DRINet: A Dual-Representation Iterative Learning Network for Point Cloud Segmentation

Maosheng Ye[1‡*] Shuangjie Xu[2*] Tongyi Cao[2] Qifeng Chen[1]
[1]Hong Kong University of Science and Technology [2]DEEPROUTE.AI
myeag@connect.ust.hk {shuangjiexu, tongyicao}@deeproute.ai cqf@ust.hk

Abstract

We present a novel and flexible architecture for point cloud segmentation with dual-representation iterative learning. In point cloud processing, different representations have their own pros and cons. Thus, finding suitable ways to represent point cloud data structure while keeping its own internal physical property such as permutation and scale-invariant is a fundamental problem. Therefore, we propose our work, DRINet, which serves as the basic network structure for dual-representation learning with great flexibility at feature transferring and less computation cost, especially for large-scale point clouds. DRINet mainly consists of two modules called Sparse Point-Voxel Feature Extraction and Sparse Voxel-Point Feature Extraction. By utilizing these two modules iteratively, features can be propagated between two different representations. We further propose a novel multi-scale pooling layer for pointwise locality learning to improve context information propagation. Our network achieves state-of-the-art results for point cloud classification and segmentation tasks on several datasets while maintaining high runtime efficiency. For large-scale outdoor scenarios, our method outperforms state-of-the-art methods with a real-time inference time of $62ms$ per frame.

1. Introduction

Point cloud data plays a significant role in various real-world applications, from autonomous driving to augmented reality (AR). One of the critical tasks in point cloud understanding is point cloud semantic segmentation, which can facilitate self-driving cars or AR applications to interact with the physical world. For real-world applications, an accurate and real-time point cloud segmentation method is highly desirable. Therefore, in this work, we will study a new framework for high-quality point cloud segmentation in real-time.

[‡]Part of the work was done during an internship at DEEPROUTE.AI.
[*]Equal contributions.

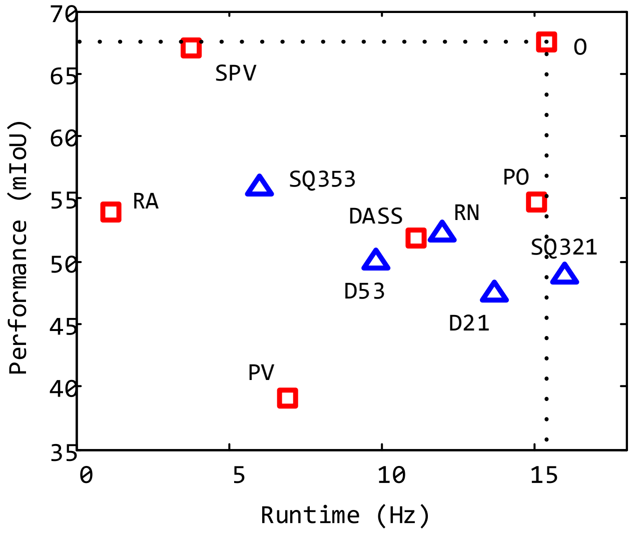

Figure 1. The mIoU performance vs. speed on the SemanticKITTI test set. Projection methods are drawn as blue triangles and other kinds of methods are drawn as red rectangles. Methods close to the right top location mean that achieving better performance within less runtime cost. Drawn methods are RA: RandLA [13], PV: PVCNN [23], SPV: Sparse PVCNN [31], DASS [34] PO: PolarNet [45], D53: Darknet53 [2], D21: Darknet21 [2], RN: RangeNet++ [24], SQ321: SqueezeSegV3-21 [37], SQ353: SqueezeSegV3-53 [37], O: our DRINet. Our DRINet outperforms all the existing methods while maintaining high runtime efficiency at 15Hz.

Although we have witnessed great progress in vision tasks on 2D images with convolutional neural networks (CNN), point cloud processing with deep learning still faces lots of challenges. Due to its sparsity and irregularity, it is difficult to directly apply 2D CNNs or some other popular operations in image processing for point cloud data. PointNet [26] is a pioneering work that directly operates on raw point clouds. PointNet++ [27] extends the PointNet by aggregating local features at different scales of neighborhoods to capture more context information and fine geometry structures. Further, VoxelNet [50] firstly combines learning-based point cloud feature extraction with a stan-

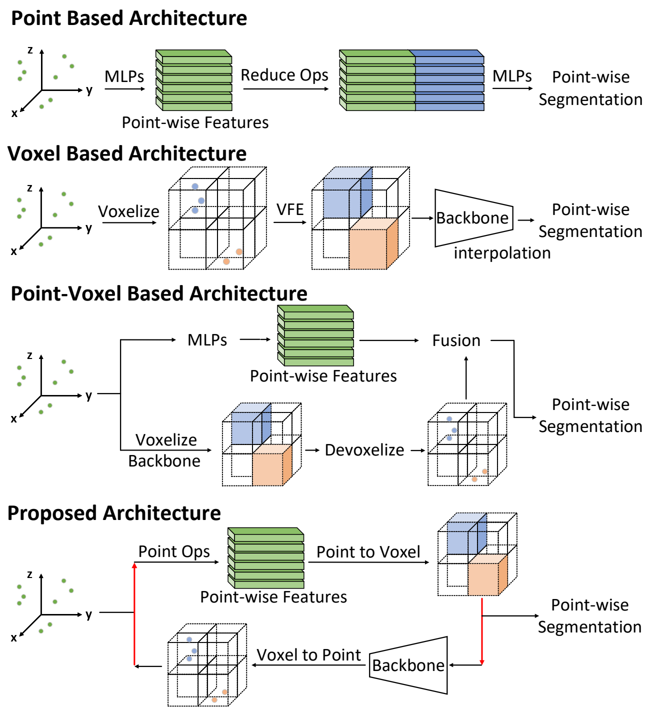

Figure 2. Three common structures of the 3D semantic segmentation task (Point Based Architecture, Voxel Based Architecture, Point-Voxel Based Architecture), and the difference compared with our proposed architecture. Notice the arrow direction of red lines that represents that dual-branches are integrated iteratively.

dard CNN structure. However, these works cannot achieve the balance between efficiency and performance, especially in large-scale outdoor scenarios where the point number in a point cloud is large. PointNet [26] or PointNet++ [27] requires a lot of memory usage and computational cost. A key hyper-parameter in VoxelNet [50] is the voxel scale: a small voxel scale brings better performance but along with less runtime efficiency. Recently, Liu et al. proposed PVCNN [23] based on a dual representation that combines the merits of 3D CNN and PointNet [26]. It deeply fuses the voxelwise and pointwise features. Also, SPVNet [31] applies a similar idea by searching architecture while replacing 3D CNN layers with a 3D sparse CNN to achieve less memory cost and better computational efficiency. Both works are aiming to obtain a better feature for scene understanding tasks by dual-representation fusion. However, there are three main drawbacks of these two works. First, they only apply a simple fusion strategy and ignore the feature propagation among these representations, which can be mutually complementary. Second, they ignore some physical properties such as accurate measurement information of point clouds that can bring internal scale invariance. Third, they use bilinear or trilinear gathering operations to fetch pointwise features from voxel feature maps that can be a large overhead when dealing with large-scale point clouds.

