SRDAN: Scale-aware and Range-aware Domain Adaptation Network for Cross-dataset 3D Object Detection

Weichen Zhang[1] Wen Li[2] Dong Xu[1]
[1] School of Electrical and Information Engineering, The University of Sydney
[2] School of Computer Science and Engineering, University of Electronic Science and Technology of China
{weichen.zhang,dong.xu}@sydney.edu.au liwenbnu@gmail.com

Abstract

*Geometric characteristic plays an important role in the representation of an object in 3D point clouds. For example, large objects often contain more points, while small ones contain fewer points. The points from objects near the capture device are denser, while those from far-range objects are sparser. These issues bring new challenges to 3D object detection, especially under the domain adaptation scenarios. In this work, we propose a new cross-dataset 3D object detection method named Scale-aware and Range-aware Domain Adaptation Network (SRDAN). We take advantage of the geometric characteristics of 3D data (*i.e.*, size and distance), and propose the scale-aware domain alignment and the range-aware domain alignment strategies to guide the distribution alignment between two domains. For scale-aware domain alignment, we design a 3D voxel-based feature pyramid network to extract multi-scale semantic voxel features, and align the features and instances with similar scales between two domains. For range-aware domain alignment, we introduce a range-guided domain alignment module to align the features of objects according to their distance to the capture device. Extensive experiments under three different scenarios demonstrate the effectiveness of our SRDAN approach, and comprehensive ablation study also validates the importance of geometric characteristics for cross-dataset 3D object detection.*

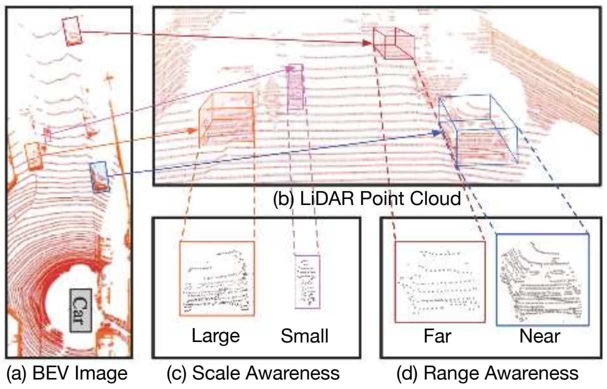

Figure 1. Motivation of the Scale-aware and Range-aware Domain Alignment strategies in our SRDAN method. (a) A Bird's-Eye-View (BEV) image is presented for better illustration. (b) A point cloud map collected by the LiDAR sensor (*i.e.*, the car with gray box in (a)) is presented to illustrate the 3D geometric characteristics. (c) At similar distances to the LiDAR sensor, large objects (*e.g.*, the car with orange box) often contain more points, while small ones (*e.g.*, the cyclist with purple box) contain fewer points. (d) The 3D points of objects (*e.g.*, the car with blue box) near the LiDAR sensor are denser, while those of far-range ones (*e.g.*, the car with red box) are sparser. We incorporate these geometric characteristics (*i.e.*, size and distance) into our method, and learn domain-invariant representation to ensure objects with similar geometric characteristics are well aligned between two domains.

1. Introduction

With the advance of autonomous driving, increasing attention has been attracted to 3D object detection [63, 56, 8, 27, 69, 78, 68, 29, 43, 30, 54, 71, 42, 53]. Although significant progress has been made, most of these works considered only the constrained setting, where the training data used to learn the detection model and the test data for performance evaluation are collected from a similar scenario. However, in practical applications, we often face the more challenging situation, where the training data and the test data captured by various devices at different time/places often exhibit considerable distribution mismatch. This could lead to significant performance drop when deploying the trained model in a new scenario, which is also known as the domain adaptation problem [19, 35, 36, 37, 13, 61, 74, 58, 3, 33, 25].

While many works have been proposed to address the domain adaptation problem for 2D images, little attention was paid to the cross-dataset object detection task for 3D point clouds. Different from the traditional cross-dataset object detection task for 2D images [9, 49, 26, 80, 66], the texture information is absent in 3D point cloud data, which makes it more challenging to deal with the data distribution mismatch issue. Consequently, the intrinsic geometric information becomes especially important when addressing the domain adaptation issue for the 3D object detection task.

However, it is non-trivial to capture intrinsic geometric information. As shown in Fig. 1, the point cloud of an object can vary significantly when the geometric characteristics (*e.g.*, the object scale and the distance of an object to the capture device like LiDAR) changes. The large-scale objects often contain more points, while the small-scale ones contain fewer points. The point clouds of objects near the capture device are denser, while those of far-range objects are sparser. Such geometric variance brings challenges to the 3D object detection task, especially in the domain adaptation scenarios where objects from the source and target domains with different geometric characteristics may be incorrectly aligned, leading to poor object detection results.

In this work, we propose a new cross-dataset 3D object detection method named Scale-aware and Range-aware Domain Adaptation Network (SRDAN). To deal with the issues related to geometric variance when performing distribution alignment, we propose to learn domain-invariant representation by incorporating the object geometric characteristic into deep neural networks. Ideally, with a good domain-invariant representation, the objects with similar geometric characteristics (*e.g.*, size and distance) should be well aligned between two domains. In other words, the large objects (*resp.*, the small objects) from the source domain should be aligned to the large ones (*resp.*, the small ones) from the target domain. The same criterion should also be applied to the objects in a similar range around the sensor.

To achieve this goal, in our SRDAN, we propose the scale-aware domain alignment and the range-aware domain alignment approach to guide distribution alignment between the two domains. For scale-aware domain alignment, we design a 3D voxel-based feature pyramid network, which can extract multiple feature maps at different scales for detecting the objects with different sizes. Then, we align the features and instances with similar scales between the two domains at each feature map. For range-aware domain alignment, we take advantage of range information in 3D data and further enhance domain alignment at both anchor-level and feature-level. For anchor-level alignment, we introduce a range-guided domain alignment (RLA) module to align the local features of objects with different sparsity levels. For feature-level alignment, we introduce a Location-related Global Alignment (LGA) module to learn a global weight map for finer alignment, which can guide our model to focus more on aligning the foreground objects.

Extensive experiments on four datasets (*i.e.*, Nuscenes [4], A*3D [39], PreSIL [22] and KITTI [14]) under three autonomous driving scenarios (*i.e.*, cross-scene adaptation, day-to-night adaptation and synthetic-to-real adaptation) clearly demonstrate the effectiveness of our approach for the cross-dataset 3D object detection task.

