Pyramid R-CNN: Towards Better Performance and Adaptability for 3D Object Detection

Jiageng Mao [1] Minzhe Niu [2] Haoyue Bai [3] Xiaodan Liang [4†] Hang Xu [2] Chunjing Xu [2]

Abstract

We present a flexible and high-performance framework, named Pyramid R-CNN, for two-stage 3D object detection from point clouds. Current approaches generally rely on the points or voxels of interest for RoI feature extraction on the second stage, but cannot effectively handle the sparsity and non-uniform distribution of those points, and this may result in failures in detecting objects that are far away. To resolve the problems, we propose a novel second-stage module, named pyramid RoI head, to adaptively learn the features from the sparse points of interest. The pyramid RoI head consists of three key components. Firstly, we propose the RoI-grid Pyramid, which mitigates the sparsity problem by extensively collecting points of interest for each RoI in a pyramid manner. Secondly, we propose RoI-grid Attention, a new operation that can encode richer information from sparse points by incorporating conventional attention-based and graph-based point operators into a unified formulation. Thirdly, we propose the Density-Aware Radius Prediction (DARP) module, which can adapt to different point density levels by dynamically adjusting the focusing range of RoIs. Combining the three components, our pyramid RoI head is robust to the sparse and imbalanced circumstances, and can be applied upon various 3D backbones to consistently boost the detection performance. Extensive experiments show that Pyramid R-CNN outperforms the state-of-the-art 3D detection models by a large margin on both the KITTI dataset and the Waymo Open dataset.

1. Introduction

3D object detection is a key component of perception systems for robotics and autonomous driving, aiming at detecting vehicles, pedestrians, and other objects with 3D point clouds as input. In this paper, we propose a general two-stage 3D detection framework, named Pyramid R-CNN, which can be applied on multiple 3D backbones to enhance the detection adaptability and performance.

Among the existing 3D detection frameworks, two-stage detection models [39, 30, 27, 5, 28] surpass most single-stage 3D detectors [45, 37, 14, 38, 29] with remarkable margins owing to the RoI refinement stage. Different from the 2D counterparts [9, 8, 26, 11, 2] which apply RoIPool [8] or RoIAlign [11] to crop dense feature maps on the second stage, the 3D detection models generally perform various RoI feature extraction operations on the *Points of Interest*. For example, Point R-CNN [29] utilizes a point-based backbone to generate 3D proposals, treats the points near the proposals as Points of Interest and applies Region Pooling on those sparse points for box refinement; Part-A^2 Net [30] utilizes a voxel-based backbone for proposal generation, uses the upsampled voxel points as Points of Interest, and applies sparse convolutions on those voxel points for each RoI; PV-RCNN [27] encodes the whole scene into a set of keypoints, and utilizes keypoints as Points of Interest for RoI-grid Pooling. Those Points of Interest originate from raw point clouds and contain rich fine-grained information, which is required for the RoI refinement stage.

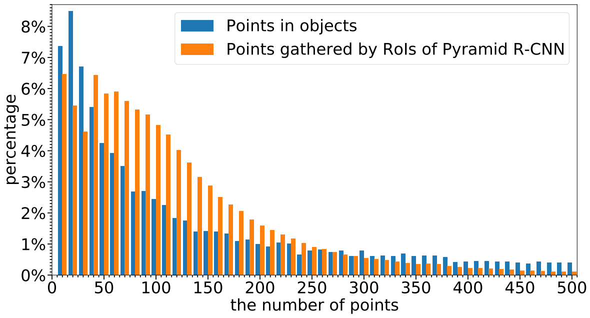


Figure 1. Statistical results on the KITTI dataset. Blue bars denote the distribution of the number of object points. Orange bars denote the distribution of the number of points gathered by RoIs in Pyramid R-CNN. Our approach can mitigate the sparsity and imbalanced distribution problems of point clouds.

However, the Points of Interest inevitably suffer from the sparsity and non-uniform distribution characteristics of input point clouds. As is demonstrated by the statistical results on the KITTI dataset [7] in Figure 1: 1) Point clouds can be quite sparse in certain objects. More than 7% of total objects have less than 10 points, and their visualized shapes are mostly incomplete. Thus it is hard to identify their categories without enough context information. 2) The distribution of object points is extremely imbalanced. The number of object points ranges from less than 10 to more than 500 on KITTI, and current RoI operations cannot handle the im-

[1] The Chinese University of Hong Kong [2] Huawei Noah's Ark Lab [3] HKUST [4] Sun Yat-Sen University
† Corresponding author: xdliang328@gmail.com

balanced conditions effectively. 3) The number of Points of Interest only accounts for a small proportion of input points or voxels, *e.g*. $2k$ keypoints in [27] relative to the $15k$ total input points, which exacerbates the above problems.

To overcome the above limitations, we propose Pyramid R-CNN, a general two-stage 3D detection framework that can effectively detect objects and adapt along with environmental changes. Our main contribution lies in the design of a novel RoI feature extraction head, named pyramid RoI head, which can be applied on multiple 3D backbones and Points of Interest. pyramid RoI head consists of three key components. Firstly, we propose RoI-grid Pyramid. Given the observation that Points of Interest inside RoIs are too sparse for object recognition, our RoI-grid Pyramid captures more Points of Interest outside RoIs while still maintaining fine-grained geometric details, by extending the standard one-level RoI-grid to a pyramid structure. Secondly, we propose RoI-grid Attention, an effective operation to extract RoI-grid features from Points of Interest. RoI-grid Attention leverages the advantages of the graph-based and attention-based point operators by combining those formulas into a unified formulation, and it can adapt to different sparsity situations by dynamically attending to the crucial Points of Interest near the RoIs. Thirdly, we propose the Density-Aware Radius Prediction (DARP) module, which can predict the feature extraction radius of each RoI, conditioning on the neighboring distribution of Points of Interest. Thus we can address the imbalanced distribution problem by adaptively adjusting the focusing range for each RoI. Combining all the above components, the pyramid RoI head shows adaptability to different point cloud sparsity levels and can accurately detect the 3D objects with only a few points. Our Pyramid R-CNN is compatible with the point-based [29], voxel-based [30] and point-voxel-based [27] frameworks, and significantly boosts the detection accuracy.

