# CoSMix: Compositional Semantic Mix for Domain Adaptation in 3D LiDAR Segmentation

Cristiano Saltori[1], Fabio Galasso[2], Giuseppe Fiameni[3],
Nicu Sebe[1], Elisa Ricci[1,4], and Fabio Poiesi[4]

[1] University of Trento, Trento, Italy
[2] Sapienza University of Rome, Rome, Italy
[3] NVIDIA AI Technology Center
[4] Fondazione Bruno Kessler, Trento, Italy
cristiano.saltori@unitn.it

**Abstract.** 3D LiDAR semantic segmentation is fundamental for autonomous driving. Several Unsupervised Domain Adaptation (UDA) methods for point cloud data have been recently proposed to improve model generalization for different sensors and environments. Researchers working on UDA problems in the image domain have shown that sample mixing can mitigate domain shift. We propose a new approach of sample mixing for point cloud UDA, namely Compositional Semantic Mix (CoSMix), the first UDA approach for point cloud segmentation based on sample mixing. CoSMix consists of a two-branch symmetric network that can process labelled synthetic data (source) and real-world unlabelled point clouds (target) concurrently. Each branch operates on one domain by mixing selected pieces of data from the other one, and by using the semantic information derived from source labels and target pseudo-labels. We evaluate CoSMix on two large-scale datasets, showing that it outperforms state-of-the-art methods by a large margin.[5]

**Keywords:** Unsupervised domain adaptation, point clouds, semantic segmentation, LiDAR.

## 1 Introduction

Point cloud semantic segmentation is the problem of assigning a finite set of semantic labels to a set of 3D points [6,42]. When deep learning-based approaches are employed to perform this task, large-scale datasets with point-level annotations are required to learn accurate models [3, 4, 20]. This implies a costly and cumbersome data collection procedure, as point clouds need to be captured in the real world and manually annotated. An alternative is to use synthetic data, which can be conveniently generated with simulators [8]. However, deep neural networks are known to underpeform when trained and tested on data from different domains, due to *domain shift* [7]. Although significant effort has been invested to design simulators that can reproduce the acquisition sensor with high

[5] Our code is available at https://github.com/saltoricristiano/cosmix-uda.

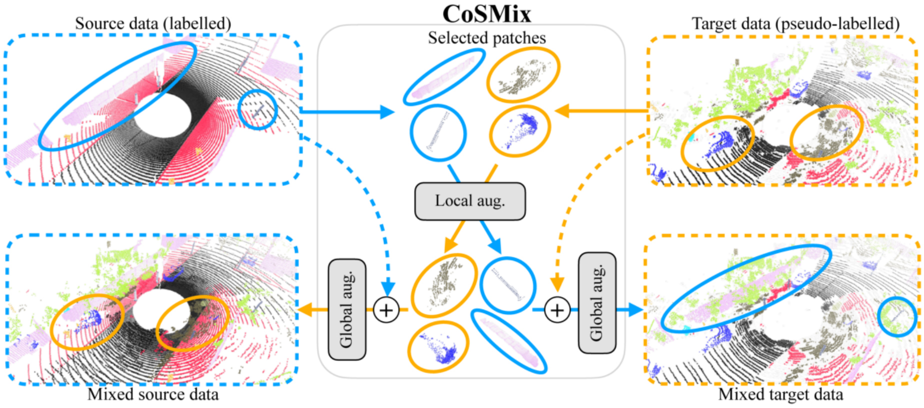


Fig. 1: CoSMix applied to source and target data. Given (labelled) source and (pseudo-labelled) target data, we select domain-specific patches with semantic information to be mixed across domains. The resulting mixed data are a compositional semantic mix between the two domains, mixing source supervision in the target domain and target self-supervision (object and scene structure) in the source domain. Augmentations are applied at both local and global levels.

fidelity, further research is still needed to neutralize the domain shift between real and synthetic domains [32].

Unsupervised Domain Adaptation (UDA) for semantic segmentation has been widely studied for image data [9,25,28,44,45], however less attention has been paid to adaptation techniques for point clouds. Approaches to address synthetic-to-real UDA for point clouds can operate in the input space [32,41] by using dropout rendering [41], or in the feature space through feature alignment [30], or can use adversarial networks [32]. In the last few years, data augmentation approaches based on mixing of training samples and their labels, such as Mixup [38] or CutMix [36], have been proposed to promote generalization. These techniques can be used for image classification [36,38], image recognition [17,33], and 2D semantic segmentation [9,25]. A few works proposed to exploit sample mixing for point cloud data [5,19,39,43], but they are formulated for supervised applications. We argue that the major challenge in extending 2D mix-based UDA approaches to point clouds lies in the application of these to geometric signals rather than photometric signals, e.g., the weighted alpha blending performed of labels in 2D [36,38] is still unclear how to extend it to 3D.