2. Related Work

2.1. 3D Object Detection

With the emergence of 3D data and high-performance computing resources, 3D object detection attracted increasing attention in many recent works, which can be roughly divided into point-based methods, voxel-based methods and fusion-based methods. The point-based methods [43, 54, 71, 42, 6, 55] directly process the unordered point clouds and extract the features with PointNet/PointNet++ [44, 45]. Different sampling and grouping strategies are usually applied in these methods to reduce memory and computational costs. On the other hand, the voxel-based methods [63, 69, 78, 68, 29, 28, 72, 7, 64] rasterize the 3D point clouds into the fix-sized voxel grids and then use the handcrafted or learnt voxel representation to further extract semantic features for 3D object detection. Finally, the fusion-based methods [56, 8, 27, 30, 53, 41, 62, 73, 11] generally learn 3D features by aggregating different types of data representations (*e.g.*, RGB, depth, points, etc.).

Our work builds upon the voxel-based method SECOND [68]. Different from the general 3D detection methods, our method aims at improving the generalization capability of the learnt model on the target domain for the cross-dataset 3D object detection task.

2.2. Domain Adaptation

In recent years, domain adaptation has been widely investigated for various computer vision tasks [18, 2, 9, 49, 46, 32] (*e.g.*, recognition, detection, segmentation). Since deep learning has achieved promising results in different areas, recently several deep domain adaptation methods have been proposed based on the convolutional neural networks (CNNs), which can be roughly categorized as the statistic-based approaches and the adversarial learning-based approaches. The statistic-based approaches [38, 58, 5, 37, 35, 36, 40, 67, 59] usually employed the statistic-based metrics to model the differences between two domains. On the other hand, the adversarial learning-based approaches [60, 12, 13, 3, 33, 61, 52, 20, 48, 75, 76, 50, 47, 77, 16, 10, 21, 70, 57] used Generative Adversarial Networks (GANs) [15] to learn domain-invariant representation, which can be explained by minimizing the $\mathcal{H}$-divergence [1] or the Jensen-Shannon divergence [17] between two domains. Most of these domain adaptation approaches are designed for the general 2D image recognition tasks, while direct adoption of these 2D techniques for the large-scale 3D object detection task may not work well due to the distinct characteristics of 3D data, especially point clouds. The works in [51, 65] projected the point clouds into 2D bird's-eye-view images or front-view images for further processing. However, the original 3D structural information of point clouds cannot be fully exploited. While the works in [46, 24] also extracted 3D domain-invariant representation, these methods [46, 24] are not specifically designed for the 3D object detection task.

To the best of our knowledge, our method is the first domain adaptation method that takes advantage of the voxel-based features for the 3D object detection task by effectively exploiting the characteristics of point cloud data.

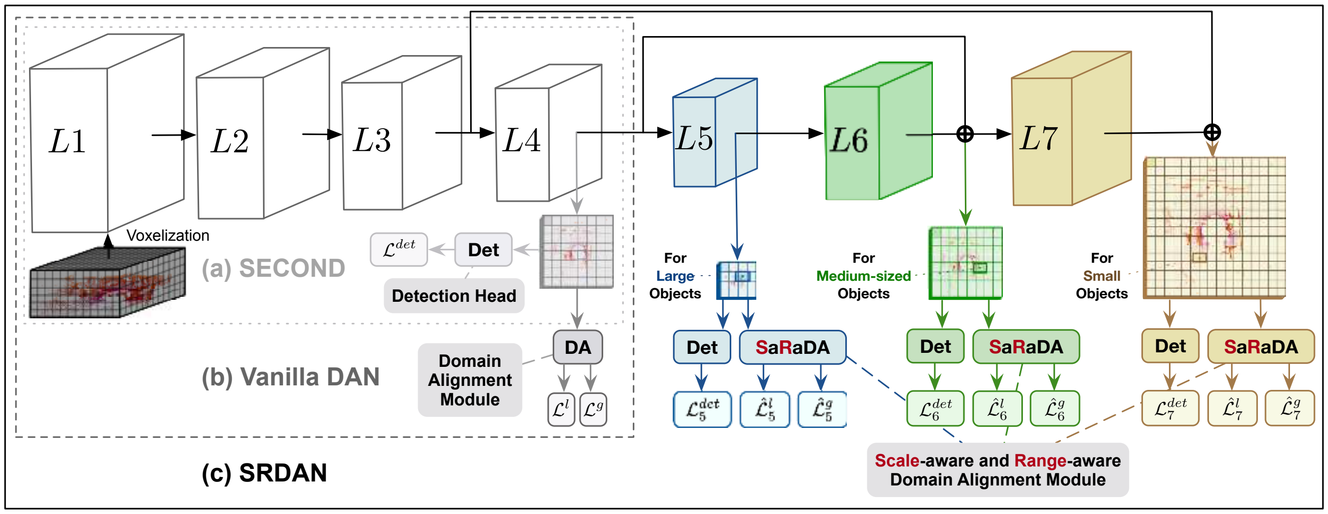

Figure 2. Overview of our SRDAN method for cross-dataset 3D object detection. (a) Our method builds upon the 3D Sparse Voxel CNN method SECOND [68], which voxelizes the point cloud data to extract the semantic features from both source and target domains. (b) We introduce a simple domain adaptive 3D object detection approach vanilla DAN, which uses the adversarial learning strategy to jointly align the features from two domains both locally at the anchor feature level and globally at the overall feature level. (c) To address the issue of objects with different sizes, we employ a 3D voxel-based feature pyramid network with three more CNN blocks and concatenate the two output features of SECOND from CNN block 3 ($L3$) and block 4 ($L4$) to the features of the feature pyramid network from block 7 ($L7$) and block 6 ($L6$) with the same feature map sizes to further enrich the representations at different scales. After that, at each scale (*i.e.*, as shown in Blue, Green and Yellow colors), we propose a newly designed Scale-aware and Range-aware Domain Alignment (SaRaDA) module to independently align the features from the two domains with similar 3D geometric characteristics (*i.e.*, size and distance). The detailed structure of our SaRaDA module is illustrated in Fig. 3. Finally, we add a detection head to generate the proposals at each scale based on source domain supervision and aggregate the proposals from all scales to produce the final detection results for the target samples.

3. Methodology

3.1. Problem Statement

In this work, we consider the 3D object detection under the domain adaptation scenario. In this scenario, we are given a set of labeled source samples and a set of unlabeled target samples. The source and target samples often have different data distributions. For example, in autonomous driving, the data collected in day-time can be used to help learn a model that is robust for the night-time data, without consuming time to manually annotate the night-time data. In this example, the day-time samples are used as the source domain data and the night time samples are used as the target domain data.

