Not All Points Are Equal: Learning Highly Efficient Point-based Detectors for 3D LiDAR Point Clouds

Yifan Zhang[1], Qingyong Hu[2*], Guoquan Xu[1], Yanxin Ma[1], Jianwei Wan[1], Yulan Guo[1]
[1]National University of Defense Technology, [2]University of Oxford
{zhangyifan16c, xuguoquan19, mayanxin, wanjianwei, yulan.guo}@nudt.edu.cn, qingyong.hu@cs.ox.ac.uk

Abstract

We study the problem of efficient object detection of 3D LiDAR point clouds. To reduce the memory and computational cost, existing point-based pipelines usually adopt task-agnostic random sampling or farthest point sampling to progressively downsample input point clouds, despite the fact that not all points are equally important to the task of object detection. In particular, the foreground points are inherently more important than background points for object detectors. Motivated by this, we propose a highly-efficient single-stage point-based 3D detector in this paper, termed ***IA-SSD****. The key of our approach is to exploit two learnable, task-oriented, instance-aware downsampling strategies to hierarchically select the foreground points belonging to objects of interest. Additionally, we also introduce a contextual centroid perception module to further estimate precise instance centers. Finally, we build our IA-SSD following the encoder-only architecture for efficiency. Extensive experiments conducted on several large-scale detection benchmarks demonstrate the competitive performance of our IA-SSD. Thanks to the low memory footprint and a high degree of parallelism, it achieves a superior speed of 80+ frames-per-second on the KITTI dataset with a single RTX2080Ti GPU. The code is available at* https://github.com/yifanzhang713/IA-SSD.

1. Introduction

Accurate recognition and localization of specific 3D objects is a fundamental research problem in 3D computer vision [10]. As a commonly-used 3D representation, point cloud has attracted increasing attention for its flexibility and compactness. However, the task of 3D object detection in LiDAR point clouds (*i.e.*, predicting 3D bounding boxes with 7 degrees-of-free including 3D-location, 3D-size, orientation, and class labels) remains highly challenging due to the complex geometrical structure and non-uniform density.

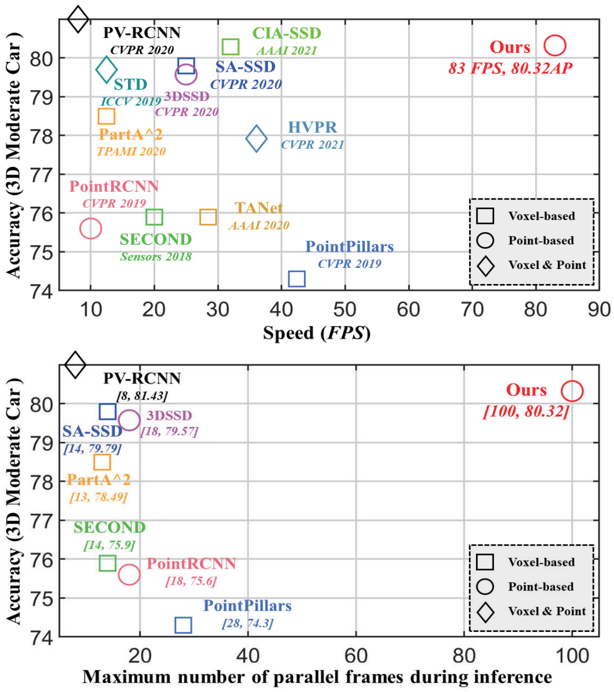

Figure 1. Comparison of the detection performance (accuracy) and efficiency (computational and memory) of different methods in KITTI benchmark. All experiments are conducted on a single RTX2080Ti GPU. Note that, we evaluate the memory efficiency by calculating the maximum number of parallel frames during inference when fully utilizing the GPU memory. Additionally, the FPS is calculated with the full utilization of GPU memory, more detailed analysis could be found in Table 6.

Due to the unstructured and orderless nature of 3D point clouds, early works usually first convert the raw point clouds into intermediate regular representation, including projecting the 3D point clouds into 2D images from birds-eye-view or frontal view [1, 17, 18, 41, 42, 50, 60], or transformed into dense 3D voxels [49, 61]. Then, several well-developed 2D detection paradigms can be deployed into the task of 3D object detection. Although remarkable progress has been achieved recently [2, 3, 7, 11, 24, 39, 53, 54], these

[*] Corresponding author

methods introduce the quantization error due to the 3D-2D projection or voxelization, which inevitably limits their performance of existing methods. Another stream of techniques following the point-based pipeline to directly operate on raw point clouds [12, 13, 15, 38, 40, 48, 52]. They usually learn point-wise features and then aggregate through specific symmetric functions such as max-pooling [31, 32]. Although promising and without any explicit information loss, these methods still suffer from expensive computational/memory costs and limited detection performance.

In this paper, we first dive deep into the existing point-based frameworks and experimentally find that the heuristic sampling strategies used are far from satisfactory, since a number of the important foreground points have been dropped before the final bounding box regression step. As such, the detection performance, especially for small objects such as pedestrians, has been fundamentally limited. In this paper, we argue that *not all points are equally important to the task of object detection.* In particular, only the foreground points, are the things we really care about.

Motivated by this, we aim to propose a task-oriented, instance-aware downsampling framework, to explicitly preserve foreground points while reducing the memory/computational cost. Specifically, two variants, namely *class-aware* and *centroid-aware* sampling strategies are proposed. In addition, we also present a contextual instance centroid perception, to fully exploit the meaningful context information around bounding boxes for instance center regression. Finally, we build our IA-SSD based on the bottom-up single-stage framework. As shown in Figure 1, the proposed IA-SSD demonstrated to be highly efficient (up to inference 100 frames in parallel in a single pass, with a speed of 83 FPS on a single RTX 2080Ti GPU) and accurate on the KITTI benchmark [8]. In particular, thanks to the high instance recall ratio of the proposed sampling strategy, the proposed IA-SSD can be directly trained with multiple object categories, rather than the common practice, *i.e.,* train separate models for different categories. Extensive experiments on Section 4 justify the compelling performance and superior efficiency of our method.

To summarize, the contributions are listed as follows:

- We identify the sampling issue in existing point-based detectors, and proposed an efficient point-based 3D detector by introducing two learning-based instance-aware downsampling strategies.
- The proposed IA-SSD is highly efficient and capable of detecting multi-class objects on LiDAR point clouds in a single pass. We also provided a detailed memory footprint vs. inference-speed analysis to further validate the superiority of the proposed method.
- Extensive experiments on several large-scale datasets demonstrate the superior efficiency and accurate detection performance of the proposed method.