In this paper, we propose a novel UDA framework for 3D LiDAR segmentation, named CoSMix, which can mitigate the domain shift by creating two new intermediate domains of composite point clouds obtained by applying a novel mixing strategy at input level (Fig. 1). Our framework is based on a two-branch symmetric deep network structure that processes synthetic labelled point clouds (source) and real-world unlabelled point clouds (target). Each branch is associated to a domain, *i.e.*, on the source branch, a given source point cloud is mixed with parts of a target point cloud and vice versa for the target branch. The mixing operation is implemented as a composition operation, which is similar to the concatenation operation proposed in [5, 19, 43], but unlike them, we account for the semantic information from source labels and target pseudo-labels to apply data augmentation both at local and global semantic level. An additional key difference is the teacher-student learning scheme that we introduce to improve pseudo-label accuracy and, thus, point cloud composition. We extensively evaluate our approach on recent and large scale segmentation benchmarks, *i.e.*, considering SynLiDAR [32] as source dataset, and SemanticPOSS [20] and SemanticKITTI [3] as target. Our results show that CoSMix successfully alleviates the domain shift and outperforms state-of-the-art methods. We also perform an in-depth analysis of CoSMix and an ablation study on each component, highlighting its strengths and discussing its main limitations. To the best of our knowledge, this is the first work to have proposed a sample mixing scheme for adaptation in the context of 3D point cloud segmentation.

Our main contributions can be summarised as follows:

- We introduce a novel scheme for mixing point clouds by leveraging semantic information and data augmentation.
- We show that the proposed mixing strategy can be used for reducing the domain shift and design CoSMix, the first UDA method for 3D LiDAR semantic segmentation based on point cloud mixing.
- We conduct extensive experiments on two challenging synthetic-to-real 3D LiDAR semantic segmentation benchmarks demonstrating the effectiveness of CoSMix, which outperforms state-of-the-art methods.

## 2 Related works

**Point cloud semantic segmentation.** Point cloud segmentation can be performed by using PointNet [21] that is based on a series of multilayer perceptrons. PointNet++ [22] improves PointNet by leveraging point aggregations performed at neighbourhood level and multi-scale sampling to encode both local features and global features. RandLA-Net [13] extends PoinNet++ [22] by embedding local spatial encoding, random sampling and attentive pooling. These methods are computationally inefficient when large-scale point clouds are processed. Recent segmentation methods have improved the computational efficiency by projecting 3D points on 2D representations or by using 3D quantization approaches. The former includes 2D projection based approaches that use 2D range maps and exploit standard 2D architectures [24] to segment these maps prior to a re-projection in the 3D space. RangeNet++ [18], SqueezeSeg networks [29, 30], 3D-MiniNet [2] and PolarNet [40] are examples of these approaches. Although these approaches are efficient, they tend to loose information when the input data are projected in 2D and re-projected in 3D. The latter includes 3D quantization-based approaches that discretize the input point cloud into a 3D discrete representations and that employ 3D convolutions [37] or 3D sparse convolutions [6, 12] to predict per-point classes. In this category, we find methods such as VoxelNet [37], SparseConv [11,12], MinkowskiNet [6] and, Cylinder3D [42]. In our work, we use the MinkowskiNet [6] which provides a trade off between accuracy and efficiency.

**Unsupervised domain adaptation for point cloud segmentation.** Unsupervised Domain Adaptation (UDA) for point cloud segmentation can be used in the case of real-to-real [15,16,35] and synthetic to real scenarios [29,30,41]. Real-to-real adaptation can be used when a deep network is trained with data of real-world scenes captured with a LiDAR sensors and then tested on unseen scenes captured with a different LiDAR sensor [16,35]. Therein, domain adaptation can be formulated as a 3D surface completion task [35] or by transferring the sensor pattern of the target domain to the source domain through ray casting [16]. Synthetic-to-real domain adaptation can be used when the source data are acquired with a simulated LiDAR sensor [8] and the target data are obtained with a real LiDAR sensor. In this case, domain shift occurs due to differences in (i) sampling noise, (ii) structure of the environment and (iii) class distributions [30,41]. Attention models can be used to aggregate contextual information [29,30] and geodesic correlation alignment with progressive domain calibration can be adopted to improve domain adaptation [30]. In [41], real dropout noise is simulated on synthetic data through a generative adversarial network. Similarly, in [32] domain shift is disentangled into appearance difference and sparsity difference and a generative network is applied to mitigate each difference. In our work, we do not use a learning-based approach to perturb the input data, but we formulate a novel compositional semantic point cloud mixing approach that enables the deep network to improve its performance on the unlabelled target domain self-supervisedly.