Inspired by PVCNN [23], SPVNet [31] and considering these aspects, we propose the DRINet that serves as a better and novel framework for dual representations point cloud segmentation learning. Our DRINet has better flexibility in converting between dual representations, with which we can learn features iteratively between point and voxel representations (shown in the Fig. 2) by our proposed novel modules: Sparse Point-Voxel Feature Extraction (SPVFE) and Sparse Voxel-Point Feature Extraction (SVPFE). Each module takes the features of the other module as input. As such, we can preserve the fine details by pointwise features and explore more context information with large receptive fields by voxelwise features. Beyond these two modules, we explore multi-scale feature extraction and aggregation for pointwise feature learning in our SPVFE to maintain its locality for better context information. Furthermore, we replace the bilinear and trilinear gathering operations with an attentive gathering layer to reduce the computation cost of feature transformation from voxelwise features to pointwise features under the SVPFE module while maintaining the performance.

In summary, our contributions include

- We propose a novel network architecture for point cloud learning that can flexibly transform representations between pointwise and voxelwise features. Both pointwise and voxelwise features can be aggregated and propagated iteratively.
- A multi-scale pooling layer is proposed at the voxel level to efficiently extract multi-scale pointwise features to gain better context information of point clouds.
- We propose a novel attentive gathering layer to gain better pointwise features from voxel features at a low memory access cost.
- To demonstrate the effectiveness of our method, extensive experiments are conducted on both indoor and outdoor datasets including ModelNet [48], ShapeNet [48], S3DIS [1], and SemanticKITTI [2]. Compared with existing methods, our DRINet achieves the state-of-the-art performance on SemanticKITTI, one of the most challenging datasets for outdoor scene parsing, while running at a real-time speed of $62ms$ per frame on an Nvidia RTX 2080 Ti GPU.

2. Related Work

Data representation is a key component in point cloud related tasks, including 3D object detection and 3D semantic segmentation. Most existing works for point cloud processing can be roughly divided into the following four categories according to their representations.

Point based methods. Most point based works can be viewed as extensions of PointNet [26] and PointNet++ [27].

They usually use the farthest sampling to sample some key points to reduce computation costs when dealing with large-scale outdoor point clouds. Then a series of variant point convolution operations like [5, 38, 40, 46] are applied to points within the given neighborhood to extract global and local context information based on PointNet architecture. However, there are two main drawbacks to this kind of method. First, the performance of these works somehow is limited by the procedure of farthest sampling that is proposed to reduce the memory cost and increase runtime efficiency. Thus, KPConv [33] introduces a new learnable way to generate kernel points rather than farthest sampling, with better and more robust distributions to represent their local neighborhood properties. RandLA-Net [13] also proposes a random sampling strategy to improve the efficiency of point cloud pre-processing significantly. Secondly, most of these methods heavily rely on *K-nearest neighbor* search to maintain the local relationship among points per frame which involves *KD Tree* building whose worst time complexity is $\mathrm{O}\left(Kn\log n\right)$.

Projection networks. Currently a lot of works [7, 9, 19, 24, 37] project points to front view representations including depth image and spherical projections. With this representation whose data organization is regular and structural, a series of standard convolution layers and recent popular 2D segmentation backbone [12, 30, 47] can be directly applied to achieve the balance between efficiency and accuracy. For example, SqueezeSeg [37] uses spherical projections and improves the segmentation network by their SAC module. The final results are refined by the CRF process. RangeNet++ [24] uses a similar projection method with better post-processing algorithms. However, the performance of these methods is highly related to projection resolution and the complex post-processing stage that aims to smooth and refine the prediction results with extra computation cost.

Voxel based methods. BEV representation with regular cartesian coordinate is the most common and popular way in voxel based method. Most works in lidar detection and segmentation [18, 50] adopt this way to form 2D/3D birdeye view image features. One of the biggest advantages of this method is that it can maintain the physical properties of point clouds and apply standard convolution layers. Recently, PolarNet [45] introduces polar representation into deep learning, presenting point cloud as a ring based structure. A ring CNN is proposed to track the special data distribution properties. Since most outdoor point clouds are obtained from scanning, this representation can reduce the effect of the non-uniform distribution phenomenon compared with normal voxelization methods. Furtherly, Cylinder3D [49] extends the 2D polar to 3D polar voxels. Besides, Su *et al* [28] voxelized points into the high dimensional space lattice and apply bilateral convolutions to the occupied sectors of the lattice.

Multiview fusion based methods. MV3D [6] is the pioneering work that explored the potential of multiview features learning in 3D object detection. With PointNet, VoxelNet [50] and PVCNN [23] integrated point feature with 3D volumetric representation. 3D CNN and MLP were used for extracting coarse and fine-grained features respectively to achieve better performance and less memory cost. Besides, a lot of works [8, 16, 18, 35] have utilized point based methods for feature extraction at each single voxel rather than handcrafted feature. They all address the importance of representation integration and fusion.
In comparison with these methods above, our proposed method belongs to multiview representation learning. While taking advantage of voxel feature learning and point feature learning, our method greatly improves the segmentation performance at very high runtime efficiency.

3. Method

In this section, we introduce our DRINet that integrates the merits of point and voxel representations to improve point cloud segmentation performance while maintaining high computational efficiency. The overall network, as shown in Fig. 3, consists of four parts: 1) **Geometry-aware Feature Extraction** 2) **Sparse Voxel-Point Feature Extraction** 3) **Sparse Point-Voxel Feature Extraction** and 4) **Iterative Dual-Representation Learning**. The sparse point-voxel feature extraction layer takes pointwise features as input and outputs voxelwise features to form sparse voxel feature maps with more hierarchical information. Then the sparse voxel-point feature extraction layer takes voxelwise features as input to generate high-quality pointwise features. The two blocks can iteratively perform the conversion between different representations, namely **iterative dual-representation learning**.

