2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Cost-Aware Evaluation and Model Scaling for LiDAR-Based 3D Object Detection

Xiaofang Wang and Kris M. Kitani

Abstract— **Considerable research effort has been devoted to LiDAR-based 3D object detection and empirical performance has been significantly improved. While progress has been encouraging, we observe an overlooked issue: it is not yet common practice to compare different 3D detectors under the same cost, *e.g.*, inference latency. This makes it difficult to quantify the true performance gain brought by recently proposed architecture designs. The goal of this work is to conduct a cost-aware evaluation of LiDAR-based 3D object detectors. Specifically, we focus on SECOND, a simple grid-based one-stage detector, and analyze its performance under different costs by scaling its original architecture. Then we compare the family of scaled SECOND with recent 3D detection methods, such as Voxel R-CNN and PV-RCNN++. The results are surprising. We find that, if allowed to use the same latency, SECOND can match the performance of PV-RCNN++, the current state-of-the-art method on the Waymo Open Dataset. Scaled SECOND also easily outperforms many recent 3D detection methods published during the past year. We recommend future research control the inference cost in their empirical comparison and include the family of scaled SECOND as a strong baseline when presenting novel 3D detection methods.**

I. Introduction

LiDAR-based 3D object detection is essential for autonomous driving. Existing research efforts have proposed a diverse range of 3D detectors, where point clouds are organized in various formats (*e.g.*, point-based [33], [40], grid-based [50], [17], [54], range view [3], [9], or hybrid [38], [39], [7] representation) and processed by different architecture components (*e.g.*, PointNet [34], [35], 3D sparse convolution [11], or 2D convolution). These efforts result in a significant boost in detection performance. The Average precision (AP) for vehicle detection on the Waymo Open Dataset [1] has been improved from 56.62% (PointPillars [17]) to 79.25% (PV-RCNN++ [39]) in just three years!

However, despite the promising empirical performance, we make a somewhat worrisome observation on the current state of 3D detection research: the computational cost (*e.g.*, inference latency) is usually not controlled during the comparison of different detectors. Recent 3D detection methods tend to emphasize and attribute the performance gain to their novel architecture design. But it is unclear whether the proposed detectors are faster or slower than the baselines they are comparing to.

Why are we worried about this observation? We note that when developing architectures on ImageNet [37], it is common practice to compare them under the same cost [45], [25], [36], [26], [48], [49]. But this has yet to be the case for 3D object detection. Since simply scaling up an architecture can already boost the accuracy [45], [2], it is unfair to compare different architectures without controlling the cost. Such an unfair comparison makes it unclear whether the performance gain in recent 3D detection methods is actually brought by their proposed architectural changes or simply due to the usage of more computation. This can cause misleading conclusions on the contribution of different architecture components.

By first addressing the unfair comparison problem, we can know the true contribution of the diverse architecture components used in existing methods. This is also important for future research to further push the frontier of 3D detection. Fully addressing this issue surely requires a community-wide effort. We take a step forward by analyzing the performance of a simple grid-based one-stage detector, *i.e.*, SECOND [50], under different costs by scaling its original architecture.

We choose SECOND for the following reasons: (1) SECOND is a widely-used baseline and generally believed to have been significantly outperformed; (2) SECOND has easy-to-use open-source implementation[1] available; and (3) most importantly, SECOND is the common part of several high-performing two-stage detectors (*e.g.*, PV-RCNN++ [39] and Voxel R-CNN [7]), where SECOND is adopted as their first stage to generate region proposals. Studying the performance of SECOND can immediately inform us about whether these sophisticated second stage detectors are necessary to achieve competitive detection performance.

To analyze the performance of SECOND under different costs, we study how to scale its backbone. We show that increasing the pre-head resolution, *i.e.*, the spatial dimension of the feature map being passed to the detection head, is often better than just increasing the network depth or width. A larger pre-head resolution allows a denser sampling of object anchors or keypoints.

Then we compare the family of scaled SECOND against recent 3D detection methods. The results are surprising. We find that, SECOND can easily outperform most recent 3D detection methods after being scaled up. Notably, scaled SECOND can match the performance of PV-RCNN++, the current state-of-the-art on the Waymo Open Dataset, if allowed to use a similar inference latency. Scaled SECOND also easily outperforms many recent methods published during the past year. Our results indicate that the gain brought by the architectural innovation in many recent methods is not

Both authors are with the Robotics Institute, Carnegie Mellon University. Email: {xiaofan2,kkitani}@cs.cmu.edu.

[1]We refer to the open-source implementation in OpenPCDet [47].

as significant as what was shown in their papers.

We summarize our contributions as follows: (1) We point out the vast importance of comparing different 3D detectors under the same cost, which may sound obvious but was overlooked in the recent literature. (2) We provide an extensive analysis on how to scale up the backbone of grid-based 3D detectors, *e.g.*, SECOND, and find that increasing the pre-head resolution is a reliable source for better performance. (3) We introduce the family of scaled SECOND by scaling up the original backbone of SECOND and conduct a cost-aware comparison of scaled SECOND against recent 3D detection methods. Our comparison leads to a surprising observation: simply scaling the backbone in SECOND can already match the state-of-the-art performance on the Waymo Open Dataset.

