ePointDA: An End-to-End Simulation-to-Real Domain Adaptation Framework for LiDAR Point Cloud Segmentation

Sicheng Zhao¹+, Yezhen Wang²#, Bo Li¹, Bichen Wu¹, Yang Gao⁵, Pengfei Xu², Trevor Darrell¹, Kurt Keutzer¹

¹University of California, Berkeley ²Didi Chuxing ³University of California, San Diego ⁴Facebook Inc ⁵Tsinghua University
{schzhao,yezhen.wang0305,drluodian}@gmail.com, wbc@fb.com, gaoyangiiis@tsinghua.edu.cn, xupengfeipf@didichuxing.com, trevor@eecs.berkeley.edu, keutzer@berkeley.edu

Abstract

Due to its robust and precise distance measurements, LiDAR plays an important role in scene understanding for autonomous driving. Training deep neural networks (DNNs) on LiDAR data requires large-scale point-wise annotations, which are time-consuming and expensive to obtain. Instead, simulation-to-real domain adaptation (SRDA) trains a DNN using unlimited synthetic data with automatically generated labels and transfers the learned model to real scenarios. Existing SRDA methods for LiDAR point cloud segmentation mainly employ a multi-stage pipeline and focus on feature-level alignment. They require prior knowledge of real-world statistics and ignore the pixel-level dropout noise gap and the spatial feature gap between different domains. In this paper, we propose a novel end-to-end framework, named ePointDA, to address the above issues. Specifically, ePointDA consists of three modules: self-supervised dropout noise rendering, statistics-invariant and spatially-adaptive feature alignment, and transferable segmentation learning. The joint optimization enables ePointDA to bridge the domain shift at the pixel-level by explicitly rendering dropout noise for synthetic LiDAR and at the feature-level by spatially aligning the features between different domains, without requiring the real-world statistics. Extensive experiments adapting from synthetic GTA-LiDAR to real KITTI and SemanticKITTI demonstrate the superiority of ePointDA for LiDAR point cloud segmentation.

Introduction

Many types of multimedia data, such as images captured by cameras and point clouds collected by LiDAR (Light Detection And Ranging) and RADAR (Radio Detection And Ranging) can help to understand the semantics of complex scenes for autonomous driving. Among these sensors, LiDAR is an essential one for its specific properties (Wu et al. 2018a). LiDAR can provide precise distance measurements; for example, the error of Velodyne HDL-64E is less than 2cm¹. Further, it is more robust to ambient lighting conditions (e.g. night and shadow) than cameras and can obtain higher resolution and field of view than RADAR. Recent research has shown that deep neural networks (DNNs) can achieve state-of-the-art performance for point cloud classification and segmentation (Qi et al. 2017a,b; Wu et al. 2018a, 2019, Zhang, Hua, and Yeung 2019) with large-scale labeled data, which is usually time-consuming and expensive to obtain (Wang et al. 2019a). However, unlimited synthetic labeled data can be created using advanced simulators, such as CARLA² and GTA-V³ for autonomous driving. Unfortunately, due to the presence of domain shift between simulation and the real-world (Wu et al. 2019), as shown in Figure 1, direct transfer often results in significant performance decay. Domain adaptation (DA) aims to learn a transferable model to minimize the impact of domain shift between the source and target domains (Patel et al. 2015; Zhao et al. 2020b).

As the only simulation-to-real domain adaptation (SRDA) method for LiDAR point cloud segmentation, SqueezeSegV2 (Wu et al. 2019) consists of three stages: learned intensity rendering, Geodesic correlation alignment, and progressive domain calibration. Although it achieved state-of-the-art SRDA performance at the time, there are some limitations. First, it employs a multi-stage pipeline and cannot be trained end-to-end. Second, it does not consider the pixel-level dropout noise gap between different domains. Third, the progressive calibration is inefficient and lacks of robustness, as the accurate real-world statistics is difficult to estimate and is evolving with incremental data. Fourth, the standard convolution in the segmentation model neglects the drastic difference between spatial features and corresponding spatial feature gap across domains.

One might argue that we can apply the DA methods for RGB image segmentation, especially the ones performing both feature-level and pixel-level alignments (e.g. GTA-GAN (Sankaranarayanan et al. 2018), CyCADA (Hoffman et al. 2018)), to the SRDA problem for LiDAR point cloud segmentation. However, the 2D LiDAR images generated from 3D LiDAR point clouds projected onto a spherical surface (Wu et al. 2018a, 2019) are significantly different from RGB images. For example, RGB images mainly consist of color and texture, the style of which can be well translated by Generative Adversarial Network (GAN) (Goodfellow et al.

