Detecting As Labeling: Rethinking LiDAR-camera Fusion in 3D Object Detection

Junjie Huang*, Yun Ye, Zhujin Liang, Yi Shan, and Dalong Du

PhiGent Robotics
{junjie.huang, yun.ye, zhujin.liang, yi.shan, dalong.du}@phigent.ai

Abstract. 3D object Detection with LiDAR-camera encounters overfitting in algorithm development derived from violating some fundamental rules. We refer to the data annotation in dataset construction for theoretical optimization and argue that the regression task prediction should not involve the feature from the camera branch. Following the cutting-edge perspective of 'Detecting As Labeling', we propose a novel paradigm dubbed DAL. With the most classical elementary algorithms, a simple predicting pipeline is constructed by imitating the data annotation process. Then we train it in the simplest way to minimize its dependency and strengthen its portability. Though simple in construction and training, the proposed DAL paradigm not only substantially pushes the performance boundary but also provides a superior trade-off between speed and accuracy among all existing methods. With comprehensive superiority, DAL is an ideal baseline for both future work development and practical deployment. The code has been released to facilitate future work `https://github.com/HuangJunJie2017/BEVDet`.

Keywords: 3D Object Detection · Multi-modality Fusion · Autonomous Driving

1 Introduction

With superior robustness, 3D object detection with LiDAR-camera fusion plays an important role in robotics. Fueled by the vision of autonomous driving, many efforts [1, 8, 15, 22, 33, 42, 44, 47] have been devoted to this topic in the past few years. Nevertheless, some fundamental rules are violated in all these existing methods, which makes them struggle with overfitting. As a remedy, complicated training strategies are applied along with the existing algorithms as listed in Tab 1. They use multiple training stages or customized learning rate policies to alleviate this problem. However, what they achieve is just a local optimum that will trap future work to a large extent, as the more customized the training strategy is the less its compatibility to the future modification. Besides, additional dependence (`e.g.` pre-training on other datasets) also gives rise to extra cost and uncertainty in practice.

* Corresponding author.

Table 1: Comparing the DAL training pipeline with the existing methods. The reported epoch numbers do not involve the pre-training stages. The training pipeline of DAL is the simplest one among all leading methods. It means minimum dependency and maximum portability.

Methods	Camera Pre.	LiDAR Pre.	Custom Learning Rate	Epochs	NDS
UVTR [15]	ImageNet → nuScenes	-	✓	20	70.4
BEVFusion [33]	ImageNet → nuScenes	TransFusion-L		10	72.1
BEVFusion(MIT) [22]	ImageNet → nuImages	TransFusion-L		6	71.4
TransFusion [47]	ImageNet → COCO	TransFusion-L		6	71.7
DeepIteraction [47]	ImageNet → COCO → nuImages	TransFusion-L		6	72.6
CMT-VOV [44]	ImageNet → DD3D → nuScenes	-	✓	20	72.9
SparseFusion [42]	ImageNet → nuImages	TransFusion-L	✓	6	72.8
UniTR [8]	ImageNet → nuImages	-		10	73.3
DAL-Large	ImageNet	-		20	**74.0**

We rethink LiDAR-camera fusion in 3D object detection by referring to the data annotation phase in dataset construction [2]. In this phase, annotations are generated in two steps: candidates are searched for in images and point cloud at first, and the 3D bounding boxes are annotated according to the LiDAR points of each instance (`i.e.` tightly includes all LiDAR points). To guarantee high quality in labeling, some rules should be strictly obeyed by the annotators:

A. Images should be incorporated with point cloud to search all possible candidates and determine their categories.
B. The 3D bounding box of each instance should be generated only according to the point cloud when the point cloud is sufficiently complete for locating the edges of the bounding box.

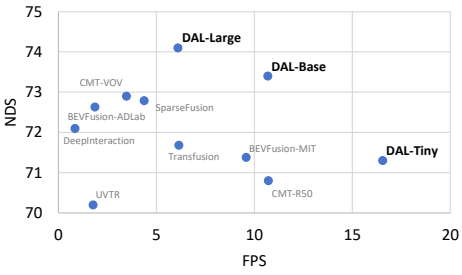

Fig. 1: Trade off between performance and inference speed on the nuScenes [2] val set. For a fair comparison, we applied the same accelerated voxelization module to all algorithms and reported their inference speed after acceleration.

Rule.B demands a priority disparity between the point cloud and the image in annotating the 3D bounding boxes. This is derived from the different robustness of these two patterns in this sub-task: The point cloud is an infallible ruler while the vision is merely an experienced gambler. The ill-posed monocular depth estimation problem makes the vision inevitably not robust on this topic. Violating Rule.B and intuitively involving the image fea-

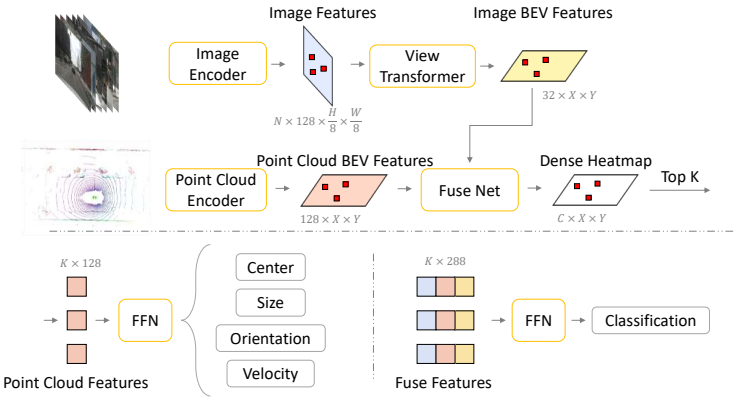

Fig. 2: The prediction pipeline of DAL paradigm. BEV features from the image and point cloud are fused to generate the dense heatmap. Top K proposals and their point cloud features are extracted for regression task prediction. Features fused with the image feature, the image BEV feature, and the point cloud BEV feature are used for category prediction. The sparse image feature is extracted according to the corresponding predicted center of each proposal.

tures in predicting the regression targets makes the existing methods fall into the dilemma of overfitting.

