Hindawi
Computational Intelligence and Neuroscience
Volume 2022, Article ID 2286818, 9 pages
https://doi.org/10.1155/2022/2286818

Research Article

PyraPVConv: Efficient 3D Point Cloud Perception with Pyramid Voxel Convolution and Sharable Attention

Yuhong Chen,[1] Weilong Peng,[1] Keke Tang,[1] Asad Khan,[1] Guodong Wei,[2] and Meie Fang[1]

[1]*Guangzhou University, Guangzhou, China*
[2]*South China University of Technology, Guangzhou, China*

Correspondence should be addressed to Keke Tang; tangbohutbh@gmail.com and Meie Fang; fme@gzhu.edu.cn

Yuhong Chen and Weilong Peng contributed equally to this work.

Received 5 January 2022; Revised 4 February 2022; Accepted 7 February 2022; Published 13 May 2022

Academic Editor: Diego Oliva

Designing efficient deep learning models for 3D point cloud perception is becoming a major research direction. Point-voxel convolution (PVConv) Liu et al. (2019) is a pioneering research work in this topic. However, since with quite a few layers of simple 3D convolutions and linear point-voxel feature fusion operations, it still has considerable room for improvement in performance. In this paper, we propose a novel pyramid point-voxel convolution (PyraPVConv) block with two key structural modifications to address the above issues. First, PyraPVConv uses a voxel pyramid module to fully extract voxel features in the manner of feature pyramid, such that sufficient voxel features can be obtained efficiently. Second, a sharable attention module is utilized to capture compatible features between multi-scale voxels in pyramid and point cloud for aggregation, as well as to reduce the complexity via structure sharing. Extensive results on three point cloud perception tasks, i.e., indoor scene segmentation, object part segmentation and 3D object detection, validate that the networks constructed by stacking PyraPVConv blocks are efficient in terms of both GPU memory consumption and computational complexity, and are superior to the state-of-the-art methods.

1. Introduction

With the advance of depth sensing devices, 3D point clouds can be captured in a much easier manner. Therefore, applications of 3D point cloud perception are now booming, e.g., simultaneous localization and mapping (SLAM) [1–3], and autonomous driving [4–6]. In the last few decades, 3D point cloud perception mainly depends on hand-crafted shape descriptors [7–9]. Until very recently, researchers start to extend deep learning models which are mature in the field of 2D computer vision to handle 3D perception tasks, significantly refreshing the state-of-the-art records [10–12]. However, 3D deep learning models with higher accuracies always have higher complexities, which now becomes the major obstacle for their application to real-world scenarios.

There have been many attempts that utilize deep networks to perceive 3D point clouds. According to the representation of data that feeds to deep networks, these methods could be divided into two categories broadly: structure-based methods and point-based methods. Structure-based methods first convert irregular point clouds into structured grid representations, e.g., projecting point clouds into bird's eye views [13, 14] or rasterizing into 3D voxel grids [10, 15–17], and then adopt traditional 2D convolutional neural networks (CNNs) or their simple extensions to extract discriminative CNN features. However, it introduces exponential computational cost and memory for detailed 3D geometric learning at high resolutions. Point-based methods instead impose multi-layer perceptrons (MLPs) followed with maximum pooling [11, 18] or irregular kernels [19] to 3D point clouds, such that they can handle points directly. Nevertheless, since these methods require accessing irregularly scattered points especially for local feature aggregation, they are also inefficient.

By analysing both the advantages and disadvantages of structure-based and point-based methods, the pioneering point-voxel convolution (PVConv) [20] proposes to combine them together for efficiency purpose. Particularly, point-based MLPs are adopted to extract 3D features in the point branch; and voxel convolutions aggregate local features coarsely in the voxel branch; in addition, linear interpolation is conducted to fuse the features of the two branches. PVConv brings large improvements in terms of GPU memory consumption and computational efficiency, but the accuracy of PVConv is somewhat sacrificed. By going through the structure of PVConv, we hypothesize that there are two key factors that limit its performance. First, the voxel-based networks with strong feature extraction capabilities are utilized too conservatively, i.e., with only two layers of simple 3D convolutions. Second, the voxel features that are already insufficient would further lose during the fusion process, since it is implemented via a linear-based interpolation that is not powerful enough.

To resolve the above two issues, we intentionally design a novel pyramid point-voxel convolution (PyraPVConv) block. First, we adopt a more powerful 3D voxel convolution branch that extracts multi-scale voxel features in the form of feature pyramid, such that powerful 3D voxel convolutions can be fully exploited with sacrificing moderate additional computational overhead. Second, to alleviate the information loss during the feature fusion process between point and voxel branches, we propose utilizing the attention mechanism to learn to combine them in a more compatible way. Particularly, we design a sharable attention module that learns the relevant scores between multiple voxel branches and the point branch with sharing a structure, such that it reduces the overhead required by multibranch attention. With these two designs against the weaknesses of PVConv, the PyraPVConv block can perceive 3D point clouds in a better tradeoff between accuracy and efficiency.

Overall, our contribution is three-fold.

(i) We propose a lightweight 3D perception block, PyraPVConv, that can perform 3D point cloud perception accurately and efficiently

(ii) We design a voxel pyramid module, which better extracts voxel features without introducing too much additional computational overhead

(iii) We devise a sharable attention module that fuses the features of point branch and multiple voxel branches in a more effective nonlinear manner

We construct PyraPVCNN by stacking multiple PyraPVConv blocks following PVCNN [20], and evaluate it on various point cloud perception tasks, e.g., indoor scene and object part segmentation, 3D object detection. Extensive experiments demonstrate the superiority of PyraPVConv to the state-of-the-art methods in terms of both accuracy and efficiency.