¹Corresponding Author. ²Equal contribution.
Copyright © 2021, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.
http://www.carla.org
https://velodyneLiDAR.com/products/hdl-64e
https://www.rockstargames.com/V
We conduct extensive experiments from synthetic GTA-LiDAR point cloud segmentation in an end-to-end manner.

In summary, the contributions of this paper are threefold:

- We are the first to study the simulation-to-real domain adaptation (SRDA) problem for LiDAR point cloud segmentation in an end-to-end manner.
- We design a novel framework, named ePointDA, to bridge the domain gap between the simulation and real domains at both the pixel-level and the feature-level through self-supervised dropout noise rendering and statistically-invariant and spatially-adaptive feature alignment.
- We conduct extensive experiments from synthetic GTA-LiDAR (Wu et al. 2019) to real KITTI (Geiger, Lenz, and Urtasun 2012) and SemanticKITTI (Behley et al. 2019), and respectively achieve 8.8% and 7.5% better IoU scores (on the “car” class) than the best DA baseline.

Related Work

Point Cloud Segmentation. Recent efforts on point cloud segmentation are typically based on DNNs. One straightforward way is to use the raw, un-ordered point clouds as input to a DNN. To deal with the order missing problem, symmetrical operators are usually applied, such as in PointNet (Qi et al. 2017a), PointNet++ (Qi et al. 2017b), and their improvements on hierarchical architecture (Klokov and Lempitsky 2017), sampling (Dovrat, Lang, and Avidan 2019), reordering (Li et al. 2018a), grouping (Li, Chen, and Hee Lee 2018), and efficiency (Liu et al. 2019b,a; Zhang, Hu, and Yeung 2019). There are also methods converting point clouds to regular 3D voxel grids (Wang et al. 2017; Huang, Wang, and Neumann 2018; Le and Duan 2018; Lei, Akhtar, and Mian 2019; Mao, Wang, and Li 2019; Meng et al. 2019) or constructing graphs from point clouds for network processing (Te et al. 2018; Jiang et al. 2019; Xu et al. 2019; Landrieu and Simonovsky 2018; Wang et al. 2019b,c). However, these methods suffer from some limitations, such as inefficiency and point collision (Lyu, Huang, and Zhang 2020). To address the efficiency problem and enable real-time inference, one popular method is to project 3D point clouds to 2D images, including sphere mapping (Wu et al. 2018a, 2019; Milioto et al. 2019; Behley et al. 2019; Xu et al. 2020), 2D grid sampling (Caltagirone et al. 2017), and graph drawing (Lyu, Huang, and Zhang 2020). In this paper, we follow the spherical projection method of SqueezeSeg (Wu et al. 2018a, 2019).

Point Cloud Simulation. Some efforts have been dedicated to creating large-scale real-world point cloud datasets, such as 3D bounding box to point-wise labeling (Wang et al. 2019a) and densely annotated SemanticKITTI dataset (Behley et al. 2019). However, it is still difficult or impossible to collect all required point cloud scenes, such as traffic accidents in autonomous driving. The synthetic data generated by advanced simulators can achieve this goal with unlimited labeled data (Richter et al. 2016; Dosovitskiy et al. 2017; Yue et al. 2018; Krähenbühl 2018; Tripathi et al. 2019). In this paper, we employ the synthetic GTA-LiDAR dataset (Wu et al. 2019) with depth segmentation map generated by (Krähenbühl 2018) and Image-LiDAR registration in GTA-V by (Yue et al. 2018).

Unsupervised Domain Adaptation. Most existing research on DA focuses on the single-source and unsupervised setting, i.e. adapting from one labeled source domain to another unlabeled target domain (Zhao et al. 2020b).Recent deep unsupervised DA methods usually employ a conjoined architecture with two streams. Besides the task loss on the labeled source domain, another alignment loss is designed to align the source and target domains, such as discrepancy loss (Long et al. 2015; Sun, Feng, and Saenko 2016; Zhuo et al. 2017; Wu et al. 2019; Chen et al. 2020), adversarial loss (Long et al. 2015; Sun, Feng, and Saenko 2016; Zhuo et al. 2017; Wu et al. 2019; Chen et al. 2020).
Figure 2: The proposed SRDA framework ePointDA for LiDAR point cloud segmentation. The colored dashed arrows correspond to different losses. For clarity the real-to-simulation cycle is omitted. See Figure 3 for detailed segmentation network.

Figure 3: Our segmentation network vs. SqueezeSegV2 (Wu et al. 2019). We replace all standard convolution (conv) and the final conditional random field (CRF) with aligned spatially-adaptive convolution (ASAC). We replace all batch normalization after each conv with instance normalization.