We summarize our key contributions as follows:

1) We propose Pyramid R-CNN, a general two-stage framework that can be applied on multiple backbones for accurate and robust 3D object detection.

2) We propose the pyramid RoI head, which combines the RoI-grid Pyramid, RoI-grid Attention, and the Density-Aware Radius Prediction (DARP) module together to mitigate the sparsity and non-uniform distribution problems.

3) Pyramid R-CNN consistently outperforms the baselines, achieves 82.08% moderate car mAP on the KITTI dataset, and ranks 1^{st} among the LiDAR-only methods on the Waymo *test* leaderboard for vehicle detection.

2. Related Work

Single-stage 3D Object Detection. Single-stage methods can be divided into 3 streams, *i.e*., point-based, voxel-based and pillar-based. The point-based single-stage detectors generally take the raw points as input, and apply set abstraction [25, 20] to obtain the point features for box prediction. 3DSSD [38] introduces Feature-FPS as a new sampling strategy for raw point clouds. Point-GNN [31] proposes a graph operator to aggregate the points information for object detection. The voxel-based single-stage detectors typically rasterize point clouds into voxel-grids and then apply 2D and 3D CNN to generate 3D proposals. VoxelNet [45] divides points into voxels and leverages a 3D CNN to aggregate voxel features for proposal generation. SECOND [37] improves the voxel feature learning process by introducing 3D sparse convolutions. CenterPoints [41] proposes a center-based assignment that can be applied on feature maps for accurate location prediction. Pillar-based approaches generally transform point clouds into Bird-Eye-View (BEV) pillars and apply 2D CNNs for 3D object detection. PointPillars [14] is the first work that introduces the pillar representation. Pillar-based network [35] extends the idea by proposing the cylindrical view projection. Unlike the two-stage approaches, the single-stage methods cannot benefit from the fine-grained point information, which is crucial for accurate box prediction.

Two-stage 3D object detection. Two-stage approaches can be divided into 3 streams, based on the representation of Points of Interest, *i.e*., point-based, voxel-based and point-voxel-based. Point-based approaches treat the sampled point clouds as Points of Interest. PointRCNN [29] generates 3D proposals from raw point clouds and proposes Region Pooling to extract RoI features for the second stage refinement. STD [39] proposes a sparse-to-dense strategy and uses the PointsPool operation for RoI refinement. Voxel-based methods use the voxel points from 3D CNNs as Points of Interest. Part-A^2 Net [30] applies 3D sparse convolutions on the upsampled voxel points for RoI refinement. Voxel R-CNN [5] utilizes Voxel RoI Pooling to extract RoI features from voxels. Point-Voxel-based approaches use the keypoints that encode the whole scene as Points of Interest. PV-RCNN [27] designs RoI-grid Pooling to aggregate keypoint features near RoIs. PV-RCNN++ [28] proposes Vector-Pooling to efficiently collect the keypoint features from different orientations. Compared with the previous methods, our Pyramid R-CNN shows better performance and robustness, and is compatible with all the representations of Points of Interest.

3. Pyramid R-CNN

In this section, we detail the design of Pyramid R-CNN, a general two-stage framework for 3D object detection. We first introduce the overall architecture in 3.1. Then we introduce three key components in the pyramid RoI head: RoI-grid Pyramid in 3.2, RoI-grid Attention in 3.3, and the Density-Aware Radius Prediction (DARP) module in 3.4.

3.1. Overall Architecture

Here, we present a new two-stage framework for accurate and robust 3D object detection, named Pyramid R-CNN, as shown in Figure 2. The framework can be compatible with multiple backbones, *e.g*. the point-based backbone, the voxel-based backbone , or the point-voxel-based back-

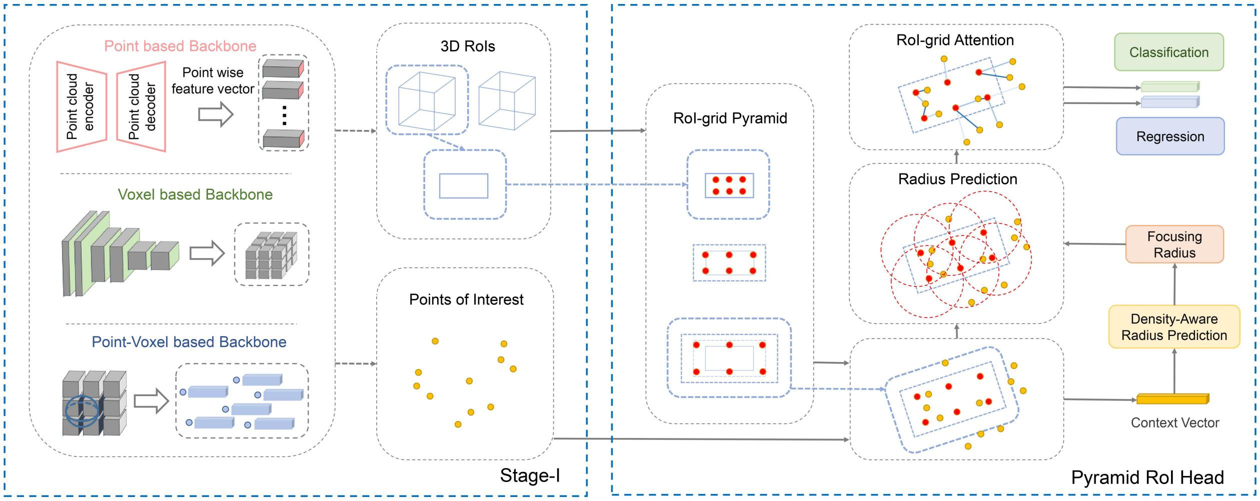


Figure 2. The overall architecture. Our Pyramid R-CNN can be plugged on diverse backbones (*e.g*. point-based, voxel-based and point-voxel-based networks), which generate 3D proposals and Points of Interest (yellow points) on the stage-1. On the stage-2, we propose the pyramid RoI head that can be applied upon the 3D proposals and Points of Interest. In the pyramid RoI head, an RoI-grid Pyramid is first built to capture more context information. Then for each RoI-grid point (red point), a focusing radius r (red dashed circle) is learned by the Density-Aware Radius Prediction module. Finally, RoI-grid Attention is performed on the Points of Interest within r for box refinement.