Formally, let us denote a point cloud as $\mathbf{X} = \{\mathbf{p}_k|_{k=1}^{n_k}\}$, which is a set of unordered points, where $\mathbf{p}_k$ is a point containing the 3D coordinate values, and n_k is the total number of points in $\mathbf{X}$. Under the domain adaptation scenario, we denote the source domain as $\mathcal{X}^s = \{(\mathbf{X}_i^s, \mathcal{Y}_i^s)|_{i=1}^{N^s}\}$, where N^s is the total number of source samples (*i.e.*, point clouds), $\mathbf{X}_i^s$ is a point cloud in the source domain, and $\mathcal{Y}_i^s$ is its annotation containing the bounding box locations and categories of all the objects in the i-th point cloud. Similarly, we are also given a set of unlabeled target samples $\mathcal{X}^t = \{\mathbf{X}_i^t|_{i=1}^{N^t}\}$, where N^t is the total number of target samples and $\mathbf{X}_i^t$ is the i-th point cloud in the target domain without any annotation. The goal of our task is to learn a robust detector, which can accurately predict the bounding box locations and categories for all the instances in the unlabeled target samples.

3.2. A Vanilla Method

We first introduce a simple domain adaptive 3D object detection method based on which we will establish our SRDAN. We refer to this method as the *vanilla domain adaptive method*, or vanilla DAN in short. Our vanilla DAN builds upon SECOND [68], which is a voxel-based 3D object detection method. As shown in Fig 2, in SECOND, a point cloud is first voxelized as a regular grid in the 3D space. Then, a few 3D convolutional layers are stacked to extract the semantic representation for the voxelized point cloud. After that, an SSD-like [34] detection head is used for detecting the objects in the point cloud.

Let us denote the 3D convolutional layers for extracting the voxelized feature as $F(\mathbf{X}; \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ denotes the parameters of the feature extraction network. Then, the detection head can be represented as $P(F(\mathbf{X}; \boldsymbol{\theta}); \boldsymbol{\beta})$ where $\boldsymbol{\beta}$ is the network parameters of the detection head. During training, the detection head outputs both bounding box coordinates and the predicted object categories in the point cloud $\mathbf{X}$, and we use the ground-truth annotation $\mathcal{Y}$ to supervise the training process. In our problem, we only have source domain supervision, so the detection loss is written as,

$$\mathcal{L}^{det} = \frac{1}{N^s}\sum_{i=1}^{N^s}\mathcal{L}_{sec}(P(F(\mathbf{X}_i^s;\boldsymbol{\theta});\boldsymbol{\beta}),\mathcal{Y}_i^s), \quad (1)$$

where $\mathcal{L}_{sec}$ is the detection loss from the SECOND detector, which contains a smooth-L1 regression loss for bounding box regression, and a class-balanced focal loss for classifying the objects. Please refer to [68] for more details.

However, under the domain adaptation scenario, due to data distribution mismatch, 3D detection performance may be degraded in the target domain by only using source domain supervision to train the model. To address this issue, as inspired by the recent domain adaptation works [13, 9], we employ the adversarial training strategy to align the feature distributions between two domains. In particular, a global domain alignment strategy and a local domain alignment strategy are used, which are described below.

Global Domain Alignment: Similar to [13], we apply a global average pooling layer and a domain classifier with two fully connected layers to align the overall features from the two domains. Let us denote $D^g(\cdot;\mathbf{w}^g)$ as the global domain classifier with $\mathbf{w}^g$ as its parameters. Then, the objective for global domain alignment can be written as,

$$\max_{\boldsymbol{\theta}}\min_{\mathbf{w}^g}\mathcal{L}^g = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_d(D^g(F(\mathbf{X}_i;\boldsymbol{\theta});\mathbf{w}^g),d_i), \quad (2)$$

where $d_i \in \{0,1\}$ is the domain label with 0 (*resp.*, 1) indicating $\mathbf{X}$ is from the source (*resp.*, the target) domain, $N = N^s + N^t$ is the total number of source and target samples, and $\mathcal{L}_d$ is the binary cross-entropy loss. As shown in [13], the above max-min optimization problem can be solved by simply introducing a gradient reversal layer.

Local Domain Alignment: Besides global alignment, we also propose a local domain alignment module to align the distribution at finer level. As in the SECOND method, the SSD-like detection head performs prediction based on the anchors at each location of the feature map, we thus apply a patch-based domain classifier similar to [23] to align the anchor-level feature distributions. Let us denote $F(\mathbf{X};\boldsymbol{\theta})^{(u,v)}$ as the anchor feature vector located at the position (u,v) of the feature map, and $D^l(\cdot;\mathbf{w}^l)$ as the patch-based domain classifier with $\mathbf{w}^l$ being its corresponding parameters, then the objective for anchor-level local domain alignment can be written as,

$$\max_{\boldsymbol{\theta}}\min_{\mathbf{w}^l}\mathcal{L}^l = \frac{1}{N}\sum_{i=1}^{N}\sum_{u,v}\mathcal{L}_d(D^l(F(\mathbf{X}_i;\boldsymbol{\theta})^{(u,v)};\mathbf{w}^l),d_i), \quad (3)$$

where d_i and $\mathcal{L}_d$ are similarly defined as in Eq. (2).

By summing up Eq. (1), (2) and (3), the overall objective for our vanilla DAN can be written as,

$$\min_{\boldsymbol{\theta}}\left\{\min_{\boldsymbol{\beta}}\mathcal{L}^{det} - \lambda\min_{\mathbf{w}^g,\mathbf{w}^l}\left(\mathcal{L}^l+\mathcal{L}^g\right)\right\}, \quad (4)$$

where λ is the trade-off parameter which is set to 0.1. Note, for ease of presentation, we change the maximization problem (*w.r.t.* $\boldsymbol{\theta}$) in the outer problem of Eq. (2) or (3) to a minimization problem (*w.r.t.* $\boldsymbol{\theta}$) in Eq. (4) by reversing the sign of the objective of the inner problem.

3.3. Scale-aware Domain Alignment

Now we consider how to use the geometric characteristics to guide feature distribution alignment in our vanilla DAN method. As discussed in Section 1, the object size and the distance to the LiDAR sensor are important factors that influence the point cloud representation.

Regarding the object size, in the ideal case, we expect the small objects in one domain to be well aligned with the small ones in another domain, and the same case for large objects. To achieve this, as inspired by the 2D image-based Feature Pyramid Network (FPN) [31], we design a 3D voxel-based feature pyramid network to extract multi-scale features. With multi-scale features, the anchors on the low-resolution feature map tend to predict large objects, while the anchors on the high-resolution feature map tend to predict small objects. Therefore, if we individually perform feature distribution alignment based on multiple feature maps at different scales, the object scales will be roughly considered when performing domain distribution alignment. We refer to this strategy as *Scale-aware Domain Alignment*.

