Uni3D: A Unified Baseline for Multi-dataset 3D Object Detection

Bo Zhang[1], Jiakang Yuan[2], Botian Shi[†,1], Tao Chen[2], Yikang Li[1], Yu Qiao[1]
[1] Shanghai AI Laboratory
[2]School of Information Science and Technology, Fudan University
{zhangbo, shibotian, liyikang, qiaoyu}@pjlab.org.cn

Abstract

Current 3D object detection models follow a single dataset-specific training and testing paradigm, which often faces a serious detection accuracy drop when they are directly deployed in another dataset. In this paper, we study the task of training a unified 3D detector from multiple datasets. We observe that this appears to be a challenging task, which is mainly due to that these datasets present substantial data-level differences and taxonomy-level variations caused by different LiDAR types and data acquisition standards. Inspired by such observation, we present a Uni3D which leverages a simple data-level correction operation and a designed semantic-level coupling-and-recoupling module to alleviate the unavoidable data-level and taxonomy-level differences, respectively. Our method is simple and easily combined with many 3D object detection baselines such as PV-RCNN and Voxel-RCNN, enabling them to effectively learn from multiple off-the-shelf 3D datasets to obtain more discriminative and generalizable representations. Experiments are conducted on many dataset consolidation settings. Their results demonstrate that Uni3D exceeds a series of individual detectors trained on a single dataset, with a 1.04× parameter increase over a selected baseline detector. We expect this work will inspire the research of 3D generalization since it will push the limits of perceptual performance. Our code is available at: https://github.com/PJLab-ADG/3DTrans.

1. Introduction

LiDAR-based 3D object detection [2, 4, 5, 10, 16, 17, 24, 31, 36] aims to recognize and localize instance objects from a given frame using LiDAR sensor, which recently has achieved great progress owing to the rapid development of the large-scale annotated 3D LiDAR datasets such as Waymo [21], nuScenes [1], and KITTI [5].

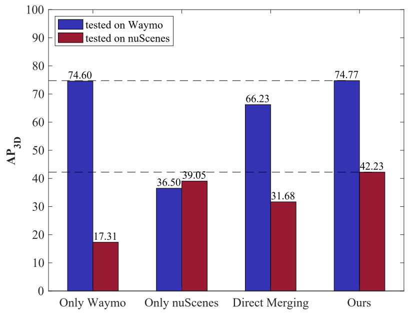

Figure 1. Challenges in training a detector from multiple datasets: 1) Only Waymo and Only nuScenes refer to the baseline detector trained on each individual dataset. 2) Direct Merging represents that we simply merge Waymo and nuScenes and train the detector on the merged dataset. 3) Ours denotes that the baseline detector is trained using the proposed method on the merged dataset.

Unfortunately, the existing supervised 3D object detection models are designed by following the typical single-dataset training-and-testing paradigm, which inevitably suffers from severe accuracy drop issue [28, 29], when these detection models are directly deployed to another dataset with different data distributions. For example, Fig 1 indicates that the baseline detector trained on Waymo [21] suffers from a serious detection accuracy degradation (from 74.60% to 17.31%) when it is evaluated on another different dataset nuScenes [1]. As a result, such a single-dataset training-and-testing paradigm cannot perform well on different datasets, further hurting the dataset-level generalization ability of the current 3D perception models.

In order to reduce the differences between different 3D datasets, some researchers [11, 24, 25, 28, 29] try to leverage Unsupervised Domain Adaptation (UDA) technique, which aims to transfer a pre-trained source-domain detector to a new domain (or dataset). Although these UDA-based 3D object detection works achieve good detection accuracy gains on the new target domain, they are still a source-to-target unidirectional model adaptation process, rather than a multi-dataset bi-directional generalization process.

Accordingly, to design a unified 3D object detection

[†]Corresponding to: Botian Shi (shibotian@pjlab.org.cn)

framework that can fully learn from different target datasets, we start by directly merging multiple datasets and re-training the baseline detector on the merged dataset, and found that the multi-dataset detection accuracy achieved by such a simple way is unsatisfactory, as illustrated in the results of `Direct Merging` in Fig. 1. This is mainly because compared with 2D image domain, 3D point cloud data present more serious cross-dataset discrepancies caused by various and complex reasons including sensor type differences, traffic scene changes, data acquisition variations, and *etc*, which is termed as **dataset-interference issue**. Further, with the constant increase of autonomous driving datasets, it becomes a very important topic for how to train a unified detector from such diversified 3D datasets.

In this paper, we propose a Unified 3D object detection framework (Uni3D) to address the dataset-interference issue. Orthogonal to the existing 3D object detection research works [4, 10, 16, 18, 26] focusing on developing an effective framework verified within a single dataset, Uni3D aims to propose a simple-and-versatile way to enable the existing 3D object detection models to have the ability of learning from many diversified 3D datasets. To achieve this goal, we design a simple data-level correction operation that can use dataset-specific channel-wise mean and variance to normalize features from each backbone layer. Besides, a semantic-level coupling-and-recoupling module is designed to strengthen the feature reusability across different 3D datasets, by calculating a spatial-wise attention map and a dataset-level attention mask to constrain the learned high-level features to be dataset-agnostic.

Extensive experiments are conducted on three public 3D autonomous driving datasets including Waymo [21], nuScenes [1], and KITTI [5], to investigate the reasons for the 3D dataset-interference issue. Besides, this paper provides many preliminary studies that explore the possibility of training a 3D object detection model under the merged datasets. The experimental results show that Uni3D has a strong dataset-level generalization ability, improving the zero-shot learning ability for unseen scenes and even surpassing the baseline methods trained on a single dataset.

2. Related Works

2.1. LiDAR-based General 3D Object Detection

Recent LiDAR-based 3D object detection works [2, 4, 10, 12, 16, 18, 19, 26, 27, 30, 36] can be roughly categorized into voxel-based methods, point-based methods, and point-voxel fusion methods. Voxel-based methods [20, 26, 36] convert irregular LiDAR points to ordered voxels before backbone feature extraction. SECOND [26] is a prior work that utilizes sparse convolution as 3D backbone and greatly improves the detection efficiency. Voxel-RCNN [4] analyses the advantages of voxel features and explores a good trade-off between detection accuracy and inference speed. Unlike voxel-based methods, point-based methods [18, 32] directly generate feature maps from raw point clouds. Inspired by PointNet [13] and PointNet++ [14], Point-RCNN [18] is a pioneer to investigate how to generate bounding boxes from point cloud data. To reduce the high memory and computational cost of point-based methods, IA-SSD [32] proposes a single-stage method by employing learning-based instance-aware down-sampling strategies. Besides, some works try to combine the benefits of point- and voxel-based representations. Among them, PV-RCNN [16] designs a point-voxel feature set abstraction to fully combine point features and voxel features. However, the above detectors are trained and evaluated within a single 3D dataset, and they will suffer from severe detection accuracy drop issues across different datasets. Further, learning generalizable representations between different datasets is more challenging in 3D scenarios due to the more serious dataset-level gaps.