bone. On the first stage, those backbones output 3D proposals and corresponding Points of Interest: *e.g*. point clouds near RoIs in [29], upsampled voxels in [30], and keypoints in [27]. On the second stage, we propose a novel pyramid RoI head, which consists of three key components: the RoI-grid Pyramid, RoI-grid Attention, and the Density-Aware Radius Prediction (DARP) module. For each RoI, we first build an RoI-grid Pyramid, by gradually enlarging the size of the original RoI in each pyramid level, and the coordinates of RoI-grid points are determined by the enlarged RoI size and the grid size. In each pyramid level, the focusing radius r of the RoI-grid points is predicted from the global context vector through the Density-Aware Radius Prediction module. Then RoI-grid Attention parameterized by r is performed to aggregate the features of Points of Interest into the RoI-grids. Finally, the RoI-grid features are enhanced and fed into two individual heads for classification and regression. We will describe those key components in the following sections.

3.2. RoI-grid Pyramid

In this section, we present the RoI-grid Pyramid, a simple and effective module that captures rich context while still maintains internal structural information. Different from 2D feature pyramid [18] which hierarchically encodes context information upon dense backbone features, our RoI-grid Pyramid is applied on each RoI by gradually placing the grid points *out of* RoIs in a pyramid manner. The idea behind this design is based on the observation that image features inside RoIs generally contain sufficient semantic contexts, while point clouds inside RoIs contain quite limited information since object points are naturally sparse and incomplete. Even though each point has a large receptive field, the sparse compositional 3D shapes inside RoIs are hard to be recognized. In the following parts we will introduce detailed formulations.

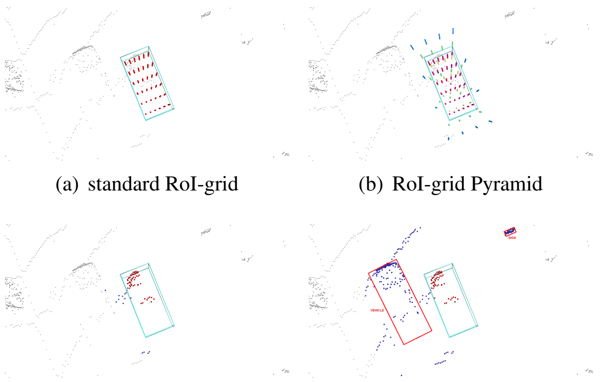
RoI feature extraction generally relies on an RoI-grid for each RoI, and RoI-grid points collect the features of adjacent pixels or neighboring Points of Interest in the 2D or 3D cases respectively. Supposing we have an RoI with W, L, H as width, length, and height and (x_c, y_c, z_c) as the bottom left corner, in standard RoI-grid representation, the (i, j, k) RoI-grid point location p_{grid}^{ijk} can be computed as:

$$p_{grid}^{ijk} = (\frac{W}{N_w}, \frac{L}{N_l}, \frac{H}{N_h}) \cdot (0.5 + (i, j, k)) + (x_c, y_c, z_c), \quad (1)$$

where (N_w, N_l, N_h) are the grid sizes in three dimensions and all grid points are generated inside RoIs.

Utilizing features only inside the RoIs works well in the 2D detection models, mainly owing to two facts: the input feature map is dense and the collected pixels have large receptive fields. However, the cases are different in 3D models. As is shown in Figure 3, the Points of Interest are naturally sparse and non-uniformly distributed inside the RoIs, and the object shape is extremely incomplete. Thus it is hard to accurately infer the sizes and categories of objects by solely collecting the features of few individual points and not referring to enough neighboring points information.

To resolve the above problems, we propose the RoI-grid Pyramid which balances the fine-grained and large context information. The detailed structure is in Figure 3. The key idea is to construct a pyramid grid structure that contains the RoI-grid points both inside and outside RoIs, so that the grid points inside RoIs can capture fine-grained shape structures for accurate box refinement, while the grid points outside RoIs can obtain large context information to identify incomplete objects. The grid points p_{grid}^{ijk} for a pyramid



(a) standard RoI-grid (b) RoI-grid Pyramid

(c) object/context points in (a) (d) object/context points in (b)

Figure 3. Illustration of the RoI-grid Pyramid. Red points in (a) are the RoI-grid points, and different colors represent different pyramid levels in (b). In (c) and (d) red points are object points and blue points are context points captured by the RoI. Compared to the standard RoI-grid, our RoI-grid Pyramid can capture more context points while maintain fine-grained internal structures, and by looking at neighboring vehicle and traffic sign (blue context points) outside the RoI, the cluster of red object points is easier to be recognized as a car.

level can be computed as:

$$p_{grid}^{ijk} = (\frac{\rho_w W}{N'_w}, \frac{\rho_l L}{N'_l}, \frac{\rho_h H}{N'_h}) \cdot (0.5 + (i,j,k)) + (x_c, y_c, z_c), \tag{2}$$

where ρ is the enlarging ratio of the original RoI size. ρ starts from 1 at the bottom level for maintaining fine-grained details, and becomes larger when the level goes higher to capture more context information. The grid size N' is initialized with the same value as the original N at the bottom level and gets smaller at higher levels to save computational resources. For each pyramid level, features of grid points f_{grid} are then aggregated by RoI-grid Attention from the features of Points of Interest. Finally, features of all pyramid levels are combined for boxes refinement.

3.3. RoI-grid Attention

In this section, we introduce RoI-grid Attention, a novel RoI feature extraction operation that combines the state-of-the-art graph-based and attention-based point operators [36, 34, 43] into a unified framework, and RoI-grid Attention can serve as a better substitute for conventional pooling-based operations [27, 5, 28] in 3D detection models. We first discuss the formulas of pooling-based, graph-based and attention-based point operators, and then we derive the formulation of RoI-grid Attention.

Preliminary. Let p_{grid} be the coordinate of an RoI-grid point, and p_i, f_i be the coordinate and the corresponding feature vector of the i_{th} Points of Interest near p_{grid}. RoI feature extraction operation aims to obtain the respective feature vector f_{grid} of the RoI-grid point p_{grid}, using the information of neighboring p_i and f_i.

