Rethinking Range View Representation for LiDAR Segmentation

Lingdong Kong[1,2] Youquan Liu[1,3] Runnan Chen[1,4] Yuexin Ma[5] Xinge Zhu[6]
Yikang Li[1] Yuenan Hou[1,✉] Yu Qiao[1] Ziwei Liu[7,✉]

[1]Shanghai AI Laboratory [2]National University of Singapore [3]Hochschule Bremerhaven [4]The University of Hong Kong
[5]ShanghaiTech University [6]The Chinese University of Hong Kong [7]S-Lab, Nanyang Technological University

{konglingdong,liuyouquan,chenrunnan,houyuenan}@pjlab.org.cn ziwei.liu@ntu.edu.sg

Abstract

LiDAR segmentation is crucial for autonomous driving perception. Recent trends favor point- or voxel-based methods as they often yield better performance than the traditional range view representation. In this work, we unveil several key factors in building powerful range view models. We observe that the "many-to-one" mapping, semantic incoherence, and shape deformation are possible impediments against effective learning from range view projections. We present ***RangeFormer*** *– a full-cycle framework comprising novel designs across network architecture, data augmentation, and post-processing – that better handles the learning and processing of LiDAR point clouds from the range view. We further introduce a* ***S****calable* ***T****raining from* ***R****ange view* ***(STR)*** *strategy that trains on arbitrary low-resolution 2D range images, while still maintaining satisfactory 3D segmentation accuracy. We show that, for the first time, a range view method is able to surpass the point, voxel, and multi-view fusion counterparts in the competing LiDAR semantic and panoptic segmentation benchmarks, i.e., SemanticKITTI, nuScenes, and ScribbleKITTI.*

1. Introduction

LiDAR point clouds have unique characteristics. As the direct reflections of real-world scenes, they are often diverse and unordered and thus bring extra difficulties in learning [26, 40]. Inevitably, a good representation is needed for efficient and effective LiDAR point cloud processing [64].

Although there exist various LiDAR representations as shown in Tab. 1, the prevailing approaches are mainly based on point view [32, 61], voxel view [15, 60, 81, 28], and multi-view fusion [41, 72, 51]. These methods, however, require computationally intensive neighborhood search [50], 3D convolution operations [43], or multi-branch networks [2, 24], which are often inefficient during both training and inference stages. The projection-based representations, such as range view [68, 46] and bird's eye view [78, 80],

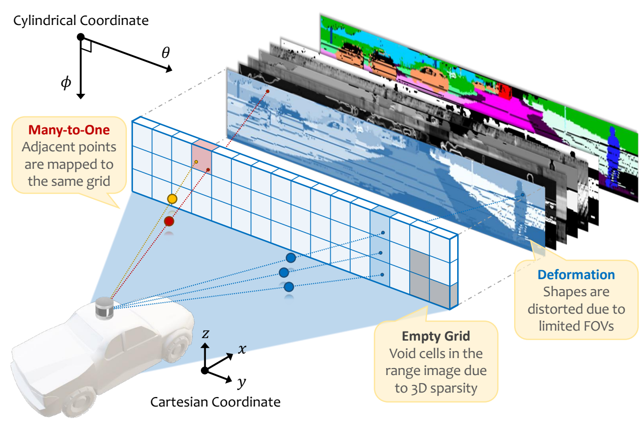


Figure 1: Three detrimental factors observed in the LiDAR range view representation: 1) the "many-to-one" problem; 2) "holes" or empty grids; and 3) shape distortions.

Table 1: Comparisons for different LiDAR representations.

View	Formation	Complexity	Representative
Raw Points	Bag-of-Points	$\mathcal{O}(N \cdot d)$	RandLA-Net, KPConv
Range View	Range Image	$\mathcal{O}(\frac{H \cdot W}{r^2} \cdot d)$	SqueezeSeg, RangeNet++
Bird's Eye View	Polar Image	$\mathcal{O}(\frac{H \cdot W}{r^2} \cdot d)$	PolarNet
Voxel (Dense)	Voxel Grid	$\mathcal{O}(\frac{H \cdot W \cdot L}{r^3} \cdot d)$	PVCNN
Voxel (Sparse)	Sparse Grid	$\mathcal{O}(N \cdot d)$	MinkowskiNet, SPVNAS
Voxel (Cylinder)	Sparse Grid	$\mathcal{O}(N \cdot d)$	Cylinder3D
Multi-View	Multiple	$\mathcal{O}((N + \frac{H \cdot W}{r^2}) \cdot d)$	AMVNet, RPVNet

are more tractable options. The 3D-to-2D rasterizations and mature 2D operators open doors for fast and scalable in-vehicle LiDAR perception [46, 71, 64]. Unfortunately, the segmentation accuracy of current projection-based methods [79, 13, 78] is still far behind the trend [73, 72, 75].

The challenge of learning from projected LiDAR scans comes from the potential detrimental factors of the LiDAR data representation [46]. As shown in Fig. 1, the range view projection[1] often suffers from several difficulties, including 1) the "many-to-one" conflict of adjacent points, caused

[1]We show a frustum of the LiDAR scan for simplicity; the complete range view projection is a cylindrical panorama around the ego-vehicle.

by limited horizontal angular resolutions; 2) the "holes" in the range images due to 3D sparsity and sensor disruptions; and 3) potential shape deformations during the rasterization process. While these problems are ubiquitous in range view learning, previous works hardly consider tackling them. Stemming from the image segmentation community [77], prior arts widely adopt the fully-convolutional networks (FCNs) [44, 8] for range view LiDAR segmentation [46, 79, 13, 35]. The limited receptive fields of FCNs cannot directly model long-term dependencies and are thus less effective in handling the mentioned impediments.

In this work, we seek an alternative in lieu of the current range view LiDAR segmentation models. Inspired by the success of Vision Transformer (ViT) and its follow-ups [19, 67, 70, 42, 57], we design a new framework dubbed *RangeFormer* to better handle the learning and processing of LiDAR point clouds from the range view. We formulate the segmentation of range view grids as a seq2seq problem and adopt the standard self-attention modules [66] to capture the rich contextual information in a "global" manner, which is often omitted in FCNs [46, 1, 13]. The hierarchical features extracted with such global awareness are then fed into multi-layer perceptions (MLPs) for decoding. In this way, every point in the range image is able to establish interactions with other points – no matter whether close or far and valid or empty – and further lead to more effective representation learning from the LiDAR range view.

It is worth noting that such architectures, albeit straightforward, still suffer several difficulties. The first issue is related to data diversity. The prevailing LiDAR segmentation datasets [7, 21, 5, 59] contain tens of thousands of LiDAR scans for training. These scans, however, are less diverse in the sense that they are collected in a sequential way. This hinders the training of Transformer-based architectures as they often rely on sufficient samples and strong data augmentations [19]. To better handle this, We design an augmentation combo that is tailored for range view. Inspired by recent 3D augmentation techniques [80, 36, 47], we manipulate the range view grids with row mixing, view shifting, copy-paste, and grid fill. As we will show in the following sections, these lightweight operations can significantly boost the performance of SoTA range view methods.

The second issue comes from data post-processing. Prior works adopt CRF [68] or k-NN [46] to smooth/infer the range view predictions. However, it is often hard to find a good balance between the under- and over-smoothing of the 3D labels in unsupervised manners [34]. In contrast, we design a supervised post-processing approach that first subsamples the whole LiDAR point cloud into equal-interval "sub-clouds" and then infer their semantics, which holistically reduces the uncertainty of aliasing range view grids.

To further reduce the overhead in range view learning, we propose *STR* – a scalable range view training paradigm. *STR* first "divides" the whole LiDAR scan into multiple groups along the azimuth direction and then "conquers" each of them. This transforms range images of high horizontal resolutions into a stack of low-resolution ones while can better maintain the best-possible granularity to ease the "many-to-one" conflict. Empirically, We find *STR* helpful in reducing the complexity during training, without sacrificing much convergence rate and segmentation accuracy.

The advantages of *RangeFormer* and *STR* are demonstrated from aspects of LiDAR segmentation accuracy and efficiency on prevailing benchmarks. Concretely, we achieve 73.3% mIoU and 64.2% PQ on SemanticKITTI [5], surpassing prior range view methods [79, 13] by significant margins and also better than SoTA fusion-based methods [73, 30, 75]. We also establish superiority on the nuScenes [21] (sparser point clouds) and ScribbleKITTI [65] (weak supervisions) datasets, which validates our scalability. While being more effective, our approaches run $2\times$ to $5\times$ faster than recent voxel [81, 60] and fusion [72, 73] methods and can operate at sensor frame rate.