3.1. GAFE: Geometry-aware Feature Extraction

Data Representation. A point cloud can be represented by an unordered point set $\{p_1, p_2, \ldots, p_N\}$ with $p_i \in \mathbb{R}^d$ that includes the point coordinate $c_i = (x_i, y_i, z_i)$ and associated point features such as intensity.

Voxelization. We introduce the voxelization process to construct a mapping relationship between the two representations. Define that the point cloud is discretized into numerous voxels with resolution of $L \times W \times H$ and N_V non-empty voxel numbers. Given a point p_i, we compute its voxel index v_i under the grid scale s:

$$v_i^s = \left(\lfloor x_i/s \rfloor, \lfloor y_i/s \rfloor, \lfloor z_i/s \rfloor\right), \tag{1}$$

where $\lfloor \cdot \rfloor$ is the floor function and s refers to the size of each voxel along xyz directions.

Scatter $\Phi^s_{\mathcal{P}\to\mathcal{V}}$ and **Gather** $\Phi^s_{\mathcal{V}\to\mathcal{P}}$. Now a mapping system for coordinate spaces between point p and voxel v has been built for indexing. We define two flexible operations **Scatter** $\Phi^s_{\mathcal{P}\to\mathcal{V}}$ and **Gather** $\Phi^s_{\mathcal{V}\to\mathcal{P}}$ to transform between

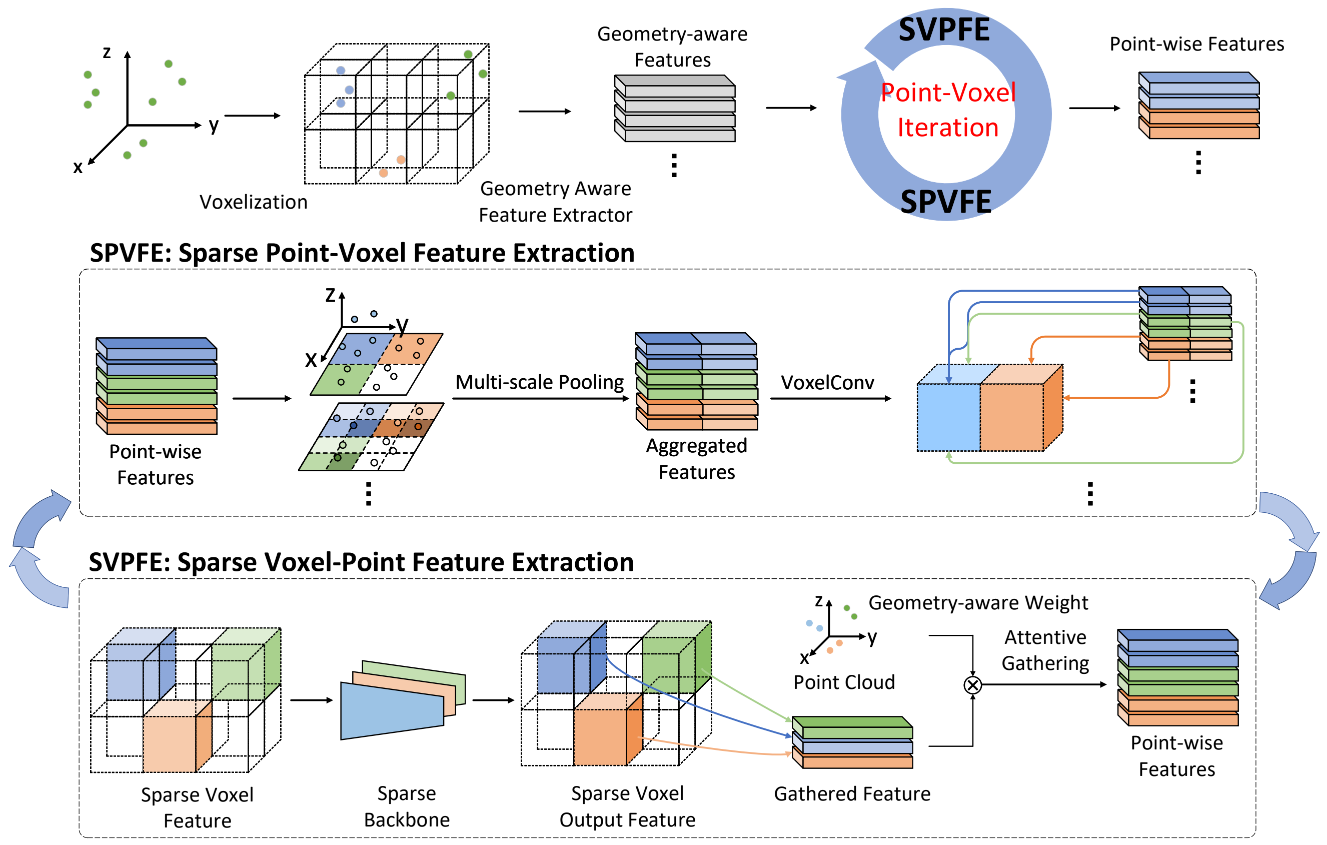

Figure 3. The first line is the whole network structure of DRINet. It includes two main modules, 1) Geometry-aware Feature Extraction, and 2) the Point and Voxel Branch. The second line describes the process of Point and Voxel Branch, consisting of Sparse Point-Voxel Feature Extraction *(SPVFE)* and Sparse Voxel-Point Feature Extraction *(SVPFE)*. a) *SVPFE* generates pointwise features with an attentive gathering layer from voxelwise features. b) *SPVFE* generates voxelwise features at target scale with a multi-scale pooling layer from pointwise features.