Following the perspective of 'Detecting As Labeling', we construct a demonstrative framework, dubbed DAL, to reveal the value of this theory to this problem. We only use the most classical elements to build its elegant predicting pipeline. Then we train it in the simplest manner to minimize its dependency and strengthen its portability. In addition, we notice that the planning task in autonomous driving takes advantage of the movable objects' velocity for active safety. However, the practical data always has an extremely imbalanced velocity distribution, which degenerates the perception algorithm's performance on this topic. This inspires us to develop a velocity augmentation strategy to alleviate this problem. Last but not least, we also propose a cutting-edge implementation of the classical Lift-Splat-Shoot [26] view transformation algorithm, dubbed BEVPoolV2. Compared with the previous works [16,17,22,26], BEVPoolV2 has a faster inference speed and far less memory occupancy, which is all crucial for deploying on edge devices with limited resources and strong requirement of time-efficiency.

As a result, though simple in construction and training, the proposed DAL paradigm not only substantially pushes the performance boundary (e.g. 74.0 NDS on the nuScenes `val` set and 74.8 NDS on the nuScenes `test` set) but also provides a superior trade-off between speed and accuracy among all existing methods.

2 Related Work

3D Object Detection with LiDAR-camera Fusion As complementary signals, LiDAR-camera fusion offers appealing performance in 3D object detection. It has been a long literature for engineers developing various works on this topic. The previous works can be roughly classified into three paradigms: Early works [5, 29, 34, 36, 40, 41] prefer to strengthen the point cloud in aspects of amount complement and feature enhancement. Recently, inspired by the successes of camera-only 3D object detection in BEV [12,13,17,25], some works [6,15,22,33] focus on feature fusion in a unified space. Following the mechanism of attention, some other works [1, 3, 8, 42, 44] transfer all modalities into tokens and always predict targets in a sparse paradigm [4]. They update the features of sparse proposals by applying attention calculation with both the image features and the point cloud features. DAL focuses on the rectification of fundamental theory. It absorbs some excellent features from the previous works to construct itself. And in turn, it is a promising direction for revising the existing works. Besides, as an impressive milestone, it is also an excellent baseline for future work and practical usage.

Implementation of LSS View Transformation Lift-Splat-Shoot (LSS) [26] is a view transformation algorithm that has been widely used in many tasks in autonomous driving (`e.g.` 3D object detection [13, 17, 22, 25, 27], BEV segmentation [26], and occupancy prediction [18]). With robust performance [2], it plays a fundamental role in incorporating the vision modality in 3D perception. However, the original implementation [26] with a slow inference speed and a tremendous memory occupancy may make people undervalue it in practice. Some pioneers [16, 17, 22] have accelerated this algorithm by applying parallel computing in voxelization. However, their implementations can only maintain a fast inference speed in a low input resolution and degenerate rapidly when the input resolution increases. Besides, tremendous memory occupancy is still inevitable. Based on the BEVPool in BEVFusion [22], We propose an upgraded version dubbed BEVPoolV2 alone with DAL to resolve the aforementioned problems.

3 Detecting As Labeling

3.1 Network

Following the perspective of 'Detecting As Labeling', we construct a predicting pipeline by imitating the data annotation process as illustrated in Fig. 2. The proposed pipeline follows the dense-to-sparse paradigm [1]. The dense perception stage focuses on feature encoding and candidate generating. The features from the images $F_I \in \mathbb{R}^{N\times 128\times H\times W}$ and the point cloud $F_P \in \mathbb{R}^{128\times X\times Y}$ are extracted with an image encoder and a point cloud encoder separately. N describes the number of views. $H \times W$ describes the size of the feature in the image

view. $X \times Y$ describes the size of the feature in Bird-Eye-View(BEV). The feature encoders have a classical structure with a backbone (`e.g.` ResNet [10] and VoxelNet [50]) and a neck (`e.g.` FPN [19] and SECOND [45]. Then the image feature is transformed from the image view to BEV with the classical view transform algorithm Lift-Splat-Shot(LSS) [26]. We just fuse the dense image BEV feature and point cloud BEV feature by concatenation and predicting the dense heatmap $H \in \mathbb{R}^{C \times X \times Y}$ by applying two extra residual blocks [10]. C describes the number of categories. Finally, K candidates with leading predicted scores in the dense heatmap are selected. In this way, we imitate the candidate-generating process in data annotation. Features from both image and point cloud are used in this process for a complete set of candidates.

In the sparse perception stage, the point cloud feature of each candidate is first gathered according to its coordinates in the dense heatmap. Then the regression targets (`e.g.` center, size, orientation, and velocity) are predicted with a simple Feed-Forward Network (FFN). No image feature is involved in this process to prevent the overfitting problem. Finally, we fuse the image feature, the image BEV feature, and the point cloud BEV feature to generate a fused feature for category prediction. The section from the image BEV feature is extracted according to the candidate's coordinates in the dense heatmap while the section from the image feature is extracted according to the predicted object center.

3.2 Training

As we have assigned tasks with proper modalities in constructing the predicting pipeline, we merely need to load the parameter of the image backbone pre-trained on ImageNet [7] like most classical vision tasks [20, 49]. Then we train DAL in an end-to-end pattern with only one stage. Only data from the target dataset nuScenes [2] is used. In this way, we train the DAL models in the most elegant manner which is rare in the literature.

Table 2: Details of the typical DAL predicting pipelines.