2.2. Joint Training on Multiple Datasets

For traditional 2D perception tasks such as object detection [15, 37] and semantic segmentation [34], training a unified model from different datasets results in a low recognition accuracy, since different datasets often present inconsistent class definitions and annotation granularity. Motivated by this, some researchers start to study how to achieve a multi-dataset perception task [3, 6, 9, 23, 33, 35]. Early works [9, 33] focus on merging the taxonomy information and train the model on a unified label space. Mseg [9] aligns pixel-level annotations of seven datasets and significantly boosts the generalization ability of the model. Zhao *et al*. [33] propose to train a dataset-specific detector to generate pseudo labels, which provide additional annotation information from another dataset, and the final network is trained on a specific dataset using both pseudo labels and ground truths. To alleviate the annotation cost of unifying the label space, recent works [23,35] attempt to use dataset-specific supervision. Wang *et al*. [23] employ a designed domain adaptation layer and attention mechanism to alleviate the dataset-level differences. Zhou *et al*. [35] introduce a novel automatic way to merge the taxonomy space, showing that the unified detector trained on multiple datasets can outperform each detector trained on the specific dataset. Although jointly training a unified detector has been recently studied in 2D perception tasks, its further exploration on 3D perception tasks, such as 3D object detection, is still insufficient.

3. The Proposed Method

The overall framework is shown in Fig. 3. We first describe our problem setting and the multi-dataset evaluation method. Next, we analyze the limitations of the current baseline detector in multi-dataset detection, and then introduce a simple solution, namely, Uni3D.

3.1. Preliminary

Problem Setting. Suppose that a domain is defined by a joint probability distribution P_{XY} on $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are the input point cloud and label space, respectively. In the scope of **Multi-Domain Fusion (MDF)**, N denotes the number of domains $\mathcal{S} = \left\{S_n = \left\{\left(\mathbf{x}^{(n)}, y^{(n)}\right)\right\}\right\}_{n=1}^{N}$ available for model training, where each individual domain S_n is associated with a specific data distribution P_{XY}^{n}. The purpose of MDF is to train a unified model from multiple labeled domains $\mathcal{S}$ to obtain more generalizable representations $F : \mathcal{X} \rightarrow \mathcal{Y}$, which would have minimum prediction error on the multiple different domains $\mathcal{S}$.

3D Multi-dataset Training and Evaluation. Assume that in real application, we can simultaneously access multiple labeled 3D point cloud-based domains or datasets (*e.g.*, Waymo [21] and nuScenes [1]), but these labeled datasets often have different label space $\mathcal{Y}$, such as Barrier category which only presents in nuScenes [1]. Our study mainly focuses on MDF under autonomous driving scenario, where the model training and evaluation are conducted on the categories of interest related to autonomous driving scenario: vehicle, pedestrian, and cyclist. Note that **such a setting of selecting common categories, such as vehicle, pedestrian, and cyclist categories, to conduct the preliminary research is very common in many cross-dataset 3D detection works**, such as ST3D [28], ST3D++ [29].

3.2. When Single-dataset 3D Detectors Meet Multiple Datasets

Single-dataset 3D Object Detection. Currently, state-of-the-art 3D object detection models [4, 10, 16, 17, 19, 26] are trained and evaluated within a single public benchmark [1, 21], which can be regarded as so-called **single-dataset training paradigm**. Here, to better illustrate our MDF task, we first abstract the optimization objective of current 3D object detection models as follows:

$$L_{det} = L_{rpn} + L_{roi} + L_{key}, \tag{1}$$

where L_{rpn} is used to generate accurate localization prediction of preset proposals, and L_{roi} helps to refine the proposals to obtain the final 3D bounding box results. Besides, some 3D baseline detectors, such as PV-RCNN [16] and PV-RCNN++ [17], use a keypoint prediction loss L_{key} that can identify important foreground points and achieve a keypoint-to-grid RoI feature extraction process.

Limitation for MDF-based 3D Object Detection. A natural method to train on multiple datasets is to simply combine all source datasets into a merged but larger one. Unfortunately, our initial attempt of such a natural dataset combination way shows a significant performance drop of the detector that is trained on the merged dataset, compared with the performance of training on each specific sub-dataset.

As demonstrated in Table 3 of Sec. 4.3, we observe that, by comparing `Single-dataset` and `Direct Merging` baselines, for 3D scene-level datasets, directly perform a dataset-level consolidation cannot help to boost the detector's cross-dataset detection accuracy. On the contrary, the detector may suffer from the feature learning interference due to the significant differences between different datasets. For example, Voxel-RCNN [4] can obtain a relatively-high detection accuracy (75.08% AP on Waymo validation set) when it is trained only on Waymo [21] dataset. But it faces a severe performance drop when Voxel-RCNN is jointly trained on the combined dataset of Waymo and nuScenes (only 66.67% AP on Waymo validation set).

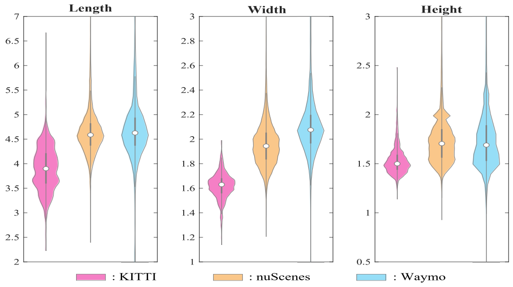

Figure 2. The statistical distribution differences of object size (Length, Width, and Height) across different datasets. To better illustrate the differences, we pick up the values within the range of [2.0, 7.0], [1.0, 3.0], and [0.5, 3.0] for Length, Width, and Height.

Datasets	Beam	VFOV	Point Range	Collection Location

Waymo [21]	64	$[-18.0°, 2.0°]$	L=[-75.2, 75.2]m
			W=[-75.2, 75.2]m
			H=[-2.0, 4.0]m
				USA

KITTI [5]	64	$[-23.6°, 3.2°]$	L=[0.0, 70.4]m
			W=[-40.0, 40.0]m
			H=[-3.0, 1.0]m
				Germany

nuScenes [1]	32	$[-30.0°, 10.0°]$	L=[-51.2, 51.2]m
			W=[-51.2, 51.2]m
			H=[-5.0, 3.0]m
				USA/Singapore

Table 1. Overview of 3D autonomous driving dataset differences. VFOV denotes vertical field of view, and L, W, and H represent the length, width, and height of LiDAR range, respectively.

Here, through extensive experiments, we give two main reasons for the above performance degradation issue.

1) Data-level differences: Compared with 2D natural images that are composed of pixels with a consistent value range of [0, 255], 3D point clouds often are collected using different sensor types with different point cloud ranges, which leads to distributional discrepancy among datasets. And the main differences of the three widely-used datasets are shown in Table 1. Actually, we found from Table 2 that sensor-derived point range difference is a major factor interfering the common feature learning from multiple datasets, which is due to that the receptive field size for the same objects are very different when data with inconsistent point cloud ranges are fed into the 3D detector. As a result, the point-cloud-range alignment is a necessary pre-processing step for achieving multi-dataset 3D object detection.