2. Related Work

Here, we give a brief overview of existing voxel-based detectors, point-based detectors, and point-voxel detectors.

2.1. Voxel-based Detectors

To process unstructured 3D point clouds, voxel-based detectors usually first convert the irregular point clouds into regular voxel grids. This further allows leveraging the mature convolution neural architectures. Early works such as [46] densely voxelized the input point clouds and then utilized convolutional neural networks to learn specific geometrical patterns. However, efficiency is one of the main limitations of these methods, since the computational and memory cost grow cubically with the input resolution. To this end, Yan et al. [49] present an efficient architecture called SECOND by leveraging the 3D submanifold sparse convolution [9]. By reducing the calculation on empty voxels, the computational and memory efficiency have been significantly improved. Further, PointPillars [18] is proposed to further simplify the voxels to pillars (*i.e.,* only voxelization in the plane).

The existing approaches can be roughly divided into single-stage [7, 11, 54, 55, 57, 58] and two-stage detectors [4, 36–39, 53]. Albeit simple and efficient, they usually failed to achieve satisfactory detection performance due to the downscaled spatial resolution and insufficient structural information, especially for small objects with sparse points. To this end, He et al. [11] present SA-SSD to leverage the structure information by introducing an auxiliary network. Ye et al. [54] introduce a Hybrid Voxel network (HVNet) to attentively aggregate and project the multi-scale feature maps to achieve better performance. Zheng et al. [58] present the Confident IoU-Aware (CIA-SSD) network to extract spatial-semantic features for object detection. In comparison, two-stage detectors can achieve better performance, but with high computational/memory cost. Shi et al. [39] propose a two-stage detector namely Part-A^2, which is composed of the part-aware and aggregation module to exploit the intra-object part locations. Deng et al. [5] extend the PV-RCNN [36] by introducing a fully convolutional network to further exploit volumetric representation for raw point cloud and refinement simultaneously.

Overall, voxel-based methods can achieve good detection performance with promising efficiency. However, voxelization inevitably introduces quantization loss. In order to compensate for the structural distortion in the pre-processing phase, complex module design needs to be introduced in [20, 25, 27, 28, 35], which in turn greatly deteriorate the final detection efficiency. Additionally, it is not easy to determine the optimal resolution in practice, considering the complex geometry and various different objects.

2.2. Point-based Detectors

Different from voxel-based methods, point-based methods [30, 38, 52] directly learning geometry from unstructured point clouds, further generate specific proposals for objects of interest. Considering the orderless nature of 3D point clouds, these methods typically adopt PointNet [31] and its variants [22, 32, 33, 45, 47] to aggregate independent point-wise features using symmetric functions. Shi et al. [38] propose PointRCNN, a two-stage 3D region proposal framework for 3D object detection. This method first generates object proposals from segmented foreground points, and high-quality 3D bounding boxes are then regressed by exploiting the semantic feature and local spatial cues. Qi et al. [30] introduce VoteNet, a one-stage point-based 3D detector based on deep Hough voting to predict the instance centroid. Inspired by single-stage detectors [21] in 2D images, Yang et al. [52] presents a 3D Single-Stage Detection (3DSSD) framework, while the key is a fusion sampling strategy comprising the Farthest Point Sampling on feature and Euclidean space. PointGNN [40] is a framework by generalizing graph neural network to 3D object detection.

Point-based methods directly operate on the raw point clouds, without any extra preprocessing steps such as voxelization, hence usually intuitive and straightforward. However, the main bottleneck of point-based methods is insufficient learning capacity and limited efficiency.

2.3. Point-Voxel Methods

To overcome the drawbacks of both point-based methods (*i.e.*, irregular and sparse data access, poor memory locality [23]) and voxel-based methods (*i.e.*, quantization loss), several methods [3, 16, 36, 37, 53] have started to learning from 3D point clouds using point-voxel joint representations. Specifically, PV-RCNN [36] and its follow-up work [37] extract point-wise features from voxel abstraction networks to refine the proposals generated from 3D voxel backbone. Further, HVPR [29], a single-stage 3D detector, introduces an efficient memory module to augment point-based features, thereby providing a better compromise between accuracy and efficiency. Qian et al. [34] propose a lightweight region aggregation refine network (BANet) via local neighborhood graph construction, which produces more accurate box boundary prediction.

Overall, different detection pipelines have their own merits. In this paper, we propose IA-SSD, a single-stage point-based detector, to simultaneously improve the detection accuracy and runtime efficiency. In particular, the key differences between our IA-SSD and existing point-based techniques lie in the instance-aware sampling strategies and the contextual instance centroid perception module, as illustrated in the following sections.

3. The Proposed IA-SSD

3.1. Overview

Different from dense prediction tasks such as 3D semantic segmentation, where point-wise prediction is required, *3D object detection naturally focus on the small yet important foreground objects* (*i.e.*, instances of interest including *car*, *pedestrian*, *etc*.). However, existing point-based detectors usually adopt task-agnostic downsampling approaches such as random sampling [14] or farthest point sampling [32, 52] in their framework. Albeit effective for memory/computational cost reduction, the most important foreground points are also diminished in progressive downsampling. Additionally, due to the large difference in size and geometrical shape of different objects, existing detectors usually train separate models with various carefully tuned hyperparameters for different types of objects. However, this inevitably affects the deployment of these models in practice. Therefore, the objective of this paper is: *Can we train a single point-based model, which is efficient and capable of detecting multi-class objects in a single pass?*

Motivated by this, we propose an efficient, single-stage detector by introducing the instance-aware downsampling and contextual centroid perception module. As shown in Figure 2, our IA-SSD follows the lightweight encoder-only architecture used in [52] for efficiency. The input LiDAR point clouds are first fed into the network to extract point-wise features, followed by the proposed instance-aware downsampling to progressively reduce the computational cost, while preserving the informative foreground points simultaneously. The learned latent features are further input to the contextual centroid perception module to generate instance proposals and regress the final bounding boxes.

3.2. Instance-aware Downsampling Strategy

For efficient 3D object detection, it is essential to reduce the memory and computational cost through progressive downsampling, especially for large-scale 3D point clouds. However, aggressive downsampling may lose most of the information of the foreground objects. Overall, it remains unclear how to achieve a desirable trade-off between computational efficiency and the preservation of foreground points. To this end, we first conduct an empirical study to quantitatively evaluate different sampling approaches. In particular, we follow the commonly-used encoding architecture (*i.e.,* PointNet++ [32] with 4 encoding layers), and report the Instance Recall (*i.e.,* the ratio of instance retained after sampling) at each layer in Table 1. In particular, random point sampling [14], FPS based on Euclidean distance (D-FPS) [32] and feature distance (Feat-FPS) [52] are reported.

