56.8
ImageTo360 [75]	ICCVW'23		-	61.6	62.6	62.8	-	-	-	-	-	60.3	60.5	60.9
FrustumMix	Ours		54.5	58.6	61.4	64.2	-	-	-	-	-	-	-	-
			55.7	62.5	63.0	64.9	60.0	70.0	72.6	74.1	45.6	55.7	58.2	60.8
<i>Sup.-only</i> PolarMix [49]	-		42.7	60.9	62.6	64.1	48.0	67.8	71.5	75.9	40.7	54.6	56.2	58.9
LaserMix [48]	NeurIPS'22		47.2	61.6	63.1	64.8	53.4	69.3	72.0	76.1	42.5	55.8	56.9	59.6
FrustumMix	CVPR'23	PTv3 [12]	51.7	63.3	64.8	65.3	57.4	71.8	74.2	77.4	45.0	56.9	58.3	60.7
			54.2	64.7	65.6	66.1	60.3	72.3	75.6	77.7	46.2	57.5	60.0	61.6
<i>Sup.-only</i> PolarMix [49]	-		44.9	60.4	61.8	63.1	51.9	68.1	70.9	74.6	42.4	53.5	55.1	57.0
LaserMix [48]	NeurIPS'22		50.1	60.9	62.0	63.8	55.6	69.6	71.0	73.8	43.2	55.0	56.1	57.3
FrustumMix	CVPR'23	FRNet	52.9	62.9	63.2	65.0	58.7	71.5	72.3	75.0	45.8	56.8	57.7	59.0
			55.8	64.8	65.2	65.4	61.2	72.2	74.6	75.4	46.6	57.0	59.5	61.2

Velodyne HDL-64E LiDAR sensor. Officially, sequences 00 to 07 and 09 to 10 (19,130 scans) are used for training, sequence 08 (4,071 scans) for validation, and sequences 11 to 21 (20,351 scans) for online testing. The original dataset is annotated with 28 classes, though only 19 merged classes are used for single-scan evaluation. The vertical FOV is $[-25^\circ, 3^\circ]$. **nuScenes** [47] is a multimodal dataset that is widely used in autonomous driving scenarios. It contains LiDAR point clouds captured around streets in Boston and Singapore, provided by 32-beam LiDAR sensors. The dataset includes 1,000 driving scenes with sparser points. It is annotated with 32 classes and only 16 semantic categories are used for evaluation. The vertical FOV is $[-30^\circ, 10^\circ]$. **SemanticPOSS** [68] is a more

challenging benchmark collected at Peking University and features much smaller and sparser point clouds. It consists of 2,988 LiDAR scenes with numerous dynamic instances. Officially, it is divided into 6 sequences, with sequence 2 used for evaluation with 13 merged categories. The vertical FOV is $[-16^\circ, 7^\circ]$. **ScribbleKITTI** [67] shares the same scenes with SemanticKITTI [32] but only provides weak annotations with line scribbles. It contains 189 million labeled points, with approximately 8.06% labeled points available during training. Additionally, we conduct a comprehensive out-of-distribution robustness evaluation experiment on **SemanticKITTI-C** and **nuScenes-C** from the Robo3D [76] benchmark. Each dataset contains eight corruption types, including “fog”, “wet ground”,

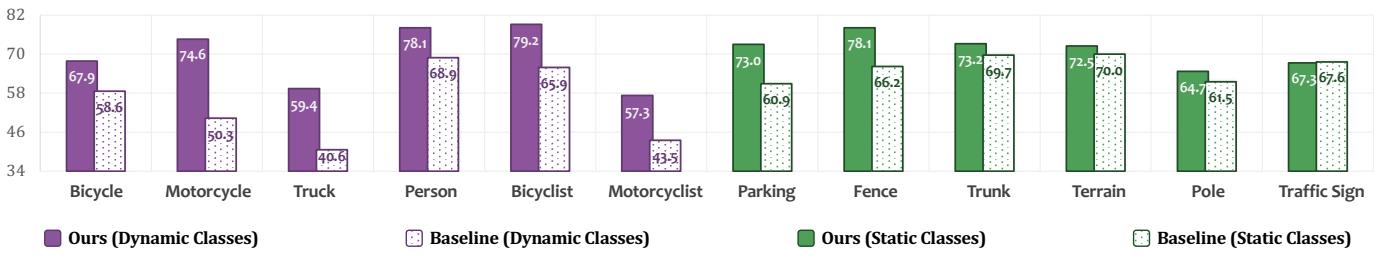


Fig. 7: Class-wise LiDAR segmentation results of FRNet and the baseline model on the *test* set of SemanticKITTI [32].

“snow”, “motion blur”, “beam missing”, “crosstalk”, “incomplete echo”, and “cross sensor”.

Evaluation Metrics. In line with standard protocols, we utilize Intersection-over-Union (IoU) and Accuracy (Acc) for each class i and compute the mean Intersection-over-Union (mIoU) and mean Accuracy (mAcc) to assess performance. IoU and Acc can be calculated as follows:

$$\text{IoU}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i + \text{FN}_i}, \text{Acc}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i}, \quad (11)$$

where $\text{TP}_i, \text{FP}_i, \text{FN}_i$ are true-positives, false-positives, and false-negatives for class i , respectively. For robustness evaluation, we follow the practice of Robo3D [76] to use Corruption Error (CE) and Resilience Rate (RR) to evaluate the robustness of FRNet. The CE and RR are calculated as follows:

$$\text{CE}_i = \frac{\sum_{l=1}^3 (1 - \text{IoU}_{i,l})}{\sum_{l=1}^3 (1 - \text{IoU}_{i,l}^{\text{base}})}, \text{RR}_i = \frac{\sum_{l=1}^3 \text{IoU}_{i,l}}{3 \times \text{IoU}_{\text{clean}}}, \quad (12)$$

where $\text{IoU}_{i,l}$ represents the task-specific IoU. $\text{IoU}_{i,l}^{\text{base}}$ and $\text{IoU}_{\text{clean}}$ denote scores of the baseline model and scores on the “clean” evaluation set, respectively.

Implementation Details. We implement FRNet using the popular MMDetection3D [77] codebase and employ CENet [61] as the 2D backbone to extract frustum features. For SemanticKITTI [32] and ScribbleKITTI [67], the point cloud is divided into 64×512 frustum regions, while for nuScenes [47] and SemanticPOSS [68], it is divided into 32×480 . For optimization, we choose AdamW [78] as our default optimizer with an initial learning rate of 0.01. We utilized the OneCycle scheduler [79] to dynamically adjust the learning rate during training. All models are trained using four GPUs for 50 epochs on SemanticKITTI [32] and ScribbleKITTI [67], and for 80 epochs on nuScenes [47] and SemanticPOSS [68], respectively. The batch size is set to 4 for each GPU. Additionally, we implement a faster variant, dubbed Fast-FRNet, which features a larger frustum area and a scaled-down 2D backbone. Specifically, the point cloud is divided into 32×360 frustum regions for all datasets, and the 2D backbone is a variant of ResNet-18 [80], following the design of CENet [61].

B. Comparative Study

Comparisons to State of the Arts. We begin by comparing FRNet with state-of-the-art LiDAR segmentation methods across different representations, including sparse-voxel, point-view, multi-view, and range-view representations, on both fully-supervised and weakly-supervised benchmarks. Tab. I provides a summary of all the results. In this benchmark,

both FrustumMix and RangeInterpolation are employed as data augmentation techniques to enhance model performance. To ensure a fair comparison, we evaluate the Frame Per Second (FPS) on a single GeForce RTX 2080Ti GPU for all models. For SemanticKITTI [32] and nuScenes [47], we report mIoU scores on both the validation and test sets. For ScribbleKITTI [67] and SemanticPOSS [68], which share the same data across the val and test sets, we report both mIoU and mAcc scores on the val set. Compared with recent multi-view methods [24], although FRNet does not exhibit a significant advantage in performance, it shows notable superiority in terms of parameters and FPS. In comparison with range-view, sparse-voxel, and raw-points methods, FRNet achieves appealing performance while still maintaining high efficiency. Additionally, Fast-FRNet demonstrates the fastest inference speed with only a minor drop in performance.

Label-Efficient LiDAR Semantic Segmentation. Recently, semi-supervised learning has gained traction in LiDAR segmentation. In this work, we adopt the approach of LaserMix [48] to apply FRNet to the semi-supervised segmentation task, as shown in Tab. II. For fair comparisons with other methods, FrustumMix and RangeInterpolation are not employed as data augmentation techniques in this benchmark, and TTA is excluded during inference. We first utilize Cylinder3D [16] and PTv3 [12] as backbones and leverage the proposed FrustumMix technique to mix labeled and unlabeled data for consistency learning, which aligns with the LaserMix framework [48]. Notably, FrustumMix demonstrates superior performance compared to the original LaserMix [48] strategy. Next, we employ FRNet as the backbone and evaluate various strategies for mixing labeled and unlabeled scans, including PolarMix [49], LaserMix [48], and our FrustumMix. The *Sup.-only* baseline results are also obtained without FrustumMix and RangeInterpolation as data augmentation techniques. The results indicate that FrustumMix consistently achieves the best scores across all data splits when compared with other mixing strategies, particularly with limited annotations.

Out-of-Distribution Robustness. Robustness, which reflects a model’s ability to generalize under different conditions, is crucial for automotive security systems, particularly in extreme weather conditions [81], [82]. To evaluate the out-of-distribution robustness of FRNet, we utilize the noisy datasets SemanticKITTI-C and nuScenes-C introduced in Robo3D [76]. In this benchmark, the models trained on SemanticKITTI [32] and nuScenes [47] are directly applied to evaluate the performance on SemanticKITTI-C and nuScenes-C [76], respectively, without any fine-tuning. As shown in

TABLE III: **Robustness probing** on SemanticKITTI-C and nuScenes-C benchmarks [76]. All **mCE**, **mRR** and **mIoU** scores are given in percentage (%). The **best** and second best scores for each model are highlighted in **bold** and underline.