2. Related Work

LiDAR Representation. The LiDAR sensor is designed to capture high-fidelity 3D structural information which can be represented by various forms, *i.e.*, raw point [49, 50, 61], range view [31, 69, 71, 1], bird's eye view (BEV) [78], voxel [43, 15, 81, 75, 10], and multi-view fusion [41, 72, 73], as summarized in Tab. 1. The point and sparse voxel methods are prevailing but suffer $\mathcal{O}(N \cdot d)$ complexity, where N is the number of points and often in the order of 10^5 [64]. BEV offers an efficient representation but only yields sub-par performance [9]. As for fusion-based methods, they often comprise multiple networks which are too heavy to yield reasonable training overhead and inference latency [51, 75, 58]. Among all representations, range view is the one that directly reflects the LiDAR sampling process [62, 20, 63]. We thus focus on this modality to further embrace its compactness and rich semantic/structural cues.

Architecture. Previous range view methods are built upon mature FCN structures [44, 68, 69, 71, 3]. RangeNet++ [46] proposed an encoder-decoder FCN based on DarkNet [53]. SalsaNext [17] uses dilated convolutions to further expand the receptive fields. Lite-HDSeg [52] proposed to adopt harmonic convolution to reduce the computation overhead. EfficientLPS [55] proposed a proximity convolution module to leverage neighborhood points in the range image. FIDNet [79] and CENet [13] switch the encoders to ResNet and replace the decoder with simple interpolations. In contrast to using FCNs, we build *RangeFormer* upon self-attentions and demonstrate potential and advantages for long-range dependency modeling in range view learning.

Augmentation. Most 3D data augmentation techniques are object-centric [76, 11, 54, 38] and thus not generalizable to

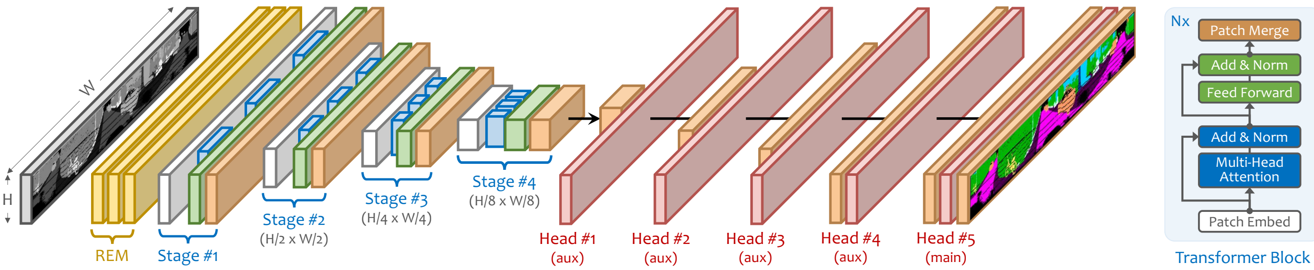


Figure 2: **Architecture overview**. The rasterized LiDAR point cloud of spatial size $H \times W$ is fed into four consecutive stages where each comprising several standard Transformer blocks as shown in the right subfigure. The multi-scale features extracted from these different stages are then fed into the MLP heads for decoding. The final predictions in 2D will be projected back to 3D in a reverse manner of Eq. (1).

scenes. Panoptic-PolarNet [80] over-samples rare instance points during training. Mix3D [47] proposed an out-of-context mixing by supplementing points from one scene to another. MaskRange [25] designs a weighted paste drop augmentation to alleviate overfitting and improve class balance. LaserMix [36] proposed to mix labeled and unlabeled LiDAR scans along the inclination axis for effective semi-supervised learning. In this work, we present a novel and lightweight augmentation combo tailored for range view learning that combines mixing, shifting, union, and copy-paste operations directly on the rasterized grids, while still maintaining the structural consistency of the scenes.

Post-Processing. Albeit being an indispensable module of range view LiDAR segmentation, prior works hardly consider improving the post-processing process [64]. Most works follow the CRF [68] or k-NN [46] to smooth or infer the semantics for conflict points. Recently, Zhao *et al.* proposed another unsupervised method named NLA for nearest label assignment [79]. We tackle this in a supervised way by creating "sub-clouds" from the full point cloud and inferring labels for each subset, which directly reduces the information loss and helps alleviate the "many-to-one" problem.

3. Technical Approach

In this section, we first revisit the details of range view rasterization (Sec. 3.1). To better tackle the impediments in range view learning, we introduce *RangeFormer* (Sec. 3.2) and *STR* (Sec. 3.3) which emphasize the effectiveness and efficiency, respectively, for scalable LiDAR segmentation.

3.1. Preliminaries

Mounted on the roof of the ego-vehicle (as illustrated in Fig. 1), the rotating LiDAR sensor emits isotropic laser beams with predefined angles and perceives the positions and reflection intensity of surroundings via time measurements in the scan cycle. Specifically, each LiDAR scan captures and returns N points in a single scan cycle, where each point p_n in the scan is represented by the Cartesian coordinates (p_n^x, p_n^y, p_n^z), intensity p_n^i, and existence p_n^e.

Rasterization. For a given LiDAR point cloud, we rasterize points within this scan into a 2D cylindrical projection $\mathcal{R}(u,v)$ (*a.k.a.*, range image) of size $H \times W$, where H and W are the height and width, respectively. The rasterization process for each point p_n can be formulated as follows:

$$\begin{pmatrix} u_n \\ v_n \end{pmatrix} = \begin{pmatrix} \frac{1}{2}\left[1 - \arctan(p_n^y, p_n^x)\pi^{-1}\right] W \\ \left[1 - \left(\arcsin(p_n^z, (p_n^d)^{-1}) + \phi^{\text{down}}\right)\xi^{-1}\right] H \end{pmatrix}, \tag{1}$$

where (u_n, v_n) denotes the grid coordinate of point p_n in range image $\mathcal{R}(u,v)$; $p_n^d = \sqrt{(p_n^x)^2 + (p_n^y)^2 + (p_n^z)^2}$ is the depth between the point and LiDAR sensor (ego-vehicle); $\xi = |\phi^{\text{up}}| + |\phi^{\text{down}}|$ denotes the vertical field-of-views (FOVs) of the sensor and ϕ^{up} and ϕ^{down} are the inclination angles at the upward and downward directions, respectively. Note that H is often predefined by the beam number of the LiDAR sensor, while W can be set based on requirements.

Formation. The final range image $\mathcal{R}(u,v) \in \mathbb{R}^{(6,H,W)}$ is composed of six rasterized feature embeddings, *i.e.*, coordinates (p^x, p^y, p^z), depth p^d, intensity p^i, and existence p^e (indicates whether or not a grid is occupied by valid point). The range semantic label $y(u,v) \in \mathbb{R}^{(H,W)}$ – which is rasterized from the per-point label in 3D – shares the same rasterization index and resolution with $\mathcal{R}(u,v)$. The 3D segmentation problem is now turned into a 2D one and the grid predictions in the range image can then be projected back to point-level in a reverse manner of Eq. (1).

3.2. RangeFormer: A Full-Cycle Framework

As discussed in previous sections, there exist potential detrimental factors in the range view representation (Fig. 1). The one-to-one correspondences from Eq. (1) are often untenable since $H \times W$ is much less than N. Typically, prior arts [46, 2, 13] adopt $(H, W) = (64, 512)$ to rasterize LiDAR scans of around 120k points each [5], resulting in over 70% information loss[2]. The restricted horizontal angular

[2]Note: # of 2D grids / # of 3D points = 64×512 / $120000 \approx 27.3\%$.

resolutions and an intensive number of empty grids in range image tend to bring extra difficulties during model training, such as shape deformation, semantic incoherence, *etc*.