Component	Ablation	Tiny	Base	Large
Image Resolution	256x704	192x544	256x704	384x1056
Image Backbone	R50 [10]	R18 [10]	R18 [10]	R50 [10]
Image Neck	FPN [19]	FPN [19]	FPN [19]	FPN [19]
Voxel Resolution (Meter)	0.100	0.100	0.075	0.050
Sparse Encoder (Base Channels)	16	16	24	32
Dense Encoder (Stride-Blocks -Channels)	1-5-128 2-5-256	1-5-128 2-5-256	1-8-192 2-8-384	1-3-128 2-3-256 2-3-256
BEV Neck	SEC. [45]	SEC. [45]	SEC. [45]	SEC. [45]

As an example, DAL shares the same design of targets and losses with TransFusion [1] and BEVFusion [22]. Other than that, we add an auxiliary classification head upon the image features to strengthen the image branch in searching candidates and discriminating different categories. This is important for DAL as the supervisions from both dense and sparse perception phases in the 3D object detection

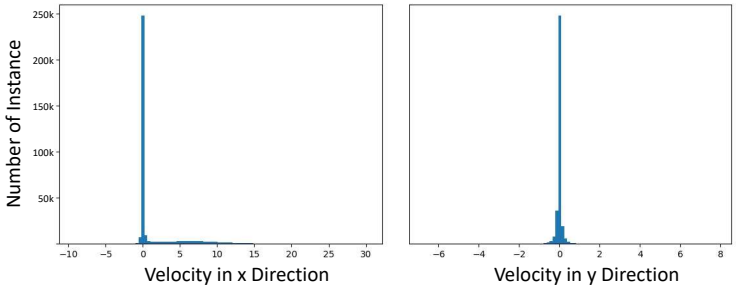

Fig. 3: The velocity distribution of the car category in nuScenes [2] train set.

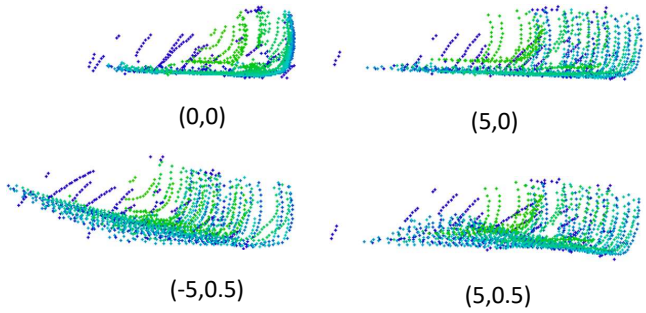

Fig. 4: Augmentation of the same static object with different pre-defined velocities (`i.e.` (v_x, v_y)).

head are defective. Specifically, in the dense perception phase, the image features are adjusted according to the predicted depth score in the view transformation. So is the gradient in back-propagation. A defective predicted depth score is inevitable and so is the supervision. In the sparse perception phase, instead of the image features of all annotated targets, only those of predicted instances are involved in the loss calculation. An auxiliary classification head with supervision from all annotated targets can resolve the aforementioned problems and strengthen the image branch to a certain extent. In practice, the gravity centers of annotated targets are used to extract the sparse feature of each annotated target. Then classification is conducted on the sparse feature with another FFN, and the loss is calculated just the same as the classification task in 3D object detection head [1]. Without re-weighting, we directly add the auxiliary loss to the existing losses:

$$L_{\text{DAL}} = L_{\text{aux}} + L_{\text{Transfusion}} \tag{1}$$

The deprecation of image features in regression task prediction not only prevents the inevitable performance degeneration but also enables a wider range of data augmentation in the image space. We take the resize augmentation as

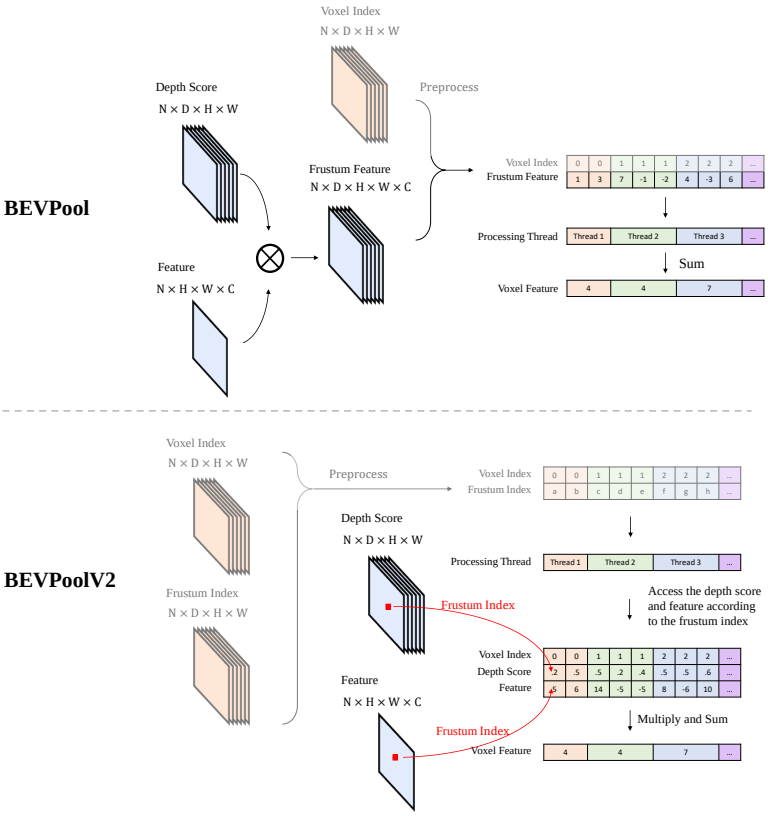

Fig. 5: Illustration of BEVPool from BEVFusion [22] and BEVPoolV2. The part in low transparency can be pre-computed offline.

an example for explanation. Camera-based 3D object detection predicts the size of the target according to its size in the image view. When the image is randomly resized, to maintain the consistency between the image features and the predicted targets, adjustment is needed to be conducted on the predicted target accordingly. Then is the point cloud in a chain reaction in 3D object detection with LiDAR-camera fusion. So existing methods always use a small range of data augmentation in the image space. This, as a result, keeps them away from the benefit of large-scale data augmentation in the image space like that in most image 2D tasks (*e.g.* classification [7], detection [20], segmentation [20].).

