2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Improving Radial Imbalances with Hybrid Voxelization and RadialMix for LiDAR 3D Semantic Segmentation

Jiale Li[1], Hang Dai[2*], Yu Wang[3], Guangzhi Cao[3], Chun Luo[3], Yong Ding[1*]

***Abstract*— Huge progress has been made in LiDAR 3D semantic segmentation, but there are two under-explored imbalances on the radial axis: points are unevenly concentrated on the near side, and the distribution of foreground object instances is skewed to the near side. This leads the training of the model to favor semantics at the near side with the majority of points and object instances. Both the cylindrical and the spherical voxelizations aim to address the problem of imbalanced point distribution by increasing the volume of voxels along the radial distance to include fewer near-side points in a smaller voxel and more far-side points in a bigger voxel. However, this causes a problem of the receptive field enlarging along the radial distance, which is not desirable in LiDAR 3D segmentation. This can be addressed in cubic voxelization which has a fixed volume of voxels. Thus, we propose a new LiDAR 3D semantic segmentation network (Hi-VoxelNet) with Hybrid Voxelization that leverages the advantages of cubic, cylindrical, and spherical voxelizations for hybrid voxel feature learning. To address the radial imbalance of object instances, we propose a novel data augmentation technique termed as RadialMix that uses radial sample duplication to increase the number of distant foreground object instances and mixes the radial duplication with another point cloud for enriching the training samples. With the joint improvements of the radial imbalances, our method archives state-of-the-art performance on nuScenes and SemanticKITTI datasets, and it shows significant improvements along the radial axis. Our code is publicly available at https://github.com/jialeli1/lidarseg3d.**

I. INTRODUCTION

Three-dimensional (3D) semantic segmentation from LiDAR point cloud predicts point-wise semantic classes, which is the key to fine-grained scene understanding for autonomous driving [1], [2]. The disorder and sparsity of large-scale point cloud are addressed by sparse convolution on 3D voxels [3], [4], [5], [6] and regular convolution on 2D projection images [7], [8], [9], [10]. Beyond that, we focus on two under-explored imbalances of the point and foreground object instance distributions on the radial axis.

The 3D perception on the far side is important for safe and trustworthy autonomous driving, but the two radial imbalances impair the performance of LiDAR segmentation models on distant objects. As shown in Fig. 1, we can find that: **i**) there is a distinct imbalance in the point distribution along the radial axis, with a large number of points unevenly concentrated on the near side, resulting in diverse near-side object patterns for model training; **ii**) the distribution of foreground object instances is skewed toward the near side, which impairs the training of models on distant objects. Foreground objects are defined as *things* by the datasets [1], [2]. Thus, the number of points and object instances is extremely imbalanced between the near side and the far side on the radial axis. This leads the training of the model to favor semantics at the near side with the majority of points and object instances. In this paper, we propose to address the two radial imbalances with hybrid voxelization and RadialMix for LiDAR 3D semantic segmentation.

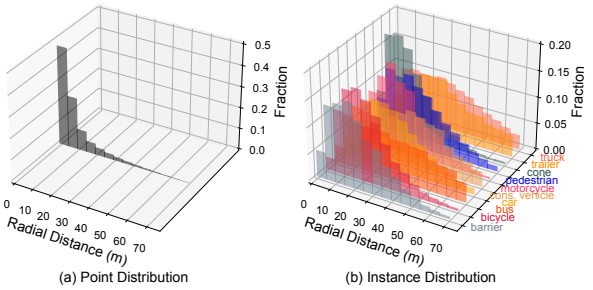

Fig. 1. Radial imbalances in point distribution (a) and foreground object instance distribution (b), calculated from the training set of the widely used nuScenes dataset [1]. The radial distance is the L2 distance $\sqrt{(x^2+y^2)}$ between a point (x, y, z) and the LiDAR in the bird's eye view (BEV).

To address the radial imbalance in point distribution, we propose a joint learning approach that incorporates hybrid voxel representations for voxel feature learning with significant improvements. Cubic voxelization divides 3D space into consistent-sized in the Cartesian coordinate system [6], while Cylinder3D [5] voxelizes LiDAR point cloud in the cylindrical coordinate system to fit the unbalanced point distribution. Since the distribution of LiDAR laser beam is close to a spherical model, the spherical partition is widely used in 2D projection methods [11], [9], [7] but not in 3D voxel-based methods. The cubic, cylindrical, and spherical voxelizations have advantages and disadvantages in LiDAR 3D semantic segmentation. As shown in Fig. 2 (b) and (c), the cylindrical and the spherical voxelizations can alleviate the imbalanced point distribution in voxelization process by placing smaller voxels on the near side to include fewer near-side points and larger voxels on the far side to include more far-side points in one voxel. With smaller voxels on the denser near-side points and larger voxels on the sparser far-side points, cylindrical and spherical voxelizations achieve a higher theoretical overall mIoU than cubic voxelization, as shown in Fig. 2 (d). A higher mIoU upper bound reflects less information loss in the point cloud representation and boosts the performance of the downstream model [12].

However, both cylindrical and spherical voxelizations have perspective discrepancies in voxel grids due to the variation of voxel volumes. The physical volume of cylindrical and spherical voxels increases with the radial distance, so an

[1]Zhejiang University. [2]University of Glasgow. [3]YUNJI Technology.
[*]{Hang.Dai@glasgow.ac.uk, dingyong09@zju.edu.cn}

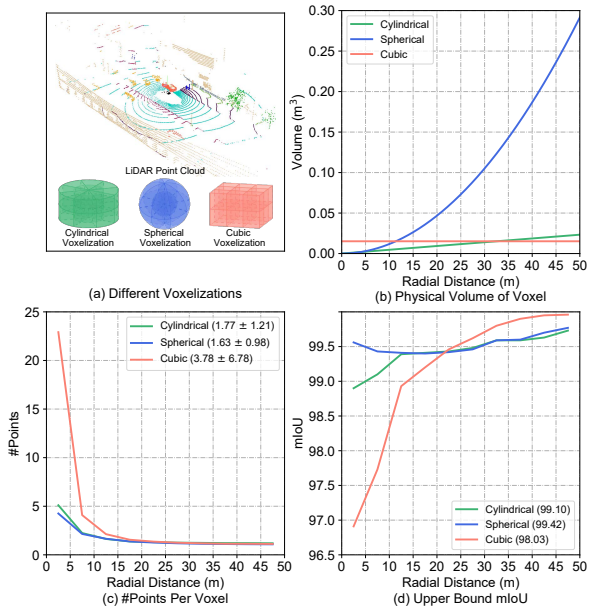

Fig. 2. Cylindrical, spherical, and cubic voxelizations on nuScenes [1] with the same voxel resolution $D \times W \times H$ of $480 \times 360 \times 32$ [5], [12]. Theoretical upper bound mIoU in (d) is evaluated between the ground truth and the intermediate point labels from the inverse mapping of voxel labels.

