Range Adaptation for 3D Object Detection in LiDAR

Ze Wang[1], Sihao Ding[2], Ying Li[2], Minming Zhao[2], Sohini Roychowdhury[2], Andreas Wallin[2], Guillermo Sapiro[1], and Qiang Qiu[1]

[1]Duke University
{ze.w, guillermo.sapiro, qiang.qiu}@duke.edu
[2]Volvo Car Technology USA
{sihao.ding, ying.li.5, minming.zhao, sohini.roy.chowdhury, andreas.wallin1}@volvocars.com

Abstract

LiDAR-based 3D object detection plays a crucial role in modern autonomous driving systems. LiDAR data often exhibit severe changes in properties across different observation ranges. In this paper, we explore cross-range adaptation for 3D object detection using LiDAR, i.e., far-range observations are adapted to near-range. This way, far-range detection is optimized for similar performance to near-range one. We adopt a bird-eyes view (BEV) detection framework to perform the proposed model adaptation. Our model adaptation consists of an adversarial global adaptation, and a fine-grained local adaptation. The proposed cross-range adaptation framework is validated on three state-of-the-art LiDAR based object detection networks, and we consistently observe performance improvement on the far-range objects, without adding any auxiliary parameters to the model. To the best of our knowledge, this paper is the first attempt to study cross-range LiDAR adaptation for object detection in point clouds. To demonstrate the generality of the proposed adaptation framework, experiments on more challenging cross-device adaptation are further conducted, and a new LiDAR dataset with high-quality annotated point clouds is released to promote future research.

1. Introduction

Modern autonomous driving systems rely heavily on accurate and robust perception, where LiDAR-based 3D object detection plays an indispensable role. Object detection in point clouds has appealing advantages such as accurate distance encoded in points and scale consistency in a 3D space. Several recent 3D object detection methods report promising results on public LiDAR datasets [**?**, **?**]. However, within a point cloud, the density of points degrades significantly with the distance to the sensor, leading to inferior performance for far-range objects. Perception of objects in long ranges is crucial for planning ahead while driving, in particular in highways, and new LiDAR technology is being developed to measure far away objects.

In this paper, addressing object detection in point clouds, we propose to perform cross-range adaptation for deep model learning, obtaining for far-range objects similar performance to near-range detection While domain adaptation on optical images has been widely studied, it is still a highly challenging task to perform adaptation on point clouds. First, the un-ordered and irregular structure of point clouds differs significantly from the gridded structure of optical images, which prevents methods in the image space, e.g., cGAN based methods, from being directly utilized on point clouds. Second, certain unique properties of point clouds, e.g., the scale consistency, bring appealing advantages over optical images, and they need to be fully exploited.

Instead of directly performing adaptation on raw point cloud, we propose model adaptation to be applied on the intermediate layer of a deep network for point cloud object detection. In the proposed framework, a key step is to align cross-range gridded features at an intermediate layer of a deep network, so that the parameters in the preceding layers are tuned at a low cost to handle range shifts, while the subsequent layers remain shared. Specifically, we use near-range areas as the source domain, and improve the feature and the detection accuracy of far-range areas which serve as the target domain.

We adopt combinations of local and global adaptation. For the global adaptation, we adopt adversarial learning to align the feature in the network feature space. Such methods have delivered outstanding performances on image-based object classification and segmentation. However, as observed for images, adaptation results depend heavily on how complicated the task is. For example, satisfactory results are obtained for adapting digit images, while limited per-

formance is observed for sophisticated scenarios like image segmentation. These observations motivate us to explore beyond adversarial learning and exploit the properties of point cloud for fine-grained adaptation as detailed next.

Beyond the global adaptation, we should note that while the size of an object varies with distance in optical images, it stays constant in a point cloud. By exploiting such scale consistency property of point clouds, we propose to mine in the point cloud space for matched local regions across the source and target ranges, and then perform fine-grained local adaptation in the corresponding feature space.

We perform extensive experiments on public 3D object detection datasets and methods. The obtained results validate the proposed framework as an effective method for point cloud range adaptation, including more challenging cross-device adaptation. Beside the superiority on performance, our method does not introduce any auxiliary network layers for the detection model, which enables the adapted object detector to run at the same speed and memory consumption as the original model but with significantly superior detection accuracy.

