Adversarially Masking Synthetic to Mimic Real: Adaptive Noise Injection for Point Cloud Segmentation Adaptation

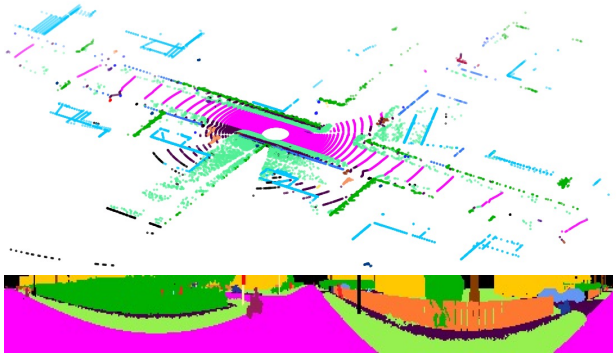
Guangrui Li[1,2†], Guoliang Kang*[3], Xiaohan Wang[4], Yunchao Wei[5], Yi Yang[4]
[1] ReLER, AAII, University of Technology Sydney, [2]Baidu
[3] Beihang University, [4] Zhejiang University, [5] Beijing Jiaotong University.

Abstract

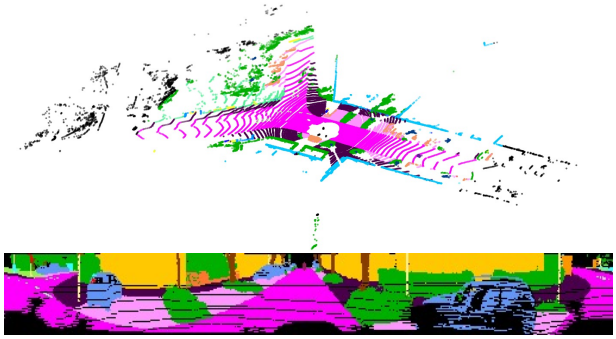
This paper considers the synthetic-to-real adaptation of point cloud semantic segmentation, which aims to segment the real-world point clouds with only synthetic labels available. Contrary to synthetic data which is integral and clean, point clouds collected by real-world sensors typically contain unexpected and irregular noise because the sensors may be impacted by various environmental conditions. Consequently, the model trained on ideal synthetic data may fail to achieve satisfactory segmentation results on real data. Influenced by such noise, previous adversarial training methods, which are conventional for 2D adaptation tasks, become less effective. In this paper, we aim to mitigate the domain gap caused by target noise via learning to mask the source points during the adaptation procedure. To this end, we design a novel learnable masking module, which takes source features and 3D coordinates as inputs. We incorporate Gumbel-Softmax operation into the masking module so that it can generate binary masks and be trained end-to-end via gradient back-propagation. With the help of adversarial training, the masking module can learn to generate source masks to mimic the pattern of irregular target noise, thereby narrowing the domain gap. We name our method "Adversarial Masking" as adversarial training and learnable masking module depend on each other and cooperate with each other to mitigate the domain gap. Experiments on two synthetic-to-real adaptation benchmarks verify the effectiveness of the proposed method.

1. Introduction

Recently, point cloud semantic segmentation task attracts increasing attention because of its important role in various real-world applications, *e.g.*, autonomous driving, augmented reality, *etc*. Despite remarkable progress [5, 17, 26,32,34,56,57], most algorithms are designed for the fully-supervised setting, where massive annotated data is available. In the real world, it is costly and time-consuming to annotate large amounts of data, especially for labeling each point in the segmentation task. Synthetic data is easy to obtain and its label can be automatically generated, which largely reduces the human effort of annotat-



LiDAR scan from a synthetic dataset (SynLiDAR)



LiDAR scan collected from real world (SemKITTI)

Figure 1. Comparison between a synthetic LiDAR scan (upper) and a real scan (lower). Both original point clouds and projected LiDAR images are given. Black points denote noise and other colors denote points from different classes. Compared with synthetic data which is integral and clean, point clouds collected from the real world typically contain unexpected and irregular noise which may impede the adaptation.

* Corresponding author.

ing data. However, it is usually infeasible to directly apply networks trained on synthetic data to real-world data due to the apparent domain gap between them. In this paper, we consider the synthetic-to-real domain adaptation [7, 8, 8, 31, 33, 36, 39, 50] for point cloud segmentation. Specifically, we aim to utilize the fully-annotated synthetic point clouds (source domain) and unlabeled point clouds collected from imperfect real-world sensors (target domain) to train a network to support the segmentation of real-world point clouds (target domain).

Domain adaptation solutions aim to discover and mitigate the domain shift from source to target domain. Through comparing the synthetic and real-world point clouds, we observe that the domain shift can be largely attributed to the unexpected and irregular noise existing in the target domain data. As with [47], we consider "noise" to be the missing points of certain instances/objects, where all pixel channels are zero. Such noise may be caused by various factors such as non-reflective surfaces (*e.g.*, glass). As shown in Fig. 1, the synthetic point cloud is integral and clean, but the real one contains large amounts of noisy points. A model trained on clean source data may find it hard to understand the scene context under the distraction of noises and thus cannot achieve satisfactory segmentation results on target point clouds.

Previous domain adaptation methods [4, 11, 12, 16, 18, 19, 25, 31, 55] (*e.g.*, adversarial training), which have been proven effective in the 2D visual tasks, can be applied to this 3D segmentation setting. For example, SqueezeSegV2 [48] employs geodesic correlation alignment [30] to align the point-wise feature distributions of two domains. However, without explicitly modeling and dealing with the noise, these methods bring quite weak benefits to the adaptation performance. Recently, several works attempt to deal with the target noise to mitigate the domain gap. Rochan *et al*. [37] randomly select target noise masks and apply the selected mask to source samples. Wu *et al*. [47] compute one dataset-level mask and apply it to all source samples. Zhao *et al*. [61] use CycleGAN [62] to perform noise inpainting which is then used to learn synthetic noise generation module. The issues of these previous works are two-fold: 1) they cannot adaptively determine the injected noises according to the context of source samples; 2) the generated mask cannot be guaranteed to reduce the domain shift. Thus, these methods may achieve sub-optimal results.