#	Method	Venue	mCE ↓	mRR ↑	mIoU ↑	Fog	Weg	Snow	Motion	Beam	Cross	Echo	Sensor
SemanticKITTI-C	MinkUNet [17]	CVPR'19	<u>100.0</u>	<u>81.9</u>	62.8	55.9	54.0	53.3	32.9	56.3	58.3	54.4	46.1
	SPVCNN [15]	ECCV'20	100.3	82.2	62.5	<u>55.3</u>	54.0	51.4	34.5	56.7	<u>58.1</u>	54.6	46.0
	FIDNet [59]	IROS'20	113.8	77.0	58.8	43.7	51.6	49.7	40.4	49.3	49.5	48.2	29.9
	Cylinder3D [16]	CVPR'21	103.3	80.1	63.4	37.1	57.5	46.9	52.5	57.6	56.0	52.5	46.2
	2DPASS [71]	ECCV'22	106.1	77.5	64.6	40.5	<u>60.7</u>	48.5	57.8	58.8	28.5	55.8	<u>50.0</u>
	CENet [61]	ICME'22	103.4	81.3	62.6	42.7	<u>57.3</u>	<u>53.6</u>	52.7	55.8	45.4	53.4	45.8
	WaffleIron [11]	ICCV'23	109.5	72.2	<u>66.0</u>	45.5	58.6	49.3	33.0	<u>59.3</u>	22.5	58.6	54.6
	FRNet	Ours	96.8	80.0	67.6	47.6	62.2	57.1	<u>56.8</u>	62.5	40.9	<u>58.1</u>	47.3
nuScenes-C	MinkUNet [17]	CVPR'19	<u>100.0</u>	74.4	75.8	53.6	73.9	40.4	73.4	68.5	26.6	63.8	51.0
	SPVCNN [15]	ECCV'20	106.7	74.7	74.4	59.0	72.5	41.1	58.4	65.4	36.8	62.3	49.2
	FIDNet [59]	IROS'20	122.4	73.3	71.4	64.8	68.0	59.0	48.9	48.1	<u>57.5</u>	48.8	23.7
	Cylinder3D [16]	CVPR'21	111.8	72.9	76.2	59.9	72.7	58.1	42.1	64.5	44.4	60.5	42.2
	2DPASS [71]	ECCV'22	98.6	75.2	77.9	64.5	76.8	54.5	62.0	67.8	34.4	63.2	<u>45.8</u>
	CENet [61]	ICME'22	112.8	<u>76.0</u>	73.3	<u>67.0</u>	69.9	<u>61.6</u>	58.3	50.0	60.9	53.3	24.8
	WaffleIron [11]	ICCV'23	106.7	72.8	76.1	56.1	73.9	49.6	59.5	65.2	33.1	61.5	44.0
	FRNet	Ours	98.6	77.5	<u>77.7</u>	69.1	<u>76.6</u>	69.5	54.5	68.3	41.4	58.7	43.1

TABLE IV: **Ablation study of each component in FRNet** on the *val* set of SemanticKITTI [32] and nuScenes [47]. **FFE**: Frustum Feature Encoder. **FP**: Frustum-Point Fusion. **FH**: Fusion Head. **FS**: Frustum-level Supervision. **RI**: RangeInterpolation. **TTA**: Test Time Augmentation. All **mIoU** and **mAcc** scores are given in percentage (%).

FFE	FP	FH	FS	RI	TTA	SemKITTI		nuScenes		mIoU	mAcc
						mIoU	mAcc	mIoU	mAcc		
✓						62.7	72.3	72.8	82.7		
✓	✓					64.3	73.5	73.6	83.4		
✓	✓		✓			66.2	73.9	76.4	84.6		
✓	✓	✓	✓			67.0	74.1	77.2	84.8		
✓	✓	✓	✓	✓		67.6	74.2	77.7	85.2		
✓	✓	✓	✓	✓	✓	68.7	74.9	79.0	85.0		

Tab. III, FRNet achieves promising performance across most corruption types and demonstrates superiority over recent works employing various LiDAR representations, including sparse voxel [15]–[17], range view [59], [61], multi-view fusion [71], and raw points [11].

Quantitative Assessments. We compare the class-wise IoU scores of FRNet with those of the baseline method [61] in Fig. 7. Notably, FRNet demonstrates significant improvement across most semantic classes, particularly for dynamic classes with intricate structures, where it achieves mIoU gains ranging from 15% to 24%. In Fig. 8, we present the prediction results on SemanticKITTI [32], compared with those of state-of-the-art range view methods, including FIDNet [59], CENet [61], and RangeViT [35]. It is evident that FRNet delivers more accurate predictions, particularly on large planar objects such as “vegetation”, “terrain”, and “sidewalk”, which are often challenging to distinguish accurately.

C. Ablation Study

In this section, we will discuss the efficacy of each design in our FRNet architecture. Unless otherwise specified, all experiments are conducted and reported on the *val* set of SemanticKITTI [32] and nuScenes [47], respectively.

TABLE V: **Ablation study of frustum resolution in FRNet** on the *val* set of SemanticKITTI [32]. All **mIoU** and **mAcc** scores are given in percentage (%) without TTA.

Number of Frustums	Height (<i>H</i>)	Width (<i>W</i>)	mIoU	mAcc
131,072	64	2048	66.8	73.8
65,536	64	1024	67.1	73.4
65,536	32	2048	65.5	72.7
32,768	64	512	67.6	74.2
32,768	32	1024	66.0	72.4
16,384	64	256	64.3	70.5
16,384	32	512	65.5	72.1
8,192	32	256	64.3	70.8

Component Designs. Tab. IV summarizes the ablation results of each component in the FRNet architecture. Notably, FrustumMix is employed across all setups in this ablation study. Firstly, we directly apply the frustum feature encoder to CENet [61], incorporating KNN post-processing to predict semantic labels over the entire point cloud, achieving 62.7% and 72.8% mIoU on SemanticKITTI [32] and nuScenes [47], respectively. Next, by integrating point-frustum fusion to hierarchically update point features, the semantic results are predicted with the updated point features in an end-to-end manner, resulting in improvements of 1.6% and 0.8% mIoU, respectively. To regularize the frustum features, the frustum-level supervision is introduced and it leads to significant gains of 1.9% and 2.8% mIoU, respectively. The fusion head module, which fuses multiple features at different levels for more accurate prediction, brings an additional 0.8% mIoU gain on both datasets. Furthermore, the proposed RangeInterpolation, aimed at alleviating empty pixels in the 2D representation by reconstructing semantic surfaces via surrounding range information, provides a performance boost of approximately 0.5% mIoU. Finally, adopting Test Time Augmentation during inference, as demonstrated in prior works, leads to improvements of 1.1% and 1.3% mIoU, respectively.

Frustum Representation Resolution. Finding a suitable frustum resolution is crucial for balancing accuracy and efficiency. In Tab. V, we conduct a series of experiments with various frustum resolutions. While higher resolution can capture more detailed features, it may not always lead to optimal results.

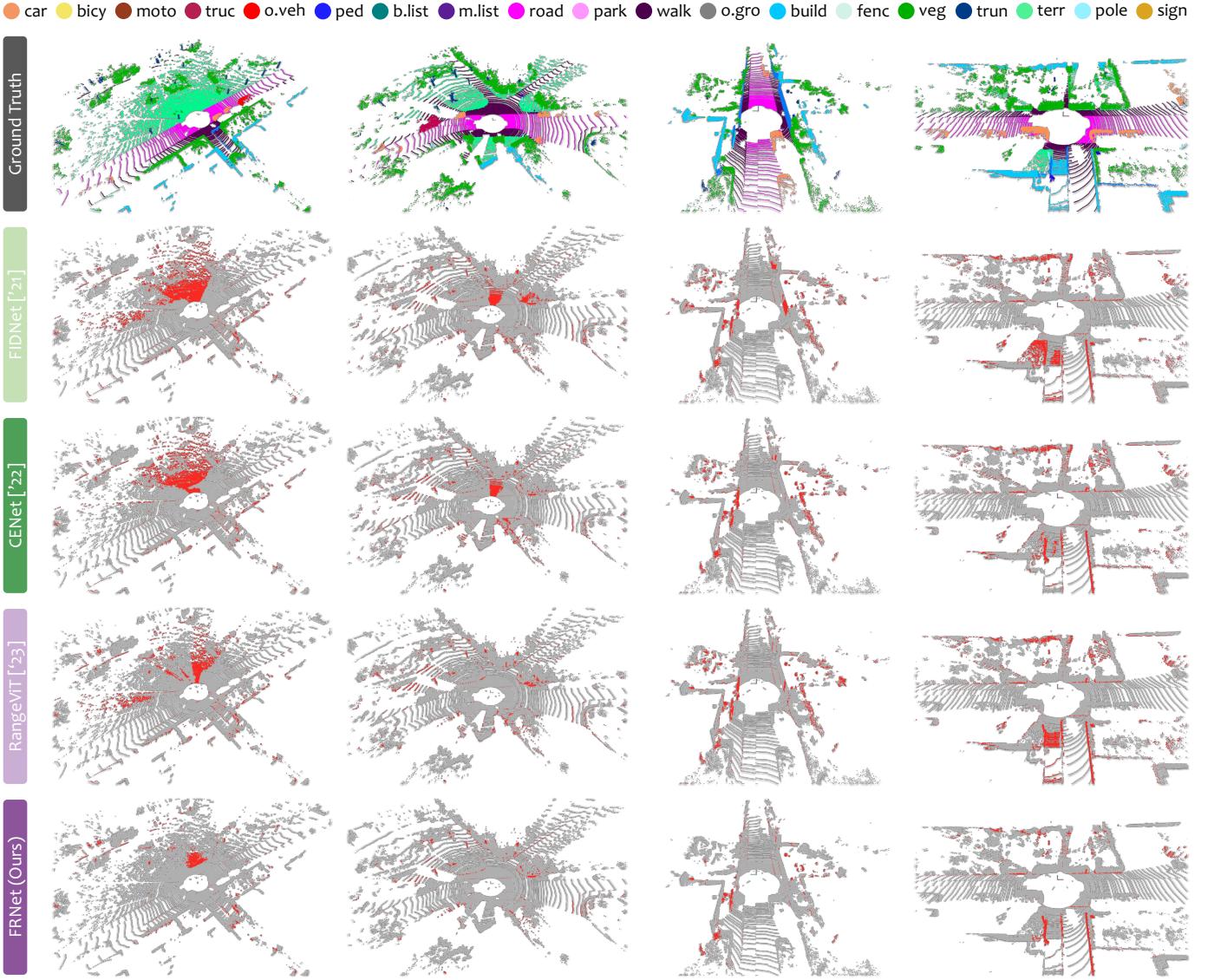


Fig. 8: **Qualitative results** among state-of-the-art LiDAR segmentation methods [35], [59], [61] on the *val* set of SemanticKITTI [32]. To highlight the differences compared with groundtruth, the **correct** and **incorrect** predictions are painted in **gray** and **red**, respectively. Best viewed in colors and zoomed-in for details.