Analysis. It can be seen that: 1) The instance recall rate dropped significantly after several random downsampling

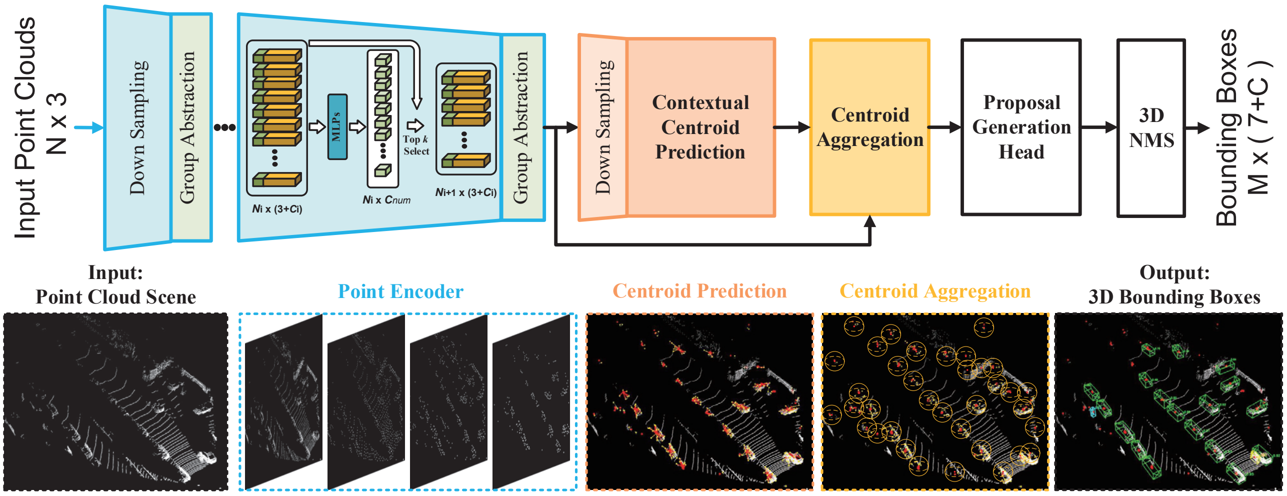

Figure 2. Illustration of the proposed IA-SSD. The input point clouds are first fed into several Set Abstraction (SA) layers, followed by the instance-aware downsampling to progressively reduce the memory and computational cost. The preserved representative points are further fed into the contextual centroid perception module for instance center prediction and proposal generation. Finally, the 3D bounding box and associated class labels are outputted.

operations, indicating massive foreground points have been dropped. 2) Both D-FPS and Feat-FPS achieve a relatively better instance recall rate at the early stage, but also fail to preserve enough foreground points at the last encoding layer. As such, it remains challenging to precisely detect the objects of interest, especially for small objects such as pedestrians and cyclists, where only extremely limited foreground points are left.

Solutions. To preserve foreground points as much as possible, we turn to leverage the latent semantics of each point, since the learned point features may incorporate richer semantic information as the hierarchical aggregation operates in each layer. Following this idea, we propose the following two task-oriented sampling approaches by incorporating the foreground semantic priors into the network training pipelines.

***Class-aware Sampling*.** This sampling strategy aims to learn semantics for each point, so as to achieve selective downsampling. To achieve this, we introduce extra branches to exploit the rich semantics in latent features. In particular, two MLP layers were appended to the encoding layers to further estimate the semantic categories of each point. The point-wise one-hot semantic labels generated from the original bounding box annotations are used for supervision. Here we use the vanilla cross-entropy loss:

$$L_{cls\text{-}aware} = -\sum_{c=1}^{C}(s_i log(\hat{s}_i) + (1-s_i) log(1-\hat{s}_i)) \qquad (1)$$

where C denotes the number of categories, s_i is the one-hot labels and $\hat{s}_i$ denotes the predicted logits. During inference, the points with the top k foreground scores are retained and regarded as the representative points that feed into the next encoding layers. As shown in Table 1, this strategy tends to preserve more foreground points, hence achieving a high ratio of instance recall.

***Centroid-aware Sampling*.** Considering instance center estimation is the key for final object detection, we further propose a centroid-aware downsampling strategy to give higher weight to points closer to instance centroid. Specifically, we define the soft point mask of instance i as follows:

$$Mask_i = \sqrt[3]{\frac{min(f^*, b^*)}{max(f^*, b^*)} \times \frac{min(l^*, r^*)}{max(l^*, r^*)} \times \frac{min(u^*, d^*)}{max(u^*, d^*)}} \qquad (2)$$

where $f^*, b^*, l^*, r^*, u^*, d^*$ represent the distance of a point to the 6 surfaces (front, back, left, right, up and down) of the bounding box, respectively. In this case, the point closer to the centroid of the box is likely to have a higher mask score (max value is 1), while the point that lies on the surface will have a mask score of 0. During training, the soft point mask will be used to assign different weights for points within a bounding box based on the spatial locations, hence implicitly incorporates the geometry priors into the network training. In particular, the weighted cross-entropy loss is calculated as follows:

$$L_{ctr\text{-}aware} = -\sum_{c=1}^{C}(Mask_i \cdot s_i log(\hat{s}_i) + (1-s_i) log(1-\hat{s}_i)) \qquad (3)$$

The soft point mask is multiplied with the loss term of foreground points, so as to assign a higher probability to the points near the center. Note that, the bounding boxes are no longer required during inference, we simply preserve the top k points with the highest scores after downsampling, if the model is well-trained.

Sampling strategies	4096 points			1024 points			512 points			256 points		
	Car	*Ped.*	*Cyc.*	*Car*	*Ped.*	*Cyc.*	*Car*	*Ped.*	*Cyc.*	*Car*	*Ped.*	*Cyc.*
Random [14]	96.6%	99.1%	97.4%	87.5%	92.7%	84.1%	78.8%	84.9%	73.3%	67.4%	72.1%	57.3%
D-FPS [32]	98.3%	100%	97.2%	97.9%	99.3%	97.2%	96.8%	90.6%	90.8%	91.4%	69.1%	71.6%
Feat-FPS [52]	98.3%	100%	97.2%	97.7%	98.0%	97.2%	96.3%	87.6%	94.5%	95.3%	80.1%	91.7%
Cls-aware (Ours)	98.3%	100%	97.2%	97.9%	99.3%	97.2%	97.9%	99.0%	95.4%	97.9%	97.4%	92.7%
Ctr-aware (Ours)	98.3%	100%	97.2%	97.9%	99.3%	97.2%	97.9%	99.0%	97.2%	97.9%	98.4%	97.2%

Table 1. The instance recall rate for foreground points (*i.e.*, *car*, *pedestrian*, and *cyclist*) after several downsampling on the entire *validation* set (3769 frames) of the KITTI benchmark. Note that, the input point clouds with 16384 points are progressively downsampled to 256 points through four downsampling layers. D-FPS are used in the first two layers for the proposed instance aware downsampling strategies.