II. Related Work

A. LiDAR-Based 3D Object Detection

Point clouds captured by LiDAR sensor are irregular. This makes it difficult to directly apply traditional convolutional architectures to point clouds, which have been successful for images and videos but require the input data to be organized in the format of regular grids [22]. Several different ways have been proposed to address this issue. Following the categorization of 3D detectors in PV-RCNN++ [39], we briefly review existing 3D detection methods based on how they represent and process the point cloud.

Point-based Representation. This line of work treats point clouds as unordered point sets and directly processes the raw point cloud. Most of them adopted PointNet or its variant [34], [35] as the backbone [33], [40], [52], [51], [32], [31]. Point-GNN [42] explored using graph neural networks to encode the point cloud by constructing a fixed radius near-neighbors graph. Pan et al. [30] proposed PointFormer, a Transformer architecture for 3D point clouds, to serve as the backbone in point-based detectors. Point-based representation can fully reserve the 3D structure and fine details. But the nearest neighbor search operation used in PointNet or PointNet++ variant is computationally prohibitive as the number of points increases. While the efficiency issue can be partially mitigated by downsampling the point cloud (*e.g.*, only keeping 16384 points [40]), the downsampling inevitably brings performance drop. This limits the application of point-based detectors to large-scale scenes [22].

Grid-based Representation. To deal with the irregularity of point clouds, previous work proposed to divide point clouds into (1) regular grids, *e.g.*, voxels, pillars, and (2) bird's-eye view (BEV), to make it possible to apply convolutional operations. VoxelNet [57] partitioned the space into equally spaced voxels, applied PointNet [34] in each voxel to generate voxel features, and then used dense 3D convolution to further aggregate the spatial context. SECOND [50] improved upon VoxleNet by using 3D sparse convolution [11] and removing the PointNet. PointPillars [17] proposed to organize the point cloud as pillars (vertical columns) to improve the voxel backbone efficiency. Voxel features or pillar features are often projected onto the ground plane, *i.e.*, BEV, before being passed to the detection head, where 2D convolution can be readily applied. Given BEV feature maps, CenterPoint [54] proposed a center-based detection head for 3D object detection without pre-defining axis-aligned anchors. The grid-based representation makes it easy to apply convolutional operations, but suffers from the quantization error caused by dividing the space into regular grids, which can limit the detection performance, especially for distant objects with a few points.

Range View Representation. Range view is a commonly used representation of LiDAR data and can be efficiently processed by 2D convolutional architectures [29]. VeloFCN [18] is the pioneering work in this line, where they designed a fully convolutional network to detect 3D objects from range images. LaserNet [29] also used a fully convolutional network as the backbone. RCD [3] proposed a novel range-conditioned dilation layer to account for the scale variation of objects in range images. RangeDet [9] proposed several strategies to improve pure rage-view-based object detection. Challenges of using range view representation include dealing with scale variation and occlusion [29].

Hybrid Representation. Since different representations have their own pros and cons, previous methods [5], [16], [23], [56], [43], [38], [39], [12] also explored combining multiple representations of the point cloud. For example, MVF [5] proposed a novel multi-view fusion algorithm to effectively use BEV and range view. M3DETR [12] explored fusing multi-representation features from points, voxels and BEV with Transformer. Shi et al. [38] proposed PV-RCNN, a two-stage detection framework that takes the advantage of both the voxel-based and point-based methods. Its extension PV-RCNN++ [39] achieved state-of-the-art performance on the Waymo Open Dataset.

B. Characterizing Architectures

ImageNet [37] has a large impact on designing novel architectures [15], [44], [13], [14], [55], [45], [8] and is the de facto benchmark for evaluating novel architectures. We note that when intending to show a novel architecture is more accurate than previous ones on ImageNet, the common practice is to ensure that the proposed architecture use similar computational cost with the baselines[2] [45], [25], [36], [26], [48], [49]. For example, RegNet [36] considered a wide range of computation regimes and conducted the comparison of architectures within each regime. The very recent Swin Transformer [26] reported model parameters, FLOPs, and throughput in their evaluation.

Previous results [45], [2] have demonstrated that scaling up an architecture, *e.g.*, increasing the number of layers or channels, can significantly improve the accuracy. Therefore, comparing architectures of different costs is unfair and cannot justify that the gain is due to the novel architecture design. Unfortunately, the aforementioned standard practice has not yet been ubiquitously adopted in 3D object detection. Our work is inspired by this standard practice and aims to

[2]Equivalently, it is also common in practice to show the proposed architecture can achieve comparable accuracy with less cost. e.g., FLOPs or latency.

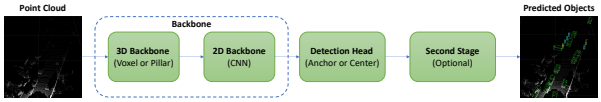


Fig. 1: Main components in a LiDAR-based 3D object detector. We list example choices for a component in the bracket but there could be other choices.

TABLE I: Component Overview of Relevant Detectors.