The limited area of the frustum region can result in a lack of meaningful structure, causing the model to struggle with noisy or sparse data. Conversely, although lower resolution can improve inference efficiency and provide a more coherent representation, the larger area of the frustum region may introduce excessive noise information, particularly at object boundaries. Thus, finding an optimal frustum resolution is crucial for balancing the richness of features with computational demands and maintaining overall model performance.

Mixing Strategies. We further explore the effectiveness of the proposed FrustumMix. As shown in Tab. II, FrustumMix significantly improves performance on semi-supervised learning tasks and demonstrates generalizability to other modalities. To contextualize FrustumMix within common data augmentation techniques, we compare it with recent popular outdoor mixing strategies, including LaserMix [48] and PolarMix [49]. As illustrated in Fig. 9(b) and Fig. 9(c), FrustumMix consistently outperforms these prior strategies.

RangeInterpolation Window Settings. RangeInterpolation reconstructs semantic surfaces based on the range image, facilitating a compact 2D representation for coherent semantic learning. Tab. VI showcases the performance of RangeInterpolation with different window settings. Notably, we observe consistent performance improvements when considering interpolation along the horizontal or vertical direction exclusively. Specifically, configurations employing horizontal windows, such as the 1×3 and 1×5 settings, demonstrate superior results compared to those focusing on the vertical direction (3×1 and 5×1 windows). We conjecture that points aligned along the same laser beams often share similar attributes, leading to more accurate and coherent semantic surfaces. Conversely, larger window sizes, such as the 3×5 , 3×3 , and 5×5 configurations, tend to exhibit non-uniform point distributions within the window, resulting in noisy points that fail to accurately represent the expected region and consequently leading to a drop in performance.

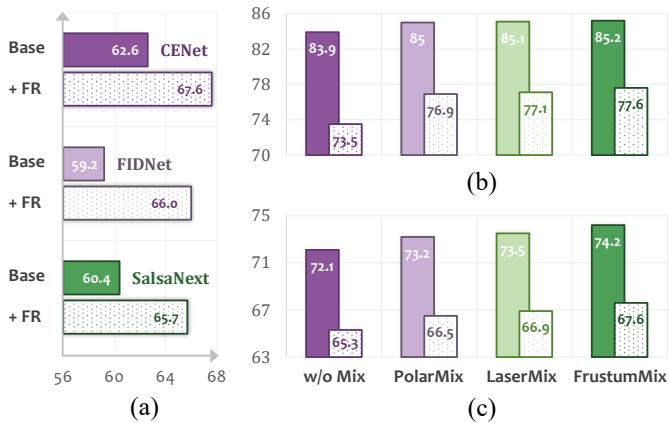


Fig. 9: **Ablation study** on different FRNet configurations. Plots show the comparisons of: a) Different range-view backbones on the *val* set of SemanticKITTI [32]. b) Different mixing strategies on the *val* set of nuScenes [47]. c) Different mixing strategies on the *val* set of SemanticKITTI [32]. Color-filled boxes denote **mAcc** while empty boxes denote **mIoU** in (b) and (c), respectively.

TABLE VI: **Ablation study on window settings** in RangeInterpolation on the *val* set of SemanticKITTI [32]. Green pixel denotes the empty pixel needed to be interpolated. Violet pixels are the surrounding pixels used to generate the point. All **mIoU** scores are given in percentage (%).

Baseline	1×3	3×1	1×5	5×1
67.0	67.6 (+0.6)	67.4 (+0.4)	67.5 (+0.5)	67.4 (+0.4)
cross 3×3	cross 5×5	3×5	3×3	5×5
67.5 (+0.5)	67.4 (+0.4)	67.1 (+0.1)	66.8 (-0.2)	66.6 (-0.4)

2D Backbones. To validate the versatility of our method, we employ various range-view methods as our 2D backbone for extracting frustum features, including SalsaNext [64], FIDNet [59], and CENet [61]. As depicted in Fig. 9(a), our proposed frustum-range representation consistently delivers performance gains ranging from 5.0% to 6.8% in mIoU across different range-view methods.

Inference Efficiency. Multiple sparse convolution backends, including MinkowskiEngine, SpConv, TorchSparse, and TorchSparse++, offer efficient processing for voxel-based methods. To compare FRNet with these backends, we implement MinkUNet [17] using the MMDetection3D codebase across various backends. Additionally, we evaluate the FPS of MinkUNet implemented with the PCSeg and PointCpt codebases, as well as the recent efficient method PTV3 [12]. All evaluations are conducted on a single NVIDIA 2080Ti GPU. As shown in Tab. VII, while TorchSparse++ delivers sat-

TABLE VII: **Inference efficiency comparison** of FRNet and MinkUNet [17] implemented with various codebases and sparse convolution backends, as well as PTV3 [12] on the *val* set of SemanticKITTI [32].

Codebase	Backbone	Backend	FPS
MMDetection3D	MinkUNet	MinkowskiEngine	9.1
	MinkUNet	SpConv	8.6
	MinkUNet	TorchSparse	10.2
	MinkUNet	TorchSparse++	21.3
	FRNet	Convolution	29.1
	Fast-FRNet	Convolution	33.8
PCSeg	MinkUNet	TorchSparse	10.6
PointCpt	MinkUNet PTV3	MinkowskiEngine FlashAttention	8.9 20.2

TABLE VIII: **The multi-source training results of the proposed Fast-FRNet** on the *val* set of the SemanticKITTI [32], nuScenes [47], and SemanticPOSS [68] datasets. All **mIoU** and **mAcc** scores are given in percentage (%).

Method	SemKITTI		nuScenes		SemPOSS	
	mIoU	mAcc	mIoU	mAcc	mIoU	mAcc
Single-Source	67.1	73.8	78.8	84.4	52.5	67.1
Multi-Source	67.4	74.2	79.1	84.7	53.2	68.0
Fast-FRNet (w/ PPT [83])	68.2	74.7	79.3	84.9	53.9	68.4

isfactory efficiency for LiDAR segmentation, FRNet achieves the fastest inference speed, making it suitable for real-time applications in autonomous driving systems.

Multi-Dataset Fusion. Recent studies [51], [83] highlight the importance of large-scale benchmarks achieved through multi-dataset fusion strategies across heterogeneous datasets. In this work, We extend Fast-FRNet to support multi-dataset joint training, as shown in Tab. VIII. Initially, we combine multi-source datasets using mean-variance normalization to align intensity values, which provides slight performance gains compared to single-source training. Additionally, inspired by PPT [83], we incorporate point-prompt techniques to facilitate multi-source joint training. This approach yields further improvements, demonstrating the effectiveness of prompt-guided strategies in multi-dataset fusion.

Open-Vocabulary Semantic Segmentation. To investigate the potential of FRNet in open-vocabulary segmentation, we adopt an approach inspired by OpenScene [84]. Specifically, we integrate text and image features into the framework, aligning them with point features. During inference, text inputs are used to guide the segmentation of point clouds with similar features. As shown in Tab. IX, FRNet demonstrates promising performance in zero-shot segmentation. Additionally, in Fig. 10, we present qualitative results of segmentation using corresponding text inputs, highlighting the capability of FRNet to handle open-world perception tasks effectively.

V. CONCLUSION

In this work, we presented FRNet, a Frustum-Range network designed for efficient and effective LiDAR segmentation, which can be seamlessly integrated into range-view approaches. FRNet comprises three key components: the Frustum Feature Encoder Module for extracting per-point features

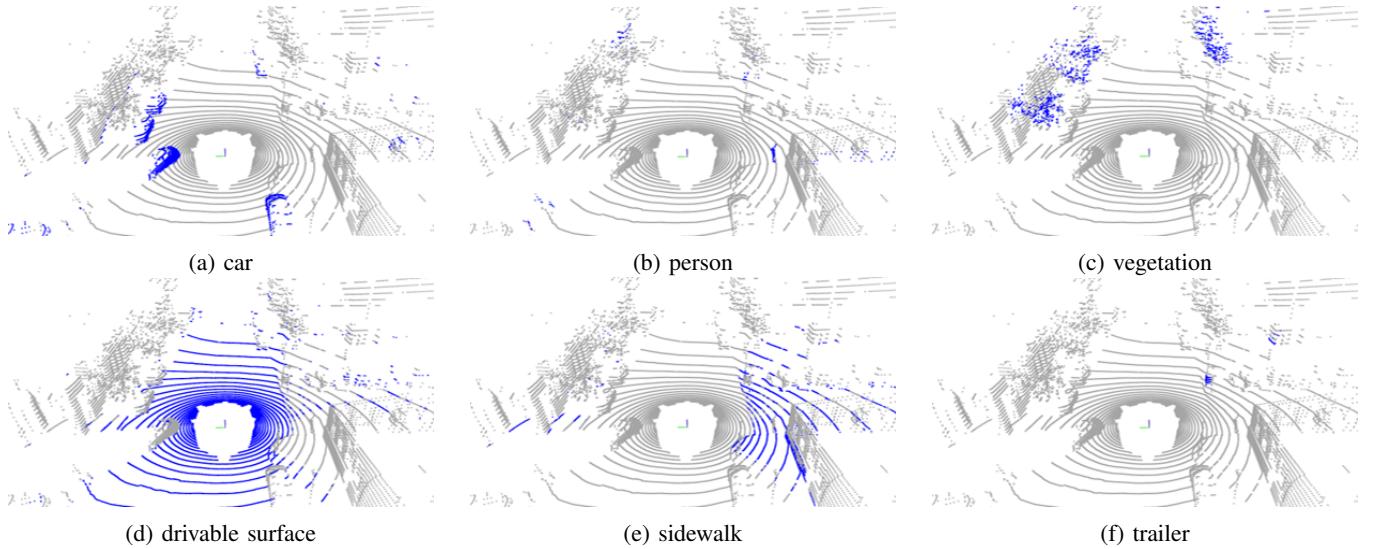


Fig. 10: **The open-vocabulary LiDAR semantic segmentation experiments of FRNet.** Our model demonstrates the ability to perform open-vocabulary segmentation with various textual inputs. The segmented points corresponding to the provided text descriptions are highlighted in blue, showcasing its potential for open-world perception tasks.

