PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection

Shaoshuai Shi[1] Chaoxu Guo[2,3] Li Jiang[4]
Zhe Wang[2] Jianping Shi[2] Xiaogang Wang[1] Hongsheng Li[1]

[1]Multimedia Laboratory, The Chinese University of Hong Kong
[2]SenseTime Research [3]NLPR, CASIA [4]CSE, CUHK

Abstract

We present a novel and high-performance 3D object detection framework, named PointVoxel-RCNN (PV-RCNN), for accurate 3D object detection from point clouds. Our proposed method deeply integrates both 3D voxel Convolutional Neural Network (CNN) and PointNet-based set abstraction to learn more discriminative point cloud features. It takes advantages of efficient learning and high-quality proposals of the 3D voxel CNN and the flexible receptive fields of the PointNet-based networks. Specifically, the proposed framework summarizes the 3D scene with a 3D voxel CNN into a small set of keypoints via a novel voxel set abstraction module to save follow-up computations and also to encode representative scene features. Given the high-quality 3D proposals generated by the voxel CNN, the RoI-grid pooling is proposed to abstract proposal-specific features from the keypoints to the RoI-grid points via keypoint set abstraction. Compared with conventional pooling operations, the RoI-grid feature points encode much richer context information for accurately estimating object confidences and locations. Extensive experiments on both the KITTI dataset and the Waymo Open dataset show that our proposed PV-RCNN surpasses state-of-the-art 3D detection methods with remarkable margins.

1. Introduction

3D object detection has been receiving increasing attention from both industry and academia thanks to its wide applications in various fields such as autonomous driving and robotics. LiDAR sensors are widely adopted in autonomous driving vehicles and robots for capturing 3D scene information as sparse and irregular point clouds, which provide vital cues for 3D scene perception and understanding. In this paper, we propose to achieve high performance 3D object detection by designing novel point-voxel integrated networks to learn better 3D features from irregular point clouds.

E-mail: {ssshi, hsli}@ee.cuhk.edu.hk

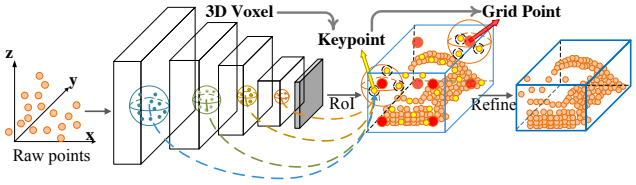

Figure 1. Our proposed PV-RCNN framework deeply integrates both the voxel-based and the PointNet-based networks via a two-step strategy including the voxel-to-keypoint 3D scene encoding and the keypoint-to-grid RoI feature abstraction for improving the performance of 3D object detection.

Most existing 3D detection methods could be classified into two categories in terms of point cloud representations, *i.e.*, the grid-based methods and the point-based methods. The grid-based methods generally transform the irregular point clouds to regular representations such as 3D voxels [29, 45, 37, 2, 28] or 2D bird-view maps [1, 12, 39, 18, 38, 13, 17, 41], which could be efficiently processed by 3D or 2D Convolutional Neural Networks (CNN) to learn point features for 3D detection. Powered by the pioneer work, PointNet and its variants [25, 26], the point-based methods [24, 27, 35, 40, 22] directly extract discriminative features from raw point clouds for 3D detection. Generally, the grid-based methods are more computationally efficient but the inevitable information loss degrades the fine-grained localization accuracy, while the point-based methods have higher computation cost but could easily achieve larger receptive field by the point set abstraction [26]. However, we show that a unified framework could integrate the best of the two types of methods, and surpass the prior state-of-the-art 3D detection methods with remarkable margins.

We propose a novel 3D object detection framework, **PV-RCNN** (Illustrated in Fig. 1), which boosts the 3D detection performance by incorporating the advantages from both the **P**oint-based and **V**oxel-based feature learning methods. The principle of PV-RCNN lies in the fact that the voxel-based operation efficiently encodes multi-scale feature representations and can generate high-quality 3D proposals, while the PointNet-based set abstraction operation

preserves accurate location information with flexible receptive fields. We argue that the integration of these two types of feature learning frameworks can help learn more discriminative features for accurate fine-grained box refinement.

The main challenge would be how to effectively combine the two types of feature learning schemes, specifically the 3D voxel CNN with sparse convolutions [6, 5] and the PointNet-based set abstraction [26], into a unified framework. An intuitive solution would be uniformly sampling several grid points within each 3D proposal, and adopt the set abstraction to aggregate 3D voxel-wise features surrounding these grid points for proposal refinement. However, this strategy is highly memory-intensive since both the number of voxels and the number of grid points could be quite large to achieve satisfactory performance.

Therefore, to better integrate these two types of point cloud feature learning networks, we propose a two-step strategy with the first voxel-to-keypoint scene encoding step and the second keypoint-to-grid RoI feature abstraction step. Specifically, a voxel CNN with 3D sparse convolution is adopted for voxel-wise feature learning and accurate proposal generation. To mitigate the above mentioned issue of requiring too many voxels for encoding the whole scene, a small set of keypoints are selected by the furtherest point sampling (FPS) to summarize the overall 3D information from the voxel-wise features. The features of each keypoint is aggregated by grouping the neighboring voxel-wise features via PointNet-based set abstraction for summarizing multi-scale point cloud information. In this way, the overall scene can be effectively and efficiently encoded by a small number of keypoints with associated multi-scale features.

For the second keypoint-to-grid RoI feature abstraction step, given each box proposal with its grid point locations, a RoI-grid pooling module is proposed, where a keypoint set abstraction layer with multiple radii is adopted for each grid point to aggregate the features from the keypoints with multi-scale context. All grid points' aggregated features can then be jointly used for the succeeding confidence prediction and fine-grained box refinement.

Our contributions can be summarized into four-fold. (1) We propose PV-RCNN framework which effectively takes advantages of both the voxel-based and point-based methods for 3D point-cloud feature learning, leading to improved performance of 3D object detection with manageable memory consumption. (2) We propose the voxel-to-keypoint scene encoding scheme, which encodes multi-scale voxel features of the whole scene to a small set of keypoints by the voxel set abstraction layer. These keypoint features not only preserve accurate location but also encode rich scene context, which boosts the 3D detection performance significantly. (3) We propose a multi-scale RoI feature abstraction layer for grid points in each proposal, which aggregates richer context information from the scene for accurate box refinement and confidence prediction. (4) Our proposed method PV-RCNN outperforms all previous methods with remarkable margins and ranks 1^{st} on the highly competitive KITTI 3D detection benchmark [11], and also surpasses previous methods on the large-scale Waymo Open dataset with a large margin.