2. Related Work

2.1. Efficient Deep Learning for Point Clouds. Deep learning techniques have been widely adopted for handling 3D point cloud perception tasks, e.g., classification and segmentation, by projecting sparse point clouds into compact semantic representations [21–25]. However, as indicated in [20], most current 3D deep learning methods for point clouds are less efficient, e.g., voxel-based methods require large amounts of memory for maintaining detailed 3D structures, and point-based methods require high CPU latency to search neighborhood points for feature aggregation.

To perceive 3D point clouds in a more efficient way, Liu et al. [20] proposed the point-voxel convolution (PVConv), that combines point-based MLPs for individual point feature extraction and voxel-based CNNs for neighborhood feature aggregation. Since with only two layers of simple 3D CNNs, the performance of PVConv is somewhat sacrificed. To better utilize the voxel information, Tang et al. [26] further designed sparse point-voxel convolution (SPVConv), which uses sparse convolution to handle high-resolution voxels on the basis of PVConv, and further adopted network architecture search (NAS) for searching the best architecture. Although their method showed good performance in large-scale scenarios, sparse convolution is complex and is less efficient in common scenes. Our method also aims to fully utilize 3D voxel information. Differently, we adopt the concept of feature pyramid to balance the accuracy and efficiency.

We also notice some other works that also aim to efficient point cloud learning by randomly key point sampling [27], by irregularly volume partition [28], and by leveraging mature 2D methods [29], etc., but these directions are out of our scope.

2.2. Multiscale Feature Modeling. Modeling multi-scale features has been validated to be a useful strategy in computer vision, e.g., maintaining scale-invariant property in SIFT [30] and controlling the receptive field in CNNs [31]. This strategy is also widely adopted in 3D deep learning. PointNet++ [18] extracts multi-scale features of point clouds by hierarchically applying PointNets [11]. 3D object detection frameworks [17, 32, 33] adopt different detection heads with multi-scale feature maps to handle both large and small object classes. ContFuse [34] uses continuous convolution to aggregate multi-scale feature maps from different ResNet blocks [35]. By extending RPN-FPN module [36] to 3D, Voxel-FPN [37] uses feature pyramid to aggregate voxel features of different voxelization resolutions. We also utilize the multi-scale feature modeling strategy, but are to balance the accuracy and efficiency of the voxel branch for point-voxel convolution, which has not been investigated before.

2.3. Attention Mechanism. Attention is originally a physiological mechanism that describes the phenomenon that humans' perception system could focus on the object of interest while suppressing the background [38]. Inspired by it, deep learning researchers attempt to exploit it to analyze networks' focus [39, 40] or enforce neural networks to focus on more important features [41, 42].

Attention is also widely adopted to model the relationship between two instances, e.g., generating the most relevant sentences for images in the task of image caption [43] or searching the most relevant sentences between two different languages in the task of machine translation [44]. By projecting the query instance into the same high-dimensional space as the target, relevant spatial regions of the query will be highlighted to guide the desired inference. We also aim to model the relationship. Differently, our inferred relevance is used for the fusion of features extracted from the voxel branch and point branch during point-voxel convolution.

3. Method

In this section, we will first review the architecture of PVConv, analyze the factors that may limit its performance and then introduce the overview of our solution, i.e., pyramid point-voxel convolution (PyraPVConv). After that, we describe the two main components of PyraPVConv: the voxel pyramid module and the sharable attention module.

3.1. Review of PVConv [20]. Methodology of PVConv Given an unordered point set $\mathbf{P} = \{(p_k, f_k)\}$ with $\{p_k\}$ denoting the point coordinates and $\{f_k\}$ as the point features, PVConv adopts an isolated *point branch* to extract individual point features using MLPs similar to PointNet. Apart from that, another *voxel branch* in PVConv is utilized to facilitate efficient and powerful neighbourhood feature aggregation. Particularly, the feeded voxels to the voxel branch are generated by first normalizing the point coordinates and then conducting voxelization to transform the normalized point cloud $\{(\widehat{p}_k, f_k)\}$ into a volume $\mathbf{V}$ by averaging all features $\{f_k\}$ whose normalized coordinates $\{\widehat{p}_k\}$ fall into that voxel grid.

Finally, the two-branch features are linearly fused, i.e., the voxel features are devoxelized to the point cloud domain using trilinear interpolation, and then added with point features.

Discussion on PVConv As mentioned in PVConv, the MLPs in the *point branch* can already output discriminative features for each point. Therefore, the main contribution of PVConv is to leverage *the voxel branch* for neighborhood feature aggregation.

However, perhaps focusing too much on the efficiency factor, PVConv is *conservative* in utilizing the *voxel branch*, e.g., very limited volume resolutions and 3D convolution layers are adopted. Indeed, 3D convolutions that are derived from mature 2D convolution techniques are powerful, and have a good tradeoff between accuracy and efficiency. Furthermore, simple trilinear interpolation would lead to information loss of those voxel features that are not sufficient originally.

3.2. Overview of PyraPVConv. To resolve the above issues of PVConv, we propose a novel PyraPVConv block with two key structural modifications: a voxel pyramid module for the voxel branch and a sharable attention module for feature fusion. Similar as in PVConv, the point branch of PyraPVConv extracts individual point features. At the same time, the voxel branch of PyraPVConv adopts the voxel pyramid module to extract more powerful voxel features in pyramid. Then, the shareable attention module fuses point features and voxel features in pyramid nonlinearly. Please refer to Figure 1 for a demonstration.

3.3. Voxel Pyramid Module. To extract sufficient voxel features without bringing too much additional computational overhead, we propose a voxel pyramid module to capture multiscale features in different pyramid levels, inspired by [36]. Please refer to Figure 2 for a demonstration.