As shown in Fig. 2, we add three additional CNN blocks[1] (represented in blue, green and yellow colors, respectively) after the original feature extraction network F. The feature map of the firstly introduced CNN block (denoted as $L5$) is half size of the last CNN block (denoted as $L4$) in F to capture larger objects. The following block (denoted as $L6$) shares the same size of $L4$ to capture similar object size as the original SECOND model, and the size of the last block is doubled to capture smaller objects. Then, we use the same domain alignment strategy on each of these three CNN blocks as in our vanilla DAN, respectively. Let us denote the losses for the detection head, the global domain alignment module, and the local domain alignment module from these blocks as $\mathcal{L}_c^{det}$, $\mathcal{L}_c^l$ and $\mathcal{L}_c^g$ with $c = 5, 6, 7$, respectively. The objective in Eq. (4) can be updated as follows,

$$\min_{\boldsymbol{\Theta}}\sum_c\left\{\min_{\boldsymbol{\beta}_c}\mathcal{L}_c^{det} - \lambda\min_{\mathbf{w}_c^g,\mathbf{w}_c^l}\left(\mathcal{L}_c^l+\mathcal{L}_c^g\right)\right\}, \quad (5)$$

where $\boldsymbol{\beta}_c$, $\mathbf{w}_c^g$ and $\mathbf{w}_c^l$ are the parameters for the detection head, the global domain alignment module, and the local domain alignment module at the c-th block defined similarly as in Eq. (1), (2) and (3), and $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_c|_{c=5}^{7}\}$ with $\boldsymbol{\theta}_c$ being the parameters for the corresponding feature extraction network consisting of all 3D convolutional layers before the c-th CNN block (inclusively).

We want to highlight that although our SRDAN employs a similar design for the backbone network as in FPN, the

[1] In our work, each CNN block consists of several sparse CNN layers.

purposes are intrinsically different. In [31], FPN is used to extract multi-scale features for boosting the detection performance across scales, while in our work we take advantage of FPN to align the feature distributions across domains based on their scale-aware representation capability.

3.4. Range-aware Domain Alignment

When extracting the point cloud representation, another issue is the distance to the capture device. For example, for the 3D point cloud captured by the widely used LiDAR sensor in the autonomous driving scenario, the 3D points near the LiDAR sensor can be quite dense, while those from far-range objects are sparser. However, we can hardly achieve results performance by simply using the same strategy for both near-range objects and far-range objects from different domains.

To this end, we design two modules, the Range-guided Local Alignmnet (RLA) module (as shown in the upper part of Fig. 3) and the Location-related Global Alignment (LGA) module (as shown in the lower part of Fig. 3) to effectively exploit range information and further strengthen domain alignment. The proposed range-aware domain alignment strategy is used to strengthen scale-aware domain alignment, so these two modules will be used on each of the three feature maps from $L5$, $L6$ and $L7$ as described in Section 3.3, which forms our overall Scale-aware and Range-aware Domain Alignment (SaRaDA) module. The details of the RLA and LGA modules are described below.

Range-guided Local Alignment (RLA): Since the sparsity levels of the 3D points from objects are correlated to their distances to the LiDAR sensor, we therefore propose to use the distance information to guide local domain alignment, namely, the objects from two domains with similar sparsity levels or distances to the LiDAR sensor should be aligned together.

To achieve this goal, as shown in the upper part of Fig 3, for the c-th CNN block where $c = 5, 6, 7$, we construct a 2-dimensional range guidance map $\mathbf{M}_c^l$, which has the same size as the feature map at the c-th block. Each element of $\mathbf{M}_c^l$ is its 2D coordinates[2], which represents the relative location to the LiDAR sensor (*i.e.*, the center of $\mathbf{M}_c^l$). In this way, the locations with similar distances to the LiDAR sensor will be assigned with the same coordinates.

After that, we concatenate the range guidance map to the original feature map along the channel dimension, and feed the concatenated feature into the local alignment module. Intuitively, it is much easier for the domain classifier to distinguish the features from two domains with different location information. However, for the features with similar relative locations, we still need to exploit more useful information to better distinguish them when learning the domain classifier. In other words, the domain classifier will focus

[2]Note, the feature maps along each z-axis in SECOND are converted to a scalar in each channel, so we use 2D coordinates.

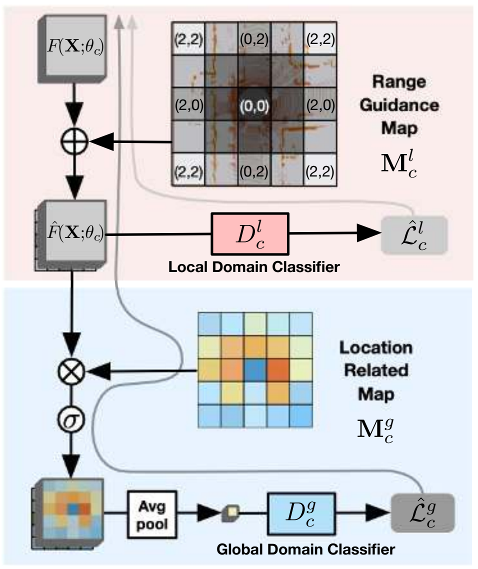

Figure 3. Overview of our Scale-aware and Range-aware Domain Alignment (SaRaDA) module. We take the SaRaDA module at the c-th CNN block as an example for better illustration. At each scale, our SaRaDA module takes the feature from the feature pyramid network (*e.g.*, $F(\mathbf{X}, \theta_c)$) as the input and jointly uses a Range-guided Local Alignment (RLA) module (*i.e.*, the upper part of Fig. 3) and a Location-related Global Alignment (LGA) module (*i.e.*, the lower part of Fig. 3) to respectively align the local and global features from different domains. The light gray and dark gray lines correspond to the back-propagation processes of the local and global domain alignment modules, respectively.

on aligning the features with the same relative location between the two domains.

Recall the original feature map in the 3D voxel-based FPN at the c-th block can be denoted as $F(\mathbf{X}; \boldsymbol{\theta}_c)$, then the concatenated feature can be represented as $\hat{F}(\mathbf{X}; \boldsymbol{\theta}_c) = F(\mathbf{X}; \boldsymbol{\theta}_c) \oplus \mathbf{M}_c^l$, where $\oplus$ is the concatenation operator along the channel dimension. Then, the local domain alignment loss at the c-th CNN block can be updated as,

$$\hat{\mathcal{L}}_c^l = \frac{1}{N}\sum_{i=1}^{N}\sum_{u,v}\mathcal{L}_d(D_c^l(\hat{F}(\mathbf{X}_i; \boldsymbol{\theta}_c)^{(u,v)}; \hat{\mathbf{w}}_c^l), d_i), \tag{6}$$

where D_c^l and $\hat{\mathbf{w}}_c^l$ are respectively the local domain classifier and its corresponding parameters at the c-th CNN block and both of them are similarly defined as in Eq. (3).

Location-related Global Alignment (LGA): We also use range information to enhance global domain alignment. For this purpose, we learn one location-related map to learn global statistics over the whole point clouds and re-weight the features from all patches for better global alignment of features from different domains. Specifically, based on the locally domain-aligned features from the RLA module at each CNN block, we learn one weight map (*i.e.*, the multi-colored map in the lower part of Fig. 3) for each domain adaptation task to learn the global adaptation weights for the features at different locations. Then, at each patch location, we multiply the features by the corresponding weight in the location-related map, so that the features from different patches make different contributions to ensure the domain invariance of the entire feature.