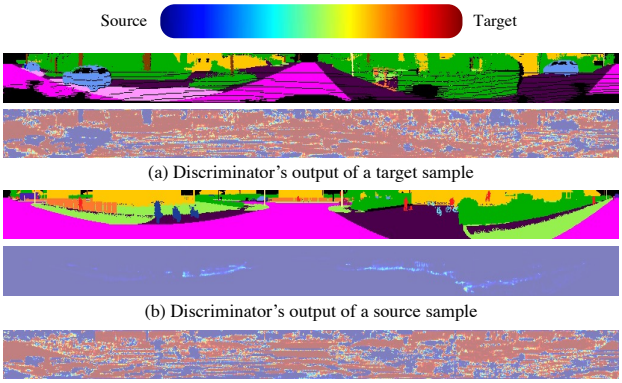
In this paper, we aim to mitigate the domain shift caused by the target noise by learning to adaptively mask the source points during the adaptation procedure. To reach this goal, we need to deal with two problems: 1) how to learn a spatial mask that can be adaptively determined according to the specific context of a source sample, and 2) how to guarantee the learned masks help narrow the domain gap. To solve the first problem, we design a learnable masking module named "Adaptive Spatial Masking (ASM)" module, which takes source Cartesian coordinates and features as input, to generate point-wise source masks. We incorporate Gumbel-Softmax operation into the masking module so that it can generate binary masks and be trained end-to-end via gradient back-propagation. To solve the second problem, we incorporate adversarial training into the masking module learning process. Specifically, during training, we add an additional domain discriminator on top of the feature extractor. By encouraging features from two domains (features of masked source samples and those of normal target samples) to be indistinguishable, the masking module is able to learn to generate masks mimicking the pattern of target noise and narrow the domain gap. Note that these two designs cooperate with each other to better align features across domains and improve the adaptation performance.

In a nutshell, our contributions can be summarized as:

- We notice that the pattern of target noise is unexpected and irregular. Thus, we propose to model the target noise in a learnable way. Previous works, which don't explicitly model the target noise or ignore such characteristics, are less effective.
- We propose to adversarially mask source samples to mimic the target noise patterns. In detail, we design a novel learnable masking module and incorporate adversarial training. Both components cooperate with each other to promote the adaptation.
- Experiments on two synthetic-to-real adaptation benchmarks, *i.e.* SynLiDAR $\rightarrow$ SemKITTI and SynLiDAR $\rightarrow$ nuScenes, demonstrate that our method can effectively improve the adaptation performance.

2. Related Works

Point Cloud Semantic Segmentation aims to classify the point clouds into predefined semantic categories in a point-wise manner. Previous researches in this area could be categorized into three streams: 1) *point-based methods* propose to handle this task in a point-wise manner and aggregate the contextual information through MLP (Multiple Layer Perception) [9, 34, 35], GCN (Graph Convolutional Network) [46, 54], or newly designed convolutions [42, 49, 53, 60]. These methods typically require massive computation, making them hard to satisfy the latency constraint in real-world applications. 2) *Voxel-based methods* [24] try to convert point clouds into 3D voxels and employ 3D convolutions to learn the geometric distributions. Some researchers [59, 63] study the partition strategy in the 3D space, while some researchers [14, 41] propose new convolution architectures to handle the sparse 3D voxels. Also, the heavy computation cost of 3D CNN hampers their applicability to real-world applications. 3) *projection-based solution* is another routine focusing on transforming 3D point



(a) Discriminator's output of a target sample

(b) Discriminator's output of a source sample

(c) Discriminator's output of the noise-injected source sample

Figure 2. Heatmap of discriminator's output, where color red indicates the target domain and blue denotes the source. As shown in (a) and (b), source samples and target ones are well identified by the domain discriminator. However, in (c), injecting noise extracted from a random target sample to the source sample can easily fool the discriminator.

clouds into 2D grids so that 2D convolutions can be utilized directly. Various architectures [6, 29, 47, 48, 52] in this direction have been proposed to cope with the projected 2D images, and have been proven to be effective and efficient. Moreover, projection-based methods also receive increasing attention on other tasks [27, 40] for their low computation cost. In this paper, we choose the projection-based architecture to perform our adaptation task as they strike a better balance between performance and efficiency.

Domain Adaptation for Semantic Segmentation aims to adapt a segmentation model from a labeled domain to an unlabeled domain. Research here can be coarsely grouped into three categories, *i.e.*, 2D, 3D, and cross-modal (*e.g.*, [21]). Here we only consider the single-modality scenario.

The 2D adaptation methods focus on the adaptation with appearance features from images and have received wide attention recently. There are several ways to deal with this problem. One way is to align the features to minimize the domain gap [23, 43, 45]. For example, AdaptSeg [43] employs adversarial training to align the outputs from two domains. Instead, ADVENT [45] uses adversarial training to align the entropy of outputs across domains. PLCA [23] proposes a different perspective, which associates cross-domain pixels via cycle consistency and encourages the similarities of associated pixels to mitigate the domain gap. Another way [64, 65] tries to overcome the distribution shift with self-training, which assigns pseudo labels to target samples with high confidence. There are also some works [58] combining both of these two solutions to form a multi-stage training. However, without explicitly considering the specific domain gap existing in point clouds, these methods may bring quite weak benefits to the adaptation.

There are several works dealing with the 3D segmentation adaptation scenario [37,38,47,49]. SqueezeSegV2 [48] fills the missing intensity channel for the synthetic point cloud and adopts geodesic correlation alignment [30] to perform point-wise feature alignment. As SynLiDAR [51] provides the intensity information, we directly utilize the intensity information along with coordinate information to train our model in this paper. Wu *et al*. [47] notice the domain shift is largely caused by the target noise, and propose to impose noise masks on source samples, where the mask is denoted as the point-wise frequency of noise over the whole target dataset. Analogously, Rochan *et al*. [37] perform masking on source samples with a randomly selected noise mask from the target domain. However, the points from the same spatial positions of two point clouds may have different surroundings or contexts and thus they should not share the same probability to be noise. Zhao *et al*. [61] employ CycleGAN [62] to perform target noise inpainting which is then used to learn synthetic noise generation module. However, the noise generation module is trained by target inpainting results and thus may still be affected by domain shift. Moreover, different from ours, the noise generation of [61] cannot be optimized towards reducing domain shift. CoSMix [38] proposes an augmentation technique that mixes scans from two domains in a semantic-aware manner. Nevertheless, the distraction of noises cannot be alleviated with the mixing between two domains.

3. Methodology

In the following sections, we first provide necessary preliminaries (§ 3.1) for domain adaptive point cloud segmentation. Second, we introduce the Adversarial Masking method (§ 3.2). Finally, we give the training objectives for the adaptation process (§ 3.3).

3.1. Preliminaries

In domain adaptive point cloud segmentation, we are provided with annotated source scans $\mathcal{S} = \{(\boldsymbol{P}_i^s, \boldsymbol{M}_i^s)\}_{i=1}^{N^s}$ and unlabeled target scans $\mathcal{T} = \{(\boldsymbol{P}_i^t)\}_{i=1}^{N^t}$, where $\boldsymbol{P}_i \in \mathbb{R}^{n_i \times 4}$ denotes the set of points with coordinates (x, y, z) and intensity, $\boldsymbol{M}_i \in \mathbb{R}^{n_i}$ denotes the ground-truth annotation for the point cloud, and n_i is the number of points in the i-th scan. For more efficient processing, we employ spherical projection to transform each raw point cloud $\boldsymbol{P}$ into 2D image $\boldsymbol{I} \in \mathbb{R}^{H \times W \times 5}$, and the labels are transformed to $\boldsymbol{Y} \in \mathbb{R}^{H \times W}$ accordingly. The details are presented below. We aim to train a segmentation model on $\mathcal{S}$ and $\mathcal{T}$ to make accurate predictions on target points.