Our contributions are summarized as follow:

- We propose cross-range adaptation to significantly improve LiDAR-based far-range object detection.
- We combine fine-grained local adaptation and adversarial global adaptation for 3D object detection models.
- To the best of our knowledge, this work is the first attempt to study adaptation for 3D object detection in point clouds.
- We release a new LiDAR dataset with high-quality annotated point clouds to promote future research on object detection and model adaptation for point clouds.

2. Related Work

2.1. 3D Object Detection

Object detection in point clouds is an intrinsically three dimensional problem. As such, it is natural to deploy a 3D convolutional network for detection, which is the paradigm of several early works [**?**, **?**]. While providing a straightforward architecture, these methods are slow; e.g. Engelcke *et al.* [**?**] require 0.5s for inference on a single point cloud. Most recent methods improve the runtime by projecting the 3D point cloud either onto the ground plane [**?**, **?**] or the image plane [**?**]. In the most common paradigm the point cloud is organized in voxels and the set of voxels in each vertical column is encoded into a fixed-length, hand-crafted feature to form a pseudo-image which can be processed by a standard image detection architecture. Some notable works here include MV3D [**?**], AVOD [**?**], PIXOR [**?**], and Complex-YOLO [**?**], which all use variations on the same fixed encoding paradigm as the first step of their architectures. The first two methods additionally fuse the lidar features with image features to create a multimodal detector. The fusion step used in MV3D and AVOD forces them to use two-stage detection pipelines, while PIXOR and Complex-YOLO use single stage pipelines. Some recent progresses have significantly improved the both the accuracy and the speed on LiDAR-only object detection. SECOND [**?**] adopt sparse convolutional layers that process 3D feature at faster speed with much less memory consumption. PointPillars [**?**] adopt a novel point cloud encoding layer that enable a high speed and high quality transformation from un-ordered points to gridded representations.

2.2. Domain Adaptation

Recently, we have witnessed great progress on domain adaptation for deep neural networks. The achievements on deep domain adaptation generally follow two directions. The first direction is to do domain adaptation in a single network, where parameters are shared across domains, while the network is forced to produce domain-invariant features by minimizing additional loss functions in the network training [**?**, **?**, **?**, **?**, **?**]. The additional losses are imposed to encourage similar features from source and target domains. The Maximum Mean Discrepancy (MMD) [**?**] emerged as a popular metric of domain distance, and was adopted in [**?**] as a domain confusion loss to encourage small distance between the source and the target domain final features in a unified network structure and to prevent the network from overfitting to the source domain. The MK-MMD [**?**] is adopted in [**?**] as the discrepancy metric between the source and the target domains. Beyond the first order statistic, second-order statistics are utilized in [**?**].

In addition to the hand-crafted distribution distance metrics, many recent efforts [**?**, **?**] resort to adversarial training by applying a feature discriminator that is trained alternatively with the main network to do a binary classification of distinguishing the features from the source and the target domain. The main network is trained to fool the feature discriminator so that domain-invariant features contain no information helping the discriminator to decide which domain the features come from.

3. Cross-range Adaptation

The cross-range adaptation is motivated by the fact that the LiDAR-based detection accuracy degrades significantly for far-range objects. While training a deep network for 3D detection, we usually have significantly more near-range training samples, which dominates the training loss. As shown in Figure 1, near-range objects are represented with significantly denser points than far-range ones. Thus, the obtained model usually exhibits superior near-range detec-

tion performance, but poor generalization to far-range detection, which needs to be addressed, as self-driving moves to highways and as new sensors with far-range capabilities are being developed.

A point cloud is usually represented by a set of unordered points with continuous values (x, y, z) denoting the cartesian coordinates in 3D space, and optional additional values carrying other physical properties, e.g., reflection value r in the KITTI dataset [**?**]. Directly processing point clouds using off-the-shelf image-based object detection methods is sub-optimal since point clouds are essentially irregular and unordered, which are not suitable to be processed directly using convolutional neural networks designed for gridded features. Transforming point clouds to an evenly spaced grid representation is usually adopted for object detection in LiDAR [**?**, **?**, **?**].