Let us denote the location-related map at the c-th block as $\mathbf{M}_c^g$, then the global domain alignment loss is updated as,

$$\hat{\mathcal{L}}_c^g = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_d(D_c^g(\hat{F}(\mathbf{X}_i;\boldsymbol{\theta}_c)\otimes \mathbf{M}_c^g;\mathbf{w}_c^g), d_i), \tag{7}$$

where $\otimes$ denotes the matrix multiplication operation between $\mathbf{M}_c^g$ and each channel of $\hat{F}(\mathbf{X}_i;\boldsymbol{\theta}_c)$, D_c^g and $\mathbf{w}_c^g$ are respectively the global domain classifier and its parameters at the c-th CNN block defined similarly as in Eq. (2). In this way, after performing the patch-wise feature re-weighting process, different importance values are learnt for different locations, which lead to better alignment of the global features from different domains.

3.5. Network Overview

By considering scale-aware and range-aware domain alignment strategies simultaneously, we arrive at the objective of our Scale-aware and Range-aware domain adaptation network (SRDAN). We use both range-aware local domain alignment and global domain alignment strategies as in Eq. (6) and (7), and update the objective in Eq. (5) for scale-aware domain alignment. The overall objective of our SRDAN can be rewritten as follows,

$$\min_{\boldsymbol{\Theta},\mathcal{M}^g}\sum_c\left\{\min_{\boldsymbol{\beta}_c}\mathcal{L}_c^{det} - \lambda \min_{\hat{\mathbf{w}}_c^g,\hat{\mathbf{w}}_c^l}\left(\hat{\mathcal{L}}_c^l+\hat{\mathcal{L}}_c^g\right)\right\}, \tag{8}$$

where $\mathcal{M}^g = \{\mathbf{M}_c^g|_{c=5}^7\}$, λ is defined in Eq. (4), $\mathcal{L}_c^{det}$ is defined in Eq. (5), and $\hat{\mathcal{L}}_c^l$ and $\hat{\mathcal{L}}_c^g$ are defined in Eq. (6) and (7).

4. Experiments

We evaluate our proposed method SRDAN under three cross-domain scenarios: **1) cross-scene adaptation**, where the data from the source and target domains are captured in different cities; **2) day-to-night adaptation**, where the data in the source and target domains are collected at day-time and night-time, respectively; **3) synthetic-to-real adaptation**, where the source domain data is captured from video games, while the target domain data comes from the real world. In addition, we also conduct ablation study to validate the effectiveness of each component in our SRDAN.

4.1. Experiment Setup

We follow the standard unsupervised domain adaptation protocol [35, 12, 13] for all methods, where the source training data contains both point clouds and their labels (including bounding box locations and object categories) and the training data from the target domain are unlabeled point clouds. The results of all methods are evaluated on the target domain, where the mean average precision (mAP) is reported with the IoU thresholds of 0.7 for cars and 0.5 for all

Boston → Singapore	**Car**	**Pedestrian**	**Barrier**
PointPillars [29]	70.9	65.6	13.6
PointRCNN [54]	71.2	66.7	14.1
SECOND [68]	71.4	67.2	14.4
SWDA-3D [49]	72.4	69.1	14.6
vanilla DAN (Ours)	71.9	68.1	14.9
SRDAN(Ours)	**74.6**	**71.8**	**16.7**
Singapore → Boston	**Car**	**Pedestrian**	**Barrier**
PointPillars [29]	67.6	66.4	12.8
PointRCNN [54]	68.8	68.3	13.0
SECOND [68]	69.2	68.9	13.1
SWDA-3D [49]	70.3	69.9	13.8
vanilla DAN (Ours)	69.9	69.8	13.7
SRDAN(Ours)	**72.5**	**72.7**	**16.4**

Table 1. 3D object detection results for the cross-scene scenario.

other classes in all scenarios. We use the OpenPCDet[3] toolbox to implement our SRDAN under the PyTorch[4] framework. SECOND [68] is used as our backbone network for feature extraction. We use Adam as the optimizer and set the initial learning rate as 0.003. Other settings (*e.g.*, the detection range) are the same as in [68].

4.2. Experimental Results

4.2.1 Cross-Scene Adaptation

Datasets: For the cross-scene adaptation scenario, we use the two subsets from the challenging dataset Nuscenes [4] captured in Boston and Singapore, respectively. We treat one city as the source domain and the other as the target domain, and vice versa. The Boston (*resp.*, Singapore) domain contains 467 (*resp.*, 383) driving scenes with 18,785 (*resp.*, 15,364) samples. We use the three most common classes (*i.e.*, Car, Pedestrian and Barrier) to conduct the experiments.

Baselines: We report the results of SECOND [68] and two state-of-the-art 3D object detection methods PointRCNN [29] and PointPillars [29] trained based on the source domain data, in which their released code is directly used. PointRCNN [29] is a two-stage detection method, which generates the proposals from the PointNet encoders [44, 45]. PointPillars [29] encodes the point features based on vertical columns (pillars) for more efficiency. In addition, we also report the results of the extended version (denoted as SWDA-3D) of SWDA [49], for which we replace the 2D feature extractor in SWDA [49] with SECOND[68]. We also include the vanilla version of our SRDAN (*i.e.*, *vanilla DAN*) for comparison.

Results: The results of the different methods for both tasks under the cross-scene scenario are summarized in Table 1. Without considering the domain adaptation issue, the baseline approach SECOND and the supervised method PointRCNN achieve similar results, while the PointPillars method

[3]https://github.com/open-mmlab/OpenPCDet
[4]https://github.com/pytorch/pytorch

Day → Night	Car			Pedestrian			Cyclist		
	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
PointPillars [29]	72.1	68.5	65.7	35.9	33.5	32.0	32.6	29.0	25.9
PointRCNN [54]	74.7	71.6	68.1	38.6	37.7	32.8	31.5	27.9	25.6
SECOND [68]	76.3	74.5	72.4	39.7	38.3	33.9	31.9	29.7	26.8
SWDA-3D [49]	78.7	76.2	73.9	44.0	41.4	36.8	35.2	33.4	29.5
vanilla DAN (Ours)	77.9	75.3	73.1	42.4	40.3	36.1	32.8	31.3	27.9
SRDAN(Ours)	**80.8**	**78.4**	**76.2**	**47.5**	**43.2**	**39.8**	**40.3**	**37.2**	**33.6**

Table 2. 3D object detection results for the day-to-night scenario.