2. Related Work

3D Object Detection with Grid-based Methods. To tackle the irregular data format of point clouds, most existing works project the point clouds to regular grids to be processed by 2D or 3D CNN. The pioneer work MV3D [1] projects the point clouds to 2D bird view grids and places lots of predefined 3D anchors for generating 3D bounding boxes, and the following works [12, 18, 17] develop better strategies for multi-sensor fusion while [39, 38, 13] propose more efficient frameworks with bird view representation. Some other works [29, 45] divide the point clouds into 3D voxels to be processed by 3D CNN, and 3D sparse convolution [5] is introduced [37] for efficient 3D voxel processing. [33, 46] utilizes multiple detection heads while [28] explores the object part locations for improving the performance. These grid-based methods are generally efficient for accurate 3D proposal generation but the receptive fields are constraint by the kernel size of 2D/3D convolutions.

3D Object Detection with Point-based Methods. F-PointNet [24] first proposes to apply PointNet [25, 26] for 3D detection from the cropped point clouds based on the 2D image bounding boxes. PointRCNN [27] generates 3D proposals directly from the whole point clouds instead of 2D images for 3D detection with point clouds only, and the following work STD [40] proposes the sparse to dense strategy for better proposal refinement. [23] proposes the hough voting strategy for better object feature grouping. These point-based methods are mostly based on the PointNet series, especially the set abstraction operation [26], which enables flexible receptive fields for point cloud feature learning.

Representation Learning on Point Clouds. Representation learning on point clouds has drawn lots of attention on improving the performance of point cloud classification and segmentation [25, 26, 45, 34, 7, 42, 16, 30, 36, 8, 32, 10, 21, 3]. In terms of 3D detection, previous methods generally project the point clouds to regular bird view grids [1, 39] or 3D voxels [45, 2] for processing point clouds with 2D/3D CNN. 3D sparse convolution [6, 5] are adopted in [37, 28] to effectively learn sparse voxel-wise features from the point clouds. Qi *et al*. [25, 26] proposes the PointNet to directly learn point-wise features from the raw point clouds, where set abstraction operation enables flexible receptive fields by setting different search radii. [20] combines both voxel-based CNN and point-based SharedMLP for efficient point cloud feature learning. In comparison, our proposed PV-RCNN takes advantages from both the voxel-based feature

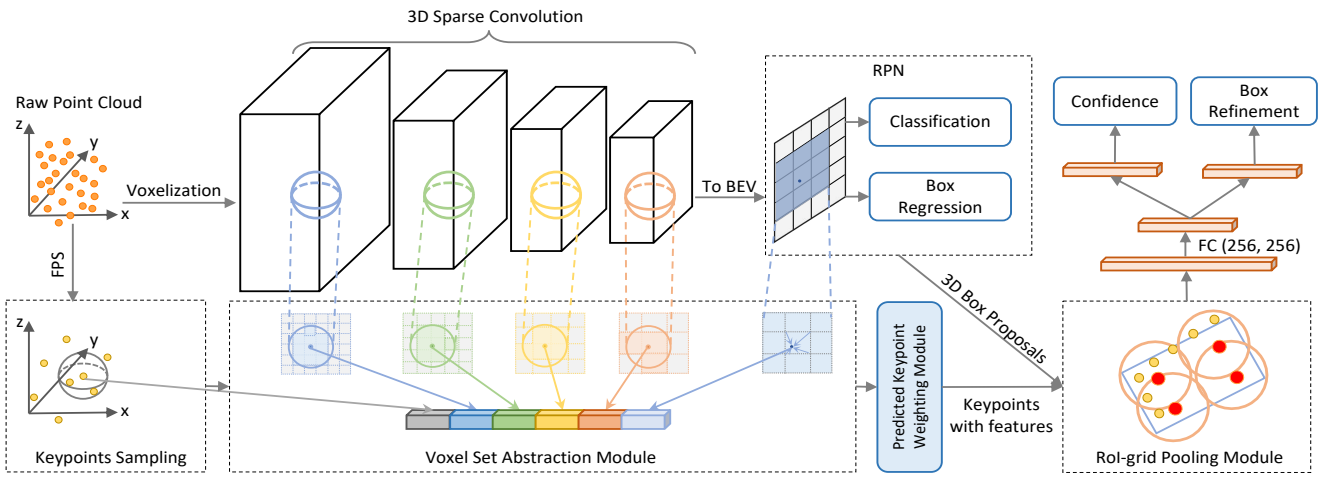

Figure 2. The overall architecture of our proposed PV-RCNN. The raw point clouds are first voxelized to feed into the 3D sparse convolution based encoder to learn multi-scale semantic features and generate 3D object proposals. Then the learned voxel-wise feature volumes at multiple neural layers are summarized into a small set of key points via the novel voxel set abstraction module. Finally the keypoint features are aggregated to the RoI-grid points to learn proposal specific features for fine-grained proposal refinement and confidence prediction.

learning (*i.e.*, 3D sparse convolution) and PointNet-based feature learning (*i.e.*, set abstraction operation) to enable both high-quality 3D proposal generation and flexible receptive fields for improving the 3D detection performance.

3. PV-RCNN for Point Cloud Object Detection

In this paper, we propose the PV-RCNN, a two-stage 3D detection framework aiming at more accurate 3D object detection from point clouds. State-of-the-art 3D detection approaches are based on either 3D voxel CNN with sparse convolution or PointNet-based networks as the backbone. Generally, the 3D voxel sparse CNNs are more efficient [37, 28] and are able to generate high-quality 3D proposals, while the PointNet-based methods can capture more accurate contextual information with flexible receptive fields.

Our PV-RCNN deeply integrates the advantages of two types of networks. As illustrated in Fig. 2, the PV-RCNN consists of a 3D voxel CNN with sparse convolution as the backbone for efficient feature encoding and proposal generation. Given each 3D proposal, to effectively pool its corresponding features from the scene, we propose two novel operations: the voxel-to-keypoint scene encoding, which summarizes all the voxels of the overall scene feature volumes into a small number of feature keypoints, and the point-to-grid RoI feature abstraction, which effectively aggregates the scene keypoint features to RoI grids for proposal confidence prediction and location refinement.