**Sample Mixing for UDA.** Deep neural networks often exhibit undesired behaviours such as memorization and overfitting. To alleviate this problem, mixing strategies [36,38] train a network on additional data derived from the convex combination of paired samples and labels, which are obtained either mixing the whole samples [38] or cutting and pasting their patches [36]. Mixing strategies showed their effectiveness also in reducing domain shift in UDA for image classification [31,33] and semantic segmentation [9,25,34]. In DACS [25], mixed samples are created by mixing pairs of images from different domains by using source ground-truth annotations pasted on pseudo-labelled target images. In DSP [9], authors adopt a strategy that prioritize the selection of long-tail classes from the source domain images, and to paste their corresponding image patches on other source images and on target images. The first point cloud mixing strategies [5,19,39] showed that point cloud pairs and their point-level annotations can be mixed for improving accuracy in semantic segmentation [19] and classification [5,39]. Zou *et al.* [43] propose to use Mix3D [19] as a pretext task for classification by predicting the rotation angle of mixed pairs. Apply mixing strategy to address UDA in 3D semantic segmentation has not been previously investigated. We fill this gap by introducing a novel compositional semantic mixing strategy that goes beyond the standard concatenation of two point clouds [19,39] or of randomly selected crops [39].

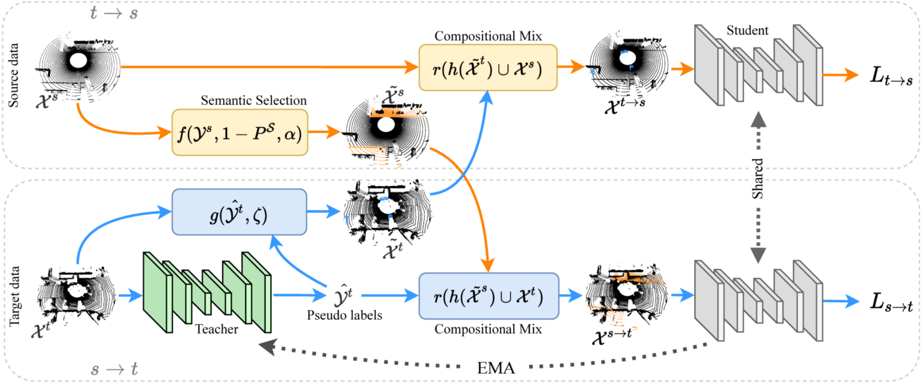


Fig. 2: Block diagram of CoSMix. In the top branch, the input source point cloud $\mathcal{X}^s$ is mixed with the target point cloud $\mathcal{X}^t$ obtaining $\mathcal{X}^{t \rightarrow s}$. In the bottom branch, the input target point cloud $\mathcal{X}^t$ is mixed with the source point cloud $\mathcal{X}^s$ obtaining $\mathcal{X}^{s \rightarrow t}$. A teacher-student learning architecture is used to improve pseudo-label accuracy while adapting over target domain. Semantic Selection ($f$ and $g$) selects subsets of points (patches) to be mixed based on the source labels $\mathcal{Y}^s$ and target pseudo-labels $\hat{\mathcal{Y}}^t$ information. Compositional Mix applies local $h$ and global $r$ augmentations and mixes the selected patches among domains.

## 3 Our approach

CoSMix implements a teacher-student learning scheme that exploits the supervision from the source domain and the self-supervision from the target domain to improve the semantic segmentation on the target domain. Our method is trained on two different mixed point cloud sets. The first is the composition of the source point cloud with pseudo-labelled pieces, or *patches*, of the target point cloud. Target patches bring the target modality in the source domain pulling the source domain closer to the target domain. The second is the composition of the target point cloud with randomly selected patches of the source point cloud. Source patches pull the target domain closer to the source domain, preventing overfitting from noisy pseudo-labels. The teacher-student network enables the iterative improvement of pseudo labels, progressively reducing the domain gap.