TABLE IX: **Zero-shot segmentation performance** on the *val* set of nuScenes [47]. All **mIoU** and **mAcc** scores are given in percentage (%).

Method	nuScenes	
	mIoU	mAcc
OpenScene [84]	42.1	61.8
+ FRNet	43.5	62.0

at lower levels, the Frustum-Point Fusion Module for hierarchical updating of per-point features, and the Fusion Head Module for fusing features from different levels to predict more accurate labels. Additionally, we introduced two efficient augmentation methods, FurustumMix and RangeInterpolation, to enhance the generalizability of range-view LiDAR segmentation. Extensive experiments across diverse driving perception datasets verified that FRNet achieves competitive performance while maintaining high efficiency. We hope this work can lay a solid foundation for future works targeted at real-time, in-vehicle 3D perception. Moving forward, we will further enhance the representation learning capability of FRNet and incorporate such a method into other 3D perception tasks, such as 3D object detection and semantic occupancy prediction, and comprehensive vehicle-road coordination perception tasks.

APPENDIX

VI. ADDITIONAL EXPERIMENTAL RESULTS

In this section, we provide the complete experimental results among the four popular LiDAR segmentation benchmarks used in this work.

A. Scalability and Robustness

In this section, we provide additional analysis of existing LiDAR segmentation approaches regarding their trade-offs between the inference speed (FPS), segmentation accuracy (mIoU), and out-of-training-distribution robustness (mCE). As

shown in Fig. 11, current LiDAR segmentation models pursue either the segmentation accuracy or inference speed. Those more accurate LiDAR segmentation models often contain larger parameter sets and are less efficient during the inference stage. This is particularly predominant for the voxel-based (as shown in yellow circulars) and point-voxel fusion (as shown in blue circulars) approaches. The range-view approaches, on the contrary, are faster in terms of inference speed. However, existing range-view models only achieve sub-par performance compared to the voxel-based and point-voxel fusion approaches. It is worth highlighting again that the proposed FRNet and Fast-FRNet provide a better trade-off in-between speed, accuracy, and robustness. Our models achieve competitive segmentation results and superior robustness over the voxel-based and point-voxel fusion counterparts, while still maintaining high efficiency for real-time LiDAR segmentation.

B. Per-Class Results

In this section, we supplement the complete (class-wise) segmentation results of the baselines, prior works, and our proposed FRNet and Fast-FRNet.

1) *SemanticKITTI*: Tab. X shows the class-wise IoU scores among different LiDAR semantic segmentation methods on the *test* set of SemanticKITTI [32]. We compare FRNet with different LiDAR representations, including point-view, range-view, sparse-voxel-view, and multi-view methods. Results demonstrate that FRNet achieves appealing performance among state-of-the-art methods. Compared with SphereFormer [72] and UniSeg [24], although FRNet obtains sub-par performance, it maintains satisfactory efficiency for real-time processing, achieving the balance between efficiency and accuracy. More results about efficiency and accuracy can be found in the main body. It is worth noting that Fast-FRNet achieves the fastest speed of inference with only a little sacrifice of performance.

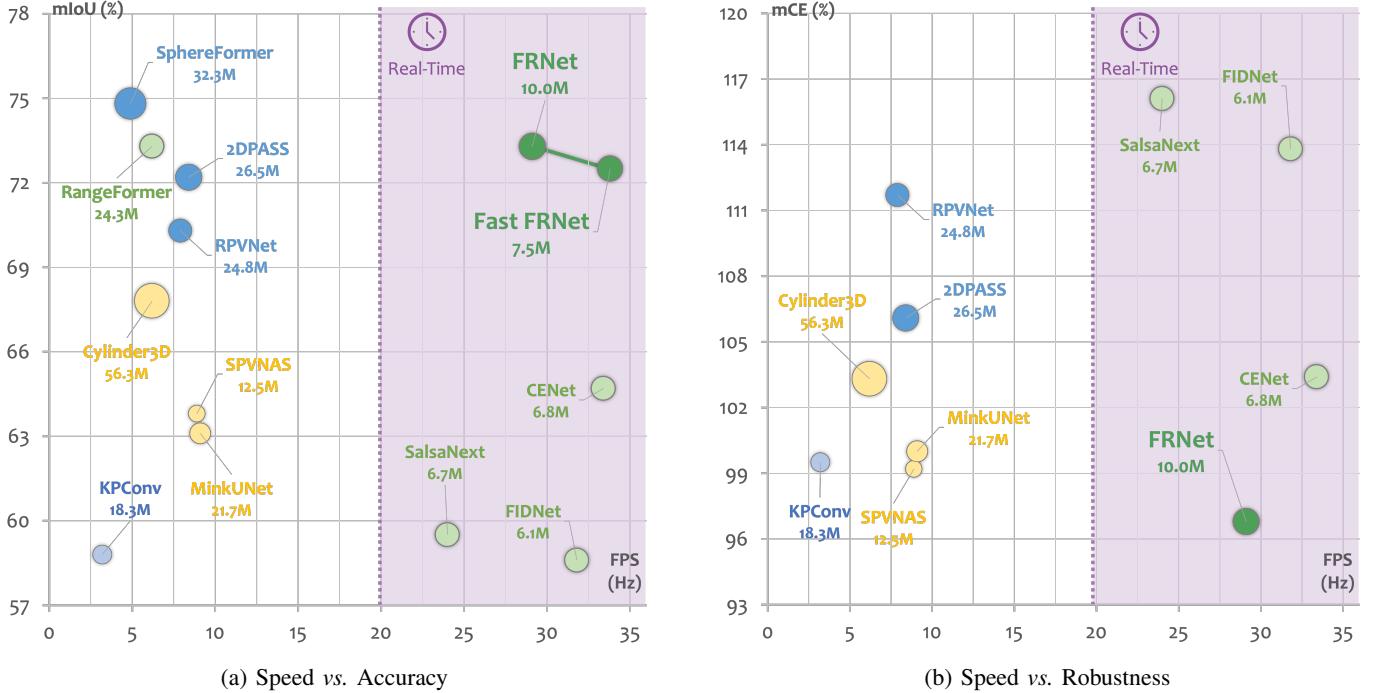


Fig. 11: **The scalability analysis** of existing LiDAR semantic segmentation approaches. Subfigure (a) The inference speed vs. segmentation accuracy (the higher the better). Subfigure (b) The inference speed vs. corruption error (the lower the better).

2) *nuScenes*: Tab. XI and Tab. XII show the class-wise IoU scores among different LiDAR semantic segmentation methods on the *val* and *test* set of nuScenes [47], respectively. The results demonstrate the great advantages of Fast-FRNet and FRNet. For the much sparser LiDAR points, the performance of Fast-FRNet is very close to FRNet with fewer parameters. Compared with popular range-view methods, FRNet achieves great improvement in some dynamic instances, such as *car*, *bicycle*, *motorcycle*, etc.

3) *ScribbleKITTI*: To prove the great advantages of FRNet among range-view methods, we also reimplement the popular LiDAR segmentation methods on ScribbleKITTI [67], including RangeNet++ [34], SalsaNext [64], FIRNet [59], CENet [61], and RangeViT [35]. All the models are trained with their official settings. As shown in Tab. XIII, FRNet achieves state-of-the-art performance. However, the weakly-annotated labels make the frustum-level pseudo labels largely covered by unannotated labels, which will limit performance improvement, especially for *other-ground*.

4) *SemanticPOSS*: We also reimplement some popular sparse-voxel-view methods, including SPVCNN [15], Cylinder3D [16], and MinkNet [17], on SemanticPOSS [68]. All the experiments are conducted based on MMDetection3D framework [77]. Tab. XIV summarizes the class-wise IoU scores among these methods and some other popular range-view methods. Results demonstrate that FRNet achieves state-of-the-art performance among both range-view and voxel-view methods. For some tiny objects, such as *traffic sign* and *cone/stone*, FRNet achieves superior improvements.

VII. BROADER IMPACT

In this section, we elaborate on the positive societal influence and potential limitations of the proposed FRNet.

A. Positive Societal Impacts

Conducting efficient and accurate range view LiDAR semantic segmentation for autonomous driving has several positive societal impacts, particularly in terms of enhancing road safety, improving transportation efficiency, and contributing to broader technological advancements.

- **Enhanced Road Safety.** One of the most significant benefits is the improvement in road safety. Autonomous vehicles equipped with advanced LiDAR semantic segmentation can accurately detect and classify objects in their environment, such as other vehicles, pedestrians, cyclists, and road obstacles. This precise perception capability allows for safer navigation and decision-making, potentially reducing accidents caused by human error.
- **Reduced Traffic Congestion.** Autonomous vehicles with advanced perception systems can optimize driving patterns, leading to smoother traffic flow. This can significantly reduce traffic congestion, especially in urban areas, thereby saving time for commuters and reducing stress associated with driving.
- **Emergency Response and Healthcare.** In emergency situations, autonomous vehicles can be used to quickly and safely transport patients or deliver medical supplies. The precision of LiDAR segmentation ensures that these vehicles can navigate through complex environments effectively.
- **Advancements in Smart City Infrastructure.** The integration of autonomous vehicles with smart city initiatives can lead to more efficient urban planning and infrastructure development. LiDAR data can be used not just for navigation but also for gathering urban data, which can inform city planning and maintenance.

TABLE X: **The class-wise IoU scores** of different LiDAR semantic segmentation approaches on the SemanticKITTI [32] leaderboard. All IoU scores are given in percentage (%). The **best** and second best scores for each class are highlighted in **bold** and underline.