3.1. 3D Voxel CNN for Efficient Feature Encoding and Proposal Generation

Voxel CNN with 3D sparse convolution [6, 5, 37, 28] is a popular choice by state-of-the-art 3D detectors for efficiently converting the point clouds into sparse 3D feature volumes. Because of its high efficiency and accuracy, we adopt it as the backbone of our framework for feature encoding and 3D proposal generation.

3D voxel CNN. The input points $\mathbf{P}$ are first divided into small voxels with spatial resolution of $L \times W \times H$, where the features of the non-empty voxels are directly calculated as the mean of point-wise features (*i.e.*, 3D coordinates, reflectance intensities) of all inside points. The network utilizes a series of $3 \times 3 \times 3$ 3D sparse convolution to gradually convert the point clouds into feature volumes with $1\times$, $2\times$, $4\times$, $8\times$ downsampled sizes. Such sparse feature volumes could be viewed as a set of voxel-wise feature vectors.

3D proposal generation. By converting the encoded $8\times$ downsampled 3D feature volumes into 2D bird-view feature maps, high-quality 3D proposals are generated following the anchor-based approaches [37, 13]. Specifically, we stack the 3D feature volume along the Z axis to obtain the $\frac{L}{8} \times \frac{W}{8}$ bird-view feature maps. Each class has $2 \times \frac{L}{8} \times \frac{W}{8}$ 3D anchor boxes which adopt the average 3D object sizes of this class, and two anchors of $0°, 90°$ orientations are evaluated for each pixel of the bird-view feature maps. As shown in Table 4, the adopted 3D voxel CNN backbone with anchor-based scheme achieves higher recall performance than the PointNet-based approaches [27, 40].

Discussions. State-of-the-art detectors mostly adopt two-stage frameworks. They require pooling RoI specific features from the resulting 3D feature volumes or 2D maps for further proposal refinement. However, these 3D feature volumes from the 3D voxel CNN have major limitations in the following aspects. (i) These feature volumes are generally of low spatial resolution as they are downsampled by up to 8 times, which hinders accurate localization of objects in the input scene. (ii) Even if one can upsample to obtain feature volumes/maps of larger spatial sizes, they are generally still quite sparse. The commonly used trilinear or bilin-

ear interpolation in the RoIPooling/RoIAlign operations can only extract features from very small neighborhoods (i.e., 4 and 8 nearest neighbors for bilinear and trilinear interpolation respectively). The conventional pooling approaches would therefore obtain features with mostly zeros and waste much computation and memory for stage-2 refinement.

On the other hand, the set abstraction operation proposed in the variants of PointNet [25, 26] has shown the strong capability of encoding feature points from a neighborhood of an arbitrary size. We therefore propose to integrate a 3D voxel CNN with a series of set abstraction operations for conducting accurate and robust stage-2 proposal refinement.

A naive solution of using the set abstraction operation for pooling the scene feature voxels would be directly aggregating the multi-scale feature volume to the RoI grids. However, this intuitive strategy simply occupies much GPU memory for calculating the pairwise distances in the set abstractions due to the large number of sparse voxels.

To tackle this issue, we propose a two-step approach to first encode voxels at different neural layers of the entire scene into a small number of keypoints and then aggregate keypoint features to RoI grids for box proposal refinement.

3.2. Voxel-to-keypoint Scene Encoding via Voxel Set Abstraction

Our framework first aggregates the multi-scale feature voxels representing the entire scene into a small number of keypoints, which serve as a bridge between the 3D voxel CNN feature encoder and the proposal refinement network.

Keypoints Sampling. Specifically, we adopt the Furthest-Point-Sampling (FPS) algorithm to sample a small number of n keypoints $\mathcal{K} = \{p_1, \cdots, p_n\}$ from the point clouds $\mathbf{P}$, where $n = 2,048$ for the KITTI dataset and $n = 4,096$ for the Waymo dataset. Such a strategy encourages that the keypoints are uniformly distributed around non-empty voxels and can be representative to the overall scene.

Voxel Set Abstraction Module. We propose the *Voxel Set Abstraction* (VSA) module to encode the multi-scale semantic features from the 3D CNN feature volumes to the keypoints. The set abstraction operation proposed by [26] is adopted for the aggregation of voxel-wise feature volumes. The surrounding points of keypoints are now regular voxels with multi-scale semantic features encoded by the 3D voxel CNN from the multiple layers, instead of the neighboring raw points with features learned from PointNet as in [26].

Specifically, denote $\mathcal{F}^{(l_k)} = \{f_1^{(l_k)}, \cdots, f_{N_k}^{(l_k)}\}$ as the set of voxel-wise feature vectors in the k-th level of 3D voxel CNN, $\mathcal{V}^{(l_k)} = \{v_1^{(l_k)}, \cdots, v_{N_k}^{(l_k)}\}$ as their 3D coordinates calculated by the voxel indices and actual voxel sizes of the k-th level, where N_k is the number of non-empty voxels in the k-th level. For each keypoint p_i, we first identify its neighboring non-empty voxels at the k-th level within a radius r_k to retrieve the set of voxel-wise feature vectors as

$$S_i^{(l_k)} = \left\{ \left[f_j^{(l_k)}; v_j^{(l_k)} - p_i \right]^T \middle| \begin{array}{l} \left\| v_j^{(l_k)} - p_i \right\|^2 < r_k, \\ \forall v_j^{(l_k)} \in \mathcal{V}^{(l_k)}, \\ \forall f_j^{(l_k)} \in \mathcal{F}^{(l_k)} \end{array} \right\}, \quad (1)$$

where we concatenate the local relative coordinates $v_j^{(l_k)} - p_i$ to indicate the relative location of semantic voxel feature $f_j^{(l_k)}$. The voxel-wise features within the neighboring voxel set $S_i^{(l_k)}$ of p_i are then transformed by a PointNet-block [25] to generate the feature for the key point p_i as

$$f_i^{(pv_k)} = \max \left\{ G \left(\mathcal{M} \left(S_i^{(l_k)} \right) \right) \right\}, \quad (2)$$

where $\mathcal{M}(\cdot)$ denotes randomly sampling at most T_k voxels from the neighboring set $S_i^{(l_k)}$ for saving computations, $G(\cdot)$ denotes a multi-layer perceptron network to encode the voxel-wise features and relative locations. Although the number of neighboring voxels varies across different keypoints, the along-channel max-pooling operation $\max(\cdot)$ maps the diverse number of neighboring voxel feature vectors to a feature vector $f_i^{(pv_k)}$ for the key point p_i. Generally, we also set multiple radii r_k at the k-th level to aggregate local voxel-wise features with different receptive fields for capturing richer multi-scale contextual information.