Spherical Projection. For more efficient processing, we transform the sparse point clouds into 2D images with spherical projection like [47, 48]. Specifically, for a point with coordinate (x, y, z), we project it into a 2D LiDAR

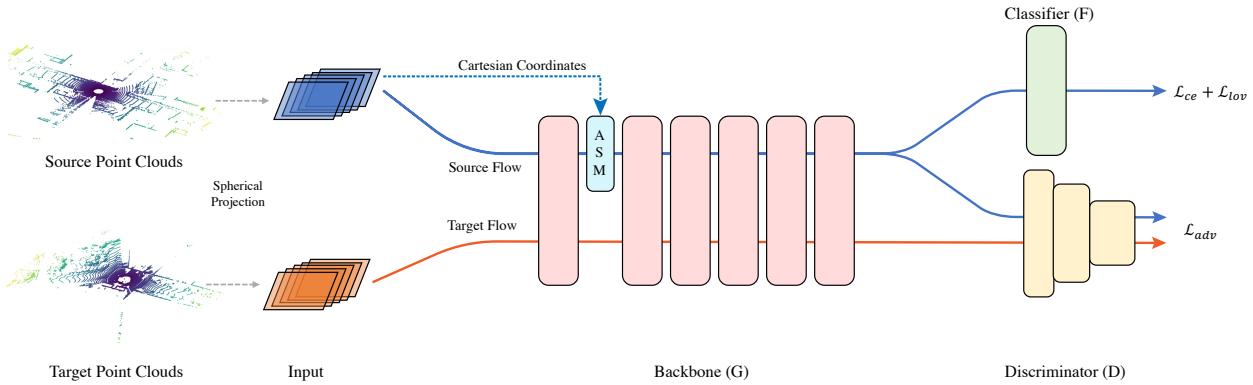


Figure 3. **Illustration of the Adversarial Masking Framework.** In this paper, we aim to mitigate the domain gap induced by target noise via masking source samples to mimic the target patterns. First, we propose a module, named Adaptive Spatial Masking (ASM), which can learn to mask the source points. Then we train the ASM-equipped model in an adversarial way. The two key components of our framework (*i.e.* ASM and adversarial training) collaboratively contribute to the final adaptation performance. Specifically, adversarial training encourages ASM to mimic target noises, while ASM eases the adversarial training to better align features across domains.

image with coordinates (p, q):

$$\begin{bmatrix} p \\ q \end{bmatrix} = \begin{bmatrix} \frac{1}{2}(1 - \arctan2^{2}(y, x)/\pi) \cdot W \\ (1 - (\arcsin(z \cdot r^{-1}) + f_{up}) \cdot f^{-1}) \cdot H \end{bmatrix}, \quad (1)$$

where $r = \sqrt{x^2 + y^2 + z^2}$ is the range of this point. $f = f_{up} + f_{down}$ is the vertical field-of-view of the LiDAR sensor. For each projected point with coordinate (p, q), we concatenate its Cartesian coordinates (x, y, z), range (r), and intensity, then obtain the projected LiDAR image $\boldsymbol{I}$ with the shape $H \times W \times 5$. The intensity channel models the strength of LiDAR beams. As such, raw point clouds with sparse and unordered structures are transformed into 2D images, so that 2D convolutions can be applied directly.

Domain Adversarial Training (DAT). Domain adversarial training [10, 15, 44] has been proven effective in aligning the feature distributions across domains. During training, an additional domain discriminator is introduced to classify the features into different domains. Through adversarial training, the network is encouraged to generate features that are indistinguishable across domains. Consequently, domain-invariant features can be learned, hence benefiting the adaptation performance.

Specifically, let D denotes the discriminator, the above min-max game can be formed as (the original GAN loss format [13]):

$$\begin{aligned} \min_{G} \max_{D} V_{GAN}(G, D) = \mathbb{E}_{I^t \sim \mathcal{T}}[\log(D(G(\boldsymbol{I}^t)))]) \\ + \mathbb{E}_{I^s \sim \mathcal{S}}[\log(1 - D(G(\boldsymbol{I}^s)))]), \end{aligned} \quad (2)$$

where G denotes the feature extractor of the model.

3.2. Adversarial Masking

Our framework is illustrated in Fig. 3. Following the general practice in domain adaptation, the network is shared across source and target domain data. The network consists of a backbone (G) to extract features and a task classifier (F) to distinguish the samples into different categories. During training, we insert our designed masking module (ASM) into the backbone to mask source points and attach an additional discriminator (D) on top of the backbone to assign domain labels to features from both domains. On the one hand, the domain discriminator is trained to differentiate masked source samples and target samples. On the other hand, the ASM is encouraged to learn to mask source points to mimic the target noise patterns, and features are trained to be domain-invariant to confuse the domain discriminator. As a result, the adversarial training and the masking module work collaboratively to narrow the domain discrepancy.

Target Noise Hinders Adversarial Training. As adversarial training has been proven effective in 2D visual domain adaptation tasks, *e.g.*, image classification, semantic segmentation, it is natural to see if adversarial training can help learn domain-invariant features in this 3D synthetic-to-real adaptation scenario. As shown in Fig. 2, we observe an interesting phenomenon that the discriminator converges quickly and can easily differentiate most of target points from the source ones. In contrast, injecting noise (from a random target sample) to source samples helps alleviate such an issue, *i.e.*, the features of many source points can confuse the domain discriminator in terms of their domain labels. We assume that in plain adversarial training, target noise may serve as a shortcut for the discriminator to classify samples into different domains. Only with adversarial training, it is hard to discover the noise patterns of the target and meanwhile align feature distributions across domains. Thus, we propose to explicitly model the noise patterns of the target and adaptively inject noise into source samples in order to ease the conventional adversarial training.

Adaptive Spatial Masking (ASM) Module. As dis-

[2]We use the arctan2 function in the Numpy library (www.numpy.org).

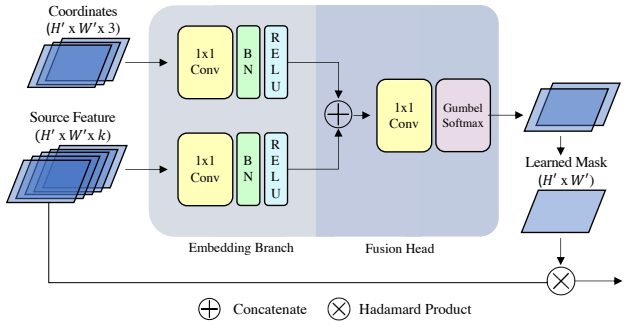


Figure 4. **Illustration of Adaptive Spatial Masking (ASM).** The proposed ASM takes source Cartesian coordinates and source features as input, and outputs two differentiable binary maps which divide points into two groups, *i.e.*, preserved and ignored. Then we impose the first mask on original source features.

cussed above, the way of randomly injecting target noise to source samples can alleviate the issue of adversarial training to an extent. However, such a way may not be an optimal choice because it cannot adaptively determine the distribution of injected noise according to the specific context of each source sample. Moreover, the pattern of injected noise may be irregular, and the copy-and-paste way of injecting noise may not accurately capture those irregular patterns.