Pooling-based Operators. The pooling-based operators are extensively applied for RoI feature extraction in most two-stage 3D detection models [27, 5, 28]. The neighboring feature f_i and the relative location $p_i - p_{grid}$ first go through a MLP layer to obtain the transformed feature vector: $V^i = MLP([f_i, p_i - p_{grid}])$, where $[\cdot]$ is the concatenation function, and then a maxpooling operation is applied upon all the transformed features V to obtain the RoI-grid feature f_{grid}^{pool}:

$$f_{grid}^{pool} = \underset{i \in \Omega(r)}{maxpool}(V^i), \tag{3}$$

where $\Omega(r)$ means Points of Interest within the fixed radius r of the RoI-grid point p_{grid}. The pooling-based operators only focus on the maximum channel response and this results in a loss of much semantic and geometric information.

Graph-based Operators. Graph-based operators can model the grid points and Points of Interest as a graph. The graph node i represents the transformed feature of f_i: $V^i = MLP(f_i)$, and the edge Q_{pos}^i can be formulated as a linear projection of the location differences between two nodes: $Q_{pos}^i = Linear(p_i - p_{grid})$. For the graph node of a grid point p_{grid}, the feature f_{grid}^{graph} is collected from adjacent nodes by a weighted combination operation. Following the same notations as Eq.3, the general formula can be represented as

$$f_{grid}^{graph} = \sum_{i \in \Omega(r)} W(Q_{pos}^i) \odot V^i, \tag{4}$$

where the function $W(\cdot)$ projects the graph edge embedding into the scalar or vector weight space, and $\odot$ denotes either the Hadamard product, dot product or scalar-vector product between learned weights and graph nodes.

Attention-based Operators. Attention-based operators can also be applied upon the grid points and Points of Interest. Q_{pos}^i in Eq.4 can be viewed as the query embedding from the grid point p_{grid} to the point p_i. V^i is the value embedding obtained from the feature f_i as Eq.4. The key embedding K^i can be formulated as $K^i = Linear(f_i)$. Thus standard attention can be formulated as

$$f_{grid}^{atten} = \sum_{i \in \Omega(r)} W(Q_{pos}^i K^i) \odot V^i. \tag{5}$$

Additional normalization function, *i.e.* softmax, is applied in $W(\cdot)$. Recently proposed Point Transformer [43] extending the idea of standard attention and the formula can be represented as

$$f_{grid}^{tr} = \sum_{i \in \Omega(r)} W(K^i + Q_{pos}^i) \odot (V^i + Q_{pos}^i). \tag{6}$$

RoI-grid Attention. In our approach, we analyze the structural similarity of Eq.4, Eq.5 and Eq.6. We find that those formulas have common basic elements and operators. Thus it's natural to merge those formulas into a unified

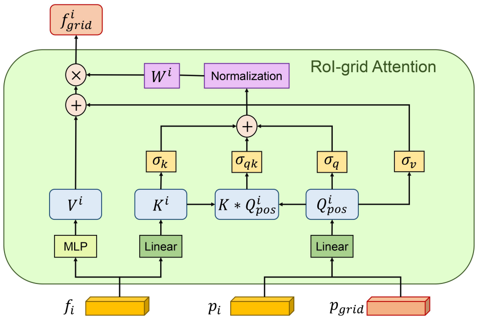


Figure 4. Illustration of RoI-grid Attention. RoI-grid Attention introduces learnable gated functions $\sigma_\star$ to dynamically select the attention components, and it provides a unified formulation that includes the conventional graph and attention operators.

framework with gated functions. We name this new formula RoI-grid Attention:

$$f_{grid} = \sum_{i \in \Omega(r)} W(\sigma_k K^i + \sigma_q Q^i_{pos} + \sigma_{qk} Q^i_{pos} K^i) \odot (V^i + \sigma_v Q^i_{pos}), \tag{7}$$

where σ_* is a learnable gated function which can be implemented by a linear projection of the respective embedding with a sigmoid activation output. RoI-grid Attention is a generalized formulation combining graph-based and attention-based operations. We can derive the graph operator Eq.4 from Eq.7 when σ_q, σ_k, σ_{qk}, σ_v are 1, 0, 0, 0 respectively. Similarly, we can derive the standard attention Eq.5 when σ_q, σ_k, σ_{qk}, σ_v are 0, 0, 1, 0, or Point Transformer Eq.6 when $\sigma_q, \sigma_k, \sigma_{qk}, \sigma_v$ are 1, 1, 0, 1.

RoI-grid Attention is a flexible and effective operation for RoI feature extraction. With the learnable gated functions, RoI-grid Attention is able to learn which point is significant to the RoI-grid points, from both the geometric information Q_{pos} and the semantic information K, as well as their combinations $Q_{pos}K$ adaptively. With σ_v, RoI-grid Attention can also learn to balance the ratio of geometric features Q_{pos} and semantic features V used in feature aggregation. Compared with the pooling-based methods, only a few linear projection layers are added in RoI-grid Attention, which maintains the computational efficiency. Replacing pooling-based operators with RoI-grid Attention consistently boosts the detection performance.

3.4. Density-Aware Radius Prediction

In this section, we investigate the learning problem of the radius r, which determines the range $\Omega(r)$ of neighboring Points of Interest that participate in the feature extraction process. The radius r is a hyper-parameter used in all the point operators in 3.3, and has to be determined by researchers in previous approaches. The fixed and predefined r cannot adapt to the density changes of point clouds, and may lead to empty spherical ranges if not set properly. In this paper, we make the prediction of r a fully-differentiable process and further propose the Density-Aware Radius Prediction (DARP) module, aiming at learning an adaptive neighborhood for RoI feature extraction. We first introduce the general formulation of RoI-grid Attention from a probabilistic perspective. Next, we propose a novel method to differentiate the learning of r. Finally, we introduce the design of the DARP module.