3.3. Contextual Instance Centroid Perception

Contextual Centroid Prediction. Inspired by the success of context prediction in 2D images [6, 51], we attempt to leverage the contextual cues around the bounding box for instance centroid prediction. Specifically, we follow [30] to explicitly predict an offset $\Delta\hat{c_{ij}}$ to the instance center. Additionally, a regularization term is added to minimize the uncertainty of the centroid prediction. Specifically, all votes per instance are aggregated in light of the surrounding interference, where the $\overline{c_i}$ is the mean destination of i-th instance. Therefore, the centroid prediction loss is formulated as follows:

$$\begin{aligned} L_{cent} &= \frac{1}{|\mathcal{F}_+|}\frac{1}{|\mathcal{S}_+|}\sum_i\sum_j(|\Delta\hat{c_{ij}} - \Delta c_{ij}| + |\hat{c_{ij}} - \overline{c_i}|)\cdot \mathbf{I}_{\mathcal{S}}(p_{ij}) \\ where \quad \overline{c_i} &= \frac{1}{|\mathcal{S}_+|}\sum_j \hat{c_{ij}}, \quad \mathbf{I}_{\mathcal{S}}:\mathcal{P}\rightarrow\{0,1\} \end{aligned} \tag{4}$$

where Δc_{ij} denotes the ground-truth offset from point p_{ij} to the center point. $\mathbf{I}_{\mathcal{S}}$ is an indicator function to determine whether this point is used to estimate the instance center or not. $|S_+|$ is the number of points used to predict the instance center. Note that, instead of only using the points or the shifted points within the bounding box for instance center prediction [30, 52], we also exploit the surrounding representative points from a large context for centroid prediction in this paper. Specifically, we empirically investigate the impact of simple contextual cues on final detection performance. In particular, we manually expand the ground-truth bounding boxes or proportional enlarge the box to cover more related context near the objects. The sampled points that fall in the expanded bounding box are utilized to estimate offset and then shifted.

Centroid-based Instance Aggregation. For shifted representative (centroid) points, we further utilize a PointNet++ module to learn a latent representation for each instance. Specifically, we transform the neighboring points to a local canonical coordinate system, then aggregate the point feature through shared MLPs and symmetric functions.

Proposal Generation Head. The aggregated centroid point features are then fed into proposal generation head to predict bounding boxes with classes. We encode the proposal as a multidimensional representation with location, scale, and orientation. Finally, all proposals are filtered by 3D-NMS post-processing with a specific IoU threshold.

3.4. End-to-End Learning

Our IA-SSD can be trained in an end-to-end fashion. Multi-task loss is used in our framework for joint optimization. The total loss L_{total} is composed of downsampling strategy loss L_{sample}, centroid prediction loss L_{cent}, classification loss L_{cls} and box generation loss L_{box}:

$$L_{total} = L_{sample} + L_{cent} + L_{cls} + L_{box} \tag{5}$$

In particular, the box generation loss can be further decomposed into location, size, angle-bin, angle-res, and corner parts:

$$L_{box} = L_{loc} + L_{size} + L_{angle\text{-}bin} + L_{angle\text{-}res} + L_{corner} \tag{6}$$

4. Experiments

4.1. Implementation Details

We build our IA-SSD based on single-stage, encoder-only architecture for efficiency. Specifically, a number of SA layers [32] are used to extract point-wise features. Multi-scale grouping with increasing radius groups is used ([0.2, 0.8], [0.8, 1.6], [1.6, 4.8]) to steady extract local geometrical features. Considering limited semantics incorporated in early layers, we adopt D-FPS in the first two encoding layers, followed by the proposed instance-aware downsampling. Next, 256 representative point features are fed into the contextual centroid prediction module, followed by three MLP layers (256→256→3) to predict the instance centroid. Finally, the classification and regression layers (three MLP layers) are appended to output the semantic labels and the corresponding bounding boxes. More implementation details are reported in the Appendix.

	Method	Reference	Type	3D Car (IoU=0.7)			3D Ped. (IoU=0.5)			3D Cyc. (IoU=0.5)			Speed
				Easy	*Mod.*	*Hard*	*Easy*	*Mod.*	*Hard*	*Easy*	*Mod.*	*Hard*	
Voxel-based	VoxelNet [61]	CVPR 2018	1-stage	77.47	65.11	57.73	39.48	33.69	31.5	61.22	48.36	44.37	4.5
	SECOND [49]	Sensors 2018	1-stage	84.65	75.96	68.71	45.31	35.52	33.14	75.83	60.82	53.67	20
	PointPillars [18]	CVPR 2019	1-stage	82.58	74.31	68.99	51.45	41.92	38.89	77.10	58.65	51.92	42.4
	3D IoU Loss [59]	3DV 2019	1-stage	86.16	76.50	71.39	-	-	-	-	-	-	12.5
	Associate-3Ddet [7]	CVPR 2020	1-stage	85.99	77.40	70.53	-	-	-	-	-	-	20
	SA-SSD [11]	CVPR 2020	1-stage	88.75	79.79	74.16	-	-	-	-	-	-	25
	CIA-SSD [58]	AAAI 2021	1-stage	89.59	80.28	72.87	-	-	-	-	-	-	32
	TANet [24]	AAAI 2020	2-stage	84.39	75.94	68.82	53.72	44.34	40.49	75.70	59.44	52.53	28.5
	Part-A^2 [39]	TPAMI 2020	2-stage	87.81	78.49	73.51	53.10	43.35	40.06	79.17	63.52	56.93	12.5
Point-Voxel	Fast Point R-CNN [3]	ICCV 2019	2-stage	85.29	77.40	70.24	-	-	-	-	-	-	16.7
	STD [53]	ICCV 2019	2-stage	87.95	79.71	75.09	53.29	42.47	38.35	78.69	61.59	55.30	12.5
	PV-RCNN [36]	CVPR 2020	2-stage	90.25	81.43	76.82	52.17	43.29	40.29	78.60	63.71	57.65	12.5
	VIC-Net [16]	ICRA 2021	1-stage	88.25	80.61	75.83	43.82	37.18	35.35	78.29	63.65	57.27	17
	HVPR [29]	CVPR 2021	1-stage	86.38	77.92	73.04	53.47	43.96	40.64	-	-	-	36.1
Point-based	PointRCNN [38]	CVPR 2019	2-stage	86.96	75.64	70.70	47.98	39.37	36.01	74.96	58.82	52.53	10
	3D IoU-Net [19]	Arxiv 2020	2-stage	87.96	79.03	72.78	-	-	-	-	-	-	10
	Point-GNN [40]	CVPR 2020	1-stage	88.33	79.47	72.29	51.92	43.77	40.14	78.60	63.48	57.08	1.6
	3DSSD [52]	CVPR 2020	1-stage	88.36	79.57	74.55	54.64	44.27	40.23	82.48	64.10	56.90	25
	3DSSD†(Reproduced)	CVPR 2020	1-stage	87.73	78.58	72.01	35.03	27.76	26.08	66.69	59.00	55.62	23
	3DSSD‡(OpenPCDet)	CVPR 2020	1-stage	87.91	79.55	74.71	3.63	3.18	2.57	27.08	21.38	19.68	28
	IA-SSD (single-class)	**-**	**1-stage**	**88.87**	**80.32**	**75.10**	**49.01**	**41.20**	**38.03**	**80.78**	**66.01**	**58.12**	**85**
	IA-SSD (multi-class)	**-**	**1-stage**	**88.34**	**80.13**	**75.04**	**46.51**	**39.03**	**35.60**	**78.35**	**61.94**	**55.70**	**83**