Last but not least, we observe extremely imbalanced velocity distribution of the training data. As illustrated in Fig. 3, most instances of car category in nuScenes [2] train set is static. To adjust the distribution, some static objects are randomly selected and their point clouds are adjusted according to a predefined velocity as illustrated in Fig. 4. We conduct velocity augmentation on static

objects only as a full set of its points from multiple LiDAR frames can be handily discriminated with its annotated bounding box.

3.3 BEVPoolV2

As illustrated in the left of Fig. 5, the pioneers [16,17,22,26] compute, store, and process the frustum feature which is required by the LSS [26] view transformation algorithm. The frustum feature is such a huge tensor that occupies a mass of memory and slows down the predicting pipeline. We upgrade the BEVPool implementation [22] by using the index of frustum points as a trace of them to be pre-processed alone with their voxel indexes. This shares the same idea with using the pointer in C++ programming. Then we access the value of the frustum feature in the processing thread according to the frustum index. In this way, we avoid explicitly computing, storing, and pre-processing the frustum feature. As a result, the memory and calculation can be saved and the inference can be sped up. In practice, both the voxel index and frustum index can be pre-computed and pre-processed offline.

4 Experiments

4.1 Implementation Details

Dataset We conduct comprehensive experiments on the large-scale benchmark nuScenes [2]. NuScenes is the up-to-date popular benchmark for verifying many outdoor tasks like 3D object detection [12, 13, 22, 24, 39, 48], occupancy prediction [32, 38], BEV semantic segmentation [23, 26, 28, 46], and End-to-end autonomous driving [11]. It includes 1000 scenes with images from 6 cameras and point clouds from a LiDAR with 32 beams. The camera group has a 360° view which is consistent with LiDAR. This makes it to be a preferable dataset for evaluating algorithms with LiDAR-camera fusion. The scenes are officially split into 700/150/150 scenes for training/validation/testing. There are up to 1.4M annotated 3D bounding boxes for 10 classes: car, truck, bus, trailer, construction vehicle, pedestrian, motorcycle, bicycle, barrier, and traffic cone.

Evaluation Metrics For 3D object detection, we report the official predefined metrics: mean Average Precision (mAP), Average Translation Error (ATE), Average Scale Error (ASE), Average Orientation Error (AOE), Average Velocity Error (AVE), Average Attribute Error (AAE), and NuScenes Detection Score (NDS). The mAP is analogous to that in 2D object detection [20] for measuring the precision and recall, but defined based on the match by 2D center distance on the ground plane instead of the Intersection over Union (IOU) [2]. NDS is the composite of the other indicators for comprehensively judging the detection capacity. The remaining metrics are designed for calculating the positive results' precision on the corresponding aspects (*e.g.*, translation, scale, orientation, velocity, and attribute).

Table 3: Comparisons on nuScenes `test` set. All present methods use a single model without any test time augmentation.

Method	Present at	mATE	mASE	mAOE	mAVE	mAAE	mAP	NDS
PointPainting [35]	CVPR'20	38.0	26.0	54.1	29.3	13.1	54.1	61.0
MVP [5]	CVPR'17	26.3	23.8	32.1	31.3	13.2	66.4	70.5
PointAugmenting [37]	CVPR'21	25.3	23.5	35.4	26.6	12.3	66.8	71.1
UVTR [15]	NeurIPS'22	30.6	24.5	35.1	22.5	12.4	67.1	71.1
FusionPainting [43]	ITSC'21	25.6	23.6	34.6	27.4	13.2	68.1	71.6
TransFusion [1]	CVPR'22	25.9	24.3	32.9	28.8	12.7	68.9	71.7
BEVFusion(MIT) [22]	ICRA'23	26.1	23.9	32.9	26.0	13.4	70.2	72.9
BEVFusion(ADLab) [33]	NeurIPS'22	25.0	24.0	35.9	25.4	13.2	71.3	73.3
ObjectFusion [3]	ICCV'23	-	-	-	-	-	71.0	73.3
DeepIteraction [47]	NeurIPS'22	25.7	24.0	32.5	24.5	12.8	70.8	73.4
SparseFusion [42]	ICCV'23	25.8	24.3	32.9	26.5	13.1	72.0	73.8
CMT [44]	ICCV'23	27.9	23.5	30.8	25.9	11.2	72.0	74.1
UniTR [8]	ICCV'23	24.1	22.9	25.6	24.0	13.1	70.9	74.5
DAL		25.3	23.9	33.4	17.4	12.0	72.0	**74.8**

Predicting Pipeline As illustrated in Tab. 2, we follow two classical 3D object detection paradigms BEVDet-R50 [13] and CenterPoint [48] to build the image branch and the LiDAR branch separately for ablation study. Besides, we also provide some recommended configurations with superior trade-offs between inference latency and accuracy.

Training and Evaluating The DAL models are trained with a batch size of 64 on 16 3090 GPUs. As listed in Tab. 1, different from most existing methods that require multiple pre-trained stages and complicated learning rate strategies, DAL loads the pre-trained weight from the ImageNet classification task only and trains the hold pipeline for a total of 20 epochs with CBGS [51]. DAL shares the same simple learning rate policy as CenterPoint [48]. Specifically, the learning rate is adjusted by following the cycle learning rate policy [30] with an initial value of 2.0×10^{-4}. During evaluation, we report the performance of a single model without test time augmentation. Inference speeds are all tested on a single 3090 GPU by default.

