# Re-VoxelDet: Rethinking Neck and Head Architectures for High-Performance Voxel-based 3D Detection

Jae-Keun Lee[2,3*] Jin-Hee Lee[1*] Joohyun Lee[1] Soon Kwon[1,2†] Heechul Jung[3†]
[1]DGIST [2]FutureDrive Inc. [3]Kyungpook National University
lejk8104@gmail.com {jhlee07,jhlee0714,soonyk}@dgist.ac.kr heechul@knu.ac.kr

## Abstract

*LiDAR-based 3D object detectors usually adopt grid-based approaches to handle sparse point clouds efficiently. However, during this process, the down-sampled features inevitably lose spatial information, which can hinder the detectors from accurately predicting the location and size of objects. To address this issue, previous researches proposed sophisticatedly designed neck and head modules to effectively compensate for information loss. Inspired by the core insights of previous studies, we propose a novel voxel-based 3D object detector, named as Re-VoxelDet, which combines three distinct components to achieve both good detection capability and real-time performance. First, in order to learn features from diverse perspectives without additional computational costs during inference, we introduce Multi-view Voxel Backbone (MVBackbone). Second, to effectively compensate for abundant spatial and strong semantic information, we design Hierarchical Voxel-guided Auxiliary Neck (HVANeck), which attentively integrates hierarchically generated voxel-wise features with RPN blocks. Third, we present Rotation-based Group Head (RGHead), a simple yet effective head module that is designed with two groups according to the heading direction and aspect ratio of the objects. Through extensive experiments on the Argoverse2, Waymo Open Dataset and nuScenes, we demonstrate the effectiveness of our approach. Our results significantly outperform existing state-of-the-art methods. We plan to release our model and code[1] in the near future.*

## 1. Introduction

Recently, grid-based 3D object detection approaches [11, 13, 14, 16, 28–30, 35] using LiDAR have attracted significant attention as major streams in the autonomous driving perception technology. These grid-based methods can be broadly divided into two categories: voxel-based and pillar-based. In general, they initially transform point clouds into regular voxels or pillars, and then encode the sparse features through 3D sparse convolutional networks (Sparse CNNs) [9, 10]. After that, the encoded features are fed into neck and head modules, to predict objects from point clouds. Although these approaches were proven effective, several critical challenges still remain to be solved.

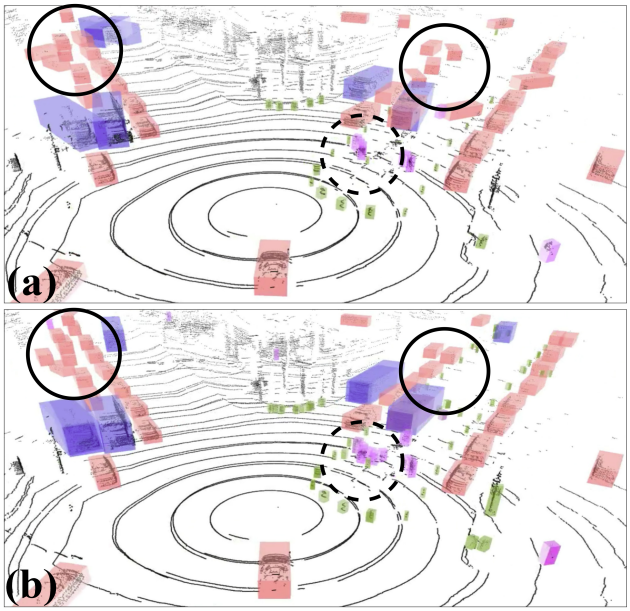


Figure 1. **The visualization results of CenterPoint and Re-VoxelDet on Argoverse2 validation split.** (a) represents the 3D bounding boxes predicted by CenterPoint. (b) indicates the 3D bounding boxes predicted by our Re-VoxelDet. Each bounding box includes fine-grained color information for class groups (i.e., blue, red, pink, brown, etc.)

Firstly, to alleviate the computational burden within the detector, most detectors perform the down-sampling process that progressively reduces the size of the features. This is because the down-sampling operation assists the network to learn various sizes of features, but it negatively affects accuracy and leads to a loss of spatial information. To ad-

*Equal contribution.
†Corresponding authors.
[1]https://github.com/JH-Research/Re-VoxelDet

dress this issue, the previous work mainly applies neck structures, such as SECOND [28], PointPillar [16], CenterPoint [29], and PillarNet [11]. These structures achieve slight compensation by simply integrating refined spatial features and weak semantic features through the sequential connection of multiple region proposal network (RPN) blocks. However, according to comparative experiments results, it is observed that the existing neck modules are still lower accuracy in detecting small objects such as pedestrians, motorcyclists, strollers, and stop signs (See Tab. 6 and Tab. 7). This is assumed to result from the previously employed neck structure [29] not effectively utilizing the abundant spatial information obtained from the backbone networks. The second challenge arises from multi-group head [36] module used for 3D object detection. This module is leveraged to improve detection performance by grouping classes with similar sizes and shapes during the training phase. Unfortunately, the multi-group head module has not been sufficiently explored in datasets like Argoverse2 [26] and nuScenes [2], which have insufficient points and exhibit long-tailed class distributions. As the number of classes increases, the current head network introduces fine-grained sub-heads to further enhance detection accuracy. Nevertheless, these sub-heads lead to increased complexity and slower inference speed. Additionally, this head may be hard to generate high-quality bounding boxes, due to the presence of objects with a large aspect ratio (i.e., vehicles, large trucks, buses, etc.). In general, accurately predicting the heading direction for these objects is difficult.

In this paper, we aim to address the above mentioned issues by improving the whole architecture of the 3D object detector, which consists of the backbone, neck, and head modules. To achieve stable and accurate detection, we propose a novel 3D object detection framework called Re-VoxelDet. This framework with a compact design comprises three essential modules: (1) Multi-view Voxel Backbone (MVBackbone) (2) Hierarchical Voxel-guided Auxiliary Neck (HVANeck), and (3) Rotation-based Group Head (RGHead). Specifically, to boost the representation capacity of the MVBackbone, we introduce Sparse Reparameterized Feature blocks (SRF blocks). These blocks are designed to train the features from multiple perspectives through their multi-branches, and to inference without additional costs by means of a reparameterized branch. Next, the HVANeck is designed as the hierarchical networks to attentively fuse multi-scale spatial features produced by the MVBackbone and semantic features from each stage of the RPN blocks. This helps to effectively minimize the loss of spatial information and generate discriminative and enriched features. Finally, the proposed RGHead is constructed with separate dual sub-heads. These sub-heads group similar classes based on their aspect ratio and heading sensitivity for accurately predicting their heading direction, by freezing heading direction within a certain rotation range of the vehicles. In particular, our detector achieves robust detection performance through HVANeck compensating spatial information and RGHead predicting accurate heading, and it also demonstrates accelerated computation for real-time applications.