On one hand, most SRDA methods exploring synthetic data focus on 2D RGB images for object classification (Peng et al. 2019), pose estimation (Shrivastava et al. 2017), and semantic segmentation (Sankaranarayanan et al. 2018; Zhao et al. 2019a). On the other hand, existing DA methods for LiDAR point cloud perception either conduct classification (Qin et al. 2019; Achituve, Maron, and Chechik 2021) or detection (Saleh et al. 2019; Rist, Enzweiler, and Gavrila 2019), tasks, or perform real-to-real segmentation (Rist, Enzweiler, and Gavrila 2019; Jiang and Saripalli 2020). The only SRDA method for LiDAR point cloud segmentation is SqueezeSegV2 (Wu et al. 2019), but it is trained stage by stage. We propose to study SRDA for LiDAR point cloud segmentation in an end-to-end manner.

**Approach**

Given labeled synthetic LiDAR and unlabeled real LiDAR, our goal is to learn a transferable segmentation model by aligning the source simulation domain and target real domain. Following SqueezeSeg (Wu et al. 2018a, 2019), we project sparse 3D LiDAR point clouds to 2D images for efficient processing, i.e., projecting each point in the Cartesian coordinate to the angular coordinate. In this way, a LiDAR point cloud is transformed to a LiDAR image with size $H \times W \times C$, where $H$, $W$ are the height and width of the projected image$^4$, and $C$ is the number of image channels$^5$.

We consider the one-source, unsupervised, homogeneous, and closed-set SRDA scenario for LiDAR point cloud segmentation. That is, there is one labeled simulation domain and one unlabeled real domain, the observed LiDAR data of different domains are from the same space, and the label categories are shared across different domains. Suppose the projected synthetic images and corresponding labels drawn from the synthetic distribution $P_s(x, y)$ are $X_s = \{x_s^i\}_{i=1}^{N_s}$ and $Y_s = \{y_s^i\}_{i=1}^{N_s}$, respectively, where $x_s^i \in \mathbb{R}^{H \times W \times C}$, $y_s^i \in \{1, 2, \ldots, L\}^{H \times W}$, $L$ is the number of label categories, and $N_s$ is the number of synthetic samples. Let $X_r = \{x_r^j\}_{j=1}^{N_r}$ denote the projected real images drawn from the real distribution $P_r(x)$, where $N_r$ is the number of real samples.

$^4$We use the LiDAR collected by Velodyne HDL-64E with 64 vertical channels, $H = 64$; and use the frontal 90 degrees of the scan, dividing it into 512 grids, $W = 512$.

$^5$In experiment, we use the Cartesian coordinates $(x, y, z)$ as features for each point, i.e., $C = 3$. We also tried other features, such as range and (rendered) intensity (Wu et al. 2019), but the experiments show that adding these channels does not result in performance improvement for domain adaptation.
of real samples. On the basis of covariate shift and concept drift (Patel et al. 2015), we aim to learn a segmentation model that can correctly predict the labels for each pixel of a real sample trained on \((\{X_s, Y_s\})\) and \(\{X_r\}\).

The framework of ePointDA is illustrated in Figure 2, which includes three modules. Self-supervised dropout noise rendering (SDNR) aims to bridge the domain shift at the pixel-level by generating adapted images based on the rendered dropout noise. Statistics-invariant and spatially-adaptive feature alignment aims to bridge the domain shift at the feature-level by considering the instance-wise statistics variations and spatial statistics differences. Transferable segmentation learning can then learn a transferable segmentation model based on the adapted images and corresponding synthetic labels.

**Self-Supervised Dropout Noise Rendering**

LiDAR point clouds in the real-world usually contain significant dropout noise, i.e., missing points, where all coordinates \((x, y, z)\) are zero. However, synthetic LiDAR does not contain such noise, as it is difficult to simulate. Besides the random dropout noise, we propose an inpainting-based rendering method in a self-supervised manner to render other dropout noises, such as the ones caused by mirror reflection.

First, we employ CycleGAN (Zhu et al. 2017) to fill the dropout noise with the following pixel-level GAN loss and cycle-consistency loss:

\[
\mathcal{L}_{GAN}^{\rightarrow s}(G_s, D_s, X_r, X_s) = \mathbb{E}_{x_r \sim X_r} \log D_s(G_s(x_r)) + \mathbb{E}_{x_s \sim X_s} \log [1 - D_s(x_s)],
\]

\[
\mathcal{L}_{GAN}^{\leftarrow r}(G_r, D_r, X_s, X_r) = \mathbb{E}_{x_s \sim X_s} \log D_r(G_r(x_s)) + \mathbb{E}_{x_r \sim X_r} \log [1 - D_r(x_r)],
\]

\[
\mathcal{L}_{cyc}(G_s, G_r, X_r, X_s) = \mathbb{E}_{x_r \sim X_r} \| G_r(G_s(x_r)) - x_r \|_1 + \mathbb{E}_{x_s \sim X_s} \| G_s(G_r(x_s)) - x_s \|_1,
\]

where \(G_s, G_r\) are generators from real-to-simulation and simulation-to-real, and \(D_s, D_r\) are discriminators for the simulation and real domains, respectively.