Volume Generation Given the normalized point cloud $\{(\widehat{p}_k, f_k)\}$ as in PVConv, we generate a volume by averaging all features $\{f_k\}$ whose coordinates $\{\widehat{p}_k\}$ fall into that voxel grid via the following equation:

$$\begin{aligned} V_r(u,v,w) = \frac{1}{N_{r,u,v,w}} \sum_{k=1}^{N} \mathbb{I}(\lfloor x_k \times r \rfloor \\ = u, \lfloor y_k \times r \rfloor = v, \lfloor z_k \times r \rfloor = w) \times f_{k,c}, \end{aligned} \quad (1)$$

where r denotes the volume resolution, $N_{r,u,v,w}$ denotes the number of points falling in the voxel grid (u,v,w) of volume $\mathbf{V}_r$, $f_{k,c}$ denotes the c^{th} channel corresponding to $\widehat{p}_k$, and $\mathbb{I}$ is a binary indicator function.

Bottom-up Feature Extraction Given $\mathbf{V}_r$, we feed it to the same 3D voxel convolution networks as in PVConv, i.e., two groups of conv3d, batch normalization and activation layers, to obtain 3D voxel feature $f_r^{\mathbf{V}}$,

$$f_r^{\mathbf{V}} = \text{Conv3D}_1(\mathbf{V}_r). \quad (2)$$

Unlike PVConv that adopts a conservative strategy in the voxel branch, we suggest making full use of the features in volume $\mathbf{V}_r$. Considering the efficiency, we conduct an average pooling operation with a scaling rate of 1/2 to $f_r^{\mathbf{V}}$, and then feed it another 3D voxel convolution networks,

$$f_{r/2}^{\mathbf{V}} = \text{Conv3D}_2\left(\text{MaxPool}\left(f\begin{pmatrix}\mathbf{V}\\ r\end{pmatrix}\right)\right). \quad (3)$$

We apply the same operations iteratively to obtain multi-scale 3D voxel features. In this paper, we adopt three-scale features $f_r^{\mathbf{V}}$, $f_{r/2}^{\mathbf{V}}$, and $f_{r/4}^{\mathbf{V}}$.

To-down Feature Aggregation Given each 3D voxel feature, we enhance it by aggregating with the feature of a higher pyramid level (if has) that is spatially coarser but semantically strong. It is implemented by applying an upsampling operation followed with addition,

$$\begin{aligned} \widehat{f}_{r/4}^{\mathbf{V}} &= f_{r/4}^{\mathbf{V}}, \\ \widehat{f}_{r/2}^{\mathbf{V}} &= \text{UpPool}\left(\widehat{f}_{r/4}^{\mathbf{V}}\right) + f_{r/2}^{\mathbf{V}}, \\ \widehat{f}_{r}^{\mathbf{V}} &= \text{UpPool}\left(\widehat{f}_{r/2}^{\mathbf{V}}\right) + f_{r}^{\mathbf{V}}. \end{aligned} \quad (4)$$

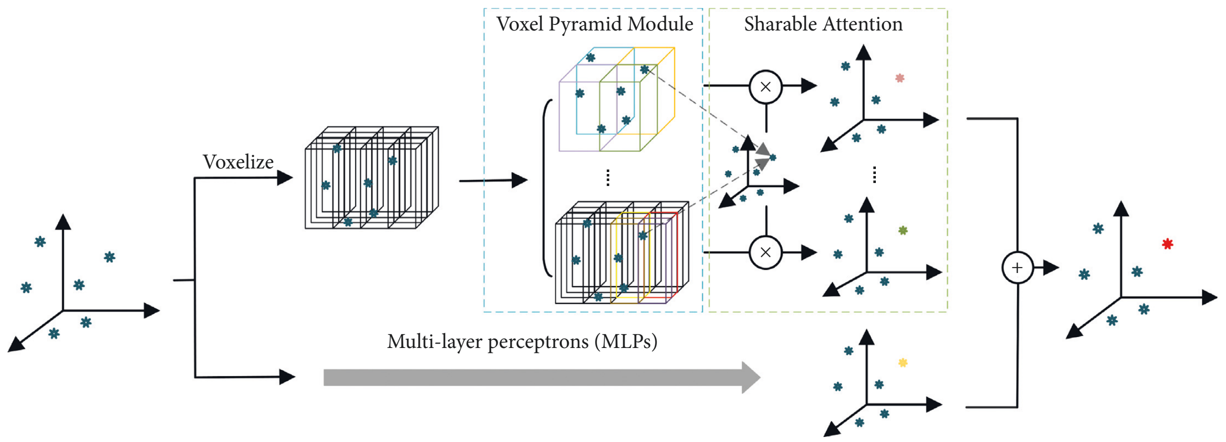

FIGURE 1: The architecture of PyraPVConv, with a point branch extracting single point features, and a voxel branch aggregating neighborhood information via a voxel pyramid module and an sharable attention mechanism.

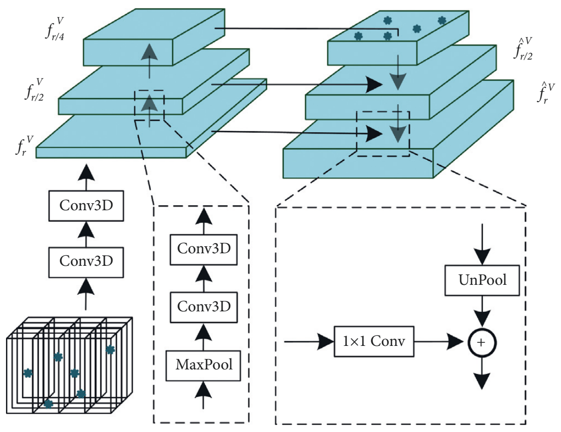

FIGURE 2: Demonstration of the bottom-up feature extraction and top-down feature aggregation in the voxel pyramid module.

Therefore, we obtain three semantic enhanced voxel features: $\hat{f}_r^{\mathbf{V}}$, $\hat{f}_{r/2}^{\mathbf{V}}$, and $\hat{f}_{r/4}^{\mathbf{V}}$.