To demonstrate the superiority and effectiveness of our method, we conduct extensive experiments on three large-scale autonomous driving datasets, such as Argoverse2 [26], Waymo Open Dataset (WOD) [25], and nuScenes [2]. Our experimental results indicate that the proposed detector achieves state-of-the-art performance on Argoverse2, WOD, and nuScenes datasets. Significantly, our model especially surpasses previous methods with remarkable improvements by achieving 38.2 mAP and 29.4 CDS on Argoverse2, 71.0 mAPH (Level 2) on WOD, and 64.6 mAP and 70.5 NDS on nuScenes. Furthermore, compared to the faster baseline detector PillarNet, our detector demonstrates about 35% faster performance on nuScenes and Argoverse2 datasets. To the best of our knowledge, Re-VoxelDet, with its outstanding object detection performance, is the first 3D detector to enhance the entire architecture.

As shown in Fig. 1, we found that Re-VoxelDet demonstrates a notably robust detection performance when compared with a strong baseline CenterPoint [29]. As indicated by the highlighted circular regions in the figure, our detector accurately predicts bounding box locations for objects such as cars and buses, which have a relatively large aspect ratio. Moreover, a noticeable decrease in missed detections of smaller objects like pedestrians and traffic cones is observed. Please refer to the dotted circular line. This observation implies that our proposed neck and head modules deliver enhanced performance in detecting smaller or more far-away objects, which contain fewer points.

## 2. Related Work

### 2.1. LiDAR-based 3D Object Detection

Depending on how to handle 3D point clouds, grid-based 3D object detectors [5,6,11,13,14,16,20,28,29,34–36] can be mainly classified into two categories: voxel-based and pillar-based approaches. Early voxel-based approaches [35] first convert point clouds into regular voxel volumes using simple PointNet [21], and then process voxelized points with 3D dense convolutional networks. However, these approaches suffer from the high computational overhead when dealing with large-scale point clouds. To resolve this issue, SECOND [28] efficiently reduces the time-consuming 3D backbone by introducing sparse convolution algorithm [9, 10]. This algorithm focuses on calculating non-empty voxel regions with valid points, thereby improving computation efficiency. Recently, the success of an enlarged receptive field in the 2D image domain [18, 27] has inspired a lot

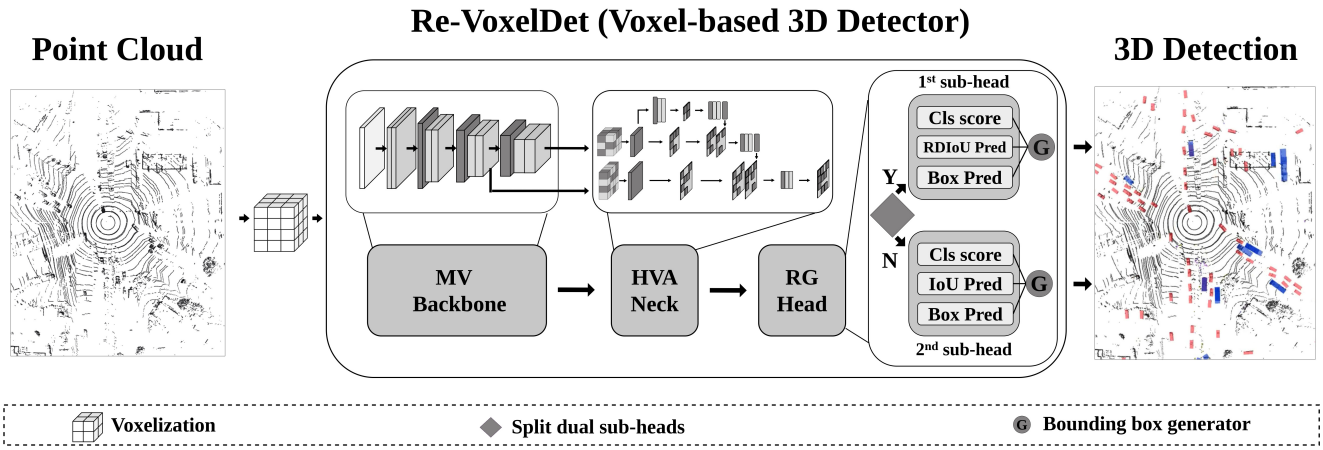


Figure 2. **The overall pipeline of Re-VoxelDet.** Our model consists of three key modules: (1) Multi-view Voxel Backbone (MVBackbone) module generates voxel-wise features at each stage, enhancing the overall feature representation capacity of the model. (2) Hierarchical Voxel-guided Auxiliary Neck (HVANeck) module meticulously merges voxel-wise features with RPN blocks, resulting in rich spatial-semantic features for detecting both smaller and larger objects. (3) Rotation-based Group Head (RGHead) module is subdivided into dual sub-heads, focusing on object's heading and aspect ratio, to improve inference time and detection accuracy. Note that, 'Cls score', 'IoU Pred', and 'Box Pred' represent classification score, IoU prediction score, and bounding box regression, respectively. SPMaxPool denotes depth-wise Sparse Max pooling.

of research to explore large kernel techniques to further improve the performance of 3D sparse convolutional networks [4, 5, 19]. Large Kernel 3D [4] proposes a novel spatial-wise partition convolution for applying 3D large kernels in sparse convolutional networks. LinK [19] focuses on efficiently expanding the 3D kernel size with minimal computational cost. To achieve this, they present a linear kernel generator-based 3D large kernel that dynamically assigns weights only to the non-empty voxels. VoxelNeXt [5] introduces a fully sparse convolutional network-based 3D detector with a simplified design.

Pillar-based approaches [11, 16, 34] mainly convert irregular point clouds into a regular 2D pseudo-image. This process is similar to the voxelization, but it does not consider the height of the voxels. These methods then utilize existing dense convolutional networks for 3D object detection. Their efficient structure strikes a good balance between speed and accuracy. However, sometimes these approaches underperform significantly when compared to voxel-based methods. Therefore, recent pillar-based approaches [11, 34] apply additional 2D backbones to enhance detection performance with minimal compromise in computational costs.