Architecture. To pursue larger receptive fields and longer dependency modeling, we design a self-attention-based network comprising standard Transformer blocks and MLP heads as shown in Fig. 2. Given a batch of rasterized range image $\mathcal{R}(u,v)$, the range embedding module (REM) which consists of three MLP layers first maps each point in the grid to a higher-dim embedding $\mathcal{F}_0 \in \mathbb{R}^{(128,H,W)}$. This is analogous to PointNet [49]. Next, we divide $\mathcal{F}_0$ into overlapping patches of size 3 by 3 and feed them into the Transformer blocks. Similar to PVT [67], we design a pyramid structure to facilitate multi-scale feature fusions, yielding $\{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3, \mathcal{F}_4\}$ for four stages, respectively, with downsampling factors 1, 2, 4, and 8. Each stage consists of customized numbers of Transformer blocks and each block include two modules. 1) *Multi-head self-attention* [66], serves as the main computing bottleneck and can be formulated as:

$$O = \mathrm{Mul}(Q,K,V) = \mathrm{Concat}(\mathrm{head}_1, ..., \mathrm{head}_h)W^O, \quad (2)$$

where $\mathrm{head}_i = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$ denotes the self-attention operation with $\mathrm{Attention} = \sigma(\frac{QK}{\sqrt{d^{\mathrm{head}}}})V$; σ denotes softmax and d^{head} is the dimension of each head; W^Q, W^K, W^V, and W^O are the weight matrices of query Q, key K, value V, and output O. As suggested in [67], the sequence length of K and V are further reduced by a factor R to save the computation overhead. 2) *Feed-forward network (FFN)*, which consists of MLPs and activation as:

$$\mathcal{F} = \mathrm{FFN}(O) = \mathrm{Linear}(\mathrm{GELU}(\mathrm{Linear}(O))) \oplus O, \quad (3)$$

where $\oplus$ denotes the residual connection [27]. Different from ViT [23], we discard the explicit position embedding and rather incorporate it directly within the feature embeddings. As introduced in [70], this can be achieved by adding a single 3 by 3 convolution with zero paddings into FFN.

Semantic Head. To avoid heavy computations in decoding, we adopt simple MLPs as the segmentation heads. After retrieving all features from the four stages, we first unify their dimensions. This is achieved in two steps: 1) *Channel unification*, where each $\mathcal{F}_i$ with embedding size $d^{\mathcal{F}_i}, i = 1,2,3,4$, is unified via one MLP layer. 2) *Spatial unification*, where $\mathcal{F}_i$ from the last three stages are resized to the range embedding size $H \times W$ by simple bi-linear interpolation. The decoding process for stage i is thus:

$$\mathcal{H}_i = \mathrm{Bi\text{-}Interpolate}(\mathrm{Linear}(\mathcal{F}_i)). \quad (4)$$

As proved in [79], the bi-linear interpolation of range view grids is equivalent to the distance interpolation (with four neighbors) in PointNet++ [50]. Here the former operation serves as the better option since it is totally parameter-free. Finally, we concatenate four $\mathcal{H}_i$ together and feed it into another two MLP layers, where the channel dimension is gradually mapped to d^{cls}, *i.e.* the class number, to form the class probability distribution. Additionally, we add an extra MLP layer for each $\mathcal{H}_i$ as the auxiliary head. The predictions from the main head and four auxiliary heads are supervised separately during training. As for inference, we only keep the main head and discard the auxiliary ones.

Panoptic Head. Similar to Panoptic-PolarNet [80], we add a panoptic head on top of *RangeFormer* to estimate the instance centers and offsets, dubbed *Panoptic-RangeFormer*. Since we tackle this problem in a bottom-up manner, the semantic predictions of the *things* classes are utilized as the foreground mask to form instance groups in 3D. Next, we conduct 2D class-agnostic instance grouping by predicting the center heatmap [12] and offsets for each point on the XY-plane. Based on [80], the predictions from the above two aspects can then be fused via majority voting. As we will show in the experiments, the advantages of *RangeFormer* in semantic learning further yield much better panoptic segmentation performance.

RangeAug. Data augmentation often helps the model learn more general representations and thus increases both accuracy and robustness. Prior arts in LiDAR segmentation conduct a series of augmentations at point-level [81], *i.e.*, global rotation, jittering, flipping, and random dropping, which we refer to as "common" augmentations. To better embrace the rich semantic and structural cues of the range view representation, we propose an augmentation combo comprising the following four operations.

1) *RangeMix*, which mixes two scans along the inclination $\phi = \arctan(\frac{p^z}{\sqrt{(p^x)^2+(p^y)^2}})$ and azimuth θ directions. This can be interpreted as switching certain rows of two range images. After calculating ϕ and θ for the current scan and the randomly sampled scan, we then split points into k_{mix} equal spanning inclination ranges, *i.e.*, different mixing strategies. The corresponding points in the same inclination range from the two scans are then switched. In our experiments, we design mixing strategies from a combination, and k_{mix} is randomly sampled from a list $[2,3,4,5,6]$.

2) *RangeUnion*, which fills in the empty grids of one scan with grids from another scan. Due to the sparsity in 3D and potential sensor disruptions, a huge number of grids are empty even after rasterization. We thus use the existence embedding p^e to search and fill in these void grids and this further enriches the actual capacity of the range image. Given a number of $N_{\mathrm{union}} = \sum_n p_n^e$ empty range view grids, we randomly select $k_{\mathrm{union}}N_{\mathrm{union}}$ candidate grids for point filling, where k_{union} is set as 50%.

3) *RangePaste*, which copies tail classes from one scan to another scan at correspondent positions in the range image. This boosts the learning of rare classes and also maintains the objects' spatial layout in the projection. The ground-

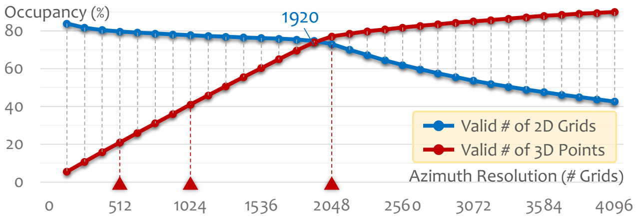


Figure 3: The **occupancy trade-off** between 2D grids & 3D points in the LiDAR range view representation. Statistics calculated on the SemanticKITTI [5] dataset.

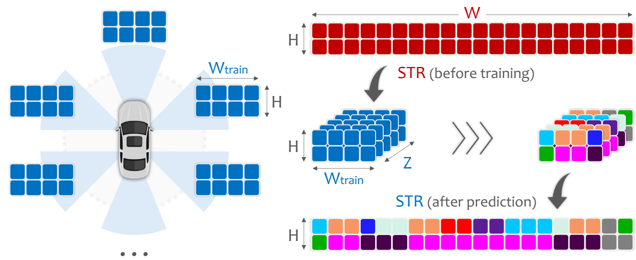


Figure 4: Illustration of the proposed **STR paradigm**. We split LiDAR points into multiple "views" (left) and rasterized them into range images with high horizontal angular resolutions (right). After training, the predictions are concatenated sequentially to form the complete LiDAR scan.

truth semantic labels of a randomly sampled scan are used to create pasting masks. The classes to be pasted are those in the "tail" distribution, which forms a semantic class list (`sem classes`). After indexing the rare classes' points, we paste them into the current scan while maintaining the corresponding positions in the range image.

4) *RangeShift*, which slides the scan along the azimuth direction $\theta = \arctan(p^y/p^x)$ to change the global position embedding. This corresponds to shifting the range view grids along the row direction with k_{shift} rows. In our experiments, k_{shift} is randomly sampled from a range of $\frac{W}{4}$ to $\frac{3W}{4}$. These four augmentations are tailored for range view and can operate on-the-fly during the data loading process, without adding extra overhead during training. As we will show in the next section, they play a vital role in boosting the performance of range view segmentation models.

RangePost. The widely-used k-NN [46] votes and assigns labels for points near the boundary in an unsupervised way, which cannot handle the "many-to-one" conflict concretely. Differently, we tackle this in a supervised manner. We first sub-sample the whole point cloud into equal-interval "sub-clouds". Since adjacent points have a high likelihood of belonging to the same class, these "sub-clouds" are sharing very similar semantics. Next, we stack and feed these subsets to the network. After obtaining the predictions, we then stitch them back to their original positions. For each scan, this will automatically assign labels for points that are merged during rasterization in just a single forward pass, which directly reduces the information loss caused by "many-to-one" mappings. Finally, prior post-processing techniques [46, 79] can then be applied to these new predictions to further enhance the re-rasterization process.

3.3. STR: Scalable Training from Range View

To pursue better training efficiency, prior works adopt low horizontal angular resolutions, *i.e.*, small values of W in Eq. (1), for range image rasterization [46, 2]. This inevitably intensifies the "many-to-one" conflict, causes more severe shape distortions, and leads to sub-par performance.