3.4. Sharable Attention Module. With enhanced voxel features $\hat{f}_r^{\mathbf{V}}$, $\hat{f}_{r/2}^{\mathbf{V}}$, $\hat{f}_{r/4}^{\mathbf{V}}$ and point features $\{f_k\}$ as inputs, we fuse them via using the attention mechanism. Since voxel features in different pyramid levels contain discriminative but complementary information, we apply attentive fusion of them with point features separately. It is implemented by converting the voxel features into the same feature space as point features, and then summarize them. See Figure 3 for a demonstration. In the following, we will describe the attention method taking $\hat{f}_r^{\mathbf{V}}$ as an example.

Attentive Fusion for a Pyramid Level Instead of fusing all voxel features, we only consider eight neighboring voxel features $\left\{\hat{f}_r^{\mathbf{V}}(k,m)\right\}$ with $m = 1, 2, \ldots, 8$ for each point $\{(\hat{p}_k, f_k)\}$ following PVConv, considering the tradeoff between efficiency and performance. Specifically, we adopt a similar attention process to that in [45] with three key steps. First, we project point feature f_k and its eight neighboring voxel features $\left\{\hat{f}_r^{\mathbf{V}}(k,m)\right\}$ into the same feature space using two networks Q_r and K_r. Second, we calculate the relevant scores between eight voxel features and that point feature by conducting dot product of their flattened features. Note that, we also adopt the Sigmoid operation to enforce the relevant scores to be between 0 and 1.

$$\mathrm{Rel}\left(f_k, \hat{f}_r^{\mathbf{V}}(k,m)\right) = \mathrm{Sigmoid}\left(Q_r(f_k) \times K_r\left(\hat{f}_r^{\mathbf{V}}(k,m)\right)\right). \quad (5)$$

Finally, we convert the voxel features $\left\{\hat{f}_r^{\mathbf{V}}(k,m)\right\}$ into a summable space with point features using the network H_r, and then aggregate them weighted with the relevant scores $\mathrm{Rel}(f_k, \hat{f}_r^{\mathbf{V}}(k,m))$.

$$f_k' = f_k + \sum_m H_r\left(\hat{f}_r^{\mathbf{V}}(k,m)\right)\mathrm{Rel}\left(f_k, \hat{f}_r^{\mathbf{V}}(k,m)\right). \quad (6)$$

Multiple Pyramid Fusion with Sharable Attention For voxel features in all three pyramid levels, we apply the above attentive fusion method, and then summarize them together.

$$\hat{f}_k' = f_k + \sum_r \sum_m H_r\left(\hat{f}_r^{\mathbf{V}}(k,m)\right)\mathrm{Rel}\left(f_k, \hat{f}_r^{\mathbf{V}}(k,m)\right). \quad (7)$$

Therefore, $\hat{f}_k'$ is the final fused feature that contains both point and voxel features. Particularly, since the inputs for networks Q_t with $t = r, r/2, r/4$ are exactly the same, we simply use one network Q and share it with all three pyramid levels for efficiency purpose.

4. Experimental Results

To validate the effectiveness and efficiency of PyraPVConv, we extensively evaluate the performance of PyraPVCNN, which is constructed by stacking multiple PyraPVConv blocks, on three different tasks, i.e., indoor scene segmentation, object part segmentation and 3D object detection.

4.1. Implementations. For PyraPVCNN, we replace all the PVConv blocks in PVCNN with PyraPVConv and use the same decoding layer. We implement PyraPVCNN and reproduce all the evaluated networks with PyTorch for fair comparisons, and report the latency and memory

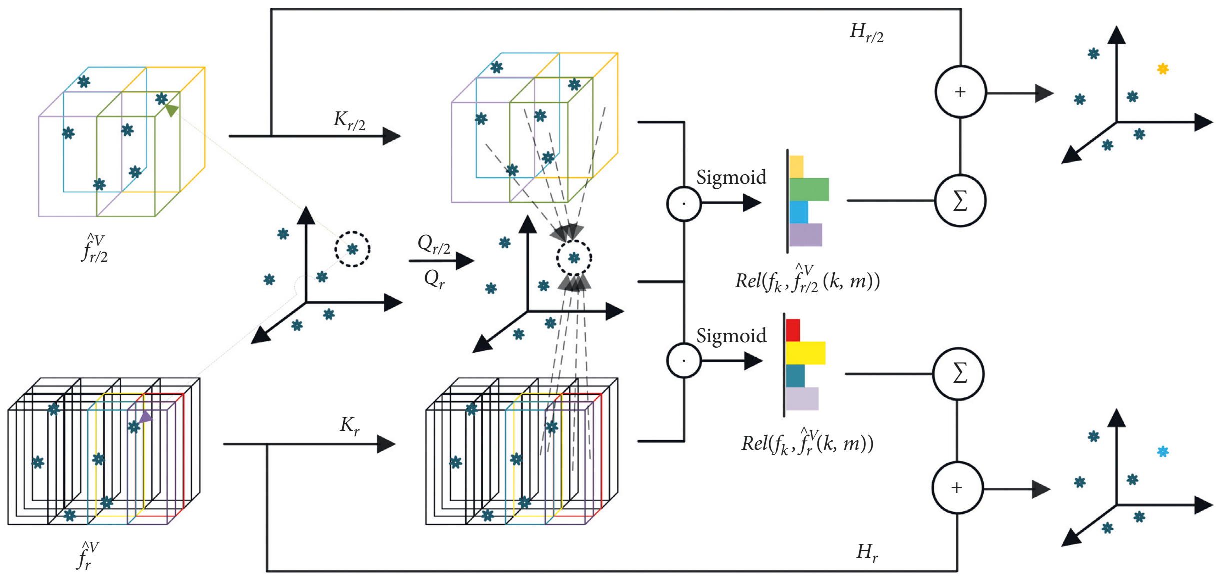

Figure 3: Demonstration of the sharable attention module.

consumption at test time on a workstation with an Intel Xeon E5-2678 CPU@2.5 Hz and 64 GB of memory using a single RTX 2080Ti GPU.