RoI-grid Attention is composed of two steps: first selects Points of Interest within the radius r, and next performs weighted combinations on those points. With the same notations in 3.3, we can reformulate the first step as sampling from a conditional distribution $p(i|r)$:

$$p(i|r) = \begin{cases} 0 & ||p_i - p_{grid}||_2 > r \\ 1 & ||p_i - p_{grid}||_2 \leq r \end{cases} \tag{8}$$

Then the second step can be represented as calculating the probabilistic expectation:

$$f_{grid} = \mathbb{E}_{i \sim p(i|r)}[W^i \odot V^i], \tag{9}$$

where W^i denotes $W(\sigma_k K^i + \sigma_q Q^i_{pos} + \sigma_{qk} Q^i_{pos} K^i)$ and V^i denotes $(V^i + \sigma_v Q^i_{pos})$ with a slight abuse of notations.

We propose a new probability distribution $s(i|r)$ as a substitute for $p(i|r)$, and $s(i|r)$ should satisfy two requirements: i) $s(i|r)$ should have similar characteristics as $p(i|r)$, which means that most points sampled from $s(i|r)$ should be inside r; ii) $s(i|r)$ should also leave a few points outside r, mainly for the exploration of the surrounding environment. Thus we formulate the probability $s(i|r)$ as:

$$s(i|r) = 1 - sigmoid(\frac{||p_i - p_{grid}||_2 - r}{\tau}), \tag{10}$$

where $sigmoid(x) = (1 + e^{-x})^{-1}$ and τ is the temperature which controls the decay rate of probability. With a small τ, $s(i|r)$ is close to 1 when p_i is inside r, and is close to 0 if outside, while near the spherical boundary the sampling probability $s(i|r)$ is between 0 and 1. With $s(i|r)$ as a smooth approximation to $p(i|r)$, we want to compute the gradient of r from the approximated RoI-grid Attention:

$$\nabla_r f_{grid} = \nabla_r \mathbb{E}_{i \sim s(i|r)}[W^i \odot V^i]. \tag{11}$$

However, taking the derivative w.r.t. r is still infeasible, since we cannot directly calculate the gradient of a parameterized distribution. The reparameterization trick [12] offers a possible solution to the problem. The key insight is sampling from a basic distribution and then move the original distribution parameters inside the expectation function as coefficients. The gradient of r can be computed as:

$$\nabla_r f_{grid} = \mathbb{E}_{i \sim U(\epsilon)}[\nabla_r [s(i,r) \cdot W^i \odot V^i]], \tag{12}$$

where $s(i,r)$ is the same as Eq.10, and the theoretical distribution $U(\epsilon) = 1$ means that the sampling probability is 1 in the whole 3D space. In practical, considering the fact that

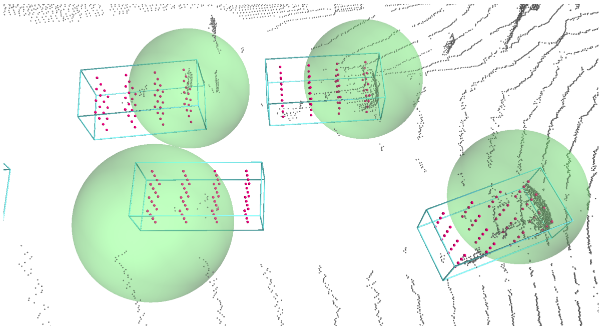


Figure 5. Illustration of dynamic radius predicted by the Density-Aware Radius Prediction module. For each RoI, an adaptive focusing radius is learned based on the sparsity conditions.

$s(i,r)$ is close to 0 when $\epsilon \gg r$, we apply an approximation and restrict the sampling range $U(\epsilon)$ within a sphere with a radius slightly larger than r, *i.e.* $r + 5\tau$ in our experiments. This approximation reduces the computational overhead to the same level as vanilla RoI-grid Attention. Since $s(i,r)$ is a differentiable function w.r.t. r, we are able to compute the gradient of r in a differential manner using Eq.12. The new formulation of RoI-grid Attention can be represented as

$$f_{grid} = \sum_{i \in U(\epsilon)} W(\sigma_k K^i + \sigma_q Q^i_{pos} + \sigma_{qk} Q^i_{pos} K^i) \odot (V^i + \sigma_v Q^i_{pos}) \cdot s(i,r). \tag{13}$$

Compared with vanilla RoI-grid Attention in Eq.7, a slightly larger sampling range $r + 5\tau$ is used and an coefficient $s(i,r)$ is added into the original formula, which costs little additional resources. Although several approximations are applied, we found that they didn't hamper the training but boost the performance in our experiments.

We further propose the DARP module based on Eq.13. For each pyramid level, a context embedding is obtained by summarizing the information of Points of Interest near this RoI, and then the embedding is utilized to predict the radius r for all grid points in this level. r is further transformed into an coefficient by $s(i,r)$ and participates in the computation of RoI-grid Attention. Since the context embedding captures point cloud information, *i.e.* density, shape, *etc.*, the predicted r is able to adapt to the environmental changes, and is more robust than the human-defined counterpart.

4. Experiments

In this section, we evaluate our Pyramid R-CNN on the commonly used Waymo Open dataset [32] and the KITTI [7] dataset. We first introduce the experimental settings in 4.1 and then compare our approach with previous state-of-the-art methods on the Waymo Open dataset in 4.2 and the KITTI dataset in 4.3. Finally, we conduct ablation studies to evaluate the efficacy of each component in 4.4.

4.1. Experimental Setup

Waymo Open Dataset. The Waymo Open Dataset contains 1000 sequences in total, including 798 sequences (around $158k$ point cloud samples) in the training set and 202 sequences (around $40k$ point cloud samples) in the validation set. The official evaluation metrics are standard 3D mean Average Precision (mAP) and mAP weighted by heading accuracy (mAPH). Both of the two metrics are based on an IoU threshold of 0.7 for vehicles and 0.5 for other categories. The testing samples are split in two ways. The first way is based on the distances of objects to the sensor: $0-30m$, $30-50m$ and $> 50m$. The second way is according to the difficulty levels: LEVEL_1 for boxes with more than five LiDAR points and LEVEL_2 for boxes with at least one LiDAR point.

KITTI Dataset. The KITTI dataset contains 7481 training samples and 7518 test samples, and the training samples are further divided into the *train* split (3712 samples) and the *val* split (3769 samples). The official evaluation metric is mean Average Precision (mAP) with a rotated IoU threshold 0.7 for cars. On the *test* set mAP is calculated with 40 recall positions by the official server. The results on the *val* set are calculated with 11 recall positions for a fair comparison with other approaches.