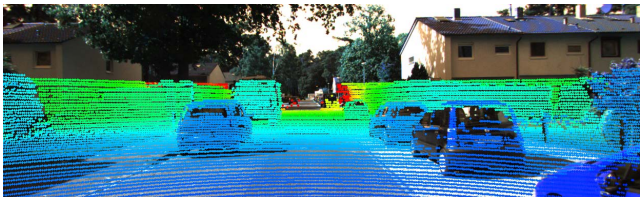
In a gridded feature space, our proposed objective is to align features across different ranges, at a chosen intermediate layer of a 3D object detection network. Such chosen layer is referred to as the *aligned layer*. Layers preceding such aligned layer, addressed as *adapted layers*, are tuned to encourage far-range observed objects to produce consistent features as similar objects observed at a near range. Layers after the aligned layer, address as *shared layers*, determine detection results based on the aligned features.

To improve the generalization to far-range object detection, we apply both adversarial global adaptation and fine-grained local adaptation. For global adaptation, as shown in Figure 2, we set the source domain and the target domain to be near-range features and far-range features of the adapted layer, and use a feature discriminator to promote consistent feature appearance across domains as discussed in Sec. 3.1. we then apply an attention mechanism to further align far-range features to near-range features of similar objects, exploiting the unique invariance of point clouds, which is detailed in Section 3.2. The loss of both the global and local adaptations are jointly propagated back to all the layers preceding to the adapted layer. Note that the proposed framework introduces no additional auxiliary parameters to a deep model.

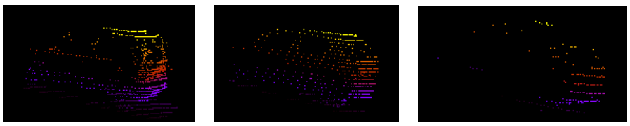
3.1. Adversarial Global Adaptation

After transforming point clouds to an evenly spaced grid representation, adversarial training can be adopted for cross-range adaptation. Specifically, a feature discriminator is imposed at the aligned layer to tune parameters at all preceding layers, so that features from far-range objects become as if they come from near-range ones. The feature discriminator is implemented as a classifier C, which takes both near-range and far-range features as inputs, and is trained to identify which range each feature comes from.

(a) Original image.



(b) Image with projected points.



(c) Distance: 8m Number of points: 1305 (d) Distance: 13m Number of points: 606 (e) Distance: 26m Number of points: 167

Figure 1. Illustration of the point density. (b) is generated by projecting the point cloud onto the image, which clearly shows that far-range objects are only covered by a very small amount of points. We further select three objects in different distances and plot the points in the corresponding 3D boxes in (c) (d) and (e).

The loss function of the discriminator is expressed as

$$\mathscr{L}_C(y_n) = -\frac{1}{N}\sum_{N}^{1}[y_n \log(\hat{y}_n) + (1-y_n)\log(1-\hat{y}_n)],$$

where N is the total number of features. $y_n \in [0, 1]$ indicates a range label with '0' for far-range and '1' for near-range, and $\hat{y_n}$ is the predication from the feature discriminator. The parameters of adapted layers and the feature discriminator are updated alternatively. The loss of the detection network becomes from $\mathscr{L}_D$ to $\mathscr{L}_D(1-y_n)$, which encourages the adapted layers to produce unified features to fool the feature discriminator. Following [**?**], we adopt a patch-based discriminator, which only penalizes structure at the scale of feature patches.

3.2. Fine-grained Local Adaptation

The above adversarial adaptation provides a global feature alignment, a fine-grained local adaptation is proposed to further improve far-range performance. In point clouds, an object has a consistent scale, regardless of viewing angles and positions. In other words, LiDAR observation patterns in the far-range areas repeat in near-range areas for similar objects but with much denser points. Thus, we can mine region pairs of similar patterns to further perform

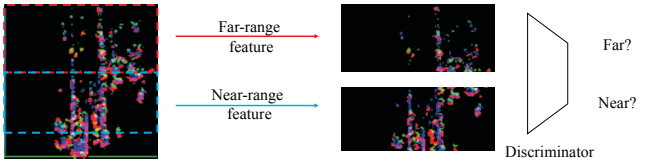


Figure 2. Illustration of the global adaptation for cross-range. The source and the target domain are the near-range feature and the far-range feature, respectively. A feature discriminator is applied to promote range-invariant features. Note that we leave out the area too close to the sensor where unique patterns exist, e.g., the shadow of the data collecting car, that can lead to a quick overfitting to the feature discriminator.

region-based feature adaptation. Note that similar tasks become significantly harder for images, as we need to simultaneously handle changes in scales, viewing angles, illumination, etc. This is therefore a unique characteristic of point clouds in general, and LiDAR in particular, we here explicitly exploit.