Besides, as illustrated in Fig. 2, the point clouds from

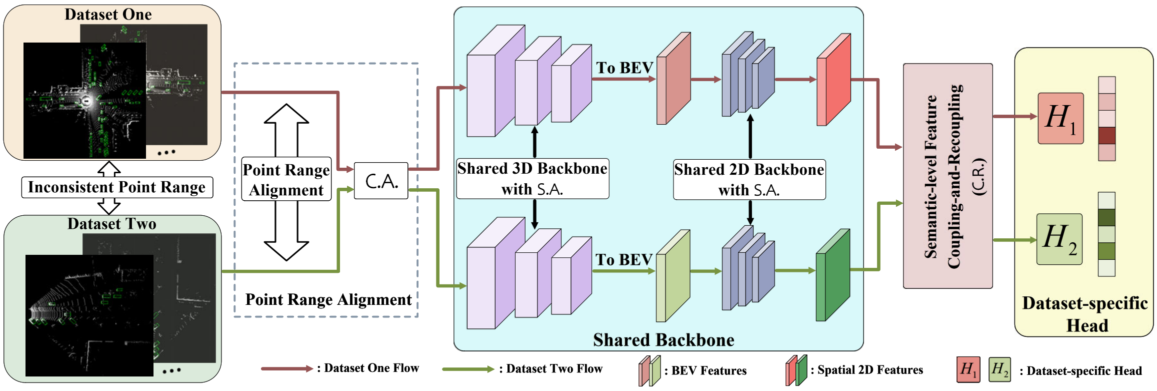

Figure 3. The overview of Uni3D including: 1) point range alignment, 2) parameter-shared 3D and 2D backbones with data-level correction operation, 3) semantic-level feature coupling-and-recoupling module, and 4) dataset-specific detection heads. `C.A.` denotes Coordinate-origin Alignment to reduce the adverse effects caused by point range alignment, and `S.A.` is the designed Statistics-level Alignment.

Methods	Waymo Range	KITTI Range	tested on Waymo AP_{3d} / APH_{3D}	tested on KITTI AP_{BEV} / AP_{3D}

Not Align.	L=[-75.2, 75.2]m
	W=[-75.2, 75.2]m
	H=[-2.0, 4.0]m
		L=[0.0, 70.4]m
		W=[-40.0, 40.0]m
		H=[-3.0, 1.0]m
			26.93 / 26.56	89.56 / **83.14**

Align. (w/ ours)	L=[-75.2, 75.2]m
	W=[-75.2, 75.2]m
	H=[-2.0, 4.0]m
		L=[-75.2, 75.2]m
		W=[-75.2, 75.2]m
		H=[-2.0, 4.0]m
			74.83 / 74.33	**90.03** / 82.39

Methods	nuScenes Range	KITTI Range	tested on nuScenes AP_{BEV} / AP_{3D}	tested on KITTI AP_{BEV} / AP_{3D}

Not Align.	L=[-51.2, 51.2]m
	W=[-51.2, 51.2]m
	H=[-5.0, 3.0]m
		L=[0.0, 70.4]m
		W=[-40.0, 40.0]m
		H=[-3.0, 1.0]m
			21.32 / 15.35	89.35 / 81.66

Align. (w/ ours)	L=[-75.2, 75.2]m
	W=[-75.2, 75.2]m
	H=[-2.0, 4.0]m
		L=[-75.2, 75.2]m
		W=[-75.2, 75.2]m
		H=[-2.0, 4.0]m
			59.25 / 41.51	**90.09 / 83.10**

Table 2. Inconsistent LiDAR ranges will cause the multi-dataset detection accuracy drop. The baseline employs Voxel-RCNN [4], and please refer to Appendix for all-category results.

different datasets present a more diverse data distribution, due to that 3D datasets were collected in different cities and countries whose instance size is very different.

2) Taxonomy-level differences: Given a fact that different autonomous driving manufacturers employ inconsistent class definitions and annotation granularity. For example, for Waymo [21], all vehicles driving on the road, including car and truck, are annotated as one unified category, namely, 'Vehicle'. While for nuScenes [1], different vehicles are annotated using different taxonomies with different granularity, such as 'Car', 'Truck, and 'Van'. As a result, the MDF task needs to consider how to train a 3D detector under an inconsistent taxonomy label space and effectively reuse domain-agnostic knowledge that can be shared across different datasets.

3.3. Uni3D: A Unified 3D Multi-dataset Object Detection Baseline

To address the data- and taxonomy-level difference issue described in the last section, we aim to develop simple modules that enable the existing 3D detectors [4,16] to learn generalizable representations from different datasets.

Data-level Correction Operation. Firstly, to accomplish a data-level correction operation, we introduce a Statistic-level Alignment (`S.A.`) that can alleviate the statistic-level differences of features extracted by common 2D or 3D backbones. This approach can be combined with any 3D detectors, including PV-RCNN [16] and Voxel-RCNN [4].

Specifically, suppose that μ^j and σ^j denote the mean and variance for each channel in the j-th network layer. Generally, the basic mean and variance statistics are used to normalize the feature in each layer, such that the input data for each layer comply with zero-mean and univariance, **e.g.,** BN [8]. However, such a statistics-shared normalization way may hurt the model transferability in MDF training, since data within one batch-size may come from different datasets having large mean and variance differences. To this end, under the MDF setting, we first obtain dataset-specific channel-wise mean μ_t^j and σ_t^j variance from the t-th dataset. Then, samples from each dataset are regularized by the current dataset-specific mean/variance as follows:

$$\hat{x}_t^j \;=\; \frac{x_t^j - \mu_t^j}{\sqrt{\sigma_t^j + \xi}}, \tag{2}$$

where x_t^j denotes the input feature of each network layer in the t-th dataset, and ξ is added to ensure the numerical stability. Further, similar to the BN [8], a transformation step is employed to restore the representation ability as follows:

$$\hat{y}_t^j \;=\; \gamma^j \hat{x}_t^j + \beta^j, \tag{3}$$

where we employ the dataset-shared gamma γ and beta β, since after the features from different datasets are normalized to zero-mean and univariance, the inter-dataset differences in the first/second-order are aligned, and further, it is reasonable to use the same gamma γ and beta β for different datasets.

Semantic-level Feature Coupling-and-Recoupling Module. In this part, we introduce a simple semantic-level Coupling-and-Recoupling module (`C.R.`) that also can be easily inserted into many single-dataset 3D detectors, to exploit the reusable features across datasets from two aspects: 1) Feature Coupling and 2) Feature Recoupling.