voxelwise features $\mathcal{V}^s$ and pointwise features $\mathcal{P}$ under the voxel scale s, where $\mathcal{V}^s \in R^{N_V \times C}$, $\mathcal{P} \in R^{N \times C}$, and C is the number of channels. For the **Scatter** operation, under the voxel scale s, the voxel feature at the voxel ζ is obtained by a boardcast operation Ψ on all points inside this voxel:

$$\mathcal{V}^s = \{\mathcal{V}^s_\zeta\} = \Phi^s_{\mathcal{P}\to\mathcal{V}}\left(\mathcal{P}\right), \quad \mathcal{V}^s_\zeta = \Psi\left(\{\mathcal{P}_i | v^s_i = \zeta\}\right), \quad (2)$$

where Ψ can be defined as the mean or max operation. In brief, $\Phi^s_{\mathcal{P}\to\mathcal{V}}$ can broadcast apply the same operation for all the input points within the same voxel. Meanwhile, we define an inverse operation **Gather**. The i-th pointwise feature with voxel ζ is gathered from the voxelwise features by a boardcast identity mapping operation (i.e., copying):

$$\mathcal{P} = \{\mathcal{P}_i\} = \Phi^s_{\mathcal{V}\to\mathcal{P}}\left(\mathcal{V}^s\right), \quad \mathcal{P}_i = \mathcal{V}^s_\zeta. \quad (3)$$

Geometry-aware Feature Extraction. Inspired by works [18, 42], we focus on fully utilizing the original point cloud geometric properties. The raw geometry-aware feature g^s_i for point p_i under a given grid size s is represented as

$$g^s_i = (c_i - \sum\nolimits_{c_j \in \mathcal{N}^s_i} c_j \Big/ |\mathcal{N}^s_i|) \oplus p_i \oplus (c_i - s \times v^s_i), \quad (4)$$

where $\oplus$ represents tensor concatenation. The neighbor collection $\mathcal{N}^s_i$, referred to the point coordinates set of points that lies in the same voxel as p_i, is denoted as $\mathcal{N}^s_i = \left\{c_j | v^s_j = v^s_i\right\}$. Let $G^s = \{g^s_i\}$ and then the final multi-scale feature G is

$$G = \sum_{s \in S} \text{MLP}(G^s) \oplus \Phi^s_{\mathcal{V}\to\mathcal{P}}\left(\Phi^s_{\mathcal{P}\to\mathcal{V}}\left(\text{MLP}(G^s)\right)\right), \quad (5)$$

where MLP represents a multilayer perceptron, S is the scale list. By simply fusing multi-scale pointwise features, we obtain the hybrid geometry-aware pointwise features G, which serve as the initial pointwise features F for SPVFE and SVPFE.

3.2. SPVFE: Sparse Point-Voxel Feature Extraction

In this part, we propose our novel Sparse Point-Voxel Feature Extraction module (SPVFE). By taking pointwise features, SPVFE provides a novel way **Multi-scale Pooling** for multi-scale pointwise features learning in point clouds with better efficiency that serves as better context information. Finally, it transforms the pointwise features into voxelwise features with our proposed **VoxelConv**.

Multi-scale Pooling Layer. To obtain better pointwise features, we propose a novel layer to explore more context information with great efficiency. Inspired by PointNet++ [27], PSPNet [47], HVNet [42], and Deeplab [4], we notice that multi-scale information is important in classification and segmentation tasks. To some extent, multi-scale information can aggregate more local context information at different scales with different receptive fields. Indeed, pyramid pooling and dilated convolution are straightforward ways in image-related tasks. However, applying these methods for sparse voxel feature maps will decrease efficiency and deteriorate the feature since many empty voxel features will be involved. Therefore, we propose a novel multi-scale pooling layer that utilizes simple MLP layers for point clouds followed by set abstraction at different scales by the scattering operation in Eq. 2. As shown in Alg. 1, for each scale s in the given scale list S, only points inside the same voxel will contribute to the output features. With the aggregation of pointwise features at different scales, there will be a stronger representation of multi-scale properties in point clouds. Compared with PointNet++ [27], our method does not rely on KNN that consists of a complex pre-processing procedure for building the KD Tree.

Algorithm 1 Multi-scale Pooling Algorithm

```
Require: Pointwise features $F$ and predefined scales $S$
1: $L = []$
2: for each $s \in S$ do
3:     $\mathcal{V}^s = \Phi^s_{\mathcal{P}\to\mathcal{V}}(F)$
4:     $F^s = \text{MLP}(\text{Concat}([F, \Phi^s_{\mathcal{V}\to\mathcal{P}}(\mathcal{V}^s)]))$
5:     $L.\text{append}(F^s)$
6: end for
7: return Concat$(L)$
```

VoxelConv Operation. With pointwise features from the Multi-scale Pooling Layer, we propose the VoxelConv operation to form the next-stage voxel feature map. Similar to graph convolutional networks [25], we deal with points in the discrete voxel space with local aggregation. VoxelConv in the voxel space with pointwise features to generate the voxelwise features can be defined as

$$\Phi^s_{\mathcal{P}\to\mathcal{V}}(WF), \qquad (6)$$

where W is a weight matrix. VoxelConv is applied to all the points within the same voxel, where we use fully connected layers with learnable weights W to each pointwise feature. Then the aggregation function in Eq. 2 is used to calculate the local response. The whole process is similar to graph convolutions [25] with learnable weights and maintains locality. VoxelConv generates sparse voxel features and skips empty voxel grids, saving a lot of computation costs. Finally, we can construct the feature map by assigning the voxel features to their corresponding locations.

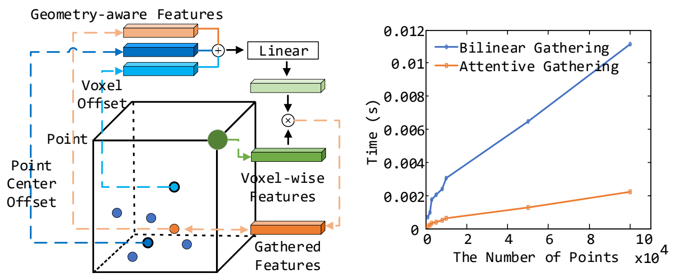

Figure 4. The left figure illustrates our attentive gathering. The right one shows the computational cost for the bilinear gathering and our attentive gathering.

3.3. SVPFE: Sparse Voxel-Point Feature Extraction

Taking 3D voxel features as input, there are several mature ways to exploit the local spatial correlation of the input in the 3D space. PVCNN [23] utilizes 3D CNN layers. However, it neglects the large-scale outdoor scenarios that contain hundreds of thousands of points per frame. As such, we propose a dual module, namely SVPFE module against SPVFE, with which the voxelwise features from SPVFE modules are fed into a 3D Sparse Voxel Learning block and then rollback to pointwise by an attentive gathering layer.