The above strategy is performed at different levels of the 3D voxel CNN, and the aggregated features from different levels can be concatenated to generate the multi-scale semantic feature for the key point p_i

$$f_i^{(pv)} = \left[f_i^{(pv_1)}, f_i^{(pv_2)}, f_i^{(pv_3)}, f_i^{(pv_4)} \right], \text{ for } i = 1, \cdots, n, \quad (3)$$

where the generated feature $f_i^{(pv)}$ incorporates both the 3D voxel CNN-based feature learning from voxel-wise feature $f_j^{(l_k)}$ and the PointNet-based features from voxel set abstraction as Eq. (2). Besides, the 3D coordinate of p_i also preserves accurate location information.

Extended VSA Module. We extend the VSA module by further enriching the keypoint features from the raw point clouds $\mathbf{P}$ and the $8\times$ downsampled bird-view feature maps, where the raw point clouds partially make up the quantization loss of the point-cloud voxelization while the 2D bird-view maps have larger receptive fields along the Z axis. The raw point-cloud feature $f_i^{(raw)}$ is also aggregated as in Eq. (2), while bird-view feature $f_i^{(bev)}$ of keypoint p_i are obtained by bilinear interpolation on the bird-view feature maps. Hence, the keypoint feature for p_i is further enriched by concatenating all its associated features

$$f_i^{(p)} = \left[f_i^{(pv)}, f_i^{(raw)}, f_i^{(bev)} \right], \text{ for } i = 1, \cdots, n, \quad (4)$$

which have strong capability of preserving 3D structural information of the entire scene to boost the final performance.

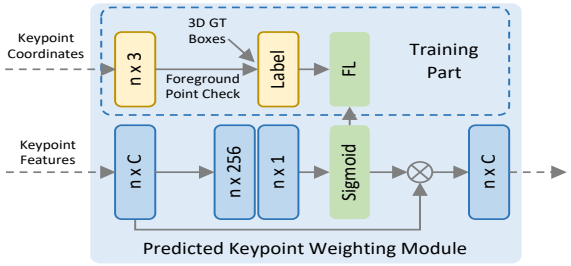

Figure 3. Illustration of Predicted Keypoint Weighting module.

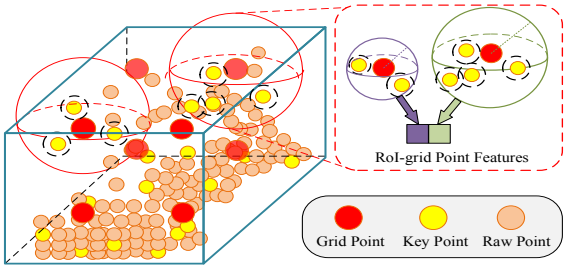

Figure 4. Illustration of RoI-grid pooling module. Rich context information of each 3D RoI is aggregated by the set abstraction operation with multiple receptive fields.

Predicted Keypoint Weighting. After the overall scene is encoded by a small number of keypoints, they would be further utilized by the succeeding stage for conducting proposal refinement. The keypoints are chosen by the Further Point Sampling strategy and some of them might only represent the background regions. Intuitively, keypoints belonging to the foreground objects should contribute more to the accurate refinement of the proposals, while the ones from the background regions should contribute less.

Hence, we propose a *Predicted Keypoint Weighting* (PKW) module (see Fig. 3) to re-weight the keypoint features with extra supervisions from point-cloud segmentation. The segmentation labels can be directly generated by the 3D detection box annotations, *i.e.* by checking whether each keypoint is inside or outside of a ground-truth 3D box. The predicted feature weighting for each keypoint's feature $\tilde{f}_i^{(p)}$ can be formulated as

$$\tilde{f}_i^{(p)} = \mathcal{A}(f_i^{(p)}) \cdot f_i^{(p)}, \tag{5}$$

where $\mathcal{A}(\cdot)$ is a three-layer MLP network with a sigmoid function to predict foreground confidence between $[0, 1]$. The PKW module is trained by focal loss [19] with default hyper-parameters for handling the unbalanced number of foreground/background points in the training set.

3.3. Keypoint-to-grid RoI Feature Abstraction for Proposal Refinement

In the previous step, the whole scene is summarized into a small number of keypoints with multi-scale semantic features. Given each 3D proposal (RoI) generated by the 3D voxel CNN, the features of each RoI need to be aggregated from the keypoint features $\tilde{\mathcal{F}} = \{\tilde{f}_1^{(p)}, \cdots, \tilde{f}_n^{(p)}\}$ for accurate and robust proposal refinement. We propose the keypoint-to-grid RoI feature abstraction based on the set abstraction operation for multi-scale RoI feature encoding.

RoI-grid Pooling via Set Abstraction. Given each 3D RoI, as shown in Fig. 4, we propose the RoI-grid pooling module to aggregate the keypoint features to the RoI-grid points with multiple receptive fields. We uniformly sample $6 \times 6 \times 6$ grid points within each 3D proposal, which are denoted as $\mathcal{G} = \{g_1, \cdots, g_{216}\}$. The set abstraction operation is adopted to aggregate the features of grid points from the keypoint features. Specifically, we firstly identify the neighboring keypoints of grid point g_i within a radius $\tilde{r}$ as

$$\tilde{\Psi} = \left\{ \left[\tilde{f}_j^{(p)}; p_j - g_i \right]^T \;\middle|\; \begin{array}{l} \|p_j - g_i\|^2 < \tilde{r}, \\ \forall p_j \in \mathcal{K}, \forall \tilde{f}_j^{(p)} \in \tilde{\mathcal{F}} \end{array} \right\}, \tag{6}$$

where $p_j - g_i$ is appended to indicate the local relative location of features $\tilde{f}_j^{(p)}$ from keypoint p_j. Then a PointNet-block [25] is adopted to aggregate the neighboring keypoint feature set $\tilde{\Psi}$ to generate the feature for grid point g_i as

$$\tilde{f}_i^{(g)} = \max \left\{ G\left(\mathcal{M}\left(\tilde{\Psi} \right) \right) \right\}, \tag{7}$$

where $\mathcal{M}(\cdot)$ and $G(\cdot)$ are defined as the same in Eq. (2). We set multiple radii $\tilde{r}$ and aggregate keypoint features with different receptive fields, which are concatenated together for capturing richer multi-scale contextual information.

