Multi-Modality Cut and Paste for 3D Object Detection

Wenwei Zhang[1] Zhe Wang[2] Chen Change Loy[1]✉
[1]S-Lab, Nanyang Technological University [2]SenseTime Research
{wenwei001, ccloy}@ntu.edu.sg wangzhe@sensetime.com

Abstract

Three-dimensional (3D) object detection is essential in autonomous driving. There are observations that multi-modality methods based on both point cloud and imagery features perform only marginally better or sometimes worse than approaches that solely use single-modality point cloud. This paper investigates the reason behind this counter-intuitive phenomenon through a careful comparison between augmentation techniques used by single-modality and multi-modality methods. We found that existing augmentations practiced in single-modality detection are equally useful for multi-modality detection. Then we further present a new multi-modality augmentation approach, ***Multi-mOdality Cut and pAste (MoCa)****. MoCa boosts detection performance by cutting point cloud and imagery patches of ground-truth objects and pasting them into different scenes in a consistent manner while avoiding collision between objects. We also explore beneficial architecture design and optimization practices in implementing a good multi-modality detector. Without using ensemble of detectors, our multi-modality detector achieves new state-of-the-art performance on nuScenes dataset and competitive performance on KITTI 3D benchmark. Our method also wins the best PKL award in the 3rd nuScenes detection challenge. Code and models will be released at https://github.com/open-mmlab/mmdetection3d.*

1. Introduction

Three-dimensional (3D) object detection is an essential vision task with wide applications. In the context of autonomous driving, encouraging results [40, 48] have been obtained by using point cloud data collected via Light Detection and Ranging (LiDAR) devices. Nonetheless, single-modality LiDAR-based 3D object detection is not without limitations. In particular, while LiDAR point cloud offers valuable cues for accurate 3D localization, a typical LiDAR system can only perceive objects within 70 meters [1] and cannot distinguish semantic categories of similar structures, *e.g.*, pedestrians and trees.

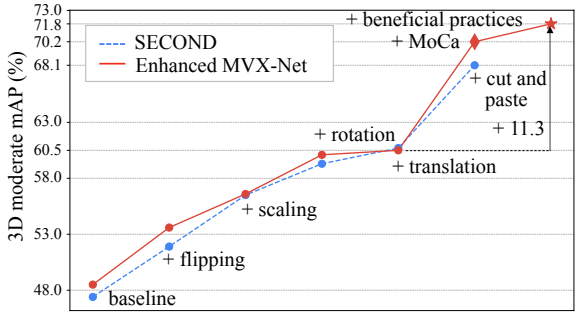

Figure 1: **Step-by-step performance improvements brought by multi-modality augmentations, MoCa, and beneficial practices.** The effectiveness of random flipping, scaling, rotation, and translation are validated on multi-modality detection. Built upon these augmentations, the proposed multi-modality cut and paste (MoCa) with beneficial practices further improves the performance of MVX-Net, surpassing its single-modality counterpart (SECOND with cut and paste) by a large margin on KITTI dataset.

Meanwhile, imagery features, by nature, play a complementary role. It is believed that imagery features facilitate more accurate detection on distant objects and provide richer semantic information. Much effort has geared towards adding RGB camera images as a second modality to complement LiDAR point cloud data. For instance, Kalman filter has been used to perform a late fusion of point cloud and imagery measurements to enhance the robustness of detection [8]. Early fusion methods have also been attempted [22, 23, 41, 42]. For example, Multimodal VoxelNet (MVX-Net) [41] performs point-wise fusion between features extracted from a 2D detector and LiDAR point cloud. The fused features are then voxelized into regular grid feature maps for 3D detection. However, top methods on the benchmarks [1, 13] are mainly using single-modality, while multi-modality approaches seem to attain only marginal improvement.

It remains obscure why multi-modality, despite having more information, only supports a marginal improvement in 3D object detection. This paper investigates this phenomenon and shows that multi-modality augmentation plays pivotal roles. We first systematically validate the effectiveness of existing augmentation strategies (Fig. 1), during which we discuss a pipeline, named *multi-modality*

transformation flow, to ensure multi-modality consistency in various augmentations. While previous methods [23, 41] already used multi-modality augmentations, how to ensure consistency during augmentations has not been systematically discussed yet, limiting the usage of augmentations.

Under the summarized *multi-modality transformation flow*, we further propose a new augmentation approach, *multi-modality cut and paste (MoCa)*. State-of-the-art LiDAR-based detectors [39, 40, 48] benefit significantly from cut and paste augmentation [46]. However, such an effective augmentation scheme is absent from multi-modality methods [41, 42], due to difficulties in ensuring consistency and plausibility across modalities; such omission severely limits their performance. We constrain cut and paste operations to avoid collisions between objects in both bird's eye view (BEV) of point cloud and 2D imagery domain. To further improve the detector's generalizability, we use mixed intersection over foreground (IoF) in imagery domain to guide the pasting process. MoCa improves object detection performance by a large margin and it is readily applicable to many existing multi-modality detectors [23, 32, 41, 42].

Along with MoCa, we also find some interesting phenomena and further explore good practices in architecture design and optimization. These practices, together with MoCa, improve the mAP of MVX-Net [41], a strong and generic multi-modality detector, by **11.3%** moderate mAP on KITTI dataset (Fig.1) and **5.8%** mAP on nuScenes dataset. Without using an ensemble of class-specialized detectors, the enhanced MVX-Net achieves new state-of-the-art results on nuScenes dataset [1] and obtains competitive results on KITTI 3D benchmark [14].

2. Related Work

3D object detection. In recent years, many approaches [27, 40, 46, 48] have been focusing on processing LiDAR point cloud to improve the performance of 3D object detection. To deal with the irregular and unstructured nature of point cloud, common approaches either apply CNN to the voxelized representation [21, 27, 46, 53] or process the raw points [31, 40, 47] by PointNets [33, 34]. Recent methods [7, 39, 48] exploit both voxel representation and raw points. There are also attempts [5, 43] that purely rely on cameras for 3D detection.

Previous works aggregate image and point cloud features from different views [6, 20]; the efficiency is limited by the view aggregation for a large quantity of anchors [20] or the proposals [6]. More recent works [22, 23, 41] fuse image features into each point, but they exhibit various feature misalignment issues. For example, MVX-Net [41] quantizes image coordinates, while methods [22, 23] based on ContFuse [23] use the nearest points for each BEV feature grid. Frustum-based methods [32, 44] obtain frustum proposals from an image and then apply PointNet [33] to point cloud for 3D object localization. Their performance is limited by the proposal qualities and they may not fully exploit the complementary information of multi-modalities. ImVoteNet [30] skips the above-mentioned issue by fusing 2D votes in images and 3D votes in point clouds. We solve the misalignment issue in MVX-Net [41] by *aligned feature fusion*, and enhance it by pyramid fusion with FPN [24] to study the training strategies and data augmentation.

Augmentations for detection. Data augmentation is crucial to reduce overfitting and to improve the models' generalization abilities. Single-modality 3D object detection methods [40, 48] use more aggressive data augmentations than existing multi-modality methods [22, 32, 41] do.