Thus, we design a learnable module to perform spatial masking on source samples. We name our module as Adaptive Spatial Masking (ASM) module. To be concrete, we present the diagram of ASM in Fig. 4. The module consists of two embedding branches and one fusion head that all employ 1×1 convolutions. First, the two branches take source Cartesian coordinates $\boldsymbol{O}^s \in \mathbb{R}^{H'\times W'\times 3}$ (*i.e.* the x, y, z channels of projected LiDAR image but downsampled to $H'\times W'$ to match the size of feature map) and source features $\boldsymbol{E}^s \in \mathbb{R}^{H'\times W'\times k}$ as inputs respectively. Then, the embedded features from two branches are fused via a fusion head to generate the desired source mask. Finally, we calculate the element-wise product between the learned mask and original source features. Then the masked source feature is forwarded through the remaining layers for predictions.

Note that we attach a Gumbel-Softmax [20] layer to the end of the fusion head. The output of Gumbel-Softmax has two channels, each of which has spatial size $H' \times W'$. We use the first channel to indicate which points should be preserved and the second channel to indicate which points should be ignored. Then, the first channel is utilized to mask the source features with the element-wise product. Note that Gumbel-Softmax enables us to apply binary masks during the forward process, while supporting gradient back-propagation to update the parameters of our network. In contrast, the plain softmax layer cannot actually zero out source points, and thus leads to inferior results. We will show an empirical comparison of these designs in Sec. 4.3.

As shown in Fig. 3, we insert ASM (*i.e.*, denoted with color blue) after a specific shallow layer to inject noises to source samples. Note that we don't directly insert ASM after the input of the projected LiDAR image. It is because that we empirically find the shallow features are also useful to learn a better source mask, while the input only contains the coordinate information. In our implementation, we place ASM after the first convolution block (*i.e.*, `conv-bn-relu`). Note that ASM is only applied to source samples during the training process. So we simply remove ASM module for the inference on target samples.

Adversarial Masking. With the proposed masking module, the model can inject noise to source samples by zeroing out shallow features of partial source points. However, we cannot guarantee that the generated mask can mimic the patterns of target noise and thus mitigating the domain gap. To solve this, we integrate the ASM-equipped model with an adversarial training paradigm. Specifically, with ASM, the corresponding discrimination and generation loss of adversarial training are

$$\min_{\theta_D} \mathcal{L}_{dis} = \mathbb{E}_{I^t\sim\mathcal{T}}[(1-D(G(\boldsymbol{I}^t)))^2] + \mathbb{E}_{I^s\sim\mathcal{S}}[(D(\bar{G}(\boldsymbol{I}^s)))^2],$$
$$\min_{\theta_G,\theta_{ASM}} \mathcal{L}_{gen} = \mathbb{E}_{I^s\sim\mathcal{S}}[(1-D(\bar{G}(\boldsymbol{I}^s)))^2], \quad (3)$$

where $\bar{G}$ is the backbone with ASM module inserted and $\theta_{ASM}, \theta_G, \theta_D$ denote the parameters of ASM module, backbone and discriminator respectively. Note that different from the loss format in Eq. 2, we choose LSGAN [28] in our implementation for its better stability.

3.3. Training Objective

Overall, the model is optimized with three objectives, *i.e.*, cross-entropy loss ($\mathcal{L}_{ce}$), Lovasz-Softmax loss [2] ($\mathcal{L}_{lov}$), and the adversarial training loss ($\mathcal{L}_{gen}$, $\mathcal{L}_{dis}$):

$$\min_{\theta_G,\theta_{ASM},\theta_F} \mathcal{L} = \mathcal{L}_{ce} + \mathcal{L}_{lov} + \lambda\mathcal{L}_{gen} \quad (4)$$

$$\min_{\theta_D} \mathcal{L}_{dis} \quad (5)$$

where θ_F denotes the parameters of the classifier and the cross-entropy loss is calculated as:

$$\mathcal{L}_{ce} = -\sum_{h,w}^{H,W} \log[F(\bar{G}(I^s)(h,w,Y_{h,w}))]. \quad (6)$$

The segmentation model and the domain discriminator are updated alternatively with objective Eq. 4 and Eq. 5, respectively. Note that $Y_{h,w}$ is the label for the pixel at position (h, w) of a projected LiDAR image.

4. Experiments

4.1. Setup

Datasets. In this paper, we perform experiments on two synthetic-to-real benchmarks, *i.e.*, SynLiDAR→ SemKITTI and SynLiDAR→ nuScenes.

Table 1. Experiments results of SynLiDAR [51] → SemKITTI [1] with SqueezeSegV3-21 [52] as the backbone.