3D Sparse Voxel Learning. Most previous works compress the Z-axis information for fast feature extraction without large GPU memory usage but with an aggressive downsampling strategy for the sake of higher efficiency, leading to ineffectiveness in capturing small instances. Other works deploy 3D features but suffer the computation inefficiency. Inspired by sparse convolution [10, 41], we adopt 3D sparse convolution as backbone for voxel features extraction due to its high efficiency and ability to capture compact voxel features. We utilize a series of ResNet Bottleneck [11] by replacing 2D convolution with 3D sparse convolution. We name it Sparse Bottleneck.

Attentive Gathering Strategy. After Sparse Bottleneck, we need to map the voxelwise features to pointwise features. A nearest gathering operation that retrieves pointwise features from the voxelwise features is applied according to Eq. 3. Since the points in the same voxel share the same voxel feature, the nearest gathering will lead to inferior feature representation capability. Especially when the voxel scale increases, meaning that each voxel will contain more points, this phenomenon will become more severe. Previous works [23, 31] adopt bilinear or trilinear gathering operations when retrieving the pointwise features. However, the memory access cost for a large number of points cannot be ignored since it could not guarantee the memory coalescing that allows an optimal usage of the global memory bandwidth. There will be a great overload for the whole model once more gathering operations are introduced.

Thus, we propose a novel and effective approach with learnable parameters to increase the uniqueness and representation capability while maintaining the voxelwise features, as shown in Fig. 4. The traditional bilinear gathering

Algorithm 2 Dual-representation Learning Algorithm

```
Require: Point cloud $P$, #Iteration $N_I$, Scale list $S$
Ensure: Pointwise semantic prediction $O_p$
1: $F$ = []
2: $G \leftarrow$ GAFE$(P, S)$
3: $F_p \leftarrow G$
4: for iter $= 1$ to $N_I$ do
5:     $F_v \leftarrow$ SPVFE$(F_p)$
6:     $F_p \leftarrow$ SVPFE$(F_v)$
7:     $F$.append$(F_p)$
8: end for
9: $O_p \leftarrow$ SoftMax(MLP(Concat$(F)$))
```

can be expressed as weighted sum of neighborhood features according to distance. As a contrast, our method can be derived as follows:

$$F_{att} = W'G, \tag{7}$$

$$F_{out} = F \odot F_{att}. \tag{8}$$

where G is the hybrid geometry-aware features from Sec. 3.1, and W' is a weight matrix. F_{att} is geometry-aware weights. Then output features are obtained by elementwise multiplication of nearest gathering features F and above geometry-aware weights. Compared with bilinear gathering, our attentive gathering layer contains statistics, including mean and voxel information as geometric prior information with learnable parameters, which can be viewed as an attention mechanism.

3.4. Iterative Dual-Representation Learning

Most previous works only utilize single form of features such as pointwise [13], voxelwise [45], etc., or extract multi-representation in parallel. On the contrary, we propose a novel dual-representation learning algorithm based on an iterative process shown in Alg. 2. The hybrid geometry-aware features G from Sec. 3.1 are fed into the iteration process, composed with SVPFE module and SPVFE module. The SVPFE module contains sparse convolution layers that can learn sufficient intra-voxel features, while the SPVFE module is capable of capturing inner-voxel features with the geometry constraint for points inside the same voxel with multi-scale pooling for better locality and context extraction. Both SPVFE and SVPFE modules utilize the output of the other module. As such, pointwise features and voxelwise features propagate mutually and iteratively, forming a natural foundation for fusing context information across multiple representations.

4. Experiments

We conduct extensive experiments for the proposed DRINet on both indoor and outdoor tasks, including classification and segmentation, to show the effectiveness and generalization ability of our proposed method.

4.1. Outdoor Scene Segmentation

Dataset. We use the SemanticKITTI [2] dataset to verify the effectiveness of our network for large-scale outdoor scenarios. SemanticKITTI has a total of 43551 scans with imbalanced point level annotations for 20 categories. It contains 22 sequences which involve the most common scenes for autonomous driving. Besides, another challenging part of this dataset is that each scan contains more than 100K points on average, posing great pressure on lightweight model design. Following the official settings, we use the sequences from 00 to 10 except 08 as the training split, sequence 08 as validation split, and the sequences from 11 to 21 as the test split.

Experiment Details. Although the maximum distance for point clouds in SemanticKITTI can be more than 80m with a non-uniform density distribution, there are few points when sensing range beyond 55m. Based on this observation, we set voxelization scale ranging from minimum $[-48, -48, -3]$ to maximum $[48, 48, 1.8]$ for x, y, z respectively, with which we can include nearly 99% points with only 1% mIoU lost. For the points outside the ranges, we mask them to unknown types.

Network Details. Following the above principle, we design DRINet with three SPVFE and SVPFE blocks with the voxel scales at $[0.4m, 0.8m, 1.6m]$ respectively and Multi-scale Pooling Layer at $[0.4m, 0.8m, 1.6m, 3.2m]$. We also do some experiments varying the number of SPVFE and SVPFE blocks. In the loss design, we adopt the Lovasz loss [3] to alleviate the great imbalance distribution among different categories. During training, global rotation and random flip are applied for data augmentation. We train DRINet for 40 epochs with the Adam [15] optimizer with batch size of 4, the initial learning rate is $2e^{-4}$ with weight decay $1e^{-4}$. Besides, the learning rate decays with a ratio of 0.8 every 5 epochs.

Experimental Results. Detailed per-class quantitative results of DRINet and other state-of-the-art methods are shown in Tab. 1. DRINet achieves state-of-the-art performance among these methods in the mean IoU score. In some small classes, such as bicycle, person and so on, the DRINet shows a far bigger improvement. Moreover, we maintain a real-time inference time with the highest performance-time ratio, shown in Fig.1. We also provide some qualitative visual results on SemanticKITTI test set, as shown in Fig. 5.

4.2. Ablation Study

To analyze the effectiveness of different components in DRINet, we conduct the following ablation studies on the SemanticKITTI validation set.