object occupies more voxels on the near side than the far side. Thus, the receptive field for the same object is smaller on the near side and larger on the far side in the cylindrical and spherical voxelizations. But the physical size of one object should have no changes on the near or far side. It is challenging to adapt to this pattern in segmentation networks [13], [14], [15], which impairs the training of networks. Since the cylindrical voxels that are placed at different radial distances have the same height, cylindrical voxelization has fewer perspective discrepancies than spherical voxelization. Cubic voxelization has two advantages: **i**) The cubic voxel volume is distance-independent in Fig. 2 (b) to provide a consistent receptive field pattern, which can reduce the difficulty of voxel feature learning. **ii**) On the far side, the theoretical mIoU upper bound is higher than that in cylindrical and spherical voxelizations [5], [6], [16] as shown in Fig. 2 (d). Overall, we propose a hybrid voxelization to leverage the advantages of different voxel representations.

To alleviate the skewed distribution of foreground object instances on the radial axis as shown in Fig. 1. (b), we use data augmentation to increase the number of distant object instances in point cloud training samples. Mix3D [17] or PolarMix [18] mixes all objects or a copy of globally rotated foreground objects from another point cloud to the current point cloud, which increases not only the distant objects but also the near objects. Neither Mix3D nor PolarMix addresses the radial imbalance in foreground object distribution. Thus, we propose a new data augmentation technique termed as RadialMix. RadialMix first augments a point cloud by selectively coping the near foreground objects with dense points, sparsifying the virtual variants, and pasting them at far locations in radial translation, thereby introducing plausible new distant foreground objects. The sparsification operation is to simulate the number of points changing in the radial distance. RadialMix then follows [17], [18] to mix a copy of all the foreground objects from the augmented point cloud sample to the current point cloud training sample. The enriched point cloud samples improve the training of LiDAR 3D semantic segmentation models.

Our **contributions** can be summarized in three folds: **i**) We systematically investigate the under-explored radial distribution imbalances on points and foreground object instances, which inspire us to improve LiDAR 3D semantic segmentation on the radial axis. **ii**) We propose to alleviate the radial imbalance of point distribution with Hi-VoxelNet by joint learning on the hybrid voxel representations, which achieves state-of-the-art performance on nuScenes [1] and SemanticKITTI [2] datasets. **iii**) We propose to improve the radial imbalance of object instance distribution with a new data augmentation technique (RadialMix) to mix object instances along the radial axis. Extensive experiments show that RadialMix achieves consistent improvements.

II. RELATED WORK

LiDAR 3D Semantic Segmentation. As a direct way of point cloud semantic segmentation, point-based methods [19], [20], [21] learn point-wise features from unordered neighboring points, but suffer from time consumption, uneven sparsity, and plain performance on large-scale LiDAR point clouds. The 2D projection-based methods [22], [11], [23], [9], [10], [24] project the point cloud into 2D representations in range view (RV) [7], [25] or bird's eye view (BEV) [12], [26], [27]. The highly optimized 2D convolution and off-the-shelf 2D networks help to achieve better computational efficiency and higher performance. The recent top-performing 3D voxel-based methods [6], [5], [16], [28] introduce sparse 3D convolution [3], [29] to perform feature extraction on non-empty voxels. Unlike typical cubic voxelization in SPVNAS [6], Cylinder3D [5] divides the point cloud into cylindrical voxels. Due to relatively uniform point partition and better performance, cylindrical voxelization draws a lot of research attention [30], [31], [32].

Integrating the additional features from points and 2D projection images with voxel features usually improves LiDAR point cloud feature learning [6], [33], [30]. SPVNAS [6] builds point and voxel branches, where the sparse 3D convolutions are on cubic voxels for faster neighborhood feature extraction [34]. RPVNet [33] improves SPVNAS by adding an RV branch, and 2DPASS [35] distills additional 2D semantic priors from camera images to the point cloud features learned by SPVNAS. AMVNet [24] learns from RV and BEV projections, limited by the lack of 3D structure.

LiDAR Data Augmentation for 3D Segmentation. Besides the conventional LiDAR point cloud augmentations such as random flipping and rotation, Mix3D [17] and PolarMix [12] explore stronger augmentations by mixing two point clouds, inspired by the mixup-based image augmentations [36], [37], [38]. Mix3D [17] mixes two randomly selected point clouds. PolarMix [12] mixes a globally rotated copy of foreground objects from another point cloud to the current point cloud, thereby avoiding the introduction of

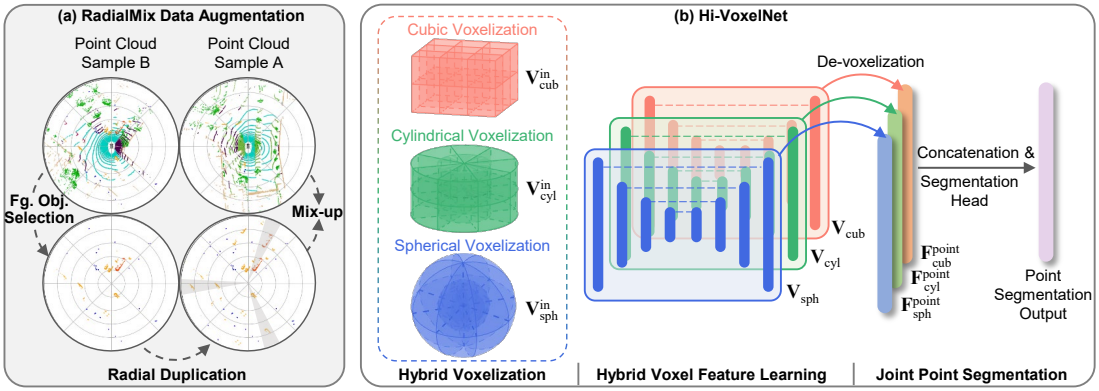

Fig. 3. Overview of the proposed method: RadialMix data augmentation (a) and Hi-VoxelNet (b). "Fg. Obj." denotes "Foreground Object".

massive easy background points and showing more improvements. Although the globally rotated copies of foreground objects approximate the LiDAR sweeping mechanism, PolarMix does not improve the skewed distribution of objects. The recent methods [39], [40] mix partial laser beams of two point clouds, leaving the radial imbalance untreated, but can be used as a pre-augmentation to our RadialMix.