4.2 Benchmark Results

Results on the nuScenes `val` set. As listed in Tab. 4 and illustrated in Fig. 1, the proposed DAL paradigm not only substantially pushes the performance boundary, but also provides a better trade-off between speed and accuracy. Configuration DAL-Large scores 71.5 mAP and 74.0 NDS surpassing the best existing record by a large margin of +1.0 mAP and +0.7 NDS. With such high accuracy, DAL-Large still runs at an inference speed of 6.10 FPS. Another recommended configuration DAL-Base runs at a similar inference speed as the fastest method CMT-R50 [44]. Its accuracy surpasses CMT-R50 by a large margin of 2.1 mAP

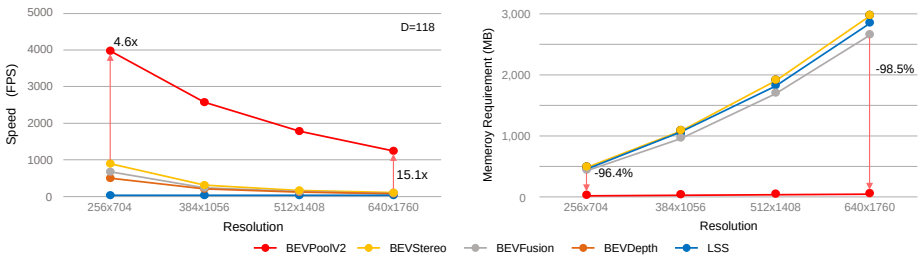

Fig. 6: Inference speed and memory requirement of different implementations of Lift-Splat-Shoot view transformation. D denotes the number of predefined depth values in LSS [26].

Table 4: Comparisons on nuScenes `val` set. FPS is measured on a 3090 GPU by default. + means modification is applied as listed in Tab. 2. † denotes the inference speed on A100 GPU.

Method	LiDAR	Camera	#Params(M)	FPS	mAP	NDS
CMT-R50 [44]	0100VoxelNet	320 × 800-R50	40.98	(14.2†)10.72	67.9	70.8
DAL-Tiny	0100VoxelNet	192 × 544-R18	21.21	(23.2†)16.55	67.4	71.3
UVTR [15]	0075VoxelNet+	900 × 1600-R101-DCN [14]	88.85	1.77	65.4	70.4
BEVFusion(MIT) [22]	0075VoxelNet	256 × 704-STTiny [21]	40.84	9.58	68.5	71.4
TransFusion [1]	0075VoxelNet	448 × 800-R50	37.03	6.51	68.9	71.7
ObjectFusion [3]	0075VoxelNet	256 × 704-STTiny [21]	-	-	69.8	72.3
DeepIteraction [47]	0075VoxelNet	448 × 800-R50	57.90	1.86	69.9	72.6
SparseFusion [42]	0075VoxelNet+	448 × 800-R50	40.16	4.38	70.5	72.8
CMT-VOV [44]	0075VoxelNet	640 × 1600-VOVNet	86.67	(6.0†)3.48	70.3	72.9
UniTR [8]	0030DSVT [9]	256 × 704-DSVT	15.56	(9.3†)	70.5	73.3
DAL-Base	0075VoxelNet+	256 × 704-R18	35.06	(15.2†)10.69	70.0	73.4
DAL-Large	0050VoxelNet+	384 × 1056-R50	47.77	(9.43†)6.10	**71.5**	**74.0**

and 2.6 NDS. With a similar accuracy as CMT-R50, DAL-Tiny offers an acceleration of 54%. On a more powerful device A100, the inference speed is up to 23.2 FPS and the acceleration is 63.3%.

Results on the nuScenes `test` set. We report the performance of configuration DAL-Large on the nuScenes `test` set without model ensemble and test time augmentation. DAL outshines all other approaches in terms of NDS 74.8.

4.3 Ablation Study

We use experiments illustrated in Tab. 5 to reveal the impact of the key designs. We take a modified version of BEVFusion as a baseline as detailed in Tab. 2. As a reference configuration, configuration A in Tab. 5 uses the LiDAR modality only. It cores 63.67 mAP and 69.00 NDS. In Tab. 5 configuration B, we train the BEVFusion paradigm as DAL. Though obvious improvement can be observed on

Table 5: Ablation study of some key design. All configurations are trained with the same pipeline as DAL in Tab. 1. Baseline BEVFusion is a modified version as listed in Tab. 2. 'Auxiliary' means using the auxiliary classification task. Resize denotes the range of scale in image resize augmentation with respect to the original image size. 'BEVFusion' denotes using BEVFusion-like predicting pipeline. '-L' denotes using the LiDAR modality only. Resize denotes the augmentation range adjustment from [0.36, 0.55] [13] to [0.36, 0.88].'-T' denotes the evaluation on the nuScenes train set.

	Pipeline	Auxiliary	Resize	Velocity Aug.	mAVE	mAP	NDS	mAP-T	mATE-T	NDS-T
A	BEVFusion-L				24.55	63.67	69.00	83.70	20.29	82.33
B	BEVFusion				25.64	63.59	68.71	92.98	16.49	87.83
C	BEVFusion	✓			25.30	63.45	68.99	93.00	16.58	87.71
D	BEVFusion	✓	✓		26.34	68.00	70.97	87.19	19.39	83.76
E	BEVFusion	✓	✓	✓	**19.26**	67.87	71.67	84.61	21.01	82.64
F	DAL	✓			24.13	64.16	69.52	91.83	18.58	86.14
G	DAL	✓	✓		25.80	68.07	70.87	87.70	19.64	84.05
H	DAL	✓	✓	✓	19.31	**68.50**	**71.94**	84.88	21.41	82.61

the train set, no superiority can be observed in the performance of this configuration (`i.e.` 63.59 mAP and 68.71 NDS) on the baseline on the validation set. This means that BEVFusion only overfits the training set with the image modality. The reported improvement in their technique report relies on a complicated pre-trained strategy as listed in Tab. 1. Directly loading pre-trained weight from the ImageNet classification task and training the hold pipeline is not feasible for BEVFusion.