Geometry-aware Feature Extractor. We firstly analyze our model with only geometry-aware feature extractor (GAFE), which is shown in the first line in Fig.3 with the iteration part removed. Compared with PointNet [26]

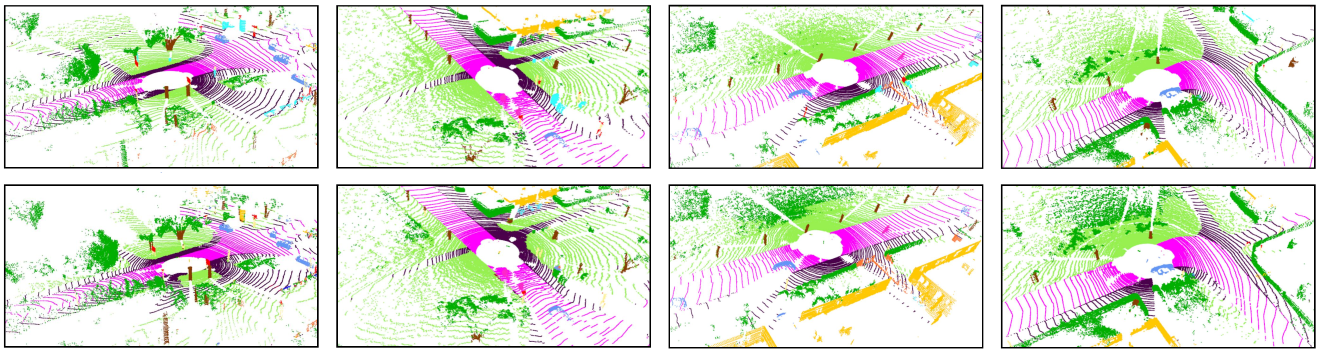
Figure 5. The results on SemanticKITTI. The top row is the ground truth, and the bottom row is the Predictions by DRINet.

Methods	road	sidewalk	parking	other ground	building	car	truck	bicycle	motorcycle	other vehicle	vegetation	trunk	terrain	person	bicyclist	motorcyclist	fence	pole	traffic sign	mIoU	speed (ms)
PointNet [26]	61.6	35.7	15.8	1.4	41.4	46.3	0.1	1.3	0.3	0.8	31.0	4.6	17.6	0.2	0.2	0.0	12.9	2.4	3.7	14.6	500
PointNet++ [27]	72.0	41.8	18.7	5.6	62.3	53.7	0.9	1.9	0.2	0.2	46.5	13.8	30.0	0.9	1.0	0.0	16.9	6.0	8.9	20.1	5900
KPConv [33]	88.8	72.7	61.3	**31.6**	90.5	96.0	33.4	30.2	42.5	44.3	84.8	69.2	69.1	61.5	61.6	11.8	64.2	56.4	47.4	58.8	-
SqueezeSegV3 [39]	91.7	74.8	63.4	26.4	89.0	92.5	29.6	38.7	36.5	33.0	82.0	58.7	65.4	45.6	46.2	20.1	59.4	49.6	58.9	55.9	238
TangentConv [32]	83.9	63.9	33.4	15.4	83.4	90.8	15.2	2.7	16.5	12.1	79.5	49.3	58.1	23.0	28.4	8.1	49.0	35.8	28.5	35.9	3000
SPVNet [31]	90.2	**75.4**	**67.6**	21.8	**91.6**	**97.2**	56.6	50.6	50.4	**58.0**	**86.1**	**73.4**	**71.0**	67.4	67.1	50.3	66.9	**64.3**	**67.3**	67.0	259
PolarNet [45]	90.8	74.4	61.7	21.7	90.0	93.8	22.9	40.3	30.1	28.5	84.0	65.5	67.8	43.2	40.2	5.6	61.3	51.8	57.5	54.3	**62**
RandLA [13]	90.7	73.7	60.2	20.4	86.9	94.2	40.1	26.0	25.8	38.9	81.4	66.8	49.2	49.2	48.2	7.2	56.3	47.7	38.1	53.9	880
RangeNet++ [24]	91.8	75.2	65.0	27.8	87.4	91.4	25.7	25.7	34.4	23.0	80.5	55.1	64.6	38.3	38.8	4.8	58.6	47.9	55.9	52.2	83.3
DASS [34]	**92.8**	71.0	31.7	0.0	82.1	91.4	**66.7**	25.8	31.0	43.8	83.5	56.6	69.6	47.7	70.8	0.0	39.1	45.5	35.1	51.8	90
DRINet(ours)	90.7	75.2	65.0	26.2	91.5	96.9	43.3	**57.0**	**56.0**	54.5	85.2	72.6	68.8	**69.4**	**75.1**	**58.9**	**67.3**	63.5	66.0	**67.5**	**62**

Table 1. The per-class mIoU results on the SemanticKITTI test set.

and PointNet++ [27], our GAFE is a stronger baseline with 22.8% mIoU on validation set which means our GAFE has better representations for the physical properties of the original data, as shown in Tab. 2.

	PointNet [26]	PointNet++ [27]	GAFE
mIoU(%)	15.3	18.1	**22.8**
Latency (s)	0.5	5.9	**0.022**

Table 2. Comparison with different feature extractors (GAFE).

Representation Analysis. The core component of our DRINet lies in dual-representation. We remove either representation to verify how the block influences the final results. As shown in Tab. 3, we set value of SPVFE number N_{spv} or SVPFE number N_{svp} to zero respectively to control the representation involved. With SPVFE and SVPFE off, there will be about 18.7% and 3.9% drop respectively. It illustrates that network with only voxelwise features performs better than that with only pointwise features. Moreover, by fusing dual representations, we can obtain better features.

Block Number Analysis. We note that block numbers N_{spv} and N_{svp} are crucial parameters for our network structure. From the left of Fig. 6, we can see the mIoU increasing in SemanticKITTI dataset (from 58.2% to 67.6%) and in S3DIS dataset (from 36.3% to 66.7%) as block number

SPVFE N_{spv}	SVPFE N_{svp}	mIoU(%)
3	0	48.6
0	3	63.4
3	3	67.3

Table 3. Ablation study of block numbers on SemanticKITTI.

increases. Nevertheless, this comes with more computation cost and larger memory usage. As shown in the right of Fig. 6, inference time increases from $62ms$ to $72ms$ when add N_{spv} and N_{svp} from 3 to 4, however the mIoU only improved from 67.3 to 67.6. According to the balance between performance and efficiency, we finally adopt $N_{spv} = 3$ and $N_{svp} = 3$ in following experiments.