PreSIL → KITTI	Car			Pedestrian			Cyclist		
	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
AVOD-FPN [27]	-	-	-	14.1	11.9	11.6	-	-	-
PointPillars [29]	16.3	15.0	14.7	9.9	9.0	8.9	4.5	4.3	4.3
PointRCNN [54]	-	15.7	-	-	9.6	-	-	5.6	-
SECOND [68]	16.7	15.1	14.9	10.9	10.1	9.8	4.6	4.5	4.5
DABEV [51]	-	17.1	-	-	10.3	-	-	5.9	-
CDN [57]	-	19.0	-	-	13.2	-	-	9.1	-
SWDA-3D [49]	22.6	18.7	16.3	12.8	11.5	10.3	7.6	7.1	6.9
vanilla DAN (Ours)	19.7	16.8	15.6	11.3	10.4	9.9	6.3	6.1	6.0
SRDAN(Ours)	**25.9**	**22.1**	**18.7**	**15.9**	**14.6**	**12.5**	**9.6**	**9.4**	**9.1**

Table 3. 3D object detection results for the synthetic-to-real scenario. '-' indicates the results are not available in their works.

is slightly worse. SWDA-3D and our vanilla DAN outperform SECOND, which show the necessity for dealing with data distribution mismatch for 3D object detection under the cross-scene scenario. However, the improvements of both methods over SECOND are relatively limited, which indicates we cannot effectively align the data distributions of two domains by simply applying the domain adversarial training strategy for 3D object detection, possibly due to the geometric variance of 3D point clouds from two domains. By exploiting the scale-aware alignment and range-aware alignment strategies, our SRDAN method gains notable improvements over SECOND and achieves the best results among all methods, which clearly demonstrates the effectiveness of our approach for guiding distribution alignment with geometric characteristics.

4.2.2 Day-to-Night Adaptation

Datasets: Under the day-to-night adaptation scenario, we investigate the influence of different lighting conditions for 3D object detection by using the day-time and night-time subsets from the A*3D dataset [39]. A*3D dataset contains around 40,000 point cloud frames with heavy occlusions. We use the day/night split file provided in the downloaded dataset to construct the two domains in our experiments, which include 4060 and 1147 high-density point cloud samples for the Day subset and the Night subset, respectively. Note, the available day/night samples in the downloaded dataset are different from those reported in their paper [39]. As a result, we use the available day-time data for the 3D object detection task in the night time (*i.e.*, the "Day → Night" task), in which we utilize the three most common classes (*i.e.*, Car, Pedestrian and Cyclist) in our experiments.
Results: Similar to the cross-scene scenario, we compare our method with the baseline method SECOND [68], two state-of-the-art methods PointRCNN [54] and PointPillars [29], and the domain adaptation method SWDA-3D [49]. The 3D object detection results of different methods under the day-to-night scenario are reported in Table 2. We have similar observations as those from the cross-scene scenario. Without dealing with the domain adaptation issue, the existing approaches SECOND, PointRCNN and PointPillars cannot achieve satisfactory performance. Our vanilla DAN and SWDA-3D [49] gain limited improvements over the baseline method SECOND. Our SRDAN method outperforms all existing methods as well as our vanilla DAN method with considerable gains, which again validates the effectiveness of our SRDAN approach.

4.2.3 Synthetic-to-Real Adaptation

Datasets: Learning from synthetic data has attracted more and more attention in recent years, mainly due to the requirement of large scale labeled training data for developing deep learning methods, and the difficulty in collecting and annotating such large-scale data in the real-world scenarios. We also evaluate the effectiveness of our proposed SRDAN method for 3D object detection by using the synthetic data as the source domain, and the real data as the target domain. In particular, we use the PreSIL dataset [22] as the source domain, which contains 51075 synthetic LiDAR data generated from the Grand Theft Auto V (GTA V) game. The well-known 3D object detection benchmark dataset KITTI [14] is used as the target domain, which contains 7,481 unlabeled samples. This leads to the "PreSIL → KITTI" task under the synthetic-to-real scenario. We utilize the three most common classes (*i.e.*, Car, Pedestrian and Cyclist) that appeared in both datasets to conduct the experiments.
Results: Besides the baseline methods discussed in the previous two scenarios, we also compare our work with one general 3D object detection method (*i.e.*, AVOD-FPN [27]) and two domain adaptation methods (*i.e.*, DABEV [51] and CDN [57]) as their results are reported for this scenario. AVOD-FPN is a multi-view image-based detection method, which aggregates the features from the RGB images, the projected Bird's-Eye-View images and 3D anchor grids into a 2D Feature Pyramid Network. DABEV uses CycleGAN [79] to transfer the style of the projected Bird-Eye-View images and performs detection based on the style-transferred 2D images. The results of DABEV [51] and CDN [57] are reported in [57], while the results of AVOD-FPN are reported in [22].

The results of different methods are reported in Table 3. For the baseline methods and our vanilla DAN method, we have similar observations as those in the previous scenarios. We also observe the two domain adaptation methods DABEV and CDN achieve better results than the baseline

Methods	SECOND	+FPN	+SaDA	+RLA	+LGA	+RaDA	SRDAN (Ours)
B → S	71.4	72.6	73.3	72.4	72.9	73.1	**74.6**

Table 4. Ablation study of our SRDAN for the B → S task (Car).

methods, which shows it is beneficial to deal with the domain distribution mismatch issue under the synthetic-to-real scenario. However, these two methods fail to consider the geometric characteristics for domain alignment. Our SRDAN outperforms all methods including those two domain adaptation methods with a large margin. The results clearly demonstrate the effectiveness of our method by exploiting the geometric characteristics for cross-domain 3D object detection.

4.2.4 Ablation Study

To validate the effectiveness of our proposed scale-aware domain alignment and range-aware domain alignment strategies, we further conduct comprehensive ablation study by varying different components in our SRDAN.

In particular, we take the "Car" category from the Boston → Singapore (B → S) task as an example, and investigate the scale-aware domain alignment and range-aware domain alignment strategies individually. The method with '+' sign indicates the variant of our SRDAN by additionally applying each newly introduced component of our SRDAN on the baseline method SECOND.

Scale-aware domain alignment: To validate the effectiveness of our scale-aware domain alignment strategy, we temporally remove all components (*i.e.*, the RLA and LGA modules) in our range-aware domain alignment strategy, and report the results of SECOND with the 3D voxel-based FPN (referred to as "+*FPN*"), and SECOND with the scale-aware domain alignment strategy (referred to as "+*SaDA*").

As shown in Table 4, while "+FPN" achieves better results than SECOND (72.6% vs. 71.4%) by using the FPN approach, our "+SaDA" further boosts the result to 73.3% by aligning the distributions between the two domains with the aid of scale information, which clearly validates the effectiveness of our scale-aware domain alignment strategy.