4.2. Indoor Scene Segmentation. Setups We conduct semantic segmentation experiments on the large-scale Stanford 3D semantic parsing dataset (S3DIS) [47, 48]. S3DIS is scanned across 271 rooms in 6 areas with a Matterport camera (combined with 3 structured light sensors at different intervals), and then each point in the scanned point cloud is annotated with a semantic label (e.g., a total of 13 objects such as chairs, tables, floors, walls, etc.). We adopt the same data processing procedure as PVCNN [20]. We train the models on regions 1, 2, 3, 4, and 6 and test them on region 5 to evaluate the mean intersection-over-union (mIoU) and the mean class-wise accuracy (mAcc), with both metrics measured as percentages.

Models Six state-of-the-art methods are adopted as baselines: PointNet [20], PointNet++ [18], DGCNN [25], PointCNN [46], PVCNN [20] and RandLA-Net [27]. Since our focus is on the efficiency of deep networks, similar as in PVCNN, we also evaluate the narrower versions of PVCNN and PyraPVCNN by reducing the number of feature channels from that of the original version (denoted as $1 \times C$) to 12.5% (denoted as $0.125 \times C$), 25% (denoted as $0.25 \times C$) and 50% (denoted as $0.5 \times C$).Comparison Results The results reported in Table 1 show that Ours $(0.125 \times C)$ performs better than two state-of-the-art point-based methods: PointNet [11] and DGCNN [25] and voxel-based methods: PVCNN $(0.25 \times C)$ [20], and is comparable to the state-of-the-art RandLA-Net [27], but with less latency and GPU memory consumption. In addition, the performance of Ours $(1 \times C)$ is nearly 10% higher than that of DGCNN [25], but the latency is reduced by 20%, and GPU memory consumption is reduced by 1.6 times. In particular, Ours $(0.25 \times C)$ outperforms the full version of PVCNN, but the latency and GPU memory consumption are much less. Please refer to Figure 4 to see an illustrative demonstration of the tradeoff between accuracy (mIoU) and the incurred overhead (the number of parameters, latency and GPU memory consumption).

We also visualize the segmentation results in Figure 5. It could be seen that Ours $(0.25 \times C)$ can better utilize neighborhood information to improve the prediction of point labels, compared with PVCNN.

Ablation Studies To validate the importance of the two key modules of PyraPVConv: voxel pyramid module and the sharable attention module, we report the results of ablation studies using two narrow versions of the PyraPVCNN (i.e., $0.25 \times C$ and $0.5 \times C$) in Table 2. It could be seen that the performance will drop, if we delete any one of them. In particular, the voxel pyramid module has a slightly larger impact on the performance.

4.3. Object Part Segmentation. Setups We choose the ShapeNet Parts dataset [49] to conduct the experiment of object part segmentation. ShapeNet Parts is a collection of 16681 point clouds selected from 16 categories of the ShapeNetCore, and is manually annotated with a total of 50 parts. Following [46], we train the models on 14006 point clouds, evaluate the part-averaged IoU for each of the remaining 2874 point clouds, and then average them as the final metric, i.e., mIoU (%).

Models We choose point-based models: PointNet [11], PointNet++ [18], and DGCNN [25] and the voxel-based model: 3D-UNet [50], and PVCNN [20] as baselines.

Comparison Results As shown in Table 3, Ours $(0.25 \times C)$ performs much better while the latency is 36 times lower and the GPU memory consumption is 11.7% lower than that of 3D-UNet. Moreover, it is superior to the most advanced point-based methods such as PointNet [11], PointNet++ [18] and DGCNN [25] in all aspects. In particular, we would like to emphasize that Ours $(0.5 \times \mathrm{C})$ outperforms PVCNN [20], and

Table 1: Indoor scene segmentation results on the S3DIS dataset. Note that the input data of PointCNN [46] include 16 × 2048 points, while the data of the other methods include 8 × 4096 points.

	mAcc	mIoU	Latency (ms)	GPU mem. (GB)	#Param.
PointNet	82.54	42.97	16.1	1.50	3.53 M
PVCNN $(0.125 \times C)$	82.60	46.94	7.6	0.46	43.16 K
DGCNN	83.64	47.94	79.0	4.50	987.00 K
PVCNN $(0.25 \times C)$	84.52	51.96	11.4	0.76	166.21 K
Ours $(0.125 \times C)$	84.88	**52.16**	10.3	0.59	143.19 K
PVCNN $(0.5 \times C)$	85.88	54.73	17.5	0.97	650.35 K
PVCNN	86.25	55.54	35.9	1.92	2.57 M
Ours $(0.25 \times C)$	86.49	**55.86**	21.6	1.04	693.72 K
Ours$(0.5 \times C)$	86.93	57.02	41.3	1.84	1.82 M
PointCNN	85.91	57.26	282.3	4.60	5.86 M
RandLA-Net	85.10	**58.60**	911.1	2.57	4.76 M
Ours $(1 \times C)$	**86.96**	57.98	71.2	2.52	3.13 M