In this paper, we propose Re-VoxelDet, which outperforms previous 3D detectors [11, 16, 28, 29], while maintaining fast inference time. Unlike pioneering works that rely on a single information flow between the backbone and neck, our approach boosts performance by leveraging rich spatial information integrated from various spatial features derived through hierarchical flows. Notably, our detector is constructed with a novel neck design that hierarchically merges the multi-scale voxel-wise features generated from each stage of 3D backbone with the RPN. Consequently, our detector focuses on more spatial-semantic fine-grained representation, leading to strong performance in both larger and smaller objects.

### 2.2. IoU-aware 3D Detection

Recently, the well-studied intersection over union (IoU) optimization techniques [22, 32] have shown promising performance in 2D object detection. Motivated by this success, there are many efforts on adopting these methods to the 3D domain [23, 31, 33]. Among these efforts, RDIoU [23] aims to improve the detection of rotated objects by jointly learning rotation decoupling parameters along with IoU. Furthermore, the concept of integrating an IoU estimation branch into the head was first proposed by IoUNet [15] in the context of 2D object detection. This idea of combining the IoU estimation branch with a multi-group head has been extensively explored for better 3D object detection [11, 13, 30, 34]. However, unfortunately, they are hardly studied to improve the accuracy of object using previous head modules. In this study, we devise a simple yet effective head module that consistently enhances the performance of objects with sensitive heading.

## 3. Methods

In this section, we explain a detailed overview of Re-VoxelDet, a novel voxel-based 3D object detector. To enhance the accuracy of predicting 3D bounding boxes from point clouds, Re-VoxelDet improves the entire 3D object detection framework, with a particular focus on the backbone, neck, and head modules. The overall architec-

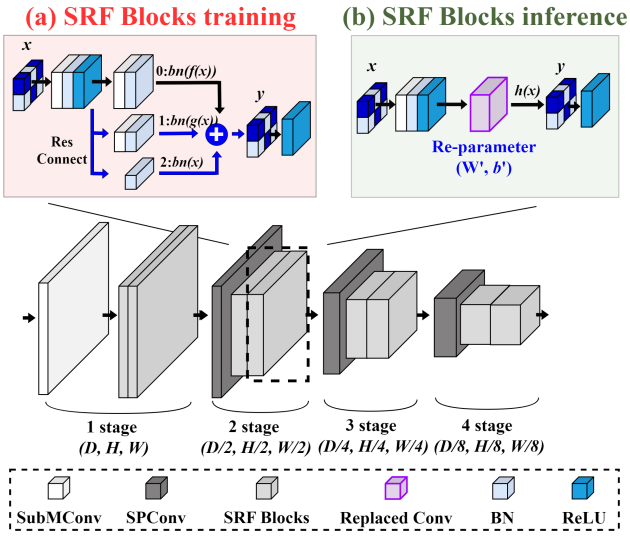


Figure 3. **Illustration of MVBackbone module.** It is composed of four stages. With the exception of the first stage, each stage consists of SPConv and SRF blocks. During training, the SRF blocks utilize three branches for encoding voxel features. These three branches are merged into a single branch through weight re-parameterization, during inference.

ture of Re-VoxelDet is illustrated in Fig. 2, which consists of MVBackbone (Sec. 3.1), HVANeck (Sec. 3.2), and RGHead module (Sec. 3.3). We elaborate on three modules in the subsequent sections.

### 3.1. Multi-view Voxel Backbone

To train an end-to-end detector on irregular and sparse point clouds, conventional voxel-based detectors commonly employ VoxelNet [35] as their backbone module. This backbone transforms point clouds into regular voxels, and the result is then passed to the subsequent stages. They utilize sparse residual convolutional blocks (SubM blocks) [11,29] with multiple submanifold sparse convolutional networks (SubMConv), and regular sparse convolutions (SPConv) for encoding. The conventional backbone modules focus solely on non-empty voxels to perform efficient operations, but they are composed of minimal channels and layers, thus having limited representation capacity. This is because the existing backbones only learn features from a single branch comprised of a few SubMConv operations.

In this research, we introduce a novel 3D backbone named MVBackbone, drawing inspiration from the design of ResNet-18 [12]. As depicted in Fig. 3, MVBackbone consists of four stages, each with output channels of $\{16, 32, 64, 128\}$ and strides of $\{1, 2, 3, 4\}$ respectively. With the exception of the first stage, each stage comprises a SPConv responsible for down-sampling the features, along with SRF blocks. Notably, SRF blocks with multiple branches enrich context information by integrating the features of each branch, in contrast to a single branch generated from previous SubM blocks. As these branches capture multiple features from different perspectives, our backbone improves overall detection performance by utilizing the multi-branch approach to incorporate the captured features. During training, given a voxel-wise feature $x$ as input, the process of SRF blocks begins. Specifically, three branches generate three different voxel-wise features $\{bn(f(x)),\ bn(g(x)),\ bn(x)\}$, with multiple views in parallel. Subsequently, three features are aggregated into a single result and it is then fed into the next stage as input (See Fig. 3(a)).

By utilizing the detector with the proposed SRF blocks, we effectively resolve the representation capacity issue of the backbone. Nevertheless, we inevitably encounter longer inference time by applying these blocks due to the increased model complexity. To deal with this problem, we leverage a structural re-parameterization method [7] to simplify the SRF blocks during inference (Refer Fig. 3(b) and Eq. (1)).

$$h(x) = (W_0\frac{\gamma_0}{\sigma_0} + W_1\frac{\gamma_1}{\sigma_1})^{\mathrm{T}} \times x + \sum_{i\in\{0,1,2\}} (\beta_i - \frac{\mu_i\gamma_i}{\sigma_i}). \quad (1)$$

In Eq. (1), $W$ and $\beta$ represent trained weight and bias, respectively. We combine them using a linear combination operation to define newly calculated $W'$ and $b'$. The subscripts 0, 1, and 2 mean different branches and are indicated by $i$. To be specific, branches 0 and 1 both undergo two steps: SubMConv and batch normalization (BN). Besides, a single branch 2 only goes through BN process. $\mu$, $\sigma$, and $\gamma$ indicate the mean, standard deviation, and scale factor, respectively.