After obtaining each grid's aggregated features from its surrounding keypoints, all RoI-grid features of the same RoI can be vectorized and transformed by a two-layer MLP with 256 feature dimensions to represent the overall proposal.

Compared with the point cloud 3D RoI pooling operations in previous works [27, 40, 28], our proposed RoI-grid pooling operation targeting the keypoints is able to capture much richer contextual information with flexible receptive fields, where the receptive fields are even beyond the RoI boundaries for capturing the surrounding keypoint features outside the 3D RoI, while the previous state-of-the-art methods either simply average all point-wise features within the proposal as the RoI feature [27], or pool many uninformative zeros as the RoI features [28, 40].

3D Proposal Refinement and Confidence Prediction. Given the RoI feature of each box proposal, the proposal refinement network learns to predict the size and location (*i.e.*, center, size and orientation) residuals relative to the input 3D proposal. The refinement network adopts a 2-layer MLP and has two branches for confidence prediction and box refinement respectively.

For the confidence prediction branch, we follow [15, 9, 28] to adopt the 3D Intersection-over-Union (IoU) between the 3D RoIs and their corresponding ground-truth boxes as the training targets. For the k-th 3D RoI, its confidence training target y_k is normalized to be between $[0, 1]$ as

$$y_k = \min\left(1,\ \max\left(0,\ 2\text{IoU}_k - 0.5\right)\right), \tag{8}$$

where IoU_k is the IoU of the k-th RoI w.r.t. its ground-truth box, and this confidence branch is optimized with the binary cross entropy loss. Our experiments in Table 8 show that this quality-aware confidence prediction strategy achieves better performance than the traditional classification targets.

The box regression targets of the box refinement branch are encoded by the traditional residual-based method as in [37, 28] and are optimized by smooth-L1 loss function.

4. Experiments

In this section, we introduce the implementation details of our PV-RCNN (Sec. 4.1) and compare with previous state-of-the-art methods on both the highly competitive KITTI dataset [4] (Sec. 4.2) and the newly introduced large-scale Waymo Open Dataset [31, 22, 44] (Sec. 4.3). In Sec. 4.4, we conduct extensive ablation studies to investigate each component of PV-RCNN to validate our design.

4.1. Experimental Setup

Datasets. *KITTI Dataset* [4] is one of the most popular dataset of 3D detection for autonomous driving. There are $7,481$ training samples and $7,518$ test samples, where the training samples are generally divided into the *train* split ($3,712$ samples) and the *val* split ($3,769$ samples).

Waymo Open Dataset is a recently released and currently the largest dataset of 3D detection for autonomous driving. There are totally 798 training sequences with around $158,361$ LiDAR samples, and 202 validation sequences with $40,077$ LiDAR samples. It annotated the objects in the full 360° field instead of 90° in KITTI dataset. We evaluate our model on this large-scale dataset to further validate the effectiveness of our proposed method.

Network Architecture. As shown in Fig. 2, the 3D voxel CNN has four levels with feature dimensions $16, 32, 64, 64$, respectively. Their two neighboring radii r_k of each level in the VSA module are set as $(0.4\text{m}, 0.8\text{m})$, $(0.8\text{m}, 1.2\text{m})$, $(1.2\text{m}, 2.4\text{m})$, $(2.4\text{m}, 4.8\text{m})$, and the neighborhood radii of set abstraction for raw points are $(0.4\text{m}, 0.8\text{m})$. For the proposed RoI-grid pooling operation, we uniformly sample $6 \times 6 \times 6$ grid points in each 3D proposal and the two neighboring radii $\tilde{r}$ of each grid point are $(0.8\text{m}, 1.6\text{m})$.

For the KITTI dataset, the detection range is within $[0, 70.4]m$ for the X axis, $[-40, 40]m$ for the Y axis and $[-3, 1]m$ for the Z axis, which is voxelized with the voxel size $(0.05m, 0.05m, 0.1m)$ in each axis. For the Waymo Open dataset, the detection range is $[-75.2, 75.2]m$ for the X and Y axes and $[-2, 4]m$ for the Z axis, and we set the voxel size to $(0.1m, 0.1m, 0.15m)$.

Training and Inference Details. Our PV-RCNN framework is trained from scratch in an end-to-end manner with the ADAM optimizer. For the KITTI dataset, we train the entire network with the batch size 24, learning rate 0.01 for 80 epochs on 8 GTX 1080 Ti GPUs, which takes around 5 hours. For the Waymo Open Dataset, we train the entire network with batch size 64, learning rate 0.01 for 50 epochs on 32 GTX 1080 Ti GPUs, which takes around 25 hours. The cosine annealing learning rate strategy is adopted for the learning rate decay. For the proposal refinement stage, we randomly sample 128 proposals with $1{:}1$ ratio for positive and negative proposals, where a proposal is considered as a positive proposal for box refinement branch if it has at least 0.55 3D IoU with the ground-truth boxes, otherwise it is treated as a negative proposal.

During training, we utilize the widely adopted data augmentation strategy of 3D object detection, including random flipping along the X axis, global scaling with a random scaling factor sampled from $[0.95, 1.05]$, global rotation around the Z axis with a random angle sampled from $[-\frac{\pi}{4}, \frac{\pi}{4}]$. We also conduct the ground-truth sampling augmentation [37] to randomly "paste" some new ground-truth objects from other scenes to the current training scenes, for simulating objects in various environments.

For inference, we keep the top-100 proposals generated from the 3D voxel CNN with a 3D IoU threshold of 0.7 for non-maximum-suppression (NMS). These proposals are further refined in the proposal refinement stage with aggregated keypoint features. We finally use an NMS threshold of 0.01 to remove the redundant boxes.

4.2. 3D Detection on the KITTI Dataset

To evaluate the proposed model's performance on the KITTI *val* split, we train our model on the *train* set and report the results on the *val* set. To conduct evaluation on the *test* set with the KITTI official test server, the model is trained with 80% of all available *train+val* data and the remaining 20% data is used for validation.

Evaluation Metric. All results are evaluated by the mean average precision with a rotated IoU threshold 0.7 for $cars$ and 0.5 for $pedestrian$ and $cyclists$. The mean average precisions on the *test* set are calculated with 40 recall positions on the official KITTI test server [11]. The results on the *val* set in Table 2 are calculated with 11 recall positions to compare with the results by the previous works.