Range-aware domain alignment: We further validate the effectiveness of our range-aware domain alignment strategy. In this ablation study, we temporally remove the components (*i.e.*, 3D voxel-based FPN) related to the scale-aware domain alignment strategy, and directly apply the range-aware domain alignment strategy on our vanilla DAN method (referred to as "+*RaDA*"). Since our range-aware domain alignment strategy consists of a Range-guided Local Alignment (RLA) module and a Location-related Global Alignment (LGA) module, we also report the results by applying two modules individually in combination with our vanilla DAN model, which are referred to as "+*RLA*" and "+*LGA*", respectively.

From the results in Table 4, the mAP results can be improved to 72.4% (*resp.*,72.9%) after adding the individual RLA module (*resp.*, LGA module) to the backbone network SECOND, and the results can be further improved to 73.1% after using both RLA and LGA modules (*i.e.*, "+*RaDA*"). The results clearly indicate the effectiveness of both modules in our RaDA approach. Finally, by combining both SaDA and RaDA strategies in our SRDAN, we can achieve the overall mAP of 74.6%, which significantly improves the mAP by 3.5% from the baseline method SECOND. The results show both SaDA and RaDA strategies are effective, which can also be well combined to further boost the results for the cross-dataset 3D object detection task.

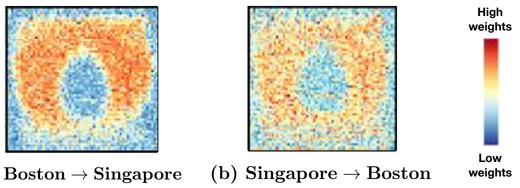

Figure 4. Visualization of the learnt location-related map from our LGA module.

4.3. Analysis for the LGA module

Based on the observation that the sparsity levels of 3D points from objects may be different at different spatial locations and the labeled objects usually appear on certain locations in the point cloud maps (*i.e.*, the labeled objects are rarely far away from or overlap with the driving car), for each detection task, we use the Location-related Global Alignment (LGA) module to learn one attention map. The learnt attention map is used as the importance score map and then multiplied to the original feature map to enhance the domain invariance of foreground objects. We take the LGA module at block 5 of CNN as an example to visualize the learnt attention map in Fig. 4. The results clearly verify the aforementioned assumption.

5. Conclusion

In this work, we have proposed a new unsupervised domain adaptation method named Scale-aware and Range-aware Domain Adaptation Network (SRDAN) for the 3D object detection task. Based on the sparse voxel CNN method SECOND, we employ a voxel-based feature pyramid network to extract multi-scale features to effectively handle objects with different sizes. Then, we introduce the scale-aware and range-aware domain alignment strategies to address the data distribution mismatch issue for better domain adaptation, in which we exploit the distinct size and distance information in 3D representations. Extensive experiments under three scenarios together with comprehensive ablation study and analysis demonstrate the effectiveness of our approach for cross-dataset 3D object detection.

Acknowledgement: Dong Xu is partially supported by the Australian Research Council (ARC) Future Fellowship under Grant FT180100116. Wen Li is supported by the Major Project for New Generation of AI under Grant No. 2018AAA0100400.

References

[1] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1):151–175, 2010.

[2] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *CVPR*, pages 3722–3731, 2017.

[3] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *NeurIPS*, pages 343–351, 2016.

[4] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, pages 11621–11631, 2020.

[5] Fabio Maria Carlucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulò. Autodial: Automatic domain alignment layers. In *ICCV*, pages 5077–5085, 2017.

[6] Jintai Chen, Biwen Lei, Qingyu Song, Haochao Ying, Danny Z Chen, and Jian Wu. A hierarchical graph network for 3d object detection on point clouds. In *CVPR*, pages 392–401, 2020.

[7] Qi Chen, Lin Sun, Zhixin Wang, Kui Jia, and Alan Yuille. Object as hotspots: An anchor-free 3d object detection approach via firing of hotspots. In *ECCV*. Springer, 2020.

[8] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *CVPR*, pages 1907–1915, 2017.

[9] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *CVPR*, pages 3339–3348, 2018.

[10] Shuhao Cui, Shuhui Wang, Junbao Zhuo, Chi Su, Qingming Huang, and Qi Tian. Gradually vanishing bridge for adversarial domain adaptation. In *CVPR*, pages 12455–12464, 2020.

[11] Jianfeng Feng, Errui Ding, and Shilei Wen. Monocular 3d object detection via feature domain adaptation. In *ECCV*. Springer, 2020.

[12] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, pages 1180–1189, 2015.

[13] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 17(59):1–35, 2016.

[14] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, pages 3354–3361. IEEE, 2012.

[15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, pages 2672–2680, 2014.

[16] Xiang Gu, Jian Sun, and Zongben Xu. Spherical space domain adaptation with robust pseudo-label loss. In *CVPR*, June 2020.

[17] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *NeurIPS*, pages 5767–5777, 2017.

[18] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *ICML*, 2018.

[19] Junlin Hu, Jiwen Lu, and Yap-Peng Tan. Deep transfer metric learning. In *CVPR*, pages 325–333, 2015.

[20] Lanqing Hu, Meina Kan, Shiguang Shan, and Xilin Chen. Duplex generative adversarial network for unsupervised domain adaptation. In *CVPR*, pages 1498–1507, 2018.

[21] Lanqing Hu, Meina Kan, Shiguang Shan, and Xilin Chen. Unsupervised domain adaptation with hierarchical gradient synchronization. In *CVPR*, pages 4043–4052, 2020.

[22] Braden Hurl, Krzysztof Czarnecki, and Steven Waslander. Precise synthetic image and lidar (presil) dataset for autonomous vehicle perception. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 2522–2529. IEEE, 2019.

[23] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, pages 1125–1134, 2017.

[24] Maximilian Jaritz, Tuan-Hung Vu, Raoul de Charette, Emilie Wirbel, and Patrick Pérez. xmuda: Cross-modal unsupervised domain adaptation for 3d semantic segmentation. In *CVPR*, pages 12605–12614, 2020.

[25] Guoliang Kang, Liang Zheng, Yan Yan, and Yi Yang. Deep adversarial attention alignment for unsupervised domain adaptation: the benefit of target expectation maximization. In *ECCV*, 2018.

[26] Seunghyeon Kim, Jaehoon Choi, Taekyung Kim, and Changick Kim. Self-training and adversarial background regularization for unsupervised domain adaptive one-stage object detection. In *ICCV*, pages 6092–6101, 2019.

[27] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *IROS*, pages 1–8. IEEE, 2018.

[28] Hongwu Kuang, Bei Wang, Jianping An, Ming Zhang, and Zehan Zhang. Voxel-fpn: Multi-scale voxel feature aggregation for 3d object detection from lidar point clouds. *Sensors*, 20:704, 01 2020.

[29] 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*, pages 12697–12705, 2019.

[30] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *ECCV*, pages 641–656, 2018.

[31] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017.

[32] Dongnan Liu, Donghao Zhang, Yang Song, Fan Zhang, Lauren O'Donnell, Heng Huang, Mei Chen, and Weidong Cai. Unsupervised instance segmentation in microscopy images via panoptic domain adaptation and task re-weighting. In *CVPR*, pages 4243–4252, 2020.