Method	mIoU	mIoU																		
		car	bicycle	motorcycle	walk	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic-sign
PointNet [85]	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++ [63]	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
SqSeg [33]	30.8	68.3	18.1	5.1	4.1	4.8	16.5	17.3	1.2	84.9	28.4	54.7	4.6	61.5	29.2	59.6	25.5	54.7	11.2	36.3
SqSegV2 [54]	39.6	82.7	21.0	22.6	14.5	15.9	20.2	24.3	2.9	88.5	42.4	65.5	18.7	73.8	41.0	68.5	36.9	58.9	12.9	41.0
RandLA-Net [8]	50.3	94.0	19.8	21.4	42.7	38.7	47.5	48.8	4.6	90.4	56.9	67.9	15.5	81.1	49.7	78.3	60.3	59.0	44.2	38.1
RangeNet++ [34]	52.2	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
PolarNet [37]	54.3	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
MFP [23]	55.5	93.4	30.2	38.3	26.1	28.5	48.1	46.1	18.1	90.6	62.3	74.5	30.6	88.5	59.7	83.5	59.7	69.2	49.7	58.1
3D-MiniNet [86]	55.8	90.5	42.3	42.1	28.5	29.4	47.8	44.1	14.5	91.6	64.2	74.5	25.4	89.4	60.8	82.8	60.8	66.7	48.0	56.6
SqSegV3 [56]	55.9	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
KPConv [7]	58.8	96.0	32.0	42.5	33.4	44.3	61.5	61.6	11.8	88.8	61.3	72.7	31.6	95.0	64.2	84.8	69.2	69.1	56.4	47.4
SalsaNext [64]	59.5	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
FIDNet [59]	59.5	93.9	54.7	48.9	27.6	23.9	62.3	59.8	23.7	90.6	59.1	75.8	26.7	88.9	60.5	84.5	64.4	69.0	53.3	62.8
FusionNet [87]	61.3	95.3	47.5	37.7	41.8	34.5	59.5	56.8	11.9	91.8	68.8	77.1	30.8	92.5	69.4	84.5	69.8	68.5	60.4	66.5
PCSCNet [88]	62.7	95.7	48.8	46.2	36.4	40.6	55.5	68.4	55.9	89.1	60.2	72.4	23.7	89.3	64.3	84.2	68.2	68.1	60.5	63.9
KPRNet [60]	63.1	95.5	54.1	47.9	23.6	42.6	65.9	65.0	16.5	93.2	73.9	80.6	30.2	91.7	68.4	85.7	69.8	71.2	58.7	64.1
TornadoNet [89]	63.1	94.2	55.7	48.1	40.0	38.2	63.6	60.1	34.9	89.7	66.3	74.5	28.7	91.3	65.6	85.6	67.0	71.5	58.0	65.9
LiteHDSSeg [66]	63.8	92.3	40.0	55.4	37.7	39.6	59.2	71.6	54.3	93.0	68.2	78.3	29.3	91.5	65.0	78.2	65.8	65.1	59.5	67.7
RangeViT [35]	64.0	95.4	55.8	43.5	29.8	42.1	63.9	58.2	38.1	93.1	70.2	80.0	32.5	92.0	69.0	85.3	70.6	71.2	60.8	64.7
CENet [61]	64.7	91.9	58.6	50.3	40.6	42.3	68.9	65.9	43.5	90.3	60.9	75.1	31.5	91.0	66.2	84.5	69.7	70.0	61.5	67.6
SVASeg [18]	65.2	96.7	56.4	57.0	49.1	56.3	70.6	67.0	15.4	92.3	65.9	76.5	23.6	91.4	66.1	85.2	72.9	67.8	63.9	65.2
AMVNet [21]	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
GFNet [90]	65.4	96.0	53.2	48.3	31.7	47.3	62.8	57.3	44.7	93.6	72.5	80.8	31.2	94.0	73.9	85.2	71.1	69.3	61.8	68.0
JS3C-Net [91]	66.0	95.8	59.3	52.9	54.3	46.0	69.5	65.4	39.9	88.9	61.9	72.1	31.9	92.5	70.8	84.5	69.8	67.9	60.7	68.7
MaskRange [92]	66.1	94.2	56.0	55.7	59.2	52.4	67.6	64.8	31.8	91.7	70.7	77.1	29.5	90.6	65.2	84.6	68.5	69.2	60.2	66.6
SPVNAS [15]	66.4	97.3	51.5	50.8	59.8	58.8	65.7	65.2	43.7	90.2	67.6	75.2	16.9	91.3	65.9	86.1	73.4	71.0	64.2	66.9
MSSNet [93]	66.7	96.8	52.2	48.5	54.4	56.3	67.0	70.9	49.3	90.1	65.5	74.9	30.2	90.5	64.9	84.9	72.7	69.2	63.2	65.1
Cylinder3D [16]	68.9	97.1	67.6	63.8	50.8	58.5	73.7	69.2	48.0	92.2	65.0	77.0	32.3	90.7	66.5	85.6	72.5	69.8	62.4	66.2
AF2S3Net [94]	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 [22]	70.3	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
SDSeg3D [95]	70.4	97.4	58.7	54.2	54.9	65.2	70.2	74.4	52.2	90.9	69.4	76.7	41.9	93.2	71.1	86.1	74.3	71.1	65.4	70.6
GASN [96]	70.7	96.9	65.8	58.0	59.3	61.0	<u>80.4</u>	82.7	46.3	89.8	66.2	74.6	30.1	92.3	69.6	<u>87.3</u>	73.0	72.5	66.1	71.6
PKVd [69]	71.2	97.0	67.9	69.3	53.5	60.2	75.1	73.5	50.5	91.8	70.9	77.5	41.0	92.4	69.4	86.5	73.8	71.9	64.9	65.8
Fast-FRNet	72.5	97.1	66.0	73.1	59.3	65.7	76.0	78.9	54.5	91.8	72.6	77.6	42.0	92.4	70.6	86.5	71.9	72.5	63.1	66.7
2DPASS [71]	72.9	97.0	63.6	63.4	<u>61.1</u>	61.5	77.9	<u>81.3</u>	74.1	89.7	67.4	74.7	40.0	93.5	<u>72.9</u>	86.2	73.9	71.0	65.0	70.4
RangeFormer [36]	73.3	96.7	69.4	73.7	59.9	66.2	78.1	75.9	58.1	92.4	73.0	78.8	42.4	92.3	70.1	86.6	73.3	72.8	66.4	66.6
FRNet	73.3	97.3	67.9	74.6	59.4	66.3	78.1	79.2	57.3	92.1	73.0	78.1	41.8	92.7	71.0	86.7	73.2	72.5	64.7	67.3
SphereFormer [72]	74.8	97.5	<u>70.1</u>	70.5	59.6	67.7	79.0	80.4	75.3	91.8	69.7	78.2	41.3	<u>93.8</u>	72.8	86.7	<u>75.1</u>	72.4	<u>66.8</u>	72.9
UniSeg [24]	75.2	97.9	71.9	<u>75.2</u>	63.6	74.1	78.9	74.8	60.6	92.6	74.0	79.5	<u>46.1</u>	93.4	72.7	87.5	76.3	73.1	68.3	68.5

B. Potential Limitations

Although the proposed FRNet and Fast-FRNet are capable of providing a better trade-off between LiDAR segmentation network efficiency and accuracy, there are several limitations within the current framework. First, frustum-level supervision counts the high-frequency semantic labels in the frustum region, which will cover the objects with few points. Such supervision weakens the performance of FRNet on tiny objects. Second, FRNet cannot handle objects with similar structures well. Some objects share a similar appearance with different semantic attributes, which limits the ability of FRNet to distinguish them in a 2D manner.

VIII. PUBLIC RESOURCES USED

In this section, we acknowledge the use of public resources, during the course of this work.

A. Public Codebase Used

We acknowledge the use of the following public codebase, during the course of this work:

- MMDetection⁴ Apache License 2.0
- PCSeg⁵ Apache License 2.0
- Pointcept⁶ MIT License

B. Public Datasets Used

We acknowledge the use of the following public datasets, during the course of this work:

- SemanticKITTI⁷ CC BY-NC-SA 4.0
- SemanticKITTI-API⁸ MIT License
- nuScenes⁹ CC BY-NC-SA 4.0
- nuScenes-devkit¹⁰ Apache License 2.0
- ScribbleKITTI¹¹ Unknown
- SemanticPOSS¹² CC BY-NC-SA 3.0
- SemanticPOSS-API¹³ MIT License
- Robo3D¹⁴ CC BY-NC-SA 4.0

⁴<https://github.com/open-mmlab/mmdetection>.

⁵<https://github.com/PJLab-ADG/PCSeg>.

⁶<https://github.com/Pointcept/Pointcept>.

⁷<http://semantic-kitti.org>.

⁸<https://github.com/PRBonn/semantic-kitti-api>.

⁹<https://www.nuscenes.org/nuscenes>.

¹⁰<https://github.com/nutonomy/nuscenes-devkit>.

¹¹<https://github.com/ouenal/scribblekitti>.

¹²<http://www.poss.pku.edu.cn/semanticposs.html>.

¹³<https://github.com/Theia-4869/semantic-poss-api>.

¹⁴<https://github.com/lkdong1205/Robo3D>.

TABLE XI: **The class-wise IoU scores** of different LiDAR semantic segmentation approaches on the *val* set of nuScenes [47]. All IoU scores are given in percentage (%). The **best** and second best scores for each class are highlighted in **bold** and underline.

Method (year)	mIoU	barrier	bicycle	bus	car	construction-vehicle	motorcycle	pedestrian	traffic-cone	trailer	truck	driveable-surface	other-ground	sidewalk	terrain	manmade	vegetation
AF2S3Net [94] ['21]	62.2	60.3	12.6	82.3	80.0	20.1	62.0	59.0	49.0	42.2	67.4	94.2	68.0	64.1	68.6	82.9	82.4
RangeNet++ [34] ['19]	65.5	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
PolarNet [37] ['20]	71.0	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
PCSCNet [88] ['22]	72.0	73.3	42.2	87.8	86.1	44.9	82.2	76.1	62.9	49.3	77.3	95.2	66.9	69.5	72.3	83.7	82.5
SalsaNext [64] ['20]	72.2	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
SVASeq [18] ['22]	74.7	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
RangeViT [35] ['23]	75.2	75.5	40.7	88.3	90.1	49.3	79.3	77.2	66.3	65.2	80.0	96.4	71.4	73.8	73.8	89.9	87.2
Cylinder3D [16] ['21]	76.1	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
AMVNet [21] ['20]	76.1	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	<u>90.8</u>	87.9
RPVNet [22] ['21]	77.6	78.2	43.4	92.7	<u>93.2</u>	49.0	85.7	80.5	66.0	66.9	84.0	96.9	73.5	75.9	70.6	90.6	<u>88.9</u>
RangeFormer [36] ['23]	78.1	78.0	45.2	94.0	92.9	58.7	83.9	77.9	<u>69.1</u>	63.7	<u>85.6</u>	96.7	74.5	75.1	75.3	89.1	87.5
SphereFormer [72] ['23]	78.4	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
Fast-FRNet	78.8	78.7	42.3	95.6	93.1	58.9	86.3	77.9	66.9	72.1	85.4	97.0	76.3	76.5	76.2	89.7	87.8
FRNet	79.0	78.5	43.9	<u>95.4</u>	93.2	56.3	85.8	79.0	68.5	72.8	86.5	<u>97.1</u>	75.9	<u>77.0</u>	76.4	89.7	88.0

TABLE XII: **The class-wise IoU scores** of different LiDAR semantic segmentation approaches on the *test* set of nuScenes [47]. All IoU scores are given in percentage (%). The **best** and second best scores for each class are highlighted in **bold** and underline.