	3D Backbone	Detection Head	Second Stage
†SECOND-Anchor [50]	Voxel	Anchor	✗
‡SECOND-Center	Voxel	Center	✗
PointPillars [17]	Pillar	Anchor	✗
CenterPoint [54]	Pillar / Voxel	Center	✓
PV-RCNN [38]	Voxel	Anchor / Center	✓
PV-RCNN++ [39]	Voxel	Anchor / Center	✓

† SECOND-Anchor is the original SECOND method using anchor head.
‡ SECOND-Center is equivalent to the first stage of CenterPoint.

quantify how much performance gain in state-of-the-art 3D detection methods is due to architectural innovation.

III. Architecture Overview

This section reviews the architecture details of relevant detection methods to provide background for our analysis. We illustrate a detector as the combination of a backbone, a detection head, and optionally a second stage in Fig. 1. The backbone is further divided into a 3D one and a 2D one. Table I gives a component overview of relevant detectors and we describe more details as follows.

A. SECOND, PointPillars & CenterPoint

SECOND. Most of our analysis focuses on SECOND [50], one of the earliest 3D detection methods and a widely-used baseline in the literature. SECOND first groups the point cloud into voxels and then extracts 3D voxel-wise features using the 3D backbone. The sparse 3D voxel-wise features are then projected onto the ground plane (x and y-axis) to obtain dense 2D BEV features, which is done by channel concatenation across the height dimension (z-axis). The obtained BEV features are then processed by the 2D backbone and passed to the detection head to generate a set of region proposals, including their location, size, orientation and class. Finally, non-maximum suppression is applied on the region proposals to remove redundant object predictions.

Fig. 2 shows the original backbone architecture of SECOND implemented in OpenPCDet [47] (A0 in Table II). The 3D backbone is formed by stacking 3D sparse convolutional layers [11] and consists of four stages. The first layer at each stage, except the first stage, has a stride of 2 to reduce the spatial dimension in 3D. The 2D backbone consists of 2D convolution layers and has two stages. All the layers have a stride of 1 except that the first layer at the second stage has a stride of 2. The output of each stage is transformed or upsampled via transposed convolution and then concatenated across the channel dimension to obtain the final BEV features, which we will refer to as "pre-head features" since the features will be directly passed to

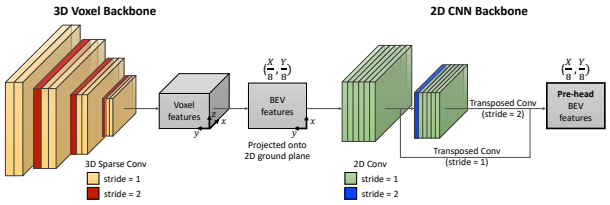


Fig. 2: Backbone Architecture of SECOND (A0 in Table II). Assuming the input point cloud is initially grouped into $X \times Y \times Z$ voxels, the pre-head resolution of the shown A0 backbone, *i.e.*, the spatial dimension of the obtained pre-head BEV features, will be $(\frac{X}{8}, \frac{Y}{8})$.

TABLE II: List of architectures used in our analysis. A0 is the original backbone of SECOND implemented in OpenPCDet.

	3D Depth	3D Width	2D Depth	2D Width	Pre-Head
A0	2, 3, 3, 3	16, 32, 64, 64	6, 6	128, 256	$(X/8, Y/8)$
A0-deep	8, 12, 12, 12	16, 32, 64, 64	24, 24	128, 256	$(X/8, Y/8)$
A0-wide	2, 3, 3, 3	32, 64, 128, 128	6, 6	256, 512	$(X/8, Y/8)$
A0-d&w	3, 5, 5, 5	28, 56, 112, 112	9, 9	224, 448	$(X/8, Y/8)$
A1	2, 4 ,4	32, 64, 64	6, 6	128, 256	$(X/4, Y/4)$
A2	3, 6, 6	48, 96, 144	12, 12	128, 256	$(X/4, Y/4)$

the detection head to predict region proposals. Relatedly, the spatial dimension of the pre-head features given by A0 will be $(\frac{X}{8}, \frac{Y}{8})$, assuming the input point cloud is initially grouped into $X \times Y \times Z$ voxels. We discuss the details of detection head after PointPillars.

PointPillars. We also include PointPillars [17] in our analysis, another popular baseline for 3D detection. PointPillars [17] is similar to SECOND except for the 3D backbone, where the point cloud is organized as pillars (vertical columns) instead of voxels. They obtain pillar features by aggregating point features inside each pillar and then convert pillar features into a pseudo-image, *i.e.*, BEV features, which are passed to the 2D backbone and detection head.

Detection Head. As listed in Table I, the original SECOND and PointPillars method use an anchor-based detection head, which pre-defines axis-aligned anchors of different classes on each location. We refer to the original SECOND method using anchor head as SECOND-Anchor. The anchor head takes BEV features as input and uses a convolutional layer to regress the residuals between the ground truth object boxes and pre-defined anchors, as well as predict the class probabilities of each anchor.

CenterPoint [54] proposed a center-based detection head, which dose not require pre-defining anchors and achieves superior performance over the anchor head. This center head first detects object centers using a keypoint detector and then regresses other attributes, *e.g.*, object size and orientation, for each detected center. The center head is generic and can be used as a drop-in replacement for anchor head. Therefore, our analysis also considers the center head and uses it within SECOND by replacing the anchor head in SECOND-Anchor as center head, which we list as SECOND-Center in Table I. For clarification, the full method of CenterPoint is a two-

stage detector and SECOND-Center is exactly the same as the first stage of CenterPoint.