### 3.2. Hierarchical Voxel-guided Auxiliary Neck

Our proposed 3D object detector aims to accurately predict 3D bounding boxes while minimizing the loss of information caused by down-sampling operations. To address this issue of information loss, we propose HVANeck, a module specifically designed to compensate for the spatial information that is lost. In comparison to existing neck modules [11, 29], our HVANeck module ultimately generates a unified feature $F_{final}$ by hierarchically combining BEV features reproduced from the voxel-wise features obtained by each stage of MVBackbone, along with the semantic features produced through RPN blocks. This unique neck module allows us to extract a strong integrated semantic-spatial feature. Consequently, our detector using this feature compensates for insufficient spatial information, especially, for smaller objects. Therefore, it achieves better detection performance (See Tab. 6 and Tab. 7). The overall process of the proposed HVANeck module is shown in Fig. 4. Given the input voxel-wise features $\{V_1,\ V_2\}$ obtained by MVBackbone, the voxel features are used for Depth-wise Sparse Max-Pooling (SPMaxPool) to encode them as BEV features '$B$'. This SPMaxPool operation is defined as follows:

$$B_{\bar{p}} = \mathrm{MaxPool}(\{v_{\bar{p}+x}\}_{x=0}^{K}), \text{where } v \in V. \quad (2)$$

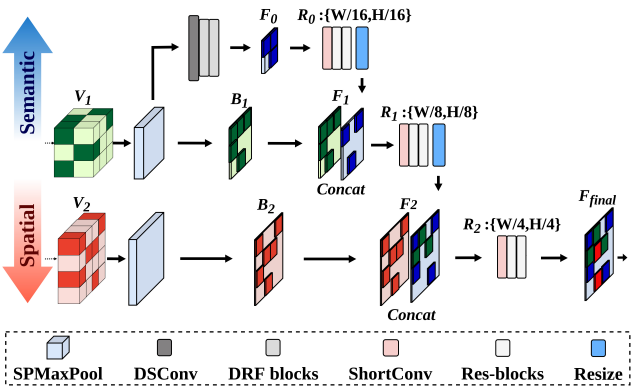


Figure 4. **Structure of HVANeck module.** It consists of SPMaxPool, DRF blocks, and RPN blocks. SPMaxPool is used to convert multi-scale voxel-wise features into the BEV features.

In Eq. (2), MaxPool $(\cdot)$ refers to the max pooling. The position of the non-empty voxels is denoted as $v_{\bar{p}}$, and $x$ means the 3D offset from $\bar{p}$. Therefore, $\bar{p} + x$ denotes the corresponding position based on the offset $x$ from $\bar{p}$. The 3D kernel space $K$, has a spatial shape as $(D, 3, 3)$, where $D$ is same as the depth size of $V$. The stride and padding parameters of SPMaxPool are set to (1, 1, 1) and (0, 1, 1), respectively. Unlike the pooling of 2D images, our SPMaxPool first creates the 3D kernel space based on non-empty voxels, considering the sparsity of point clouds. It calculates the maximum value among non-empty voxels within the kernel space. Following this process, we extract BEV features denoted as $\{B_1, B_2\}$ by encoding voxel features. After generating these BEV features, our neck module concatenates BEV features and the features produced by the RPN blocks $\{R_0, R_1, R_2\}$ to fuse semantic and spatial information from each feature. These features are integrated in a top-down manner as follows:

$$F_{i+1} = \begin{cases} RPN_i(F_i), & \text{if } i = 2 \\ Concat(B_{i+1}, Resize(RPN_i(F_i))). & \text{if } i < 2 \end{cases} \quad (3)$$

In Eq. (3), $RPN$ refers to the RPN blocks. These blocks comprise a single $1 \times 1$ convolutional network (ShortConv) and $3 \times 3$ convolutional network blocks equipped with a residual connection (Res-blocks). The feature $F_{i+1}$ is generated by fusing feature $F_i$ via the $i$-th RPN blocks and the BEV feature $B_{i+1}$ of the next level. To fuse the features of different sizes, we adopt an up-sampling operation denoted as $Resize$ for the results of the RPN blocks. This operation increases the resolution to match the size of the features $F_i$ and $B_{i+1}$. After resizing, the two features are integrated into a single feature using an element-wise concatenation operation ($Concat$). Following this step, the newly created $F_{i+1}$ is fed into RPN blocks to encode more abstract information. Exceptionally, $F_0$ is generated by a down-sampling convolutional network (DSConv) and Dense Reparameterized Feature blocks (DRF blocks). These blocks replace the 3D SubMConv of the aforementioned SRF blocks with 2D convolutional networks. Finally, our neck module produces the final output feature $F_{final}$ by combining finer spatial information from low-level voxel-wise features and further abstract information from high-level features.

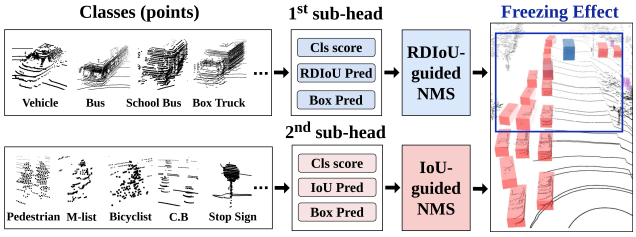


Figure 5. **Illustration of RGHead module.** It separates all classes into two sub-heads based on their aspect ratio and heading direction, thereby increasing accuracy and reducing the memory burden on the head. Note that, 'Vehicle' represents regular vehicle, 'C.B.' is construction barrel, and 'M-list' indicates motorcyclist.

### 3.3. Detection Head and Loss Function

**Rotation-based Group Head.** Unlike the previous multi-group head approach [36], our RGHead divides all classes into two sub-heads, considering their heading direction and aspect ratio (see Fig. 5). In the Argoverse2 dataset [26], the first sub-head groups common vehicle classes with considerable differences in aspect ratio, such as regular vehicles, buses, school buses, box trucks, etc. Meanwhile, the second sub-head groups the remaining classes. Therefore, our detector achieves robust performance in the first group, which is sensitive to heading. At the same time, it boosts computational efficiency by reducing the significant memory burden required by the head module.