Conventional image augmentations include but are not limited to random cropping, random flipping, and multi-scale training [4, 16]. For point cloud data, common augmentation techniques are random flipping, rotation, translation, and scaling [40, 46, 53]. However, in multi-modality 3D detection, the practices of augmentations vary across different methods [22, 23, 41]. For example, ContFuse [23] skips random flip, whereas MVX-Net [41] does not apply random rotation and translation to maintain consistency between image and point cloud. In this study, we carefully validate the effectiveness of these augmentations for multi-modality 3D object detection.

There are also augmentations applied to regions that contain objects of an image [9, 12, 51]. A representative method is to cut and paste objects [10–12]. While Dwibedi *et al*. [11] paste objects randomly, recent works use a location probability map [12] or a visual context model [10] to guide the pasting process. Cutting and pasting the points of objects is also common for LiDAR-based 3D detection methods [27, 46, 48] but is absent from multi-modality methods [22, 32, 41]. Therefore, we make the first attempt to enable *multi-modality cut and paste (MoCa)* to bridge the gap in the augmentation techniques.

3. Methodology

We are curious with the limited performance gain after extending single-modality to multi-modality 3D object detectors. We believe that a more careful investigation in the implementation of existing pipelines could shed lights for improving the performance of multi-modality detectors. First, we revisit common data augmentation techniques for single-modality 3D object detection and validate them for multi-modality 3D object detection (Sec. 3.1). Then we propose *multi-modality cut and paste (MoCa)* to further improve the performance (Sec. 3.2). We choose MVX-Net [41] as a strong and generic baseline for our study. Finally, we explore different architecture design and training strategies for better performance (Sec. 3.3). The findings can be easily extended to other multi-modality 3D object detectors that share a similar pipeline.

3.1. Multi-Modality Augmentation

Data augmentation plays a pivotal role in 3D object detection to reduce model overfitting. Overfitting is usually caused by insufficient training data due to difficulty in data collection and annotation. For instance, there are fewer than 15K frames in KITTI [14] 3D object detection dataset. This is far fewer than the conventional image datasets like COCO [25] and ImageNet [38], which contain about 328K and 1.46 billion annotated images, respectively.

Existing augmentation techniques. We first carefully validate the gain of existing augmentation techniques on multi-modality data. Such experiments are new in the literature. In 3D object detection, the augmentation strategies adopted by single-modality methods are more aggressive than those used by multi-modality methods, *e.g.*, global rotation and random flip are widely applied by single-modality methods [27, 40, 48] but are absent from some multi-modality methods [22, 23, 41] due to concerns of multi-modality consistency. We revisit existing augmentation techniques that can be extended from single-modality to multi-modality augmentations and compare these augmentation techniques step by step (Fig. 1). The results show that global flipping, scaling, rotation, and translation are all essential augmentations to improve a multi-modality detector, as long as the multi-modality consistency can be maintained.

Multi-modality transformation flow. The essence of maintaining multi-modality consistency during augmentation is to maintain the correct correspondence between points and image pixels; thus, the points can still be correctly fused with their corresponding image features after a series of augmentations. While previous studies [23, 41] have applied augmentation in multi-modality detection, they use much fewer augmentations than those single-modality methods and the issue of maintaining consistency across different modalities has not been systematically studied. Here, we discuss *multi-modality transformation flow*, which is useful for maintaining the multi-modality consistency during augmentation, enabling more aggressive augmentation strategies for multi-modality detection.

As shown in Fig. 2, the multi-modality transformation flow records all the transformations of point cloud and image data during data augmentations. Such transformation flow is required to transform the augmented data back for finding the correct correspondence between the point cloud and image pixels during fusion. Most augmentations are reversible, *i.e.*, they contain a forward transformation to augment the data, with a reverse transformation to transform the data back into its original state. Before training, the image and point cloud data are augmented by different augmentations independently. Note that the transformations of points are equivalent to transforming the LiDAR sensor to a new position, resulting in new point coordinates but not affecting the captured image as the camera is not transformed. Consequently, rotation and translation can be applied to point cloud data, but they do not need to be applied to the image simultaneously.

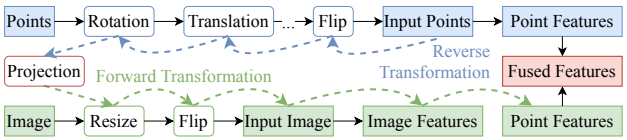

Figure 2: Multi-modality transformation flow.

During fusion, the reverse transformation of each augmentation is applied to the augmented points following the inverse order of point cloud augmentations. Next, we can safely project the points onto the imagery pixel coordinates using the calibration information of data [1, 13]. The projected points then obtain their corresponding image features after going through the forward transformations following the same order of the corresponding image augmentations.

Within such multi-modality transformation flow, any augmentation, as long as it is reversible, can be used to augment multi-modality data without sacrificing the consistency. Some methods [46, 53] apply a small amount of noise (*e.g.*, random translation and rotation) to each ground truth object separately. However, recent works suggest that such strategy either does not add values [39] or it hurts the performance in some scenarios [21]. Therefore, we leave it to future research.

3.2. Multi-modality Cut and Paste

Cut and paste is an effective strategy to create a diverse combination of scenes and objects when data is limited [10–12]. It is a common augmentation technique in single-modality detectors but not in multi-modality detectors. Recently, SECOND [46] introduces cut and paste into the point cloud domain, named as ground truth sampling (GT-sampling). It is shown that GT-sampling not only accelerates model convergence but also reduces class imbalance issues. Therefore, almost all state-of-the-art single-modality 3D object detectors [27, 39, 40, 47, 48] adopt GT-sampling to boost their performance.

It is non-trivial to consistently extend GT-sampling to the imagery domain in multi-modality methods. A point cloud patch that is visible from the bird's eye view (BEV) does not guarantee its corresponding image patch is also perceivable in the imagery domain. Often, an object may be occluded in the image plane and thus its imagery content only captures the features of the occluding object. A blind cut and paste would risk having inconsistent point cloud and imagery patches. To circumvent this intricate problem, multi-modality methods [22, 41] resort to a lower starting baseline without using GT-sampling, but they still need to compare performance with single-modality methods [40, 48] that use GT-sampling.

To our knowledge, this is the first work that investigates underlying challenges in multi-modality cut and paste for