B. Part-A2-Net, PV-RCNN, PV-RCNN++ & Voxel R-CNN

Part-A2-Net [41], PV-RCNN [38], PV-RCNN++ [39], and Voxel R-CNN [7] are all two-stage 3D detectors achieving competitive performance. They all use SECOND as their first stage to generate initial region proposals, which are then further refined in the second stage. Similar to SECOND, both anchor head and center head can be used in these two-stage detectors. Notably, PV-RCNN++ achieves state-of-the-art performance on the Waymo Open Dataset.

IV. Experimental Setup

We conduct experiments with OpenPCDet, an open-source code base for LiDAR-based 3D object detection. OpenPCDet is the official code release of Part-A2-Net [41], PV-RCNN [38], PV-RCNN++ [39], and Voxel R-CNN [7], and also supports many other methods, including SECOND [50] and PointPillars [17].

Dataset and Metrics. We use the Waymo Open Dataset [1], the largest public benchmark for LiDAR-based 3D object detection, in our experiments. It contains 798 train sequences (∼158k frames) and 202 validation sequences (∼40k frames). Following the standard protocol, we adopt average precision (AP) and average precision weighted by heading (APH) as the metrics and evaluate in two difficulty levels (LEVEL_1 and LEVEL_2).

Training Setup. By default, we train on all the train sequences and evaluate on all the validation sequences (100% training setup). To save training time for the analysis in Sec. V, we adopt the 20% training setup provided in OpenPCDet [47]. Under this setup, we train on 20% frames uniformly sampled from the train sequences but still evaluate on all the validation sequences. Since LiDAR frames in one sequence are highly correlated, 20% of data is usually representative enough.

Inference Latency. We use batch size 1 when measuring the latency of a detector, following the convention in detection [46]. The latency is measured on a Nvidia GeForce RTX 3090 GPU and a AMD EPYC 7402 24-Core CPU.

V. Depth, Width, and Pre-Head Resolution

Scaling the depth (number of layers), width (number of channels), or input image resolution have been widely used to improve the performance of a model [13], [14], [45], [2]. But it is still an open question about what the optimal scaling strategy is. There can be a large performance variation among different scaling configurations even when then the amount of cost is controlled, especially in the large computation regime [2]. For example, EfficientNet [45] demonstrated that compound scaling, *i.e.*, scaling all three dimensions including depth, width, and resolution, is better than scaling only one of the dimensions. Bello et al. [2] observed diminishing returns in very large image resolutions in EfficientNet and suggested that one should increase the resolution slowly. To form the basis of our analysis, this section analyzes different ways to scale the backbone of SECOND.

TABLE III: Performance of SECOND-Anchor with different backbones on the Waymo validation set (20% training). Scaling the pre-head resolution from $(\frac{X}{8}, \frac{Y}{8})$ to $(\frac{X}{4}, \frac{Y}{4})$ significantly improve the performance on all classes. But further increasing the resolution does not help.

Anchor Head	LEVEL_2 3D APH				Latency	Params	Memory	Pre-Head
	Vehicle	Pedestrian	Cyclist	mAPH	(ms)	(M)	(GB)	Resolution
A0	62.02	47.49	53.53	54.35	27	5.33	5.8	$(X/8, Y/8)$
A0+Upsample	64.61	51.92	59.81	58.78	38	5.69	9.0	$(X/4, Y/4)$
A0+Upsample×2	64.51	50.50	59.82	58.28	52	5.69	20.0	$(X/2, Y/2)$
A1	**65.65**	**59.20**	**64.33**	**63.06**	65	5.56	14.4	$(X/4, Y/4)$

TABLE IV: Performance of SECOND-Center with different backbones on the Waymo validation set (20% training). The advantage of scaling pre-head resolution generalizes to the center-based detection head.

Center Head	LEVEL_2 3D APH				Latency	Params	Memory	Pre-Head
	Vehicle	Pedestrian	Cyclist	mAPH	(ms)	(M)	(GB)	Resolution
A0	62.65	58.23	64.87	61.92	28	5.78	5.6	$(X/8, Y/8)$
A0+Upsample	64.86	61.26	65.79	63.98	43	6.14	10.0	$(X/4, Y/4)$
A1	**64.92**	**65.35**	**67.96**	**66.08**	67	6.01	14.7	$(X/4, Y/4)$

A. Pre-Head Resolution

Unlike images, there is no notion of resolution for raw point clouds. Therefore, we consider the pre-head resolution, *i.e.*, the spatial dimension of the pre-head BEV features, as an alternative choice to scale the backbone.

We note that previous work on 2D object detection [24], [10], [46] explored using multi-scale feature maps, whose main motivation is to handle the scale variation of objects in images, *i.e.*, using a higher-resolution feature map for larger anchors, since the size of the same object could vastly vary depending on its distance to the camera. But this is not the case for LiDAR point clouds as the size of a specific object in point clouds is fixed no matter its distance to the sensor.

The advantage of a larger pre-head resolution for LiDAR-based detection is in the denser sampling of anchors or keypoints. The anchor head places pre-defined anchors on every location of the BEV features. Therefore, a larger pre-head feature map resolution means more anchors are used. For center head [54], which detects object centers via keypoint estimation, a larger resolution means more keypoints are classified.