1) Feature Coupling: Suppose that $f^{bev} \in \mathbb{R}^{C \times H \times W}$ denotes the Bird-Eye-View (BEV) features extracted by 2D backbone network, where C denotes the channel number, H and W are the height and width of the features, respectively. BEV features $f^{bev} \in \mathbb{R}^{C \times H \times W}$ from different datasets are coupled together along the channel dimension to learn dataset-agnostic representations using a foreground-aware and dataset-level attention mask as follows:

$$\begin{aligned} f_{cat}^{bev} &= [f_i^{bev}, ..., f_j^{bev}], \\ \hat{f}_{shared}^{bev} &= [M_{shared} \odot \phi_d(Conv(f_{cat}^{bev}))]\, f_{cat}^{bev}, \end{aligned} \tag{4}$$

where [..., ...] is the concatenation operation along the channel dimension, f_i^{bev} and f_j^{bev} represent the BEV features from the i-th and j-th dataset, respectively. And $M_{shared} = \phi_p(f_{cat}^{bev})$, where ϕ_p denotes the foreground-aware spatial attention operation which is achieved by calculating the channel-wise maximum value of BEV features. Besides, the ϕ_d represents the dataset-level attention mask achieved by a Multiple Layer Perceptron (MLP) $Conv(f_{cat}^{bev})$ followed by a N-cls softmax operation, where N denotes the number of datasets to be merged. Such a dataset-level attention mask means that the ϕ_d can predict a re-scaling score to recombine BEV features from different datasets so that the combined BEV features are dataset-agnostic.

2) Feature Recoupling: Since the shared features $\hat{f}_{shared}^{bev}$ mainly focus on the common knowledge of multiple datasets, we are expected to fuse such shared features $\hat{f}_{shared}^{bev}$ with previous dataset-related BEV features f_i^{bev} or f_j^{bev} using a channel-wise re-scaling operation:

$$\begin{aligned} \hat{f}_i^{bev} &= SE_i(\hat{f}_{shared}^{bev}) + f_i^{bev}, \\ \hat{f}_j^{bev} &= SE_j(\hat{f}_{shared}^{bev}) + f_j^{bev}, \end{aligned} \tag{5}$$

where SE is the Squeeze-and-Excitation Network [7], and its network architecture is described in Appendix. The overall network structure of the designed coupling-and-recoupling module is illustrated in Fig. 4. However, exploring such feature relations across datasets will cause the inconsistency between the model multi-dataset training and single-dataset testing, mainly due to that the shared features $\hat{f}_{shared}^{bev}$ are dependent on multiple inputs. To tackle this issue, we simply use the **BEV feature copy** method, meaning that during the single-dataset inference stage, BEV features from the single dataset will be simultaneously copied to f_i^{bev} and f_j^{bev}, to obtain the shared features $\hat{f}_{shared}^{bev}$. As a result, BEV feature copy method enables us to perform the inference on a single dataset, and Uni3D is not depending on some given frame from another domain during the inference. Ablation studies on different training-and-testing methods (including **BEV feature copy** and **BEV feature mask**) are described in Appendix.

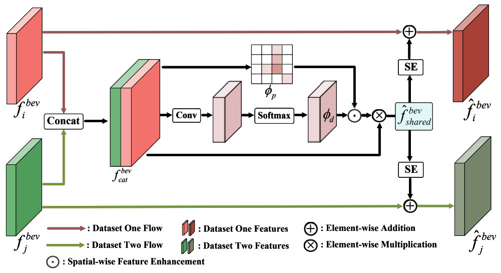

Figure 4. Semantic-level feature coupling-and-recoupling during the multi-dataset training stage.

Dataset-specific Detection Heads. To further address the taxonomy-level differences between datasets, we assume that the prior knowledge of 3D data sources is known, and propose to use different detection heads H_k followed by dataset-specific detection loss L_{det}^k to perform the instance-level prediction on different datasets. The MDF training loss function $L_{det}^{overall}$ can be written as follows:

$$L_{det}^{overall} = \sum_{k} L_{det}^k(H_k(\hat{f}_k^{bev})), \tag{6}$$

where L_{det}^k is the dataset-specific loss from the k-th dataset.

During the inference phase, we use the data-level correction operation to address the sensor-induced or data-distribution-induced differences, and the parameter-shared 3D and 2D backbones to extract the point and voxel features. Meanwhile, the detection head assigned to the corresponding dataset is used to produce final prediction results.

4. Experiments

4.1. Experimental Setup.

Datasets. We conduct experiments on three commonly-used autonomous driving datasets including Waymo [21], nuScenes [1], and KITTI [5]. These datasets present: 1) data-level distribution differences caused by different LiDAR types and geographic locations of data collection; and 2) taxonomy-level variations caused by different class annotation definitions. In our experiments, we first consider the task setting of merging two different datasets, and then, perform the study of consolidating all the above datasets.

Implementation Details. All experiments are implemented using OpenPCDet [22]. In particular, since we observe that point cloud range differences extremely degrade the cross-dataset detection accuracy as illustrated in Table 2, we align the point cloud range of all the above datasets to $[-75.2, 75.2]m$ for X and Y axes and $[-2, 4]m$ for Z axis. For all experimental settings, following the common optimization employed by PV-RCNN [16] and Voxel-RCNN [4], Adam optimizer with an initial learning rate of 0.01 is used, and the learning rate decay schedule utilizes the well-known OneCycle strategy. We train the network using a batch-size of 32, a momentum of 0.9 on 8 NVIDIA Tesla A100 GPUs, and the total training epoch

