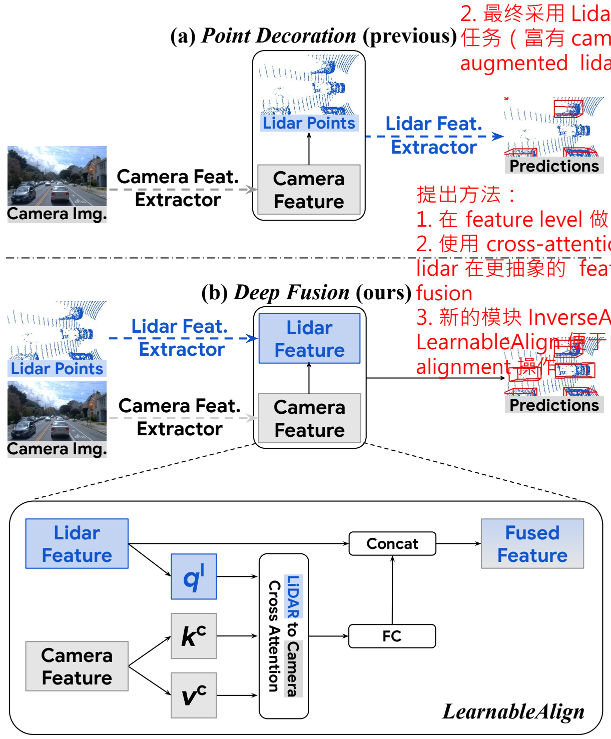
DeepFusion: Lidar-Camera Deep Fusion for Multi-Modal 3D Object Detection

Yingwei Li[1,2*] Adams Wei Yu[2*] Tianjian Meng[2] Ben Caine[2] Jiquan Ngiam[2] Daiyi Peng[2]
Junyang Shen[2] Yifeng Lu[2] Denny Zhou[2] Quoc V. Le[2] Alan Yuille[1] Mingxing Tan[2]
[1]Johns Hopkins University [2]Google
{ywli, adamsyuwei, tanmingxing}@google.com

Abstract

*Lidars and cameras are critical sensors that provide complementary information for 3D detection in autonomous driving. While prevalent multi-modal methods [34, 36] simply decorate raw lidar point clouds with camera features and feed them directly to existing 3D detection models, our study shows that fusing camera features with deep lidar features instead of raw points, can lead to better performance. However, as those features are often augmented and aggregated, a key challenge in fusion is how to effectively align the transformed features from two modalities. In this paper, we propose two novel techniques: **InverseAug** that inverses geometric-related augmentations, e.g., rotation, to enable accurate geometric alignment between lidar points and image pixels, and **LearnableAlign** that leverages cross-attention to dynamically capture the correlations between image and lidar features during fusion. Based on InverseAug and LearnableAlign, we develop a family of generic multi-modal 3D detection models named **DeepFusion**, which is more accurate than previous methods. For example, DeepFusion improves PointPillars, CenterPoint, and 3D-MAN baselines on Pedestrian detection for 6.7, 8.9, and 6.2 LEVEL_2 APH, respectively. Notably, our models achieve state-of-the-art performance on Waymo Open Dataset, and show strong model robustness against input corruptions and out-of-distribution data. Code will be publicly available at* `https://github.com/tensorflow/lingvo`.



原有方法：
1. 只在 input level 对于两者做 fusion
2. 最终采用 Lidar detector 来做下游任务（富有 camera 特点的 augmented lidar features）

提出方法：
1. 在 feature level 做 fusion
2. 使用 cross-attention 对于 camera 和 lidar 在更抽象的 feature-level 上做 fusion
3. 新的模块 InverseAug 和 LearnableAlign 便于自定义的 feature alignment 操作

Figure 1. Our method fuses two modalities on **deep feature level**, while previous state-of-the-art methods (PointPainting [34] and PointAugmenting [36] as examples) decorate lidar points with camera features on **input level**. To address the modality alignment issue (see Section 1) for deep feature fusion, we propose two techniques **InverseAug** (see Figure 2 and 3) and **LearnableAlign**, a cross-attention-based feature-level alignment technique.

1. Introduction

Lidars and cameras are two types of complementary sensors for autonomous driving. For 3D object detection, lidars provide low-resolution shape and depth information, while cameras provide high-resolution shape and texture information. While one would expect the combination of both sensors to provide the best 3D object detector, it turns out that most state-of-the-art 3D object detectors use only lidar as the input (Waymo Challenge Leaderboard, accessed on Oct 14, 2021). This indicates that how to effectively fuse the signals from these two sensors still remain challenging. In this paper, we strive to provide a generic and effective solution to this problem.

Existing approaches in the literature for fusing lidars and cameras broadly follow two approaches (Figure 1): they either fuse the features at an early stage, such as by decorating points in the lidar point cloud with the corresponding camera features [34, 36], or they use a mid-level fusion where

[*] Equal contribution. Work done when YL was an intern at Google.

1. 总结了现有的 feature fusion 方法：input-level 和 mid-level，两者都比较暴力直接，缺乏逻辑上的 justification
2. 在 input-level 提出了 InverseAug 可以支持数据增广下的 fusion
3. 在 deep-level 提出了 LearnableAlign 可以利用
4. 都很 cost-efficient with marginal computation
5. 指出成功的关键在于 effective feature alignment
6. 涨点的主要贡献在于 long-range objects（有待实验观察研究）

the features are combined after feature extraction [5, 14, 18]. One of the biggest challenges in both kinds of approaches is to figure out the correspondence between the lidar and camera features. To tackle this issue, we propose two methods: *InverseAug* and *LearnableAlign* to enable effective mid-level fusion. **InverseAug** inverses geometric-related data augmentations (*e.g.*, RandomRotation [46]) and then uses the original camera and lidar parameters to associate the two modalities. **LearnableAlign** leverages cross-attention to dynamically learn the correlation between a lidar feature and its corresponding camera features. These two proposed techniques are simple, generic, and efficient. Given a popular 3D point cloud detection framework, such as PointPillars [17] and CenterPoint [44], InverseAug and LearnableAlign help the camera images effectively align with lidar point cloud with marginal computational cost (*i.e.*, only one cross-attention layer). When fusing the aligned multi-modal features, the camera signals, with much higher resolution, significantly improve the model's recognition and localization ability. These advantages are especially beneficial for the long-range object detection.