Fig. 2 shows the block diagram of CoSMix. Let $\mathcal{S} = \{(\mathcal{X}^s, \mathcal{Y}^s)\}$ be the source dataset that is composed of $N^s = |\mathcal{S}|$ labelled point clouds, where $\mathcal{X}^s$ is a point cloud and $\mathcal{Y}^s$ is its point-level labels, and $|.|$ is the cardinality of a set. Labels take values from a set of semantic classes $\mathcal{C} = \{c\}$, where $c$ is a semantic class. Let $\mathcal{T} = \{\mathcal{X}^t\}$ be the target dataset composed of $N^t = |\mathcal{T}|$ unlabelled point clouds. On the top branch, the source point cloud $\mathcal{X}^s$ is mixed with selected patches of the target point cloud $\mathcal{X}^t$. The target patches are subsets of points that correspond to the most confident pseudo-labels $\hat{\mathcal{Y}}^t$ that the teacher network produces during training. On the bottom branch, the target point cloud $\mathcal{X}^t$ is mixed with the selected patches of the source point cloud $\mathcal{X}^s$. The source patches are subsets of points that are randomly selected based on their class frequency distribution in the training set. Let $\mathcal{X}^{t\to s}$ be the mixed point cloud obtained from the top branch, and $\mathcal{X}^{s\to t}$ be the mixed point cloud obtained from the bottom branch. We define the branch that mixes target point cloud patches to the source point cloud as $t \to s$ and the branch that does the vice versa as $s \to t$. Lastly, let $\Phi_\theta$ and $\Phi_{\theta'}$ be the student and teacher deep networks with learnable parameters $\theta$ and $\theta'$, respectively.

We explain how the semantic selection operates on the source and target point clouds in Sec. 3.1. We detail the modules in charge of mixing the point clouds coming from the different domains in Sec. 3.2. Then, we describe how the teacher network is updated during training and the loss functions that we use to train the student networks in Sec. 3.3.

### 3.1 Semantic selection

In order to train the student networks with balanced data, we select reliable and informative point cloud patches prior to mixing points and labels across domains. A point cloud patch corresponds to a subset of points of the same semantic class. To select patches from the source point cloud, we rely on the class frequency distribution by counting the number of points for each semantic class within $\mathcal{S}$. Unlike DSP [9], we do not select long-tail classes in advance, but we instead exploit the source distribution and the semantic classes available to dynamically sample classes at each iteration.

We define the class frequency distribution of $\mathcal{S}$ as $P^s_{\mathcal{Y}}$ and create a function $f$ that randomly selects a subset of classes based on the labels $\tilde{\mathcal{Y}}^s \subset \mathcal{Y}^s$ for supervision at each iteration. The likelihood that $f$ selects a class $c$ is inversely proportional to its class frequency in $\mathcal{S}$. Specifically,

$$\tilde{\mathcal{Y}}^s = f(\mathcal{Y}^s, 1 - P^s_{\mathcal{Y}}, \alpha), \tag{1}$$

where $\alpha$ is an hyperparameter that regulates the ratio of selected classes for each point cloud. For example, by setting $\alpha = 0.5$, the algorithm will select a number of patches corresponding to the 50% of the classes available by sampling them based on their class frequency distribution, i.e., long-tailed classes will have a higher likelihood to be selected. We define the set of points that correspond to $\tilde{\mathcal{Y}}^s$ as $\tilde{\mathcal{X}}^s$, and a patch as the set of points $\tilde{\mathcal{X}}^s_c \subset \tilde{\mathcal{X}}^s$ that belong to class $c \in \mathcal{C}$.

To select patches from the target point clouds, we apply the same set of operations but using the pseudo-labels produced by the teacher network based on their prediction confidence. Specifically, we define a function $g$ that selects reliable pseudo-labels based on their confidence value. The selected pseudo-labels are defined as

$$\tilde{\mathcal{Y}}^t = g(\Phi_{\theta'}(\mathcal{X}^t), \zeta), \tag{2}$$

where $\Phi_{\theta'}$ is the teacher network, $\zeta$ is the confidence threshold used by the function $g$ and $\tilde{\mathcal{Y}}^t \subset \hat{\mathcal{Y}}^t$. We define the set of points that correspond to $\tilde{\mathcal{Y}}^t$ as $\tilde{\mathcal{X}}^t$.

### 3.2 Compositional mix

The goal of our compositional mixing module is to create mixed point clouds based on the selected semantic patches. The compositional mix involves three consecutive operations: *local random augmentation*, patches are augmented randomly and independently from each other; *concatenation*, the augmented patches are concatenated to the point cloud of the other domain to create the mixed point cloud; *global random augmentation*, the mixed point cloud is randomly augmented. This module is applied twice, once for the $t \rightarrow s$ branch (top of Fig. 2), where target patches are mixed within the source point cloud, and once for the $s \rightarrow t$ branch (bottom of Fig. 2), where source patches are mixed within the target point cloud. Unlike Mix3D [19], our mixing strategy embeds data augmentation at local level and global level.