Second, based on the binary dropout noise mask \(M = \{0, 1\}^{H \times W}\), we can train a pixel-wise rendering network \(R\) with the following cross-entropy loss:

\[
\mathcal{L}_{mask}(R, G_s, X_s, M) = -\mathbb{E}_{(x_r, m) \sim (X_r, M)} \sum_{n=1}^{H} \sum_{h=1}^{W} I_{[m_h = m_n]} \log(\sigma(R_{n,h,w}(G_s(x_r))))
\]

where \(\sigma\) is the softmax function, \(I\) is an indicator function, and \(R_{n,h,w}(G_s(x_r))\) is the value of \(F(G_s(x_r))\) at index \((n, h, w)\). After training \(R\), we can render dropout noise for synthetic data and obtain adapted LiDAR images:

\[
x_s = R(x_r) \odot x_s
\]

that is similar to the real domain, which can be viewed as a pixel-level dropout noise alignment. Further, these two modules are designed for different purposes: CAM focuses on decreasing the negative effect of dropout noise in one specific domain, while SDNR aims to eliminate the domain gap between two different domains.

**Statistics-Invariant and Spatially-Adaptive Feature Alignment**

**Motivation.** SqueezeSegV2 (Wu et al. 2019) aligns the features between the simulation and real domains during training by Geodesic correlation alignment (Morierio, Cavazza, and Murino 2018), and employs progressive domain calibration (PDC) (Li et al. 2018b) to progressively calibrate the statistic shift during post-processing. There are some limitations of this feature alignment method: (1) The PDC module requires the DA pipeline to be designed as multi-stage. Further, it depends heavily on a good sampling of the real-world distribution to obtain accurate statistics, which is very difficult in real applications. (2) The correlation alignment only matches the second-order (covariance) statistics of different distributions, which cannot completely characterize the complex non-Gaussian deep features (Chen et al. 2020). (3) It neglects the spatial feature gap. Xu et al. (2020) found that the feature distribution of LiDAR images changes drastically at different spatial locations while natural images hold a relatively identical distribution among various locations. Spatially-adaptive convolution (SAC) is proposed by learning a location-wise attention map (Xu et al. 2020). However, directly transplanting the SAC module into SRDA tasks does not guarantee better performance, because there also exists a spatial feature gap between synthetic LiDAR and real LiDAR, as shown in Figure 4.

In this paper, we propose statistic-invariant and spatially-adaptive feature alignment to address the above issues by (1) extracting statistics-invariant features by instance normalization (Ulyanov, Vedaldi, and Lempitsky 2016), (2) align-
ing the feature maps in high-dimensional space by higher-order moment matching (Chen et al. 2020), and (3) generating domain-invariant spatial attention map by improving the SAC module (Xu et al. 2020).

**Statistics-Invariant Feature Extraction.** To eliminate the influence of statistics variations among different instances across domains, we employ instance normalization (IN) (Ulyanov, Vedaldi, and Lempitsky 2016) to normalize each channel of the CNN feature maps, which has been demonstrated to be effective for fast style transfer in RGB images (Wu et al. 2018b). Specifically, suppose the feature maps for synthetic image $x_i^s$ and real image $x_i^r$ after the same activation layer are $f_i^s$ and $f_i^r$ respectively, of the same size $\mathbb{R}^{C \times H \times W}$. We can then easily conduct IN by:

$$f_i^s = \frac{f_i^s - \mu(f_i^s)}{\sigma(f_i^s)}, f_i^r = \frac{f_i^r - \mu(f_i^r)}{\sigma(f_i^r)},$$

$$\mu_c(f) = \frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} f_{chw},$$

$$\sigma_c(f) = \sqrt{\frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} (f_{chw} - \mu_c(f))^2},$$

where $\mu_c(f)$ and $\sigma_c(f)$ are the mean and variance across spatial dimensions for the $c$-th channel.

**Higher-Order Moment Matching.** We employ higher-order moment matching (HOMM) (Chen et al. 2020), a discrepancy-based feature-level alignment method, to align the high-order statistics between the simulation and real domains with the following discrepancy loss:

$$L_{HOMM}(\phi, R, X_s, X_r) = \frac{1}{N^2} \| \mathbb{E}_{x_s \sim X_s} (\phi(R(x_s) \odot X_r)^{\odot p}) - \mathbb{E}_{x_r \sim X_r} (\phi(x_r)^{\odot p}) \|_F^2,$$

where $\| \cdot \|_F$ is the Frobenius norm, $\phi(\cdot)$ and $N$ denote the activation outputs and the number of hidden neurons of the adapted layer respectively, and $\odot^p$ represents the $p$-level tensor power of the vector $\phi$.