We develop a family of multi-modal 3D detection models named **DeepFusions**, which offer the advantage that they (1) can be trained end-to-end and (2) are generic building blocks compatible with many existing voxel-based 3D detection methods. DeepFusion serves as a plug-in that can be easily applied to most of the voxel-based 3D detection methods, such as PointPillars [17] and CenterPoint [44].

Our extensive experiments demonstrate that (1) effective deep feature alignment is the key for multi-modal 3D object detection, (2) by improving alignment quality with our proposed InverseAug and LearnableAlign, DeepFusion significantly improves the detection accuracy, and (3) compared with its single-modal baseline, DeepFusion is more robust against input corruptions and out-of-distribution data.

On the Waymo Open Dataset, DeepFusion improves several prevalent 3D detection models such as PointPillars [17], CenterPoints [44], and 3D-MAN [43] by 6.7, 8.9, and 6.2 LEVEL_2 APH, respectively. We achieve state-of-the-art results on Waymo Open Dataset that DeepFusion improves 7.4 Pedestrian LEVEL_2 APH over PointAugmenting [36], the previous best multi-modal method, on the validation set. This result shows that our method is able to effectively combine the lidar and camera modalities, where the largest improvements come from the recognition and localization for *long-range* objects.

Our contributions can be summarized as three folds:

- To our best knowledge, we are the first to systematically study the influence of deep feature alignment for 3D multi-modality detectors;
- We propose InverseAug and LearnableAlign to achieve deep-feature-level alignment, leading to accurate and robust 3D object detector;
- Our proposed models, DeepFusions, achieve state-of-the-art performance on Waymo Open Dataset.

2. Related Work

3D Object Detection on Point Clouds. Lidar point clouds are often represented as unordered set, and many 3D object detection methods tend to directly deal with such raw unordered points. PointNet [25] and PointNet++ [26] are early seminal works that directly apply neural networks on point cloud. Following them, [22, 24, 31, 42] also learn features with PointNet-like [25] layers. Lidar point clouds can be also represented as dense range images, where each pixel contains extra depth information. [1, 19] directly work on the range images to predict 3D bounding boxes.

Another set of 3D detection methods convert lidar points to voxels or pillars, leading to two more commonly used 3D detection methods: voxel-based and pillar-based method. VoxelNet [46] proposes a voxel-based approach, which discretizes the point-cloud into a 3D grid with each sub-space is called a voxel. A dense 3D convlutional network can then be applied to this grid to learn detection features. SECOND [40] builds on VoxelNet and proposes using sparse 3D convolutions to increase efficiency. Since 3D voxels are often expensive to process, PointPillars [17] and PIXOR [41] further simplify 3D voxels to bird-eye-view 2D pillars, where all voxels with the same z-axis are collapsed to a single pillar. These 2D pillars can then be processed with existing 2D convolutional detection networks to produce the bird-eye-view bounding boxes. Since 2D pillars are usually easy and fast to process, many recent 3D detection methods [34, 38, 43, 44] are built upon PointPillars. In this paper, we also choose PointPillar as our baseline approach for dealing with lidar point clouds.

Lidar-camera Fusion. Instead of relying on lidar point cloud, monocular detection approaches directly predict 3D boxes from 2D images [3, 16, 27]. A key challenge for these approaches is 2D images do not have depth information, so most monocular detectors need to implicitly or explicitly predict depth for each 2D image pixel, which is often another very difficult task. Recently, there is a trend to combine lidar and camera data to improve 3D detection. Some approaches [24, 39] first detect objects in 2D images then use the information to further process the point cloud. Previous works [4, 15] also use a two-stage framework to perform object-centric modality fusion. In contrast to these methods, our approach is easier to plug-in into most existing voxel-based 3D detection methods.

Point Decoration Fusion. PointPainting [34] proposes to augment each lidar point with the semantic scores of camera images, which are extracted with a pre-trained semantic segmentation network. PointAugmenting [36] points out the limitation of semantic scores, and proposes to augment lidar points with the deep features extracted from a 2D object detection network on top of camera images. As shown in Figure 1 (a), those methods rely on a pretrained module (*e.g*., 2D detection or segmentation model) to extract the features from the camera images, which are used to decorate the raw point clouds and then fed into a lidar feature voxelizer to construct the bird-eye view pseudo images.

Mid-level Fusion. Deep Continuous Fusion [18], EP-Net [14] and 4D-Net [23] attempt to fuse the two modalities by sharing the information between 2D and 3D backbones. However, an important missing piece in those works is an effective alignment mechanism between camera and lidar features, which is confirmed in our experiments to be the key for building an effective end-to-end multi-modal 3D object detector. Even knowing the importance of effective alignment, we point out it is challenging to do so for the following reasons. First, to achieve the best performance on existing benchmarks, *e.g*., Waymo Open Dataset, various data augmentation strategies are applied to lidar points and camera images before the fusion stage. For example, RandomRotation [46] that rotates the 3D world along z-axis, is usually applied to lidar points but not applicable to camera images, making it difficult for the subsequent feature alignment. Second, since multiple lidar points are aggregated into the same 3D cube, i.e., *voxel*, in the scene, one voxel corresponds to a number of camera features, and these camera features are not equally important for 3D detection.

3. DeepFusion

In Section 3.1, we first introduce our deep feature fusion pipeline. Then, we conduct a set of preliminary experiments to quantitatively illustrate the importance of alignment for deep feature fusion in Section 3.2. Finally, we propose two techniques, InverseAug and LearnableAlign, to improve the alignment quality in Section 3.3.

3.1. Deep Feature Fusion Pipeline

As shown in Figure 1 (a), previous methods, such as PointPainting [34] and PointAugmenting [36], usually use an extra well-trained detection or segmentation model as camera feature extractor. For example, PointPainting uses Deeplabv3+[1] to generate per-pixel segmentation labels as camera features [34]. Then, the raw lidar points are decorated with the extracted camera features. Finally, the camera-feature-decorated lidar points are fed into a 3D point cloud object detection framework.