III. METHOD

A. Hi-VoxelNet Network

Hybrid Voxelization. Let $\mathbf{P}_{\text{cub}} = \left\{\left(x_i, y_i, z_i, f_i^{\text{p-in}}\right) : i = 1, \cdots, N_{\text{P}}\right\}$ denote an input point cloud of N_{P} points for cubic voxelization in the Cartesian coordinate system, where (x, y, z) represents the 3D point coordinates, and $f^{\text{p-in}} \in \mathbb{R}^{C_{\text{in}} \times 1}$ represents the C_{in}-dimensional point-wise input features such as coordinates and reflection intensity. The cubic voxelization partitions $\mathbf{P}_{\text{cub}}$ as the structural non-empty cubic voxels located at v within a spatial shape $D \times W \times H$ as

$$v = (\lfloor \frac{x - X_{\text{min}}}{d_x} \rfloor, \lfloor \frac{y - Y_{\text{min}}}{d_y} \rfloor, \lfloor \frac{z - Z_{\text{min}}}{d_z} \rfloor), \quad (1)$$

$$d = (\frac{X_{\text{max}} - X_{\text{min}}}{D}, \frac{Y_{\text{max}} - Y_{\text{min}}}{W}, \frac{Z_{\text{max}} - Z_{\text{min}}}{H}), \quad (2)$$

where the voxel size d is determined by the point cloud range of $[[X_{\text{min}}, X_{\text{max}}], [Y_{\text{min}}, Y_{\text{max}}], [Z_{\text{min}}, Z_{\text{max}}]]$ and spatial shape. The points with the same value of v are gathered into a voxel and their point-wise input features are averaged as the voxel-wise input features $f^{\text{v-in}}$ [6]. The non-empty voxels located at unique coordinates v with the features $f^{\text{v-in}}$ constitute $\mathbf{V}_{\text{cub}}^{\text{in}} = \left\{\left(v_i, f_i^{\text{v-in}}\right) : i = 1, \cdots, N_{\text{V}}\right\}$, which serves as the input sparse tensor to point cloud backbone.

For cylindrical and spherical voxelizations, the slight difference lies in the transformation across coordinate systems. We define $\mathbf{P}_{\text{cyl}} = \left\{\left(r_i, \theta_i, z_i, f_i^{\text{p-in}}\right) : i = 1, \cdots, N_{\text{P}}\right\}$ and $\mathbf{P}_{\text{sph}} = \left\{\left(\rho_i, \theta_i, \phi_i, f_i^{\text{p-in}}\right) : i = 1, \cdots, N_{\text{P}}\right\}$ as the point clouds in the cylindrical and spherical coordinate systems, respectively. The coordinate transformations from the Cartesian coordinate system can be denoted as

$$(r, \theta, z) = (\sqrt{x^2 + y^2}, arctan(\frac{y}{x}), z), \quad (3)$$

$$(\rho, \theta, \phi) = (\sqrt{r^2 + z^2}, arctan(\frac{y}{x}), arctan(\frac{z}{r})). \quad (4)$$

We follow Eq. 1 and Eq. 2 for the voxelization from $(\mathbf{P}_{\text{cyl}}, \mathbf{P}_{\text{sph}})$ to $(\mathbf{V}_{\text{cyl}}^{\text{in}}, \mathbf{V}_{\text{sph}}^{\text{in}})$, using the specific cylindrical coordinates (r, θ, z) in the point cloud range of $([R_{\text{min}}, R_{\text{max}}], [\Theta_{\text{min}}, \Theta_{\text{max}}], [Z_{\text{min}}, Z_{\text{max}}])$, and the spherical coordinates (ρ, θ, ϕ) in the point cloud range of $([P_{\text{min}}, P_{\text{max}}], [\Theta_{\text{min}}, \Theta_{\text{max}}], [\Phi_{\text{min}}, \Phi_{\text{max}}])$, with the same spatial shape $D \times W \times H$. The hybrid voxelization employs three different types of voxelization and provides three voxel sparse tensors $\mathbf{V}_{\text{sph}}^{\text{in}}$, $\mathbf{V}_{\text{cyl}}^{\text{in}}$ and $\mathbf{V}_{\text{cub}}^{\text{in}}$ for the subsequent voxel feature learning on parallel branches. Notably, N_{V} is different among $\mathbf{V}_{\text{sph}}^{\text{in}}$, $\mathbf{V}_{\text{cyl}}^{\text{in}}$ and $\mathbf{V}_{\text{cub}}^{\text{in}}$, since different types of voxelization have different non-empty voxel numbers.

Hybrid Voxel Feature Learning. As shown in Fig. 3, we propose Hi-VoxelNet with three parallel branches of voxel feature learning backbone for extracting hybrid voxel features from the spherical, cylindrical, and cubic voxels. Instead of using three heavier Cylinder3D [5] backbones, we employ a lightweight counterpart sparse 3D U-Net from SDSeg3D [16], [41], which stacks four down-sampling convolutional blocks with increasing receptive fields for more informative voxel-wise features, and four up-sampling convolutional blocks. Each convolutional block is also accelerated by the sparse 3D convolution [3], [4], where computations are on the non-empty voxels for efficiency. Let α be a subscript and $\alpha \in \{\text{sph}, \text{cyl}, \text{cub}\}$. After passing sparse 3D U-Net, each $\mathbf{V}_{\alpha}^{\text{in}}$ becomes the $\mathbf{V}_{\alpha} = \left\{\left(v_i, f_i^{\text{voxel}}\right) : i = 1, \cdots, N_{\text{V}}\right\}$, with increased feature dimensions from C_{in} to C and unchanged voxel coordinates v. The corresponding voxel features can be denoted as $\mathbf{F}_{\alpha}^{\text{voxel}} = [f_1^{\text{voxel}}, \cdots, f_{N_{\text{V}}}^{\text{voxel}}] \in \mathbb{R}^{C \times N_{\text{V}}}$. Note that the voxel feature learning is compatible and extensible with other voxel-based networks.