Given object annotations, during training, we divide objects in each mini-batch into two groups based on the range associated, i.e., a far-range group $\mathcal{F}$ for objects beyond a range threshold, and a near-range group $\mathcal{N}$ for objects within the range threshold. To further encourage a far-range object to share consistent features, at the aligned layer, as similar objects at a near range, we compute the targeted feature of a far-range object as a weighted average of all object features in the near-range group. The weight is determined based on the object similarity in the original point cloud space. Specifically, each object is represented as $\{\mathbf{o}_i, \mathbf{f}_i\}$, where $\mathbf{o}_i$ denotes its representation in a point cloud space parametrized by the width w, height h, and yaw-angle r; and $\mathbf{f}_i$ denotes its feature at the aligned layer of a deep network. Given a far-range object $\{\mathbf{o}_t, \mathbf{f}_t\}$, its targeted feature $\hat{\mathbf{f}}_t$ is determined as,

$$\hat{\mathbf{f}}_t = \sum_{i \in \mathcal{N}} w_{it} \mathbf{f}_i, \tag{1}$$

and the weight w_{it} is computed as,

$$w_{it} = \frac{e^{|\mathbf{o}_i - \mathbf{o}_t|}}{\sum_{j \in \mathcal{N}} e^{|\mathbf{o}_j - \mathbf{o}_t|}}. \tag{2}$$

The final optimization loss then becomes

$$\mathscr{L}_l = \sum_{t \in \mathcal{F}} ||\mathbf{f}_t - \hat{\mathbf{f}}_t||^2, \tag{3}$$

which minimizes the distance to the corresponding target feature for each far-range object. Note that we cut off the gradient propagates through $\hat{\mathbf{f}}_t$ to prevent the network from aligning cross-range features by degrading near-range features.

4. Experiments

In this section, we present experiments to validate the effectiveness of the proposed framework.

4.1. 3D Object Detection Methods

We evaluate the proposed adaptation methods on several state-of-the-art BEV object detection frameworks. Despite significantly different network architectures adopted in these frameworks, each can be briefly described as a two-stage network as shown in Figure 3, where the first stage is to transfer the un-ordered and irregular structure into a pseudo-image representation, and a standard object detection head is then applied for 3D box prediction as the second stage. Note that the first stage does not have to be a parametric network, e.g., for Complex-YOLO [**?**], the pseudo-image is generated using predefined rules without learning.

We select three network architectures as baseline models in the experiments, and demonstrate that the detection performance of all three models can be significantly improved with the proposed adaptation method, without any additional parameters to the models. Complex-YOLO [**?**] is selected as it is an effective method that uses hand-craft rules to generate the pseudo-image. VoxelNet [**?**] is selected as it is a powerful network architecture that motivated several follow-up methods. SECOND [**?**] is selected as it reports at the moment the best detection performance. These three network architectures share a similar detection pipeline as shown in Figure 3. In the encoder stage, Complex-YOLO generates a pseudo-image using predefined rules, SECOND and VoxelNet adopt operations on local regions, e.g., the VFE layers and the sparse convolutional layers. Such operations are not suitable for performing adaptation since they are all local operations that operate on a small region on the feature map. Therefore, we propose to apply the adaptation on the intermediate layer of the decoder stage as shown in Figure 3, since the corresponding layer has both strong semantic encoded and compact feature size.

4.2. Quantitative Results

We apply the proposed adaptation framework on the above three network models, and report the quantitative results for fair comparisons.

Dataset. The experiments for cross-range adaptation are performed on the KITTI benchmark [**?**]. We follow the standard settings as in [**?**] to split the provided 7,481 samples into a training set of 3,712 samples and an evaluation set of 3,769 samples. The evaluation of a detector for KITTI 3D object detection is performed on three levels of difficulty: easy, moderate, and hard. The difficulty assessment is based on the object heights, occlusion, and truncation.