Similar to commonly used LiDAR-based 3D object detection methods, the proposed RGHead adopts center-based approaches [11, 13, 29], which predict bounding boxes by leveraging the central locations of objects. Specifically, RGHead consists of classification, regression, and IoU estimation branches. Given the unified feature ($F_{final} \in \mathbb{R}^{C \times W \times H}$), the branches predict centerness heatmap, bounding box position, and 3D IoU score, respectively. Here, $W$, $H$, and $C$ represent the width, height, and channel of the final feature. Moreover, the detailed attributes of each bounding box include position, size, heading, and velocity vectors. In the IoU estimation branch, we propose a dual-IoU loss for both training and inference, capturing the characteristics of each group. Especially, to freeze the heading direction within a specific rotation like vehicle classes, the first sub-head is trained with RDIoU loss [23], which constrains the heading direction of objects. In contrast, the second sub-head is learned with IoU loss [33], which offers more flexibility in terms of heading. Thus, the training of

| Methods | mAP | CDS | Vehicle | Bus | Pedestrian | Box Truck | C.B. | M-list | MPC-Sign | Motorcycle | Bicycle | A.B. | School Bus | C.C. | V-Trailer | Bollard | Sign | Large Veh. | Stop Sign | Stroller | Bicyclist |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CenterPoint ∗ [29] | 22.0 | 17.6 | 67.6 | 38.9 | 46.5 | 40.1 | 32.2 | 28.6 | 27.4 | 33.4 | 24.5 | 8.70 | 22.6 | 29.5 | 22.4 | 37.4 | 6.30 | 3.90 | 16.9 | 0.50 | 20.1 |
| FSD [8] | 28.2 | 22.7 | 68.1 | 40.9 | 59.0 | 38.5 | 42.6 | 39.7 | 26.2 | 49.0 | 38.6 | 20.4 | 14.8 | 41.2 | 26.9 | 41.8 | 11.9 | 5.90 | 29.0 | 13.8 | 33.4 |
| CenterPoint ‡ [29] | 29.5 | 22.1 | 72.2 | 38.0 | 59.4 | 37.0 | 56.9 | 46.2 | 37.7 | 46.4 | 42.0 | 20.2 | 23.1 | 41.3 | 22.7 | 50.5 | 14.1 | 4.8 | 34.4 | 13.6 | 33.7 |
| PillarNet-34 ‡ [11] | 29.9 | 22.4 | 72.2 | 42.0 | 61.5 | 41.0 | 67.3 | 48.2 | 33.2 | 47.2 | 37.8 | 19.7 | 28.7 | 39.3 | 21.5 | 49.8 | 10.7 | 5.7 | 34.4 | 11.6 | 32.2 |
| VoxelNeXt [5] | 30.5 | 23.0 | 72.0 | 39.7 | 63.2 | 39.7 | 64.5 | 46.0 | 34.8 | 44.9 | 40.7 | 21.0 | 18.4 | 45.7 | 22.2 | 53.7 | 15.6 | **7.30** | 40.1 | 15.7 | 32.4 |
| Re-VoxelDet | 33.6 | 26.0 | 76.8 | 44.7 | 69.1 | 39.2 | 65.1 | 47.3 | 52.5 | 52.0 | 40.6 | 23.9 | 34.8 | 44.3 | 27.5 | 57.2 | 17.4 | 6.20 | 44.4 | 18.9 | 37.8 |
| **Re-VoxelDet †** | **38.2** | **29.4** | **78.4** | **46.4** | **73.5** | **42.5** | **71.9** | **59.4** | **60.1** | **59.4** | **45.0** | **28.0** | **41.8** | **50.7** | **31.9** | **61.9** | **20.8** | 6.70 | **49.6** | **23.6** | **41.6** |

Table 1. Comparison with existing LiDAR-based methods (without camera) on Argoverse2 validation split. Note that, 'Vehicle' is regular vehicle, and 'C.B.' is denoted as construction barrel. 'M-list' indicates motorcyclist. 'MPC-Sign' represents mobile pedestrian crossing sign, 'A.B.' denotes articulated bus, and 'C.C.' is construction cone. 'V-Trailer' represents vehicular trailer and 'Large Veh.' means large vehicle. ∗ is re-implemented by FSD, and ‡ is re-implemented by ourselves. † means test-time augmentation.

IoU estimation branch is supervised by RGH loss:

$$L_{rgh} = \begin{cases} L_1(RDIoU(b_{pred}, b_{gt}), p), & \text{if } k = 0 \\ L_1(IoU(b_{pred}, b_{gt}), p), & \text{if } k = 1 \end{cases} \quad (4)$$

where $k$ is the order of sub-heads. If $k$=0, it refers to the first sub-head, and if $k$=1, it corresponds to the second sub-head. Each IoU estimation branch is trained with $L_1$ loss, where $p$ represents the IoU prediction score. $b_{pred}$ and $b_{gt}$ are the predicted and ground truth (GT) box, respectively. $RDIoU(\cdot)$ and $IoU(\cdot)$ are the real 3D RDIoU and 3D IoU between the predicted and GT box, respectively.

During inference, to better detect objects with large aspect ratios and heading sensitivities like vehicle classes, we employ a combined score, which includes both the classification score and the predicted IoU score generated by the IoU estimation branch. By leveraging a rotation-aware IoU score, our detector constrains the heading direction within a specific range to ensure accurate heading predictions. This approach significantly enhances the detection accuracy, especially for challenging objects to predict their heading due to the lack of sufficient points or being far away.

**Loss Function.** During the training phase, we apply the conventional settings [11, 29] to optimize our proposed Re-VoxelDet. The overall loss function is defined as:

$$L = \alpha \cdot L_{cls} + \beta \cdot L_{reg} + \gamma \cdot (L_{rgh} + L_{diou}), \quad (5)$$

where $\alpha$, $\beta$, and $\gamma$ are the weight parameters designed to balance the proportion of different losses. For the classification branch, we use a penalty-minimized focal loss ($L_{cls}$) following CenterPoint [17, 29]. In addition, we employ $L_1$ loss ($L_{reg}$) and RGH loss ($L_{rgh}$), for the regression and IoU estimation branch. Moreover, DIoU loss ($L_{diou}$) [32] is used for further performance.

| Team Name | Rank | mAP | CDS |
|---|---|---|---|
| Le3DE2E (public) | 1 | 48.0 | 39.0 |
| BEV (public) | 2 | 46.0 | 37.0 |
| Detectors (public) | 3 | 41.0 | 34.0 |
| **Re-VoxelDet (public)** | 4 | **39.3** | **30.8** |
| Fengf (private) | 5 | 33.0 | 26.0 |
| Match (private) | 6 | 26.0 | 21.0 |
| Baseline (private) | 7 | 18.0 | 14.0 |

Table 2. The leaderboard results for 3D object detection on Argoverse2 test split.