Methods	Type	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	road	parking	sidewalk	building	fence	vegetation	trunk	terrain	pole	traffic-sign	mIoU
Source Only	–	4.2	12.8	6.7	0.9	3.7	9.6	14.3	1.4	0.3	23.2	47.6	5.0	54.3	21.4	27.0	23.2	3.1	$15.2_{\pm 0.3}$
AdaptSeg [43]	2D	0.6	5.9	3.4	1.9	5.4	8.2	13.6	60.0	1.7	37.8	36.4	3.7	31.8	17.3	40.1	20.9	4.4	$17.2_{\pm 0.3}$
ADVENT [45]	2D	16.6	9.3	8.0	2.1	3.8	8.4	16.5	46.7	3.0	31.4	34.7	6.8	44.5	20.4	28.9	20.0	4.0	$17.9_{\pm 0.3}$
CBST [64]	2D	7.6	9.0	4.8	1.2	1.1	10.5	11.1	8.8	1.9	20.0	48.5	**8.9**	35.8	20.3	25.2	28.6	4.9	$14.6_{\pm 0.5}$
PLCA [23]	2D	7.8	7.1	4.0	3.1	3.8	**14.3**	18.1	44.4	**10.2**	19.8	44.9	4.6	43.9	12.6	21.2	26.3	3.6	$17.0_{\pm 0.3}$
SqueezeSegV1 [47]	3D	**43.7**	7.4	5.1	**4.6**	**5.8**	6.5	13.9	36.7	3.1	31.0	37.4	7.4	28.9	**26.2**	29.9	27.1	5.0	$17.4_{\pm 0.3}$
SqueezeSegV2 [48]	3D	20.4	8.0	4.0	2.4	5.1	8.0	17.3	59.4	3.2	34.9	39.4	8.4	41.3	21.9	32.2	29.0	4.7	$20.0_{\pm 0.4}$
ePointDA [61]	3D	18.8	8.0	6.1	2.6	3.6	6.4	14.6	50.8	9.4	32.7	33.3	8.2	27.6	22.4	30.2	26.9	**7.2**	$18.2_{\pm 0.6}$
LiDAR-Net [22]	3D	22.3	7.2	10.3	2.0	2.3	8.1	18.8	43.2	5.6	33.5	32.6	4.9	29.8	26.5	22.5	21.4	5.4	$17.4_{\pm 0.4}$
CoSMix [38]	3D	15.1	4.3	3.2	1.0	1.4	5.4	5.5	47.6	2.9	32.9	54.1	7.8	58.1	24.8	**41.0**	**32.1**	3.0	$20.0_{\pm 0.6}$
RandMask+ADV	3D	42.0	10.0	**13.0**	2.4	4.7	8.2	**20.7**	30.5	3.7	29.2	37.7	5.0	30.7	22.9	26.7	24.7	4.1	$18.6_{\pm 0.3}$
FreqMask+ADV	3D	22.9	9.3	5.3	2.2	3.0	5.4	13.9	50.6	3.8	31.9	38.1	6.5	32.0	25.4	38.1	25.5	4.6	$18.7_{\pm 0.3}$
SpatialDropout+ADV	3D	30.1	7.3	3.2	3.1	5.1	10.3	11.6	44.3	3.2	30.4	40.1	7.7	30.8	22.3	27.9	23.9	3.9	$18.0_{\pm 0.2}$
Ours	3D	19.7	**13.8**	9.7	2.1	4.1	8.0	8.2	**64.5**	8.0	**36.0**	**54.6**	6.7	**58.0**	24.7	35.8	29.1	4.2	$\mathbf{22.8}_{\pm 0.3}$
Oracle	–	91.9	25.5	42.0	42.7	26.1	32.9	54.7	94.2	42.8	82.0	80.8	39.9	84.5	48.9	72.2	54.0	28.8	$55.5_{\pm 0.2}$

SynLiDAR [51] is a synthetic LiDAR dataset for point cloud segmentation, which is collected from a simulated driving scene environment. This dataset collects 198,396 scans with 19482 M points, covering various scenes on Unreal Engine 4 platform. This dataset provides point-wise annotations for 32 classes that are in line with SemKITTI.

SemKITTI [1] is a large-scale point cloud dataset for point cloud segmentation. This dataset is collected from a Velodyne HDL-64E LiDAR and contains 22 sequences with 41000 frames. This dataset contains annotations for 25 categories. Following [52, 63], we choose sequences 00-10 for training except sequence 08 for validation.

nuScenes-lidarseg [3] is another LiDAR dataset that collected from real world. This dataset is collected from a different LiDAR sensor, *i.e.*, a 32-beam LiDAR with FOV of $[-30°, 10°]$. Following its guideline, 850 scenes are chosen for training and the other 150 scenes for validation.

For SynLiDAR → SemKITTI and SynLiDAR → nuScenes, part of the labels are merged to match across domains, the detailed mapping is provided in the appendix.

Evaluation. Following common practice [17, 53, 63], we adopt mean intersection over union (*i.e.*, mIoU) as the evaluation metric, which is averaged over all classes. Note that we report the averaged results over 3 random runs. No post-processing is applied, *e.g.*, conditional random field.

Implementation. For the segmentation task, we choose two representative backbones, *i.e.*, SqueezeSegV3-21 [52] and SalsaNext [6]. The ASM employs the Straight Through variant of Gumbel-Softmax [20]. The model is optimized using momentum SGD with momentum of 0.9 and weight decay 1×10^{-4}. Warmup is applied for the first epoch to linearly increase the learning rate to the base learning rate. Then learning rate decays exponentially. The base learning rate is set to 4×10^{-3} and 2×10^{-3} for SynLiDAR→SemKITTI and SynLiDAR→nuScenes, respectively. The discriminator is optimized using Adam optimizer with learning rate of 1×10^{-3}, and its architecture is presented in the Appendix. The batch size is set to 24 and the model is optimized for 50 epochs totally. The output channel of the embedding branch in ASM is set to 32. The λ in Eq. 4 is set to 0.001.

4.2. Comparisons with Previous Methods

We compare our method with previous 2D and 3D domain adaptation methods. The "2D" and "3D" indicate the settings that the methods are originally designed for. For 2D methods, we re-implement representative methods, *i.e.*, CBST [64], PLCA [23], AdaptSeg [43], and ADVENT [45]. As for 3D adaptation techniques, we re-implement SqueezeSegV1 [47] SqueezeSegV2 [48], ePointDA [61], LiDAR-Net [22], and CoSMix [38] with the identical backbone. Besides, we also present the results with three variants of source masks, *i.e.*, "SpatialDropout" that randomly drops the source points spatially, "RandMask" randomly selects masks from the target samples, and "FreqMask" where points are randomly dropped according to the point-wise frequency map of target noise over the dataset. We use "ADV" to denote the adversarial training paradigm and use "Oracle" to denote the full supervision baseline. For a fair comparison, all presented results use the same supervision on source samples, *i.e.*, $\mathcal{L}_{ce}+\mathcal{L}_{lov}$.

In Table 1 and Table 2, we present the results on SynLiDAR → SemKITTI and SynLiDAR → nuScenes, respectively. First, compared with the source-only baseline, our method achieves apparent improvements, *i.e.*, +7.6% mIoU absolute gain on SemKITTI and +5.5% mIoU on nuScenes, which justifies the necessity of performing adaptation. Second, compared with 2D techniques, our method still holds

Table 2. Experiments results of SynLiDAR [51] → nuScenes [3] with SalsaNext [6] as the backbone.