Table 2. Quantitative detection performance achieved by different methods on the KITTI *test* set. All results are evaluated by mean Average Precision with 40 recall positions via the official KITTI evaluation server. The results of our IA-SSD are shown in bold, and the best results are underlined.

	Method	References	Type	Car Mod (IoU=0.7)	Ped. Mod (IoU=0.5)	Cyc. Mod (IoU=0.5)
Voxel-based	VoxelNet [61]	CVPR 2018	1-stage	65.46	53.42	47.65
	SECOND [49]	Sensors 2018	1-stage	76.48	-	-
	PointPillars [18]	CVPR 2019	1-stage	77.98	-	-
	TANet [24]	AAAI 2020	1-stage	77.85	63.45	64.95
	Associate-3Ddet [7]	CVPR 2020	1-stage	79.17	-	-
	SA-SSD [11]	CVPR 2020	1-stage	79.91	-	-
	CIA-SSD [58]	AAAI 2021	1-stage	79.81	-	-
	Part-A^2 [39]	TPAMI 2020	2-stage	79.47	63.84	73.07
Point-Voxel	Fast Point R-CNN [3]	ICCV 2019	2-stage	79.00	-	-
	STD [53]	ICCV 2019	2-stage	79.8	-	-
	PV-RCNN [36]	CVPR 2020	2-stage	83.90	-	-
	VIC-Net [16]	ICRA 2021	1-stage	79.25	-	-
Point-based	PointRCNN [38]	CVPR 2019	2-stage	78.63	-	-
	3D IoU-Net [19]	Arxiv 2020	2-stage	79.26	-	-
	PointGNN [40]	CVPR 2020	1-stage	78.34	-	-
	3DSSD [52]	CVPR 2020	1-stage	79.45	-	-
	IA-SSD (Ours)	**-**	**1-stage**	**79.57**	**58.91**	**71.24**

Table 3. Quantitative comparison of different approaches on the *validation* split of the KITTI dataset. The average precision is measured with 11 recall positions (vs. 40 recall positions in the KITTI *test* set) [8]. The results achieved by our IA-SSD are shown in bold, while the top-performed results are shown in underline.

4.2. Comparison with State-of-the-Art Methods

Evaluation on KITTI Dataset. In the KITTI benchmark, objects belong to *car*, *pedestrian* and *cyclist* are classified into three subsets ("Easy", "Moderate" and "Hard") based on the levels of difficulty. The results on "Moderate" are usually adopted as the main indicator for final ranking. We report the results achieved by different methods (voxel, point, and point-voxel-based methods) on the test set of the KITTI dataset in Table 2. Note that, since [52] does not provide reproducible implementation or pre-trained models for *pedestrian* and *cyclist*, we have no choice but to provide both the results reported in their paper, the best-reproduced results, and the results achieved by OpenPCDet[1] implementation for a fair comparison.

Analysis. It can be seen that: 1) the proposed IA-SSD achieves the best *cyclist* detection performance, even outperforming several strong point-voxel and voxel detectors [36, 39]. This is mainly because the proposed instance aware sampling can effectively preserve foreground points, enabling accurate detection of small objects. 2) Our IA-SSD also achieves best *car* detection performance compared with other point-based detectors, outperforming PointRCNN [38] by (1.91%, 4.68%, 4.4%), and the SoTA method 3DSSD [52] by (0.51%, 0.75%, 0.55%) mAP. 3) Despite the competitive detection performance, the proposed IA-SSD also shows superior efficiency. It can detect with a speed of 85 FPS on a single NVIDIA RTX 2080Ti with Intel I9-10900X CPU@3.7GHz. 4) Thanks to the instance-aware sampling strategy and the contextual centroid perception module, our framework can be trained with multi-class together (*i.e.*, training a single model for detecting multi-class objects), rather than training separate models for different objects [52]. In particular, the performance is still comparable with other state-of-the-art approaches. This allows our model much more efficient and flexible during inference. Finally, we also show the qualitative results achieved by our IA-SSD in Figure 3. We can clearly see

[1] https://github.com/open-mmlab/OpenPCDet

Method	Type	Vehicle (LEVEL 1)		Vehicle (LEVEL 2)		Ped. (LEVEL 1)		Ped. (LEVEL 2)		Cyc. (LEVEL 1)		Cyc. (LEVEL 2)	
		mAP	mAPH	mAP	mAPH	mAP	mAPH	mAP	mAPH	mAP	mAPH	mAP	mAPH
PointPillars [18]	Voxel-based	60.67	59.79	52.78	52.01	43.49	23.51	37.32	20.17	35.94	28.34	34.60	27.29
SECOND [49]	Voxel-based	68.03	67.44	59.57	59.04	61.14	50.33	53.00	43.56	54.66	53.31	52.67	51.37
Part-A^2 [39]	Voxel-based	71.82	71.29	64.33	63.82	63.15	54.96	54.24	47.11	65.23	63.92	62.61	61.35
PV-RCNN [36]	Point-Voxel	74.06	73.38	64.99	64.38	62.66	52.68	53.80	45.14	63.32	61.71	60.72	59.18
IA-SSD (Ours)	Point-based	**70.53**	**69.67**	**61.55**	**60.80**	**69.38**	**58.47**	**60.30**	**50.73**	**67.67**	**65.30**	**64.98**	**62.71**

Table 4. Quantitative detection performance achieved by different methods on the Waymo [44] *validation* set. The results of our IA-SSD are shown in bold, and the best results are underlined.