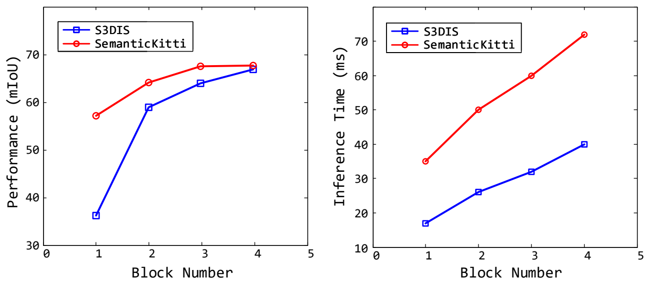

Figure 6. Performance and inference time vary with block number. We set both N_{spv} and N_{svp} from 1 to 4. The left / right one is Performance / Inference time vs. Block number respectively.

Multi-scale Pooling Layers. Multi-scale (MS) Pooling Layer tends to aggregate more levels of context information in a wider neighborhood and different receptive fields. By removing this unit, the pointwise features will only go through several MLP Layers. Shown in Tab. 4, the MS Pooling Layer improves the mIoU from 65.4% to 67.3%.

Baseline	Attentive Gathering	MS Pooling	mIoU(%)
✓	×	×	64.6
✓	✓	×	65.4
✓	✓	✓	67.3

Table 4. Ablation study on SemanticKITTI.

Attentive gathering. Without attentive gathering, we only use nearest gathering due to the consideration of computation cost, and the comparison between nearest gathering and bilinear gathering is shown in Fig. 4. With precomputed attentive weights within the voxel, the performance is improved with 0.8% in DRINet, showing that attentive gathering can prevent the feature degradation problem.

4.3. Indoor Tasks

ModelNet40 classification. ModelNet40 [48], one of the most popular 3D classification datasets, contains 12,311 meshed CAD models from 40 categories. We use normals as extra features and sample 1024 points as input. Following the standard processing procedure, the input points are first normalized to unit range. As a result, the scale of multi-scale pooling is set to $[0.2m, 0.4m, 0.6m, 0.8m]$. During training, data augmentation including rotation, scaling, flipping and perturbation is applied. The DRINet is built with two SVPFE and SPVFE blocks with the voxel scales at $[0.2m, 0.2m]$. We gather all pointwise features from point blocks and apply max operation same as PointNet [26], then add one fully connected layer to generate output scores. In Tab. 5, our DRINet directly improves the performance in the 3D classification task. We observe that DRINet with dual-representations performs better than the previous state-of-the-art method with single representation.

ShapeNet Parts segmentation. We also evaluate our method on the segmentation task of ShapeNet Parts dataset [43] which is a collection of 16681 point clouds of 16 categories. Since ShapeNet Parts is also generated from CAD model, we apply the same settings for data preprocessing and augmentation as ModelNet40. For this task, we set block number to 4 to increase the overall network complexity, which can boost the performance. From Tab. 6, our network achieves state-of-the-art according to mIoU criteria with high runtime efficiency (30ms).

S3DIS. The S3DIS dataset [1] consists of 271 rooms belonging to 6 large-scale indoor areas with 13 classes. Following the previous works, we use Area 5 as the test set and the rest as the training set. For a fair comparison, we use the same data processing and evaluation protocol as these works [21, 23]. We also use four blocks of SPVFE and SVPFE modules with the voxel scales at $[0.2m, 0.2m, 0.2m, 0.2m]$.

Method	Input	Size	Acc(%)
MVCNN [29]	*I*	3 × 1024	90.1
PointNet++ [27]	*P*	6 × 1024	91.9
PointCNN [21]	*P*	6 × 1024	92.2
LP-3DCNN [17]	*V*	6 × 1024	92.1
LDGCNN [44]	*G*	6 × 1024	92.9
KPConv [33] *regid*	*P*	3 × 1024	92.9
KPConv [33] *deform*	*P*	3 × 1024	92.7
CloserLook3D [22]	*P*	3 × 1024	92.9
DRINet(ours)	*P + V*	6 × 1024	**93.0**

Table 5. Results on ModelNet40 [48],*P*, *I*, *G*, *V* mean point, image, graph and volumetric respectively

Method	Input	mIoU(%)	Latency(**ms**)
KPConv [33] *regid*	*P*	86.2	-
KPConv [33] *deform*	*P*	**86.4**	-
PointNet++ [27]	*P*	85.1	77.9
PointCNN [21]	*P*	86.13	-
SO-Net [20]	*P*	84.9	-
PVCNN [23]	*P + V*	86.2	50.7
DRINet(ours)	*P + V*	**86.4**	30

Table 6. Results on ShapeNet

Methods	Input	mIoU(%)
PointNet [26]	*P*	41.09
RSNet [14]	*P*	56.5
TangentConv [32]	*P*	52.6
PointCNN [21]	*P*	57.3
PVCNN [23]	*P + V*	58.98
ASIS [36]	*P*	53.4
KPConv [33]	*P*	**67.1**
CloserLook3D [22]	*P*	66.7
DRINet(ours)	*P + V*	66.7

Table 7. Results on the Area5 of S3DIS.

As shown in Tab. 7, our DRINet achieves state-of-the-art compared with single representation methods. For the dual-representation methods PVCNN [23], we even outperform its largest model with less computation cost.

5. Conclusion

We have proposed DRINet, a novel and flexible architecture for dual-representation point cloud learning. DRINet decouples the feature learning process as SPVFE and SVPFE. In SPVFE, DRINet generates better pointwise features with the novel multi-scale pooling layer that can aggregate features at different scales. In SVPFE, the attentive gathering is proposed to deal with feature degradation when abandoning bilinear gathering operations that introduce huge memory footprints. Experiments show that our method achieves state-of-the-art *mIoU* and *Accuracy* in both indoor and outdoor classification and segmentation tasks with real-time speed.