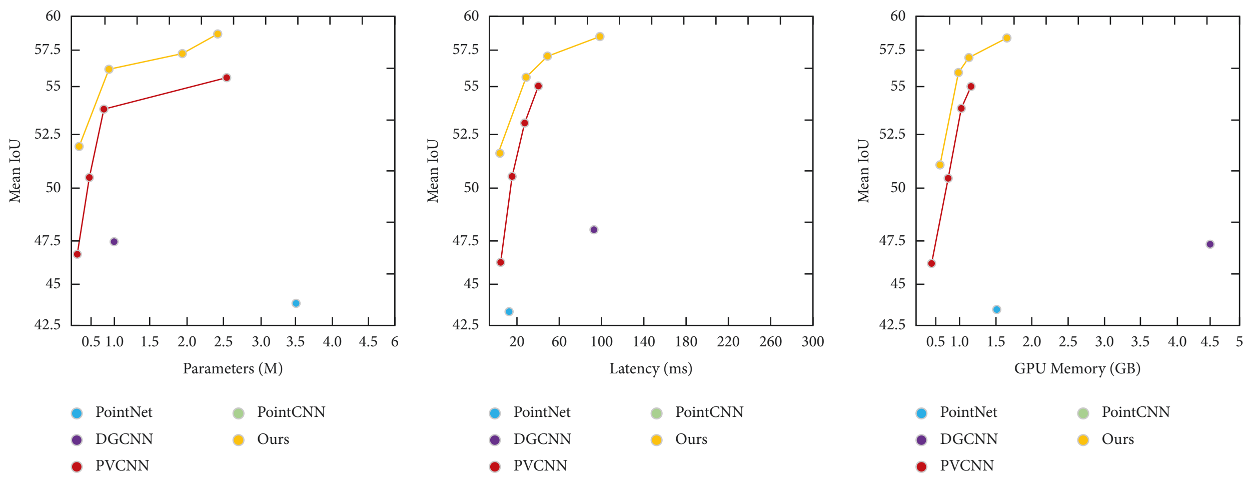

Figure 4: The tradeoff between accuracy and the number of parameters, measured latency, and GPU memory consumption for PyraPVCNN and the state-of-the-art baselines on S3DIS. (a) Parameters (M), (b) Latency (ms), (c) GPU memory (GB).

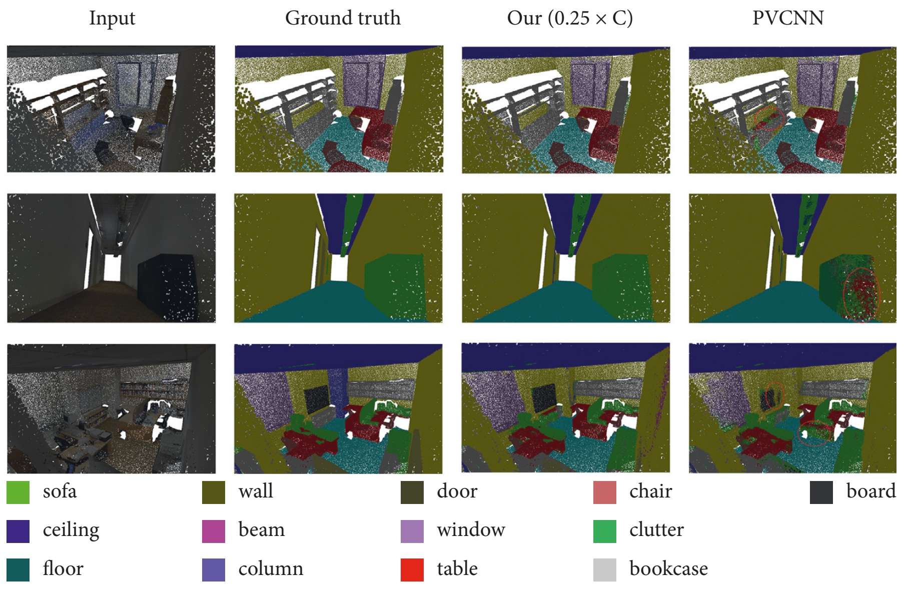

Figure 5: Visualization of the semantic segmentation results on the S3DIS dataset.

Table 2: The indoor scene segmentation performance on the S3DIS dataset with ablating the voxel pyramid module (VPM.) and the sharable attention module (ASM.).

Channel	Metric	Ours	W/o VPM.	W/o ASM.
$0.25 \times C$	mIoU	**55.86**	53.01	53.87
	mAcc	**86.49**	84.91	85.56
$0.5 \times C$	mIoU	**57.02**	55.07	55.68
	mAcc	**86.93**	85.67	86.31

Table 3: Object part segmentation results on the ShapeNet Parts dataset.

	Input data	Convolution	mIoU	Latency (ms)	GPU mem. (GB)
PointNet	Points (8×4096)	None	83.70	16.0	1.5
3D-UNet	Voxels (8×963)	Voxel-based	84.60	480.0	17.6
PointNet++	Points (8×4096)	Point-based	84.70	55.6	4.0
DGCNN	Points (8×4096)	Point-based	84.70	61.8	4.8
PVCNN ($0.25 \times$ C)	Points (8×4096)	Voxel-based	84.72	10.1	1.2
Ours($0.25 \times$ C)	Points (8×4096)	Voxel-based	**85.12**	13.2	1.5
PVCNN ($0.5 \times$ C)	Points (8×4096)	Voxel-based	85.38	17.5	1.8
PVCNN	Points (8×4096)	Voxel-based	85.70	35.2	3.1
Ours ($0.5 \times$ C)	Points (8×4096)	Voxel-based	**85.97**	29.7	2.4
PointCNN	Points (16×2048)	Point-based	86.10	95.6	5.1
Ours ($1 \times$ C)	Points (8×4096)	Voxel-based	**86.82**	56.7	3.2

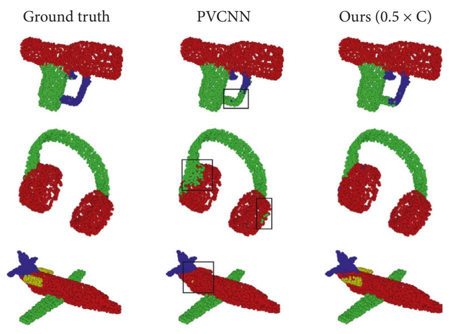

Figure 6: Visualization of object part segmentation results on the ShapeNet Parts dataset.