With the DAL predicting pipeline and the auxiliary classification task, configuration F in Tab. 5 scores 64.16 mAP and 69.52 NDS mildly surpassing the LiDAR-only baseline (A) by +0.49 mAP and +0.52 NDS. This indicates that the DAL pipeline built upon the 'Detecting As Labeling' concept is a feasible solution to take advantage of the visual modality even with a simple training pipeline. By applying a large range of image resize augmentation in Tab. 5 configuration G, the accuracy is improved by a large margin of +3.91 mAP and +1.35 NDS to 68.07 mAP and 70.87 NDS. Freeing the image branch from regression tasks and making a large range of image resize augmentation feasible is another key factor of the DAL paradigm. In addition, velocity augmentation applied in Tab. 5 configuration H offers extra performance boosting of 0.47 mAP and 1.07 NDS. A more balanced velocity distribution in the training set significantly decreases the predicted error by 25%.

Last but not least, we use configurations C, D, and E in Tab. 5 to reveal something interesting. First of all, an extra auxiliary classification task in configuration C does not offer an improvement on a BEVFusion-like predicting pipeline. We conjecture that fusing the feature before the dense BEV encoder is too early to maintain a relatively independent judgment from image cues. Second, by using a large range of resizing augmentation, configuration D with a

Table 6: Ablation study on the pivotal construction details.

Factor	Value	FPS	mAP	NDS
Image Resolution	160×448	15.86	67.08	71.28
	192×544	15.40	67.59	71.51
	256×704	12.88	68.23	71.94
	384×1056	10.23	68.69	72.13
	512×1408	7.80	68.77	71.98
Image Backbone	R18	15.87	67.49	71.47
	R50	12.88	68.23	71.94
	R101	10.99	67.86	71.62
VoxelNet Resolution (Meter)	0.100	12.88	68.23	71.94
	0.075	11.69	69.33	72.64
	0.050	9.70	69.23	72.18
VoxelNet Sparse Encoder (Base Channels)	16	12.88	68.23	71.94
	24	11.93	68.24	72.01
	32	11.12	69.36	72.66
	48	8.67	67.82	72.02
VoxelNet Dense Encoder (Blocks)	3	13.29	67.54	71.27
	5	12.88	68.23	71.94
	8	12.73	67.84	71.57
(0.075) VoxelNet Dense Encoder (Blocks)	5	11.69	69.33	72.64
	8	11.37	69.84	73.06
	12	10.94	69.52	72.78

BEVFusion-like predicting pipeline performs close to the DAL-like one in configuration G. A large range of resizing augmentation destroys the connection between the image cues and regression task prediction (e.g. mATE on train set is degenerated to 19.39 from 16.58 in configuration C) and forces the model to concentrate on the point cloud cues instead. So explicitly removing image cues from regression task prediction like the DAL-like predicting pipeline is not the only way to achieve the goal of 'Detecting As Labeling'. Instead of a novel predicting pipeline, what is more valuable to this problem is the cognitive change by updating the fundamental theory. With additional velocity augmentation in Configuration E, a BEVFusion-like predicting pipeline scores 67.87 mAP and 71.67 NDS, slightly lagging behind the DAL-like one in Configuration H. We conjecture that the Velocity augmentation challenges the regression task prediction with point cloud cues, which forces the model to utilize the image cues. So we prefer to use a DAL-like predicting pipeline to prevent the model from overfitting the image cues in regression task prediction under some extreme situations (`e.g.` strong augmentation on the point cloud modality, image branch capacity enlargement).

BEVPoolV2 As illustrated in the left of Fig. 6, the inference speed of the BEVPoolV2 is up to 3,958 PFS in a low input resolution of 256×704 and still maintains 1,223 FPS in a high resolution of 640×1760. It is 4.6 times the previous fastest implementation [16] in a low input resolution of 256×704 and

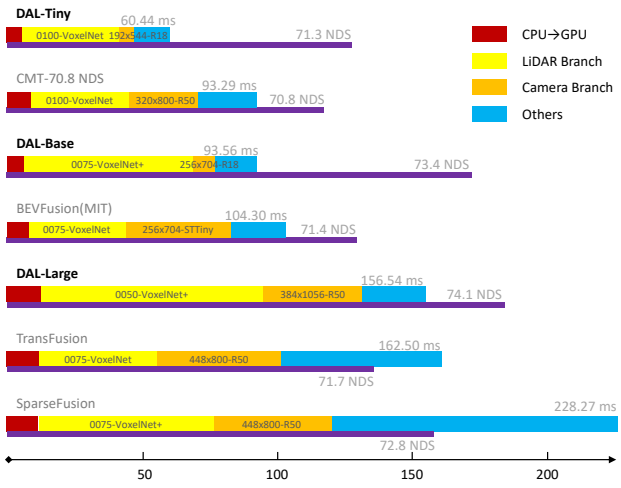

Fig. 7: Inference latency analysis and comparison.

15.1 times in a high resolution of 640×1760. BEVPoolV2 is so fast that the view transformation will no longer be a bottleneck in accelerating the predicting pipeline. Besides, the memory saved by avoiding storing the frustum feature is also appealing. As illustrated in the right of Fig. 6, the memory occupancy of BEVPoolV2 is far less than that of the previous implementations [16, 17, 22, 26] and can be maintained at a low value (`i.e.` $\leq$ 30 MB) when the input solution is increasing. It is just 1.5% of the best existing implementation in a high resolution of 640×1760. This makes the Lift-Splat-Shoot view transformer feasible to be deployed on edge devices.

Predicting Pipeline In Tab. 6, we conduct ablation on some key factors of the DAL predicting pipeline from the perspective of accuracy and inference latency. Compared with the default setting R50-256×704, further improving the resolution of the input image or the capacity of the backbone offers finite improvement in accuracy. However, the inference latency increases remarkably. As the camera branch mainly focuses on the classification task, a small backbone with low input resolution is enough for the nuScenes [2] dataset with only 10 classes and 750 scenes. By contrast, further improving the resolution of the LiDAR branch alone with a larger capacity of the backbones offers remarkable improvement in accuracy. So, it is recommended to incorporate a small image branch with a large LiDAR branch in DAL for a better trade-off between inference latency and accuracy. We follow this to construct some recommended configurations in Tab. 2.