Joint Point Segmentation. Since the LiDAR 3D semantic segmentation is a point-wise segmentation task, we devoxlize the hybrid voxel features $\mathbf{F}_{\alpha}^{\text{voxel}}$ at point-wise. In each voxel branch, we follow [16], [6], [42] to first perform devoxelization for mapping the voxel features $\mathbf{F}_{\alpha}^{\text{voxel}} \in \mathbb{R}^{C \times N_{\text{V}}}$ back to point features $\mathbf{F}_{\alpha}^{\text{point}} \in \mathbb{R}^{C \times N_{\text{P}}}$. Given a target point at (x, y, z), its cubic point features $f_{\text{cub}}^{\text{point}}$ are interpolated from the three nearest neighboring cubic voxels in $\mathbf{V}_{\text{cub}}$. We similarly interpolate its cylindrical point features $f_{\text{cyl}}^{\text{point}}$ from $\mathbf{V}_{\text{cyl}}$ and spherical point features $f_{\text{sph}}^{\text{point}}$ from $\mathbf{V}_{\text{sph}}$, respectively. The final point features are the concatenation of $(f_{\text{cub}}^{\text{point}}, f_{\text{cyl}}^{\text{point}}, f_{\text{sph}}^{\text{point}})$. Intuitively, hybrid voxelization has

the potential to provide complementary clues about the surroundings. As shown in Fig. 3, we feed the concatenated point features $\mathbf{F}^{\text{point}} \in \mathbb{R}^{3C \times N_{\text{P}}}$ from three branches to a Multi-Layer Perceptron (MLP) based segmentation head for joint point segmentation prediction. Beyond that, more refined fusion deserves to be explored.

Loss Function. Following [5], [12], [16], [28], we combine cross-entropy loss $\mathcal{L}_{\text{ce}}$ and lovasz loss [43] $\mathcal{L}_{\text{lovasz}}$ as the segmentation loss $\mathcal{L}_{\text{seg}} = \mathcal{L}_{\text{ce}} + \mathcal{L}_{\text{lovasz}}$. In addition to a top-level loss $\mathcal{L}^{\text{point}}$ on point segmentation prediction, we add three auxiliary voxel segmentation heads on the hybrid voxel features for auxiliary supervision [5], [16]. Therefore, the joint optimization is driven by a total loss $\mathcal{L}^{\text{total}} = \mathcal{L}^{\text{point}} + \mathcal{L}^{\text{voxel}}_{\text{sph}} + \mathcal{L}^{\text{voxel}}_{\text{cyl}} + \mathcal{L}^{\text{voxel}}_{\text{cub}}$, where each loss term is instantiated by $\mathcal{L}_{\text{seg}}$.

B. RadialMix Data Augmentation

RadialMix aims to generate augmented training samples of point cloud for facilitating the LiDAR 3D semantic segmentation model training, with improved foreground object instance distribution on the radial axis. Let ($\mathbf{P}_{\text{A}}$, $\mathbf{P}_{\text{B}}$) denote two point cloud samples A and B in the Cartesian coordinate system, along with their respective semantic segmentation labels ($\mathbf{S}_{\text{A}}$, $\mathbf{S}_{\text{B}}$) and object instance labels ($\mathbf{I}_{\text{A}}$, $\mathbf{I}_{\text{B}}$). The datasets [1], [2] provide $\mathbf{S}$ and $\mathbf{I}$ to annotate each point in $\mathbf{P}$ separately with the semantic class and the object instance ID to which it belongs. As shown in Fig. 3, RadialMix generates an augmented point cloud in three steps: foreground object selection, radial duplication, and mix-up.

Foreground Object Selection. We instance-wisely select out M foreground objects. Given a class list $\mathbf{C}$ containing the defined foreground classes, we mask all the foreground points $\mathbf{P}^{\text{fg}}_{\text{B}}$ with the associated segmentation labels $\mathbf{S}^{\text{fg}}_{\text{B}}$, and instance labels $\mathbf{I}^{\text{fg}}_{\text{B}}$ belonging to the foreground objects from point cloud $\mathbf{P}_{\text{B}}$. The ($\mathbf{P}^{\text{fg}}_{\text{B}}$, $\mathbf{S}^{\text{fg}}_{\text{B}}$) are then partitioned into a foreground object instance set $\Omega = \{o_i : i = 1, \cdots, M\}$ according to the unique instance IDs in $\mathbf{I}^{\text{fg}}_{\text{B}}$. Each object instance o contains the paired points and segmentation labels.

Radial Duplication. We expand the foreground object instance set Ω with some virtual objects copied from the near side of the radial axis. Specifically, for each object instance $o \in \Omega$ located within the radial distance R_{m}, we copy the object instance and paste its virtual variant to a more distant radial location, which is randomly chosen from $[R_{\text{m}}, R_{\text{max}}]$. To generate more realistic local points for the virtual variant, we sparsify the pasted virtual variant along with the radial distance to simulate the condition that distant objects have sparser points than near objects.

The sparsification operation simulates the limited measurement resolution $W_0 \times H_0$ of LiDAR, where W_0 and H_0 denote horizontal and vertical resolutions for θ-axis and ϕ-axis in a spherical coordinate system (Eq. 4). We follow Eq. 4 to project the points of the virtual variant onto a clean θ-ϕ plane in a spherical manner. Since multiple points can be projected inside the same resolution bin b computed as Eq. 5, we only keep the nearest point for the purpose of sparsification. Note that we sparsify the paired segmentation labels synchronously along with the points. Then, the virtual variant is appended into Ω. Overall, the foreground object instance set Ω can be effectively expanded and the foreground object instance distribution becomes more balanced along the radial axis.

$$b = \left(\frac{(\theta - \Theta_{\text{min}})W_0}{\Theta_{\text{max}} - \Theta_{\text{min}}}, \frac{(\phi - \Phi_{\text{min}})H_0}{\Phi_{\text{max}} - \Phi_{\text{min}}}\right). \tag{5}$$

Mix-up. The final step is to mix the foreground object instance set Ω into the point cloud $\mathbf{P}_{\text{A}}$ to create a new training sample. For all instances in Ω, the points, and paired segmentation labels are concatenated together to form $\mathbf{P}_{\Delta}$ and $\mathbf{S}_{\Delta}$, respectively. We concatenate $\mathbf{P}_{\Delta}$ and $\mathbf{P}_{\text{A}}$, and the corresponding segmentation labels $\mathbf{S}_{\Delta}$ and $\mathbf{S}_{\text{A}}$:

$$(\tilde{\mathbf{P}}, \tilde{\mathbf{S}}) = (\mathbf{P}_{\text{A}} \copyright \mathbf{P}_{\Delta}, \mathbf{S}_{\text{A}} \copyright \mathbf{S}_{\Delta}), \tag{6}$$

where © is concatenation. The generated point cloud $\tilde{\mathbf{P}}$ and segmentation labels $\tilde{\mathbf{S}}$ can be used in the standard training pipeline of LiDAR 3D semantic segmentation models.