We provide 3 architectures of Pyramid R-CNN, compatible with the point-based, the voxel-based and the point-voxel-based backbone, respectively. We would like readers to refer to [33] for the detailed design of those backbones.

Pyramid-P. ***Pyramid*** *R-CNN for* ***Points*** is built upon the point-based method PointRCNN [29]. In particular, we replace the Canonical 3D Box Refinement module of PointRCNN, with our proposed pyramid RoI head in Pyramid R-CNN, and we still use the sampled points in [29] as Points of Interest. The point cloud backbone and other configurations are kept the same for a fair comparison.

Pyramid-V. ***Pyramid*** *R-CNN for* ***Voxels*** is built upon the voxel-based method Part-A^2 Net [30]. Specifically, we replace the 3D sparse convolutional head of Part-A^2 Net, with our proposed pyramid RoI head in Pyramid R-CNN, and we still use the upsampled voxels as Points of Interest. The voxel-based backbone and other configurations are kept the same for a fair comparison.

Pyramid-PV. ***Pyramid*** *R-CNN for* ***Point-Voxels*** is designed upon the point-voxel-based method PV-RCNN [27]. In particular, we replace the RoI-grid Pooling module of PV-RCNN, with our proposed pyramid RoI head in Pyramid R-CNN, and we still use the keypoints as Points of Interest. The keypoints encoding process, the 3D sparse convolutional networks and other configurations are kept the same for a fair comparison.

Implementation Details. Here we only introduce the architecture of Pyramid-PV on the Waymo Open dataset. The implementations of other models are similar and can be found in the supplementary materials. In RoI-grid Attention, the number of attention heads is set to 4 and each head contains 16 feature channels. In the DARP module, the context embedding is extracted from the neighboring Points of Interest within two spheres with the radius $2.4m$ and $4.8m$. The temperature τ starts from 0.02 and exponentially de-

Methods	LEVEL_1 3D mAP/mAPH	LEVEL_2 3D mAP/mAPH	LEVEL_1 3D mAP/mAPH by Distance 0-30m	 30-50m	 50m-Inf
PointPillars [14]	63.3/62.7	55.2/54.7	84.9/84.4	59.2/58.6	35.8/35.2
MVF [44]	62.93/-	-	86.30/-	60.02/-	36.02/-
Pillar-OD [35]	69.8/-	-	88.5/-	66.5/-	42.9/-
AFDet [6]	63.69/-	-	87.38/-	62.19/-	29.27/-
LaserNet [21]	52.1/50.1	-	70.9/68.7	52.9/51.4	29.6/28.6
CVCNet [3]	65.2/-	-	86.80/-	62.19/-	29.27/-
StarNet [22]	64.7/56.3	45.5/39.6	83.3/82.4	58.8/53.2	34.3/25.7
RCD [1]	69.0/68.5	-	87.2/86.8	66.5/66.1	44.5/44.0
Voxel R-CNN [5]	75.59/-	66.59/-	92.49/-	74.09/-	53.15/-
PointRCNN* [29]	45.05/44.25	37.41/36.74	72.24/71.31	31.21/30.41	23.77/23.15
Pyramid-P (ours)	**47.02/46.58**	**39.10/38.76**	**74.24/73.78**	**32.49/31.96**	**25.68/25.24**
Part-A^2 Net* [30]	71.69/71.16	64.21/63.70	91.83/91.37	69.99/69.37	46.26/45.41
Pyramid-V (ours)	**75.83/75.29**	**66.77/66.28**	**92.63/92.20**	**74.46/73.84**	**53.40/52.44**
PV-RCNN [27]	70.3/69.7	65.4/64.8	91.9/91.3	69.2/68.5	42.2/41.3
Pyramid-PV (ours)	**76.30/75.68**	**67.23/66.68**	**92.67/92.20**	**74.91/74.21**	**54.54/53.45**

Table 1. Performance comparison on the Waymo Open Dataset with 202 validation sequences for the vehicle detection. $\star$: re-implemented by ourselves with the official code.

Methods	LEVEL_1 3D mAP/mAPH	LEVEL_2 3D mAP/mAPH	LEVEL_1 3D mAP/mAPH by Distance 0-30m	 30-50m	 50m-Inf
CenterPoint* [41]	81.05/80.59	73.42/72.99	92.52/92.13	79.94/79.43	61.06/60,42
PV-RCNN* [27]	81.06/80.57	73.69/73.23	93.40/92.98	80.12/79.57	61.22/60.47
Pyramid-PV‡ (ours)	**81.77/81.32**	**74.87/74.43**	**93.19/92.80**	**80.53/80.04**	**64.55/63.84**

Table 2. Performance comparison on the Waymo Open Dataset *test* leaderboard for the vehicle detection. $\star$: test submissions are the modified version of original architectures. ‡: We append another frame following [27] and use a larger voxel backbone.

cays to 0.0001 in the end. The RoI-grid Pyramid consists of 5 levels, with the number of grid points as $6^3, 4^3, 4^3, 4^3, 1$ respectively, and for each pyramid level, a focusing radius r is predicted and shared across all the grid points in this level. The enlarging ratio ρ_w and ρ_l are set to $1, 1, 1.5, 2, 4$ for the respective level of the RoI-grid Pyramid, and ρ_h is set to 1 in all pyramid levels. The maximum number of points that participate in RoI-grid Attention for each grid point is set to $8, 16, 16, 16, 32$ for the corresponding pyramid level.

Training and Inference Details. Our Pyramid R-CNN is trained from scratch with the ADAM optimizer. On the KITTI dataset, Pyramid-P, Pyramid-V and Pyramid-PV are trained with the same batch size 16, the learning rate $0.01, 0.01, 0.005$ respectively for 80 epochs on 8 V100 GPUs. On the Waymo Open dataset, we uniformly sample 20% frames for training and use the full validation set for evaluation following [27]. Pyramid-P, Pyramid-V and Pyramid-PV are trained with the same batch size 32, the learning rate 0.01 for 40 epochs. The cosine annealing learning rate strategy is adopted for the learning rate decay. Other configurations are kept the same as the corresponding baselines [29, 30, 27] for a fair comparison.