Methods	Type	bicyle	bus	car	other-vehicle	motorcycle	pedestrian	truck	road	other-ground	sidewalk	terrain	manmade	vegetation	mIoU
Source Only	–	0.4	1.0	3.7	0.2	2.1	16.0	17.0	24.2	0.1	14.9	8.6	36.0	25.5	$11.5_{\pm 0.3}$
AdaptSeg [43]	2D	0.7	0.5	13.2	0.7	2.2	17.0	13.8	44.3	0.4	18.3	10.6	39.2	**27.5**	$14.5_{\pm 0.3}$
ADVENT [45]	2D	0.5	1.6	11.4	0.7	2.1	18.0	18.3	43.2	0.7	19.2	10.3	40.2	27.2	$14.8_{\pm 0.3}$
CBST [64]	2D	0.4	0.2	8.2	0.4	1.3	5.5	13.4	38.2	0.7	15.0	10.4	26.0	30.8	$11.6_{\pm 0.5}$
PLCA [23]	2D	0.5	8.5	12.3	1.4	1.1	19.3	16.6	33.9	3.6	16.6	4.8	33.1	22.2	$13.4_{\pm 0.3}$
SqueezeSegV1 [47]	3D	0.7	5.7	19.7	0.8	2.9	17.9	**19.0**	33.8	0.3	14.9	**15.1**	26.5	23.5	$13.9_{\pm 0.4}$
SqueezeSegV2 [48]	3D	0.7	2.5	10.6	0.6	**4.1**	19.5	14.6	33.4	0.2	17.3	6.2	43.5	23.7	$13.6_{\pm 0.3}$
ePointDA [61]	3D	0.4	3.0	9.8	0.6	3.0	18.5	13.8	31.9	0.3	15.0	8.2	35.6	23.4	$12.6_{\pm 0.3}$
LiDAR-Net [22]	3D	0.6	4.0	10.1	0.8	2.7	18.8	13.3	34.1	0.5	14.9	8.8	38.7	20.1	$12.9_{\pm 0.3}$
CoSMix [38]	3D	0.3	2.6	0.5	0.4	0.6	7.1	1.7	**60.1**	**14.3**	11.9	8.6	33.4	18.1	$12.3_{\pm 0.4}$
RandMask+ADV	3D	0.5	6.0	17.4	1.6	2.6	18.9	15.0	33.1	0.5	15.7	8.0	38.9	27.1	$14.3_{\pm 0.3}$
FreqMask+ADV	3D	0.5	6.7	19.0	1.1	3.1	**22.3**	14.1	31.1	0.4	16.6	7.3	40.0	26.5	$14.5_{\pm 0.3}$
SpatialDropout+ADV	3D	0.5	**8.6**	19.9	1.6	1.7	13.9	16.5	50.7	3.6	16.9	8.9	30.6	20.3	$14.9_{\pm 0.3}$
Ours	3D	**0.9**	1.2	**26.9**	**2.2**	2.6	17.4	18.2	57.4	0.8	**21.8**	7.6	**43.9**	20.1	$\mathbf{17.0}_{\pm 0.3}$
Oracle	–	25.3	71.8	85.1	34.3	44.0	65.3	63.2	95.3	69.3	70.7	71.1	81.2	73.9	$65.4_{\pm 0.3}$

its superiority, *e.g.*, our method outperforms AdaptSeg by 5.6% and 2.5% mIoU on SynLiDAR → SemKITTI and SyncLiDAR → nuScenes respectively. Especially, we notice that CBST shows inferior performance, which may be because of the low quality of pseudo labels resulting from the large gap between source and target point clouds. Third, compared with 3D solutions, our method also attains superior results, *e.g.* on SemKITTI, we achieve 2.8% and 4.6% absolute gain compared to SqueezeSegV2 and ePointDA, respectively. Moreover, even with adversarial training, various non-learnable masking strategies (RandMask, SpatialDropout, FreqMask) fail to achieve competitive results against ours. This is because these masking strategies cannot be adaptively adjusted according to the different contexts, and adversarial training is not able to impact the imposed source masks as they are not learnable.

4.3. Ablation Studies

Effect of Adaptive Spatial Masking (ASM). First, in Fig. 2, we show qualitatively that injecting noise can ease the adversarial training. And the quantitative comparison in Table 1 and 2 with other masking strategies (SpatialDropout, RandMask, FreqMask) also verifies that ASM derives better masks for easing the adaptation.

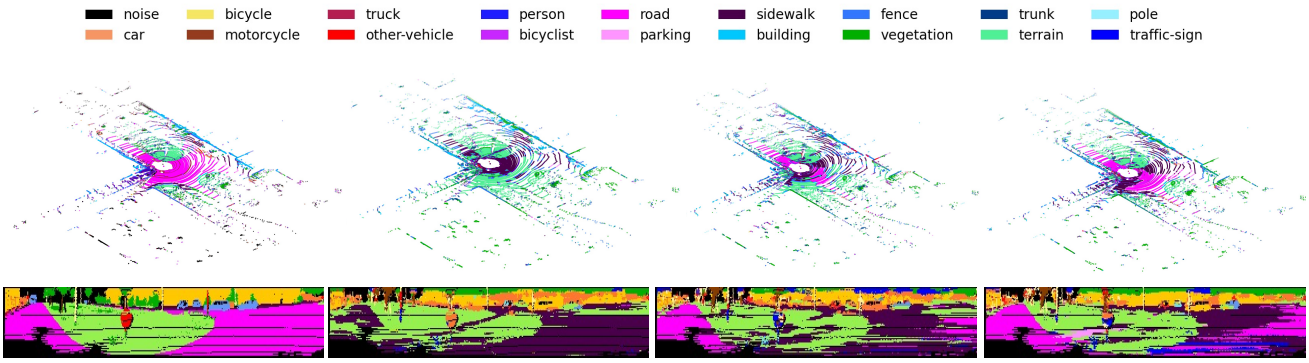
Effect of different branches of masking module. As discussed in Sec. 3.2, we use two embedding branches in the proposed ASM module. We evaluate the contribution of the two branches in Table 3 (a), where branch e receives source feature as input and branch o receives source Cartesian coordinates as input. It can be seen that removing either of them leads to an obvious drop in mIoU compared to the result using both branches. This verifies that both branches contribute to generating more effective masks.

Effect of Gumbel-Softmax. To evaluate the contribution of Gumbel-Softmax, we compare the results with training using plain Softmax which generates soft masks (*i.e.*, each mask value is within $[0, 1]$) for both forward and backward processes. As shown in Table 3 (b), using plain Softmax results in an obvious drop of mIoU, *i.e.*, -3.2% mIoU. This is because plain Softmax cannot actually zero out source points, rendering it hard to mimic the target noise patterns to mitigate the domain shift.

Table 3. Ablation studies on Adaptive Spatial Masking. Experiments are conducted on SynLiDAR [51] → SemKITTI [1].

Module	Modification	mIoU
(a) Two Branches	Branch e (embedding only)	21.7
	Branch o (coordinate only)	22.0
	Both Branch	22.8
(b) Mask Type	Plain Softmax (soft)	19.6
	Gumbel-Softmax (binary)	22.8
(c) Masking Layer	Input	22.0
	Ours	22.8
	Middle	21.7
	End of the backbone	21.6
(d) Update Strategy	$\mathcal{L}_{gen}$ optimizes θ_{ASM} only	21.6
	$\mathcal{L}_{gen}$ optimizes θ_{ASM} and θ_G	22.8

Effect of different masking layers. In Table 3 (c), we compare the results of inserting ASM at different layers of the network, including input (*i.e.*, masking the projected LiDAR image), ours (*i.e.*, after the first conv. layer), middle (between the encoder and the decoder), and end of the backbone. From the table, we observe that inserting ASM at the shallower layer can achieve better results, which avoids features being affected by domain shift from the early stage. Compared with the result of inserting ASM directly after the input, ours achieves better results, verifying the important role of exploiting shallow feature information in learning better masks. Besides, inserting ASM at the end of the backbone is worse but not that far from "Ours". This is be-



(a) Ground Truth (b) Source-only Model (c) AdaptSeg (d) Ours

Figure 5. **Visualization of Segmentation Results** (SynLiDAR → SemKITTI). We compare our method (d) with (a) ground truth, (b) source-only , and (c) AdaptSeg [43]. We present visualizations of both raw points (the first row) and projected point clouds (the second row). We show representative crops of projected 2D images due to the space limit.