**Domain-Invariant Spatial Attention Generation.** SAC (Xu et al. 2020) introduces one convolution to learn a location-wise attention map. To eliminate the spatial feature gap, we modify the structure by extracting the attention map from the previous feature maps instead of from the original images (Xu et al. 2020), as shown in Figure 5. The basic motivation is that by extracting the spatial attention map according to preceding feature maps, we can align those feature maps between different domains so that the inputs of the aligned SAC (ASAC) module are those domain-invariant features. Once the ASAC module can only see domain-invariant features during the training stage, it is more robust to generate attention map when dealing with the target data. It is worth noting that we do not need any extra operation because those feature maps have already been aligned by the employed HoMM method.

**Transferable Segmentation Learning**

After generating adapted LiDAR images that have similar styles to real images and aligning the features of the adapted images and real images, we can train a transferable task segmentation model $F$ based on adapted images $\{R(x_s) \odot x_s\}$ and corresponding synthetic labels $Y_s$ with the following focal loss (Lin et al. 2017; Wu et al. 2019):

$$L_{seg}(F, R, X_s, Y_s) = - \mathbb{E}_{(x_s, y_s)} \sum_{l=1}^{L} \sum_{h=1}^{H} \sum_{w=1}^{W} (1 - p_{l,h,w})^\gamma \log p_{l,h,w},$$

where $p_{l,h,w} = \sigma(F_{l,h,w}(R(x_s) \odot x_s))$, $F_{l,h,w}(\cdot)$ is the value of $F(\cdot)$ at index $(l, h, w)$, and $\gamma$ is a focusing parameter to adjust the rate at which well-classified examples are down-weighted. When $\gamma = 0$, focal loss equals cross-entropy loss. The advantage of focal loss is that it can deal with the imbalanced distribution of point cloud categories, e.g. more background points than foreground objects.

**ePointDA Learning**

The proposed ePointDA learning framework utilizes adaptation techniques to bridge the domain shift at both pixel-level and feature-level. Combining these alignment modules with transferable segmentation learning, we can obtain the overall objective loss function of ePointDA as:

$$L_{ePointDA}(G_s, D_s, G_r, D_r, R, F) = L_{GAN}(G_s, D_s, X_r, X_s) + \frac{L_{eSeg}(F, R, X_s, X_r) + L_{HoMM}(\phi, R, X_r, X_s) + L_{mcm}(R, G_s, X_s, M)}{4}.$$  \hspace{1cm} (9)

The training process corresponds to solving for the target segmentation model $F$ according to:

$$F^* = \arg \min_{F} \min_{R} \max_{G_s, G_r} L_{ePointDA}(G_s, D_s, G_r, D_r, R, F).$$  \hspace{1cm} (10)

As shown in Eq. (9), there are 6 different losses in the overall objective loss function. The SDNR module involves 4 losses: GAN losses in two directions, cycle-consistency loss, and dropout noise rendering (DNR) loss. The feature alignment and segmentation learning modules correspond to HoMM loss and segmentation supervision loss, respectively. The GAN losses and cycle-consistency loss are used to update the CycleGAN network while the DNR loss is used to update the dropout noise prediction network. The gradient backpropagation of DNR loss would be truncated at the position of dropout noise prediction network’s inputs. Similarly, the HoMM loss and segmentation supervision loss are used to update the segmentation network, indicating that the gradient will be backpropagated only to the segmentation network.
Table 1: Comparison with the state-of-the-art DA methods for LiDAR point cloud segmentation from GTA-LiDar to KITTI, where +ASAC denotes using the spatial feature aligned SAC module, and +HHead denotes replacing the CRF layer with an conv layer. The best IoU of each category trained on the simulation domain is emphasized in bold.