4.2. Comparisons on the Waymo Open Dataset

We evaluate the performance of Pyramid R-CNN on the Waymo Open dataset. The validation results in Table 1 show that our Pyramid-P, Pyramid-V and Pyramid-PV significantly outperform the baseline methods with 2.0%, 4.1% and 6.0% mAP gain respectively, and achieves superior mAP on all difficulty levels and all distance ranges, which demonstrates the effectiveness and generalizability of our approach. It is worth noting that Pyramid-V surpasses PV-RCNN by 12.3% mAP in detecting objects that are $> 50m$, which indicates the adaptability of our approach to the extremely sparse conditions. Our Pyramid-PV outperforms all the previous approaches with a remarkable margin, and achieves the new state-of-the-art performance 76.30% mAP and 67.23% mAP for the LEVEL_1 and LEVEL_2 difficulty. In table 2, our Pyramid-PV‡ achieves 81.77% LEVEL_1 mAP, ranks 1^{st} on the Waymo vehicle detection leaderboard as of March 10th, 2021, and surpasses all the LiDAR-only approaches.

4.3. Comparisons on the KITTI Dataset

We evaluate our Pyramid R-CNN on the KITTI dataset. The *test* results in Table 3 show that our Pyramid-P, Pyramid-V and Pyramid-PV consistently outperform the baseline methods with 4.66%, 2.79% and 0.65% mAP gain respectively on the moderate car class, and Pyramid-PV achieves 82.08% mAP, becoming the new state-of-the-art. The validation results in Table 4 show that Pyramid-P, Pyramid-V and Pyramid-PV improve the baselines by 4.47%, 3.67% and 0.69% mAP on the moderate car class, and 1.06%, 0.07% and 0.14% mAP on the hard car class respectively. We note that the performance gains are mainly from the hard cases, which indicates the adaptability of our

Methods	Modality	AP_{3D} (%)		
		Easy	Mod.	Hard
MV3D [4]	R+L	74.97	63.63	54.00
AVOD-FPN [13]	R+L	83.07	71.76	65.73
F-PointNet [24]	R+L	82.19	69.79	60.59
MMF [16]	R+L	88.40	77.43	70.22
3D-CVF [42]	R+L	89.20	80.05	73.11
CLOCs [23]	R+L	88.94	80.67	77.15
ContFuse [17]	R+L	83.68	68.78	61.67
VoxelNet [45]	L	77.47	65.11	57.73
PointPillars [14]	L	82.58	74.31	68.99
SECOND [37]	L	84.65	75.96	68.71
STD [39]	L	87.95	79.71	75.09
Patches [15]	L	88.67	77.20	71.82
3DSSD [38]	L	88.36	79.57	74.55
SA-SSD [10]	L	88.75	79.79	74.16
TANet [19]	L	85.94	75.76	68.32
Voxel R-CNN [5]	L	90.90	81.62	77.06
HVNet [40]	L	87.21	77.58	71.79
PointGNN [31]	L	88.33	79.47	72.29
PointRCNN [29]	L	86.96	75.64	70.70
Pyramid-P (ours)	L	**87.03**	**80.30**	**76.48**
Part-A^2 Net [30]	L	87.81	78.49	73.51
Pyramid-V (ours)	L	**87.06**	**81.28**	**76.85**
PV-RCNN [27]	L	90.25	81.43	76.82
Pyramid-PV (ours)	L	**88.39**	**82.08**	**77.49**

Table 3. Performance comparison on the KITTI *test* set with AP calculated by 40 recall positions for the car category. R+L denotes the methods that combines RGB data and point clouds. L denotes LiDAR-only approaches.

Methods	AP_{3D} (%)		
	Easy	Mod.	Hard
PointRCNN [29]	88.88	78.63	77.38
Pyramid-P (ours)	**88.47**	**83.10**	**78.44**
Part-A^2 Net [30]	89.47	79.47	78.54
Pyramid-V (ours)	**88.44**	**83.14**	**78.61**
PV-RCNN [27]	89.35	83.69	78.70
Pyramid-PV (ours)	**89.37**	**84.38**	**78.84**

Table 4. Performance comparison on the KITTI *val* split with AP calculated by 11 recall positions for the car category.

Methods	R.P.	D.A.R.P.	R.A.	LEVEL_1 mAP
PV-RCNN				70.30
PV-RCNN*				74.06
(a)	√			75.26
(b)	√	√		75.63
(c)	√		√	75.77
(d)	√	√	√	**76.30**

Table 5. Effects of different components in Pyramid-PV on the Waymo dataset. R.P.: the RoI-grid Pyramid. D.A.R.P.: the Density-Aware Radius Prediction module. R.A.: RoI-grid Attention. $\star$: re-implemented by ourselves with the official code.

Methods	grid size	ρ_w, ρ_l	LEVEL_1 mAP
PV-RCNN	[6, 6]	[1, 1]	74.06
(a)	[6,4,4]	[1,1,2]	74.55
(b)	[6,4,4,4]	[1,1,2,4]	74.71
(c)	[6,4,4,4,1]	[1,1,1.5,2,4]	**75.26**

Table 6. Effects of different RoI pyramids in Pyramid-PV on the Waymo dataset. Each element in $[\cdot]$ stands for the respective parameter of a pyramid level.

Methods	Inference speed (Hz)
PointRCNN [29]	10.08
Pyramid-P (ours)	8.92
Part-A^2 Net [30]	11.75
Pyramid-V (ours)	9.68
PV-RCNN [27]	9.25
Pyramid-PV (ours)	7.86

Table 7. Comparisons on the inference speeds of different detection models on the KITTI dataset.

approach, and the observations on the KITTI dataset are consistent with those on the Waymo Open dataset.

4.4. Ablation Studies

The effects of different components. As is shown in Table 5, on the Waymo validation set, the RoI-grid Pyramid of the Pyramid-PV model improves over the baseline by 1.20% mAP, mainly because the RoI-grid Pyramid is able to capture large context information which benefits the detection of the hard cases. Based on the RoI-grid Pyramid, replacing RoI-grid Pooling with RoI-grid Attention boosts the performance by 0.51% mAP, which indicates that RoI-grid Attention is a more effective operation than RoI-grid Pooling. Using the adaptive radius r instead of the fixed radius boosts the performance by 0.37% mAP, which demonstrates the efficacy of the DARP module.