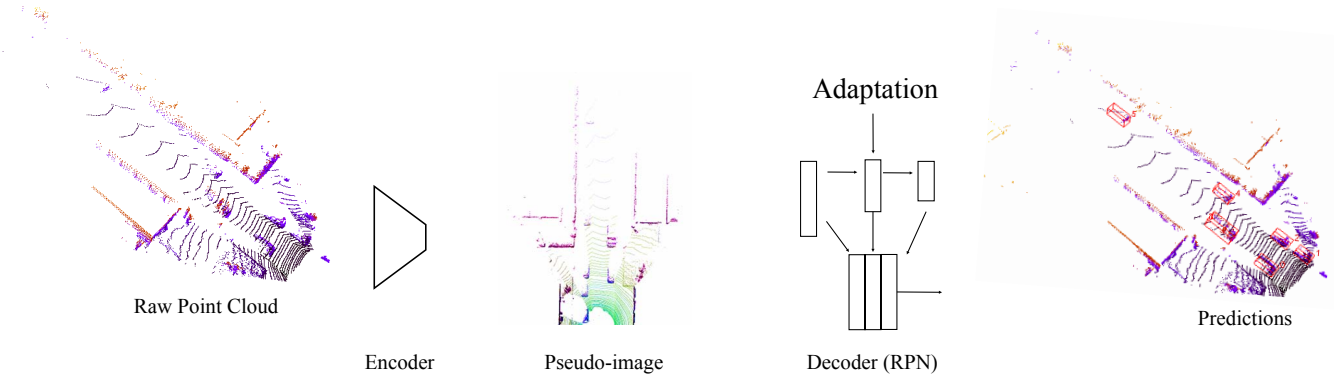


Figure 3. Overview of the two-stage BEV object detection pipeline. Different methods apply different techniques to transfer un-ordered point clouds to pseudo-images in either parametric or non-parametric ways. A decoder which is usually formed as a variant of classic detection networks is then applied to generate the final prediction. Our adaptation methods are performed at an intermediate layer of the decoder stage.

	Methods	Near-range (0-40m)			Full-range (0-70m)		
		Easy	Moderate	Hard	Easy	Moderate	Hard
3D AP	Complex-YOLO (w/o)	85.42	76.33	69.12	85.24	73.51	68.33
	Complex-YOLO (w)	85.90	77.01	72.98	85.89	77.02	71.43
	VoxelNet (w/o)	88.12	77.12	75.42	87.98	76.12	74.42
	VoxelNet (w)	**89.04**	78.12	76.01	**89.02**	77.98	75.82
	SECOND (w/o)	88.28	85.21	77.57	88.07	77.12	75.27
	SECOND (w)	88.81	**85.84**	**78.01**	88.80	**78.31**	**76.16**
2D AP	Complex-YOLO (w/o)	90.62	88.89	87.12	90.48	88.48	86.76
	Complex-YOLO (w)	90.62	88.91	87.12	90.61	88.82	88.24
	VoxelNet (w/o)	90.25	89.98	89.15	89.46	87.90	87.72
	VoxelNet (w)	90.28	90.02	89.65	90.07	89.19	88.42
	SECOND (w/o)	90.77	90.32	89.15	90.26	89.19	88.08
	SECOND (w)	**90.78**	**90.40**	**89.94**	**90.78**	**89.69**	**88.83**
BEV AP	Complex-YOLO (w/o)	89.72	88.99	87.66	89.45	80.90	79.49
	Complex-YOLO (w)	89.92	89.05	88.08	89.64	82.01	81.85
	VoxelNet (w/o)	89.72	88.99	87.66	89.72	87.37	79.23
	VoxelNet (w)	89.74	89.12	**88.18**	89.72	**88.17**	80.02
	SECOND (w/o)	88.62	87.81	86.39	88.62	86.25	86.21
	SECOND (w)	**90.34**	**89.37**	87.67	**90.32**	87.82	**87.51**

Table 1. Average precisions. Three metrics including 3D bounding box, 3D bounding box, and BEV are reported. Best results are marked in bold.

Metrics. Following the official KITII evaluation detection metrics, we report average precision (AP) on 2D box, bird's eye view (BEV), and 3D box. The results on 2D box are computed by projecting the 3D boxes on to the image planes, and calculating the average precision in 2D space. Note that the 3D box precision is the primary metric in our work.