The pipeline above is improvable due to following reasons. First, the camera features are fed into several modules that are specially designed for processing point cloud data. For example, if PointPillars [17] is adopted as the 3D detection framework, the camera features need to be voxelized together with the raw point clouds to construct bird's eye view pseudo images. However, the voxelization module is not designed for processing camera information. Second, the camera feature extractor is learned from other independent tasks (*i.e*., 2D detection or segmentation), which may lead to (1) domain gap, (2) annotation efforts, (3) additional computation cost, and more importantly, (4) sub-optimal extracted features because the features are heuristically chosen rather than learned in an end-to-end manner.

To tackle the above two issues, we propose a deep feature fusion pipeline. To address the first problem, we fuse deep camera and lidar features instead of decorating raw lidar points at the input level so that the camera signals do not go through the modules designed for point cloud. For the second problem, we use convolution layers to extract camera features and train these convolution layers together with other components of the network in an end-to-end manner. To summarize, our proposed deep feature fusion pipeline is shown in Figure 1 (b): the lidar point clouds are fed into an existing lidar feature extractor (*e.g*., Pillar Feature Net from PointPillars [17]) to obtain lidar feature (*e.g*., the pseudo image from PointPillars [17]); the camera images are fed into a 2D image feature extractor (*e.g*., ResNet [11]) to obtain camera feature; then, the camera feature is fused to the lidar feature; finally, the fused feature is processed by the remaining components of the selected lidar detection framework (*e.g*., Backbone and Detection Head from Pointpillars [17]) to obtain the detection results.

In contrast to previous designs, our method enjoys two benefits: (1) high-resolution camera features with rich contextual information do not need to be wrongly voxelized and then converted from perspective view to bird's eye view; (2) domain gap and annotation issues are alleviated, and better camera features can be obtained due to the end-to-end training. However, the disadvantages are also obvious: compared with input-level decoration, aligning camera features with lidar signals becomes less straightforward on deep feature level. For example, the inaccurate alignment caused by heterogeneous data augmentation for two modalities could pose a potential challenge for the fusion stage. In Section 3.2, we verify that the misalignment harms the detection model, and provide our solution in Section 3.3.

3.2. Impact of Alignment Quality

To quantitatively assess the influence of alignment to deep feature fusion, we disable all other data augmentations but only twist the magnitudes of RandomRotation [46] to the lidar point cloud of our deep fusion pipeline during training. Since we only augment the lidar point cloud but keep the camera images unchanged, stronger geometry-related data augmentation leads to worse alignment. As

[1] https://github.com/NVIDIA/semantic-segmentation

Max Rotation	0°	15°	30°	45°
Single-Modal	72.6	75.2	76.6	77.6
Multi-Modal	75.2	76.1	77.3	78.0
Improvement	+2.6	+0.9	+0.8	+0.4

Table 1. Performance gain by multi-modal fusion diminishes as the magnitude of RandomRotation [46] goes up, indicating the importance of accurate alignment. InverseAug is not used here. On the Waymo Open Dataset pedestrian detection task, the LEVEL 1 AP improvements from single-modal to multi-modal are reported. See Section 3.2 for more details.

shown in Table 1, the benefit from multi-modal fusion diminishes as the rotation angle enlarges. For example, when no augmentation is applied (maximum rotation $= 0^\circ$), the improvement is the most significant (+2.6 AP); when maximum rotation is 45°, only +0.4 AP gain is observed. Based on these observation we conclude that alignment is critical for deep feature fusion that if the alignment is not accurate, the benefit from the camera input becomes marginal.

3.3. Boosting Alignment Quality

In view of the importance of aligning deep features, we propose two techniques, InverseAug and LearnableAlign, to effectively align deep features from two modalities.

InverseAug. To achieve the very best performance on existing benchmarks, most of the methods requires strong data augmentation, as the training usually falls into an overfitting scenario. The importance of data augmentation can be seen from Table 1, where the accuracy can be boosted by up to 5.0 for single-modal model. Besides, Cheng *et al.* [6] also suggest the importance of data augmentation for training 3D object detection models. However, the necessity of data augmentation poses a non-trivial challenge in our DeepFusion pipeline. Specifically, the data from the two modalities are usually augmented with different augmentation strategies (*e.g.*, rotating along z-axis for 3D point clouds combined with random flipping for 2D images), making the alignment challenging.

To address the alignment issue caused by geometry-related data augmentation, we propose InverseAug. As shown in Figure 2, after data augmentation is applied to a point cloud, given a 3D *key point* (which can be any 3D coordinate, such as Lidar point, voxel center, *etc*.) in the augmented space, the corresponding camera feature cannot be located in the 2D space by simply using the original lidar and camera parameters. To make the localization feasible, InverseAug first saves the augmentation parameters (*e.g.*, the rotation degree for RandomRotate [46]) when applying the geometry-related data augmentation. At the fusion stage, it reverses all those data augmentation to get the original coordinate for the 3D key point (Figure 2 (c)), and then finds its corresponding 2D coordinates in the camera space. Note that our method is generic as it can align different types of key points(*e.g.*, the voxel centers), although we only adopt lidar points in Figure 2 for simplicity, and it can also handle situations where both modalities are augmented. In contrast, existing fusion methods like PointAugmenting [36] can only deal with the data before augmentation. Finally, we show an example of the alignment quality improvement by InverseAug in Figure 3 (b).

LearnableAlign. For input-level decoration methods such as PointPainting [34] and PointAugmenting [36], given a 3D lidar point, the only corresponding camera pixel can be exactly located as there is a one-to-one mapping. In contrast, when fusing deep features in our DeepFusion pipeline, each lidar feature represents a voxel containing a subset of points and hence its corresponding camera pixels are in a polygon. So the alignment becomes a one-voxel-to-many-pixels problem. A naive approach is to average over all pixels corresponding to the given voxel. However, intuitively, and as supported by our visualized results, these pixels are not equally important because the information from the lidar deep feature unequally aligns with every camera pixel. For example, some pixels may contain critical information for detection, such as the target object to detect, while others may be less informative, consisting of background such as roads, plants, occluders, *etc*.