Table 4: 3D object detection results of the Frustum Networks [29] on the validation set of KITTI with different backbones.

Backbone	Efficiency		Car			Pedestrian			Cyclist		
	Latency (ms)	GPU mem. (GB)	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
PointNet	29.1	1.5	83.26	69.28	62.56	65.08	55.85	49.28	74.54	55.95	52.65
PointNet++	101.2	2.5	83.76	70.92	63.65	70.00	61.32	53.59	77.15	56.49	53.37
PVCNN	51.9	1.9	84.22	71.11	**63.66**	69.16	60.28	52.52	78.67	57.79	54.16
Ours ($0.25 \times$ C)	**39.6**	**1.4**	**85.42**	**71.24**	63.09	**71.80**	**64.67**	**56.87**	**79.32**	**58.57**	**55.01**

Ours ($1 \times C$) outperforms PointCNN [46] with only about 58% latency and GPU memory consumption. Please refer to the visualization results in Figure 6 for intuitive comparisons.

4.4. 3D Object Detection. Setups We conduct 3D object detection experiments on the large outdoor dataset KITTI [51]. KITTI provides 7481 training and verification samples for objects such as cars, pedestrians, and cyclists, of which 3711 samples are used for training and the remaining 3769 samples are for verification. We evaluate all models 20 times and report the average 3D average accuracy (AP) measured as percentages.

Models We build multiple Frustum Networks [29] with PointNet [11], PointNet++ [18], PVCNN [20], a narrow version of PyraPVCNN ($0.25 \times C$) as backbones.

Comparison Results As shown in Table 4, the Frustum Network with PyraPVCNN ($0.25 \times$ C) (denoted as F-PyraPVCNN) as backbone outperforms than all three other Frustum Networks with PointNet [11], PointNet++ [18] and PVCNN [20] as backbones (denoted as F-PointNet, F-PointNet++ and F-PVCNN), but with much less latency and GPU memory consumption. In particular, F-PyraPVCNN ($0.25 \times C$) improves the detection rate of pedestrian by 4% compared with the F-PVCNN, and 7.5%

compared with the F-PointNet, validating the usefulness of PyraPVCNN in 3D detection tasks.

5. Conclusion

In this paper, we have presented a novel PyraPVConv block with the aim of efficient 3D point cloud perception. The key idea is to utilize voxel pyramid to make full use of voxel information and then adopt the attention mechanism for better point-voxel feature fusion. Extensive experiments validate the superiority of PyraPVConv. We hope that PyraPVConv can act as an important part of various deep networks for 3D point cloud perception. In the future, we plan to utilize the technique of neural architecture search for designing more efficient network architectures.

Data Availability

All datasets used in this paper are publicly available.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Yuhong Chen and Weilong Peng contributed equally to this work.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (62102105, 62072126, and 61772164), Guangdong Basic and Applied Basic Research Foundation 2022A1515011501, 2022A1515010138 and 2020A1515110997 and Science and Technology Program of Guangzhou (202002030263 and 202102010419).

References

[1] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: an accurate open-source library for visual, visual-inertial, and multimap SLAM," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[2] M. Frosi and M. Matteucci, "Art-slam: accurate real-time 6dof lidar slam," *IEEE Robotics and Automation Letters*, vol. 7, 2022.

[3] K. Xu, L. Zheng, Z. Yan et al., "Autonomous reconstruction of unknown indoor scenes guided by time-varying tensor fields," *ACM Transactions on Graphics*, vol. 36, no. 6, pp. 1–15, 2017.

[4] Y. Cui, R. Chen, W. Chu et al., "Deep learning for image and point cloud fusion in autonomous driving: a review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, 2021.

[5] Y. Zeng, Y. Hu, S. Liu et al., "Rt3d: real-time 3-d vehicle detection in lidar point cloud for autonomous driving," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3434–3440, 2018.

[6] Y. Li and J. Ibanez-Guzman, "Lidar for autonomous driving: the principles, challenges, and trends for automotive lidar and perception systems," *IEEE Signal Processing Magazine*, vol. 37, no. 4, pp. 50–61, 2020.

[7] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan, "Rotational projection statistics for 3d local surface description and object recognition," *International Journal of Computer Vision*, vol. 105, no. 1, pp. 63–86, 2013.

[8] K. Tang, P. Song, and X. Chen, "Signature of geometric centroids for 3d local shape description and partial shape matching," in *Proceedings of the Asian Conference on Computer Vision*, pp. 311–326, Springer, Taipei, Taiwan, November 2016.

[9] K. Tang, P. Song, and X. Chen, "3d object recognition in cluttered scenes with robust shape description and correspondence selection," *IEEE Access*, vol. 5, no. 99, pp. 1833–1845, 2017.

[10] Y. Zhou and O. Tuzel, "Voxelnet: end-to-end learning for point cloud based 3d object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4490–4499, Salt Lake City, UT, USA, June 2018.

[11] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 652–660, Honolulu, HI, USA, July 2017.

[12] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3d point clouds: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 4338–4364, 2020.

[13] M. Simony, S. Milzy, K. Amendey, and H.-M. Gross, "Complex-yolo: an euler-region-proposal for real-time 3d object detection on point clouds," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pp. 197–209, Munich, Germany, September 2018.

[14] B. Yang, W. Luo, and R. Urtasun, "Pixor: real-time 3d object detection from point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7652–7660, Salt Lake City, UT, USA, June 2018.

[15] C. He, H. Zeng, J. Huang, X.-S. Hua, and L. Zhang, "Structure aware single-stage 3d object detection from point cloud," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11873–11882, Seattle, WA, USA, June 2020.