In the $s \rightarrow t$ branch, we apply the local random augmentation $h$ to all the points $\tilde{\mathcal{X}}^s_c \subset \tilde{\mathcal{X}}^s$. We repeat this operation for all $c \in \tilde{\mathcal{Y}}^s$. Note that $h$ is a random augmentation that produces a different result each time it is applied to a set of points. Therefore, we define the result of this operation as

$$h(\tilde{\mathcal{X}}^s) = \left\{ h(\tilde{\mathcal{X}}^s_c), \forall c \in \tilde{\mathcal{Y}}^s \right\}. \tag{3}$$

Then, we concatenate $h(\tilde{\mathcal{X}}^s)$ with the source point cloud and apply the global random augmentation. Their respective labels are concatenated accordingly, such as

$$\mathcal{X}^{s\rightarrow t} = r(h(\tilde{\mathcal{X}}^s) \cup \mathcal{X}^t), \quad \mathcal{Y}^{s\rightarrow t} = \tilde{\mathcal{Y}}^s \cup \mathcal{Y}^t, \tag{4}$$

where $r$ is the global augmentation function. The same operations are also performed in the $t \rightarrow s$ branch by mixing target patches within the source point cloud. Instead of using source labels, we use the teacher network's pseudo-labels obtained from the target data and concatenate them with the labels of the source data. This results in $\mathcal{X}^{t\rightarrow s}$ and $\mathcal{Y}^{t\rightarrow s}$.

### 3.3 Network update

We leverage the teacher-student learning scheme to facilitate the transfer of knowledge acquired during the course of the training with mixed domains. We use the teacher network $\Phi_{\theta'}$ to produce target pseudo-labels $\hat{\mathcal{Y}}^t$ for the student network $\Phi_\theta$, and train $\Phi_\theta$ to segment target point clouds by using the mixed point clouds $\mathcal{X}^{s\rightarrow t}$ and $\mathcal{X}^{t\rightarrow s}$ based on their mixed labels and pseudo-labels (Sec. 3.2).

At each batch iteration, we update the student parameters $\Phi_\theta$ to minimize a total objective loss $\mathcal{L}_{tot}$ defined as

$$\mathcal{L}_{tot} = \mathcal{L}_{s\rightarrow t} + \mathcal{L}_{t\rightarrow s}, \tag{5}$$

where $\mathcal{L}_{s\rightarrow t}$ and $\mathcal{L}_{t\rightarrow s}$ are the $s \rightarrow t$ and $t \rightarrow s$ branch losses, respectively. Given $\mathcal{X}^{s\rightarrow t}$ and $\mathcal{Y}^{s\rightarrow t}$, we define the segmentation loss for the $s \rightarrow t$ branch as

$$\mathcal{L}_{s\rightarrow t} = \mathcal{L}_{seg}(\Phi_\theta(\mathcal{X}^{s\rightarrow t}), \mathcal{Y}^{s\rightarrow t}), \tag{6}$$

the objective of which is to minimize the segmentation error over $\mathcal{X}^{s\to t}$, thus learning to segment source patches in the target domain. Similarly, given $\mathcal{X}^{t\to s}$ and $\mathcal{Y}^{t\to s}$, we define the segmentation loss for the $t\to s$ branch as

$$\mathcal{L}_{t\to s} = \mathcal{L}_{seg}(\Phi_{\theta}(\mathcal{X}^{t\to s}), \mathcal{Y}^{t\to s}), \tag{7}$$

whose objective is to minimize the segmentation error over $\mathcal{X}^{t\to s}$ where target patches are composed with source data. We implement $\mathcal{L}_{seg}$ as the Dice segmentation loss [14], which we found effective for the segmentation of large-scale point clouds as it can cope with long-tail classes well.

Lastly, we update the teacher parameters $\theta'$ every $\gamma$ iterations following the exponential moving average (EMA) [9] approach

$$\theta'_i = \beta\theta'_{i-1} + (1-\beta)\theta, \tag{8}$$

where $i$ indicates the training iteration and $\beta$ is a smoothing coefficient hyperparamenter.