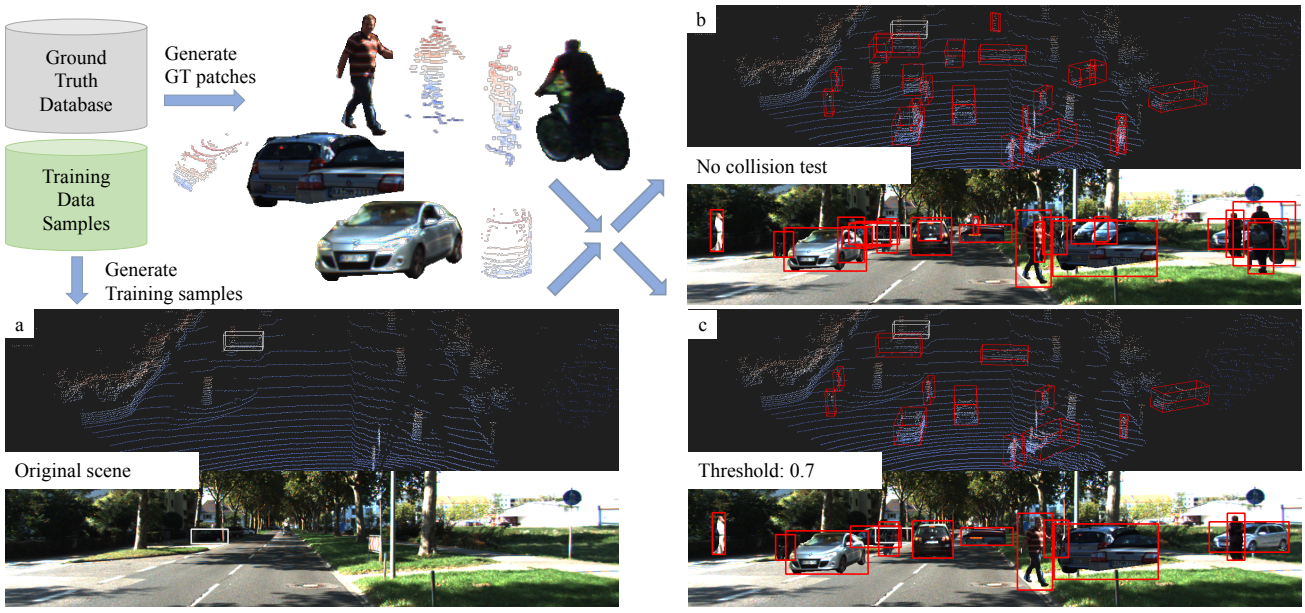

Figure 3: **Multi-modality cut and paste.** Given one training frame and some GT objects generated by the database, MoCa checks collisions in both BEV and 2D image based on Intersection over Foreground (IoF). It then pastes the valid patches of objects to all the modalities. A blind pasting operation will cause many heavily overlapped objects (b), which is far from the natural distribution of data. The gray and red bounding boxes indicate the original object in the current frame and the pasted objects, respectively. The original frame (a) contains very few objects and is enriched with more objects after multi-modality cut and paste (c). The figure is best seen in color.

3D object detection. We further propose a more general form of GT-sampling named *multi-modality cut and paste (MoCa)*. It is readily applicable to existing multi-modality methods [23,32,41,42] and brings substantial improvement.

MoCa first builds a ground truth database for each annotated object offline. Specifically, point cloud for each object and its corresponding image patch is cropped before training, using ground truth 3D bounding boxes and 2D masks, respectively. During training, MoCa randomly samples point cloud-image patch pairs and paste them to the original scene according to their 3D bounding boxes and 2D masks (Fig. 3). To avoid boundary artifacts caused by image patches, we follow [11] to apply random blending to smoothen the boundaries of image patches. Such an operation is not needed for point cloud since the data is sparse.

Occlusion handling. The non-trivial part of multi-modality cut and paste lies in occlusion handling. Image-based cut and paste [10, 11] usually pastes objects at different locations in the image and ignores the physical plausibility. On the other hand, point cloud cut and paste [46] only avoids occlusion in BEV because objects are generally assumed to be on the same ground plane and well separated in BEV. Potential occlusions in 2D image are neglected because current 3D object detectors [21,46,48] usually predict bounding boxes only from BEV. However, during the multi-modality fusion process, due to the occlusion, the projected points of an occluded object might obtain the image features of occluding objects (Fig. 3 (b)). This makes the image features ambiguous and increases the difficulty in training feature extractors. Therefore, not handling the occlusion in 2D image will affect the overall performance as validated by our experiments (Table 2).

MoCa considers the consistency in both point cloud and image modalities. Specifically, given a batch of objects with their point cloud and their corresponding image patches, the multi-modality cut and paste first discards overlapped objects in BEV and then carefully handles the occlusion in 2D images. Given a set $P = \{p_i | i = 1, 2, ..., N\}$ containing the original objects and the objects to be pasted, we use Intersection-over-Foreground (IoF) to represent the occlusion degree of an object p_i in the 2D image as

$$IoF(p_i, P) = \max \left\{ \frac{p_i \bigcap p_j}{p_i} | j \neq i \right\}. \quad (1)$$

Once an sampled object's IoF is greater than a given threshold or that object makes any one of the original boxes' IoF greater than the given threshold, the sampled object will not be pasted in the current training iteration. The original objects will not be discarded.

Mixed IoF thresholds. Different IoF thresholds lead to a different number of objects being pasted. Therefore, we propose to use a mix of occlusion thresholds to provide more diverse occlusion cases and scenes during training to improve the detector's robustness and generalization ability. Specifically, given a threshold set (we use $\{0, 0.3, 0.5, 0.7\}$ in this work), a threshold will be randomly chosen from the set. Then objects whose occlusion degrees (in the batch) are greater than the threshold will be discarded during each

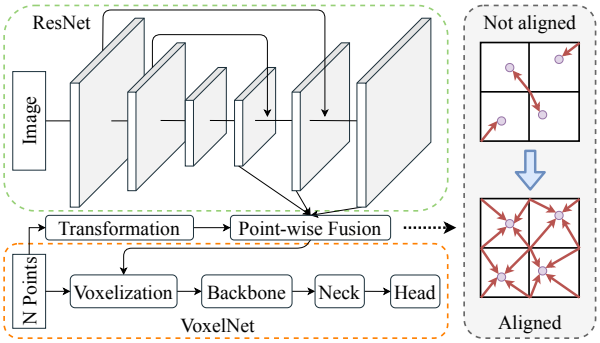

Figure 4: Enhanced MVX-Net. We fuse FPN features with point cloud features using not-quantized coordinate. The fused features are voxelized into regular grid feature maps, which are then fed to VoxelNets (*e.g.*, PointPillars [21]) for 3D detection.

iteration. The remaining objects' point cloud and image patches will then be pasted to the positions specified by their 3D and 2D bounding boxes, respectively. Image patches are pasted in the order of their depth, *i.e.*, the farther the object, the earlier it is pasted.

3.3. Exploring Beneficial Practices

In this paper, we use MVX-Net [41] to study multi-modality augmentation as it is simple and generic. MVX-Net uses a pre-trained Faster R-CNN as an image feature extractor and adopts VoxelNets (*e.g.*, PointPillars [21] or SECOND [46]) for 3D detection. During implementation, we explore some beneficial practices in architecture design and optimization.

Aligned pyramid feature fusion. For feature fusion, the point cloud is first transformed from LiDAR coordinates to camera coordinates and then projected to the image pixel coordinates[1]. Once the pixel coordinates P_{img} are obtained, MVX-Net selects image features using the quantized coordinates P'_{img} and concatenate them with the point cloud feature. However, as shown in Fig. 4, the quantization introduces feature misalignment in the fusion process since the quantized coordinates are not an accurate projection from the given points. Such misalignment brings adverse effects in multi-modality detection because the quantization is applied to at least 10K projected points with their corresponding features. Inspired by RoIAlign [15], we introduce *aligned feature fusion*, which uses differentiable bilinear sampling kernel [18] to overcome the misalignment issue and obtain the feature of a given point as follows:

$$\sum_{H}^{n}\sum_{W}^{m} U_{nm}\max(0,1-|P_x-m|)\max(0,1-|P_y-n|), \tag{2}$$

where H and W are the height and width, respectively, of the feature map U, and $P_{img}=(P_x,P_y)$ is the pixel coordinates of a projected point.

[1]Details are provided in the appendix.