Method	Type	Vehicle				Pedestrian				Cyclist				mAP
		overall	0-30m	30-50m	>50m	overall	0-30m	30-50m	>50m	overall	0-30m	30-50m	>50m	
PointPillars [18]	Voxel-based	68.57	80.86	62.07	47.04	17.63	19.74	15.15	10.23	46.81	58.33	40.32	25.86	44.34
SECOND [49]	Voxel-based	71.19	84.04	63.02	47.25	26.44	29.33	24.05	18.05	58.04	69.96	52.43	34.61	51.89
CenterPoints [56]	Voxel-based	66.79	80.10	59.55	43.39	49.90	56.24	42.61	26.27	63.45	74.28	57.94	41.48	60.05
PV-RCNN [36]	Point-Voxel	77.77	89.39	72.55	58.64	23.50	25.61	22.84	17.27	59.37	71.66	52.58	36.17	53.55
PointRCNN [38]	Point-based	52.09	74.45	40.89	16.81	4.28	6.17	2.40	0.91	29.84	46.03	20.94	5.46	28.74
IA-SSD (Ours)	Point-based	**70.30**	**83.01**	**62.84**	**47.01**	**39.82**	**47.45**	**32.75**	**18.99**	**62.17**	**73.78**	**56.31**	**39.53**	**57.43**

Table 5. Quantitative detection performance on the ONCE [26] *validation* set. The results of our IA-SSD are shown in bold, and the best results are underlined.

that the proposed IA-SSD is capable of detecting small and far-away instances such as *pedestrian* and *cyclist*.

Apart from the detection results on the *test* split, we also report the performance comparison on the *validation* set of the KITTI dataset in Table 3. We can see our IA-SSD achieved the best performance for all three classes among all point-based detectors. In particular, our IA-SSD is single-stage, lightweight, and efficient, requiring only a single model for detecting multi-class objects.

Evaluation on Waymo Dataset. We further evaluate the performance of our IA-SSD on Waymo [44] dataset. This dataset is composed of nearly 160k 360-degree LiDAR samples in the *training* set and 40k in the *validation* set with panoramic annotated objects. For a fair comparison, we adapt our framework on the Waymo Dataset by only changing the number of input points from 16384 to 65536, and increasing the sampling scale up to fourfold in each sampling layer, while remaining the rest unchanged. Additionally, all baselines are implemented based on the OpenPCDet codebase for a rigorous comparison. As shown in Table 4, our IA-SSD achieves significantly better detection performance on *pedestrian* and *cyclist* compared with other strong baselines, showing that the proposed instance-aware sampling can indeed improve the perception capacity of small objects. We also noticed that our IA-SSD shows slightly inferior detection performance on *vehicle* compared with other voxel-based methods, this may be caused by the relatively complicated distribution of the 3D size of such instances. We will leave this issue for future exploration.

Evaluation on ONCE Dataset. To further verify the generalization of IA-SSD on more complex and realistic situations, we also evaluate the performance of our IA-SSD on the latest ONCE Dataset [26]. Specifically, we feed 60k points into the IA-SSD similar to the setting on the Waymo Dataset, and train 80 epochs for a fair comparison. As shown in Table 5, our method yields the competitive performance among all baselines. This again verifies the superiority of the proposed component and the efficiency of our method applied on the large-scale complicated LiDAR scenarios.

Method	*Mem.*	*Paral.*	*Speed*$^{\perp}$	*Speed*$^{\top}$	*Input Scale*
PointPillars [18]	354 MB	28	48	58	2∼9k
SECOND [49]	710 MB	14	30	40	11∼17k
TANet [24]	3000 MB	3	20	28	<12k
3DSSD [52]	502 MB	19	11	28	16384
PointRCNN [38]	560 MB	18	10	14	16384
Part-A^2 [39]	702 MB	13	12	19	11∼17k
PV-RCNN [36]	1223 MB	8	8	10	11∼17k
IA-SSD (Ours)	**102 MB**	**100**	**23/48**†	**83**	16384

Table 6. Efficiency comparison of different methods on the KITTI *validation* set. Here, "Mem." and "Paral." denote the GPU memory footprint per frame during inference and the maximum number of batches that can be parallelized on one RTX2080Ti (11GB). "Speed$^{\perp}$", "Speed$^{\top}$" is inference speed when processing one frame or full-loaded GPU memory, † means dividing the scene into four parallel parts to speed up the first sampling layer. For a fair comparison, we also report each input scale of voxels/points according to their official setting.

Efficiency of IA-SSD. Next, we evaluate the computational and memory efficiency of the proposed IA-SSD. In light of the performance variations on different hardware configurations, we re-implemented several representative approaches and report the memory and speed on the same platform for a fair comparison. Note that, we report the memory consumption by feeding the same input point cloud with 16384 points following the OpenPCDet configuration. For speed evaluation, different models are inference with batch point clouds with the full utilization of the same hardware. As shown in Table 6, the proposed IA-SSD has the lowest GPU memory consumption (up to 100 frames in parallel) with the highest inference speed (83 FPS) compared with existing benchmark approaches.

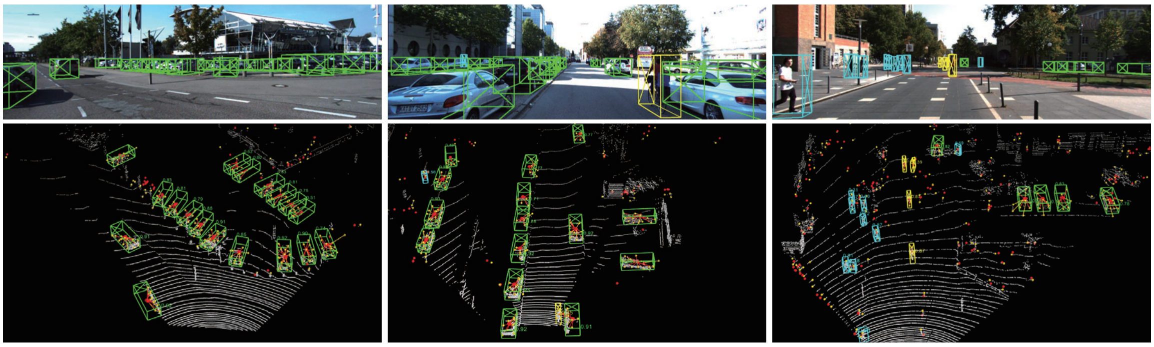

Figure 3. Qualitative results achieved on the KITTI *test* set. Red point for centroid perception, while gold points denote the 256 representative points. Green boxes for *car*, cyan for *pedestrian* and yellow for *cyclist*. Best viewed in color.