2D & 3D Occupancy. Instead of directly assigning small W for $\mathcal{R}(u, v)$, we first lookup for the best possible options. We find an "occupancy trade-off" between the number of points in the LiDAR scan and the desired capacity of the range image. As shown in Fig. 3, the conventional choices, *i.e.*, 512, 1024, and 2048, are not optimal. The crossover of two lines indicates that the range image of width 1920 tends to be the most *informative* representation. However, this configuration inevitably consumes much more memory than the conventionally used 512 or 1024 resolutions and further increases the training and inference overhead.

Multi-View Partition. To maintain the relatively high resolution of W while pursuing efficiency at the same time, we propose a "divide-and-conquer" learning paradigm. Specifically, we first partition points in the LiDAR scan into multiple groups based on the unique azimuth angle of each point, *i.e.*, $\theta_i = \arctan(p_i^y/p_i^x)$. This will constitute Z non-overlapping "views" of the complete 360° range view panorama as shown in Fig. 4, where Z is a hyperparameter and determines the total number of groups to be split. Next, points from each group will be rasterized separately with a high horizontal resolution to mitigate "many-to-one" and deformation issues. In this way, the actual horizontal training resolution of the range image is eased by Z times, *i.e.*, $W_{\text{train}} = \frac{W}{Z}$, while the granularity (# of grids) of the range view projection in each "view" is perfectly maintained.

Training & Inference. During training, for each LiDAR scan, we randomly select only one of the Z point groups for rasterization. That is to say, the model will be trained with a batch of randomly sampled "views" at each step. During inference, we rasterize all groups for a given scan and stack the range images along the batch dimension. All "views" can now be inferred in a single pass and the predictions are then wrapped back to form the complete scan. Despite being an empirical design, we find this *STR* paradigm highly scalable during training. The convergence rate of training from multiple "views" tends to be consistent with the conventional training paradigm, *i.e.*, *STR* can achieve competitive results using the *same* number of iterations, while the memory consumption has now been reduced to only $\frac{1}{Z}$, which liberates the use of *small-memory* GPUs for training.

4. Experimental Analysis

4.1. Settings

Benchmarks. We conduct experiments on three standard LiDAR segmentation datasets. ***SemanticKITTI*** [5] provides 22 sequences with 19 semantic classes, captured by a 64-beam LiDAR sensor. Sequences 00 to 10 (*exc.* 08), 08, and 11 to 21 are used for training, validation, and testing, respectively. ***nuScenes*** [21] consists of 1000 driving scenes collected from Boston and Singapore, which are sparser due to the use of a 32-beam sensor. 16 classes are adopted after merging similar and infrequent classes. ***ScribbleKITTI*** [65] shares the exact same data configurations with [5] but is weakly annotated with line scribbles, which corresponds to around 8.06% semantic labels available during training.

Evaluation Metrics. Following the standard practice, we report the Intersection-over-Union (IoU) for class i and the average score (mIoU) over all classes, where $\text{IoU}_i = \frac{\text{TP}_i}{\text{TP}_i+\text{FP}_i+\text{FN}_i}$. TP_i, FP_i and FN_i are the true-positive, false-positive, and false-negative. For panoptic segmentation, the models are measured by the Panoptic Quality (PQ) [33]

$$\text{PQ} = \underbrace{\frac{\sum_{(i,j)\in\text{TP}} \text{IoU}(i,j)}{|\text{TP}|}}_{\text{SQ}} \times \underbrace{\frac{|\text{TP}|}{|\text{TP}| + \frac{1}{2}(|\text{FP}| + |\text{FN}|)}}_{\text{RQ}}, \quad (5)$$

which consists of Segmentation Quality (SQ) and Recognition Quality (RQ). We also report the separated scores for *things* and *stuff* classes, *i.e.*, PQ^{Th}, SQ^{Th}, RQ^{Th}, and PQ^{St}, SQ^{St}, RQ^{St}. $\text{PQ}^{\dagger}$ is defined by swapping the PQ of each *stuff* class to its IoU then averaging over all classes [48].

Network Configurations. After range view rasterization, the input $\mathcal{R}(u,v)$ of size $6 \times H \times W$ is first fed into REM for range view point embedding. It consists of three MLP layers that map the embedding dim of $\mathcal{R}(u,v)$ from 6 to 64, 128, and 128, respectively, with the batch norm and GELU activation. The output of size $128 \times H \times W$ from REM serves as the input of the Transformer blocks. Specifically, for each of the four stages, the patch embedding layer divides an input of size $H_{\text{embed}}, W_{\text{embed}}$ into 3×3 patches with overlap stride equals to 1 (for the first stage) and 2 (for the last three stages). After the overlap patch embedding, the patches are processed with the standard multi-head attention operations as in [19, 67, 70]. We keep the default setting of using the residual connection and layer normalization (Add & Norm). The number of heads for each of the four stages is $[3,4,6,3]$. The hierarchical features extracted from different stages are stored and used for decoding. Specifically, each of the four stages produces features of spatial size $[(H,W),(\frac{H}{2},\frac{W}{2}),(\frac{H}{4},\frac{W}{4}),(\frac{H}{8},\frac{W}{8})]$, with the channel dimension of $[128,128,320,512]$. As described in previous sections, we perform two unification steps to unify the channel and spatial sizes of different feature maps. We first map their channel dimensions to 256, *i.e.*, $[128,H,W] \rightarrow [256,H,W]$ for stage 1, $[128,\frac{H}{2},\frac{W}{2}] \rightarrow [256,\frac{H}{2},\frac{W}{2}]$ for stage 2, $[320,\frac{H}{4},\frac{W}{4}] \rightarrow [256,\frac{H}{4},\frac{W}{4}]$ for stage 3, and $[512,\frac{H}{8},\frac{W}{8}] \rightarrow [256,\frac{H}{8},\frac{W}{8}]$ for stage 4. We then interpolate four feature maps to the spatial size of $H \times W$. The probabilities of conducting the four augmentations in *RangeAug* are set as $[0.9, 0.2, 0.9, 1.0]$. For *RangePost*, we divide the whole scan into three "subclouds" for the 2D-to-3D re-rasterization.

Implementation Details. Following the conventional settings [46, 13], we conduct experiments with $W_{\text{train}} = 512, 1024, 2048$ on SemanticKITTI [5] and $W_{\text{train}} = 1920$ on nuScenes [21]. We use the AdamW optimizer [45] and OneCycle scheduler [56] with $lr = $ 1e-3. For *STR* training, we first partition points into 5 and 2 views and then rasterize them into range images of size 64×1920 ($W_{\text{train}} = 384$) and of size 32×960 ($W_{\text{train}} = 480$), for SemanticKITTI [5] and nuScenes [21], respectively. The models are pre-trained on Cityscapes [16] for 20 epochs and then trained for 60 epochs on SemanticKITTI [5] and ScribbleKITTI [65] and for 100 epochs on nuScenes [21], respectively, with a batch size of 32. Similar to [52, 13], we include the cross-entropy dice loss, Lovasz-Softmax loss [6], and boundary loss [52] to supervise the model training. All models can be trained on *single* NVIDIA A100/V100 GPUs for around 32 hours.

4.2. Comparative Study

Semantic Segmentation. Firstly, we compare the proposed *RangeFormer* with 13 prior and SoTA range view LiDAR segmentation methods on SemanticKITTI [5] (see Tab. 2). In conventional 512, 1024, and 2048 settings, we observe 9.3%, 9.8%, and 8.6% mIoU improvements over the SoTA method CENet [13] and 7.2% mIoU higher than MaskRange [25]. Such superiority is general for almost all classes and especially overt for dynamic and small-scale ones like *bicycle* and *motorcycle*. In Tab. 3, we further compare *RangeFormer* with 11 methods from other modalities. We can see that the current trend favors fusion-based methods which often combine the point and voxel views [30, 14]. Albeit using only range view, *RangeFormer* achieves the best scores so far; it surpasses the best fusion-based method 2DPASS [73] by 0.4% mIoU and the best voxel-only method GASN [75] by 2.9% mIoU. Similar observations also hold for nuScenes [21] (see Tab. 5).

STR Paradigm. As can be seen from the last three rows of Tab. 2, under the *STR* paradigm ($W_{\text{train}} = 384$), FIDNet [79] and CENet [13] have achieved even better scores compared to their high-resolution ($W_{\text{train}} = 2048$) versions. *RangeFormer* achieves 72.2% mIoU with *STR*, which is better than most of the methods on the leaderboard (see Tab. 3) while being 13.5% faster than the high training resolution

Table 2: Comparisons among state-of-the-art LiDAR **range view** semantic segmentation methods with different spatial resolutions (512, 1024, and 2048) on the *test* set of SemanticKITTI [5]. All IoU scores are given in percentage (%). For each resolution block: **bold** - best in column; underline - second best in column. Symbol †: $W_{\text{train}} = 384$.