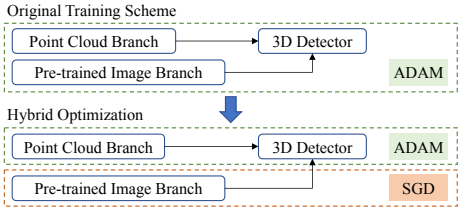

Figure 5: Hybrid Optimization

We further enhance the MVX-Net [41] by adding a FPN [24] in image branch (Fig. 4). For the feature map in each scale, the pixel coordinate P_{img} is divided by the strides of the current feature map and then used to select the corresponding image features. Then we obtain the multi-scale imagery features of each point from different scales of feature maps, concatenate them together, and fuse them with the points' point cloud features by a linear layer.

Hybrid optimization. Multi-modality 3D object detectors [22, 23, 32, 41] usually use a pre-trained model for image feature extraction because there are limited image data in the problem domain. The image feature extractor is typically pre-trained on 2D recognition tasks [25, 38] using an SGD optimizer to ensure good performance. For 3D object detection, a common practice is to train the point cloud feature extractor from scratch jointly with the pre-trained image feature extractor using an ADAM optimizer. Interestingly, we observe that the switch of optimizer from SGD to ADAM in the image branch causes a slight performance drop. We resolve this problem by a *hybrid optimization* strategy. Specifically, we retain the use of a SGD optimizer for the image feature extractor and an ADAM optimizer for the point cloud branch (Fig. 5) during the joint training. Such a choice of using different optimizer for each modality can be explained by common practices in each individual modality – existing methods usually use SGD for image recognition tasks [3, 16, 37] while ADAM is more commonly used for point cloud representation learning because ADAM is shown superior in handling irregular and unstructured point cloud data [33, 34].

4. Experiments

KITTI dataset. We first validate our methods on KITTI dataset [14]. The KITTI dataset contains 7481 training images and 7518 test images, both with their corresponding point cloud. We train all the models using the train split containing 3712 samples and evaluate them on the validation split consisting of 3769 samples, following previous works [6, 40, 46]. The KITTI benchmark evaluates the models by Average Precision (AP) of each class (car, pedestrain, and cyclist) under easy, moderate, and hard conditions. For simplicity, we use mean AP over three classes to measure overall performance of the models in the ablation study.

nuScenes dataset. We further evaluate our method on the more challenging and large-scale nuScenes dataset [1]. The

Table 1: Comparison of different augmentation strategies for single-modality 3D detector SECOND [46] and multi-modality 3D detector MVX-Net [41] on KITTI. The augmentation technique is in order, whereby, each augmentation is added onto the previous ones sequentially

Method	Augmentation	mAP (%)	Pedestrian			Cyclist			Car		
		Mod.	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
SECOND	no augmentation	47.4	46.5	40.0	34.7	55.3	38.7	32.8	71.5	63.4	63.4
	+ flipping	51.9	57.6	50.9	45.1	53.7	37.6	37.2	77.3	67.3	65.8
	+ scaling	56.5	54.8	47.0	44.5	67.2	48.4	42.3	84.5	74.1	67.6
	+ rotation	59.3	59.8	53.0	46.6	66.5	48.8	48.0	87.1	76.0	68.9
	+ translation	60.7	61.0	54.6	51.9	69.0	50.7	49.2	87.2	76.8	75.6
	+ cut and paste	68.1	68.1	61.1	54.0	79.4	65.8	60.8	87.5	77.4	75.5
MVXNet	no augmentation	48.5	49.1	45.5	40.5	53.2	35.4	31.2	75.3	64.5	57.8
	+ flipping	53.6	49.7	43.3	41.7	57.2	45.6	40.4	81.5	71.9	65.9
	+ scaling	56.6	54.4	47.0	44.6	66.2	49.5	42.9	82.7	73.4	67.1
	+ rotation	60.1	62.7	54.7	52.8	67.2	49.5	42.6	87.7	76.0	68.8
	+ translation	60.5	62.7	55.6	53.2	65.7	49.1	48.4	87.2	76.9	75.5
	+ MoCa (ours)	**70.2**	**68.6**	**61.9**	**54.7**	**86.0**	**71.2**	**65.0**	**87.9**	**77.6**	**76.0**

Table 2: Comparison of different IoF thresholds. 'No test' means not applying collision test, and 'mixed' means using different thresholds during training

Threshold	AP_{easy}	$AP_{mod.}$	AP_{hard}
No test	84.93	78.20	72.13
0	**86.99**	78.45	73.33
0.3	86.09	79.03	73.92
0.5	85.76	79.05	73.56
0.7	86.78	78.65	73.17
Mixed	86.30	**79.30**	**74.42**

nuScenes dataset contains 28130 syncronized multi-view images and point cloud samples for training, 6019 samples for validation, and 6008 samples for test benchmark. The dataset annotates 10 categories and we evaluate the models on these 10 classes by NDS metric [1]. The NDS metric not only measures the mean AP but also takes translation, scale, orientation, velocity, and attribute errors of the true positives into considerations.

4.1. Ablation Study on KITTI Dataset

Multi-modality augmentation. We first validate the effectiveness of different multi-modality augmentation techniques. To obtain a clear comparison with SECOND [46], we use ResNet-50 [16] with FPN [24] in image branch and use SECOND as the point cloud branch in MVX-Net [41], without hybrid optimization. Some points may not be visible in the pasted image after multi-modality cut and paste; for those points, we set their corresponding image features as zeros during multi-modality fusion.

Table 1 verifies that random flipping, scaling, rotating, and translating point cloud are essential for both single-modality and multi-modality 3D detectors. With these augmentations added sequentially, both the single-modality detector (SECOND [46]) and the multi-modality detector (MVX-Net [41]) obtain significant improvement. Both single-modality and multi-modality methods obtain large improvement with the help of cut and paste (moderate mAP from 60.7% and 60.5% to 68.1% and 70.2%, respectively).

Notably, the results also show that *multi-modality cut and paste* plays a critical role in bridging the performance gap between multi-modality detectors [41] and single-modality detectors [46]. Without cut and paste, MVX-Net cannot even surpass SECOND (60.5% vs. 60.7%). But with the help of *MoCa*, MVX-Net surpasses its counterpart by a large margin (70.2% vs. 68.1%).

Table 3: Comparison of Faster R-CNN trained with different augmentations and different epochs. 'Baseline' means not using cut and paste, and 'AutoAug' indicates using searched augmentation strategies [55]

Method	Epochs	AP_{easy}	$AP_{mod.}$	AP_{hard}
Baseline	20	86.29	75.23	71.80
	36	86.57	73.31	67.42
AutoAug [55]	20	86.40	78.53	73.66
MoCa (ours)	20	86.30	79.30	**74.42**
	36	**86.56**	**79.45**	74.11
	48	85.24	79.33	73.83

Multi-modality cut and paste (MoCa). We evaluate the effectiveness of each component in MoCa. We use the instance masks in KINS dataset [35] to build the multi-modality GT database. We empirically find that having 6 pedestrians, 6 cyclists, and 12 cars in a frame for training yields the best performance. Because the influence of collision tests are mainly related to 2D images, for simplicity and without losing generalizability, we compare different settings on Faster R-CNN [37] with FPN [24]. The Faster R-CNN is trained by 20 epochs following the standard setting in MMDetection [4].