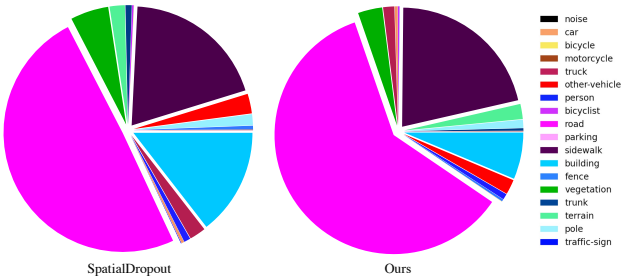


Figure 6. **Statistics of ignored source points** (SynLiDAR → SemKITTI). Compared with performing masking randomly, our method exhibits a different preference toward different classes. For example, contrary to SpatialDropout, fewer points from class "Building" are ignored and more points from "Road" are dropped.

cause masking at the end of the backbone also introduces noises to the discriminator and the classifier.

Optimization strategy with ASM . We investigate the optimization with the ASM module and present it in Table 3 (d). Only adversarially updating ASM leads to inferior results than updating both (*i.e.*, θ_G and θ_{ASM} in Eq.(4)). This shows that besides adversarially updating ASM, adversarially updating features also contributes to a better adaptation.

Analysis of masked samples To better understand the masking module, in Fig. 6, we present the class distribution of points that are ignored and compare with random dropout using a similar ignore ratio. Compared with random dropout, our result exhibits a different pattern/distribution, *e.g.*, our method ignores more points of class "Road" but fewer points of class "Building" and "Vegetation". However, our method outperforms it with a large margin, *i.e.*, +4.8% on SemKITTI and +2.1% on nuScenes. This indicates that our method can derive more reasonable noise distributions for mitigating the domain gap.

Sensitivity to hyper parameters. In Table 4, we present the sensitivity of our method to λ on both datasets. The performance of our method first increases and then decreases a little bit with the increase of λ from 5×10^{-4} to 5×10^{-3}. The bell shape of change verifies the regularization effect of adversarial training on the adaptation performance. Note that, within a wide range of choices of λ, our method consistently outperforms previous solutions by a large margin, which further verifies the effectiveness of our design.

Table 4. Sensitivity Analysis of λ (coefficient of $\mathcal{L}_{gen}$).

Transfer	5×10^{-4}	1×10^{-3}	5×10^{-3}
SynLiDAR → SemKITTI	21.9	22.8	21.6
SynLiDAR → nuScenes	16.6	17.0	16.3

Visualization In Fig. 5, we present the visualization of segmentation results on SynLiDAR→SemKITTI. From these figures, we observe that our method attains obvious improvement against source-only baseline and previous approach, which is in line with the superior results of our method shown in Table 1 and 2.

5. Conclusion

In this paper, we aim to mitigate the domain gap caused by target noises in synthetic-to-real point cloud segmentation adaptation. To this end, we propose Adversarial Masking, where a masking module is designed to derive learnable masks and the adversarial training paradigm encourages the masking module to mimic injecting target noises to source samples. The adversarial training and the masking module cooperate with each other to promote domain-invariant feature learning. Extensive experiments are conducted to prove the effectiveness of the proposed method.

Broader Impact and Limitations. Our method will not introduce bias but it may be impacted by the bias contained in the dataset. In terms of limitations, although our method outperforms the source-only baseline by a large margin, there is still a large gap to the oracle results. In future, we will explore more effective ways to narrow the domain gap.

References

[1] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *ICCV*, 2019. 6, 7

[2] Maxim Berman, Amal Rannen Triki, and Matthew B Blaschko. The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In *CVPR*, 2018. 5

[3] 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. *CVPR*, 2020. 6, 7

[4] Wei-Lun Chang, Hui-Po Wang, Wen-Hsiao Peng, and Wei-Chen Chiu. All about structure: Adapting structural information across domains for boosting semantic segmentation. In *CVPR*, 2019. 2

[5] Jaesung Choe, Chunghyun Park, Francois Rameau, Jaesik Park, and In So Kweon. Pointmixer: Mlp-mixer for point cloud understanding. *ECCV*, 2022. 1

[6] Tiago Cortinhal, George Tzelepis, and Eren Erdal Aksoy. Salsanext: Fast, uncertainty-aware semantic segmentation of lidar point clouds for autonomous driving, 2020. 3, 6, 7

[7] Runyu Ding, Jihan Yang, Li Jiang, and Xiaojuan Qi. Doda: Data-oriented sim-to-real domain adaptation for 3d semantic segmentation. In *ECCV*, 2022. 2

[8] Hehe Fan, Xiaojun Chang, Wanyue Zhang, Yi Cheng, Ying Sun, and Mohan Kankanhalli. Self-supervised global-local structure modeling for point cloud domain adaptation with reliable voted pseudo labels. In *CVPR*, 2022. 2

[9] Siqi Fan, Qiulei Dong, Fenghua Zhu, Yisheng Lv, Peijun Ye, and Fei-Yue Wang. Scf-net: Learning spatial contextual features for large-scale point cloud segmentation. In *CVPR*, 2021. 2

[10] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016. 4

[11] Rui Gong, Yuhua Chen, Danda Pani Paudel, Yawei Li, Ajad Chhatkuli, Wen Li, Dengxin Dai, and Luc Van Gool. Cluster, split, fuse, and update: Meta-learning for open compound domain adaptive semantic segmentation. In *CVPR*, 2021. 2

[12] Rui Gong, Wen Li, Yuhua Chen, and Luc Van Gool. Dlow: Domain flow for adaptation and generalization. In *IEEE CVPR*, 2019. 2

[13] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014. 4

[14] Tong He, Chunhua Shen, and Anton van den Hengel. DyCo3d: Robust instance segmentation of 3d point clouds through dynamic convolution. In *CVPR*, 2021. 2

[15] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. CyCADA: Cycle-consistent adversarial domain adaptation. In *ICML*, 2018. 4

[16] Lukas Hoyer, Dengxin Dai, and Luc Van Gool. Daformer: Improving network architectures and training strategies for domain-adaptive semantic segmentation. In *CVPR*, 2022. 2

[17] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Learning semantic segmentation of large-scale point clouds with random sampling. *IEEE TPAMI*, 2021. 1, 6

[18] Jiaxing Huang, Dayan Guan, Aoran Xiao, and Shijian Lu. Cross-view regularization for domain adaptive panoptic segmentation. In *CVPR*, 2021. 2