#	**Method** (year)	mIoU	car	bicy	moto	truc	o.veh	ped	b.list	m.list	road	park	walk	o.gro	build	fenc	veg	trun	terr	pole	sign
64 × 512	RangeNet++ [46] ['19]	41.9	87.4	26.2	26.5	18.6	15.6	31.8	33.6	4.0	**91.4**	57.0	74.0	26.4	81.9	52.3	77.6	48.4	63.6	36.0	50.0
	MPF [2] ['21]	48.9	91.1	22.0	19.7	18.8	16.5	30.0	36.2	4.2	91.1	61.9	74.1	29.4	86.7	56.2	82.3	51.6	68.9	38.6	49.8
	FIDNet [79] ['21]	51.3	90.4	28.6	30.9	34.3	27.0	43.9	48.9	16.8	90.1	58.7	71.4	19.9	84.2	51.2	78.2	51.9	64.5	32.7	50.3
	CENet [13] ['22]	60.7	92.1	45.4	42.9	43.9	46.8	56.4	63.8	29.7	91.3	66.0	75.3	31.1	88.9	60.4	81.9	60.5	67.6	49.5	59.1
	RangeFormer	**70.0**	**94.7**	**60.5**	**70.2**	**58.4**	**64.6**	**72.8**	**73.0**	**55.4**	90.8	**70.4**	**75.4**	**39.9**	**90.7**	**66.6**	**84.6**	**68.6**	**70.5**	**59.4**	**63.6**
64 × 1024	RangeNet++ [46] ['19]	48.0	90.3	20.6	27.1	25.2	17.6	29.6	34.2	7.1	90.4	52.3	72.7	22.8	83.9	53.3	77.7	52.5	63.7	43.8	47.2
	MPF [2] ['21]	53.6	92.7	28.2	30.5	26.9	25.2	42.5	45.5	9.5	90.5	64.7	74.3	32.0	88.3	59.0	83.4	56.6	69.8	46.0	54.9
	FIDNet [79] ['21]	56.0	92.4	44.0	41.5	33.2	30.8	57.9	52.6	18.0	91.0	61.2	73.8	12.6	88.2	57.9	80.8	59.5	65.1	45.3	58.4
	CENet [13] ['22]	62.3	93.0	50.5	47.6	41.7	43.4	64.5	65.2	32.5	90.5	65.5	74.1	29.2	90.9	65.4	81.6	65.4	65.6	55.9	61.0
	RangeFormer	**72.1**	**95.7**	**66.2**	**72.9**	**59.8**	**66.5**	**75.8**	**74.5**	**56.5**	**91.8**	**71.9**	**77.4**	**41.6**	**91.6**	**68.9**	**85.8**	**71.5**	**71.6**	**64.2**	**65.8**
64 × 2048	SqSeg [68] ['18]	30.8	68.3	18.1	5.1	4.1	4.8	16.5	17.3	1.2	84.9	28.4	54.7	4.6	61.5	29.2	59.6	25.5	54.7	11.2	36.3
	SqSegV2 [69] ['19]	39.6	82.7	21.0	22.6	14.5	15.9	20.2	24.3	2.9	88.5	42.4	65.5	18.7	73.8	41.0	68.5	36.9	58.9	12.9	41.0
	RangeNet++ [46] ['19]	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
	SqSegV3 [71] ['20]	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
	3D-MiniNet [3] ['20]	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
	SalsaNext [17] ['20]	59.5	91.9	48.3	38.6	38.9	31.9	60.2	59.0	19.4	91.7	63.7	75.8	29.1	90.2	64.2	81.8	63.6	66.5	54.3	62.1
	KPRNet [34] ['21]	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
	LiteHDSeg [52] ['21]	63.8	92.3	40.0	55.4	37.7	39.6	59.2	71.6	54.3	93.0	68.2	78.3	29.3	91.5	65.0	78.2	65.8	65.1	59.5	**67.7**
	MPF [2] ['21]	55.5	93.4	30.2	38.3	26.1	28.5	48.1	46.1	18.1	90.6	62.3	74.5	30.6	88.5	59.7	83.5	59.7	69.2	49.7	58.1
	FIDNet [79] ['21]	59.5	93.9	54.7	48.9	27.6	23.9	62.3	59.8	23.7	90.6	59.1	75.8	26.7	88.9	60.5	84.5	64.4	69.0	53.3	62.8
	RangeViT [4] ['23]	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 [13] ['22]	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
	MaskRange [25] ['22]	66.1	94.2	56.0	55.7	59.2	52.4	67.6	64.8	31.8	91.7	70.7	77.1	29.5	90.6	65.2	84.6	68.5	69.2	60.2	66.6
	RangeFormer	**73.3**	**96.7**	**69.4**	**73.7**	**59.9**	**66.2**	**78.1**	**75.9**	**58.1**	92.4	73.0	78.8	**42.4**	**92.3**	**70.1**	**86.6**	**73.3**	**72.8**	**66.4**	66.6
STR†	**FIDNet *w/* STR**	60.1	93.6	48.8	44.4	45.0	38.4	58.1	65.5	7.0	**92.2**	68.3	76.2	27.4	88.1	61.3	82.8	61.0	69.5	55.6	58.4
	CENet *w/* STR	65.8	93.6	60.2	60.0	43.5	47.4	69.4	67.6	19.7	92.0	70.2	77.6	**43.6**	90.2	66.9	84.7	66.2	71.3	60.5	**65.4**
	RangeFormer *w/* STR	**72.2**	**96.4**	**67.1**	**72.2**	**58.8**	**67.4**	**74.9**	**74.7**	**57.5**	92.1	**72.5**	**78.2**	42.4	**91.8**	**69.7**	**85.8**	**70.4**	**72.3**	**62.8**	65.0

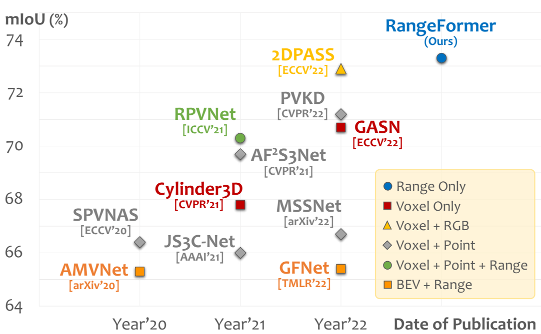


Table 3: State-of-the-art LiDAR semantic segmentation methods on the *test* set of SemanticKITTI [5].

Table 4: Comparisons among state-of-the-art LiDAR **panoptic segmentation** methods on the *test* set of SemanticKITTI [5]. All scores are given in percentage (%). For each metric: **bold** - best in column; underline - second best in column. RN denotes RangeNet++ [46]. PP denotes PointPillars [37]. Symbol †: $W_{\text{train}} = 384$.