## 4. Experiments

To verify the superiority of our proposed model, we conduct extensive experiments to compare it with previous models on three widely used large-scale datasets for autonomous driving: Argoverse2 [26], Waymo Open Dataset (WOD) [25], and nuScenes [2].

### 4.1. Dataset

**Argoverse2.** This dataset contains point cloud data collected from two 32-channel LiDARs and encompasses 30 different categories within a large perception range of 400m $\times$ 400m. The dataset is divided into 700, 150, and 150 sequences for training, validation, and testing, respectively. The primary metrics used for evaluation are distance-based 3D mean average precision (mAP) and the composite detection score (CDS).

**WOD.** This dataset is composed of a total of 1,150 sequences, divided into 798 for training, 202 for validation, and 150 for testing. The data is collected from a well-synchronized 64-channel LiDAR and 5 cameras, covering an area of 150m $\times$ 150m. For evaluation, the metrics target three primary classes (i.e., vehicles, pedestrians, and cyclists). These metrics involve the IoU-based mAP and mean average precision with heading (mAPH), which is weighted

| Methods | Reference | mAP / mAPH | | Vehicle AP / APH | | Pedestrian AP / APH | | Cyclist AP / APH | |
|---|---|---|---|---|---|---|---|---|---|
| | | L1 | L2 | L1 | L2 | L1 | L2 | L1 | L2 |
| PointPillars [16] | CVPR'19 | - / - | - / - | 68.6 / 68.1 | 60.5 / 60.1 | 68.0 / 55.5 | 61.4 / 50.1 | - / - | - / - |
| CenterPoint [29] | CVPR'21 | - / - | - / - | 80.2 / 79.7 | 72.2 / 71.8 | 78.3 / 72.1 | 72.2 / 66.4 | - / - | - / - |
| PillarNet-34 [11] | ECCV'22 | 77.5 / 74.7 | 72.1 / 69.6 | **82.5 / 82.0** | **75.1 / 74.7** | 80.8 / 74.1 | 74.8 / 68.5 | 69.1 / 67.9 | 66.6 / 65.5 |
| AFDetV2 [13] | AAAI'22 | 77.5 / 75.3 | 72.2 / 70.0 | 80.5 / 80.4 | 73.0 / 72.6 | 79.8 / 74.4 | 73.7 / 68.6 | 72.4 / 71.2 | 69.8 / 68.7 |
| PV-RCNN ++ [24] | IJCV'23 | 78.0 / 75.7 | 72.4 / 70.2 | 81.6 / 81.2 | 73.9 / 73.5 | 80.4 / 75.0 | 74.1 / 69.0 | 71.9 / 70.8 | 69.3 / 68.2 |
| **Re-VoxelDet** | **Ours** | **78.7 / 76.3** | **73.3 / 71.0** | 81.4 / 80.9 | 73.8 / 73.3 | **81.3 / 76.0** | **75.3 / 70.2** | **73.3 / 72.1** | **70.7 / 69.5** |

Table 3. The LiDAR-only non-ensemble performance comparisons between Re-VoxelDet and other state-of-the-art methods on WOD test split. All detectors listed, take single-frame point clouds as input. The evaluation metrics are divided into L1 (Level 1) and L2 (Level 2) according to number of points within an object.

| Methods | Reference | mAP | NDS |
|---|---|---|---|
| CBGS [36] | arXiv'19 | 51.4 | 62.6 |
| CenterPoint [29] | CVPR'21 | 59.0 | 66.4 |
| PillarNet-18 [11] | ECCV'22 | 59.9 | 67.4 |
| VoxelNeXt [5] | CVPR'23 | 60.0 | 67.1 |
| TransFusion-L [1] | CVPR'22 | 60.0 | 66.8 |
| Focals Conv [3] | CVPR'22 | 61.2 | 68.1 |
| LargeKernel3D [4] | CVPR'23 | 63.3 | 69.1 |
| LinK [19] | CVPR'23 | 63.3 | 69.5 |
| **Re-VoxelDet** | **Ours** | **64.6** | **70.5** |

Table 4. The LiDAR-only performance comparison with other state-of-the-art methods on nuScenes validation split.

by heading accuracy.

**nuScenes.** This dataset consists of 10 categories and 1,000 driving scenarios. It includes 700 scenarios for training, 150 for validation, and 150 for testing. Each scenario contains point clouds collected from a 32-channel LiDAR operating at 20Hz, and 3D annotation data at 2Hz. For 3D object detection task, the official evaluation metrics are the distance-based mAP, and the nuScenes detection score (NDS).

### 4.2. Implementation Details

For the WOD dataset, our model is trained for 30 epochs with a maximum learning rate of 0.003. The detection range is set to [-75.2m, 75.2m] for the $X$ and $Y$ axes, and [-4m, 2m] for the $Z$ axis. On the nuScenes dataset, we set the detection range to [-54m, 54m] for the $X$ and $Y$ axes, and [-5m, 3m] for the $Z$ axis, and the model is trained for 20 epochs. For the Argoverse2 dataset, we define the detection ranges differently for the validation and test splits. On the validation split, the detection range is set to [-200m, 200m] for $X$ and $Y$ axes, while on the test split, the detection range is set to [-150m, 150m] for $X$ and $Y$ axes. $Z$ axis is defined as [-3m, 3m] for both the validation and test splits. All models are trained using the Adam optimizer with 5 RTX 3090 GPUs, while inference is performed on a single RTX 3090 GPU.

### 4.3. Comparison with State-of-the-art Methods

**Argoverse2 Results.** We compare Re-VoxelDet with other existing methods on the Argoverse2 validation and test splits. As presented in Tab. 1, our Re-VoxelDet achieves 33.6 mAP and 26.0 CDS, surpassing previous state-of-the-art methods without relying on ensemble techniques or test-time augmentation (TTA). Notably, when detecting smaller objects such as pedestrians, stop signs, and bicyclists, our method brings tremendous performance gains (4.3-5.9 AP) over the most recent work, VoxelNeXt [5]. By incorporating TTA, Re-VoxelDet further enhances its performance, achieving 38.2 mAP and 29.4 CDS. We suppose that these remarkable results are due to our proposed neck modules, which are designed to more effectively compensate for the loss of spatial information, especially for smaller objects. Furthermore, as indicated in Tab. 2, Re-VoxelDet ranks fourth, reaching 39.3 mAP and 30.8 CDS on the public Argoverse2 detection leaderboard.