<table>
<thead>
<tr>
<th>Method</th>
<th>Car</th>
<th>Pedestrian</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pre</td>
<td>Rec</td>
</tr>
<tr>
<td>Source-only</td>
<td>34.4</td>
<td>67.6</td>
</tr>
<tr>
<td>DAN (Long et al. 2015)</td>
<td>56.3</td>
<td>78.4</td>
</tr>
<tr>
<td>CORAL (Sun and Saenko 2016)</td>
<td>56.5</td>
<td>82.1</td>
</tr>
<tr>
<td>ADDA (Chen et al. 2020)</td>
<td>59.4</td>
<td>85.2</td>
</tr>
<tr>
<td>SqueezeSegV2 (Hoffman et al. 2018)</td>
<td>70.4</td>
<td>82.1</td>
</tr>
<tr>
<td>ePointDA (Ours)</td>
<td>76.0</td>
<td>84.7</td>
</tr>
<tr>
<td>Oracle (SqueezeSegV2)</td>
<td>77.2</td>
<td>84.7</td>
</tr>
<tr>
<td>Oracle+SAC</td>
<td>76.1</td>
<td>92.1</td>
</tr>
<tr>
<td>Oracle+SAC+HHead</td>
<td>78.4</td>
<td>91.4</td>
</tr>
<tr>
<td>Oracle+SAC+HHead</td>
<td>77.8</td>
<td>93.1</td>
</tr>
<tr>
<td>SqueezeSegV2 (Wu et al. 2019)</td>
<td>76.0</td>
<td>92.1</td>
</tr>
</tbody>
</table>

Table 2: Comparison with the state-of-the-art DA methods from GTA-LiDar to SemanticKITTI.

<table>
<thead>
<tr>
<th>Method</th>
<th>Car</th>
<th>Pedestrian</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pre</td>
<td>Rec</td>
</tr>
<tr>
<td>Source-only</td>
<td>65.8</td>
<td>40.8</td>
</tr>
<tr>
<td>DAN (Long et al. 2015)</td>
<td>73.6</td>
<td>68.4</td>
</tr>
<tr>
<td>CORAL (Sun and Saenko 2016)</td>
<td>73.4</td>
<td>71.3</td>
</tr>
<tr>
<td>ADDA (Chen et al. 2020)</td>
<td>73.8</td>
<td>69.5</td>
</tr>
<tr>
<td>SqueezeSegV2 (Wu et al. 2019)</td>
<td>65.3</td>
<td>84.5</td>
</tr>
<tr>
<td>Oracle+SAC</td>
<td>78.3</td>
<td>85.1</td>
</tr>
<tr>
<td>Oracle+SAC+HHead</td>
<td>75.9</td>
<td>93.8</td>
</tr>
<tr>
<td>Oracle+SAC+HHead</td>
<td>77.9</td>
<td>88.5</td>
</tr>
</tbody>
</table>

Experiments

In this section, we introduce the experimental settings and compare ePointDA with state-of-the-art approaches, followed by series of ablation studies and visualizations.

Experimental Settings

Datasets. We perform SRDA from synthetic GTA-LiDar (Wu et al. 2019) to real KITTI (Geiger, Lenz, and Urtasun 2012) and SemanticKITTI (Behley et al. 2019) datasets for LiDAR point cloud segmentation. GTA-LiDar (Wu et al. 2019) contains 100,000 LiDAR point clouds synthesized in GTA-V. The depth segmentation map is generated by (Krähenbühl 2018) and the Image-LiDar registration is conducted by (Yue et al. 2018). There are one label and x, y, z coordinates for each point in the synthetic point cloud, without dropout noise and intensity.

KITTI (Geiger, Lenz, and Urtasun 2012; Wu et al. 2018a) contains 10,848 samples with point-wise labels obtained from the original 3D bounding boxes. As in SqueezeSegV2 (Wu et al. 2019), we split the dataset into a training set with 8,057 samples and a test set with 2,791 samples.

SemanticKITTI (Behley et al. 2019) is a recently released large-scale dataset for LiDAR point-cloud segmentation with 21 sequences and 43,442 densely annotated scans. Following (Behley et al. 2019), we employ sequences-0-7 and 9, 10 (19,130 scans) for training, sequence-08 (4,071 scans) for validation, and sequences-11-21 (20,351 scans) for testing.

Since there are only two categories in the GTA-LiDar dataset, i.e. car and pedestrian, we select the images in KITTI and SemanticKITTI that contain these two categories and report the segmentation adaptation results. Constructing a synthetic dataset with more categories for LiDAR point cloud segmentation and performing SRDA remains our future work.

Evaluation Metrics. Similar to (Wu et al. 2018a), we employ precision, recall, and intersection-over-union (IoU) to evaluate the class-level segmentation results by comparing the predicted results with ground-truth labels point-wisely:

\[
\text{Pre}_l = \frac{|P_l \cap G_l|}{|P_l|}, \quad \text{Rec}_l = \frac{|P_l \cap G_l|}{|G_l|}, \quad \text{IoU}_l = \frac{|P_l \cap G_l|}{|P_l| + |G_l| - |P_l \cap G_l|},
\]

where \(P_l\) and \(G_l\) respectively denote the predicted and ground-truth point sets that belong to class-\(l\), and \(|\cdot|\) represents the cardinality of a set. Larger precision, recall, and IoU values represent better results. We employ IoU as the primary metric.