Implementations. Since there is no official implementation for Complex-YOLO [**?**] and VoxelNet [**?**], we follow the descriptions in the papers and reimplement the networks. For the implementation of SECOND, we directly use the authors' publicly available implementation for a fair comparison.[1] All the experiments are implemented in PyTorch [**?**], and are trained on a single GTX 1080Ti graphic

[1] https://github.com/traveller59/second.pytorch

Methods	3D AP			2D AP			BEV AP		
	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
SECOND (w/o)	0.0	5.52	5.52	0.0	9.09	12.30	0.0	10.91	11.76
SECOND (w)	0.0	**7.02**	**7.02**	0.0	**13.64**	**13.64**	0.0	**13.22**	**13.22**

Table 2. Far-range (60-70m) only comparisons. Note the for the easy level, the AP is always 0 since there is no easy object in this area.

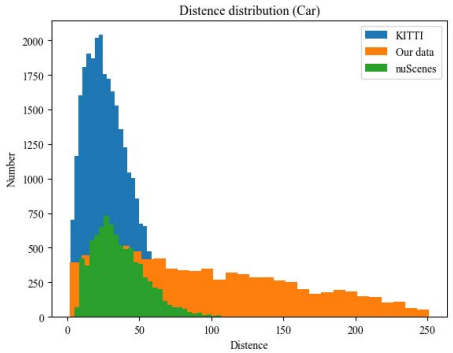


Figure 4. Histograms for the distance of cars to the sensor for all the three datasets. We choose the car class for illustration.

card.

Since all these three networks we select use different configurations for object categories, we only report detection results on cars, as cars are the object with a significantly dominant amount in the dataset for a reliable comparison. Following [**?**], we set the full scale detection range to be $[-3, 1] \times [-40, 40] \times [0, 70.4]$ meters along the Z, Y, X axis, respectively. Note that as clearly indicated in [**?**], Complex-YOLO neglects the objects that are farther than 40m for the sake of efficiency. We use the same range of $[0, 70.4]$ meters along the X axis across all the experiments for a fair comparison. Based on the distribution presented in Figure 4, we set the range threshold to be 40m. We report detection accuracies on near-range 0-40m and full-range 0-90m with and without adaptation, to demonstrate that the proposed framework can improve the far-range detection accuracy without any compromise on the near-range performance. The qualitative results performed on the three networks with and without adaptation are presented in Table 1. We observe from the results that the proposed adaptation framework not only boosts the far-range performance as shown in Table 2, but also increase the near-range performance at the same time, thus we achieve a superior full-range detection accuracy in Table 1.

We further plot the AP-distance curves in different training stages in Figure 5. Our framework significantly accelerates the training speed in the early stage as shown in Figure 5(a), and helps the network converge with a better performance (30000 iterations) as shown in Figure 5(d). In the entire training procedure, the proposed framework consistently improves the performance in both the near-range and the far-range areas. We further present a comparison performed on SECOND to validate the performance growth in the far-range only area, the results are presented in Table 2.

4.3. Qualitative Results

Qualitative results are presented in Figure 6. We present paired samples that are generated by SECOND [**?**] with and without the proposed adaptation framework, respectively. We consistently observe that the proposed framework improves the quality in the far-range area, e.g., the false positives are reduced, and hard positive objects are detected. Meanwhile, the performance in the near-range area remain robust without any degradation. Note that the networks for presenting the paired examples are trained using exact the same initialization and mini-batches by fixing the random seed for a fair comparison.

4.4. Ablation Studies

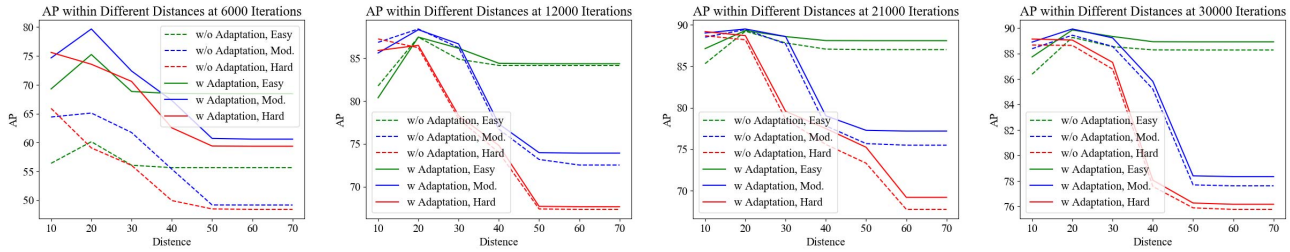
We further provide more experiment details and self-comparisons. All the experiments presented in the ablation studies are conducted on SECOND [**?**]. We select SECOND since it reports the best results among the three networks we adopt, and the experiments are implemented based on the publicly available source code online. Specifically, we validate the contributions of the proposed global and local adaptation by imposing each adaptation method individually on the training of SECOND, and compare the final performances. The results are presented in Table 3.