The effects of different pyramid configurations. As is shown in Table 6, we found that the RoI-grid Pyramid with $\rho_w, \rho_l > 1$ enhances the performance compared with the standard RoI-grid only with $\rho_w, \rho_l = 1$, mainly because placing some grid points outside RoIs encodes richer contexts. The total number of used grid points is 409, which is comparable to 432 grid points used in [27].

Inference speed analysis. We test the inference speed of different frameworks under a single V100 GPU with batch size 1, and obtain the average running speed of all samples in KITTI *val* split. Table 7 shows that our models maintain computational efficiency compared to the baselines, and the pyramid RoI head only adds little latency per frame.

5. Conclusion

We present a general two-stage framework Pyramid R-CNN which can be applied upon diverse backbones. Our framework can handle the sparse and non-uniform distribution problems of point clouds by introducing the pyramid RoI head. For future work, we plan to optimize Pyramid R-CNN for efficient inference.

References

[1] Alex Bewley, Pei Sun, Thomas Mensink, Dragomir Anguelov, and Cristian Sminchisescu. Range conditioned dilated convolutions for scale invariant 3d object detection. *arXiv preprint arXiv:2005.09927*, 2020. 7

[2] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018. 1

[3] Qi Chen, Lin Sun, Ernest Cheung, and Alan L Yuille. Every view counts: Cross-view consistency in 3d object detection with hybrid-cylindrical-spherical voxelization. *Advances in Neural Information Processing Systems*, 33, 2020. 7

[4] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017. 8

[5] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. *arXiv preprint arXiv:2012.15712*, 2020. 1, 2, 4, 7, 8

[6] Runzhou Ge, Zhuangzhuang Ding, Yihan Hu, Yu Wang, Sijia Chen, Li Huang, and Yuan Li. Afdet: Anchor free one stage 3d object detection. *arXiv preprint arXiv:2006.12671*, 2020. 7

[7] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 1, 6

[8] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 1

[9] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 1

[10] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3d object detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11873–11882, 2020. 8

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 1

[12] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 5

[13] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8. IEEE, 2018. 8

[14] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019. 1, 2, 7, 8

[15] Johannes Lehner, Andreas Mitterecker, Thomas Adler, Markus Hofmarcher, Bernhard Nessler, and Sepp Hochreiter. Patch refinement–localized 3d object detection. *arXiv preprint arXiv:1910.04093*, 2019. 8

[16] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7345–7353, 2019. 8

[17] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 641–656, 2018. 8

[18] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 3

[19] Zhe Liu, Xin Zhao, Tengteng Huang, Ruolan Hu, Yu Zhou, and Xiang Bai. Tanet: Robust 3d object detection from point clouds with triple attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11677–11684, 2020. 8

[20] Jiageng Mao, Xiaogang Wang, and Hongsheng Li. Interpolated convolutional networks for 3d point cloud understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1578–1587, 2019. 2

[21] Gregory P Meyer, Ankit Laddha, Eric Kee, Carlos Vallespi-Gonzalez, and Carl K Wellington. Lasernet: An efficient probabilistic 3d object detector for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12677–12686, 2019. 7

[22] Jiquan Ngiam, Benjamin Caine, Wei Han, Brandon Yang, Yuning Chai, Pei Sun, Yin Zhou, Xi Yi, Ouais Alsharif, Patrick Nguyen, et al. Starnet: Targeted computation for object detection in point clouds. *arXiv preprint arXiv:1908.11069*, 2019. 7

[23] Su Pang, Daniel Morris, and Hayder Radha. Clocs: Camera-lidar object candidates fusion for 3d object detection. *arXiv preprint arXiv:2009.00784*, 2020. 8

[24] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 918–927, 2018. 8

[25] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017. 2

[26] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016. 1

[27] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020. 1, 2, 3, 4, 6, 7, 8

[28] Shaoshuai Shi, Li Jiang, Jiajun Deng, Zhe Wang, Chaoxu Guo, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn++: Point-voxel feature set abstraction with local vector representation for 3d object detection. *arXiv preprint arXiv:2102.00463*, 2021. 1, 2, 4

[29] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019. 1, 2, 3, 6, 7, 8

[30] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 1, 2, 3, 6, 7, 8

[31] Weijing Shi and Raj Rajkumar. Point-gnn: Graph neural network for 3d object detection in a point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1711–1719, 2020. 2, 8

[32] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, 2020. 6

[33] OpenPCDet Development Team. Openpcdet: An open-source toolbox for 3d object detection from point clouds. https://github.com/open-mmlab/OpenPCDet, 2020. 6

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017. 4

[35] Yue Wang, Alireza Fathi, Abhijit Kundu, David Ross, Caroline Pantofaru, Tom Funkhouser, and Justin Solomon. Pillar-based object detection for autonomous driving. *arXiv preprint arXiv:2007.10323*, 2020. 2, 7

[36] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 4

[37] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 1, 2, 8

[38] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11040–11048, 2020. 1, 2, 8

[39] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Std: Sparse-to-dense 3d object detector for point cloud. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1951–1960, 2019. 1, 2, 8

[40] Maosheng Ye, Shuangjie Xu, and Tongyi Cao. Hvnet: Hybrid voxel network for lidar based 3d object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1631–1640, 2020. 8

[41] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking. *arXiv preprint arXiv:2006.11275*, 2020. 2, 7

[42] Jin Hyeok Yoo, Yecheol Kim, Ji Song Kim, and Jun Won Choi. 3d-cvf: Generating joint camera and lidar features using cross-view spatial feature fusion for 3d object detection. *arXiv preprint arXiv:2004.12636*, 3, 2020. 8

[43] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. *arXiv preprint arXiv:2012.09164*, 2020. 4

[44] Yin Zhou, Pei Sun, Yu Zhang, Dragomir Anguelov, Jiyang Gao, Tom Ouyang, James Guo, Jiquan Ngiam, and Vijay Vasudevan. End-to-end multi-view fusion for 3d object detection in lidar point clouds. In *Conference on Robot Learning*, pages 923–932. PMLR, 2020. 7

[45] Yin Zhou and Oncel 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*, pages 4490–4499, 2018. 1, 2, 8