Comparison with state-of-the-art methods. Table 1 shows the performance of PV-RCNN on the KITTI *test* set from the official online leaderboard. For the most important 3D object detection benchmark of the car class, our method outperforms previous state-of-the-art methods with remarkable margins, *i.e.* increasing the mAP by 1.58%, 1.72%, 1.73% on easy, moderate and hard difficulty levels, respectively. For the bird-view detection of the car class, our method also achieves new state-of-the-art performance on the easy and moderate difficulty levels while dropping slightly on the hard difficulty level. For the performance of pedestrian and cyclist, our method achieves better or comparable results on all the moderate and hard difficulty levels

Method	Modality	Car - 3D Detection			Car - BEV Detection			Ped. - 3D Detection			Ped. - BEV Detection			Cyc. - 3D Detection			Cyc. - BEV Detection		
		Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
MV3D [1]	RGB + LiDAR	74.97	63.63	54.00	86.62	78.93	69.80	-	-	-	-	-	-	-	-	-	-	-	-
ContFuse [18]	RGB + LiDAR	83.68	68.78	61.67	94.07	85.35	75.88	-	-	-	-	-	-	-	-	-	-	-	-
AVOD-FPN [12]	RGB + LiDAR	83.07	71.76	65.73	90.99	84.82	79.62	50.46	42.27	39.04	58.49	50.32	46.98	63.76	50.55	44.93	69.39	57.12	51.09
F-PointNet [24]	RGB + LiDAR	82.19	69.79	60.59	91.17	84.67	74.77	50.53	42.15	38.08	57.13	49.57	45.48	72.27	56.12	49.01	77.26	61.37	53.78
F-ConvNet [35]	RGB + LiDAR	87.36	76.39	66.69	91.51	85.84	76.11	52.16	43.38	38.80	57.04	48.96	44.33	81.98	65.07	56.54	84.16	68.88	60.05
UberATG-MMF [17]	RGB + LiDAR	88.40	77.43	70.22	93.67	88.21	81.99	-	-	-	-	-	-	-	-	-	-	-	-
SECOND-V1.5 [37]	LiDAR only	84.65	75.96	68.71	91.81	86.37	81.04	-	-	-	-	-	-	-	-	-	-	-	-
PointPillars [13]	LiDAR only	82.58	74.31	68.99	90.07	86.56	82.81	51.45	41.92	38.89	57.60	48.64	45.78	77.10	58.65	51.92	79.90	62.73	55.58
PointRCNN [27]	LiDAR only	86.96	75.64	70.70	92.13	87.39	82.72	47.98	39.37	36.01	54.77	46.13	42.84	74.96	58.82	52.53	82.56	67.24	60.28
3D IoU Loss [43]	LiDAR only	86.16	76.50	71.39	91.36	86.22	81.20	-	-	-	-	-	-	-	-	-	-	-	-
Fast Point R-CNN [2]	LiDAR only	85.29	77.40	70.24	90.87	87.84	80.52	-	-	-	-	-	-	-	-	-	-	-	-
STD [40]	LiDAR only	87.95	79.71	75.09	94.74	89.19	**86.42**	**53.29**	42.47	38.35	**60.02**	48.72	44.55	78.69	61.59	55.30	81.36	67.23	59.35
Patches [14]	LiDAR only	88.67	77.20	71.82	92.72	88.39	83.19	-	-	-	-	-	-	-	-	-	-	-	-
Part-A^2 [28]	LiDAR only	87.81	78.49	73.51	91.70	87.79	84.61	53.10	**43.35**	40.06	59.04	49.81	45.92	**79.17**	63.52	56.93	**83.43**	68.73	61.85
PV-RCNN (Ours)	LiDAR only	**90.25**	**81.43**	**76.82**	**94.98**	**90.65**	86.14	52.17	43.29	**40.29**	59.86	**50.57**	**46.74**	78.60	**63.71**	**57.65**	82.49	**68.89**	**62.41**
Improvement	-	*+1.58*	*+1.72*	*+1.73*	*+0.24*	*+1.46*	*-0.28*	*-1.12*	*-0.06*	*+0.23*	*-0.16*	*+0.76*	*+0.82*	*-0.57*	*+0.19*	*+0.72*	*-0.94*	*+0.16*	*+0.56*

Table 1. Performance comparison on the KITTI *test* set. The results are evaluated by the mean Average Precision with 40 recall positions.

Method	Reference	Modality	3D mAP
MV3D [1]	CVPR 2017	RGB + LiDAR	62.68
ContFuse[18]	ECCV 2018	RGB + LiDAR	73.25
AVOD-FPN [12]	IROS 2018	RGB + LiDAR	74.44
F-PointNet [24]	CVPR 2018	RGB + LiDAR	70.92
VoxelNet [45]	CVPR 2018	LiDAR only	65.46
SECOND [37]	Sensors 2018	LiDAR only	76.48
PointRCNN [27]	CVPR 2019	LiDAR only	78.63
Fast Point R-CNN [2]	ICCV 2019	LiDAR only	79.00
STD [40]	ICCV 2019	LiDAR only	79.80
PV-RCNN (Ours)	-	LiDAR only	**83.90**

Table 2. Performance comparison on the moderate level car class of KITTI *val* split with mAP calculated by 11 recall positions.

IoU	3D mAP			BEV mAP		
Thresh.	Easy	Moderate	Hard	Easy	Moderate	Hard
0.7	92.57	84.83	82.69	95.76	91.11	88.93

Table 3. Performance on the KITTI *val* split set with mAP calculated by 40 recall positions for car class.

Method	PointRCNN [27]	STD [40]	PV-RCNN (Ours)
Recall (IoU=0.7)	74.8	76.8	85.5

Table 4. Recall of different proposal generation networks on the car class at moderate difficulty level of the KITTI *val* split set.

while achieving slightly worse results on the easy difficulty levels, where we think the limited number of keypoints may harm the performance of the objects with small sizes.

As of Nov. 15th, 2019, our method ranks 1^{st} on the car 3D detection leaderboard among all methods including both the RGB+LiDAR methods and LiDAR-only methods, and ranks 1^{st} on the cyclist 3D detection leaderboard among all published LiDAR-only methods. The significant improvements manifest the effectiveness of the PV-RCNN.

We also report the performance of the most important car class on the KITTI *val* split with mAP from $R11$. Similarly, as shown in Table 2, our method outperforms previous state-of-the-art methods with large margins. The performance with $R40$ are also provided in Table 3 for reference.