[33] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In *NeurIPS*, pages 469–477, 2016.

[34] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, pages 21–37. Springer, 2016.

[35] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *ICML*, pages 97–105, 2015.

[36] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. In *NeurIPS*, pages 136–144, 2016.

[37] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *ICML*, pages 2208–2217, 2017.

[38] Massimiliano Mancini, Lorenzo Porzi, Samuel Rota Bulò, Barbara Caputo, and Elisa Ricci. Boosting domain adaptation by discovering latent domains. In *CVPR*, 2018.

[39] Quang-Hieu Pham, Pierre Sevestre, Ramanpreet Singh Pahwa, Huijing Zhan, Chun Ho Pang, Yuda Chen, Armin Mustafa, Vijay Chandrasekhar, and Jie Lin. A* 3d dataset: Towards autonomous driving in challenging environments. In *ICRA*, pages 2267–2273. IEEE, 2020.

[40] Pedro O Pinheiro and AI Element. Unsupervised domain adaptation with similarity learning. In *CVPR*, 2018.

[41] Charles R Qi, Xinlei Chen, Or Litany, and Leonidas J Guibas. Imvotenet: Boosting 3d object detection in point clouds with image votes. In *CVPR*, pages 4404–4413, 2020.

[42] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3d object detection in point clouds. In *ICCV*, pages 9277–9286, 2019.

[43] 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*, pages 918–927, 2018.

[44] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 652–660, 2017.

[45] 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*, pages 5099–5108, 2017.

[46] Can Qin, Haoxuan You, Lichen Wang, C-C Jay Kuo, and Yun Fu. Pointdan: A multi-scale 3d domain adaption network for point cloud representation. In *NeurIPS*, pages 7192–7203, 2019.

[47] Artem Rozantsev, Mathieu Salzmann, and Pascal Fua. Beyond sharing weights for deep domain adaptation. *PAMI*, 2018.

[48] Paolo Russo, Fabio M Carlucci, Tatiana Tommasi, and Barbara Caputo. From source to target and back: symmetric bi-directional adaptive gan. In *CVPR*, 2018.

[49] Kuniaki Saito, Yoshitaka Ushiku, Tatsuya Harada, and Kate Saenko. Strong-weak distribution alignment for adaptive object detection. In *CVPR*, pages 6956–6965, 2019.

[50] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, 2018.

[51] Khaled Saleh, Ahmed Abobakr, Mohammed Attia, Julie Iskander, Darius Nahavandi, Mohammed Hossny, and Saeid Nahvandi. Domain adaptation for vehicle detection from bird's eye view lidar point cloud data. In *ICCV-W*, pages 0–0, 2019.

[52] Swami Sankaranarayanan, Yogesh Balaji, Carlos D Castillo, and Rama Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. In *CVPR*, 2018.

[53] 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*, pages 10529–10538, 2020.

[54] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*, pages 770–779, 2019.

[55] Weijing Shi and Raj Rajkumar. Point-gnn: Graph neural network for 3d object detection in a point cloud. In *CVPR*, June 2020.

[56] Shiyu Song and Manmohan Chandraker. Joint sfm and detection cues for monocular 3d localization in road scenes. In *CVPR*, pages 3734–3742, 2015.

[57] Peng Su, Kun Wang, Xingyu Zeng, Shixiang Tang, Dapeng Chen, Di Qiu, and Xiaogang Wang. Adapting object detectors with conditional domain normalization. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XI*, volume 12356 of *Lecture Notes in Computer Science*, pages 403–419. Springer, 2020.

[58] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *ECCV*, pages 443–450, 2016.

[59] Hui Tang, Ke Chen, and Kui Jia. Unsupervised domain adaptation via structurally regularized deep clustering. In *CVPR*, pages 8725–8735, 2020.

[60] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *ICCV*, pages 4068–4076. IEEE, 2015.

[61] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, pages 2962–2971, 2017.

[62] Sourabh Vora, Alex H Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection. In *CVPR*, pages 4604–4612, 2020.

[63] Dominic Zeng Wang and Ingmar Posner. Voting for voting in online point cloud object detection. In *Robotics: Science and Systems*, pages 10–15607, 2015.

[64] Jun Wang, Shiyi Lan, Mingfei Gao, and Larry S Davis. Infofocus: 3d object detection for autonomous driving with dynamic information modeling. In *ECCV*. Springer, 2020.

[65] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue, and Kurt Keutzer. Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud. In *ICRA*, pages 4376–4382. IEEE, 2019.

[66] Chang-Dong Xu, Xing-Ran Zhao, Xin Jin, and Xiu-Shen Wei. Exploring categorical regularization for domain adaptive object detection. In *CVPR*, pages 11724–11733, 2020.

[67] Renjun Xu, Pelen Liu, Liyan Wang, Chao Chen, and Jindong Wang. Reliable weighted optimal transport for unsupervised domain adaptation. In *CVPR*, June 2020.

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

[69] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *CVPR*, pages 7652–7660, 2018.

[70] Jianfei Yang, Han Zou, Yuxun Zhou, Zhaoyang Zeng, and Lihua Xie. Mind the discriminability: Asymmetric adversarial domain adaptation. In *ECCV*. Springer, 2020.

[71] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Std: Sparse-to-dense 3d object detector for point cloud. In *ICCV*, pages 1951–1960, 2019.

[72] Maosheng Ye, Shuangjie Xu, and Tongyi Cao. Hvnet: Hybrid voxel network for lidar based 3d object detection. In *CVPR*, pages 1631–1640, 2020.

[73] Jin Hyeok Yoo, Yeocheol 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. In *ECCV*. Springer, 2020.

[74] Xingyu Zeng, Wanli Ouyang, Meng Wang, and Xiaogang Wang. Deep learning of scene-specific classifier for pedestrian detection. In *ECCV*, pages 472–487. Springer, 2014.

[75] Weichen Zhang, Wanli Ouyang, Wen Li, and Dong Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *CVPR*, pages 3801–3809, 2018.

[76] Weichen Zhang, Dong Xu, Wanli Ouyang, and Wen Li. Self-paced collaborative and adversarial network for unsupervised domain adaptation. *T-PAMI*, 2019.

[77] Yabin Zhang, Hui Tang, Kui Jia, and Mingkui Tan. Domain-symmetric networks for adversarial domain adaptation. In *CVPR*, pages 5031–5040, 2019.

[78] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, pages 4490–4499, 2018.

[79] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *CVPR*, pages 2223–2232, 2017.

[80] Xinge Zhu, Jiangmiao Pang, Ceyuan Yang, Jianping Shi, and Dahua Lin. Adapting object detectors via selective cross-domain alignment. In *CVPR*, pages 687–696, 2019.