Trained on	Baseline Detectors	Tested on Waymo			Tested on nuScenes		
		Vehicle	Pedestrian	Cyclist	Car	Pedestrian	Cyclist
only Waymo	Voxel-RCNN [4] (w/o P.T.)	75.08 / 74.60	75.17 / **68.76**	65.28 / 64.33	34.10 / 17.31	2.99 / 1.69	0.05 / 0.01
	Voxel-RCNN [4] (w/ P.T. on nuScenes)	**75.46 / 74.99**	74.58 / 68.06	**65.92 / 64.98**	34.34 / 21.95	2.84 / 1.57	0.09 / 0.02
only nuScenes	Voxel-RCNN [4] (w/o P.T.)	36.77 / 36.50	4.64 / 3.18	2.49 / 2.45	53.63 / 39.05	22.47 / 17.85	10.86 / 9.70
	Voxel-RCNN [4] (w/ P.T. on Waymo)	6.11 / 5.90	0.77 / 0.56	0.01 / 0.01	55.23 / 39.14	23.65 / 16.47	8.51 / 5.80
Waymo+nuScenes	Voxel-RCNN [4] (w/ D.M.)	66.67 / 66.23	60.36 / 54.08	52.03 / 51.25	51.40 / 31.68	15.04 / 9.99	5.40 / 3.87
	Voxel-RCNN [4] (w/ C.A.)	69.40 / 68.86	63.43 / 56.49	52.83 / 51.93	51.39 / 29.04	16.24 / 10.96	4.55 / 3.13
	Voxel-RCNN [4] (w/ C.A.+S.A.)	75.16 / 74.67	74.83 / 68.07	64.68 / 63.73	58.41 / 40.84	26.52 / 20.98	9.19 / 7.65
	Voxel-RCNN [4] (w/ C.A.+C.R.)	74.56 / 74.05	74.29 / 67.04	63.14 / 62.21	59.10 / **42.25**	29.86 / 23.76	14.46 / **12.73**
	Voxel-RCNN [4] (w/ C.A.+S.A.+C.R.)	75.26 / 74.77	**75.46** / 68.75	65.02 / 63.12	**60.18** / 42.23	**30.08 / 24.37**	**14.60** / 12.32
only Waymo	PV-RCNN [16] (w/o P.T.)	74.97 / 74.46	73.41 / 66.57	**64.58 / 63.49**	32.99 / 17.55	3.34 / 1.94	0.02 / 0.01
	PV-RCNN [16] (w/ P.T. on nuScenes)	74.77 / 74.26	73.32 / 66.31	64.06 / 63.05	33.86 / 17.47	2.88 / 1.53	0.04 / 0.01
only nuScenes	PV-RCNN [16] (w/o P.T.)	41.01 / 40.58	4.57 / 2.96	0.98 / 0.95	57.78 / 41.10	24.52 / 18.56	10.24 / 8.25
	PV-RCNN [16] (w/ P.T. on Waymo)	44.59 / 44.24	7.67 / 6.33	8.77 / 8.58	57.92 / 41.53	24.32 / 17.31	11.52 / 9.19
Waymo+nuScenes	PV-RCNN [16] (w/ D.M.)	66.22 / 65.75	55.41 / 49.29	56.50 / 55.48	48.67 / 30.43	12.66 / 8.12	1.67 / 1.04
	PV-RCNN [16] (w/ C.A.)	66.90 / 65.61	56.41 / 51.06	56.00 / 55.00	48.93 / 31.21	14.47 / 10.31	1.70 / 1.07
	PV-RCNN [16] (w/ C.A.+S.A)	74.24 / 73.71	67.38 / 60.79	60.20 / 59.16	59.49 / 42.05	**27.44** / 20.94	12.69 / 10.34
	PV-RCNN [16] (w/ C.A.+C.R.)	74.88 / 74.36	73.39 / 66.02	62.84 / 61.79	59.01 / 41.16	26.59 / 20.49	9.86 / 7.60
	PV-RCNN [16] (w/ C.A.+S.A.+C.R.)	**75.54 / 74.90**	**74.12 / 66.90**	63.28 / 62.12	**60.77 / 42.66**	**27.44 / 21.85**	**13.50 / 11.87**

Table 3. Results of joint training on Waymo and nuScenes datasets. Following the existing 3D object detection works [16, 28, 29], we report the car (Vehicle on Waymo), pedestrian, and cyclist results under IoU threshold of 0.7, 0.5, and 0.5, respectively, and utilize AP and APH of LEVEL 1 metric on Waymo, and $\mathrm{AP}_{\mathrm{BEV}}$ and $\mathrm{AP}_{\mathrm{3D}}$ over 40 recall positions on nuScenes. The best detection results are marked using **bold**. Due to the page limitation, the average accuracy of multiple datasets is reported in Appendix.

Trained on	Baseline Detectors	Tested on KITTI			Tested on nuScenes		
		Car	Pedestrian	Cyclist	Car	Pedestrian	Cyclist
only KITTI	Voxel-RCNN [4] (w/o P.T.)	89.34 / 80.91	59.67 / 56.88	61.10 / 60.49	11.37 / 4.64	0.15 / 0.11	0.01 / 0.00
	Voxel-RCNN [4] (w/ P.T. on nuScenes)	89.90 / 81.25	59.49 / 56.17	54.55 / 54.15	12.89 / 5.52	0.24 / 0.18	0.05 / 0.03
only nuScenes	Voxel-RCNN [4] (w/o P.T.)	69.41 / 33.48	28.06 / 19.20	0.44 / 0.43	53.63 / 39.05	22.47 / 17.85	10.86 / 9.70
	Voxel-RCNN [4] (w/ P.T. on KITTI)	71.61 / 40.64	39.67 / 29.99	7.29 / 6.88	53.57 / 39.65	24.93 / 21.17	11.42 / 9.95
KITTI+nuScenes	Voxel-RCNN [4] (w/ D.M.)	89.24 / 73.72	61.03 / 54.55	62.71 / 59.92	41.88 / 20.48	12.58 / 8.32	1.77 / 0.97
	Voxel-RCNN [4] (w/ C.A.)	89.35 / 76.77	59.01 / 53.67	43.45 / 42.41	49.95 / 28.43	16.63 / 11.93	3.84 / 3.12
	Voxel-RCNN [4] (w/ S.A.)	89.21 / 82.68	62.32 / 57.99	63.10 / 61.67	57.87 / 40.23	27.21 / 21.44	13.65 / 12.24
	Voxel-RCNN [4] (w/ C.R.)	89.13 / 82.50	61.45 / 56.65	61.72 / 58.66	58.13 / 40.26	27.27 / 21.50	13.81 / 12.18
	Voxel-RCNN [4] (w/ S.A.+C.R.)	**90.09 / 83.10**	**62.99 / 58.30**	**70.20 / 68.10**	**59.25 / 41.51**	**29.12 / 23.18**	**15.16 / 13.16**
only KITTI	PV-RCNN [16] (w/o P.T.)	89.41 / 83.15	59.09 / 54.73	62.25 / 61.71	6.58 / 2.54	0.22 / 0.16	0.03 / 0.01
	PV-RCNN [16] (w/ P.T. on nuScenes)	89.26 / 83.14	**60.56 / 55.90**	63.60 / 62.88	13.43 / 5.61	0.69 / 0.27	0.04 / 0.00
only nuScenes	PV-RCNN [16] (w/o P.T.)	74.37 / 36.54	39.30 / 29.07	0.58 / 0.55	57.78 / 41.10	24.52 / 18.56	10.24 / 8.25
	PV-RCNN [16] (w/ P.T on KITTI)	69.40 / 38.25	33.24 / 24.88	1.68 / 1.61	53.24 / 36.72	20.65 / 17.09	8.95 / 7.58
KITTI+nuScenes	PV-RCNN [16] (w/ D.M.)	87.79 / 77.95	55.52 / 48.29	59.15 / 55.10	41.29 / 21.57	10.21 / 7.08	1.23 / 1.15
	PV-RCNN [16] (w/ C.A.)	88.53 / 77.20	47.13 / 39.53	44.22 / 41.64	46.34 / 25.28	12.70 / 9.64	2.18 / 1.34
	PV-RCNN [16] (w/ S.A.)	87.51 / 78.13	56.13 / 49.21	61.22 / 58.49	56.93 / 40.11	20.15 / 15.33	10.19 / 8.73
	PV-RCNN [16] (w/ C.R.)	**90.93** / 83.56	58.96 / 55.78	60.92 / 58.13	57.76 / 41.31	24.65 / 18.96	12.19 / 10.13
	PV-RCNN [16] (w/ S.A.+C.R.)	89.77 / **85.49**	60.03 / 55.58	**69.03 / 66.10**	**59.08 / 41.67**	**25.27 / 19.26**	**12.26 / 10.83**

Table 4. Results of joint training on KITTI and nuScenes datasets. The experiment and evaluation settings follow Table 3.

is equal to 30. Besides, for the experiments on Waymo-KITTI and nuScenes-KITTI consolidations, the weight decay is set to 0.01, and for the remaining experiments, the weight decay is set to 0.001. For Waymo dataset, we only use the uniformly-sampled 20% frames (about 32k frames) for model training.