	D-FPS	Feat-FPS	Cls-aware	Ctr-aware	Car Mod (IoU=0.7)	Ped. Mod (IoU=0.5)	Cyc. Mod (IoU=0.5)
(1)	✓	-	-	-	78.12	50.46	65.19
(2)	✓	✓	-	-	79.00	54.31	71.08
(3)	✓	-	✓	-	79.19	58.81	70.15
(4)	✓	-	✓	✓	79.54	58.49	71.33
(5)	✓	-	-	✓	**79.57**	**58.91**	**71.24**

Table 7. Ablation study of IA-SSD on different sampling strategies, in which we report the 3D mAP with 11 recalls. Here D-FPS is the traditional Farthest Point Sampling, Feat-FPS represents the Feature-based FPS and Cls/Ctr-aware denote the proposed two learning based sampling methods.

4.3. Ablation Experiments

In the following ablation studies, we train our IA-SSD with multi-class objects in a single model, and all experiments are conducted on the KITTI validation set.

Ablation on Downsampling Strategies. To further verify the effectiveness of the proposed instance-aware sampling, we replace it with the D-FPS and Feat-FPS. As shown in Table 7, the proposed instance-aware sampling achieves the best detection performance in all three categories, especially the small objects such as *pedestrians* and *cyclists*. This shows that the proposed sampling strategy can effectively preserve the foreground information during downsampling process, thereby achieving better detection. We also find that centroid-aware sampling (row 5) performs better on *pedestrians* and *cars*, but slightly inferior in detect *cyclist* compared with mixed sampling (row 4), primarily because this sampling strategy focus on the instance center, hence tends to ignore the distal geometric details of objects such as *cyclist* with large aspect ratios.

Ablation on Contextual Centroid Perception. We further validate the effectiveness of the proposed contextual centroid perception module. Here, by replacing this module with the vanilla center-assign[2] or original-assign used in [30], the detection performance shows clearly decrease. Table 8 shows that the inclusion of contextual points can indeed improve the detection performance, especially for small objects, since the representative points lie in the ground-truth bounding boxes are actually quite limited. We are also noticed that both the extend-factor (2× size for each bounding box) and extend-length (+1.0m for each bounding box) have their own advantage in specific categories, showing that different contextual information may have a varying impact on different objects.

	Centroid Perception Type	Car Mod (IoU=0.7)	Ped. Mod (IoU=0.5)	Cyc. Mod (IoU=0.5)
(1)	Centers-assign	79.27	24.90	37.12
(2)	Origin-assign	79.18	55.62	69.37
(3)	Extend-factor assign	79.37	58.36	68.16
(4)	Extend-length assign	**79.57**	**58.91**	**71.24**

Table 8. Ablation study of IA-SSD framework with different centroid perception strategies.

5. Conclusion

In this paper, we propose an efficient solution termed IA-SSD for point-based 3D object detection in LiDAR point clouds. Considering the task of object detection inherently focuses on the foreground information, we propose an instance-aware learning-based downsampling way to automatically select the sparse yet important instance points. Additionally, a dedicated contextual centroid perception module is proposed to fully exploit the geometrical structure around the bounding boxes. Extensive experiments conducted on three detection benchmarks demonstrated the superior efficiency and accuracy of the proposed IA-SSD.

Limitations. Although the proposed IA-SSD can achieve remarkable efficiency in object detection of large-scale LiDAR points clouds, it also has limitations. *e.g.,* the instance-aware sampling relies on the semantic prediction of each point, which is susceptible to class imbalances distribution. For future work, we will further explore advanced techniques to alleviate the imbalanced issue.

Acknowledgements. This work was partially supported by the National Natural Science Foundation of China (No. 61972435, U20A20185), China Scholarship Council (CSC) scholarship, and Huawei UK AI Fellowship.

[2] https://github.com/qiqihaer/3DSSD-pytorch-openPCDet

References

[1] Waleed Ali, Sherif Abdelkarim, Mahmoud Zidan, Mohamed Zahran, and Ahmad El Sallab. Yolo3D: End-to-end real-time 3D oriented object bounding box detection from lidar point cloud. In ECCVW, pages 0–0, 2018. 1

[2] Sheng Ao, Qingyong Hu, Bo Yang, Andrew Markham, and Yulan Guo. Spinnet: Learning a general surface descriptor for 3d point cloud registration. In CVPR, pages 11753–11762, 2021. 1

[3] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Fast point r-cnn. In ICCV, pages 9775–9784, 2019. 1, 3, 6

[4] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. AAAI, 2021. 2

[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. In AAAI, 2021. 2

[6] Santosh K Divvala, Derek Hoiem, James H Hays, Alexei A Efros, and Martial Hebert. An empirical study of context in object detection. In CVPR, pages 1271–1278, 2009. 5

[7] Liang Du, Xiaoqing Ye, Xiao Tan, Jianfeng Feng, Zhenbo Xu, Errui Ding, and Shilei Wen. Associate-3Ddet: perceptual-to-conceptual association for 3D point cloud object detection. In CVPR, pages 13329–13338, 2020. 1, 2, 6

[8] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In CVPR, pages 3354–3361, 2012. 2, 6

[9] Benjamin Graham and Laurens van der Maaten. Submanifold sparse convolutional networks. arXiv preprint arXiv:1706.01307, 2017. 2

[10] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3D point clouds: A survey. IEEE TPAMI, 2020. 1

[11] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3D object detection from point cloud. In CVPR, pages 11873–11882, 2020. 1, 2, 6

[12] Qingyong Hu, Bo Yang, Guangchi Fang, Yulan Guo, Ales Leonardis, Niki Trigoni, and Andrew Markham. Sqn: Weakly-supervised semantic segmentation of large-scale 3d point clouds with 1000x fewer labels. arXiv preprint arXiv:2104.04891, 2021. 2

[13] Qingyong Hu, Bo Yang, Sheikh Khalid, Wen Xiao, Niki Trigoni, and Andrew Markham. Sensaturban: Learning semantics from urban-scale photogrammetric point clouds. International Journal of Computer Vision, pages 1–28, 2022. 2

[14] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In CVPR, pages 11108–11117, 2020. 3, 5

[15] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Learning semantic segmentation of large-scale point clouds with random sampling. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021. 2

[16] Tianyuan Jiang, Nan Song, Huanyu Liu, Ruihao Yin, Ye Gong, and Jian Yao. Vic-net: Voxelization information compensation network for pointcloud 3D object detection. In ICRA, 2021. 3, 6

[17] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3D proposal generation and object detection from view aggregation. In IROS, pages 1–8, 2018. 1