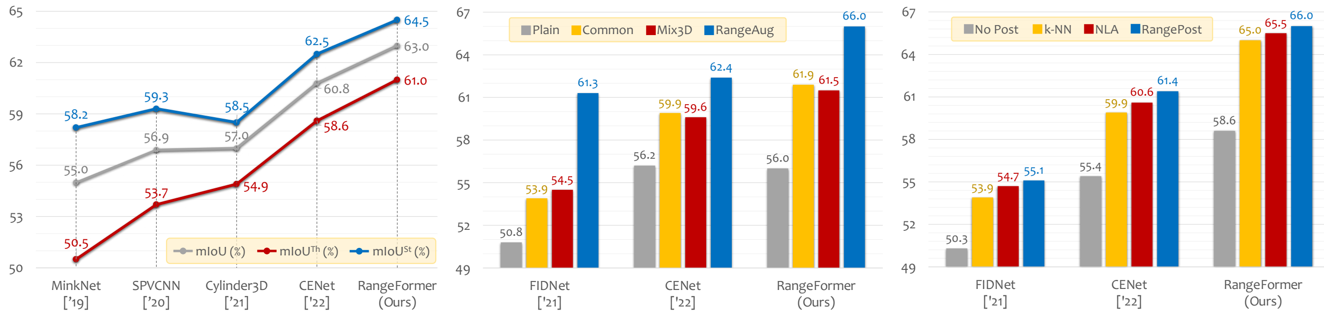
Method	PQ	PQ†	RQ	SQ	PQ$^{\text{Th}}$	RQ$^{\text{Th}}$	SQ$^{\text{Th}}$	PQ$^{\text{St}}$	RQ$^{\text{St}}$	SQ$^{\text{St}}$	mIoU
RN + PP	37.1	45.9	47.0	75.9	20.2	25.2	75.2	49.3	62.8	76.5	52.4
KPConv + PP	44.5	52.5	54.4	80.0	32.7	38.7	81.5	53.1	65.9	79.0	58.8
Panoster [22]	52.7	59.9	64.1	80.7	49.4	58.5	83.3	55.1	68.2	78.8	59.9
MaskRange [25]	53.1	59.2	64.6	81.2	44.9	53.0	83.5	59.1	73.1	79.5	61.8
P-PolarNet [80]	54.1	60.7	65.0	81.4	53.3	60.6	87.2	54.8	68.1	77.2	59.5
DS-Net [29]	55.9	62.5	66.7	82.3	55.1	62.8	87.2	56.5	69.5	78.7	61.6
EfficientLPS [55]	57.4	63.2	68.7	83.0	53.1	60.5	87.8	60.5	74.6	79.5	61.4
P-PHNet [39]	61.5	67.9	72.1	**84.8**	**63.8**	70.4	**90.7**	59.9	73.3	80.5	66.0
P-RangeFormer	**64.2**	**69.5**	**75.9**	83.8	63.6	**73.0**	86.8	**64.6**	**78.1**	**81.7**	**72.0**
***w/* STR†**	61.8	67.6	73.8	83.1	60.3	69.6	86.3	62.9	76.8	80.8	71.0

(*i.e.*, 2048) option (see Tab. 5) and saves 80% memory consumption. It is worth highlighting again that the convergence rate tends not to be affected. The *same* number of training epochs are applied to both *STR* and conventional training to ensure that the comparison is accurate.

Panoptic Segmentation. The advantages of *RangeFormer* in semantic segmentation have further yielded better panoptic segmentation performance. From Tab. 4 we can see that *Panoptic-RangeFormer* achieves better scores than the recent SoTA method Panoptic-PHNet [39] in terms of PQ, PQ†, and RQ. Such superiority still holds under the *STR* paradigm and is especially overt for the *stuff* classes. The ability to unify both semantic and instance LiDAR segmentation further validates the scalability of our framework.

Weakly-Supervised Segmentation. Recently, [65] adopts line scribbles to label LiDAR point clouds, which further saves the annotation budget. From Fig. 5a we can observe that the range view methods are performing much better than the voxel-based methods [15, 60, 81] under weak supervisions. This is credited to the compact and semantic-abundant properties of the range view, which maintains better representations for learning. Without extra modules or procedures, *RangeFormer* achieves 63.0% mIoU and exhibits clear advantages for both the *things* and *stuff* classes.

Accuracy *vs.* Efficiency. The trade-offs between segmentation accuracy and inference run-time are crucial for in-vehicle LiDAR segmentation. Tab. 5 summarizes the latency and mIoU scores of recent methods. We observe



(a) ScribbleKITTI Leaderboard (b) 3D Augmentation (c) Post-Processing

Figure 5: Comparative & ablation study. (a) Weakly-supervised LiDAR semantic segmentation results on the *val* set of ScribbleKITTI [65] (the same as SemanticKITTI [5]). (b) Results of different 3D data augmentation approaches on the *val* set of SemanticKITTI [5]. (c) Results of different post-processing methods on the *val* set of SemanticKITTI [5].

that the projection-based methods [78, 79, 13] tend to be much faster than the voxel- and fusion-based methods [51, 72, 81], thanks to the dense and computation-friendly 2D representations. Among all methods, *RangeFormer* yields the best-possible trade-offs; it achieves much higher mIoU scores than prior range view methods [79, 13] while being $2\times$ to $5\times$ faster than the voxel and fusion counterparts [73, 60, 72]. Furthermore, the range view methods also benefit from using pre-trained models on image datasets, *e.g.* ImageNet [18] and Cityscapes [16], as tested in Tab. 6.

Qualitative Assessment. Fig. 6 provides some visualization examples of SoTA range view LiDAR segmentation methods [79, 13] on sequence 08 of SemanticKITTI [5]. As clearly shown from the error maps, prior arts find segmenting sparsely distributed regions difficult, *e.g.*, *terrain* and *sidewalk*. In contrast, *RangeFormer* – which has the ability to model long-range dependencies and maintain large receptive fields – is able to mitigate the errors holistically. We also find advantages in segmenting object shapes and boundaries. More visual comparisons are in Appendix.

4.3. Ablation Study

Following [13, 71], we probe each component in *RangeFormer* with inputs of size 64×512 on the *val* set of SemanticKITTI [5]. Since our contributions are generic, we also report results on SoTA range view methods [79, 13].

Augmentation. As shown in Fig. 5b, data augmentations help alleviate data scarcity and boost the segmentation performance by large margins. The attention-based models are known to be more dependent on data diversity [19]. As a typical example, the "plain" version of *RangeFormer* yields a slightly lower score than CENet [13]. On all three methods, *RangeAug* helps to boost performance significantly and exhibits clear superiority over the common augmentations and the recent Mix3D [47]. It is worth mentioning that the extra overhead needed for *RangeAug* is negligible on GPUs.

Post-Processing. Fig. 5c attests again the importance of

Table 5: The **trade-off** comparisons between **efficiency** (run-time) and **accuracy** (mIoU). Symbol ♣: results on SemanticKITTI [5] *test* set. Symbol ★ / ■: results on nuScenes [21] *val* / *test* set. Latency is calculated on SemanticKITTI [5] and given in *ms*. Symbol †: $W_{\text{train}} = 384$ (SemanticKITTI) and 480 (nuScenes), respectively.

Method (year)	Size	Latency	♣	★	■	Modality
RangeNet++ [46]['19]	50.4M	126	52.2	65.5	–	Range Image
KPConv [61]['19]	18.3M	279	58.8	–	–	Bag-of-Points
MinkNet [15]['19]	21.7M	294	63.1	–	–	Sparse Voxel
SalsaNext [17]['20]	6.7M	71	59.5	72.2	–	Range Image
RandLA-Net [32]['20]	1.2M	880	53.9	–	–	Bag-of-Points
PolarNet [78]['20]	13.6M	62	57.2	71.0	69.4	Polar Image
AMVNet [41]['20]	–	–	65.3	76.1	77.3	Multiple
SPVNAS [60]['20]	12.5M	259	66.4	–	77.4	Sparse Voxel
Cylinder3D [81]['21]	56.3M	170	67.8	76.1	77.2	Sparse Voxel
FIDNet [79]['21]	6.1M	<u>16</u>	58.6	71.4	72.8	Range Image
AF2-S3Net [14]['21]	–	–	69.7	62.2	–	Multiple
RPVNet [72]['21]	24.8M	111	68.3	77.6	–	Multiple
2DPASS [73]['22]	–	62	<u>72.9</u>	–	<u>80.8</u>	Multiple
GFNet [51]['22]	–	100	65.4	76.8	–	Multiple
LidarMultiNet [74]['22]	–	–	–	**82.0**	**81.4**	Multiple
CENet [13]['22]	6.8M	**14**	64.7	73.3	74.7	Range Image
RangeViT [4]['23]	–	–	–	–	75.2	Range Image
RangeFormer	24.3M	37	**73.3**	<u>78.1</u>	80.1	Range Image
***w/* STR**†	24.3M	32	72.2	77.1	78.7	Range Image

Table 6: Effect of **pre-training** strategies on the *val* sets of SemanticKITTI [5] (left) and nuScenes [21] (right), with spatial sizes 64×2048 and 32×1920, respectively.