4.2. Design of Comparison Baselines

1) w/o P.T.(Single-dataset): We employ the off-the-shelf 3D detectors, *e.g.*, Voxel-RCNN [4] and PV-RCNN [16], as the baseline detection model, which is trained from scratch and evaluated within a single dataset.
2) P.T.(Pre-training): Since the MDF setting allows the detector to access the annotated data from both datasets, we first pre-train the baseline detector on another dataset, and fine-tune the detector on the current dataset.

3) D.M.(Direct Merging): By simply combing multiple 3D datasets into a merged dataset, a single-dataset baseline detector is able to train from the merged dataset using a common detection loss, which can be regarded as a direct method to verify whether the existing 3d models can be improved under the directly-merged datasets. **Note that** for such baseline, we align the point cloud range and merge the label space of each dataset for multi-dataset training.

4) C.A.: Coordinate-origin Alignment baseline is designed to alleviate the sensor installation position differences. As previously described, to train detectors from multiple 3D datasets with inconsistent point cloud ranges, we **have to** align the point-cloud-range of all datasets. But such a point-range-level alignment operation would cause the distribution variations in coordinate origin and center point of ob-

Trained on	Baseline Detectors	Tested on KITTI			Tested on Waymo		
		Car	Pedestrian	Cyclist	Vehicle	Pedestrian	Cyclist
only KITTI	Voxel-RCNN [4] (w/o P.T.)	89.34 / 80.91	59.67 / 56.88	61.10 / 60.49	6.81 / 6.75	16.52 / 13.65	14.74 / 14.00
	Voxel-RCNN [4] (w/ P.T. on Waymo)	89.51 / 81.41	60.30 / 57.10	55.53 / 51.34	8.70 / 8.62	19.14 / 16.01	21.87 / 20.83
only Waymo	Voxel-RCNN [4] (w/o P.T.)	67.07 / 19.80	**65.44 / 61.92**	59.48 / 54.10	**75.08 / 74.60**	**75.17 / 68.76**	65.28 / 64.33
	Voxel-RCNN [4] (w/ P.T. on KITTI)	64.84 / 19.99	62.58 / 59.01	56.44 / 49.43	72.76 / 72.26	72.42 / 64.94	63.27 / 62.23
KITTI+Waymo	Voxel-RCNN [4] (w/ D.M.)	74.53 / 32.11	60.11 / 54.85	59.69 / 55.94	74.35 / 73.85	74.80 / 68.39	64.87 / 63.95
	Voxel-RCNN [4] (w/ S.A.+C.R.)	**90.03 / 82.39**	62.51 / 57.01	**69.52 / 66.30**	74.83 / 74.33	74.79 / 68.24	**66.83 / 65.82**
only KITTI	PV-RCNN [16] (w/o P.T.)	89.41 / 83.15	59.09 / 54.73	62.25 / 61.71	2.98 / 2.94	7.99 / 6.56	5.84 / 5.54
	PV-RCNN [16] (w/ P.T. on Waymo)	89.40 / **83.42**	62.69 / 58.86	59.96 / 59.43	8.75 / 8.64	12.12 / 9.90	9.20 / 8.76
only Waymo	PV-RCNN [16] (w/o P.T.)	56.20 / 54.81	60.04 / 57.06	54.29 / 50.05	74.97 / 74.46	**73.41 / 66.57**	**64.58 / 63.49**
	PV-RCNN [16] (w/ P.T. on KITTI)	69.25 / 25.91	59.16 / 55.92	56.09 / 50.50	71.08 / 70.54	70.12 / 62.91	62.37 / 61.40
KITTI+Waymo	PV-RCNN [16] (w/ D.M.)	87.49 / 68.35	**62.84 / 60.06**	68.09 / 65.75	50.68 / 50.31	58.76 / 52.59	55.14 / 54.17
	PV-RCNN [16] (w/ S.A.+C.R.)	**89.42** / 83.15	60.85 / 57.49	**71.61 / 65.88**	**75.07 / 74.54**	72.95 / 66.08	63.80 / 62.92

Table 5. Results of joint training on KITTI and Waymo. The experiment and evaluation settings follow Table 3.

Trained on	Tested on K	Tested on N	Tested on W	Avg. on KNW
K	89.34 / 80.91	11.37 / 4.64	6.81 / 6.75	35.84 / 30.77
N	69.41 / 33.48	53.63 / 39.05	36.77 / 36.50	53.27 / 36.34
W	67.07 / 19.80	34.10 / 17.31	75.08 / 74.60	58.75 / 37.23
K+N+W (Uni3D)	**89.65 / 83.41**	**60.42 / 42.30**	**75.47 / 74.97**	**75.18 / 66.89**

Table 6. Results for car class of jointly train on K (denoting KITTI), N (denoting nuScenes), and W (denoting Waymo) using Voxel-RCNN [4], and Avg. denotes the average detection accuracy evaluated on all the three datasets.

jects. To this end, following previous 3D cross-dataset research [24,28,29], the coordinate origin of different datasets needs to be transferred to the ground plane. For KITTI and nuScenes datasets, the coordinate shift is set to 1.6m and 1.8m along the height direction. We use the C.A. baseline to check the effectiveness of the existing origin alignment method on multi-dataset 3D object detection.

5) S.A.: As introduced in Sec. 3.3, Statistics-level Alignment baseline aims to reduce the statistical distribution (*e.g.*, μ^j and σ^j) differences of the learned features between datasets.

6) C.R.: Coupling-and-Recoupling baseline tries to mine reusable dataset-agnostic representations across multiple datasets, and outputs the dataset-specific semantic representations for better single-dataset detection accuracy.

Evaluation Metric. We adopt the officially released evaluation tools for evaluating our all baselines, where for KITTI and nuScenes datasets, the AP in both the Bird's Eye View (BEV) and 3D over 40 recall positions are reported, and for Waymo dataset, we employ the Average Precision (AP) and Average Precision re-weighted by Heading (APH) of each class for model evaluation. We report the moderate case results for KITTI dataset, and LEVEL_1 metric on Waymo dataset, and please refer to Appendix for the results using LEVEL_2 metric. AP is evaluated under an IoU threshold of 0.7 for car category (Vehicle on Waymo) and 0.5 for pedestrian and cyclist classes. In this paper, all experimental results are reported on the official validation set.