[18] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In CVPR, pages 12697–12705, 2019. 1, 2, 6, 7, 12

[19] Jiale Li, Shujie Luo, Ziqi Zhu, Hang Dai, Andrey S Krylov, Yong Ding, and Ling Shao. 3d iou-net: Iou guided 3D object detector for point clouds. arXiv preprint arXiv:2004.04962, 2020. 6

[20] Zhichao Li, Feng Wang, and Naiyan Wang. Lidar r-cnn: An efficient and universal 3d object detector. In CVPR, pages 7546–7555, 2021. 2

[21] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In ECCV, pages 21–37, 2016. 3

[22] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In CVPR, pages 8895–8904, 2019. 3

[23] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel cnn for efficient 3d deep learning. arXiv preprint arXiv:1907.03739, 2019. 3

[24] 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 AAAI, pages 11677–11684, 2020. 1, 6, 7

[25] Jiageng Mao, Minzhe Niu, Haoyue Bai, Xiaodan Liang, Hang Xu, and Chunjing Xu. Pyramid r-cnn: Towards better performance and adaptability for 3d object detection. In ICCV, pages 2723–2732, 2021. 2

[26] Jiageng Mao, Minzhe Niu, Chenhan Jiang, Xiaodan Liang, Yamin Li, Chaoqiang Ye, Wei Zhang, Zhenguo Li, Jie Yu, Chunjing Xu, et al. One million scenes for autonomous driving: Once dataset. 2021. 7, 12

[27] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. Voxel transformer for 3d object detection. In ICCV, pages 3164–3173, 2021. 2

[28] Zhenwei Miao, Jikai Chen, Hongyu Pan, Ruiwen Zhang, Kaixuan Liu, Peihan Hao, Jun Zhu, Yang Wang, and Xin Zhan. Pvgnet: A bottom-up one-stage 3d object detector with integrated multi-level features. In CVPR, pages 3279–3288, 2021. 2

[29] Jongyoun Noh, Sanghoon Lee, and Bumsub Ham. Hvpr: Hybrid voxel-point representation for single-stage 3D object detection. In CVPR, 2021. 3, 6

[30] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3D object detection in point clouds. In ICCV, pages 9277–9286, 2019. 3, 5, 8, 12

[31] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3D classification and segmentation. In CVPR, pages 652–660, 2017. 2, 3

[32] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In NeurIPS, pages 5099–5108, 2017. 2, 3, 5, 11

[33] Rui Qian, Divyansh Garg, Yan Wang, Yurong You, Serge Belongie, Bharath Hariharan, Mark Campbell, Kilian Q Weinberger, and Wei-Lun Chao. End-to-end pseudo-lidar for image-based 3D object detection. In CVPR, pages 5881–5890, 2020. 3

[34] Rui Qian, Xin Lai, and Xirong Li. Boundary-aware 3D object detection from point clouds. arXiv preprint arXiv:2104.10330, 2021. 3

[35] Hualian Sheng, Sijia Cai, Yuan Liu, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, and Min-Jian Zhao. Improving 3d object detection with channel-wise transformer. In ICCV, pages 2743–2752, 2021. 2

[36] 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 CVPR, pages 10529–10538, 2020. 2, 3, 6, 7, 12

[37] 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. 2, 3

[38] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3D object proposal generation and detection from point cloud. In CVPR, pages 770–779, 2019. 2, 3, 6, 7, 12

[39] 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 TPAMI, 2020. 1, 2, 6, 7, 12

[40] Weijing Shi and Raj Rajkumar. Point-gnn: Graph neural network for 3D object detection in a point cloud. In CVPR, pages 1711–1719, 2020. 2, 3, 6

[41] Martin Simon, Karl Amende, Andrea Kraus, Jens Honer, Timo Samann, Hauke Kaulbersch, Stefan Milz, and Horst Michael Gross. Complexer-yolo: Real-time 3D object detection and tracking on semantic point clouds. In CVPR Workshops, pages 0–0, 2019. 1

[42] Martin Simony, Stefan Milzy, Karl Amendey, and Horst-Michael Gross. Complex-yolo: An euler-region-proposal for real-time 3D object detection on point clouds. In ECCVW, pages 0–0, 2018. 1

[43] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications, page 1100612, 2019. 11

[44] 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 CVPR, pages 2446–2454, 2020. 7, 12

[45] Sumesh Thakur and Jiju Peethambaran. Dynamic edge weights in graph neural networks for 3D object detection. arXiv preprint arXiv:2009.08253, 2020. 3

[46] Dominic Zeng Wang and Ingmar Posner. Voting for voting in online point cloud object detection. In Robotics: Science and Systems, pages 10–15607, 2015. 2

[47] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. TOG, pages 1–12, 2019. 3

[48] Yimin Wei, Hao Liu, Tingting Xie, Qiuhong Ke, and Yulan Guo. Spatial-temporal transformer for 3d point cloud sequences. In WACV, pages 1171–1180, 2022. 2

[49] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. Sensors, page 3337, 2018. 1, 2, 6, 7, 12

[50] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3D object detection from point clouds. In CVPR, pages 7652–7660, 2018. 1

[51] Ming Yang, Ying Wu, and Gang Hua. Context-aware visual tracking. IEEE TPAMI, pages 1195–1209, 2008. 5

[52] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3D single stage object detector. In CVPR, pages 11040–11048, 2020. 2, 3, 5, 6, 7, 12

[53] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Std: Sparse-to-dense 3D object detector for point cloud. In ICCV, pages 1951–1960, 2019. 1, 2, 3, 6

[54] Maosheng Ye, Shuangjie Xu, and Tongyi Cao. Hvnet: Hybrid voxel network for lidar based 3D object detection. In CVPR, pages 1631–1640, 2020. 1, 2

[55] 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 ICRA, pages 2274–2280, 2020. 2

[56] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In CVPR, pages 11784–11793, 2021. 7

[57] Na Zhao, Tat-Seng Chua, and Gim Hee Lee. Sess: Self-ensembling semi-supervised 3D object detection. In CVPR, pages 11079–11087, 2020. 2

[58] Wu Zheng, Weiliang Tang, Sijin Chen, Li Jiang, and Chi-Wing Fu. Cia-ssd: Confident iou-aware single-stage object detector from point cloud. In AAAI, 2021. 2, 6

[59] Dingfu Zhou, Jin Fang, Xibin Song, Chenye Guan, Junbo Yin, Yuchao Dai, and Ruigang Yang. Iou loss for 2D/3D object detection. In 3DV, pages 85–94, 2019. 6

[60] 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 CoRL, pages 923–932, 2020. 1

[61] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3D object detection. In CVPR, pages 4490–4499, 2018. 1, 6