References

[1] Iro Armeni, Sasha Sax, Amir R Zamir, and Silvio Savarese. Joint 2d-3d-semantic data for indoor scene understanding. *arXiv preprint arXiv:1702.01105*, 2017. 2, 8

[2] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *ICCV*, pages 9297–9307, 2019. 1, 2, 6

[3] Maxim Berman, Amal Rannen Triki, and Matthew B Blaschko. The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In *CVPR*, pages 4413–4421, 2018. 6

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 5

[5] Siheng Chen, Sufeng Niu, Tian Lan, and Baoan Liu. Pct: Large-scale 3d point cloud representations via graph inception networks with applications to autonomous driving. In *2019 IEEE international conference on image processing (ICIP)*, pages 4395–4399. IEEE, 2019. 3

[6] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *CVPR*, 2017. 3

[7] Xieyuanli Chen, Andres Milioto, Emanuele Palazzolo, Philippe Giguère, Jens Behley, and Cyrill Stachniss. Suma++: Efficient lidar-based semantic slam. In *IROS*, pages 4530–4537. IEEE, 2019. 3

[8] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Fast Point R-CNN. In *CVPR*, 2019. 3

[9] Tiago Cortinhal, George Tzelepis, and Eren Erdal Aksoy. Salsanext: Fast semantic segmentation of lidar point clouds for autonomous driving. *arXiv preprint arXiv:2003.03653*, 2020. 3

[10] Benjamin Graham, Martin Engelcke, and Laurens Van Der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *CVPR*, 2018. 5

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 5

[12] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 3

[13] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *CVPR*, pages 11108–11117, 2020. 1, 3, 6, 7

[14] Qiangui Huang, Weiyue Wang, and Ulrich Neumann. Recurrent slice networks for 3d segmentation of point clouds. In *CVPR*, pages 2626–2635, 2018. 8

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[16] Jason Ku, Alex D. Pon, Sean Walsh, and Steven L. Waslander. Improving 3D Object Detection for Pedestrians with Virtual Multi-View Synthesis Orientation Estimation. *arXiv:1907.06777 [cs]*, 2019. 3

[17] Sudhakar Kumawat and Shanmuganathan Raman. Lp-3dcnn: Unveiling local phase in 3d convolutional neural networks. In *CVPR*, pages 4903–4912, 2019. 8

[18] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. PointPillars: Fast Encoders for Object Detection From Point Clouds. In *CVPR*, pages 12697–12705, 2019. 3, 4

[19] Felix Järemo Lawin, Martin Danelljan, Patrik Tosteberg, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Deep projective 3d semantic segmentation. In *International Conference on Computer Analysis of Images and Patterns*, pages 95–107. Springer, 2017. 3

[20] Jiaxin Li, Ben M Chen, and Gim Hee Lee. So-net: Self-organizing network for point cloud analysis. In *CVPR*, pages 9397–9406, 2018. 8

[21] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *NeurIPS*, pages 820–830, 2018. 8

[22] Ze Liu, Han Hu, Yue Cao, Zheng Zhang, and Xin Tong. A closer look at local aggregation operators in point cloud analysis. In *ECCV*, pages 326–342. Springer, 2020. 8

[23] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. In *NeurIPS*, pages 965–975, 2019. 1, 2, 3, 5, 8

[24] Andres Milioto, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. Rangenet++: Fast and accurate lidar semantic segmentation. In *IROS*, pages 4213–4220. IEEE, 2019. 1, 3, 7

[25] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *ICML*, pages 2014–2023. PMLR, 2016. 5

[26] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 652–660, 2017. 1, 2, 6, 7, 8

[27] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *NeurIPS*, volume 30, pages 5099–5108. Curran Associates, Inc., 2017. 1, 2, 5, 7, 8

[28] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *CVPR*, pages 2530–2539, 2018. 3

[29] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik G. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, 2015. 8

[30] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, pages 6105–6114. PMLR, 2019. 3

[31] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *ECCV*, pages 685–702. Springer, 2020. 1, 2, 5, 7

[32] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. Tangent convolutions for dense prediction in 3d. In *CVPR*, pages 3887–3896, 2018. 7, 8

[33] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, pages 6411–6420, 2019. 3, 7, 8

[34] Ozan Unal, Luc Van Gool, and Dengxin Dai. Improving point cloud semantic segmentation by learning 3d object detection. In *WACV*, pages 2950–2959, 2021. 1, 7

[35] Bei Wang, Jianping An, and Jiayan Cao. Voxel-FPN: Multi-scale voxel feature aggregation in 3D object detection from point clouds. *arXiv:1907.05286 [cs, stat]*, 2019. 3

[36] Xinlong Wang, Shu Liu, Xiaoyong Shen, Chunhua Shen, and Jiaya Jia. Associatively segmenting instances and semantics in point clouds. In *CVPR*, pages 4096–4105, 2019. 8

[37] Bichen Wu, Alvin Wan, Xiangyu Yue, and Kurt Keutzer. Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. In *ICRA*, pages 1887–1893. IEEE, 2018. 1, 3

[38] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *CVPR*, pages 9621–9630, 2019. 3

[39] Chenfeng Xu, Bichen Wu, Zining Wang, Wei Zhan, Peter Vajda, Kurt Keutzer, and Masayoshi Tomizuka. Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation. *arXiv preprint arXiv:2004.01803*, 2020. 7

[40] Qiangeng Xu, Xudong Sun, Cho-Ying Wu, Panqu Wang, and Ulrich Neumann. Grid-gcn for fast and scalable point cloud learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5661–5670, 2020. 3

[41] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 5

[42] Maosheng Ye, Shuangjie Xu, and Tongyi Cao. Hvnet: Hybrid voxel network for lidar based 3d object detection. In *CVPR*, pages 1631–1640, 2020. 4, 5

[43] Li Yi, Vladimir G. Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Trans. Graph.*, 35(6), Nov. 2016. 8

[44] Kuangen Zhang, Ming Hao, Jing Wang, Clarence W de Silva, and Chenglong Fu. Linked dynamic graph cnn: Learning on point cloud via linking hierarchical features. *arXiv preprint arXiv:1904.10014*, 2019. 8

[45] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, Boqing Gong, and Hassan Foroosh. Polarnet: An improved grid representation for online lidar point clouds semantic segmentation. In *CVPR*, pages 9601–9610, 2020. 1, 3, 6, 7

[46] Zhiyuan Zhang, Binh-Son Hua, and Sai-Kit Yeung. Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1607–1616, 2019. 3

[47] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017. 3, 5

[48] Zhirong Wu, S. Song, A. Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, pages 1912–1920, 2015. 2, 8

[49] Hui Zhou, Xinge Zhu, Xiao Song, Yuexin Ma, Zhe Wang, Hongsheng Li, and Dahua Lin. Cylinder3d: An effective 3d framework for driving-scene lidar semantic segmentation. *arXiv preprint arXiv:2008.01550*, 2020. 3

[50] Yin Zhou and Oncel Tuzel. VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. In *CVPR*, pages 4490–4499, 2018. 1, 2, 3