4.3. Results of Multi-Dataset 3D Object Detection

Results on Waymo-nuScenes, nuScenes-KITTI, Waymo-KITTI Consolidations. To investigate the feasibility of training 3D baseline detectors from multiple public datasets, we conduct experiments by selecting two representative 3D datasets from three widely-used autonomous driving datasets: Waymo [21], KITTI [5], and nuScenes [1]. According to the results from Table 3 to 5, we can observe the following five important findings:

1) Large gap between 3D datasets: We first train the baseline detector only on a single dataset (*e.g.*, Waymo or nuScenes), and evaluate this well-trained baseline on two different datasets (*e.g.*, Waymo and nuScenes). As can be seen in Table 3, the baseline performs well only on its original training dataset (*e.g.*, 75.08% AP for Vehicle). When the baseline detector is deployed to nuScenes dataset, its detection accuracy is seriously degraded (only 34.34% AP for Car). This is mainly because the single-dataset detection model is overfitted to its training dataset, yet fails to consider the source-to-target dataset shift. The same accuracy drop issue can also be observed on another baseline detector such as PV-RCNN [16].

2) Pre-trained model cannot work well under MDF setting: Another way that simultaneously improves the detection accuracy of the baseline detector for Waymo and nuScenes is fully-supervised pre-training. Such a way means that we first pre-train the baseline on the fully-labeled nuScenes (or Waymo), and fine-tune the well-trained model on Waymo (or nuScenes). By comparing P.T and w/o P.T. baselines, we observe from Table 3 that, although the model has been pre-trained on nuScenes, the detection accuracy for nuScenes is still unsatisfactory, which is due to that the model has been fine-tuned to Waymo, forgetting the knowledge learned from the previous pre-trained dataset.

3) 3D single-dataset training paradigm cannot work well under MDF setting: By comparing w/o P.T. and D.M baselines from Table 3, it can be observed that the typical 3D detectors (*e.g.*, Voxel-RCNN [4] and PV-RCNN [16]) trained on the merged dataset cannot achieve a high detection accuracy on both datasets.

4) Coordinate-origin alignment boosts the MDF accuracy: By comparing C.A. and D.M. baselines, it can be concluded that coordinate-origin shift scheme can reduce the negative impact caused by the point-cloud-range alignment

Trained on	Baseline Detectors	#nuScenes	Tested on KITTI			Tested on nuScenes		
			Car	Pedestrian	Cyclist	Car	Pedestrian	Cyclist
only nuScenes	Voxel-RCNN [4]	100%	-	-	-	53.63 / 39.05	22.47 / 17.85	10.86 / 9.70
only nuScenes	Voxel-RCNN [4]	10%	-	-	-	45.42 / 31.09	10.39 / 7.16	1.55 / 0.89
only nuScenes	Voxel-RCNN [4]	5%	-	-	-	30.01 / 16.15	4.70 / 2.56	0.06 / 0.05
only nuScenes	Voxel-RCNN [4]	1%	-	-	-	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00
KITTI+nuScenes	Voxel-RCNN [4] (ours)	100%	90.09 / 83.10	62.99 / 58.30	70.20 / 68.10	59.25 / 41.51	29.12 / 23.18	15.16 / 13.16
	Voxel-RCNN [4] (ours)	10%	88.81 / 81.75	60.09 / 56.61	70.03 / 68.54	52.08 / 34.40	20.40 / 15.60	8.42 / 7.40
	Voxel-RCNN [4] (ours)	5%	89.10 / 81.86	59.17 / 54.42	73.30 / 70.25	51.81 / 34.43	19.82 / 14.94	5.52/ 4.58
	Voxel-RCNN [4] (ours)	1%	89.06 / 81.55	56.74 / 52.28	71.11 / 69.06	44.74 / 28.28	15.94 / 11.11	1.28 / 0.99
only nuScenes	PV-RCNN [16]	100%	-	-	-	57.78 / 41.10	24.52 / 18.56	10.24 / 8.25
only nuScenes	PV-RCNN [16]	10%	-	-	-	50.39 / 31.68	13.64 / 8.75	0.85 / 0.51
only nuScenes	PV-RCNN [16]	5%	-	-	-	35.87 / 19.76	5.89 / 3.15	0.00 / 0.00
only nuScenes	PV-RCNN [16]	1%	-	-	-	0.08 / 0.01	0.02 / 0.01	0.00 / 0.00
KITTI+nuScenes	PV-RCNN [16] (ours)	100%	89.77 / 85.49	60.03 / 55.58	69.03 / 66.10	59.08 / 41.67	25.27 / 19.26	12.26 / 10.83
	PV-RCNN [16] (ours)	10%	88.99 / 83.12	57.06 / 52.48	71.14 / 70.60	51.75 / 33.85	15.60 / 10.78	3.33 / 2.09
	PV-RCNN [16] (ours)	5%	88.95 / 82.83	56.62 / 53.25	71.99 / 69.86	50.32 / 34.35	16.11 / 11.20	2.59 / 2.00
	PV-RCNN [16] (ours)	1%	88.92 / 82.81	55.22 / 51.84	71.12 / 69.73	41.09 / 25.38	11.27 / 7.00	0.60 / 0.33

Table 7. Results of reducing the number of samples in nuScenes dataset under the nuScenes-KITTI consolidation setting.

operation. Besides, it should be noted that for KITTI-nuScenes consolidation setting, applying the coordinate-origin shift operation for raw point clouds yields a severe detection accuracy drop of the Pedestrian and Cyclist classes in KITTI datasets. This may be due to that the preset coordinate-origin shift parameters are shared across different classes, which is sensitive to correct the distributions for the classes with few-shot samples, such as Pedestrian and Cyclist. Thus, sharing the same parameter of coordinate-origin shift between classes is harmful to some scenarios, and needs to be further studied in our future work.

5) The effectiveness and generality of each designed module: As reported in Tables 3, 4, and 5, the average detection results achieved by Uni3D exceed those of all designed baselines, verifying the effectiveness of Uni3D in learning from multiple 3D datasets. Furthermore, we also conduct experiments by selecting PV-RCNN [16] as another baseline detector and repeat the above experiments, observing consistent detection accuracy gains.

Results on Waymo-KITTI-nuScenes Consolidation. Table 6 shows the results of jointly training the Voxel-RCNN [4] from Waymo, nuScenes, and KITTI. Also, Uni3D achieves high detection results simultaneously on multiple datasets.

4.4. Further Analyses

Uni3D: Reduce the Data Acquisition Cost. In real application, we may not be able to collect massive LiDAR data for a new scene due to the expensive data acquisition cost. Uni3D gives another option for addressing such a dilemma, namely, training on the combined set between the few-shot data from the new scene and the full data from the previous well-constructed dataset. It can be seen from Table 7 that the detection accuracy on nuScenes dataset achieved by Uni3D outperforms the one only using few-shot nuScenes. The improvement mainly comes from the ability of Uni3D to learn more generalizable features, which is less prone to over-fitting under few-shot samples.