Method (year)	mIoU	barrier	bicycle	bus	car	construction-vehicle	motorcycle	pedestrian	traffic-cone	trailer	truck	driveable-surface	other-ground	sidewalk	terrain	manmade	vegetation
PolarNet [37] ['20]	69.4	72.2	16.8	77.0	86.5	51.1	69.7	64.8	54.1	69.7	63.5	96.6	67.1	77.7	72.1	87.1	84.5
JS3C-Net [91] ['21]	73.6	80.1	26.2	87.8	84.5	55.2	72.6	71.3	66.3	76.8	71.2	96.8	64.5	76.9	74.1	87.5	86.1
PMF [97] ['21]	77.0	82.0	40.0	81.0	88.0	64.0	79.0	80.0	76.0	81.0	67.0	97.0	68.0	78.0	74.0	90.0	88.0
Cylinder3D [16] ['21]	77.2	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
AMVNet [21] ['20]	77.3	80.6	32.0	81.7	88.9	67.1	84.3	76.1	73.5	84.9	67.3	97.5	67.4	79.4	75.5	91.5	88.7
SPVCNN [15] ['20]	77.4	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
AF2S3Net [94] ['21]	78.3	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
2D3DNet [98] ['21]	80.0	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
RangeFormer [36] ['23]	80.1	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
GASN [96] ['22]	80.4	85.5	43.2	90.5	<u>92.1</u>	64.7	86.0	83.0	73.3	83.9	75.8	97.0	71.0	<u>81.0</u>	77.7	91.6	90.2
2DPASS [71] ['22]	80.8	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
LidarMultiNet [99] ['23]	81.4	80.4	48.4	94.3	90.0	71.5	87.2	85.2	80.4	<u>86.9</u>	74.8	<u>97.8</u>	67.3	80.7	76.5	92.1	<u>89.6</u>
SphereFormer [72] ['23]	81.9	83.3	39.2	94.7	92.5	<u>77.5</u>	84.2	<u>84.4</u>	<u>79.1</u>	88.4	78.3	97.9	69.0	81.5	<u>77.2</u>	93.4	90.2
Fast-FRNet	82.1	85.3	62.9	92.1	91.5	76.5	87.7	75.9	73.2	86.2	76.0	97.8	71.9	80.9	77.1	90.8	88.1
FRNet	<u>82.5</u>	<u>85.8</u>	<u>65.4</u>	92.1	91.6	77.4	<u>87.9</u>	77.4	74.3	86.0	75.7	<u>97.8</u>	<u>71.8</u>	80.8	77.0	91.0	88.3
UniSeg [24] ['23]	83.5	85.9	71.2	92.1	91.6	80.5	88.0	80.9	76.0	86.3	<u>76.7</u>	<u>97.7</u>	<u>71.8</u>	80.7	76.7	91.3	88.8

C. Public Implementations Used

We acknowledge the use of the following public implementations, during the course of this work:

- RangeNet++¹⁵ MIT License
- SalsaNext¹⁶ MIT License
- FIDNet¹⁷ Unknown
- CENet¹⁸ MIT License
- RangeViT¹⁹ Apache License 2.0
- SphereFormer²⁰ Apache License 2.0
- 2DPASS²¹ MIT License

- Cylinder3D²² Apache License 2.0
- SPVNas²³ MIT License
- KPConv²⁴ MIT License
- RandLA-Net²⁵ CC BY-NC-SA 4.0
- Codes-for-PVKD²⁶ MIT License
- LaserMix²⁷ Apache License 2.0

REFERENCES

- [1] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, “Deep learning for 3d point clouds: A survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 12, pp. 4338–4364, 2020.

¹⁵<https://github.com/PRBonn/lidar-bonnetal>.

¹⁶<https://github.com/TiagoCortinhal/SalsaNext>.

¹⁷<https://github.com/placeforyiming/IROS21-FIDNet-SemanticKITTI>.

¹⁸<https://github.com/huixiancheng/CENet>.

¹⁹<https://github.com/valeoai/rangevit>.

²⁰<https://github.com/dvlab-research/SphereFormer>.

²¹<https://github.com/yanx27/2DPASS>.

²²<https://github.com/xinge008/Cylinder3D>.

²³<https://github.com/mit-han-lab/spvnas>.

²⁴<https://github.com/HuguesTHOMAS/KPConv-PyTorch>.

²⁵<https://github.com/QingyongHu/RandLA-Net>.

²⁶<https://github.com/cardwing/Codes-for-PVKD>.

²⁷<https://github.com/lkdong1205/LaserMix>.

TABLE XIII: **The class-wise IoU scores** of different LiDAR semantic segmentation approaches on the ScribbleKITTI [67] leaderboard. All IoU scores are given in percentage (%). The **best** and second best scores for each class are highlighted in **bold** and underline.

Method	mIoU	car	bicycle	motorcycle	walk	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic-sign
RangeNet++ [34]	44.6	84.6	24.6	38.9	13.7	12.9	29.0	51.4	0.0	86.0	35.2	72.4	<u>4.5</u>	80.2	41.6	80.1	51.2	66.9	43.9	34.5
SalsaNext [64]	50.3	85.6	33.8	42.7	8.4	20.8	64.3	71.5	<u>0.2</u>	85.6	31.6	72.0	1.1	84.0	39.9	81.0	62.0	63.6	60.5	46.3
RangeViT [35]	53.6	85.6	31.6	50.1	40.3	36.3	57.6	68.7	0.0	86.1	32.6	75.2	0.3	87.9	49.3	83.6	62.8	67.5	59.6	43.7
FIDNet [59]	54.1	85.6	36.7	48.7	60.8	38.4	63.3	68.2	0.0	84.1	25.9	71.2	0.4	85.6	41.3	81.7	64.1	62.7	61.5	48.0
MinkNet [17]	55.0	88.1	13.2	55.1	72.3	36.9	61.3	77.1	0.0	83.4	32.7	71.0	0.3	90.0	50.0	84.1	<u>66.6</u>	65.8	61.6	35.2
CENet [61]	55.7	86.1	39.4	53.2	61.0	46.1	69.2	72.2	0.0	85.7	28.7	72.6	1.1	85.8	43.1	81.8	64.2	63.8	59.6	45.0
SPVCNN [15]	56.9	88.6	25.7	55.9	67.4	48.8	65.0	78.2	0.0	82.6	30.4	70.1	0.3	90.5	49.6	84.4	67.6	66.1	61.6	48.7
Cylinder3D [16]	57.0	88.5	39.9	58.0	58.4	48.1	68.6	77.0	<u>0.5</u>	84.4	30.4	72.2	2.5	89.4	48.4	81.9	64.6	59.8	61.2	48.7
Fast-FRNet	62.4	<u>90.8</u>	41.0	66.8	<u>81.7</u>	64.0	70.5	84.2	0.0	91.3	35.1	78.3	0.0	89.4	<u>64.4</u>	85.0	65.5	69.5	60.5	<u>47.7</u>
RangeFormer [36]	63.0	92.6	51.6	65.7	74.4	49.6	71.6	82.1	0.0	94.8	44.4	80.6	<u>11.4</u>	85.6	56.9	87.2	64.1	77.0	62.7	45.1
FRNet	63.1	90.5	42.3	66.8	82.4	63.0	73.4	86.2	0.0	92.1	35.1	79.1	0.3	90.4	64.7	<u>85.4</u>	65.8	<u>70.7</u>	61.8	<u>48.6</u>

TABLE XIV: **The class-wise IoU scores** of different LiDAR semantic segmentation approaches on the SemanticPOSS [68] leaderboard. All IoU scores are given in percentage (%). The **best** and second best scores for each class are highlighted in **bold** and underline.

Method (year)	mIoU	person	rider	car	truck	plants	traffic sign	pole	trashcan	building	cone/stone	walk	fence	bike
SqSeg [33] ['18]	18.9	14.2	1.0	13.2	10.4	28.0	5.1	5.7	2.3	43.6	0.2	15.6	31.0	75.0
SqSegV2 [54] ['19]	30.0	48.0	9.4	48.5	11.3	50.1	6.7	6.2	14.8	60.4	5.2	22.1	36.1	71.3
RangeNet++ [34] ['19]	30.9	57.3	4.6	35.0	14.1	58.3	3.9	6.9	24.1	66.1	6.6	23.4	28.6	73.5
MINet [100] ['21]	43.2	62.4	12.1	63.8	22.3	68.6	16.7	30.1	28.9	75.1	28.6	32.2	44.9	76.3
FIDNet [59] ['21]	46.4	72.2	23.1	72.7	23.0	68.0	22.2	28.6	16.3	73.1	34.0	40.9	50.3	79.1
SPVCNN [15] ['20]	48.4	72.5	24.7	72.1	<u>31.4</u>	72.7	10.8	<u>41.3</u>	31.8	78.4	23.8	42.6	51.7	75.3
CENet [61] ['22]	50.3	75.5	22.0	77.6	25.3	72.2	18.2	31.5	48.1	76.3	27.7	47.7	51.4	80.3
Fast-FRNet	52.5	76.9	28.3	79.9	28.8	73.8	<u>30.2</u>	32.9	28.5	80.9	<u>40.7</u>	47.3	55.9	79.0
Cylinder3D [16] ['21]	52.9	75.9	30.0	75.8	28.7	75.7	29.5	37.2	<u>36.7</u>	82.3	34.1	47.5	53.9	<u>80.1</u>
MinkNet [17] ['19]	53.1	77.8	<u>29.1</u>	81.3	33.9	<u>75.2</u>	22.0	42.5	36.4	80.7	23.9	51.2	57.5	79.1
FRNet	53.5	<u>77.7</u>	28.7	<u>81.2</u>	28.9	74.3	30.4	34.5	29.8	<u>81.1</u>	45.6	<u>47.9</u>	<u>56.4</u>	79.2