In order to better align the information from lidar features with the most related camera features, we introduce LearnableAlign, that leverages cross-attention mechanism to dynamically capture the correlations between two modalities as shown in Figure 1. Specifically, the input contains a voxel cell, and all its corresponding N camera features. LearnableAlign uses three fully-connected layers to respectively transform the voxel to the query q^{l}, and camera features to the keys k^{c} and values v^{c}. For each query (*i.e.*, voxel cell), we conduct inner product between the query and the keys to obtain the attention affinity matrix that contains $1 \times N$ correlations between the voxel and all its corresponding N camera features. Normalized by a softmax operator, the attention affinity matrix is then used to weigh and aggregate the values v^{c} that contains camera information. The aggregated camera information is then processed by a fully-connected layer, and finally concatenated with the original lidar feature. The output is finally fed into any standard 3D detection framework, such as PointPillars or CenterPoint for model training.

4. Experiments

We evaluate DeepFusion on Waymo Open Dataset [32], a large scale 3D object detection dataset for self-driving cars. Waymo Open Dataset contains 798 training, 202 validation, and 150 testing sequences. Each sequences have about 200 frames with lidar points, camera images, and labeled 3D bounding boxes. We evaluate and compare mod-

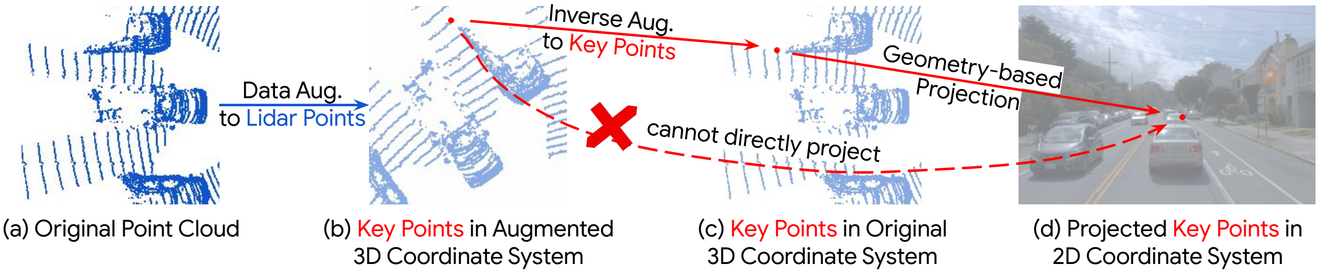

Data Aug.
to Lidar Points
Inverse Aug.
to Key Points
Geometry-based
Projection
cannot directly project
(a) Original Point Cloud
(b) Key Points in Augmented 3D Coordinate System
(c) Key Points in Original 3D Coordinate System
(d) Projected Key Points in 2D Coordinate System

Figure 2. **The pipeline of InverseAug.** The goal of the proposed InverseAug is to project the *key points* obtained after the data augmentation stage, *i.e.*, (a) → (b), to the 2D camera coordinate system. The key point is a generic concept that can be any 3D coordinate, such as a lidar point or a voxel center. For simplicity, we use a lidar point here to illustrate the idea. It is less accurate to directly project the key points from the augmented 3D coordinate system to 2D camera coordinate system by using camera and lidar parameters, *i.e.*, directly from (b) to (d). Here we propose to first find all key points in the original coordinate by inversely applying all data augmentation to the 3D key points, *i.e.*, (b) → (c). Then, the lidar and camera parameters can be used to project 3D key points to camera features, *i.e.*, (c) → (d). InverseAug significantly improves the alignment quality as shown in Figure 3.

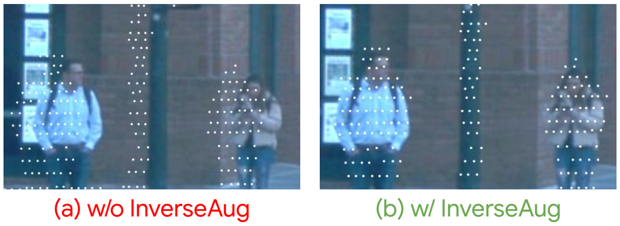


Figure 3. The camera-lidar alignment quality comparison before and after applying InverseAug. As shown in (a), without InverseAug, the lidar points (marked as white) are not well-aligned to the pedestrians and the pillar in camera view. In contrast, as shown in (b), the lidar points align better to the camera data with InverseAug. Note that we only add a small magnitude of data augmentation in this figure. The misalignment is more severe without InverseAug during training where strong data augmentations are widely applied.

els using the recommended metrics, Average Precision (AP) and Average Precision weighted by Heading (APH), and report the results on both LEVEL_1 (L1) and LEVEL_2 (L2) difficulty objects. We highlight LEVEL_2 APH in tables since it is the main metric for ranking in the Waymo Challenge Leaderboard.

4.1. Implementation Details

3D detection models. We reimplement three popular point cloud 3D object detection methods, PointPillars [17], CenterPoint [44], and 3D-MAN [43], as baselines. Besides, we also find that their improved versions (denote as PointPillars++, CenterPoint++, and 3D-MAN++) can be better baselines, which use 3 layers of multilayer perceptron with hidden size 256 to construct pseudo image from point cloud inputs, and change the non-linear activation function from ReLU [10, 21] to SILU [8, 28]. By default, all experiments are conducted with the 3D-MAN++ pedestrian models. The final model submitted to the test server also combined with other techniques such as model ensemble (noted as "Ens"), which are discussed in Section **??**.

LearnableAlign. We use fully connected layers with 256 filters to embed the lidar feature and its corresponding camera features. In the lidar to camera cross-attention module, a dropout operator with 30% drop rate is applied to the attention affinity matrix as a regularization during training. The MLP layer after the cross-attention module is a fully connected layer with 192 filters. Finally, the concatenated feature is processed by another fully connected layer to squeeze the number of channels. Different from standard implementation of attention module, our implementation is combined with dynamic voxelization [45].