4.5. Cross-device Adaptation

The proposed framework on cross-range feature adaptation makes the first step on studying the adaptation in point clouds. Here we take one further step by showing the framework can be extended to, for example, a more challenging cross-device scenario. The experiment is conducted by introducing other datasets, i.e., nuScenes [**?**] 3D object detection dataset and our own dataset,[2] which are collected using different devices. As we see from Figure 4, our dataset has a significantly larger range, and adapting to it without heavily re-labeling is critical for its full exploitation.

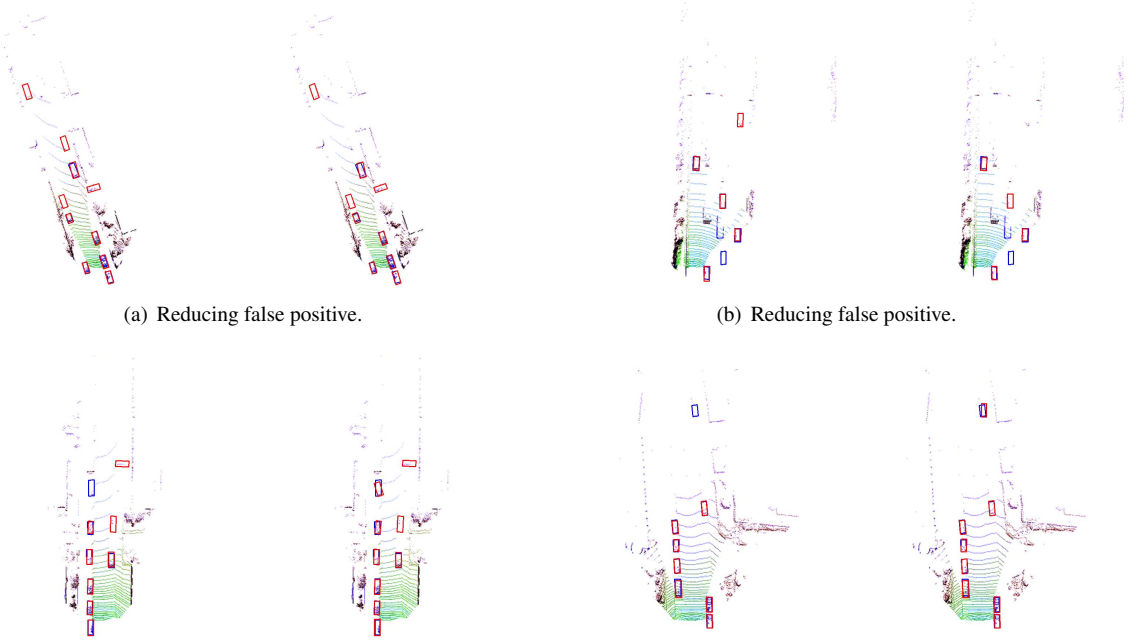
Sensor parameters for each dataset are presented in Table 4. Our dataset provides the densest point clouds with 128 channels, and farther range with an effective range of

[2]The dataset will be publicly available.



(a) Comparion at 6,000 iterations. (b) Comparion at 12,000 iterations. (c) Comparion at 21,000 iterations. (d) Comparion at 30,000 iterations.

Figure 5. 3D bbox AP at different iterations. The cures are obtained by training SECOND with and without the proposed adaptation framework. Training with adaptation gives a significant improvement on AP at the initial stage. It's clearly demonstrated that the proposed adaptation delivers not only a performance improvement on far-range objects, but also a improvement at the near-range objects.



(a) Reducing false positive. (b) Reducing false positive.

(c) Improving far-range recall. (d) Improving far-range recall.

Figure 6. Qualitative results (please zoom in for details). We present paired samples where in each pair, the left image is the result trained on the original network, while the right image is the result with the proposed adaptation framework. We use red and blue boxes to denote detections and ground truth boxes, respectively. It's clearly shown that the proposed framework increases the recall on far-range objects and reduces the false positive in the far-range area.