Now we empirically investigate whether scaling the pre-head resolution leads to performance improvement. Starting from A0, the original backbone in SECOND with a pre-head resolution of $(\frac{X}{8}, \frac{Y}{8})$, we consider the following backbones to increase the pre-head resolution:

- **A0+Upsample:** we change the stride of the transposed convolution at the end of first stage in the 2D backbone to 2, and add another transposed convolutional layer of a stride 2 at the end of the second stage. This yields a pre-head resolution of $(\frac{X}{4}, \frac{Y}{4})$.
- **A0+Upsample×2:** we further upsamples the pre-head feature map given by 'A0+Upsample' by 2x with Bilinear interpolation. This yields a resolution of $(\frac{X}{2}, \frac{Y}{2})$.
- **A1:** we remove the last stage in the 3D backbone of A0 to perform less downsampling in the network. The number of layers and channels are slightly adjusted so that A1 has a similar number of parameters to A0. A1 has a pre-head resolution of $(\frac{X}{4}, \frac{Y}{4})$.

TABLE V: Depth vs. Width vs. Pre-Head Resolution. We compare SECOND-Anchor with different scaled backbones under the same latency on the Waymo validation set (20% training). Scaling the pre-head resolution provides the highest overall mAPH with the fewest parameters.

Anchor Head	LEVEL_2 3D APH				Latency	Params	Memory
	Vehicle	Pedestrian	Cyclist	mAPH	(ms)	(M)	(GB)
A0	62.02	47.49	53.53	54.35	27	5.33	5.8
Without Residual Connections							
A0-deep	59.98	35.53	43.15	46.22	55	20.92	13.5
A0-wide	65.10	53.01	59.61	59.24	61	19.96	8.0
A0-d&w	65.50	51.94	59.46	58.97	68	23.76	9.9
A1	**65.65**	**59.20**	**64.33**	**63.06**	65	5.56	14.4
With Residual Connections							
A0-deep$_{res}$	**66.98**	54.80	60.35	60.71	59	21.23	15.4
A0-wide$_{res}$	65.82	55.17	60.41	60.47	64	20.19	8.0
A0-d&w$_{res}$	66.69	56.00	61.70	61.46	66	21.66	10.4
A1$_{res}$	65.78	**59.82**	**64.27**	**63.29**	67	5.65	13.7

We show the performance of different backbones in Table III. We also report the inference latency, number of parameters, and memory footprint at batch size 2 during training for completeness. Here we focus on analyzing whether increasing the pre-head resolution can improve the performance, so we do not control the latency of different backbones to be the same.

As shown in Table III, scaling the resolution to $(\frac{X}{4}, \frac{Y}{4})$ is beneficial as A1 and 'A0+Upsample' outperform A0 by a large margin on all classes. Table IV provides the results of SECOND-Center with different backbones and we see the benefit of a larger pre-head resolution generalizes to center head. But we also notice that further increasing the pre-head resolution does not bring any additional gain ('A0+Upsample' vs. 'A0+Upsample$\times$2'). Therefore, we conclude that scaling the pre-head resolution is a reliable source for better performance but one should refrain from increasing the resolution aggressively.

B. Depth vs. Width vs. Pre-Head Resolution

We now compare the following ways to scale the backbone in SECOND: (1) depth only (A0-deep), (2) width only (A0-wide), (3) depth and width at the same time (A0-d&w), and (4) pre-head resolution only (A1). The architecture details are available in Table II and the results are shown in Table V. We control the inference latency of of different backbones are to be similar for fair comparison.

While other backbones can easily improve the performance, we notice that A0-deep underperforms the original A0 backbone. We conjecture this is due to the lack of residual connections. A0-deep is much deeper than other networks and harder to train. Therefore, we add residual connections to all the backbones (subscripted by 'res' in Table V).

All the backbones benefit from adding residual connections and significantly outperform A0. Among the different choices, the compound scaling of depth and width is better than scaling one dimension only, echoing Tan et al. [45]. Scaling the pre-head resolution achieves the best overall performance, while scaling depth is the best for vehicle detection. Scaling the pre-head resolution also significantly saves the number of parameters compared with other choices.

VI. Scaled SECOND vs. Recent Methods

A. Family of Scaled SECOND

Based on the above analysis on how to scale the backbone in SECOND, we introduce the family of scaled SECOND. In addition to the A0 and A1$_{res}$ backbone, we design a larger backbone A2$_{res}$ to cover the high latency regime. A2$_{res}$ is obtained by adding residual connections in A2, where A2 is obtained by increasing the depth and width of A1 (see Table II for architecture details).

Then the family of scaled SECOND includes three backbones: A0, A1$_{res}$, and A2$_{res}$. As mentioned in Sec. III, both the anchor head and center head can be used within SECOND. So we have both SECOND-Anchor and SECOND-Center with the three backbones. To be clear, SECOND-Anchor is the original method of SECOND [50] and SECOND-Center replaces the anchor head in the original SECOND with the center head proposed in CenterPoint [54].