InverseAug. Inspired by PPBA [6], we sequentially apply the following data augmentation strategies to lidar point cloud during training: RandomRotation → WorldScaling → GlobalTranslateNoise → RandomFlip → FrustumDropout → RandomDropLaserPoints. More details about the augmentation operators can be found in [6]. Different from PPBA [6] and other works, here we save all randomly generated parameters for all geometry-related data augmentation (*i.e.*, RandomRotation, WorldScaling, GlobalTranslateNoise, RandomFlip). During the fusion stage, we inversely apply all these augmentation with saved parameters to all 3D key points to find the coordinates in the original 3D coordinate system before applying data augmentation. Besides, we also need to reverse the order of augmentation operations (*i.e.*, RandomFlip → GlobalTranslateNoise → WorldScaling → RandomRotation).

4.2. State-of-the-art performance on Waymo Data

We compare our method with the published and unpublished 3D object detection methods on Waymo Open Dataset `validation` and `test` sets.

Method Name	AP/L1	APH/L1	AP/L2	**APH/L2**
DeepFusion-Ens (ours)†	**84.37**	**83.22**	**79.54**	**78.41**
InceptioLidar	83.80	82.46	79.15	77.84
AFDetV2-Ens [13]	84.07	82.63	79.04	77.64
Octopus_Noah	83.10	81.67	78.65	77.27
HorizonLiDAR3D [7]†	83.28	81.85	78.49	77.11
DeepFusion (ours)†*	**81.89**	**80.48**	**76.91**	**75.54**
Cascade3D	81.17	79.63	75.84	74.36
INT	80.29	78.81	75.30	73.89
IUI	80.00	78.60	74.94	73.60
XMU	80.53	78.77	75.14	73.45
Octopus-det	79.25	77.75	74.63	73.20
AFDetV2 [13]*	79.77	78.21	74.60	73.12
LENOVO_LR_PCIE_Det	79.46	78.07	74.31	72.97
LENOVO_LR_PCIE_RT_Det	79.42	77.98	74.31	72.97
CenterPoint++ [44]*	79.41	77.96	74.22	72.82
SST_v1 [9]*	79.99	78.31	74.41	72.81

Table 2. Leaderboard of the Waymo Open Dataset Challenge on 3D detection track. *: to our best knowledge, these entries (highlighted by light blue) do not use model ensemble. †: multi-modal methods.

Diff-iculty	Method	Veh.		Ped.	
		AP	APH	AP	APH
L1	PointPillars [17, 33]	63.3	62.7	68.9	56.6
	PPBA [6]	62.4	_	66.0	_
	MVF [45]	62.9	_	65.3	_
	CVCNet [2]	65.2	_	_	_
	PointAugmenting [36]†	67.4	_	75.4	_
	Pillar-OD [37]	69.8	_	72.5	_
	PV-RCNN [30]	74.4	73.8	61.4	53.4
	CenterPoint [44]	76.7	76.2	79.0	72.9
	RSN [33]	78.4	78.1	79.4	76.2
	DeepFusion (ours)†	80.6	80.1	85.8	83.0
	DeepFusion-Ens (ours)†	**83.6**	**83.2**	**87.1**	**84.7**
L2	PointPillars [17, 33]	55.2	54.7	60.0	49.1
	PointAugmenting [36]†	62.7	_	70.6	_
	PV-RCNN [30]	65.4	64.8	53.9	46.7
	CenterPoint [44]	68.8	68.3	71.0	65.3
	RSN [33]	69.5	69.1	69.9	67.0
	DeepFusion (ours)†	72.9	72.4	78.7	76.0
	DeepFusion-Ens (ours)†	**76.0**	**75.6**	**80.4**	**78.1**

Table 3. Performance comparison between models for 3D detection on Waymo validation set. †: multi-modal methods.

According to the test results in Table 2, DeepFusion achieves the best results on Waymo Challenge Leaderboard demonstrating the effectiveness of our approach. For example, DeepFusion-Ens achieves the best results on the Waymo Challenge Leaderboard; DeepFusion improves 2.42 APH/L2 compared with previous state-of-the-art single-model method, AFDetV2 [13].

We also compare different methods on the validation set as shown in Table 3. DeepFusion significantly outperforms existing methods, demonstrating its effectiveness.

4.3. DeepFusion is a generic fusion method

Now we examine how generic our method is by plugging it to prevalent 3D detection frameworks. We conduct six pairs of comparison, each of which is between a single-modal method and its multi-modal counterpart. Those six lidar-only models are PointPillars, CenterPoint, 3D-MAN and their improved version (marked as "++"). As shown in Table 4, DeepFusion shows consistent improvement for all single-modal detection baselines. These results indicate DeepFusion is generic and can be potentially applied to other 3D object detection frameworks.

Model	Modal	LEVEL 1		LEVEL 2	
		AP	APH	AP	**APH**
PointPillars [17]	L	67.4	55.6	58.4	48.1
+DeepFusion	L + C	72.0	62.8	63.0	54.8 (+6.7)
PointPillars++	L	72.5	62.9	63.3	54.8
+DeepFusion	L + C	73.9	65.1	64.9	57.0 (+2.2)
CenterPoint [44]	L	71.9	62.0	63.1	54.3
+DeepFusion	L + C	78.3	70.8	70.2	63.2 (+8.9)
CenterPoint++	L	77.5	69.5	68.5	61.3
+DeepFusion	L + C	81.2	75.0	73.1	67.2 (+5.9)
3D-MAN [43]	L	71.3	58.9	63.4	52.2
+DeepFusion	L + C	75.9	65.9	67.4	58.4 (+6.2)
3D-MAN++	L	78.5	70.2	70.7	63.0
+DeepFusion	L + C	81.2	75.0	72.8	67.0 (+4.0)

Table 4. Plugging DeepFusion into different single-modal baselines on Waymo validation set. `L` denotes lidar-only; `L+C` denotes lidar + camera. We evaluate Pointpillar, CenterPoint, 3D-MAN, and their improved versions (denoted with "++"). By adding camera information, our DeepFusion consistently improves the quality over lidar-only models.

4.4. Where does the improvement come from?

To better understand how DeepFusion utilizes the camera signal to improve the 3D object detection models, we provide both qualitative and quantitative analysis in depth.