## 4 Experiments

We evaluate our method in the synthetic-to-real UDA scenario for LiDAR segmentation. We use the SynLiDAR dataset [32] as (synthetic) source domain, and the SemanticKITTI [1,3,10] and SemanticPOSS [20] datasets as (real) target domains (more details in Sec. 4.1). We describe CoSMix implementation in Sec. 4.2. We compare CoSMix with five state-of-the-art UDA methods: two general purpose adaptation methods (ADDA [26], Ent-Min [27]), one image segmentation method (ST [45]) and, two point cloud segmentation methods (PCT [32], ST-PCT [32]) (Sec. 4.3). Like [32], we compare CoSMix against methods working on 3D point clouds for synthetic to real, such as PCT [32] and ST-PCT [32]. These are the only two state-of-the-art methods for synthetic-to-real UDA that use 360° LiDAR point clouds. Results of baselines are taken from [32].

### 4.1 Datasets and metrics

**SynLiDAR** [32] is a large-scale synthetic dataset that is captured with the Unreal Engine [8]. It is composed of 198,396 LiDAR scans with point-level segmentation annotations over 32 semantic classes. We follow the authors' instructions [32], and use 19,840 point clouds for training and 1,976 point clouds for validation.
**SemanticPOSS** [20] consists of 2,988 real-world scans with point-level annotations over 14 semantic classes. Based on the official benchmark guidelines [20], we use the sequence 03 for validation and the remaining sequences for training.
**SemanticKITTI** [3] is a large-scale segmentation dataset consisting of LiDAR acquisitions of the popular KITTI dataset [1,10]. It is composed of 43,552 scans captured in Karlsruhe (Germany) and point-level annotations over 19 semantic classes. Based on the official protocol [3], we use sequence 08 for validation and the remaining sequences for training.

**Class mapping.** Like [32], we make source and target labels compatible across our datasets, i.e., SynLiDAR → SemanticPOSS and SynLiDAR → SemanticKITTI. We map SynLiDAR labels into 14 segmentation classes for SynLiDAR → SemanticPOSS and 19 segmentation classes for SynLiDAR → SemanticKITTI [32].
**Metrics.** We follow the typical evaluation protocol for UDA in 3D semantic segmentation [32] and evaluate the segmentation performance before and after adaptation. We compute the Intersection over the Union (IoU) [23] for each segmentation class and report the per-class IoU. Then, we average the IoU over all the segmented classes and report the mean Intersection over the Union (mIoU).

### 4.2 Implementation details

We implemented CoSMix in PyTorch and run our experiments on 4×NVIDIA A100 (40GB SXM4). We use MinkowskiNet as our point cloud segmentation network [6]. For a fair comparison, we use MinkUNet32 as in [32]. We use warm-up, i.e., our network is pre-trained on the source domain for 10 epochs with Dice loss [14] starting from randomly initialized weights. During the adaptation step, we initialize student and teacher networks with the parameters obtained after warm-up. The warm-up and adaptation stage share the same hyperparameters. In both the warm-up and adaptation steps, we use Stochastic Gradient Descent (SGD) with a learning rate of 0.001. We set $\alpha$ by analyzing the long-tailed classes in the source domain during adaptation. We experimentally found $\alpha = 50\%$ to be a good value in each task. In the target semantic selection function $g$, we set $\zeta$ such that about 80% of pseudo-labelled points per scene can be selected. On SynLiDAR→SemanticPOSS, we use a batch size of 12 and perform adaptation for 10 epochs. We set source semantic selection $f$ with $\alpha = 0.5$ while target semantic selection $g$ with a confidence threshold $\zeta = 0.85$ (Sec. 3.1). On SynLiDAR→SemanticKITTI, we use a batch size of 16, adapting for 3 epochs. During source semantic selection $f$ we set $\alpha = 0.5$ while in target semantic selection $g$ we use a confidence threshold of $\zeta = 0.90$. We use the same compositional mix (Sec. 3.2) parameters for both the adaptation directions. We implement the local augmentation $h$ as rigid rotation around the $z$-axis, scaling along all the axes and random point downsampling. We bound rotations between $[-\pi/2, \pi/2]$ and scaling between $[0.95, 1.05]$, and perform random downsampling for 50% of the patch points. For global augmentation $r$, we use a rigid rotation, translation and scaling along all the three axes. We set $r$ parameters to the same used in [6]. During the network update step (Sec. 3.3), we obtain the teacher parameters $\theta'_i$ with $\beta = 0.99$ every $\gamma = 1$ steps on SynLiDAR→SemanticPOSS and every $\gamma = 500$ steps on SynLiDAR→SemanticKITTI.