Discussion on RadialMix. We compare our RadialMix with PolarMix [18]. We provide the visualizations of PolarMix and RadialMix. The main differences lie in the duplication of foreground objects. **i**) PolarMix rotates a copy of the foreground objects. Although the PolarMix duplication ensures that the surface of the object close to LiDAR is partially visible, both the far and near objects are increased simultaneously, which does not alleviate the imbalance of the foreground object instance distribution on the radial axis. Instead, RadialMix selectively increases distant objects with sparse points copied from the near. The sparsification operation also allows a reasonable simulation of LiDAR visibility pattern. **ii**) RadialMix duplication potentially avoids collisions among foreground objects, compared with random global rotation in PolarMix duplication. The far region behind the near objects is usually empty because of the perspective visibility in the LiDAR sensor. **iii**) RadialMix instance-wisely selects the positions for object instance pasting, which is more flexible and powerful in augmented point cloud generation. Experimental results show that RadialMix achieves more significant improvements than PolarMix.

IV. EXPERIMENT

A. Experimental Setup

nuScenes Dataset [1] is a widely used public dataset for autonomous driving with Velodyne-HDL32E LiDAR. It officially splits 28,130 point clouds for training, 6,019 point clouds for validation, and 6,008 point clouds for testing. There are 16 valid semantic classes for evaluation [44].

SemanticKITTI Dataset [2] collected in Germany by Velodyne-HDL64E LiDAR has 19 valid classes for evaluation. It is split into the training sequences 00 to 10 (excluding 08) with 19,130 point clouds, validation sequence 08 with 4,071 point clouds, and testing sequences 11 to 21 with 20,351 point clouds [2], [5], [12].

Evaluation Metric is the mean Intersection-over-Union (mIoU) defined as $\frac{1}{C}\sum_{c=1}^{N_{\text{cls}}} \frac{TP_c}{TP_c + FP_c + FN_c}$, where TP_c, FP_c, FN_c denote the number of true positive, false positive,

TABLE I

COMPARISON OF VOXELIZATION SETTINGS ON NUSCENES VALIDATION SET. MIOU* (%) AND #PARAMS INDICATE THE SEGMENTATION PERFORMANCE OF THE OFFLINE MODEL ENSEMBLE AND THE NUMBER OF MODEL PARAMETERS, RESPECTIVELY. THE LAST ROW INDICATES A MODEL VARIANT CONSISTING OF THREE CUBIC BRANCHES WITH $\times 0.75$, $\times 1$, AND $\times 1.25$ SCALING TO THE DEFAULT SPATIAL SHAPE.

Voxelization			mIoU	mIoU*	#Params (M)	Latency (S)	Mem (GB)
Cylindrical	Spherical	Cubic					
√	-	-	67.27	-	21.28	0.099	1.75
-	√	-	67.76	-	21.28	0.120	1.77
-	-	√	66.79	-	21.28	0.095	1.74
√	√	-	70.45	69.22	42.57	0.170	1.98
√	-	√	70.93	69.42	42.57	0.155	1.93
-	√	√	71.53	69.63	42.57	0.160	1.94
√	√	√	72.64	70.51	63.86	0.236	2.16
-	-	[×0.75, ×1, ×1.25]	69.84	68.97	63.86	0.278	2.20

TABLE II

COMPARISON OF DATA AUGMENTATION ACROSS 2D PROJECTION BASED POLARNET[12], CYLINDRICAL VOXELIZATION BASED CYLINDER3D [5], AND HYBRID VOXELIZATION BASED HI-VOXELNET. THE HI-VOXELNET HERE IS SET AS THE BEST-PERFORMING MODEL ("CYLINDRICAL + SPHERICAL + CUBIC") IN TAB. I.

	PolarNet [12]	Cylinder3D [5]	Hi-VoxelNet (Ours)
Baseline	60.05	68.00	72.64
+Mix3D [17]	60.37	68.94	73.48
+PolarMix [18]	61.17	69.41	73.98
+RadialMix (w.o. sparsification)	61.31	69.79	74.23
+RadialMix	62.89	70.54	74.90
#Params (M)	13.64	55.70	63.86
Latency (S)	0.080	0.131	0.236
Mem (GB)	2.47	2.10	2.16

TABLE III

PERFORMANCE BREAKDOWN ON MIOU (%) OVER RADIAL DISTANCES, EVALUATED ON VALIDATION SETS OF DIFFERENT LIDAR 3D SEMANTIC SEGMENTATION DATASETS. THE "IMPROV." DENOTES IMPROVEMENTS OVER THE CYLINDRICAL BASELINE MODEL IN TAB. I, WHICH CAN SERVE AS A LIGHTWEIGHT VARIANT OF THE POPULAR CYLINDER3D THAT ACHIEVES 76.1 MIOU ON NUSCENES VALIDATION SET [5].

Dataset	Component	Overall		[0m,10m)		[10m,20m)		[20m,30m)		[30m,40m)		[40m,50m)	
		mIoU	Improv.	mIoU	Improv.	mIoU	Improv.	mIoU	Improv.	mIoU	Improv.	mIoU	Improv.
nuScenes [1]	Cylindrical Baseline	75.02	-	79.02	-	75.87	-	59.44	-	40.97	-	23.10	-
	+Hybrid Voxelization	77.94	+2.92	80.50	+1.48	78.89	+3.02	66.29	+6.85	50.07	+9.10	33.76	+10.66
	++RadialMix	79.02	+4.00	82.41	+3.39	79.45	+3.58	66.47	+7.03	50.33	+9.36	36.18	+13.08
SemanticKITTI [2]	Cylindrical Baseline	64.99	-	65.53	-	65.11	-	59.14	-	48.48	-	41.09	-
	+Hybrid Voxelization	67.74	+2.75	68.60	+3.07	68.41	+3.30	62.05	+2.91	53.17	+4.69	44.51	+3.42
	++RadialMix	71.13	+6.14	72.05	+6.52	70.65	+5.54	65.32	+6.18	55.85	+7.37	47.27	+6.18

and false negative predictions for the c-th class out of N_{cls} valid classes [1], [2]. Instead of taking the mean of the IoUs across all the classes, nuScenes leaderboard returns another metric fwIoU where each IoU is weighted by the point frequency of its class.