**WOD Results.** Tab. 3 shows the performance comparisons between our method and other LiDAR-only non-ensemble techniques on the WOD test split. Our method achieves 78.7 mAP and 76.3 mAPH on L1 difficulties, and 73.3 mAP and 71.0 mAPH on L2 difficulties. Notably, regarding category-specific performance, our model outperforms PV-RCNN++ [24] for pedestrians and cyclists.

**nuScenes Results.** We also compare Re-VoxelDet with other 3D object detection models on the nuScenes validation split. As shown in Tab. 4, our Re-VoxelDet achieves 64.6 mAP and 70.5 NDS, which are 1.3 AP and 1.0 NDS higher than the previous state-of-the-art, LinK [19]. These substantial improvements demonstrate the effectiveness of our approach.

### 4.4. Analysis

**Component-wise Analysis.** In Tab. 5, we separately analyze the effectiveness of each proposed module by integrating the backbone, neck, and head into the baseline model, CenterPoint [29]. Firstly, when replacing the backbone with our MVBackbone, we observe a slight increase in accuracy, with gains of 0.2 mAP and 0.1 CDS (see row 2). Furthermore, when combining MVBackbone and HVANeck modules, the model achieves an improvement of 1.2 mAP and 1.1 CDS (see row 3). This indicates that our proposed neck module is more effective by capturing fine-grained spatial-

| Methods | Components | | | mAP | CDS | Veh. | A-Bus | S.B. |
|---|---|---|---|---|---|---|---|---|
| | MVB. | HVA. | RGH. | | | | | |
| Baseline | | | | 29.3 | 22.0 | 72.2 | 20.2 | 23.1 |
| Re-VoxelDet | ✓ | | | 29.5 | 22.1 | 72.1 | 18.0 | 23.4 |
| Re-VoxelDet | ✓ | ✓ | | 30.7 | 23.2 | 73.3 | 20.9 | 25.3 |
| Re-VoxelDet | ✓ | ✓ | ✓ | **33.6** | **26.0** | **76.8** | **23.9** | **34.8** |

Table 5. The analysis study on the effects of each component in Re-VoxelDet. The last row represents the results achieved by utilizing MVBackbone (MVB.), HVANeck (HVA.), and RGHead (RGH.). Here, 'Veh.', 'A-Bus', and 'S.B.' denote regular vehicle, articulated bus, and school bus, respectively. All evaluations are conducted on Argoverse2 validation split.

| Methods | Neck | mAP | CDS | Ped. | M-list | Str. | S.S. |
|---|---|---|---|---|---|---|---|
| CenterPoint ‡ | CenterPoint | 29.3 | 22.0 | 59.4 | 46.2 | 13.6 | 34.4 |
| | **HVANeck** | **30.7** | **23.2** | **61.7** | **47.9** | **15.6** | **37.2** |
| PillarNet-34 ‡ | PillarNet | 29.9 | 22.4 | **61.5** | 48.2 | 11.6 | 34.4 |
| | **HVANeck** | **31.0** | **23.3** | 61.1 | **50.0** | **12.7** | **37.2** |

Table 6. The analysis study on the effects of different neck modules with various detectors. To fair comparison, we only switch the neck module, while other components such as backbone and head are same. Note that, 'Ped.', 'M-list', 'Str.', and 'S.S.' mean pedestrian, motorcyclist, stroller, and stop sign, respectively. ‡ is re-implemented by ourselves. All evaluations are conducted on the Argoverse2 validation split.

| Methods | Neck | mAP | CDS | Ped. | M-list | Str. | S.S. |
|---|---|---|---|---|---|---|---|
| Re-VoxelDet | CenterPoint | 31.9 | 23.8 | 68.3 | 53.1 | 6.7 | 39.3 |
| | PillarNet | 33.4 | 25.0 | **69.5** | 52.8 | 11.7 | 40.9 |
| | **HVANeck** | **33.6** | **26.0** | 69.1 | **54.3** | **13.9** | **41.3** |

Table 7. The analysis study on the effects of different neck modules with same detector. All modules except for the neck utilize the components from Re-VoxelDet, including the MVBackbone and RGHead. All evaluations are conducted on Argoverse2 validation split. Note that, 'Ped.', 'M-list', 'Str.', and 'S.S.' follow the same definitions as in Tab. 6.

| Dataset | Range | Methods | Runtime (head) |
|---|---|---|---|
| nuScenes | [-54m, 54m] | PillarNet-18 ‡ | 126 (47)ms |
| | | **Re-VoxelDet** | **81 (23)ms** |
| Argoverse2 | [-75m, 75m] | PillarNet-18 ‡ | 122 (48)ms |
| | | **Re-VoxelDet** | **79 (25)ms** |

Table 8. Runtime analysis of different detectors on Argoverse2 and nuScenes validation split. ‡ is re-implemented by ourselves. 'head' denotes the runtime of head module.

semantic features than the baseline neck. Lastly, with the integration of RGHead module (see row 4), the detection accuracy further improves to 33.6 mAP and 26.0 CDS. From this observation, the RGHead module is generally robust in detecting extreme-aspect ratio categories such as regular vehicles, articulated buses, and school buses, resulting in increases of up to 9.5 AP.

**Neck Analysis.** To verify the advantages of our newly proposed HVANeck module, we conduct two experiments to compare our HVANeck module with the existing neck modules in CenterPoint and PillarNet [11]. As shown in Tab. 6, we only switch the neck module while keeping all other modules. The results reveal that our neck module substantially enhances detection accuracy, especially for smaller and more challenging objects such as pedestrians, motorcyclists, strollers, and stop signs. This suggests that a hierarchical merging process in our neck module more effectively reduces the loss of spatial information than other existing neck modules. We further investigate the impact of our neck module when paired with Re-VoxelDet, which consists of MVBackbone and RGHead. As shown in Tab. 7, our HVANeck displays performance improvements of 1.7 mAP and 2.2 CDS over CenterPoint neck, and 0.2 mAP and 1.0 CDS over PillarNet neck. These experiments in Tab. 6 and Tab. 7 demonstrate the importance of meticulously designing neck modules for improved detection performance.