Methods	Baseline Models	Pre-trained on	Tested on KITTI AP_{BEV} / AP_{3D}
Source-only	PV-RCNN	Waymo	61.18 / 22.01
Source-only	PV-RCNN	nuScenes	68.15 / 37.17
Source-only	PV-RCNN (ours)	**Waymo+nuScenes**	**73.51 / 39.71**
ST3D [28] (w/ SN)	PV-RCNN	Waymo	86.65 / 76.86
ST3D [28] (w/ SN)	PV-RCNN	nuScenes	84.29 / 72.94
ST3D [28] (w/ SN)	PV-RCNN (ours)	**Waymo+nuScenes**	**88.25 / 77.01**

Table 8. Generalization study from two aspects including: 1) Zero-shot detection accuracy on KITTI, and 2) Model adaptability coupled with the off-the-shelf UDA method (ST3D [28]). Source-only denotes that the model is trained on the source domain and directly tested on the target domain.

Uni3D: Strengthen the Zero-shot and Domain Adaptation Ability. Another advantage of Uni3D is that it can largely boost the zero-shot learning ability of the baseline detector, by learning generalizable features from multiple datasets. As shown in Table 8, by utilizing the pre-trained model provided by Uni3D, the zero-shot inference accuracy is significantly improved (from $68.15\% AP_{BEV}$ to $73.51\% AP_{BEV}$). One possible reason for such improvement is that Uni3D learns about the potential inter-dataset variations by jointly training on Waymo and nuScenes, and the Waymo-to-nuScenes dataset variations are beneficial to recognize an unforeseen domain.

5. Conclusion

For the first time, we study how to train a unified 3D detection model using the off-the-shelf public 3D benchmarks, and present a unified 3D detection framework (Uni3D) which can be easily combined with the existing 3D detectors. We conduct extensive experiments on many public benchmarks, and the results show the effectiveness of Uni3D in obtaining dataset-level generalizable features.

Acknowledgement

This work is supported by Science and Technology Commission of Shanghai Municipality (grant No. 22DZ1100102).

References

[1] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 1, 2, 3, 4, 5, 7

[2] 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*, pages 1907–1915, 2017. 1, 2

[3] Xiyang Dai, Yinpeng Chen, Bin Xiao, Dongdong Chen, Mengchen Liu, Lu Yuan, and Lei Zhang. Dynamic head: Unifying object detection heads with attentions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7373–7382, 2021. 2

[4] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1201–1209, 2021. 1, 2, 3, 4, 5, 6, 7, 8

[5] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012. 1, 2, 3, 5, 7

[6] Rui Gong, Dengxin Dai, Yuhua Chen, Wen Li, and Luc Van Gool. mdalu: Multi-source domain adaptation and label unification with partial datasets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8876–8885, 2021. 2

[7] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 5

[8] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015. 4

[9] John Lambert, Zhuang Liu, Ozan Sener, James Hays, and Vladlen Koltun. Mseg: A composite dataset for multi-domain semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2879–2888, 2020. 2

[10] 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*, pages 12697–12705, 2019. 1, 2, 3

[11] Zhipeng Luo, Zhongang Cai, Changqing Zhou, Gongjie Zhang, Haiyu Zhao, Shuai Yi, Shijian Lu, Hongsheng Li, Shanghang Zhang, and Ziwei Liu. Unsupervised domain adaptive 3d detection with multi-level consistency. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8866–8875, 2021. 1

[12] 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 conference on computer vision and pattern recognition*, pages 918–927, 2018. 2

[13] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2

[14] 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*, 30, 2017. 2

[15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 2

[16] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020. 1, 2, 3, 4, 5, 6, 7, 8

[17] Shaoshuai Shi, Li Jiang, Jiajun Deng, Zhe Wang, Chaoxu Guo, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn++: Point-voxel feature set abstraction with local vector representation for 3d object detection. *arXiv preprint arXiv:2102.00463*, 2021. 1, 3

[18] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 770–779, 2019. 2

[19] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *IEEE transactions on pattern analysis and machine intelligence*, 43(8):2647–2664, 2020. 2, 3

[20] Vishwanath A Sindagi, Yin Zhou, and Oncel Tuzel. Mvx-net: Multimodal voxelnet for 3d object detection. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7276–7282. IEEE, 2019. 2

[21] 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*, pages 2446–2454, 2020. 1, 2, 3, 4, 5, 7

[22] OpenPCDet Development Team. Openpcdet: An open-source toolbox for 3d object detection from point clouds. `https://github.com/open-mmlab/OpenPCDet`, 2020. 5

[23] Xudong Wang, Zhaowei Cai, Dashan Gao, and Nuno Vasconcelos. Towards universal object detection by domain attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7289–7298, 2019. 2

[24] Yi Wei, Zibu Wei, Yongming Rao, Jiaxin Li, Jie Zhou, and Jiwen Lu. Lidar distillation: Bridging the beam-

induced domain gap for 3d object detection. *arXiv preprint arXiv:2203.14956*, 2022. 1, 7

[25] Qiangeng Xu, Yin Zhou, Weiyue Wang, Charles R Qi, and Dragomir Anguelov. Spg: Unsupervised domain adaptation for 3d object detection via semantic point generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15446–15456, 2021. 1

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

[27] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7652–7660, 2018. 2

[28] Jihan Yang, Shaoshuai Shi, Zhe Wang, Hongsheng Li, and Xiaojuan Qi. St3d: Self-training for unsupervised domain adaptation on 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10368–10378, 2021. 1, 3, 6, 7, 8

[29] Jihan Yang, Shaoshuai Shi, Zhe Wang, Hongsheng Li, and Xiaojuan Qi. St3d++: Denoised self-training for unsupervised domain adaptation on 3d object detection. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (01):1–17, 2022. 1, 3, 6, 7

[30] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Std: Sparse-to-dense 3d object detector for point cloud. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1951–1960, 2019. 2

[31] 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*, pages 11784–11793, 2021. 1

[32] Yifan Zhang, Qingyong Hu, Guoquan Xu, Yanxin Ma, Jianwei Wan, and Yulan Guo. Not all points are equal: Learning highly efficient point-based detectors for 3d lidar point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18953–18962, 2022. 2

[33] Xiangyun Zhao, Samuel Schulter, Gaurav Sharma, Yi-Hsuan Tsai, Manmohan Chandraker, and Ying Wu. Object detection with a unified label space from multiple datasets. In *European Conference on Computer Vision*, pages 178–193. Springer, 2020. 2

[34] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6881–6890, 2021. 2

[35] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Simple multi-dataset detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7571–7580, 2022. 2

[36] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4490–4499, 2018. 1, 2

[37] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 2