B. Results

Hybrid Voxelization is investigated in Tab. I as follows:

i) The first three rows show that the spherical voxelization outperforms the cylindrical and cubic voxelizations in LiDAR 3D semantic segmentation with a single-branch network. This lies in a more balanced point partition and a higher upper bound of theoretical overall mIoU from spherical voxelization as illustrated in Fig. 2. Different voxelizations that result in different numbers of non-empty voxels have different inference latencies. And cubic voxelization achieves the fastest inference time.

ii) The middle three rows show that the combination of two voxelizations is better than single voxelization. Although cylindrical voxelization is better than cubic voxelization, "spherical + cylindrical" voxelization is worse than "spherical + cubic" voxelization. This lies in the fact that cylindrical and spherical voxelizations have similarities in both the voxel volume variations and the theoretical mIoU curves in Fig. 2.

iii) Our hybrid voxelization achieves the best segmentation performance as shown in the penultimate row of Tab. I.

iv) We further compare the multi-scale homogeneous voxelization to our hybrid voxelization, considering that hybrid voxelization also produces voxels of different volumes. We select cubic voxelization that has distance-independent voxel volume. From the last two rows, we compare the models in equal model capacity. The multi-scale homogeneous voxelization achieves substantially worse performance than the hybrid voxelization with more inference latency. This indicates that hybrid voxelization derives a more informative and efficient voxel feature learning than multi-scale homogeneous voxelization with the same model capacity.

Joint Learning v.s. Offline Ensemble. Across all the last five rows of Tab. I, mIoU is consistently larger than mIoU* achieved by offline ensemble of multiple independently trained single-branch models, which implies that the joint learning manner exploits the learnable capacity of the neural network, and adaptively fuse the different types of voxel features from hybrid voxelization.

RadialMix. Tab. II validates that the mixup-based augmentation techniques can improve the performance of LiDAR 3D semantic segmentation networks. Mix3D [17] is inferior compared with PolarMix [18] and RadialMix. This implies that the excessive object overlaps impair the semantic information while mixing additional foreground points is a more effective data augmentation strategy in LiDAR 3D semantic segmentation. Although the vanilla version of RadialMix without (w.o.) sparsification already outperforms PolarMix, the full version of RadialMix with the sparsification operation achieves further performance gains by generating more reasonable distant objects. The consistent improvements in different network architectures verify the generalization and robustness of RadialMix.

Complexity Comparison is objectively discussed in Tab. II and Tab. I. The complexity and segmentation performance of Hi-VoxelNet are both higher than PolarNet [12] and Cylinder3D [5] in Tab. II. The lighter variant of Hi-VoxelNet with "spherical + cubic" outperforms Cylinder3D with close inference latency but fewer model parameters and less GPU memory consumption. The lightest "cubic" variant of Hi-VoxelNet outperforms PolarNet with more model parameters but close inference latency and less GPU memory consumption. In the BEV projection-based PolarNet, the dense image-like tensor not only spends computational resources on the empty regions but also lacks the capability to explore the 3D structures of objects.

Radial Evaluation. We evaluate the segmentation performance with mIoU breakdown on radial distances for our two main components in Tab. III. Due to the decreasing

TABLE IV

SEMANTIC SEGMENTATION RESULTS OF IOU (%) AND FWIOU(%) ON NUSCENES TESTING SET. THE BEST RESULTS ARE IN BOLD. *FOREGROUND CLASS IN [44]. METHODS PUBLISHED BEFORE THE SUBMISSION DEADLINE (9/15/2023) ARE LISTED.

Modality	Method	**mIoU**	**fwIoU**	barrier*	bicycle*	bus*	car*	cons. vehicle*	motorcycle*	pedestrian*	cone*	trailer*	truck*	driveable	other	sidewalk	terrain	manmade	vegetation
Multi-sensor	PMF [45]	77.03	89.34	82.11	40.33	80.94	86.42	63.72	79.22	79.75	75.86	81.17	67.05	97.28	67.69	78.05	74.48	89.94	88.46
	2D3DNet [46]	79.96	90.08	83.01	**59.35**	87.99	85.09	63.70	84.39	**81.95**	75.96	84.79	71.93	96.88	67.35	79.81	75.96	**92.05**	**89.18**
	2DPASS [35]	80.80	90.10	81.70	55.30	92.00	91.80	73.30	**86.50**	78.50	72.50	84.70	75.50	97.60	69.10	79.90	75.50	90.20	88.00
LiDAR-only	PolarNet [12]	69.42	87.38	72.16	16.81	77.01	86.53	51.14	69.65	64.80	54.11	69.70	63.53	96.64	67.14	77.70	72.13	87.13	84.47
	JS3C-Net [47]	73.60	88.06	80.14	26.15	87.79	84.54	55.17	72.56	71.28	66.26	76.79	71.11	96.80	64.47	76.86	74.09	87.48	86.10
	Cylinder3D [5]	77.16	89.92	82.76	29.75	84.34	89.41	63.03	79.29	77.21	73.40	84.55	69.17	97.66	70.24	80.29	75.51	90.41	87.55
	AMVNet [24]	77.27	90.08	80.64	31.96	81.73	88.93	67.07	84.33	76.11	73.48	84.87	67.30	97.37	67.37	79.41	75.45	91.45	88.69
	SPVNAS [6]	77.35	89.68	80.00	29.98	91.92	90.81	64.68	78.99	75.62	70.94	81.01	74.64	97.44	69.23	79.95	76.10	89.28	87.06
	Cylinder3D++ [5]	77.86	89.93	82.76	33.89	84.34	89.41	69.63	79.42	77.26	73.40	84.55	69.41	97.66	70.24	80.29	75.51	90.42	87.55
	AF2S3Net [28]	78.34	88.51	78.87	52.21	89.93	84.17	**77.42**	74.30	77.32	71.95	83.88	73.78	97.13	66.47	77.51	74.01	87.69	86.80
	Ours	**80.97**	**90.63**	**87.08**	37.93	**92.72**	**92.26**	71.78	84.23	80.82	**78.08**	**86.13**	**77.73**	**97.86**	**71.89**	**81.63**	**77.43**	90.52	87.38

TABLE V

SEMANTIC SEGMENTATION RESULTS OF IOU (%) ON SEMANTICKITTI TESTING SET. THE BEST AND SECOND-BEST RESULTS ARE IN BOLD AND UNDERLINED. *FOREGROUND CLASS IN [2]. METHODS PUBLISHED BEFORE THE SUBMISSION DEADLINE (9/15/2023) ARE LISTED.