[19] Jiaxing Huang, Shijian Lu, Dayan Guan, and Xiaobing Zhang. Contextual-relation consistent domain adaptation for semantic segmentation. In *ECCV*, 2020. 2

[20] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *ICLR*, 2017. 5, 6

[21] 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*, 2020. 3

[22] Peng Jiang and Srikanth Saripalli. Lidarnet: A boundary-aware domain adaptation model for point cloud semantic segmentation. *ICRA*, 2021. 6, 7

[23] Guoliang Kang, Yunchao Wei, Yi Yang, Yueting Zhuang, and Alexander G Hauptmann. Pixel-level cycle association: A new perspective for domain adaptive semantic segmentation. In *NeurIPS*, 2020. 3, 6, 7

[24] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*, 2019. 2

[25] Qing Lian, Fengmao Lv, Lixin Duan, and Boqing Gong. Constructing self-motivated pyramid curriculums for cross-domain semantic segmentation: A non-adversarial approach. In *ICCV*, 2019. 2

[26] Hanxue Liang, Hehe Fan, Zhiwen Fan, Yi Wang, Tianlong Chen, Yu Cheng, and Zhangyang Wang. Point cloud domain adaptation via masked local 3d structure prediction. *ECCV*, 2022. 1

[27] Zhidong Liang, Zehan Zhang, Ming Zhang, Xian Zhao, and Shiliang Pu. Rangeioudet: Range image based real-time 3d object detector optimized by intersection over union. In *CVPR*, 2021. 3

[28] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *ICCV*, 2017. 5

[29] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss. RangeNet++: Fast and Accurate LiDAR Semantic Segmentation. In *IROS*, 2019. 3

[30] Pietro Morerio, Jacopo Cavazza, and Vittorio Murino. Minimal-entropy correlation alignment for unsupervised deep domain adaptation. *ICLR*, 2018. 2, 3

[31] Fei Pan, Inkyu Shin, Francois Rameau, Seokju Lee, and In So Kweon. Unsupervised intra-domain adaptation for semantic segmentation through self-supervision. In *CVPR*, 2020. 2

[32] Chunghyun Park, Yoonwoo Jeong, Minsu Cho, and Jaesik Park. Fast point transformer. In *CVPR*, 2022. 1

[33] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017. 2

[34] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CVPR*, 2017. 1, 2

[35] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *NeurIPS*, 2017. 2

[36] 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. *NeurIPS*, 2019. 2

[37] Mrigank Rochan, Shubhra Aich, Eduardo R Corral-Soto, Amir Nabatchian, and Bingbing Liu. Unsupervised domain adaptation in lidar semantic segmentation with self-supervision and gated adapters. *ICRA*, 2022. 2, 3

[38] Cristiano Saltori, Fabio Galasso, Giuseppe Fiameni, Nicu Sebe, Elisa Ricci, and Fabio Poiesi. Cosmix: Compositional semantic mix for domain adaptation in 3d lidar segmentation. *ECCV*, 2022. 3, 6, 7

[39] Yuefan Shen, Yanchao Yang, Mi Yan, He Wang, Youyi Zheng, and Leonidas J. Guibas. Domain adaptation on point clouds via geometry-aware implicits. In *CVPR*, 2022. 2

[40] Pei Sun, Weiyue Wang, Yuning Chai, Gamaleldin Elsayed, Alex Bewley, Xiao Zhang, Cristian Sminchisescu, and Dragomir Anguelov. Rsn: Range sparse net for efficient, accurate lidar 3d object detection. In *CVPR*, 2021. 3

[41] Haotian* Tang, Zhijian* Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *ECCV*, 2020. 2

[42] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. *ICCV*, 2019. 2

[43] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker. Learning to adapt structured output space for semantic segmentation. In *CVPR*, 2018. 3, 6, 7, 8

[44] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017. 4

[45] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Mathieu Cord, and Patrick Pérez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. In *CVPR*, 2019. 3, 6, 7

[46] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM TOG*, 2019. 2

[47] Bichen Wu, Alvin Wan, Xiangyu Yue, and Kurt Keutzer. Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. In *ICRA*, 2018. 2, 3, 6, 7

[48] 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*, 2019. 2, 3, 6, 7

[49] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. *CVPR*, 2019. 2, 3

[50] Zhonghua Wu, Yicheng Wu, Guosheng Lin, Jianfei Cai, and Chen Qian. Dual adaptive transformations for weakly supervised point cloud segmentation. *ECCV*, 2022. 2

[51] Aoran Xiao, Jiaxing Huang, Dayan Guan, Fangneng Zhan, and Shijian Lu. Synlidar: Learning from synthetic lidar sequential point cloud for semantic segmentation. *AAAI*, 2022. 3, 6, 7

[52] Chenfeng Xu, Bichen Wu, Zining Wang, Wei Zhan, Peter Vajda, Kurt Keutzer, and Masayoshi Tomizuka. Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation. In *ECCV*, 2020. 3, 6

[53] Mutian Xu, Runyu Ding, Hengshuang Zhao, and Xiaojuan Qi. Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds. In *CVPR*, 2021. 2, 6

[54] Qiangeng Xu, Xudong Sun, Cho-Ying Wu, Panqu Wang, and Ulrich Neumann. Grid-gcn for fast and scalable point cloud learning. *CVPR*, 2020. 2

[55] Yanchao Yang and Stefano Soatto. Fda: Fourier domain adaptation for semantic segmentation. In *CVPR*, 2020. 2

[56] Li Yi, Boqing Gong, and Thomas Funkhouser. Complete & label: A domain adaptation approach to semantic segmentation of lidar point clouds. In *CVPR*, 2021. 1

[57] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. *CVPR*, 2022. 1

[58] Pan Zhang, Bo Zhang, Ting Zhang, Dong Chen, Yong Wang, and Fang Wen. Prototypical pseudo label denoising and target structure learning for domain adaptive semantic segmentation. *CVPR*, 2021. 3

[59] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, Boqing Gong, and Hassan Foroosh. Polarnet: An improved grid representation for online lidar point clouds semantic segmentation. In *CVPR*, June 2020. 2

[60] Zhiyuan Zhang, Binh-Son Hua, and Sai-Kit Yeung. Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics. In *ICCV*, 2019. 2

[61] Sicheng Zhao, Yezhen Wang, Bo Li, Bichen Wu, Yang Gao, Pengfei Xu, Trevor Darrell, and Kurt Keutzer. epointda: An end-to-end simulation-to-real domain adaptation framework for lidar point cloud segmentation. In *AAAI*, 2021. 2, 3, 6, 7

[62] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 2, 3

[63] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In *CVPR*, 2021. 2, 6

[64] Yang Zou, Zhiding Yu, B.V.K. Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *ECCV*, 2018. 3, 6, 7

[65] Yang Zou, Zhiding Yu, Xiaofeng Liu, B.V.K. Vijaya Kumar, and Jinsong Wang. Confidence regularized self-training. In *ICCV*, 2019. 3