4.3. 3D Detection on the Waymo Open Dataset

To further validate the effectiveness of our proposed PV-RCNN, we evaluate the performance of PV-RCNN on the newly released large-scale Waymo Open Dataset.

Evaluation Metric. We adopt the official released evaluation tools for evaluating our method, where the mean average precision (mAP) and the mean average precision weighted by heading (mAPH) are used for evaluation. The rotated IoU threshold is set as 0.7 for vehicle. The test data are split in two ways. The first way is based on objects' different distances to the sensor: $0 - 30m$, $30 - 50m$ and $> 50m$. The second way is to split the data into two difficulty levels, where the LEVEL_1 denotes the ground-truth objects with more than 5 inside points while the LEVEL_2 denotes the ground-truth objects with at least 1 inside points.

Comparison with state-of-the-art methods. Table 5 shows that our method outperforms previous state-of-the-art [44] significantly with a 7.37% mAP gain for the 3D object detection and a 2.56% mAP gain for the bird-view object detection. The results show that our method achieves remarkably better mAP on all distance ranges of interest, where the maximum gain is 9.19% for the 3D detection in the range of $30 - 50m$, which validates that our proposed multi-scale point-voxel integration strategy is able to effectively capture more accurate contextual information for improving the 3D detection performance. As shown in Table 5, our method also achieves superior performance in terms of mAPH, which demonstrates that our model predicted accurate heading direction for the vehicles. The results on the LEVEL_2 difficult level are also reported in Table 5 for reference, and we could see that our method performs well even for the objects with fewer than 5 inside points. The experimental results on the large-scale Waymo Open dataset further validate the generalization ability of our proposed framework on various datasets.

4.4. Ablation Studies

In this section, we conduct extensive ablation experiments to analyze individual components of our proposed method. All models are trained on the *train* split and evaluated on the *val* split for the car class of KITTI dataset [4].

Effects of voxel-to-keypoint scene encoding. We validate the effectiveness of voxel-to-keypoint scene encoding strategy by comparing with the native solution that directly

Difficulty	Method	3D mAP (IoU=0.7)				3D mAPH (IoU=0.7)				BEV mAP (IoU=0.7)				BEV mAPH (IoU=0.7)			
		Overall	0-30m	30-50m	50m-Inf	Overall	0-30m	30-50m	50m-Inf	Overall	0-30m	30-50m	50m-Inf	Overall	0-30m	30-50m	50m-Inf
LEVEL_1	PointPillar [13]	56.62	81.01	51.75	27.94	-	-	-	-	75.57	92.1	74.06	55.47	-	-	-	-
	MVF [44]	62.93	86.30	60.02	36.02	-	-	-	-	80.40	93.59	79.21	63.09	-	-	-	-
	PV-RCNN (Ours)	**70.30**	**91.92**	**69.21**	**42.17**	**69.69**	**91.34**	**68.53**	**41.31**	**82.96**	**97.35**	**82.99**	**64.97**	**82.06**	**96.71**	**82.01**	**63.15**
	Improvement	*+7.37*	*+5.62*	*+9.19*	*+6.15*	-	-	-	-	*+2.56*	*+3.76*	*+3.78*	*+1.88*	-	-	-	-
LEVEL_2	PV-RCNN (Ours)	65.36	91.58	65.13	36.46	64.79	91.00	64.49	35.70	77.45	94.64	80.39	55.39	76.60	94.03	79.40	53.82

Table 5. Performance comparison on the Waymo Open Dataset with 202 validation sequences for the vehicle detection. Note that the results of PointPillar [13] on the Waymo Open Dataset are reproduced by [44].

Method	RPN with 3D Voxel CNN	Keypoints Encoding	RoI-grid Pooling	Easy	Mod.	Hard
RPN Baseline	✓			90.46	80.87	77.30
Pool from Encoder	✓		✓	91.88	82.86	80.52
PV-RCNN	✓	✓	✓	**92.57**	**84.83**	**82.69**

Table 6. Effects of voxel-to-keypoint scene encoding strategy and RoI-grid pooling refinement.

$f_i^{(pv_1)}$	$f_i^{(pv_2)}$	$f_i^{(pv_3)}$	$f_i^{(pv_4)}$	$f_i^{(bev)}$	$f_i^{(raw)}$	Moderate mAP
					✓	81.98
				✓		83.32
			✓			83.17
		✓	✓			84.54
		✓	✓	✓		84.69
		✓	✓	✓	✓	84.72
	✓	✓	✓	✓	✓	84.75
✓	✓	✓	✓	✓	✓	**84.83**

Table 7. Effects of different feature components for VSA module.

aggregating feature volumes from encoder to the RoI-grid points (see Sec. 3.1). As shown in the 2^{nd} and 3^{rd} rows of Table 6, the voxel-to-keypoint scene encoding strategy contributes significantly to the performance in all difficulty levels. This benefits from that the keypoints enlarge the receptive fields by bridging the 3D voxel CNN and RoI-grid points, and the segmentation supervision of keypoints also enables a better multi-scale feature learning from the 3D voxel CNN. Besides, a small set of keypoints as the intermediate feature representation also decreases the GPU memory usage when compared with the directly pooling strategy.

Effects of different features for VSA module. In Table 7, we investigate the importance of each feature component of keypoints in Eq. (3) and Eq. (4). The 1^{st} row shows that the performance drops a lot if we only aggregate features from $f_i^{(raw)}$, since the shallow semantic information is not enough for the proposal refinement. The high level semantic information from $f_i^{(pv_3)}$, $f_i^{(pv_4)}$ and $f_i^{(bev)}$ improves the performance significantly as shown in 2^{nd} to 5^{th} rows. As shown in last four rows, the additions of relative shallow semantic features $f_i^{(pv_1)}$, $f_i^{(pv_2)}$, $f_i^{(raw)}$ further improves the performance slightly and the best performance is achieved with all the feature components as the keypoint features.

Effects of PKW module. We propose the predicted keypoint weighting (PKW) module in Sec. 3.2 to re-weight the point-wise features of keypoint with extra keypoint segmentation supervision. Table 8 (1^{st} and 4^{th} rows) shows that removing the PKW module drops performance a lot, which demonstrates that the PKW module enables better multi-scale feature aggregation by focusing more on the foreground keypoints, since they are more important for the succeeding proposal refinement network.