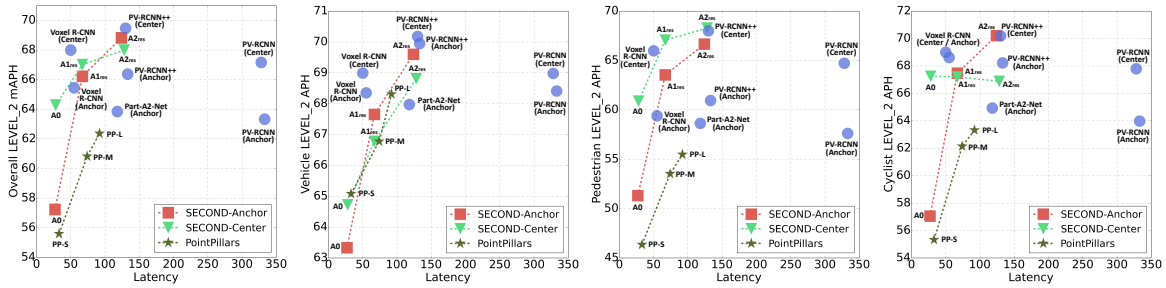
Next, we will compare the family of scaled SECOND against recent LiDAR-based 3D object detection methods. We first compare scaled SECOND against several two-stage detectors in Sec. VI-B. These two-stage detectors are specifically selected as they use SECOND as their first stage, including Part-A2-Net [41], PV-RCNN [38], PV-RCNN++ [39], and Voxel R-CNN [7]. Then we extend the comparison to more methods in Sec. VI-C.

B. Comparison Against Selected Two-Stage Detectors

We compare the family of scaled SECOND against the following two-stage detectors in Figure 3: Part-A2-Net [41], PV-RCNN [38], PV-RCNN++ [39], and Voxel R-CNN [7]. As mentioned above, we select them as they all use SECOND as their first stage. We summarize the results in Figure 3. All the results in Figure 3 are obtained using OpenPCDet [47], the official implementation of the selected two-stage detectors, to ensure reproducibility and fair comparison. Since the detection head can have a big influence on the performance, we consider both the anchor head and center head for PV-RCNN, PV-RCNN++ and Voxel R-CNN in Figure 3 to provide a more complete comparison.

Overall Comparison. We observe from Figure 3a that scaled SECOND significantly outperforms Part-A2-Net and PV-RCNN after controlling the latency in the comparison. Only Voxel R-CNN and PV-RCNN++ can pass the test of scaled SECOND, *i.e.*, more accurate than scaled SECOND when using a similar or smaller latency. But this only happens if they use the center head. Voxel R-CNN (Anchor) and PV-RCNN++ (Anchor) do not pass the test of scaled SECOND either.

PV-RCNN++. We take a closer look at PV-RCNN++ as it is the current state-of-the-art on the Waymo Open Dataset. The original SECOND method (SECOND-Anchor-A0) significantly underperforms PV-RCNN++ (Anchor). But simply scaling up its original backbone (A0) to A1$_{res}$ can easily achieve a similar mAPH with PV-RCNN++ (Anchor) while being twice faster during inference. The highest-performing method PV-RCNN++ (Center) is only slightly



(a) Overall mAPH (b) Vehicle APH (c) Pedestrian APH (d) Cyclist APH

Fig. 3: Scaled SECOND vs. Selected Two-Stage Detectors. Only Voxel-RCNN (Center) and PV-RCNN++ (Center) can outperform the overall mAPH of scaled SECOND with a similar or smaller latency.

TABLE VI: Scaled SECOND vs. Other Recent 3D Detection Methods. Most methods fail to outperform SECOND-Anchor-A1$_{\text{res}}$, which is exactly the same as the original SECOND [50] except for using a larger backbone.

	Venue	Latency (ms)	Vehicle AP	Vehicle APH	Pedestrian AP	Pedestrian APH	Cyclist AP	Cyclist APH	Overall mAPH
PPBA [6]	ECCV 2020	-	-	53.4	-	53.9	-	-	-
LiDAR R-CNN [22]	CVPR 2021	-	64.7	64.2	63.1	51.7	66.1	64.4	60.10
3D-MAN [53]	CVPR 2021	-	67.61	67.14	62.58	59.04	-	-	-
PPC [4]	CVPR 2021	-	-	56.7	-	61.5	-	-	-
MGAF-3DSSD [19]	ACM MM 2021	†∼ 60	65.35	-	-	-	-	-	-
RangeDet [9]	ICCV 2021	‡∼ 58	64.03	63.57	67.60	**63.89**	63.33	62.08	63.18
VoTr-TSD [28]	ICCV 2021	♯> 300	65.91	65.29	-	-	-	-	-
Pyramid-PV [27]	ICCV 2021	♯> 300	67.23	66.68	-	-	-	-	-
SECOND-Anchor-A1$_{\text{res}}$		67	**68.13**	**67.65**	**71.57**	63.54	**68.53**	**67.47**	**66.22**
Voxel-to-Point [20]	ACM MM 2021	-	69.77	-	-	-	-	-	-
SECOND-Anchor-A2$_{\text{res}}$		124	**70.06**	**69.60**	**73.98**	**66.65**	**71.22**	**70.21**	**68.82**

† We estimate its latency to be ∼ 60 ms on Waymo as it is ∼2x faster than Part-A2-Net on KITTI [19]
‡ Measured as 12 fps on 2080Ti [9]. We estimate its latency to be 58 ms on RTX 3090 based on [21].
♯ We estimate its latency to be larger than 300 ms as it is slower than PV-RCNN on KITTI [28], [27].

better than SECOND-Anchor-A2$_{\text{res}}$ if SECOND is allowed to use a similar latency. The performance gain brought by PV-RCNN++ is much smaller than what was shown in paper.