### 4.3 Quantitative comparisons

Tab. 1 and Tab. 2 reports the adaptation results on SynLiDAR→SemanticPOSS, and on SynLiDAR→SemanticKITTI, respectively. The Source model is the lower bound of each scenario with 20.7 mIoU on SynLiDAR→SemanticPOSS and 22.2 mIoU on SynLiDAR→SemanticKITTI. We highlight in gray the associated results

Table 1: Adaptation results on SynLiDAR → SemanticPOSS. Source corresponds to the model trained on the source synthetic dataset (lower bound in gray). Results are reported in terms of mean Intersection over the Union (mIoU).

| **Model** | **pers.** | **rider** | **car** | **trunk** | **plants** | **traf.** | **pole** | **garb.** | **buil.** | **cone.** | **fence** | **bike** | **grou.** | **mIoU** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Source | 3.7 | 25.1 | 12.0 | 10.8 | 53.4 | 0.0 | 19.4 | 12.9 | 49.1 | 3.1 | 20.3 | 0.0 | 59.6 | 20.7 |
| ADDA [26] | 27.5 | 35.1 | 18.8 | 12.4 | 53.4 | 2.8 | 27.0 | 12.2 | 64.7 | 1.3 | 6.3 | 6.8 | 55.3 | 24.9 |
| Ent-Min [27] | 24.2 | 32.2 | 21.4 | 18.9 | 61.0 | 2.5 | 36.3 | 8.3 | 56.7 | 3.1 | 5.3 | 4.8 | 57.1 | 25.5 |
| ST [45] | 23.5 | 31.8 | 22.0 | 18.9 | 63.2 | 1.9 | **41.6** | 13.5 | 58.2 | 1.0 | 9.1 | 6.8 | 60.3 | 27.1 |
| PCT [32] | 13.0 | 35.4 | 13.7 | 10.2 | 53.1 | 1.4 | 23.8 | 12.7 | 52.9 | 0.8 | 13.7 | 1.1 | 66.2 | 22.9 |
| ST-PCT [32] | 28.9 | 34.8 | 27.8 | 18.6 | 63.7 | 4.9 | 41.0 | 16.6 | 64.1 | 1.6 | 12.1 | 6.6 | 63.9 | 29.6 |
| CoSMix (Ours) | **55.8** | **51.4** | **36.2** | **23.5** | **71.3** | **22.5** | 34.2 | **28.9** | **66.2** | **20.4** | **24.9** | **10.6** | **78.7** | **40.4** |

Table 2: Adaptation results on SynLiDAR → SemanticKITTI. Source corresponds to the model trained on the source synthetic dataset (lower bound in gray). Results are reported in terms of mean Intersection over the Union (mIoU).

| **Model** | **car** | **bi.cle** | **mt.cle** | **truck** | **oth-v.** | **pers.** | **b.clst** | **m.clst** | **road** | **park.** | **sidew.** | **oth-g.** | **build.** | **fence** | **veget.** | **trunk** | **terra.** | **pole** | **traff.** | **mIoU** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Source | 42.0 | 5.0 | 4.8 | 0.4 | 2.5 | 12.4 | 43.3 | 1.8 | 48.7 | 4.5 | 31.0 | 0.0 | 18.6 | 11.5 | 60.2 | 30.0 | 48.3 | 19.3 | 3.0 | 20.4 |
| ADDA [26] | 52.5 | 4.5 | 11.9 | 0.3 | 3.9 | 9.4 | 27.9 | 0.5 | 52.8 | 4.9 | 27.4 | 0.0 | 61.0 | 17.0 | 57.4 | 34.5 | 42.9 | 23.2 | 4.5 | 23.0 |
| Ent-Min [27] | 58.3 | 5.1 | 14.3 | 0.3 | 1.8 | 14.3 | **44.5** | 0.5 | 50.4 | 4.3 | 34.8 | 0.0 | 48.3 | 19.7 | 67.5 | 34.8 | **52.0** | 33.0 | 6.1 | 25.8 |
| ST [45] | 62.0 | 5.0 | 12.4 | 1.3 | 9.2 | 16.7 | 44.2 | 0.4 | 53.0 | 2.5 | 28.4 | 0.0 | 57.1 | 18.7 | 69.8 | **35.0** | 48.7 | 32.5 | 6.9 | 26.5 |
| PCT [32] | 53.4 | 5.4 | 7.4 | 0.8 | 10.9 | 12.0 | 43.2 | 0.3 | 50.8 | 3.7 | 29.4 | 0.0 | 48.0 | 10.4 | 68.2 | 33.1 | 40.0 | 29.5 | 6.9 | 23.9 |
| ST-PCT [32] | 70.8 | **7.3** | 13.1 | 1.9 | 8.4 | 12.6 | 44.0 | 0.6 | 56.4 | 4.5 | 31.8 | 0.0 | **66.7** | **23.7** | **73.3** | 34.6 | 48.4 | **39.4** | 11.7 | 28.9 |
| CoSMix (Ours) | **75.1** | 6.8 | **29.4** | **27.1** | **11.1** | **22.1** | 25.0 | **24.7** | **79.3** | **14.9** | **46.7** | **0.1** | 53.4 | 13.0 | 67.7 | 31.4 | 32.1 | 37.9 | **13.4** | **32.2** |