Point Cloud Representation	Methods	**mIoU**	car*	bicycle*	motorcycle*	truck*	other-vehicle*	person*	bicyclist*	motorcyclist*	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic-sign
Point	PointASNL [20]	46.8	87.4	0.0	25.1	39.0	29.2	34.2	57.6	0.0	87.4	24.3	74.3	1.8	83.1	43.9	84.1	52.2	70.6	57.8	36.9
	RandLA-Net [19]	53.9	94.2	26.0	25.8	40.1	38.9	49.2	48.2	7.2	90.7	60.3	73.7	20.4	86.9	56.3	81.4	61.3	66.8	49.2	47.7
	KPConv [48]	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	90.5	64.2	84.8	69.2	69.1	56.4	47.4
	BAAF-Net [49]	59.9	95.4	31.8	35.5	48.7	46.7	49.5	55.7	53.0	90.9	62.2	74.4	23.6	89.8	60.8	82.7	63.4	67.9	53.7	52.0
2D Projection	RangeNet++ [7]	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 [12]	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
	SqueezeSegv3 [9]	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
	M2SKD-PolarNet [50]	58.0	93.5	45.9	36.3	27.6	34.9	55.0	51.4	15.8	91.1	64.7	73.8	26.1	92.5	67.0	84.6	63.4	67.4	50.7	59.5
	Knowledge3DtoBEV [27]	61.1	95.2	53.5	45.9	41.8	42.0	60.8	54.1	29.8	91.8	63.6	75.8	24.8	90.5	61.6	82.1	65.2	66.3	54.4	61.3
	KPRNet [23]	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
	RangeViT [25]	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 [10]	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
3D Voxel	MinkNet42 [51]	54.3	94.3	23.1	26.2	26.1	36.7	43.1	36.4	7.9	91.1	63.8	69.7	29.3	92.7	57.1	83.7	68.4	64.7	57.3	60.1
	3D-MiniNet [52]	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
	FusionNet [53]	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
	SPVNAS [6]	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
	Cylinder3D [5]	67.8	97.1	67.6	64.0	59.0	58.6	73.9	67.9	36.0	91.4	65.1	75.5	32.3	91.0	66.5	85.4	71.8	68.5	62.6	65.6
	RPVNet [33]	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 [16]	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**
	PVKD [30]	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
	2DPASS [35]	72.9	97.0	63.6	63.4	**61.1**	61.5	**77.9**	**81.3**	**74.1**	89.7	67.4	74.7	40.0	**93.5**	**72.9**	86.2	73.9	71.0	65.0	70.4
	Ours	**73.5**	97.5	**71.2**	**69.3**	59.3	**68.6**	77.0	76.2	60.3	92.5	72.7	79.2	36.3	**93.5**	72.8	**86.7**	74.0	**72.0**	**66.9**	70.2

number of points, the LiDAR 3D semantic segmentation performance degrades with the increasing distance [45], [12], [35]. However, our hybrid voxelization and RadialMix show significant improvements across radial distances, especially in the more challenging regions beyond thirty meters.

Results on nuScenes. In Tab. IV, our method achieves the highest number (12/18) of Top-1 results against other methods on nuScenes testing set [1]. Our method significantly improves the segmentation on most foreground objects (e.g., barrier, cone, trailer, and truck). Although the performances on some foreground objects (e.g., pedestrian and motorcycle) are not the best, the gaps are slight. As a LiDAR-only method, our method still outperforms the methods that require multi-sensor modalities [45], [46], [35].

Results on SemanticKITTI. We also evaluate our method on the SemanticKITTI testing set [2]. From Tab. V, the voxel-based methods usually outperform the point-based and the 2D projection-based methods, which implies that the voxel representation is promising in high-quality LiDAR 3D semantic segmentation. With the proposed hybrid voxelization and RadialMix data augmentation, our method outperforms other voxel-based methods, including the RPVNet [33] that combines three LiDAR point cloud representations of point, 2D projection, and 3D voxel. Our method achieves the best mIoU, 8/20 Top-1, and 13/20 Top-2 results in Tab. V. The state-of-the-art results on both nuScenes and SemanticKITTI validate the effectiveness of our method.

V. CONCLUSION

We propose a new LiDAR 3D semantic segmentation method with Hi-VoxelNet and RadialMix addressing two under-explored distribution imbalances on the radial axis. The proposed Hi-VoxelNet improves the imbalanced point distribution on the radial axis with hybrid voxelization. The proposed RadialMix alleviates the skewed distribution of instance objects on the radial distance via foreground object selection, radial duplication, and mix-up. The method achieves state-of-the-art performance on nuScenes and SemanticKITTI. The comprehensive ablation studies validate that Hi-VoxelNet and RadialMix are effective in LiDAR 3D semantic segmentation. The problem in the trade-off between performance and complexity/latency can be eased with the light version of Hi-VoxelNet.

REFERENCES

[1] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A multimodal dataset for autonomous driving," in *CVPR*, 2020, pp. 11 618–11 628.

[2] 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 *ICCV*, 2019, pp. 9296–9306.

[3] B. Graham, M. Engelcke, and L. van der Maaten, "3D semantic segmentation with submanifold sparse convolutional networks," in *CVPR*, 2018, pp. 9224–9232.

[4] Y. Yan, Y. Mao, and B. Li, "SECOND: sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, 2018.

[5] 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 *CVPR*, 2021, pp. 9939–9948.

[6] 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 *ECCV*, vol. 12373, 2020, pp. 685–702.

[7] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet++: Fast and accurate LiDAR semantic segmentation," in *IROS*, 2019, pp. 4213–4220.

[8] Z. Zhong, Z. Q. Lin, R. Bidart, X. Hu, I. B. Daya, Z. Li, W. Zheng, J. Li, and A. Wong, "Squeeze-and-attention networks for semantic segmentation," in *CVPR*, 2020, pp. 13 062–13 071.

[9] 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 *ECCV*, 2020, pp. 1–19.

[10] H. Cheng, X. Han, and G. Xiao, "CENet: Toward concise and efficient LiDAR semantic segmentation for autonomous driving," in *ICME*, 2022, pp. 1–6.

[11] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a LiDAR point cloud," in *ICRA*, 2019, pp. 4376–4382.

[12] Y. Zhang, Z. Zhou, P. David, X. Yue, Z. Xi, B. Gong, and H. Foroosh, "PolarNet: An improved grid representation for online LiDAR point clouds semantic segmentation," in *CVPR*, 2020, pp. 9598–9607.

[13] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *CVPR*, 2017, pp. 6230–6239.

[14] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu, and B. Xiao, "Deep high-resolution representation learning for visual recognition," *IEEE TPAMI*, vol. 43, no. 10, pp. 3349–3364, 2021.

[15] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "UNet++: Redesigning skip connections to exploit multiscale features in image segmentation," *IEEE Trans. Medical Imaging*, vol. 39, no. 6, pp. 1856–1867, 2020.