Voxel R-CNN. Voxel R-CNN (Center) outperforms SECOND-Center-A1$_{\text{res}}$ in the overall mAP with a smaller latency. The strong performance and efficiency of Voxel R-CNN is mainly due to their proposed Voxel RoI pooling [7] in the second stage, which does not use the expensive ball query to find nearest neighbors in the 3D space but rather uses an efficient voxel query operation.

C. Full Comparison Against Recent Methods

We now extend the comparison to other recent 3D object detection methods in Table VI. While all these methods are proposed after SECOND [50], we observe that most methods fail to outperform SECOND-Anchor-A1$_{\text{res}}$, which is exactly the same as the original SECOND except for using a larger backbone. SECOND-Anchor-A1$_{\text{res}}$ is both faster and more accurate than some recent methods, such as VoTR-TSD [28] and Pyramid-PV [27].

The strong performance and simplicity of scaled SECOND should motivate future research to include them as baselines whenever proposing novel 3D object detectors.

D. Comparison Under Multiple Latencies

In the above, we only consider one specific latency when comparing two methods. The family of scaled SECOND contains backbones of different sizes and allows us to have a more complete comparison between methods under multiple latencies. We observe that the conclusion under one latency may not generalize to another and the ranking of two methods could flip.

Anchor Head vs. Center Head. The center head was proposed in CeterPoint [54] and demonstrated impressive performance gain over the anchor head. But CenterPoint only experimented with a small backbone (similar size to A0). We observe that the benefit of center head shrinks as the backbone size grows. As shown in Figure 3a, the anchor head considerably underperforms the mAPH of center head when using A0. But after scaling up A0 to A2$_{\text{res}}$, the anchor head obtains a higher mAPH than the center head. We observe a similar trend for vehicle or pedestrian detection. For example, for vehicle detection in Figure 3b, the ranking of anchor head and center head flips after the backbone is scaled up to A1$_{\text{res}}$.

SECOND-Anchor vs. PointPillars. We observe that the ranking of SECOND-Anchor and PointPillars changes under different latencies. For example, PointPillars outperforms SECOND by $\sim 2\%$ on vehicle detection under the low latency regime (PointPillars-S vs. SECOND-Anchor-A0 in Figure 3b). The low latency regime is where the PointPillars focused on when it was proposed. However, as the backbone size grows, PointPillars-M underperforms SECOND-Anchor-A1$_{\text{res}}$ on vehicle detection.

VII. Conclusion

To correctly evaluate the architecture design space of 3D detectors, we point out that it is important to compare different architectures under the same cost. Following this philosophy, we conduct an analysis of how to scale the backbone of SECOND, a simple baseline that is generally believed to have been significantly surpassed, and then introduce the family of scaled SECOND. Scaled SECOND sets a strong baseline for future research on 3D object detection: it outperforms most recent methods and can match the performance of the state-of-the-art method PV-RCNN++ on the Waymo Open Dataset if allowed to use a similar latency. We hope our analysis can encourage future research to adopt the good practice of cost-aware evaluation and include the family of scaled SECOND as a strong baseline when presenting novel 3D detection methods. We also show that the ranking of two methods can flip under different latencies and suggests that one should conduct the comparison under multiple latencies if possible.

REFERENCES

[1] "Waymo open dataset: An autonomous driving dataset," 2019.

[2] I. Bello, W. Fedus, X. Du, E. D. Cubuk, A. Srinivas, T.-Y. Lin, J. Shlens, and B. Zoph, "Revisiting resnets: Improved training and scaling strategies," *NeurIPS*, 2021.

[3] A. Bewley, P. Sun, T. Mensink, D. Anguelov, and C. Sminchisescu, "Range conditioned dilated convolutions for scale invariant 3d object detection," in *CoRL*, 2020.

[4] Y. Chai, P. Sun, J. Ngiam, W. Wang, B. Caine, V. Vasudevan, X. Zhang, and D. Anguelov, "To the point: Efficient 3d object detection in the range image with graph convolution kernels," in *CVPR*, 2021.

[5] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *CVPR*, 2017.

[6] S. Cheng, Z. Leng, E. D. Cubuk, B. Zoph, C. Bai, J. Ngiam, Y. Song, B. Caine, V. Vasudevan, C. Li, *et al.*, "Improving 3d object detection through progressive population based augmentation," in *ECCV*, 2020.

[7] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li, "Voxel r-cnn: Towards high performance voxel-based 3d object detection," in *AAAI*, 2021.

[8] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021.

[9] L. Fan, X. Xiong, F. Wang, N. Wang, and Z. Zhang, "Rangedet: In defense of range view for lidar-based 3d object detection," in *CVPR*, 2021.

[10] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "Nas-fpn: Learning scalable feature pyramid architecture for object detection," in *CVPR*, 2019.