We first divide the objects into three groups based on their distance to the ego-car: within 30 meters, from 30 to 50 meters, and beyond 50 meters. Figure 4 shows the *relative* gain by the multi-modal fusion of each group. In a nutshell, DeepFusion can uniformly improve the accuracy in every single distance range. In particular, it can achieve much more accuracy gains for long-range objects (by 6.6% for LEVEL_2 objects $> 50m$) than short-range objects (by 1.5% for LEVEL_2 objects $< 30m$), possibly because long-range objects are often covered by very sparse lidar points, but the high-resolution camera signals fill the information gap to a large extent.

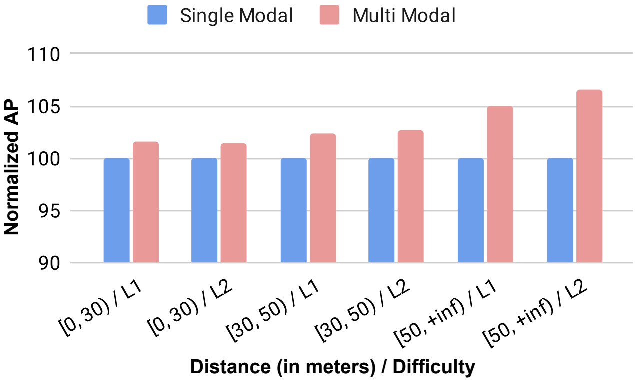


Figure 4. Comparison between the single-modal baseline and DeepFusion by showing the AP metric (with all the blue bars normalized to 100%) across different ground-truth depth ranges. The results show DeepFusion marginally improves the performance on short-range objects (*e.g.*, within 30 meters) but significantly improves the performance on long-range objects (*e.g.*, beyond 50 meters).

Modal	IA	LA	LEVEL 1		LEVEL 2	
			AP	APH	AP	**APH**
Lidar	N/A	N/A	78.5	70.2	70.7	63.0
Lidar + Camera	✗	✓	78.4	70.5	70.9	63.5
Lidar + Camera	✓	✗	80.4	74.0	72.3	66.4
Lidar + Camera	✓	✓	**81.2**	**75.0**	**72.8**	**67.0**

Table 5. Ablation studies on InverseAug (IA) and LearnableAlign (LA). Both techniques contribute to the performance gain, while InverseAug carries more weights.

Then, we visualize the attention map of LearnableAlign in Figure 5. We observe that model tends to focus on the regions with strong discriminative power such as the head of the pedestrian, and on the object extremities such as the back of the pedestrian. Base on these observations, we conclude that the high-resolution camera signals could help with recognizing, and predicting the object boundaries.

4.5. Impact of InverseAug and LearnableAlign

In this section, we show the effectiveness of both proposed components, InverseAug and LearnableAlign. According to Table 5, we observe that both components can improve the the performance over the single-modal baseline. In particular, the gain by InverseAug is more prominent. For example, without InverseAug, the performance of LEVEL_2 detection drops drastically from 67.0 APH to 63.5 APH, which is already very close to the performance of the lidar only model, 63.0 APH. On the other hand, though milder, the improvement by LearnableAlign is not negligible. For example, LearnableAlign improves the final performance of LEVEL_2 objects from 66.4 APH to 67.0 APH. The ablation study indicates both components are so critical that we should remove neither of them.

Model	Latency (sec)	LEVEL 1		LEVEL 2	
		AP	APH	AP	**APH**
Single-Modal	0.16	78.5	70.2	70.7	63.0
InputFusion	0.28	79.4	74.1	72.1	66.6
LateFusion	0.35	79.8	73.7	72.3	66.5
DeepFusion (ours)	0.32	**81.2**	**75.0**	**72.8**	**67.0**

Table 6. Comparison with other fusion strategies. InputFusion comes from PointPainting [34] and PointAugmenting [36]. LateFusion comes from PointAugmenting [36]. All latency are measured on a V100 GPU with the same Lingvo [29] 3D object detection implementation, same 3D detection backbone, and same camera feature extractor. DeepFusion achieves the best performance on all evaluation metrics, while the latency is comparable to other fusion methods.

4.6. DeepFusion is an effective fusion strategy

In this section, we compare DeepFusion with other fusion strategies. Specifically, the methods we consider are (1) InputFusion, that fuses the camera features with the lidar points at the input stage [34, 36], (2) LateFusion, where the lidar points and camera features go through voxelizer separately, followed by a concatenation [36], and (3) our proposed DeepFusion.

The results are shown in Table 6. We observe that DeepFusion is notably better than other fusion strategies. For example, DeepFusion improves 0.5 LEVEL_2 APH (from 66.5 to 67.0) upon LateFusion. Note that in our experiments, InputFusion is on par with LateFusion, but in [36], LateFusion is better as it addresses the modality gap issue between lidar and camera. We hypothesis that in our setting, the modality gap issue is already taken care of by the end-to-end training that it will no longer occur regardless when the fusion is conducted.

4.7. DeepFusion is more robust

Robustness is an important metric to deploy models on autonomous driving cars [20]. In this subsection, we examine the model robustness against corrupted input [12] and Out-Of-Distribution (OOD) data [35].

Robustness against input corruptions. We first test the model robustness on validation set against common corruptions for both modality, Laser Noise (randomly adding noise to lidar reflections) and Pixel Noise (randomly adding noise to camera pixels). For single-modal models, only Laser Noise is applicable while both Laser Noise and Pixel Noise are applicable for multi-modal models. As shown in Table 7, with the presence of corruptions, the multi-modal models in general are much more robust than their single-modal counterpart. Notably, the Laser / Pixel Noise corruption only can hardly drag down the performance of our multi-modal method (with only 0.2 / 0.5 L2 APH drop). Even when applying both Laser and Pixel Noise corrup-