Table 2 shows that collision test improves the detector's performance, and different thresholds lead to different APs under different conditions. The proposed mixed IoF thresholds yield the best performance and is adopted in the subsequent experiments. As shown in Table 3, cut and paste significantly improves the results, in comparison with those not using cut and paste and those using searched augmentation strategies [55]. Although training more epochs does not bring significant improvement, cut and paste maintains its high performance under all conditions, while training longer degrades baseline's performance on moderate and hard conditions. The results of sustaining a longer training suggest the effectiveness of MoCa in reducing overfitting.

Table 4: Comparison of different training strategies. 'Pre-train' indicates which pre-trained model is used. 'COCO' means the model is pre-trained on COCO dataset, and '+ KITTI' means the model is further pre-trained on KITTI 2D dataset. 'Freeze' indicates whether ResNet-50 [16] in the image branch is frozen. The average results and standard deviation over 5 runs are reported. The best results in each setting are bolded

Pre-train	Freeze	Optimizer	3D mAP (%)		
			Easy	Mod.	Hard
COCO	×	SGD	80.2 ± 1.0	69.9 ± 0.8	66.1 ± 1.1
		ADAM	80.9 ± 0.7	70.2 ± 0.7	66.3 ± 1.2
		Hybrid	$\mathbf{81.1 \pm 0.7}$	$\mathbf{71.0 \pm 0.6}$	$\mathbf{67.2 \pm 0.6}$
COCO + KITTI	×	SGD	80.5 ± 0.8	70.3 ± 0.2	67.0 ± 0.4
		ADAM	80.6 ± 0.3	69.6 ± 0.3	65.7 ± 0.6
		Hybrid	$\mathbf{81.1 \pm 0.5}$	$\mathbf{70.6 \pm 0.7}$	$\mathbf{67.0 \pm 1.0}$
	✓	SGD	80.3 ± 0.6	70.3 ± 0.4	66.5 ± 0.8
		ADAM	80.9 ± 0.6	70.7 ± 0.5	67.0 ± 0.3
		Hybrid	$\mathbf{81.8 \pm 0.4}$	$\mathbf{71.2 \pm 0.5}$	$\mathbf{67.8 \pm 0.7}$

Hybrid optimization. As discussed in Sec. 3.3, the choice of optimizer for the image feature extractor matters. Here, we compare the following variants: (1) *SGD*, the SGD optimizer is used for the image-branch pre-training as well as for both the image and point cloud branches during joint training; (2) *ADAM*, the SGD optimizer is used for the image-branch pre-training but the ADAM optimizer is used for both the image and point cloud branches during joint training; (3) *Hybrid*, we use the hybrid optimization as presented in Sec. 3.3. The optimal learning rate and other hyper-parameters for all the aforementioned variants are found using a grid search to ensure fair comparisons. Furthermore, different pre-trained image feature extractors are tested to ensure completeness. More details on pre-training can be found in the appendix. As shown in Table 4, the proposed hybrid optimization exhibits advantages in all these training settings. Our results here are purely empirical but they reveal the interesting tendency of using different optimizers in different modality branches. The phenomenon here worths further exploration especially from the perspective of optimization routes.

Aligned pyramid feature fusion. We evaluate the proposed aligned pyramid feature fusion with other beneficial components step by step in Table 5. MVX-Net enhanced by our methods significantly surpasses its previous version by **9.5%**, **11.3%**, and **10.4%** in mAP of easy, moderate, and hard conditions, respectively. The enhanced MVX-Net also surpasses its single-modality counterpart, SECOND [46], by **3.1%**, **3.7%**, and **6.1%** in mAP of easy, moderate, and hard conditions, respectively.

4.2. Ablation Study on nuScenes Dataset

We further validate MoCa and the enhanced MVX-Net on the more challenging nuScenes dataset. Since existing results in the literature are mainly based on PointPillars [21], we also adopt PointPillars as the point cloud branch in MVX-Net [41]. We reimplement PointPillars [21] but supplement it with FPN [24]. PointPillars [21] with FPN [24] achieves 53.4% in the NDS score, higher than that reported by the dataset provider (44.2% [1]).

Table 5: Ablation study of each component on KITTI validation dataset. Modifications are added sequentially

Method	mAP (%)		
	Easy	Mod.	Hard
SECOND [46]	78.3	68.1	63.4
+ image branch [41]	71.9	60.5	59.1
+ MoCa	80.9	70.2	65.2
+ hybrid optimization	81.2	70.8	65.7
+ aligned pyramid fusion	81.4	71.8	69.5

Table 6: Ablation study on nuScenes validation set. Modifications are added sequentially

Method	NDS (%)	mAP (%)
PointPillars [21] + FPN [24]	53.4	40.1
+ image branch [41]	54.3	41.3
+ MoCa	54.7	42.4
+ hybrid optimization	57.2	46.3
+ aligned pyramid fusion	57.4	47.1

Table 7: Comparison of image branches pre-trained by different detectors

Detector	Faster R-CNN	Mask R-CNN	Cascade	HTC
NDS (%)	57.4	57.6	57.9	58.1

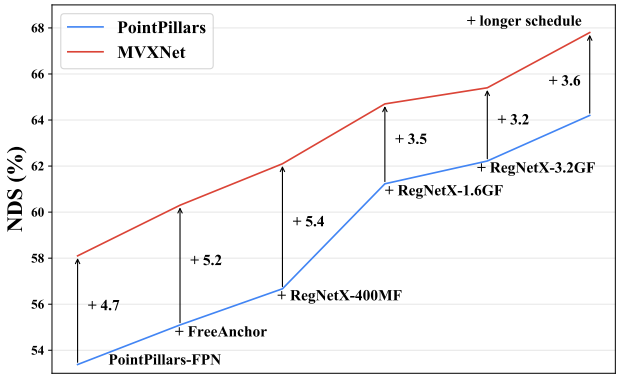

Figure 6: Improvement by MVX-Net on different baselines.

As shown in Table 6, MoCa, aligned pyramid feature fusion, and hybrid optimization are also beneficial on nuScenes dataset and the changes allows MVX-Net to surpass PointPillars by a large margin (7% mAP and 4% NDS).

Pre-training. We observe different effectiveness in the image branch for our experiments when it is pre-trained in Faster R-CNN [37], Mask R-CNN [15], Cascade Mask R-CNN [2], and HTC [3] (Table 7). Using similar backbones and necks, the image branch from HTC pre-trained on nuImages dataset [1] shows a gain of 0.7% NDS against the image branch from Faster R-CNN. Therefore, we adopt the image branch from HTC in the following experiments on nuScenes dataset.