Comparison with the State-of-the-art

The performance comparisons between ePointDA and the state-of-the-art DA methods are shown in Table 1 and Ta-
One may argue that ePointDA contains some extra modules (i.e., Aligned SAC and one more conv layer), which might be unfair to compare with SqueezeSegV2. Even if we drop these modules, ePointDA still outperforms SqueezeSegV2 by a large margin (63.4 vs. 57.4). However, this still does not make our method the best. As demonstrated in Table 5, our method can improve the oracle performance. Replacing the CRF layer in SqueezeSegV2 with a conv layer also works under the oracle setting.

**Table 3:** Ablation study on different modules, where Baseline denotes a simplified SqueezeSegV2 model (Wu et al. 2019) for fair comparison taking the Cartesian coordinates as input and using batch normalization, frequency-based DNR, and Geodesic correlation alignment.

<table>
<thead>
<tr>
<th>Method</th>
<th>Car</th>
<th>Pedestrian</th>
<th>Car</th>
<th>Pedestrian</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pre Rec IoU</td>
<td>Pre Rec IoU</td>
<td>Pre Rec IoU</td>
<td>Pre Rec IoU</td>
</tr>
<tr>
<td>Baseline</td>
<td>75.8 83.2 53.1</td>
<td>21.5 77.1 20.2</td>
<td>72.1 81.9 63.4</td>
<td>24.7 75.4 23.9</td>
</tr>
<tr>
<td>+SDNR</td>
<td>65.8 84.8 58.7</td>
<td>24.7 75.4 22.8</td>
<td>73.4 81.9 63.4</td>
<td>29.4 56.0 23.9</td>
</tr>
<tr>
<td>+SDNR+IN</td>
<td>69.3 86.9 62.7</td>
<td>28.8 57.6 23.8</td>
<td>73.4 81.9 63.4</td>
<td>29.4 56.0 23.9</td>
</tr>
<tr>
<td>+SDNR+IN+HoMM</td>
<td>72.1 85.6 64.2</td>
<td>31.2 57.5 28.3</td>
<td>72.1 85.6 64.2</td>
<td>31.2 57.5 28.3</td>
</tr>
<tr>
<td>+SDNR+IN+HoMM+ASAC</td>
<td>75.2 84.7 66.2</td>
<td>28.7 65.2 24.8</td>
<td>72.1 85.6 64.2</td>
<td>31.2 57.5 28.3</td>
</tr>
</tbody>
</table>

**Table 4:** Ablation study on different normalization schemes using both frequency-based DNR and our learned DNR without feature alignment. ‘BN’, ‘IN’, ‘LN’, ‘GN’ are short for batch normalization, instance normalization, layer normalization, and group normalization, respectively.

<table>
<thead>
<tr>
<th>Method</th>
<th>Car</th>
<th>Pedestrian</th>
<th>Car</th>
<th>Pedestrian</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pre Rec IoU</td>
<td>Pre Rec IoU</td>
<td>Pre Rec IoU</td>
<td>Pre Rec IoU</td>
</tr>
<tr>
<td>Baseline</td>
<td>73.4 81.9 63.4</td>
<td>29.4 56.0 23.9</td>
<td>73.4 81.9 63.4</td>
<td>29.4 56.0 23.9</td>
</tr>
<tr>
<td>SAC (Xu et al. 2020)</td>
<td>68.5 83.2 60.2</td>
<td>25.2 62.4 21.9</td>
<td>72.1 85.6 64.2</td>
<td>31.2 57.5 25.3</td>
</tr>
<tr>
<td>ASAC (ours)</td>
<td>72.1 85.6 64.2</td>
<td>31.2 57.5 25.3</td>
<td>72.1 85.6 64.2</td>
<td>31.2 57.5 25.3</td>
</tr>
</tbody>
</table>

**Table 5:** Comparison between ordinary SAC (Xu et al. 2020) and our aligned SAC (ASAC). Baseline corresponds the “+SDNR+IN+HoMM” setting in Table 3.

**Ablation Study**

We conduct a series of ablation studies when adapting from GTA-LiDAR to KITTI. First, we incrementally investigate the effectiveness of different modules in ePointDA. From the results in Table 3, we can observe that: (1) adding each module can improve the IoU scores, which demonstrates that all the modules contained in ePointDA contribute to the SRDA task; (2) among all these modules, SDNR provides the highest performance improvement (5.6%), which demonstrates the important role that dropout noise plays in the domain gap between the simulation and real domains and the necessity of exploring effective DNR model.