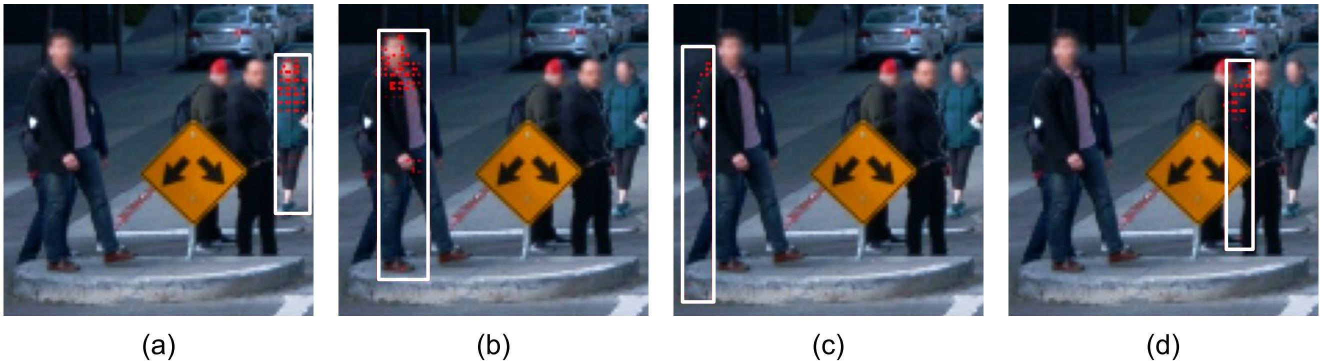


Figure 5. The the attention map visualization for LearnableAlign. For each subfigure, we study one 3D point pillar, which is marked by the white box in the 2D image. The important region indicated by the attention map is marked by red points. We have two interesting observations: first, as shown in (a) and (b), LearnableAlign usually attends to the heads of the pedestrians, probably because the head is a discriminative part of a human from camera image (it is difficult to recognize a head according to lidar signals); second, as shown in (c) and (d), LearnableAlign also attends to object extremities (such as the back), which strives to use the high-resolution camera information for predicting the object boundary to get accurate object size.

Corruptions	Modal	LEVEL 1		LEVEL 2	
		AP	APH	AP	**APH**
No Corruption	L	78.5	70.2	70.7	63.0
Laser Noise	L	69.8	59.8	61.3	52.3 **(-10.7)**
No Corruption	L + C	81.2	75.0	72.8	67.0
Laser Noise	L + C	81.1	74.8	72.6	66.8 **(-0.2)**
Pixel Noise	L + C	80.9	74.6	72.3	66.5 **(-0.5)**
Pixel + Laser Noise	L + C	80.9	74.7	72.4	66.6 **(-0.4)**

Table 7. Model robustness against input corruptions. Given the same well-trained single-modal (**L**idar) and multi-modal (**L**idar + **C**amera) models, we evaluate on the original Waymo validation set (NoCorruption), and manually erode the samples from validation set by Laser and Pixel Noise. For Laser Noise, we add perturbation to the reflection values of all laser points. For Pixel Noise, we add perturbation to the camera images. Note that Pixel Noise is only applicable to multi-modal models which use camera images as input. The perturbations are sampled from a uniform distribution with at most 2.5% of the original value for both Laser and Pixel Noise corruptions. We observe DeepFusion is more robust to these corruptions compared to the single-modal version. `L` denotes lidar-only; `L+C` denotes lidar + camera.

tions, the performance drop is still marginal (0.4 L2 APH drop). Meanwhile, the single-modal model drops more than 10 APH by simply applying the Laser Noise corruption.

Robustness against OOD data. To test our method's robustness against OOD data, we train our model on the data from cities of Mountain View, San Francisco and Phoenix and evaluate it on Kirkland. The results are summarized in Table 8. We observe the multi-modal model shows more improvement on OOD data. For example, DeepFusion improves 8.0 LEVEL_2 APH on out-of-distribution data while only improves 4.0 LEVEL 2 APH on in-distribution data.

Validation Set	Modal	LEVEL 1		LEVEL 2	
		AP	APH	AP	**APH**
Default	L	78.5	70.2	70.7	63.0
Default	L + C	81.2	75.0	72.8	67.0 (+4.0)
Kirkland	L	43.8	38.8	31.2	27.6
Kirkland	L + C	52.0	47.4	38.0	34.6 **(+8.0)**

Table 8. Model robustness against out-of-distribution data. We evaluate both single-modal (**L**idar) and multi-modal (**L**idar + **C**amera) models on the in-distribution validation set (Default) and out-of-distribution validation set (Kirkland). DeepFusion achieves larger improvement on out-of-distribution validation set. `L` denotes lidar-only; `L+C` denotes lidar + camera.

5. Conclusion

This paper studies how to effectively fuse lidar and camera data for multi-modal 3D object detection. Our study shows deep feature fusion in late stage can be more effective when they are aligned well, but aligning two deep features from different modality are challenging. To address this challenge, we propose two techniques, InverseAug and LearnableAlign, to get effective alignment among multi-modal features. Based on these techniques, we develop a family of simple, generic, yet effective multi-modal 3D detectors, named DeepFusions, which achieves state-of-the-art performance on the Waymo Open Dataset.

Acknowledgement. We would like to thank Zhaoqi Leng and Ekin Dogus Cubuk for data augmentation discussion, and Pei Sun for model ensemble discussion. We would like to thank Google Cloud Neural Architecture Search (NAS) project leads Vishy Tirumalashetty and Shengyang Dai. Yingwei Li would like to thank Longlong Jing for the detailed tutorial on 3D object detection task, and Zhiwen Wang for suggestions on figures. YL also would like to thank Tianwei Yin, Xingyi Zhou, and Chenxu Luo for tutorial on 3D object detection task.