Stronger PointPillars baselines. We further verify the enhanced MVX-Net with MoCa over stronger baselines. The original baseline is PointPillars with FPN in the neck. Then

Table 8: Comparison with previous methods on nuScenes validation set. 'Con. Veh.', 'Ped.', and 'T.C.' are the abbreviations of construction vehicle, pedestrian, and traffic cone, respectively. 'FA' means FreeAnchor [52] and '3×' means longer training schedule. NDS score, mAP, and APs of each categories are reported. The single class AP not reported in the paper is marked by '-'. The best results are bolded

Method	Modality	NDS (%)	mAP (%)	Car	Truck	Bus	Trailer	Con. Veh.	Ped.	Motor	Bicycle	T.C.	Barrier
PointPillars [21]	L	46.8	28.2	75.5	31.6	44.9	23.7	4.0	49.6	14.6	0.4	8.0	30.0
3D-CVF [50]	L + I	49.8	42.2	79.7	37.9	55.0	36.3	-	71.3	37.2	-	40.8	47.1
PointPillars [21]+FPN [24]	L	53.4	40.1	80.6	35.9	43.5	29.2	5.4	71.9	34.9	11.8	35	52.6
3DSSD [47]	L	56.4	42.6	81.2	47.2	61.4	30.5	12.6	70.2	36.0	8.6	31.1	47.9
PointPainting [42]	L + I	58.1	46.4	77.9	35.8	36.1	37.3	15.8	73.3	41.5	24.1	62.4	60.2
CenterPoint [49]	L	65.0	56.6	84.6	54.7	66	32.3	15.1	84.5	56.9	38.6	67.4	66.1
MoCa	L+I	58.1	47.9	82.4	41.5	49.6	28.6	9.1	79.1	50.3	27.2	49	61.9
MoCa with more improvements													
+ FA [52]		60.3	52.9	83.6	48.5	56.4	31.4	10.8	81.6	61.0	35.7	58.1	61.7
+ FA + RegNetX-400MF [36]		62.1	55.2	84.2	51.7	63.6	34.2	18.0	82.0	61.8	32.9	59.4	64.0
+ FA + RegNetX-1.6GF	L + I	64.7	58.4	85.2	55.5	64.5	35.0	20.8	84.5	68.0	42.8	62.3	65.1
+ FA + RegNetX-3.2GF		65.4	59.2	85.7	56.8	66.2	35.8	21.7	84.4	67.4	44.2	62.8	66.3
+ FA + RegNetX-3.2GF + 3×		**67.8**	**62.5**	**87.0**	**60.8**	**68.2**	**39.7**	**23.9**	**85.8**	**70.2**	**51.6**	**68.1**	**69.5**

Table 9: Comparison with published multi-modality methods on KITTI 3D test benchmark. 'Ensembled' indicates whether the results are ensembled by class-specialized detectors. The best results are bolded

Method	Ensembled	mAP (%)			Pedestrian			Cyclist			Car		
		Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
AVOD-FPN [20]	√	65.8	54.9	49.9	50.5	42.3	39.0	63.8	50.6	44.9	83.1	71.8	65.7
F-PointNet [32]		68.3	56.0	49.2	50.5	42.2	38.1	72.3	56.1	49.0	82.2	69.8	60.6
PointPainting [42]		70.0	58.8	53.6	50.3	41.0	37.9	77.6	63.8	55.9	82.1	71.7	67.0
F-ConvNet [44]		**73.8**	**61.6**	54.0	**52.2**	43.4	38.8	**82.0**	**65.1**	**56.5**	**87.4**	**76.4**	66.7
MoCa (ours)	×	71.0	60.2	**54.7**	50.9	**43.7**	**40.0**	76.1	61.0	53.4	86.0	75.9	**70.7**

we replace the head with FreeAnchor head [52] to predict 3D boxes. We further replace the backbone of PointPillars with RegNetX [36], including RegNetX-400MF, RegNetX-1.6GF, and RegNetX-3.2GF. As shown in Figure 6, despite the increasing performance of PointPillars due to enhancements applied to its backbone, neck, head, and longer training schedule, the enhanced MVX-Net equipped with MoCa still improves the baseline consistently. The results suggest the effectiveness and generalizability of MoCa and the beneficial practices.

4.3. Benchmark Results

nuScenes dataset. We compare our method with other published methods on the validation set of nuScenes dataset [1]. We report the performance of enhanced MVX-Net with MoCa, using the image branch from HTC pre-trained on nuImages dataset. The results show that MoCa surpasses PointPainting [42], the previous multi-modality state of the art, by **1.5%** mAP. We also report the performance of MoCa enhanced by FreeAnchor [52], RegNetX [36], and longer schedule with stronger augmentations. MoCa achieves new state-of-the-art results not only on the overall metric, but also on the AP of all the categories. The final performance of MoCa surparsses the previous best result achieved by a large model of CenterPoint [49] trained by CBGS [54], with an absolute improvement of **2.8%** NDS and **5.9%** mAP.

KITTI dataset. Lastly, we compare our method with other published methods on the KITTI 3D detection benchmark for completeness. Note that many previous works [20, 21, 27, 44, 46, 48] train **specialized models with different hyperparameters for different categories and ensemble their results on the benchmark**. However, using multiple detectors for multiple classes is not ideal for real-world applications. Therefore, in this work, we train all the models including all baselines on all three classes without tuning the models for specific categories. Table 9 shows that MoCa achieves promising performance among multi-modality methods. Despite a single model for all three classes, MoCa achieves competitive performance against other baselines that use an ensemble of class-specific detectors. On hard conditions, only MoCa obtains top-3 results of all the categories. In comparison, other methods do not exhibit such generalizability.

5. Conclusion

This paper investigates and discusses the pitfalls of applying cut and paste augmentation to multi-modality 3D object detection. We summarize a useful pipeline, multi-modality transformation flow, to ensure consistency during augmentations and enable more aggressive augmentation strategies. Under the pipeline, we validate the effectiveness of different augmentations and further present multi-modality cut and paste (MoCa) to bridge the gap of augmentations between multi-modality and single-modality 3D detectors. Under different strong baselines, our method improves the performance consistently and achieves new state-of-the-art performance on nuScenes dataset.

Appendix A. Implementation Details

Projection from LiDAR to image. For feature fusion, the point cloud is first transformed from LiDAR coordinates P_{lidar} to camera coordinates and then projected to image pixel coordinates P_{img}.

On KITTI dataset [14], the projection is calculated as follows:

$$P_{img} = P^0_{rect} R^0_{rect} T_{cam \leftarrow lidar} P_{lidar}, \quad (3)$$

where $T_{cam \leftarrow lidar}$ is the transformation matrix from LiDAR coordinates to camera coordinates, R^0_{rect} is the rectifying rotation matrix of the left camera, and P^0_{rect} is the calibration matrix of the left camera.

On nuScenes dataset [1], the points in LiDAR coordinates are first transformed to the ego car's global coordinates since the LiDAR (20Hz) and camera (12Hz) work at different frequencies. Therefore, the transformation is calculated as follows:

$$P_{img} = T_{cam \leftarrow ego} T_{ego_c \leftarrow ego_l} T_{ego \leftarrow lidar} P_{lidar}, \quad (4)$$

where $T_{ego \leftarrow lidar}$ is the transformation matrix from LiDAR to the ego pose at timestamp t_l when the LiDAR frame is recorded, $T_{ego_c \leftarrow ego_l}$ is the transformation matrix from ego pose at timestamp t_l to the ego pose at timestamp t_c when the camera frame is captured, and $T_{cam \leftarrow ego}$ is the transformation matrix from ego pose at timestamp t_c to the camera.

Training details. On KITTI dataset, both SECOND [46] and MVX-Net [41] are trained by 80 epochs with a batch size of 16. We adopt a half-period cosine schedule [28] for learning rate decaying and use a linear warm-up strategy in the first 1K iterations. The initial learning rate is 0.003 for all the 3D detectors that use ADAM [19] optimizer. When adopting *hybrid optimization* (Sec. 3.3) for MVX-Net, we train the image branch using SGD optimizer with momentum and the initial learning rate is 0.05. The hyperparameters for point cloud branch remain the same as those for SECOND.