Inference Latency Analysis In Fig. 7, we illustrate the inference latency of each component in the DAL predicting pipeline and compare it with some leading

methods. The inference latency includes the time for transferring data from CPU to GPU, extracting the point cloud feature with the LiDAR branch, extracting the image feature with the camera branch, and executing the remainder processes like multi-modality feature fusion and target prediction. Thanks to the elegant predicting pipeline design in DAL, it spends less time in the 'Other' aspect. Besides, the LiDAR branch in DAL always occupies a high proportion of the inference time. As the LiDAR branch undertakes more missions than the camera branch, the recommended configurations in DAL always use a larger LiDAR branch incorporated with a relatively small camera branch. In contrast, involving the image feature in the regression task prediction, existing methods are always equipped with a large image branch. This degenerates their time efficiency to a certain extent.

5 Conclusion, Limitation, and Future Work

Conclusion In this paper, we proposed a cutting-edge perspective 'Detecting As Labeling' for 3D object detection with LiDAR-camera fusion. DAL is developed as a template by following this perspective. DAL is an extremely elegant paradigm with a concise predicting pipeline and an easy training process. Though simple in these aspects, it substantially pushes the performance boundary in 3D object detection with LiDAR-camera fusion alone with the best trade-off between speed and accuracy. So it is an excellent milestone for both future work and practical usage.

Limitation and Future Work Objects beyond the scope of LiDAR have not been considered in DAL. We have tried discriminating this situation by predicting dense heatmaps with the point cloud feature only and comparing them with those predicted with the fuse feature. Then, the regression targets of these instances are predicted with another FFN on the fuse feature instead. However, this modification contributes less to the final accuracy. This is because only the targets with more than 1 LiDAR point will be annotated in nuScenes [2]. In addition, the scope range is small enough in nuScenes evaluation which ensures sufficient LiDAR points for predicting the regression aspects.

Besides, the simple classification task in the nuScenes dataset limits DAL to apply the advanced image backbone like SwinTransformer [21], DCN [14], and EfficientNet [31]. The open-world classification task is far more complicated and thus difficult. Thus, the image branch may take advantage of the advanced image backbones in practice.

Though DAL has an attention-free predicting pipeline, it is just a template to reveal the value of 'Detecting As Labeling'. So we use the most classical algorithms [10, 50] without applying attention. However, we do not intentionally exclude it from DAL. On the contrary, we regard attention as an appealing mechanism for further developing DAL in many aspects. For example, we can apply advanced DSVT [9] backbone like UniTR [8], apply attention-based LiDAR-camera fusion like CMT [44], and apply an attention-based sparse detection paradigm like DETR [4].

References

1. Bai, X., Hu, Z., Zhu, X., Huang, Q., Chen, Y., Fu, H., Tai, C.L.: TransFusion: Robust LiDAR-Camera Fusion for 3D Object Detection with Transformers. In: CVPR (2022)
2. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: CVPR (2020)
3. Cai, Q., Pan, Y., Yao, T., Ngo, C.W., Mei, T.: ObjectFusion: Multi-modal 3D Object Detection with Object-Centric Fusion. In: ICCV (2023)
4. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-End Object Detection with Transformers. In: ECCV (2020)
5. Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-View 3D Object Detection Network for Autonomous Driving. In: CVPR (2017)
6. Chen, Y., Li, Y., Zhang, X., Sun, J., Jia, J.: Focal Sparse Convolutional Networks for 3D Object Detection. In: CVPR (2022)
7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A Large-Scale Hierarchical Image Database. In: CVPR (2009)
8. Haiyang Wang, Hao Tang, S.S.A.L.Z.L.B.S.L.W.: UniTR: A Unified and Efficient Multi-Modal Transformer for Bird's-Eye-View Representation. In: ICCV (2023)
9. Haiyang Wang, Chen Shi, S.S.M.L.S.W.D.H.B.S., Wang, L.: DSVT: Dynamic Sparse Voxel Transformer with Rotated Sets. In: CVPR (2023)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: CVPR (2016)
11. Hu, Y., Yang, J., Chen, L., Li, K., Sima, C., Zhu, X., Chai, S., Du, S., Lin, T., Wang, W., Lu, L., Jia, X., Liu, Q., Dai, J., Qiao, Y., Li, H.: Planning-oriented Autonomous Driving. In: CVPR (2023)
12. Huang, J., Huang, G.: BEVDet4D: Exploit Temporal Cues in Multi-camera 3D Object Detection. arXiv preprint arXiv:2203.17054 (2022)
13. Huang, J., Huang, G., Zhu, Z., Yun, Y., Du, D.: BEVDet: High-performance Multi-camera 3D Object Detection in Bird-Eye-View. arXiv preprint arXiv:2112.11790 (2021)
14. Jifeng Dai, Haozhi Qi, Y.X.Y.L.G.Z.H.H.Y.W.: Deformable Convolutional Networks. arXiv preprint arXiv:1703.06211 (2017)
15. Li, Y., Chen, Y., Qi, X., Li, Z., Sun, J., Jia, J.: Unifying Voxel-based Representation with Transformer for 3D Object Detection. In: NeurIPS (2022)
16. Li, Y., Bao, H., Ge, Z., Yang, J., Sun, J., Li, Z.: BEVStereo: Enhancing Depth Estimation in Multi-view 3D Object Detection with Dynamic Temporal Stereo. In: AAAI (2022)
17. Li, Y., Ge, Z., Yu, G., Yang, J., Wang, Z., Shi, Y., Sun, J., Li, Z.: BEVDepth: Acquisition of Reliable Depth for Multi-view 3D Object Detection. In: AAAI (2022)
18. Li, Z., Yu, Z., Wang, W., Anandkumar, A., Lu, T., Álvarez, J.M.: FB-BEV: BEV Representation from Forward-Backward View Transformations. ICCV (2023)
19. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature Pyramid Networks for Object Detection. In: CVPR (2017)
20. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common Objects in Context. In: ECCV (2014)
21. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In: ICCV (2021)