PKW	RoI Pooling	Confidence Prediction	Easy	Moderate	Hard
✗	RoI-grid Pooling	IoU-guided scoring	92.09	82.95	81.93
✓	RoI-aware Pooling	IoU-guided scoring	92.54	82.97	80.30
✓	RoI-grid Pooling	Classification	91.71	82.50	81.41
✓	RoI-grid Pooling	IoU-guided Scoring	**92.57**	**84.83**	**82.69**

Table 8. Effects of predicted keypoint weighting module, RoI-grid pooling module and IoU-guided confidence prediction.

Effects of RoI-grid pooling module. We investigate the effects of RoI-grid pooling module by replacing it with the RoI-aware pooling [28] and keeping the other modules consistent. Table 8 shows that the performance drops significantly when replacing RoI-grid pooling module, which validates that our proposed set abstraction based RoI-grid pooling could learn much richer contextual information, and the pooled features also encode more discriminative RoI features by pooling more effective features with large search radii for each grid point. 1^{st} and 2^{nd} rows of Table 6 also shows that comparing with the 3D voxel RPN, the performance increases a lot after the proposal is refined by the features aggregated from the RoI-grid pooling module.

5. Conclusion

We have presented the PV-RCNN framework, a novel method for accurate 3D object detection from point clouds. Our method integrates both the multi-scale 3D voxel CNN features and the PointNet-based features to a small set of keypoints by the new proposed voxel set abstraction layer, and the learned discriminative features of keypoints are then aggregated to the RoI-grid points with multiple receptive fields to capture much richer context information for the fine-grained proposal refinement. Experimental results on the KITTI dataset and the Waymo Open dataset demonstrate that our proposed voxel-to-keypoint scene encoding and keypoint-to-grid RoI feature abstraction strategy significantly improve the 3D object detection performance compared with previous state-of-the-art methods.

Acknowledgments This work is supported in part by SenseTime Group Limited, in part by the General Research Fund through the Research Grants Council of Hong Kong under Grants CUHK14202217, CUHK14203118, CUHK14205615, CUHK14207814, CUHK14213616, CUHK14208417, CUHK14239816, in part by Research Impact Fund R5001-18, and in part by CUHK Direct Grant.

References

[1] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[2] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Fast point r-cnn. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, 2019.

[3] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019.

[4] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

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

[6] Benjamin Graham and Laurens van der Maaten. Submanifold sparse convolutional networks. *CoRR*, abs/1706.01307, 2017.

[7] Qiangui Huang, Weiyue Wang, and Ulrich Neumann. Recurrent slice networks for 3d segmentation of point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2626–2635, 2018.

[8] Maximilian Jaritz, Jiayuan Gu, and Hao Su. Multi-view pointnet for 3d scene understanding. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[9] Borui Jiang, Ruixuan Luo, Jiayuan Mao, Tete Xiao, and Yuning Jiang. Acquisition of localization confidence for accurate object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 784–799, 2018.

[10] Li Jiang, Hengshuang Zhao, Shu Liu, Xiaoyong Shen, Chi-Wing Fu, and Jiaya Jia. Hierarchical point-edge interaction network for point cloud semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 10433–10441, 2019.

[11] KITTI leader board of 3D object detection benchmark. `http://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=3d`, Accessed on 2019-11-15.

[12] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven Waslander. Joint 3d proposal generation and object detection from view aggregation. *IROS*, 2018.

[13] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. *CVPR*, 2019.

[14] Johannes Lehner, Andreas Mitterecker, Thomas Adler, Markus Hofmarcher, Bernhard Nessler, and Sepp Hochreiter. Patch refinement - localized 3d object detection. *CoRR*, abs/1910.04093, 2019.

[15] Buyu Li, Wanli Ouyang, Lu Sheng, Xingyu Zeng, and Xiaogang Wang. Gs3d: An efficient 3d object detection framework for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1019–1028, 2019.

[16] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *Advances in Neural Information Processing Systems*, pages 820–830, 2018.

[17] Ming Liang*, Bin Yang*, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *CVPR*, 2019.

[18] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *ECCV*, 2018.

[19] Tsung-Yi Lin, Priyal Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[20] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel CNN for efficient 3d deep learning. *CoRR*, abs/1907.03739, 2019.

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

[22] Jiquan Ngiam, Benjamin Caine, Wei Han, Brandon Yang, Yuning Chai, Pei Sun, Yin Zhou, Xi Yi, Ouais Alsharif, Patrick Nguyen, Zhifeng Chen, Jonathon Shlens, and Vijay Vasudevan. Starnet: Targeted computation for object detection in point clouds. *CoRR*, abs/1908.11069, 2019.

[23] Charles R. Qi, Or Litany, Kaiming He, and Leonidas J. Guibas. Deep hough voting for 3d object detection in point clouds. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[25] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas 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*, pages 652–660, 2017.

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

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

[28] 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.

[29] Shuran Song and Jianxiong Xiao. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 808–816, 2016.

[30] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2530–2539, 2018.

[31] 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. *arXiv*, pages arXiv–1912, 2019.

[32] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, Francois Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[33] Bei Wang, Jianping An, and Jiayan Cao. Voxel-fpn: multi-scale voxel feature aggregation in 3d object detection from point clouds. *CoRR*, abs/1907.05286, 2019.

[34] 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):146, 2019.

[35] Zhixin Wang and Kui Jia. Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3d object detection. In *IROS*. IEEE, 2019.

[36] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9621–9630, 2019.

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

[38] Bin Yang, Ming Liang, and Raquel Urtasun. Hdnet: Exploiting hd maps for 3d object detection. In *2nd Conference on Robot Learning (CoRL)*, 2018.

[39] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7652–7660, 2018.

[40] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. STD: sparse-to-dense 3d object detector for point cloud. *ICCV*, 2019.

[41] Hongwei Yi, Shaoshuai Shi, Mingyu Ding, Jiankai Sun, Kui Xu, Hui Zhou, Zhe Wang, Sheng Li, and Guoping Wang. Segvoxelnet: Exploring semantic context and depth-aware features for 3d vehicle detection from point cloud. In *IEEE International Conference on Robotics and Automation*, 2020.

[42] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5565–5573, 2019.

[43] Dingfu Zhou, Jin Fang, Xibin Song, Chenye Guan, Junbo Yin, Yuchao Dai, and Ruigang Yang. Iou loss for 2d/3d object detection. In *International Conference on 3D Vision (3DV)*. IEEE, 2019.

[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. *CoRR*, abs/1910.06528, 2019.

[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.

[46] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced grouping and sampling for point cloud 3d object detection. *CoRR*, abs/1908.09492, 2019.