250m. The significant gap among the parameters of the above sensors makes them a set of perfect comparisons for performing cross-sensor experiments. The significant cross-device gap can be observed in Figure 7. Although nuScenes provides annotations on all directions, here we only considerate objects that can be seen in the front camera for a fair comparison with KITTI. There are totally 28,742 cars in the 7,481 frames in KITTI training set, and totally 8,426 cars in the 3,977 frames in nuScenes V0.1. In our dataset, there are totally 8,550 cars annotated in 898 frames. For both KITTI and nuScenes datasets, it is clearly shown in Figure 4 that the objects in about 50m have a dominant number and only a small number of objects are more than 70m away from the sensor. Our dataset has a clearly more average distribution, and a considerably large amount of objects in the far range (up to 250m).

We use only the initial released proportion of nuScenes V0.1 dataset which contains 3977 frames with annotations. We split the 3977 frame into a training set with the first 2000 frames, and a testing set with the rest 1977 frames. For our proposed new dataset, there are totally 898 frames in the initial release, and we use the first 600 frames for training, and the rest for testing. We use KITTI as the source domain, and conduct two experiments using nuScenes and our dataset as the target domain, respective. The proposed global and local adaptations are imposed to promote objects in the tar-

Methods	Near-range (0-40m)			Full-range (0-70m)		
	Easy	Moderate	Hard	Easy	Moderate	Hard
SECOND (w/o)	88.28	85.21	77.57	88.07	77.12	75.27
SECOND (w L)	88.29	85.41	77.58	88.23	77.88	75.74
SECOND (w G)	88.62	85.51	77.99	88.60	78.02	75.57
SECOND (w L+G)	**88.81**	**85.84**	**78.01**	**88.80**	**78.31**	**76.16**

Table 3. Ablation study. 3D bounding box average precision. L and G denote local adaptation and global adaptation respectively. The results are obtained by running five rounds of comparisons, where an identical random seed is used in each round for each experiment. The presented result is the average of the five round results. The proposed adaptation further improves the best detection model without introducing additional parameters or computation.

(a) Sample for KITTI. (b) Sample for nuScenes.

(c) Sample for our dataset.

Figure 7. A side-by-side comparison on the point density of samples from KITTI, nuScenes, and our dataset (please zoom in for details). We clearly observe that the nuScenes data has the lowest density, and our data has densest points and longest effective range.

Dataset	# of channels	Effective range
nuScenes	32	70m
KITTI	64	120m
Our dataset	64	250m

Table 4. Sensors that used for the collection of different dataset. High density and far effective range make our dataset a strong benchmark for evaluating cross-device adaptation.

get dataset to have consistent feature with objects in KITTI. The experiment is also conducted on the car ('vehicle.car' for nuScenes) class only. We compare our results with two baselines, training using target dataset only, and training both source and target datasets jointly without adaptation.

Method	AP
nuScenes only	36.2
Joint	44.3
Joint + Adaptation (w L+G)	**47.4**
Our data only	31.7
Joint	34.9
Joint + Adaptation (w L+G)	**37.7**

Table 5. Cross-device adaptation. Experiments are performed on two datasets. Joint training with proposed adaptation methods consistently improve the performance on target datasets.

We report 3D box AP across the entire range only with no subsets of different difficulties. The results are presented in Table 5. Joint training the samples from two datasets with the proposed adaptations delivers the best performance. We hypothesize that the small amount of samples in nuScenes is not enough for training a robust detector. Massive data in KITTI dataset and the promoted consistent features improve the robustness for the training on nuScenes and our dataset, so that higher performances are observed on the test sets.

5. Conclusion and Future Work

We proposed for the first time a model adaptation for object detection in 3D point clouds. Specifically, we addressed cross-range and cross-device adaptation using adversarial global adaptation and fine-grained local adaptation. We evaluated our adaptation method on various BEV-based object detection methods, and demonstrated that the combinations of the global and local adaptation can significantly improve model detection accuracy without adding any auxiliary parameters to the model. Beyond the range and device adaptations here studied, we will further investigate adaptations under other settings, e.g., adaptation across point clouds collected with different scanning patterns (i.e., Gaussian pattern and uniform pattern), and developing further adaptation methods that deliver better adaptation results with fewer or even no annotations on target domain.

Acknowledgments

Work partially supported by NSF, ARO, ONR, NGA, and gifts from AWS.