- [2] D. Kong, X. Li, Q. Xu, Y. Hu, and P. Ni, “Sc_lpr: Semantically consistent lidar place recognition based on chained cascade network in long-term dynamic environments,” *IEEE Transactions on Image Processing*, vol. 33, pp. 2145–2157, 2024.
- [3] Y. Li, L. Kong, H. Hu, X. Xu, and X. Huang, “Is your lidar placement optimized for 3d scene understanding?” in *Advances in Neural Information Processing Systems*, vol. 36, 2024.
- [4] L. Kong, X. Xu, J. Cen, W. Zhang, L. Pan, K. Chen, and Z. Liu, “Calib3d: Calibrating model preferences for reliable 3d scene understanding,” in *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2025.
- [5] T. Sun, Z. Zhang, X. Tan, Y. Qu, and Y. Xie, “Image understands point cloud: Weakly supervised 3d semantic segmentation via association learning,” *IEEE Transactions on Image Processing*, vol. 33, pp. 1838–1852, 2024.
- [6] P. Hu, S. Sclaroff, and K. Saenko, “Leveraging geometric structure for label-efficient semi-supervised scene segmentation,” *IEEE Transactions on Image Processing*, vol. 31, pp. 6320–6330, 2022.
- [7] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, “Kpcconv: Flexible and deformable convolution for point clouds,” in *IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6411–6420.
- [8] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, “Randla-net: Efficient semantic segmentation of large-scale point clouds,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 108–11 117.
- [9] H. Shuai, X. Xu, and Q. Liu, “Backward attentive fusing network with local aggregation classifier for 3d point cloud semantic segmentation,” *IEEE Transactions on Image Processing*, vol. 30, pp. 4973–4984, 2021.
- [10] T. Zhang, M. Ma, F. Yan, H. Li, and Y. Chen, “Pids: Joint point interaction-dimension search for 3d point cloud,” in *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 1298–1307.
- [11] G. Puy, A. Boulch, and R. Marlet, “Using a waffle iron for automotive point cloud semantic segmentation,” in *IEEE/CVF International Conference on Computer Vision*, 2023, pp. 3379–3389.
- [12] X. Wu, L. Jiang, P.-S. Wang, Z. Liu, X. Liu, Y. Qiao, W. Ouyang, T. He, and H. Zhao, “Point transformer v3: Simpler faster stronger,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 4840–4851.
- [13] J. Li, X. He, Y. Wen, Y. Gao, X. Cheng, and D. Zhang, “Panoptic-phnet: Towards real-time and high-precision lidar panoptic segmentation via clustering pseudo heatmap,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 809–11 818.
- [14] F. Hong, L. Kong, H. Zhou, X. Zhu, H. Li, and Z. Liu, “Unified 3d and 4d panoptic segmentation via dynamic shifting networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
- [15] H. Tang, Z. Liu, S. Zhao, Y. Lin, J. Lin, H. Wang, and S. Han, “Searching efficient 3d architectures with sparse point-voxel convolution,” in *European Conference on Computer Vision*, 2020, pp. 685–702.
- [16] X. Zhu, H. Zhou, T. Wang, F. Hong, Y. Ma, W. Li, H. Li, and D. Lin, “Cylindrical and asymmetrical 3d convolution networks for lidar segmentation,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9939–9948.
- [17] C. Choy, J. Gwak, and S. Savarese, “4d spatio-temporal convnets: Minkowski convolutional neural networks,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3075–3084.
- [18] L. Zhao, S. Xu, L. Liu, D. Ming, and W. Tao, “Svaseg: Sparse voxel-based attention for 3d lidar point cloud semantic segmentation,” *Remote Sensing*, vol. 14, no. 18, p. 4471, 2022.
- [19] V. Vanian, G. Zamanakos, and I. Pratikakis, “Improving performance of deep learning models for 3d point cloud semantic segmentation via attention mechanisms,” *Computers & Graphics*, vol. 106, pp. 277–287, 2022.
- [20] A. Boulch, C. Sautier, B. Michele, G. Puy, and R. Marlet, “Also: Automotive lidar self-supervision by occupancy estimation,” in *IEEE/CVF*

- Conference on Computer Vision and Pattern Recognition*, 2023, pp. 13 455–13 465.
- [21] 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,” *arXiv preprint arXiv:2012.04934*, 2020.
- [22] J. Xu, R. Zhang, J. Dou, Y. Zhu, J. Sun, and S. Pu, “Rpynet: A deep and efficient range-point-voxel fusion network for lidar point cloud segmentation,” in *IEEE/CVF International Conference on Computer Vision*, 2021, pp. 16 024–16 033.
- [23] Y. A. Alnaggar, M. Afifi, K. Amer, and M. ElHelw, “Multi projection fusion for real-time semantic segmentation of 3d lidar point clouds,” in *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 1800–1809.
- [24] Y. Liu, R. Chen, X. Li, L. Kong, Y. Yang, Z. Xia, Y. Bai, X. Zhu, Y. Ma, Y. Li *et al.*, “Uniseg: A unified multi-modal lidar segmentation network and the openpcseg codebase,” in *IEEE/CVF International Conference on Computer Vision*, 2023, pp. 21 662–21 673.
- [25] R. Chen, Y. Liu, L. Kong, X. Zhu, Y. Ma, Y. Li, Y. Hou, Y. Qiao, and W. Wang, “Clip2scene: Towards label-efficient 3d scene understanding by clip,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7020–7030.
- [26] R. Chen, Y. Liu, L. Kong, N. Chen, X. Zhu, Y. Ma, T. Liu, and W. Wang, “Towards label-free scene understanding by vision foundation models,” in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 75 896–75 910.
- [27] M. Jaritz, T.-H. Vu, R. De Charette, É. Wirbel, and P. Pérez, “Cross-modal learning for domain adaptation in 3d semantic segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 2, pp. 1533–1544, 2022.
- [28] J. Xu, W. Yang, L. Kong, Y. Liu, R. Zhang, Q. Zhou, and B. Fei, “Visual foundation models boost cross-modal unsupervised domain adaptation for 3d semantic segmentation,” *arXiv preprint arXiv:2403.10001*, 2024.
- [29] H. Yu, Y. Luo, M. Shu, Y. Huo, Z. Yang, Y. Shi, Z. Guo, H. Li, X. Hu, J. Yuan *et al.*, “Dair-v2x: A large-scale dataset for vehicle-infrastructure cooperative 3d object detection,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 21 361–21 370.
- [30] R. Xu, X. Xia, J. Li, H. Li, S. Zhang, Z. Tu, Z. Meng, H. Xiang, X. Dong, R. Song *et al.*, “V2v4real: A real-world large-scale dataset for vehicle-to-vehicle cooperative perception,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 13 712–13 722.
- [31] H. Yu, W. Yang, H. Ruan, Z. Yang, Y. Tang, X. Gao, X. Hao, Y. Shi, Y. Pan, N. Sun *et al.*, “V2x-seq: A large-scale sequential dataset for vehicle-infrastructure cooperative perception and forecasting,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 5486–5495.
- [32] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, “Semantickitti: A dataset for semantic scene understanding of lidar sequences,” in *IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9297–9307.
- [33] 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 *IEEE International Conference on Robotics and Automation*, 2018, pp. 1887–1893.
- [34] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, “Rangenet++: Fast and accurate lidar semantic segmentation,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 4213–4220.
- [35] A. Ando, S. Gidaris, A. Bursuc, G. Puy, A. Boulch, and R. Marlet, “Rangevit: Towards vision transformers for 3d semantic segmentation in autonomous driving,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 5240–5250.
- [36] L. Kong, Y. Liu, R. Chen, Y. Ma, X. Zhu, Y. Li, Y. Hou, Y. Qiao, and Z. Liu, “Rethinking range view representation for lidar segmentation,” in *IEEE/CVF International Conference on Computer Vision*, 2023, pp. 228–240.
- [37] Y. Zhang, Z. Zhou, P. David, X. Yue, Z. Xi, B. Gong, and H. Foroosh, “Polarinet: An improved grid representation for online lidar point clouds semantic segmentation,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9601–9610.
- [38] Q. Chen, S. Vora, and O. Beijbom, “Polarstream: Streaming object detection and segmentation with polar pillars,” in *Advances in Neural Information Processing Systems*, vol. 34, 2021, pp. 26 871–26 883.
- [39] E. E. Aksoy, S. Baci, and S. Cavdar, “Salsanet: Fast road and vehicle segmentation in lidar point clouds for autonomous driving,” in *IEEE Intelligent Vehicles Symposium*, 2020, pp. 926–932.
- [40] Z. Zhou, Y. Zhang, and H. Foroosh, “Panoptic-polarinet: Proposal-free lidar point cloud panoptic segmentation,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13 194–13 203.
- [41] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.
- [42] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2881–2890.
- [43] 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 *Advances in Neural Information Processing Systems*, vol. 34, 2021, pp. 12 077–12 090.
- [44] Z. Liang, M. Zhang, Z. Zhang, X. Zhao, and S. Pu, “Rangerenn: Towards fast and accurate 3d object detection with range image representation,” *arXiv preprint arXiv:2009.00206*, 2020.
- [45] Z. Tian, X. Chu, X. Wang, X. Wei, and C. Shen, “Fully convolutional one-stage 3d object detection on lidar range images,” in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 34 899–34 911.
- [46] L. Kong, N. Quader, and V. E. Liong, “Conda: Unsupervised domain adaptation for lidar segmentation via regularized domain concatenation,” in *IEEE International Conference on Robotics and Automation*, 2023, pp. 9338–9345.
- [47] W. K. Fong, R. Mohan, J. V. Hurtado, L. Zhou, H. Caesar, O. Beijbom, and A. Valada, “Panoptic nuscenes: A large-scale benchmark for lidar panoptic segmentation and tracking,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3795–3802, 2022.
- [48] L. Kong, J. Ren, L. Pan, and Z. Liu, “Lasermix for semi-supervised lidar semantic segmentation,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21 705–21 715.
- [49] A. Xiao, J. Huang, D. Guan, K. Cui, S. Lu, and L. Shao, “Polarmix: A general data augmentation technique for lidar point clouds,” in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 11 035–11 048.
- [50] A. Nekrasov, J. Schult, O. Litany, B. Leibe, and F. Engelmann, “Mix3d: Out-of-context data augmentation for 3d scenes,” in *IEEE International Conference on 3D Vision*, 2021, pp. 116–125.
- [51] Y. Liu, L. Kong, X. Wu, R. Chen, X. Li, L. Pan, Z. Liu, and Y. Ma, “Multi-space alignments towards universal lidar segmentation,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 14 648–14 661.
- [52] Y. Liu, L. Kong, J. Cen, R. Chen, W. Zhang, L. Pan, K. Chen, and Z. Liu, “Segment any point cloud sequences by distilling vision foundation models,” in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 37 193–37 229.
- [53] R. Li, A.-Q. Cao, and R. de Charette, “Coarse3d: Class-prototypes for contrastive learning in weakly-supervised 3d point cloud segmentation,” in *British Machine Vision Conference*, 2022.
- [54] 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 *IEEE International Conference on Robotics and Automation*, 2019, pp. 4376–4382.
- [55] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, “Squeezezenet: Alexnet-level accuracy with 50x fewer parameters and 0.5 mb model size,” *arXiv preprint arXiv:1602.07360*, 2016.
- [56] C. Xu, B. Wu, Z. Wang, W. Zhan, P. Vajda, K. Keutzer, and M. Tomizuka, “Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation,” in *European Conference on Computer Vision*, 2020, pp. 1–19.
- [57] Y. Chen, V. T. Hu, E. Pavlos, T. Mensink, P. Mettes, P. Yang, and C. G. Snoek, “Pointmixup: Augmentation for point clouds,” in *European Conference on Computer Vision*, 2020, pp. 330–345.
- [58] J. Zhang, L. Chen, B. Ouyang, B. Liu, J. Zhu, Y. Chen, Y. Meng, and D. Wu, “Pointcutmix: Regularization strategy for point cloud classification,” *Neurocomputing*, vol. 505, pp. 58–67, 2022.
- [59] Y. Zhao, L. Bai, and X. Huang, “Fidnet: Lidar point cloud semantic segmentation with fully interpolation decoding,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021, pp. 4453–4458.
- [60] D. Kochanov, F. K. Nejadasl, and O. Booij, “Kprnet: Improving projection-based lidar semantic segmentation,” *arXiv preprint arXiv:2007.12668*, 2020.
- [61] H.-X. Cheng, X.-F. Han, and G.-Q. Xiao, “Cenet: Toward concise and efficient lidar semantic segmentation for autonomous driving,” in *IEEE International Conference on Multimedia and Expo*, 2022, pp. 01–06.
- [62] Q. Hu, B. Yang, S. Khalid, W. Xiao, N. Trigoni, and A. Markham, “Towards semantic segmentation of urban-scale 3d point clouds: A