[16] J. Li, H. Dai, and Y. Ding, "Self-distillation for robust LiDAR semantic segmentation in autonomous driving," in *ECCV*, vol. 13688, 2022, pp. 659–676.

[17] A. Nekrasov, J. Schult, O. Litany, B. Leibe, and F. Engelmann, "Mix3D: Out-of-context data augmentation for 3D scenes," in *3DV*, 2021, pp. 116–125.

[18] A. Xiao, J. Huang, D. Guan, K. Cui, S. Lu, and L. Shao, "PolarMix: A general data augmentation technique for LiDAR point clouds," in *NeurIPS*, 2022.

[19] 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 *CVPR*, 2020, pp. 11 105–11 114.

[20] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui, "PointASNL: Robust point clouds processing using nonlocal neural networks with adaptive sampling," in *CVPR*, 2020, pp. 5588–5597.

[21] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *NeurIPS*, 2017, pp. 5099–5108.

[22] 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 *ICRA*, 2018, pp. 1887–1893.

[23] D. Kochanov, F. K. Nejadasl, and O. Booij, "KPRNet: Improving projection-based LiDAR semantic segmentation," in *ECCV*, 2020.

[24] 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," *CoRR*, vol. abs/2012.04934, 2020.

[25] 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 *CVPR*, 2023, pp. 5240–5250.

[26] Z. Zhou, Y. Zhang, and H. Foroosh, "Panoptic-PolarNet: Proposal-free LiDAR point cloud panoptic segmentation," in *CVPR*, 2021, pp. 13 194–13 203.

[27] F. Jiang, H. Gao, S. Qiu, H. Zhang, R. Wan, and J. Pu, "Knowledge distillation from 3D to bird's-eye-view for LiDAR semantic segmentation," in *ICME*, 2023, pp. 402–407.

[28] R. Cheng, R. Razani, E. Taghavi, E. Li, and B. Liu, "(AF)2-S3Net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network," in *CVPR*, 2021, pp. 12 547–12 556.

[29] B. Graham, "Sparse 3D convolutional neural networks," in *BMVC*, 2015, pp. 150.1–150.9.

[30] Y. Hou, X. Zhu, Y. Ma, C. C. Loy, and Y. Li, "Point-to-voxel knowledge distillation for LiDAR semantic segmentation," in *CVPR*, 2022, pp. 8469–8478.

[31] M. Liu, Y. Zhou, C. R. Qi, B. Gong, H. Su, and D. Anguelov, "LESS: label-efficient semantic segmentation for LiDAR point clouds," in *ECCV*, vol. 13699, 2022, pp. 70–89.

[32] F. Hong, H. Zhou, X. Zhu, H. Li, and Z. Liu, "LiDAR-based panoptic segmentation via dynamic shifting network," in *CVPR*, 2021, pp. 13 090–13 099.

[33] J. Xu, R. Zhang, J. Dou, Y. Zhu, J. Sun, and S. Pu, "RPVNet: A deep and efficient range-point-voxel fusion network for LiDAR point cloud segmentation," in *ICCV*, 2021, pp. 16 004–16 013.

[34] J. Park, C. Kim, S. Kim, and K. Jo, "PCSCNet: Fast 3D semantic segmentation of LiDAR point cloud for autonomous car using point convolution and sparse convolution network," *Expert Systems with Applications*, vol. 212, pp. 118 815–118 824, 2023.

[35] Y. Xu, J. Gao, C. Zheng, C. Zheng, R. Zhang, S. Cui, and Z. Li, "2DPASS: 2D priors assisted semantic segmentation on LiDAR point clouds," in *ECCV*, vol. 13688, 2022, pp. 677–695.

[36] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "MixUp: Beyond empirical risk minimization," in *ICLR*, 2018.

[37] S. Yun, D. Han, S. Chun, S. J. Oh, Y. Yoo, and J. Choe, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *ICCV*, 2019, pp. 6022–6031.

[38] G. Ghiasi, Y. Cui, A. Srinivas, R. Qian, T. Lin, E. D. Cubuk, Q. V. Le, and B. Zoph, "Simple copy-paste is a strong data augmentation method for instance segmentation," in *CVPR*, 2021, pp. 2918–2928.

[39] L. Kong, J. Ren, L. Pan, and Z. Liu, "LaserMix for semi-supervised LiDAR semantic segmentation," in *CVPR*, 2023, pp. 21 706–21 716.

[40] 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 *ICCV*, 2023, pp. 228–240.

[41] L. Contributors, "LiDARSeg3D: A generic repository for LiDAR 3D semantic segmentation in autonomous driving scenarios." https://github.com/jialeli1/lidarseg3d, 2023.

[42] J. Li, H. Dai, L. Shao, and Y. Ding, "From voxel to point: IoU-guided 3D object detection for point cloud with voxel-to-point decoder," in *ACM MM*, 2021, pp. 4622–4631.

[43] 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 *CVPR*, 2018, pp. 4413–4421.

[44] 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 Autom. Lett.*, vol. 7, no. 2, pp. 3795–3802, 2022.

[45] Z. Zhuang, R. Li, K. Jia, Q. Wang, Y. Li, and M. Tan, "Perception-aware multi-sensor fusion for 3D LiDAR semantic segmentation," in *ICCV*, 2021, pp. 16 260–16 270.

[46] K. Genova, X. Yin, A. Kundu, C. Pantofaru, F. Cole, A. Sud, B. Brewington, B. Shucker, and T. A. Funkhouser, "Learning 3D semantic segmentation with only 2D image supervision," in *3DV*. IEEE, 2021, pp. 361–372.

[47] 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*, 2021, pp. 3101–3109.

[48] H. Thomas, C. R. Qi, J. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *ICCV*, 2019, pp. 6410–6419.

[49] S. Qiu, S. Anwar, and N. Barnes, "Semantic segmentation for real point cloud scenes via bilateral augmentation and adaptive fusion," in *CVPR*, 2021, pp. 1757–1767.

[50] S. Qiu, F. Jiang, H. Zhang, X. Xue, and J. Pu, "Multi-to-single knowledge distillation for point cloud semantic segmentation," in *ICRA*, 2023, pp. 9303–9309.
[51] C. B. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal ConvNets: Minkowski convolutional neural networks," in *CVPR*, 2019, pp. 3075–3084.
[52] 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 Autom. Lett.*, vol. 5, no. 4, pp. 5432–5439, 2020.
[53] F. Zhang, J. Fang, B. W. Wah, and P. H. S. Torr, "Deep fusionnet for point cloud semantic segmentation," in *ECCV*, vol. 12369, 2020, pp. 644–663.