Second, we explore the differences caused by various normalization schemes, including batch normalization (BN) (Ioffe and Szegedy 2015), instance normalization (IN) (Ulyanov, Vedaldi, and Lempitsky 2016), layer normalization (LN) (Ba, Kiros, and Hinton 2016), and group normalization (GN) (Wu and He 2018). From Table 4, it is clear that IN, LN, and GN all outperform BN. The relative poor performance of BN results from the statistics gap between the simulation and real domains. IN, LN, and GN can all eliminate such gap to some extent (Wu et al. 2018b) and thus achieve better DA results than BN.

Third, we compare our aligned SAC (ASAC) and the ordinary SAC (Xu et al. 2020) in Table 5. Without consider-
Table 6: Ablation study on the number of convolution layers (#Conv) that are appended to the last deconvolution layer. This experiment is conducted after dropout noise rendering and feature alignment, i.e., +SDNR+IN+HoMM+ASAC.

<table>
<thead>
<tr>
<th>#Conv</th>
<th>Car Pre</th>
<th>Car Rec</th>
<th>Car IoU</th>
<th>Pedestrian Pre</th>
<th>Pedestrian Rec</th>
<th>Pedestrian IoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>72.1</td>
<td>85.6</td>
<td>64.2</td>
<td>31.2</td>
<td>57.5</td>
<td>25.3</td>
</tr>
<tr>
<td>2</td>
<td>75.2</td>
<td>84.7</td>
<td><strong>66.2</strong></td>
<td>28.7</td>
<td>65.2</td>
<td>24.8</td>
</tr>
<tr>
<td>3</td>
<td>74.1</td>
<td>82.3</td>
<td>63.8</td>
<td>26.9</td>
<td>64.8</td>
<td>23.4</td>
</tr>
<tr>
<td>4</td>
<td>70.6</td>
<td>83.4</td>
<td>61.9</td>
<td>25.1</td>
<td>59.7</td>
<td>21.5</td>
</tr>
<tr>
<td>5</td>
<td>68.6</td>
<td>84.0</td>
<td>60.7</td>
<td>26.5</td>
<td>52.3</td>
<td>21.3</td>
</tr>
</tbody>
</table>

Finally, we study the influence of convolution (conv) layers appended to the last deconvolution layer of the segmentation network. From Table 6, we can conclude that adding 2 conv layers performs the best. As stated in (Li et al. 2019; Yu and Koltun 2016; Dai et al. 2017, 2016), the receptive field can affect the network’s effectiveness. As the number of conv layers increases, the segmentation results become better; after reaching the best receptive field (2 conv layers), the segmentation results decrease gradually.

**Visualization**

First, we qualitatively visualize the LiDAR point cloud segmentation results from GTA-LiDAR to KITTI in Figure 6. We can clearly see that after adaptation by ePointDA, the segmentation results are improved notably as compared to source-only and SqueezeSegV2 (Wu et al. 2019). For example, in the first column, ePointDA avoids falsely detecting some pedestrians. In the second column, ePointDA classifies the cyclist as a pedestrian, which is reasonable because there is no cyclist in the GTA-LiDAR dataset. Please note that there does exist some missing objects in the original annotations, such as the car on the right corner in the third column. This is either because some objects are not labeled in the original 3D point clouds, or because some information is missing during the projection from 3D point clouds to 2D images. For fair comparison, we used the same setting as SqueezeSegV2. We will refine the annotations in our future work to make the proposed method more practical.

Second, we visualize the DNR results on KITTI. From the results in Figure 7, it is clear that the rendered dropout noise is very close to the ground truth, which demonstrates the effectiveness of our SDNR method.

**Conclusion**

In this paper, we proposed an end-to-end simulation-to-real domain adaptation (SRDA) framework, named ePointDA, for LiDAR point cloud segmentation. By explicitly rendering dropout noise for the real domain in a self-supervised manner and spatially aligning higher-level moments between the simulation and real domains, ePointDA bridges the domain shift at both the pixel-level and feature-level. Further, ePointDA does not require prior statistics of the real domain, which makes it more robust and practical. The extensive experiments adapting from synthetic GTA-LiDAR to real KITTI and SemanticKITTI demonstrated that ePointDA significantly outperforms the state-of-the-art SRDA methods. The proposed ePointDA can be easily applied to other applications, such as robotics grasping. It can also be applied to other depth data, like the depth from Kinect, which has similar properties to LiDAR. Another promising extension is the specific adaptation setting where the source and target domains have different channels, such as adapting from RGB to RGB-Depth.

In future studies, we plan to construct a large-scale synthetic dataset for LiDAR point cloud segmentation containing more compatible categories with SemanticKITTI and extend our framework to corresponding SRDA tasks. We will explore multi-modal domain adaptation by jointly modeling multiple modalities, such as image and LiDAR.
Acknowledgments
This work is supported by the National Natural Science Foundation of China (No. 61701273), Berkeley DeepDrive, and the Zhongguancun Haihua Institute for Frontier Information Technology.

References