- dataset, benchmarks and challenges,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4977–4987.
- [63] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” in *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [64] T. Cortinhal, G. Tzlepis, and E. Erdal Aksoy, “Salsanext: Fast, uncertainty-aware semantic segmentation of lidar point clouds,” in *Advances in Visual Computing*, 2020, pp. 207–222.
- [65] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4413–4421.
- [66] R. Razani, R. Cheng, E. Taghavi, and L. Bingbing, “Lite-hdseg: Lidar semantic segmentation using lite harmonic dense convolutions,” in *IEEE International Conference on Robotics and Automation*, 2021, pp. 9550–9556.
- [67] O. Unal, D. Dai, and L. Van Gool, “Scribble-supervised lidar semantic segmentation,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 2697–2707.
- [68] Y. Pan, B. Gao, J. Mei, S. Geng, C. Li, and H. Zhao, “Semanticposse: A point cloud dataset with large quantity of dynamic instances,” in *IEEE Intelligent Vehicles Symposium*, 2020, pp. 687–693.
- [69] Y. Hou, X. Zhu, Y. Ma, C. C. Loy, and Y. Li, “Point-to-voxel knowledge distillation for lidar semantic segmentation,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8479–8488.
- [70] X. Wu, Y. Lao, L. Jiang, X. Liu, and H. Zhao, “Point transformer v2: Grouped vector attention and partition-based pooling,” in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 33 330–33 342.
- [71] X. Yan, J. Gao, C. Zheng, C. Zheng, R. Zhang, S. Cui, and Z. Li, “2dpass: 2d priors assisted semantic segmentation on lidar point clouds,” in *European Conference on Computer Vision*, 2022, pp. 677–695.
- [72] X. Lai, Y. Chen, F. Lu, J. Liu, and J. Jia, “Spherical transformer for lidar-based 3d recognition,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 17 545–17 555.
- [73] X. Wu, Y. Hou, X. Huang, B. Lin, T. He, X. Zhu, Y. Ma, B. Wu, H. Liu, D. Cai *et al.*, “Taseg: Temporal aggregation network for lidar semantic segmentation,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 15 311–15 320.
- [74] L. Li, H. P. Shum, and T. P. Breckon, “Less is more: Reducing task and model complexity for 3d point cloud semantic segmentation,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 9361–9371.
- [75] L. Reichardt, N. Ebert, and O. Wasenmüller, “360deg from a single camera: A few-shot approach for lidar segmentation,” in *IEEE/CVF International Conference on Computer Vision Workshops*, 2023, pp. 1075–1083.
- [76] L. Kong, Y. Liu, X. Li, R. Chen, W. Zhang, J. Ren, L. Pan, K. Chen, and Z. Liu, “Robo3d: Towards robust and reliable 3d perception against corruptions,” in *IEEE/CVF International Conference on Computer Vision*, 2023, pp. 19 994–20 006.
- [77] M. Contributors, “MMDetection3D: OpenMMLab next-generation platform for general 3D object detection,” <https://github.com/open-mmlab/mmdetection3d>, 2020.
- [78] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” in *International Conference on Learning Representations*, 2019.
- [79] L. N. Smith and N. Topin, “Super-convergence: Very fast training of neural networks using large learning rates,” in *Artificial intelligence and machine learning for multi-domain operations applications*, vol. 11006, 2019, pp. 369–386.
- [80] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.
- [81] L. Kong, S. Xie, H. Hu, L. X. Ng, B. Cottreau, and W. T. Ooi, “Robodepth: Robust out-of-distribution depth estimation under corruptions,” in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 21 298–21 342.
- [82] Q. Huang, X. Dong, D. Chen, H. Zhou, W. Zhang, K. Zhang, G. Hua, Y. Cheng, and N. Yu, “Pointcat: Contrastive adversarial training for robust point cloud recognition,” *IEEE Transactions on Image Processing*, vol. 33, pp. 2183–2196, 2024.
- [83] X. Wu, Z. Tian, X. Wen, B. Peng, X. Liu, K. Yu, and H. Zhao, “Towards large-scale 3d representation learning with multi-dataset point prompt training,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 19 551–19 562.
- [84] S. Peng, K. Genova, C. Jiang, A. Tagliasacchi, M. Pollefeys, T. Funkhouser *et al.*, “Opencene: 3d scene understanding with open vocabularies,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 815–824.
- [85] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “Pointnet: Deep learning on point sets for 3d classification and segmentation,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017, pp. 652–660.
- [86] I. Alonso, L. Riazuelo, L. Montesano, and A. C. Murillo, “3d-mininet: Learning a 2d representation from point clouds for fast and efficient 3d lidar semantic segmentation,” *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5432–5439, 2020.
- [87] F. Zhang, J. Fang, B. Wah, and P. Torr, “Deep fusionnet for point cloud semantic segmentation,” in *European Conference on Computer Vision*, 2020, pp. 644–663.
- [88] J. Park, C. Kim, S. Kim, and K. Jo, “Pcsnnet: Fast 3d semantic segmentation of lidar point cloud for autonomous car using point convolution and sparse convolution network,” *Expert Systems with Applications*, vol. 212, p. 118815, 2023.
- [89] M. Gerdzhev, R. Razani, E. Taghavi, and L. Bingbing, “Tornadonet: multiview total variation semantic segmentation with diamond inception module,” in *IEEE International Conference on Robotics and Automation*, 2021, pp. 9543–9549.
- [90] H. Qiu, B. Yu, and D. Tao, “Gfnet: Geometric flow network for 3d point cloud semantic segmentation,” *arXiv preprint arXiv:2207.02605*, 2022.
- [91] X. Yan, J. Gao, J. Li, R. Zhang, Z. Li, R. Huang, and S. Cui, “Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion,” in *AAAI Conference on Artificial Intelligence*, vol. 35, no. 4, 2021, pp. 3101–3109.
- [92] Y. Gu, Y. Huang, C. Xu, and H. Kong, “Maskrange: A mask-classification model for range-view based lidar segmentation,” *arXiv preprint arXiv:2206.12073*, 2022.
- [93] Y. Su, L. Jiang, and J. Cao, “Point cloud semantic segmentation using multi scale sparse convolution neural network.” *arXiv preprint arXiv:2205.01550*, 2022.
- [94] R. Cheng, R. Razani, E. Taghavi, E. Li, and B. Liu, “Af2-s3net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 547–12 556.
- [95] J. Li, H. Dai, and Y. Ding, “Self-distillation for robust lidar semantic segmentation in autonomous driving,” in *European Conference on Computer Vision*, 2022, pp. 659–676.
- [96] M. Ye, R. Wan, S. Xu, T. Cao, and Q. Chen, “Efficient point cloud segmentation with geometry-aware sparse networks,” in *European Conference on Computer Vision*, 2022, pp. 196–212.
- [97] Z. Zhuang, R. Li, K. Jia, Q. Wang, Y. Li, and M. Tan, “Perception-aware multi-sensor fusion for 3d lidar semantic segmentation,” in *IEEE/CVF International Conference on Computer Vision*, 2021, pp. 16 280–16 290.
- [98] K. Genova, X. Yin, A. Kundu, C. Pantofaru, F. Cole, A. Sud, B. Brewington, B. Shucker, and T. Funkhouser, “Learning 3d semantic segmentation with only 2d image supervision,” in *International Conference on 3D Vision*, 2021, pp. 361–372.
- [99] D. Ye, Z. Zhou, W. Chen, Y. Xie, Y. Wang, P. Wang, and H. Foroosh, “Lidarmultinet: Towards a unified multi-task network for lidar perception,” in *AAAI Conference on Artificial Intelligence*, vol. 37, no. 3, 2023, pp. 3231–3240.
- [100] S. Li, X. Chen, Y. Liu, D. Dai, C. Stachniss, and J. Gall, “Multi-scale interaction for real-time lidar data segmentation on an embedded platform,” *Robotics and Automation Letters*, vol. 7, no. 2, pp. 738–745, 2021.