[11] B. Graham, M. Engelcke, and L. Van Der Maaten, "3d semantic segmentation with submanifold sparse convolutional networks," in *CVPR*, 2018.

[12] T. Guan, J. Wang, S. Lan, R. Chandra, Z. Wu, L. Davis, and D. Manocha, "M3detr: Multi-representation, multi-scale, mutual-relation 3d object detection with transformers," in *WACV*, 2022.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[14] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv:1704.04861*, 2017.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NeurIPS*, 2012.

[16] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3d proposal generation and object detection from view aggregation," in *IROS*, 2018.

[17] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *CVPR*, 2019.

[18] B. Li, T. Zhang, and T. Xia, "Vehicle detection from 3d lidar using fully convolutional network," in *RSS*, 2016.

[19] J. Li, H. Dai, L. Shao, and Y. Ding, "Anchor-free 3d single stage detector with mask-guided attention for point cloud," in *ACM Multimedia*, 2021.

[20] ——, "From voxel to point: Iou-guided 3d object detection for point cloud with voxel-to-point decoder," in *ACM Multimedia*, 2021.

[21] M. Li, "Deep learning gpu benchmark: A latency-based approach," https://mtli.github.io/gpubench/, 2022.

[22] Z. Li, F. Wang, and N. Wang, "Lidar r-cnn: An efficient and universal 3d object detector," in *CVPR*, 2021.

[23] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3d object detection," in *ECCV*, 2018.

[24] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *CVPR*, 2017.

[25] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," in *ICLR*, 2019.

[26] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *ICCV*, 2021.

[27] J. Mao, M. Niu, H. Bai, X. Liang, H. Xu, and C. Xu, "Pyramid r-cnn: Towards better performance and adaptability for 3d object detection," in *ICCV*, 2021.

[28] J. Mao, Y. Xue, M. Niu, H. Bai, J. Feng, X. Liang, H. Xu, and C. Xu, "Voxel transformer for 3d object detection," in *ICCV*, 2021.

[29] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington, "Lasernet: An efficient probabilistic 3d object detector for autonomous driving," in *CVPR*, 2019.

[30] X. Pan, Z. Xia, S. Song, L. E. Li, and G. Huang, "3d object detection with pointformer," in *CVPR*, 2021.

[31] C. R. Qi, X. Chen, O. Litany, and L. J. Guibas, "Imvotenet: Boosting 3d object detection in point clouds with image votes," in *CVPR*, 2020.

[32] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3d object detection in point clouds," in *ICCV*, 2019.

[33] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3d object detection from rgb-d data," in *CVPR*, 2018.

[34] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *CVPR*, 2017.

[35] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *NeurIPS*, 2017.

[36] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollár, "Designing network design spaces," in *CVPR*, 2020.

[37] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *IJCV*, 2015.

[38] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "Pv-rcnn: Point-voxel feature set abstraction for 3d object detection," in *CVPR*, 2020.

[39] S. Shi, L. Jiang, J. Deng, Z. Wang, C. Guo, J. Shi, X. Wang, and H. Li, "Pv-rcnn++: Point-voxel feature set abstraction with local vector representation for 3d object detection," *arXiv:2102.00463*, 2022.

[40] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *CVPR*, 2019.

[41] S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li, "From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network," *TPAMI*, 2020.

[42] W. Shi and R. Rajkumar, "Point-gnn: Graph neural network for 3d object detection in a point cloud," in *CVPR*, 2020.

[43] P. Sun, W. Wang, Y. Chai, G. Elsayed, A. Bewley, X. Zhang, C. Sminchisescu, and D. Anguelov, "Rsn: Range sparse net for efficient, accurate lidar 3d object detection," in *CVPR*, 2021.

[44] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015.

[45] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *ICML*, 2019.

[46] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *CVPR*, 2020.

[47] O. D. Team, "Openpcdet: An open-source toolbox for 3d object detection from point clouds," https://github.com/open-mmlab/OpenPCDet, 2020.

[48] X. Wang, S. Cao, M. Li, and K. M. Kitani, "Neighborhood-aware neural architecture search," in *BMVC*, 2021.

[49] X. Wang, D. Kondratyuk, E. Christiansen, K. M. Kitani, Y. Alon, and E. Eban, "Wisdom of committees: An overlooked approach to faster and more accurate models," in *ICLR*, 2022.

[50] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors*, 2018.

[51] Z. Yang, Y. Sun, S. Liu, and J. Jia, "3dssd: Point-based 3d single stage object detector," in *CVPR*, 2020.

[52] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, "Std: Sparse-to-dense 3d object detector for point cloud," in *ICCV*, 2019.

[53] Z. Yang, Y. Zhou, Z. Chen, and J. Ngiam, "3d-man: 3d multi-frame attention network for object detection," in *CVPR*, 2021.

[54] T. Yin, X. Zhou, and P. Krahenbuhl, "Center-based 3d object detection and tracking," in *CVPR*, 2021.

[55] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *CVPR*, 2018.

[56] Y. Zhou, P. Sun, Y. Zhang, D. Anguelov, J. Gao, T. Ouyang, J. Guo, J. Ngiam, and V. Vasudevan, "End-to-end multi-view fusion for 3d object detection in lidar point clouds," in *CoRL*, 2019.

[57] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *CVPR*, 2018.