For nuScenes dataset [1], both PointPillars [21] and MVX-Net [41] are trained by 20 epochs with batch sizes of 32 and 16, respectively. Notably, this is different from the official implementation [1], where PointPillars [21] is trained by 125 epochs, which costs much time. We adopt step learning rate decaying schedule following the practice in mmdetection [4], *i.e.*, the learning rate is decayed by 0.1 after the 16th and 19th epoch, respectively. The initial learning rate is 0.001 for all the 3D detectors using ADAM [19] optimizer. When training MVX-Net, we use *hybrid optimization*, where the image branch is trained by SGD optimizer with momentum and the initial learning rate is 0.0012. The hyperparameters for point cloud branch remain the same as those for PointPillars.

Because one training sample for MVX-Net contains six images from multiple views and one LiDAR point cloud frame, each GPU can only contain one training sample during each iteration. This degrades the performance siginificantly because the batch size in one GPU is so small that the inaccurate statistics in Batch Normalization (BN) [17] affects the training process. Therefore, we use Syncronized Batch Normalization (SyncBN) [26] to solve this issue. We report all the results of our methods using SyncBN in Table 6, 7, and 8 on nuScenes dataset. We do not use SyncBN for models on KITTI dataset because the batch size is 2 in each GPU as there is only one image and one LiDAR point cloud frame in one training sample.

Appendix B. Experiments

Ablation study. We provide the detailed results of Tables 5 and 6 of the main text for further analysis, as shown in Tables 10 and 11. Comparing APs on each category shows that our method brings more improvements in hard conditions and challenging categories for both KITTI and nuScenes dataset.

Hybrid optimization. In Table 4, to verify the versatility of hybrid optimization and find a good training strategy, we adopt three training strategies for pre-training image feature extractors and jointly training multi-modality detectors. The first strategy trains a Faster R-CNN [37] on COCO2017 dataset by the multi-scale 3× schedule [4, 45] with ResNet-50 [16] pre-trained on ImageNet [38]. Then we train Faster R-CNN on the subset of COCO training split that only contains three classes: people (for pedestrian in KITTI), bicycle (for cyclist in KITTI), and car, following Frustum-PointNet [32]. The weights of ResNet-50 and FPN pre-trained in the Faster R-CNN are adopted in the image branch of the multi-modality detector for joint training. The second strategy further fine-tunes the Faster R-CNN using *MoCa* for 20 epochs with mixed IoF thresholds (Table 2) before joint training. The third strategy uses a similar pre-training strategy as the second one but freezes the ResNet-50 backbone in joint training. Table 4 shows that the synergy of the third strategy and hybrid optimization works best among these training and optimization strategies.

Pre-training on nuScenes dataset. The image branch from HTC pre-trained on nuImages dataset [1] shows a gain of 0.7% NDS against that from Faster R-CNN in Table 7 of the main text. Those pre-trained detectors are first trained by 20 epochs using the standard setting on COCO dataset and released by mmdetection [4]. We further fine-tune those detectors on the nuImages dataset [1] by 20 epochs using similar hyperparameters as those for the COCO dataset. When jointly training the multi-modality detectors, the weights in ResNet-50 backbone and FPN of those pre-trained detectors are adopted to initialize the image branch.

Table 10: Ablation study of each component on KITTI validation dataset.

Method	mAP (%)			Pedestrian			Cyclist			Car		
	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
SECOND [46]	78.3	68.1	63.4	68.1	61.1	54.0	79.4	65.8	60.8	87.5	77.4	75.5
+ image branch [41]	71.9	60.5	59.1	62.7	55.6	53.2	65.7	49.1	48.4	87.2	76.9	75.5
+ MoCa	80.9	70.2	65.2	68.6	61.9	54.7	86.0	71.2	65.0	87.9	77.6	76.0
+ hybrid optimization	81.2	70.8	65.7	86.6	62.2	55.1	88.6	72.7	66.3	97.8	77.5	75.6
+ aligned pyramid fusion	81.4	71.8	69.5	87.5	64.5	60.9	88.9	73.4	71.6	90.8	77.5	76.0

Table 11: Ablation study on nuScenes validation set. Modifications are added sequentially. 'Con. Veh.', 'Ped.', and 'T.C.' are the abbreviations of construction vehicle, pedestrian, and traffic cone, respectively. NDS score, mAP, and APs of each categories are reported

Method	NDS (%)	mAP (%)	Car	Truck	Bus	Trailer	Con. Veh.	Ped.	Motor	Bicycle	T.C.	Barrier
PointPillars [21] + FPN [24]	53.4	40.1	80.6	35.9	43.5	29.2	5.4	71.9	34.9	11.8	35.0	52.6
+ image branch [41]	54.3	41.3	81.4	34.6	41.0	24.1	5.1	76.2	42.6	13.1	42.2	53.0
+ MoCa	54.7	42.4	81.7	37.2	44.0	24.3	4.0	77.1	42.9	15.9	43.3	53.6
+ hybrid optimization	57.2	46.3	81.4	39.9	45.5	30.8	5.6	76.8	50.0	27.1	48.5	57.7
+ aligned pyramid fusion	57.4	47.1	82.3	40.2	48.0	30.5	5.2	78.3	51.9	24.1	50.3	60.6

Table 12: Step-by-step results of stronger baselines on nuScenes validation set. 'Con. Veh.', 'Ped.', and 'T.C.' are the abbreviations of construction vehicle, pedestrian, and traffic cone, respectively. 'FA' means FreeAnchor [52] and '3×' means longer training schedule. NDS score, mAP, and APs of each categories are reported. The best results are bolded

Method	Modality	NDS (%)	mAP (%)	Car	Truck	Bus	Trailer	Con. Veh.	Ped.	Motor	Bicycle	T.C.	Barrier
PointPillars [21] + FPN [24]	L	53.4	40.1	80.6	35.9	43.5	29.2	5.4	71.9	34.9	11.8	35	52.6
+ FA [52]		55.1	43.7	81.5	40.0	50.0	29.4	9.2	74.3	44.5	16.5	39.6	52.4
+ FA + RegNetX-400MF [36]		56.7	45.5	82.0	41.9	50.7	32.3	11.0	75.4	50.1	19.1	44.1	48.8
+ FA + RegNetX-1.6GF		61.2	51.4	83.2	48.2	60.5	30.4	16.6	78.1	59.4	25.9	49.1	62.3
+ FA + RegNetX-3.2GF		62.2	52.1	83.6	51.1	62.3	36.0	17.3	78.2	56.1	24.7	50.0	62.0
+ FA + RegNetX-3.2GF + 3×		64.2	56.9	85.5	54.9	66.8	35.4	22.2	81.2	62.4	35.6	59.2	65.4

Table 13: Performance of top-3 submissions to the 3rd nuScenes Detection Challenge on test set. 'Con. Veh.', 'Ped.', and 'T.C.' are the abbreviations of construction vehicle, pedestrian, and traffic cone, respectively. 'I', 'L', and 'R' means whether data from image, LiDAR, and RADAR is used, respectively. NDS score, PKL [29], mAP, and APs of each categories are reported

Method	Modality	PKL	NDS (%)	mAP (%)	Car	Truck	Bus	Trailer	Con. Veh.	Ped.	Motor	Bicycle	T.C.	Barrier
CenterPoint v2 [42, 49]	I + L + R	0.581	71.4	67.1	87.0	57.3	69.3	60.4	28.8	90.4	71.3	49.0	86.8	71.0
PointAugmenting	I + L	0.595	71.1	66.8	87.5	57.3	65.2	60.7	28.0	87.9	74.3	50.9	83.6	72.6
Ours	I + L	0.574	70.9	66.6	86.7	58.6	67.2	60.3	32.6	87.1	67.8	52.0	81.3	72.3

Stronger PointPillars baselines. We adopt stronger PointPillars baselines in MVX-Net to verify our method's generalizability by enhancing the head, backbone, and training schedule of PointPillars. For completeness, we also put the detailed step-by-step results of Figure 6 in Table 12. The corresponding results of MoCa are in Table 8. Notably, our PointPillars baseline already achieves very high performance on the validation set. However, our methods consistently improve performance, especially on challenging classes for LiDAR-based detectors, such as bicycles, motorcycles, traffic cones, and pedestrians.