Method (year)	FIDNet [79]['21]	CENet [13]['22]	**RangeFormer**
No Pre-Train	$60.4_{+0.0}$ / $71.4_{+0.0}$	$63.4_{+0.0}$ / $73.3_{+0.0}$	$68.1_{+0.0}$ / $77.1_{+0.0}$
ImageNet	$61.6_{+1.2}$ / $72.1_{+0.7}$	$64.1_{+0.7}$ / $73.9_{+0.6}$	$68.9_{+0.8}$ / $77.6_{+0.5}$
Cityscapes	– / –	– / –	$69.6_{+1.5}$ / $78.1_{+1.0}$

post-processing in range view LiDAR segmentation. Without applying it, the "many-to-one" problem will cause severe performance drops. Compared to the widely-adopted k-NN [46] and the recent NLA [79], *RangePost* can better restore correct information since the aliasing among adjacent points has been reduced holistically. We also find the

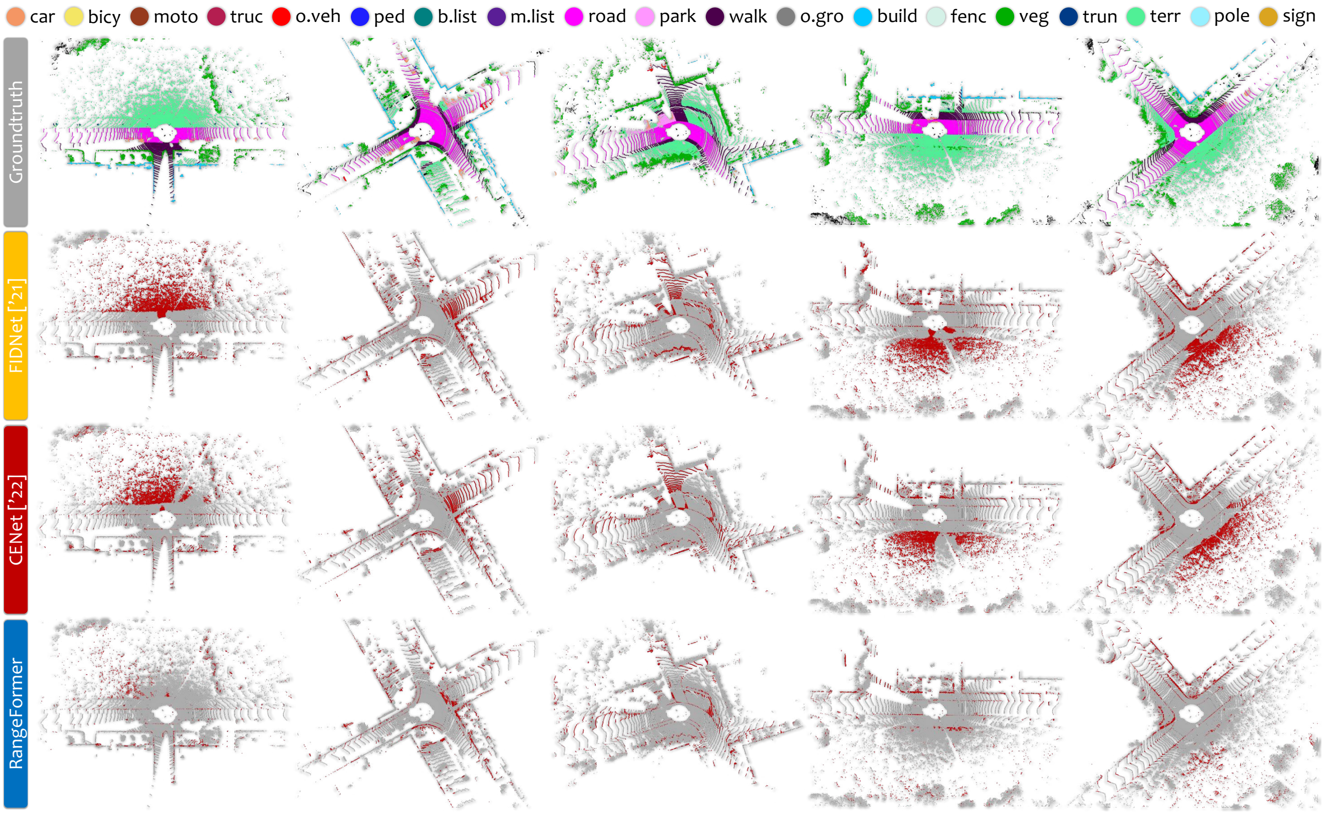


Figure 6: Qualitative comparisons of state-of-the-art range view LiDAR segmentation methods [79, 13]. To highlight the differences, the correct / **incorrect** predictions are painted in gray / **red**, respectively. Each point cloud scene covers a region of size 50m by 50m, centered around the ego-vehicle. Best viewed in colors.

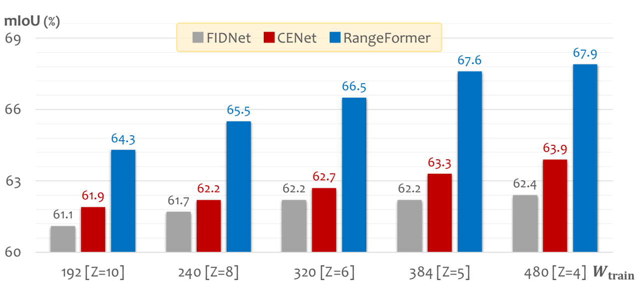


Figure 7: Exploring best-possible "view" partitions in STR.

extra overhead negligible since the "sub-clouds" are stacked along the batch dimension and can be processed in one forward pass. It is worth noting that such improvements happen after the training stage and are off-the-shelf and generic for various range view segmentation methods.

Scalable Training. To unveil the best-possible granularity in *STR*, we split the point cloud into 4, 5, 6, 8, and 10 views and show their results in Fig. 7. We apply the *same* training iteration for them hence their actual memory consumption becomes $\frac{1}{Z}$. We see that training on 4 or 5 views tends to yield better scores; while on more views the convergence rate will be affected, possibly by limited correlations in low-resolution range images. In summary, *STR* opens up a new training paradigm for range view LiDAR segmentation which better balances the accuracy and efficiency.

5. Conclusion

In this work, in defense of the traditional range view representation, we proposed *RangeFormer*, a novel framework that achieves superior performance than other modalities in both semantic and panoptic LiDAR segmentation. We also introduced *STR*, a more scalable way of handling LiDAR point cloud learning and processing that yields better accuracy-efficiency trade-offs. Our approach has promoted more possibilities for accurate in-vehicle LiDAR perception. In the future, we seek more lightweight self-attention structures and computations to further increase efficiency.

Acknowledgements. This study is supported by the Ministry of Education, Singapore, under its MOE AcRF Tier 2 (MOE-T2EP20221-0012), NTU NAP, and under the RIE2020 Industry Alignment Fund – Industry Collaboration Projects (IAF-ICP) Funding Initiative, as well as cash and in-kind contribution from the industry partner(s).

References

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

[2] Yara Ali Alnaggar, Mohamed Afifi, Karim Amer, and Mohamed ElHelw. Multi projection fusion for real-time semantic segmentation of 3d lidar point clouds. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1800–1809, 2021.

[3] Iñigo Alonso, Luis Riazuelo, Luis Montesano, and Ana C Murillo. 3d-mininet: Learning a 2d representation from point clouds for fast and efficient 3d lidar semantic segmentation. *IEEE Robotics and Automation Letters (RA-L)*, 5(4):5432–5439, 2020.

[4] Angelika Ando, Spyros Gidaris, Andrei Bursuc, Gilles Puy, Alexandre Boulch, and Renaud Marlet. Rangevit: Towards vision transformers for 3d semantic segmentation in autonomous driving. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5240–5250, 2023.

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

[6] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4413–4421, 2018.

[7] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11621–11631, 2020.

[8] 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 (T-PAMI)*, 40(4):834–848, 2018.

[9] Qi Chen, Sourabh Vora, and Oscar Beijbom. Polarstream: Streaming lidar object detection and segmentation with polar pillars. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[10] Runnan Chen, Youquan Liu, Lingdong Kong, Xinge Zhu, Yuexin Ma, Yikang Li, Yuenan Hou, Yu Qiao, and Wenping Wang. Clip2scene: Towards label-efficient 3d scene understanding by clip. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7020–7030, 2023.

[11] Yunlu Chen, Vincent Tao Hu, Efstratios Gavves, Thomas Mensink, Pascal Mettes, Pengwan Yang, and Cees GM Snoek. Pointmixup: Augmentation for point clouds. In *European Conference on Computer Vision (ECCV)*, pages 330–345, 2020.

[12] Bowen Cheng, Maxwell D. Collins, Yukun Zhu, Ting Liu, Thomas S. Huang, and Hartwig Adam. Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12475–12485, 2020.

[13] Huixian Cheng, Xianfeng Han, and Guoqiang Xiao. Cenet: Toward concise and efficient lidar semantic segmentation for autonomous driving. In *IEEE International Conference on Multimedia and Expo (ICME)*, 2022.

[14] Ran Cheng, Ryan Razani, Ehsan Taghavi, Enxu Li, and Bingbing Liu. Af2-s3net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12547–12556, 2021.