[16] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: fast encoders for object detection from point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12697–12705, Long Beach, CA, USA, June 2019.

[17] M. Ye, S. Xu, and T. Cao, "Hvnet: hybrid voxel network for lidar based 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1631–1640, Seattle, WA, USA, June 2020.

[18] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: deep hierarchical feature learning on point sets in a metric space," in *Proceedings of the Advances in Neural Information Processing Systems*, vol. 30, pp. 1–14, Long Beach, CA, USA, December 2017.

[19] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "Kpconv: flexible and deformable convolution for point clouds," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6411–6420, Seoul, Republic of Korea, November 2019.

[20] Z. Liu, H. Tang, Y. Lin, and S. Han, "Point-voxel cnn for efficient 3d deep learning," *Advances in Neural Information Processing Systems*, vol. 32, pp. 965–975, 2019.

[21] D. Maturana and S. Scherer, "Voxnet: a 3d convolutional neural network for real-time object recognition," in *Proceedings of the 2015 IEEE/RSJ International Conference on*

Intelligent Robots and Systems (IROS), pp. 922–928, Hamburg, Germany, October 2015.

[22] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas, “Volumetric and multi-view cnns for object classification on 3d data,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5648–5656, Las Vegas, NV, USA, June 2016.

[23] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong, “O-CNN: octree-based convolutional neural networks for 3D shape analysis,” *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 1–11, 2017.

[24] R. Klokov and V. Lempitsky, “Escape from cells: deep kd-networks for the recognition of 3d point cloud models,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 863–872, Venice, Italy, October 2017.

[25] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, “Dynamic graph cnn for learning on point clouds,” *ACM Transactions on Graphics*, vol. 38, no. 5, pp. 1–12, 2019.

[26] H. Tang, Z. Liu, S. Zhao et al., “Searching efficient 3d architectures with sparse point-voxel convolution,” in *Proceedings of the European Conference on Computer Vision*, pp. 685–702, Springer, Glasgow, Scotland, August 2020.

[27] Q. Hu, B. Yang, L. Xie et al., “Randla-net: efficient semantic segmentation of large-scale point clouds,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11108–11117, Seattle, WA, USA, June 2020.

[28] X. Zhu, H. Zhou, T. Wang et al., “Cylindrical and asymmetrical 3d convolution networks for lidar segmentation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9939–9948, Montreal, Canada, October 2021.

[29] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, “Frustum pointnets for 3d object detection from rgb-d data,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 918–927, Salt Lake City, UT, USA, June 2018.

[30] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proceedings of the Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, Lake Tahoe, Nevada, December 2012.

[32] J. Wang, S. Lan, M. Gao, and L. S. Davis, “Infofocus: 3d object detection for autonomous driving with dynamic information modeling,” in *Proceedings of the European Conference on Computer Vision*, pp. 405–420, Springer, Glasgow, Scotland, August 2020.

[33] P. Hu, J. Ziglar, D. Held, and D. Ramanan, “What you see is what you get: exploiting visibility for 3d object detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11001–11009, Seattle, WA, USA, June 2020.

[34] M. Liang, B. Yang, S. Wang, and R. Urtasun, “Deep continuous fusion for multi-sensor 3d object detection,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 641–656, Munich, Germany, September 2018.

[35] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, NV, USA, June 2016.

[36] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2117–2125, Honolulu, HI, USA, July 2017.

[37] H. Kuang, B. Wang, J. An, M. Zhang, and Z. Zhang, “Voxel-fpn: multi-scale voxel feature aggregation for 3d object detection from lidar point clouds,” *Sensors*, vol. 20, no. 3, p. 704, 2020.

[38] C. Bundesen, “A theory of visual attention,” *Psychological Review*, vol. 97, no. 4, pp. 523–547, 1990.

[39] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2921–2929, Las Vegas, NV, USA, June 2016.

[40] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: visual explanations from deep networks via gradient-based localization,” in *Proceedings of the International Journal of Computer Vision*, pp. 336–359, Venice, Italy, October 2017.

[41] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, “Squeeze-and-excitation networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 8, pp. 2011–2023, 2019.

[42] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, “Cbam: convolutional block attention module,” in *Proceedings of the European Conference on Computer Vision*, pp. 3–19, Munich, Germany, September 2018.

[43] P. Anderson, X. He, C. Buehler et al., “Bottom-up and top-down attention for image captioning and visual question answering,” in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition CVPR*, pp. 6077–6086, Lake City, Utah, June 2018.

[44] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” pp. 1–15, 2014, https://arxiv.org/abs/1409.0473.

[45] A. Vaswani, N. Shazeer, N. Parmar et al., “Attention is all you need,” in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 5998–6008, Long Beach California USA, December 2017.

[46] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, “Pointcnn: convolution on x-transformed points,” in *Proceedings of the Advances in Neural Information Processing Systems*, vol. 31, pp. 820–830, Montréal, Canada, December 2018.

[47] I. Armeni, S. Sax, A. R. Zamir, and S. Savarese, “Joint 2d-3d-semantic data for indoor scene understanding,” pp. 1–9, 2017, https://arxiv.org/abs/1702.01105.

[48] I. Armeni, O. Sener, A. R. Zamir et al., “3d semantic parsing of large-scale indoor spaces,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1534–1543, Las Vegas, NV, USA, June 2016.

[49] A. X. Chang, T. Funkhouser, L. Guibas et al., “Shapenet: an information-rich 3d model repository,” pp. 1–11, 2015, https://arxiv.org/abs/1512.03012.

[50] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, “3d u-net: learning dense volumetric segmentation from sparse annotation,” in *Proceedings of the International Conference on Medical Image Computing and Computer-assisted Intervention*, pp. 424–432, Springer, Athens, Greece, October 2016.

[51] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: the kitti dataset,” *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.