Appendix C. nuScenes Detection Challenge

Based on the model that achieves the best result in Table 8 (last row), we ensemble the other five models with test-time augmentation and submit the results of test set to the 3rd nuScenes detection challenge. The challenge introduces Planning KL-Divergence (PKL) [29], a novel planning-based metric that measures the influence of predictions on the downstream tasks in autonomous driving. The lower value of PKL means that the predictions are more similar to ground truth and are more suitable for planning. We list the top-3 entries including our submission of the challenge in Table 13. The results show that our submission achieves the best PKL results on the test set, which has the least deviation from the ground truth and is more favorable for autonomous driving.

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. *CoRR*, abs/1903.11027, 2019. 1, 2, 3, 5, 6, 7, 8, 9

[2] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: Delving into high quality object detection. In *CVPR*, 2018. 7

[3] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. Hybrid task cascade for instance segmentation. In *CVPR*, 2019. 5, 7

[4] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tian-

heng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 2, 6, 9

[5] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals using stereo imagery for accurate object class detection. *TPAMI*, 2018. 2

[6] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3D object detection network for autonomous driving. In *CVPR*, 2017. 2, 5

[7] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Fast Point R-CNN. In *ICCV*, 2019. 2

[8] Hyunggi Cho, Young-Woo Seo, B. V. K. Vijaya Kumar, and Ragunathan Rajkumar. A multi-sensor fusion system for moving object detection and tracking in urban driving environments. In *ICRA*, 2014. 1

[9] Terrance Devries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *CoRR*, abs/1708.04552, 2017. 2

[10] Nikita Dvornik, Julien Mairal, and Cordelia Schmid. Modeling visual context is key to augmenting object detection datasets. In *ECCV*, 2018. 2, 3, 4

[11] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *ICCV*, 2017. 2, 3, 4

[12] Hao-Shu Fang, Jianhua Sun, Runzhong Wang, Minghao Gou, Yong-Lu Li, and Cewu Lu. InstaBoost: Boosting instance segmentation via probability map guided copy-pasting. In *ICCV*, 2019. 2, 3

[13] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *I. J. Robotics Res.*, 2013. 1, 3

[14] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 2, 3, 5, 9

[15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *ICCV*, 2017. 5, 7

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2, 5, 6, 7, 9

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 9

[18] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *NeurIPS*, 2015. 5

[19] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 9

[20] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L. Waslander. Joint 3D proposal generation and object detection from view aggregation. In *IROS*, 2018. 2, 8

[21] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. PointPillars: Fast encoders for object detection from point clouds. In *CVPR*, 2019. 2, 3, 4, 5, 7, 8, 9, 10

[22] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3D object detection. In *CVPR*, 2019. 1, 2, 3, 5

[23] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *ECCV*, 2018. 1, 2, 3, 4, 5

[24] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 2, 5, 6, 7, 8, 10

[25] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 3, 5

[26] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *CVPR*, 2018. 9

[27] Zhe Liu, Xin Zhao, Tengteng Huang, Ruolan Hu, Yu Zhou, and Xiang Bai. TANet: Robust 3D object detection from point clouds with triple attention. In *AAAI*, 2020. 2, 3, 8

[28] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *ICLR*, 2017. 9

[29] Jonah Philion, Amlan Kar, and Sanja Fidler. Learning to evaluate perception models using planner-centric metrics. In *CVPR*, 2020. 10

[30] Charles R. Qi, Xinlei Chen, Or Litany, and Leonidas J. Guibas. ImVoteNet: Boosting 3d object detection in point clouds with image votes. In *CVPR*, 2020. 2

[31] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3D object detection in point clouds. In *ICCV*, 2019. 2

[32] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum PointNets for 3D object detection from RGB-D data. In *CVPR*, 2018. 2, 4, 5, 8, 9

[33] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *CVPR*, 2017. 2, 5

[34] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 2, 5

[35] Lu Qi, Li Jiang, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Amodal instance segmentation with kins dataset. In *CVPR*, 2019. 6

[36] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollar. Designing network design spaces. In *CVPR*, 2020. 8, 10

[37] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 5, 6, 7, 9

[38] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015. 3, 5, 9

[39] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN: Point-voxel feature set abstraction for 3D object detection. In *CVPR*, 2020. 2, 3

[40] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. PointRCNN: 3D object proposal generation and detection from point cloud. In *CVPR*, 2019. 1, 2, 3, 5

[41] Vishwanath A. Sindagi, Yin Zhou, and Oncel Tuzel. MVXNet: Multimodal voxelnet for 3D object detection. In *ICRA*. IEEE, 2019. 1, 2, 3, 4, 5, 6, 7, 9, 10

[42] Sourabh Vora, Alex H. Lang, Bassam Helou, and Oscar Beijbom. PointPainting: Sequential fusion for 3D object detection. In *CVPR*, 2020. 1, 2, 4, 8, 10

[43] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark E. Campbell, and Kilian Q. Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3D object detection for autonomous driving. In *CVPR*, 2019. 2

[44] Zhixin Wang and Kui Jia. Frustum ConvNet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection. In *IROS*, 2019. 2, 8

[45] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 9

[46] Yan Yan, Yuxing Mao, and Bo Li. SECOND: Sparsely embedded convolutional detection. *Sensors*, 2018. 2, 3, 4, 5, 6, 7, 8, 9, 10

[47] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3DSSD: Point-based 3D single stage object detector. 2020. 2, 3, 8

[48] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. STD: Sparse-to-dense 3D object detector for point cloud. In *ICCV*, 2019. 1, 2, 3, 4, 8

[49] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3D object detection and tracking. *CoRR*, abs/2006.11275, 2020. 8, 10

[50] Jin Hyeok Yoo, Yecheol Kim, Ji Song Kim, and Jun Won Choi. 3D-CVF: Generating joint camera and lidar features using cross-view spatial feature fusion for 3D object detection. 2020. 8

[51] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. CutMix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019. 2

[52] Xiaosong Zhang, Fang Wan, Chang Liu, Rongrong Ji, and Qixiang Ye. FreeAnchor: Learning to match anchors for visual object detection. In *NeurIPS*, 2019. 8, 10

[53] Yin Zhou and Oncel Tuzel. VoxelNet: End-to-end learning for point cloud based 3D object detection. In *CVPR*, 2018. 2, 3

[54] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced grouping and sampling for point cloud 3D object detection. *CoRR*, abs/1908.09492, 2019. 8

[55] Barret Zoph, Ekin D. Cubuk, Golnaz Ghiasi, Tsung-Yi Lin, Jonathon Shlens, and Quoc V. Le. Learning data augmentation strategies for object detection. *CoRR*, abs/1906.11172, 2019. 6