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

[16] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, 2016.

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

[18] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[19] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

[20] Lue Fan, Xuan Xiong, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Rangedet: In defense of range view for lidar-based 3d object detection. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2918–2927, 2021.

[21] Whye Kit Fong, Rohit Mohan, Juana Valeria Hurtado, Lubing Zhou, Holger Caesar, Oscar Beijbom, and Abhinav Valada. Panoptic nuscenes: A large-scale benchmark for lidar panoptic segmentation and tracking. *IEEE Robotics and Automation Letters (RA-L)*, 7:3795–3802, 2022.

[22] Stefano Gasperini, Mohammad-Ali Nikouei Mahani, Alvaro Marcos-Ramiro, Nassir Navab, and Federico Tombari. Panoster: End-to-end panoptic segmentation of lidar point clouds. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):3216–3223, 2021.

[23] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 32(11):1231–1237, 2013.

[24] Martin Gerdzhev, Ryan Razani, Ehsan Taghavi, and Liu Bingbing. Tornado-net: multiview total variation semantic segmentation with diamond inception module. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 9543–9549, 2021.

[25] Yi Gu, Yuming Huang, Chengzhong Xu, and Hui Kong. Maskrange: A mask-classification model for range-view based lidar segmentation. *arXiv preprint arXiv:2206.12073*, 2022.

[26] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 43(12):4338–4364, 2020.

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[28] Fangzhou Hong, Lingdong Kong, Hui Zhou, Xinge Zhu, Hongsheng Li, and Ziwei Liu. Unified 3d and 4d panoptic segmentation via dynamic shifting network. *arXiv preprint arXiv:2203.07186*, 2022.

[29] Fangzhou Hong, Hui Zhou, Xinge Zhu, Hongsheng Li, and Ziwei Liu. Lidar-based panoptic segmentation via dynamic shifting network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13090–13099, 2021.

[30] Yuenan Hou, Xinge Zhu, Yuexin Ma, Chen Change Loy, and Yikang Li. Point-to-voxel knowledge distillation for lidar semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8479–8488, 2022.

[31] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7132–7141, 2018.

[32] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11108–11117, 2020.

[33] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9404–9413, 2019.

[34] Deyvid Kochanov, Fatemeh Karimi Nejadasl, and Olaf Booij. Kprnet: Improving projection-based lidar semantic segmentation. *arXiv preprint arXiv:2007.12668*, 2020.

[35] Lingdong Kong, Niamul Quader, and Venice Erin Liong. Conda: Unsupervised domain adaptation for lidar segmentation via regularized domain concatenation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 9338–9345, 2023.

[36] Lingdong Kong, Jiawei Ren, Liang Pan, and Ziwei Liu. Lasermix for semi-supervised lidar semantic segmentation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21705–21715, 2023.

[37] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12697–12705, 2019.

[38] Dogyoon Lee, Jaeha Lee, Junhyeop Lee, Hyeongmin Lee, Minhyeok Lee, Sungmin Woo, and Sangyoun Lee. Regularization strategy for point cloud via rigidly mixed sample. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15900–15909, 2021.

[39] Jinke Li, Xiao He, Yang Wen, Yuan Gao, Xiaoqiang Cheng, and Dan Zhang. Panoptic-phnet: Towards real-time and high-precision lidar panoptic segmentation via clustering pseudo heatmap. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11809–11818, 2022.

[40] Ying Li, Lingfei Ma, Zilong Zhong, Fei Liu, Michael A Chapman, Dongpu Cao, and Jonathan Li. Deep learning for lidar point clouds in autonomous driving: A review. *IEEE Transactions on Neural Networks and Learning Systems (T-NNLS)*, 32(8):3412–3432, 2020.

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

[42] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022, 2021.

[43] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

[44] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015.

[45] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2018.

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

[47] Alexey Nekrasov, Jonas Schult, Or Litany, Bastian Leibe, and Francis Engelmann. Mix3d: Out-of-context data augmentation for 3d scenes. In *International Conference on 3D Vision (3DV)*, pages 116–125, 2021.

[48] Lorenzo Porzi, Samuel Rota Bulo, Aleksander Colovic, and Peter Kontschieder. Seamless scene segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8277–8286, 2019.

[49] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification

and segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, 2017.

[50] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5099–5108, 2017.

[51] Haibo Qiu, Baosheng Yu, and Dacheng Tao. Gfnet: Geometric flow network for 3d point cloud semantic segmentation. *Transactions on Machine Learning Research (TMLR)*, 2022.

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

[53] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[54] Jiawei Ren, Liang Pan, and Ziwei Liu. Benchmarking and analyzing point cloud classification under corruptions. In *International Conference on Machine Learning (ICML)*, 2022.

[55] Kshitij Sirohi, Rohit Mohan, Daniel Büscher, Wolfram Burgard, and Abhinav Valada. Efficientlps: Efficient lidar panoptic segmentation. *IEEE Transactions on Robotics (T-RO)*, 38(3):1894–1914, 2022.

[56] Leslie N. Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. *arXiv preprint arXiv:1708.07120*, 2017.

[57] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7262–7272, 2021.

[58] Yunzheng Su, Lei Jiang, and Jie Cao. Point cloud semantic segmentation using multi-scale sparse convolution neural network. *arXiv preprint arXiv:2205.01550*, 2022.

[59] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, and Benjamin Caine. Scalability in perception for autonomous driving: Waymo open dataset. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2446–2454, 2020.

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

[61] 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 *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6411–6420, 2019.

[62] Zhi Tian, Xiangxiang Chu, Xiaoming Wang, Xiaolin Wei, and Chunhua Shen. Fully convolutional one-stage 3d object detection on lidar range images. *arXiv preprint arXiv:2205.13764*, 2022.

[63] Larissa T Triess, David Peter, Christoph B Rist, and J Marius Zöllner. Scan-based semantic segmentation of lidar point clouds: An experimental study. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 1116–1121, 2020.

[64] Marc Uecker, Tobias Fleck, Marcel Pflugfelder, and J. Marius Zöllner. Analyzing deep learning representations of point clouds for real-time in-vehicle lidar perception. *arXiv preprint arXiv:2210.14612*, 2022.

[65] Ozan Unal, Dengxin Dai, and Luc Van Gool. Scribble-supervised lidar semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2697–2707, 2022.

[66] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

[67] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 568–578, 2019.

[68] 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 *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1887–1893, 2018.

[69] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue, and Kurt Keutzer. Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud. In *International Conference on Robotics and Automation (ICRA)*, pages 4376–4382. IEEE, 2019.

[70] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 12077–12090, 2021.

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

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

[73] Xu Yan, Jiantao Gao, Chaoda Zheng, Chao Zheng, Ruimao Zhang, Shuguang Cui, and Zhen Li. 2dpass: 2d priors assisted semantic segmentation on lidar point clouds. In *European Conference on Computer Vision (ECCV)*, pages 677–695, 2022.

[74] Dongqiangzi Ye, Zixiang Zhou, Weijia Chen, Yufei Xie, Yu Wang, Panqu Wang, and Hassan Foroosh. LidarmultiNet: Towards a unified multi-task network for lidar perception. *arXiv preprint arXiv:2209.09385*, 2022.

[75] Maosheng Ye, Rui Wan, Shuangjie Xu, Tongyi Cao, and Qifeng Chen. Efficient point cloud segmentation with geometry-aware sparse networks. In *European Conference on Computer Vision (ECCV)*, pages 196–212, 2022.

[76] Jinlai Zhang, Lyujie Chen, Bo Ouyang, Binbin Liu, Jihong Zhu, Yujin Chen, Yanmei Meng, and Danfeng Wu. Pointcutmix: Regularization strategy for point cloud classification. *Neurocomputing*, 505:58–67, 2022.

[77] Wenwei Zhang, Jiangmiao Pang, Kai Chen, and Chen Change Loy. K-net: Towards unified image segmentation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 10326–10338, 2021.

[78] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, and Hassan Foroosh. Polarnet: An improved grid representation for online lidar point clouds semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9601–9610, 2020.

[79] Yiming Zhao, Lin Bai, and Xinming Huang. Fidnet: Lidar point cloud semantic segmentation with fully interpolation decoding. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4453–4458. IEEE, 2021.

[80] Zixiang Zhou, Yang Zhang, and Hassan Foroosh. Panoptic-polarnet: Proposal-free lidar point cloud panoptic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13194–13203, 2021.

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