References

[1] Alex Bewley, Pei Sun, Thomas Mensink, Dragomir Anguelov, and Cristian Sminchisescu. Range conditioned dilated convolutions for scale invariant 3d object detection. *arXiv preprint arXiv:2005.09927*, 2021. 2

[2] Qi Chen, Lin Sun, Ernest Cheung, and Alan L Yuille. Every view counts: Cross-view consistency in 3d object detection with hybrid-cylindrical-spherical voxelization. *Advances in Neural Information Processing Systems*, 2020. 6

[3] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2156, 2016. 2

[4] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1907–1915, 2017. 2

[5] Xuanyao Chen, Tianyuan Zhang, Yue Wang, Yilun Wang, and Hang Zhao. Futr3d: A unified sensor fusion framework for 3d detection. *arXiv preprint arXiv:2203.10642*, 2022. 2

[6] Shuyang Cheng, Zhaoqi Leng, Ekin Dogus Cubuk, Barret Zoph, Chunyan Bai, Jiquan Ngiam, Yang Song, Benjamin Caine, Vijay Vasudevan, Congcong Li, et al. Improving 3d object detection through progressive population based augmentation. In *European Conference on Computer Vision (ECCV)*, pages 279–294. Springer, 2020. 4, 5, 6

[7] Zhuangzhuang Ding, Yihan Hu, Runzhou Ge, Li Huang, Sijia Chen, Yu Wang, and Jie Liao. 1st place solution for waymo open dataset challenge–3d detection and domain adaptation. *arXiv preprint arXiv:2006.15505*, 2020. 6

[8] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018. 5

[9] Lue Fan, Ziqi Pang, Tianyuan Zhang, Yu-Xiong Wang, Hang Zhao, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Embracing single stride 3d object detector with sparse transformer. *arXiv preprint arXiv:2112.06375*, 2021. 6

[10] Richard HR Hahnloser, Rahul Sarpeshkar, Misha A Mahowald, Rodney J Douglas, and H Sebastian Seung. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature*, 405(6789):947–951, 2000. 5

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 3

[12] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations (ICLR)*, 2018. 7

[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. *arXiv preprint arXiv:2112.09205*, 2021. 6

[14] Tengteng Huang, Zhe Liu, Xiwu Chen, and Xiang Bai. Ep-net: Enhancing point features with image semantics for 3d object detection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *ECCV*, 2020. 2, 3

[15] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8. IEEE, 2018. 2

[16] Jason Ku, Alex D Pon, and Steven L Waslander. Monocular 3d object detection leveraging accurate proposals and shape reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11867–11876, 2019. 2

[17] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12697–12705, 2019. 2, 3, 5, 6

[18] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *ECCV*, 2018. 2, 3

[19] Gregory P Meyer, Ankit Laddha, Eric Kee, Carlos Vallespi-Gonzalez, and Carl K Wellington. Lasernet: An efficient probabilistic 3d object detector for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12677–12686, 2019. 2

[20] Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking robustness in object detection: Autonomous driving when winter is coming. *arXiv preprint arXiv:1907.07484*, 2019. 7

[21] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning (ICML)*, pages 807–814, 2010. 5

[22] Jiquan Ngiam, Benjamin Caine, Wei Han, Brandon Yang, Yuning Chai, Pei Sun, Yin Zhou, Xi Yi, Ouais Alsharif, Patrick Nguyen, et al. Starnet: Targeted computation for object detection in point clouds. *arXiv preprint arXiv:1908.11069*, 2019. 2

[23] AJ Piergiovanni, Vincent Casser, Michael S Ryoo, and Anelia Angelova. 4d-net for learned multi-modal alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15435–15445, 2021. 3

[24] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 918–927, 2018. 2

[25] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, 2017. 2

[26] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017. 2

[27] Rui Qian, Divyansh Garg, Yan Wang, Yurong You, Serge Belongie, Bharath Hariharan, Mark Campbell, Kilian Q Weinberger, and Wei-Lun Chao. End-to-end pseudo-lidar for image-based 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5881–5890, 2020. 2

[28] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017. 5

[29] Jonathan Shen, Patrick Nguyen, Yonghui Wu, Zhifeng Chen, et al. Lingvo: a modular and scalable framework for sequence-to-sequence modeling, 2019. 7

[30] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN: point-voxel feature set abstraction for 3d object detection. In *CVPR*, 2020. 6

[31] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*, 2019. 2

[32] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2446–2454, 2020. 4

[33] Pei Sun, Weiyue Wang, Yuning Chai, Gamaleldin Elsayed, Alex Bewley, Xiao Zhang, Cristian Sminchisescu, and Dragomir Anguelov. Rsn: Range sparse net for efficient, accurate lidar 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5725–5734, 2021. 6

[34] Sourabh Vora, Alex H Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 4604–4612, 2020. 1, 2, 3, 4, 7

[35] Apoorv Vyas, Nataraj Jammalamadaka, Xia Zhu, Dipankar Das, Bharat Kaul, and Theodore L Willke. Out-of-distribution detection using an ensemble of self supervised leave-out classifiers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 550–564, 2018. 7

[36] Chunwei Wang, Chao Ma, Ming Zhu, and Xiaokang Yang. Pointaugmenting: Cross-modal augmentation for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11794–11803, 2021. 1, 2, 3, 4, 6, 7

[37] Yue Wang, Alireza Fathi, Abhijit Kundu, David A Ross, Caroline Pantofaru, Tom Funkhouser, and Justin Solomon. Pillar-based object detection for autonomous driving. In *European Conference on Computer Vision (ECCV)*, pages 18–34. Springer, 2020. 6

[38] Yue Wang, Alireza Fathi, Abhijit Kundu, David A. Ross, Caroline Pantofaru, Thomas A. Funkhouser, and Justin Solomon. Pillar-based object detection for autonomous driving. In *ECCV*, 2020. 2

[39] Zhixin Wang and Kui Jia. Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3d object detection. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1742–1749. IEEE, 2019. 2

[40] Yan Yan, Yuxing Mao, and Bo Li. SECOND: sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 2

[41] Bin Yang, Wenjie Luo, and Raquel Urtasun. PIXOR: real-time 3d object detection from point clouds. In *CVPR*, 2018. 2

[42] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Ipod: Intensive point-based object detector for point cloud. *arXiv preprint arXiv:1812.05276*, 2018. 2

[43] Zetong Yang, Yin Zhou, Zhifeng Chen, and Jiquan Ngiam. 3d-man: 3d multi-frame attention network for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1863–1872, 2021. 2, 5, 6

[44] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11784–11793, 2021. 2, 5, 6

[45] Yin Zhou, Pei Sun, Yu Zhang, Dragomir Anguelov, Jiyang Gao, Tom Ouyang, James Guo, Jiquan Ngiam, and Vijay Vasudevan. End-to-end multi-view fusion for 3d object detection in lidar point clouds. In *Conference on Robot Learning*, pages 923–932. PMLR, 2020. 5, 6

[46] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4490–4499, 2018. 2, 3, 4