22. Liu, Z., Tang, H., Amini, A., Yang, X., Mao, H., Rus, D.L., Han, S.: BEVFusion: Multi-Task Multi-Sensor Fusion with Unified Bird's-Eye View Representation. In: ICRA (2023)
23. Pan, B., Sun, J., Leung, H.Y.T., Andonian, A., Zhou, B.: Cross-View Semantic Segmentation for Sensing Surroundings. IEEE Robotics and Automation Letters (2020)
24. Park, D., Ambrus, R., Guizilini, V., Li, J., Gaidon, A.: Is Pseudo-Lidar Needed for Monocular 3D Object Detection? In: ICCV (2021)
25. Park, J., Xu, C., Yang, S., Keutzer, K., Kitani, K., Tomizuka, M., Zhan, W.: Time Will Tell: New Outlooks and A Baseline for Temporal Multi-View 3D Object Detection (2023)
26. Philion, J., Fidler, S.: Lift, Splat, Shoot: Encoding Images from Arbitrary Camera Rigs by Implicitly Unprojecting to 3D. In: ECCV (2020)
27. Reading, C., Harakeh, A., Chae, J., Waslander, S.L.: Categorical Depth Distribution Network for Monocular 3D Object Detection. In: CVPR (2021)
28. Roddick, T., Cipolla, R.: Predicting Semantic Map Representations from Images using Pyramid Occupancy Networks. In: CVPR (2020)
29. Sindagi, V.A., Zhou, Y., Tuzel, O.: MVX-Net: Multimodal Voxelnet for 3D Object Detection. In: ICRA (2019)
30. Smith, L.N.: Cyclical Learning Rates for Training Neural Networks. In: WACV (2017)
31. Tan, M., Le, Q.: EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In: ICLR (2019)
32. Tian, X., Jiang, T., Yun, L., Wang, Y., Wang, Y., Zhao, H.: Occ3D: A Large-Scale 3D Occupancy Prediction Benchmark for Autonomous Driving. arXiv preprint arXiv:2304.14365 (2023)
33. Tingting Liang, Hongwei Xie, K.Y.Z.X.Z.L.Y.W.T.T.B.W., Tang, Z.: BEVFusion: A Simple and Robust LiDAR-Camera Fusion Framework. In: NeurIPS (2022)
34. Vora, S., Lang, A.H., Helou, B., Beijbom, O.: PointPainting: Sequential Fusion for 3D Object Detection. In: CVPR (2020)
35. Vora, S., Lang, A.H., Helou, B., Beijbom, O.: PointPainting: Sequential Fusion for 3D Object Detection. In: CVPR (2020)
36. Wang, C., Ma, C., Zhu, M., Yang, X.: PointAugmenting: Cross-Modal Augmentation for 3D Object Detection. In: CVPR (2021)
37. Wang, C., Ma, C., Zhu, M., Yang, X.: PointAugmenting: Cross-Modal Augmentation for 3D Object Detection. In: CVPR (2021)
38. Wang, X., Zhu, Z., Xu, W., Zhang, Y., Wei, Y., Chi, X., Ye, Y., Du, D., Lu, J., Wang, X.: OpenOccupancy: A Large Scale Benchmark for Surrounding Semantic Occupancy Perception. In: ICCV (2023)
39. Wang, Y., Guizilini, V., Zhang, T., Wang, Y., Zhao, H., , Solomon, J.M.: DETR3D: 3D Object Detection from Multi-view Images via 3D-to-2D Queries. In: CoRL (2021)
40. Wu, H., Wen, C., Shi, S., Li, X., Wang, C.: Virtual Sparse Convolution for Multimodal 3D Object Detection. In: CVPR (2023)
41. Wu, X., Peng, L., Yang, H., Xie, L., Huang, C., Deng, C., Liu, H., Cai, D.: Sparse fuse dense: Towards high quality 3d detection with depth completion. In: CVPR (2022)
42. Xie, Y., Xu, C., Rakotosaona, M.J., Rim, P., Tombari, F., Keutzer, K., Tomizuka, M., Zhan, W.: SparseFusion: Fusing Multi-Modal Sparse Representations for Multi-Sensor 3D Object Detection. In: ICCV (2023)

43. Xu, S., Zhou, D., Fang, J., Yin, J., Bin, Z., Zhang, L.: FusionPainting: Multimodal Fusion with Adaptive Attention for 3D Object Detection. In: ITSC (2021)
44. Yan, J., Liu, Y., Sun, J., Jia, F., Li, S., Wang, T., Zhang, X.: Cross Modal Transformer via Coordinates Encoding for 3D Object Dectection. In: ICCV (2023)
45. Yan, Y., Mao, Y., Li, B.: SECOND: Sparsely Embedded Convolutional Detection. Sensors (2018)
46. Yang, W., Li, Q., Liu, W., Yu, Y., Ma, Y., He, S., Pan, J.: Projecting Your View Attentively: Monocular Road Scene Layout Estimation via Cross-View Transformation. In: CVPR (2021)
47. Yang, Z., Chen, J., Miao, Z., Li, W., Zhu, X., Zhang, L.: DeepInteraction: 3D Object Detection via Modality Interaction. In: NeurIPS (2022)
48. Yin, T., Zhou, X., Krahenbuhl, P.: Center-based 3D Object Detection and Tracking. In: CVPR (2021)
49. Zhou, B., Zhao, H., Puig, X., Xiao, T., Fidler, S., Barriuso, A., Torralba, A.: Semantic Understanding of Scenes through the ADE20K Dataset. IJCV (2019)
50. Zhou, Y., Tuzel, O.: VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. In: CVPR (2018)
51. Zhu, B., Jiang, Z., Zhou, X., Li, Z., Yu, G.: Class-balanced Grouping and Sampling for Point Cloud 3D Object Detection. arXiv preprint arXiv:1908.09492 (2019)