**Runtime Analysis.** Tab. 8 shows the runtime comparisons between our proposed Re-VoxelDet and PillarNet on the Argoverse2 and nuScenes datasets. Re-VoxelDet consistently achieves runtimes of 81ms and 79ms on nuScenes and Argoverse2 datasets, respectively. It operates on average 35% faster than PillarNet. This result demonstrates that our model can achieve real-time performance while delivering superior accuracy.

## 5. Conclusion

In this paper, we have presented Re-VoxelDet, a novel voxel-based 3D object detection framework. This framework consists of three key components: MVBackbone, HVANeck, and RGHead. MVBackbone is constructed with SRF blocks to generate multiple voxel features from diverse viewpoints without additional computational costs. HVANeck is meticulously designed to extract powerful spatial-semantic features by utilizing various spatial feature details obtained from hierarchical connection processes. It efficiently compensates for the lost spatial information as well as the strong semantic information. Finally, RGHead not only significantly reduces inference time but also ensures robust detection performance, considering both the aspect ratio and heading sensitivity. Our framework achieves state-of-the-art performance by delivering a favorable trade-off between the computational efficiency and the accuracy improvement on three large-scale datasets.

**Acknowledgement** This work was supported by the DGIST R&D Program of the Ministry of Science and ICT (23-IT-02), and the Technology Commercialization Capacity Building Project of the Ministry of Science and ICT (2023-DG-RD-0041-01).

## References


[1] Xuyang Bai, Zeyu Hu, Xinge Zhu, Qingqiu Huang, Yilun Chen, Hongbo Fu, and Chiew-Lan Tai. Transfusion: Robust lidar-camera fusion for 3d object detection with transformers. In *CVPR*, pages 1090–1099, 2022. 7

[2] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, pages 11621–11631, 2020. 2, 6

[3] Yukang Chen, Yanwei Li, Xiangyu Zhang, Jian Sun, and Jiaya Jia. Focal sparse convolutional networks for 3d object detection. In *CVPR*, pages 5428–5437, 2022. 7

[4] Yukang Chen, Jianhui Liu, Xiangyu Zhang, Xiaojuan Qi, and Jiaya Jia. Largekernel3d: Scaling up kernels in 3d sparse cnns. In *CVPR*, pages 13488–13498, 2023. 3, 7

[5] Yukang Chen, Jianhui Liu, Xiangyu Zhang, Xiaojuan Qi, and Jiaya Jia. Voxelnext: Fully sparse voxelnet for 3d object detection and tracking. In *CVPR*, pages 21674–21683, 2023. 2, 3, 6, 7

[6] 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*, pages 1201–1209, 2022. 2

[7] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *CVPR*, pages 13733–13742, 2021. 4

[8] Lue Fan, Feng Wang, Naiyan Wang, and ZHAO-XIANG ZHANG. Fully sparse 3d object detection. In *NeurIPS*, pages 351–363, 2022. 6

[9] Benjamin Graham. Sparse 3d convolutional neural networks. In *BMVC*, 2015. 1, 2

[10] Benjamin Graham, Martin Engelcke, and Laurens Van Der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *CVPR*, pages 9224–9232, 2018. 1, 2

[11] Chao Ma Guangsheng Shi, Ruifeng Li. Pillarnet: Real-time and high-performance pillar-based 3d object detection. In *ECCV*, pages 35–52, 2022. 1, 2, 3, 4, 5, 6, 7, 8

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2015. 4

[13] Yihan Hu, Zhuangzhuang Ding, Runzhou Ge, Wenxin Shao, Li Huang, Kun Li, and Qiang Liu. Afdetv2: Rethinking the necessity of the second stage for object detection from point clouds. In *AAAI*, pages 969–979, 2022. 1, 2, 3, 5, 7

[14] Dihe Huang, Ying Chen, Yikang Ding, Jinli Liao, Jianlin Liu, Kai Wu, Qiang Nie, Yong Liu, and Chengjie Wang. Rethinking dimensionality reduction in grid-based 3d object detection. *arXiv preprint arXiv:2209.09464*, 2022. 1, 2

[15] Borui Jiang, Ruixuan Luo, Jiayuan Mao, Tete Xiao, and Yuning Jiang. Acquisition of localization confidence for accurate object detection. In *ECCV*, pages 784–799, 2018. 3

[16] 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, 3, 7

[17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017. 6

[18] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *CVPR*, pages 11976–11986, 2022. 2

[19] Tao Lu, Xiang Ding, Haisong Liu, Gangshan Wu, and Limin Wang. Link: Linear kernel for lidar-based 3d perception. In *CVPR*, pages 1105–1115, 2023. 3, 7

[20] 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 *CVPR*, pages 2723–2732, 2021. 2

[21] 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

[22] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *CVPR*, pages 658–666, 2019. 3

[23] Hualian Sheng, Sijia Cai, Na Zhao, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, Min-Jian Zhao, and Gim Hee Lee. Rethinking iou-based optimization for single-stage 3d object detection. In *ECCV*, pages 544–561, 2022. 3, 5

[24] 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. *IJCV*, 131(2):531–551, 2023. 7

[25] 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. 2, 6

[26] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting, 2021. Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks 2021. 2, 5, 6

[27] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *CVPR*, pages 16133–16142, 2023. 2

[28] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 1, 2, 3

[29] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *CVPR*, pages 11784–11793, 2021. 1, 2, 3, 4, 5, 6, 7

[30] 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*, pages 3555–3562, 2021. 1, 3

[31] Wu Zheng, Weiliang Tang, Li Jiang, and Chi-Wing Fu. Se-ssd: Self-ensembling single-stage object detector from point cloud. In *CVPR*, pages 14494–14503, 2021. 3

[32] Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression. In *CVPR*, pages 12993–13000, 2020. 3, 6

[33] 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. 3, 5

[34] Sifan Zhou, Zhi Tian, Xiangxiang Chu, Xinyu Zhang, Bo Zhang, Xiaobo Lu, Chengjian Feng, Zequn Jie, Patrick Yin Chiang, and Lin Ma. Fastpillars: A deployment-friendly pillar-based 3d detector. *arXiv preprint arXiv:2302.02367*, 2023. 2, 3

[35] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, pages 4490–4499, 2018. 1, 2, 4

[36] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced grouping and sampling for point cloud 3d object detection. *arXiv preprint arXiv:1908.09492*, 2019. 2, 5, 7