in both tables. In SynLiDAR→SemanticPOSS (Tab. 1), CoSMix outperforms the baselines on all the classes, with the exception of *pole* where ST achieves better results. On average, we achieve 40.4 mIoU surpassing ST-PCT by +10.8 mIoU and improving over the Source of +19.7 mIoU. Interestingly, CoSMix improves also on difficult classes as in the case of *person*, *traffic-sign*, *cone* and, *bike*, whose performance were low before adaptation. SemanticKITTI is a more challenging domain as the validation sequence includes a wide range of different scenarios with a large number of semantic classes. In SynLiDAR→SemanticKITTI (Tab. 2), CoSMix improves on all the classes when compared to Source, with the exception of *bicyclist* and *terrain*. We relate this behaviour to the additional noise introduced by pseudo labels on these classes and in related classes such as *sidewalk*. Compared to the other baselines, CoSMix improves on 11 out of 19 classes, with a large margin in the classes *car*, *motorcycle*, *truck*, *person*, *road*, *parking* and *sidewalk*. On average, also in this more challenging scenario, we achieve the new state-of-the-art performance of 32.2 mIoU, outperforming ST-PCT by +3.3 mIoU and improving over Source of about +11.8 mIoU.

### 4.4 Qualitative results

We report qualitative examples of the adaptation performance before (source) and after CoSMix adaptation (ours), and compare them to ground-truth annotations

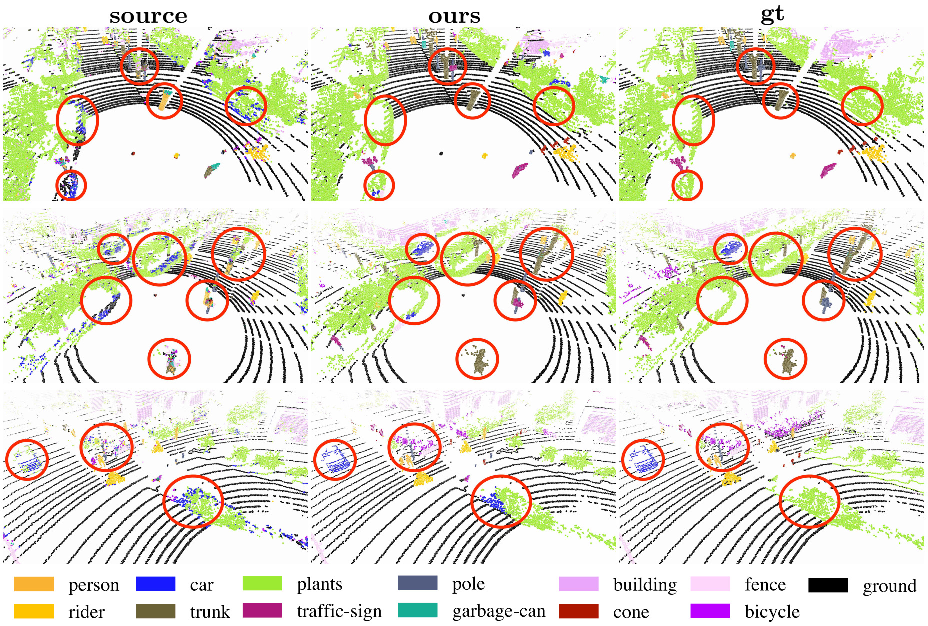


Fig. 3: Results on SynLiDAR→SemanticPOSS. Source predictions are often wrong and mingled in the same region. After adaptation, CoSMix improves segmentation with homogeneous predictions and correctly assigned classes. The red circles highlight regions with interesting results.

(gt). Fig. 3 shows the adaptation results on SynLiDAR→SemanticPOSS, while Fig. 4 show the results on SynLiDAR→SemanticKITTI. Red circles highlight regions with interesting results. In Fig. 3, improvements are visible in multiple regions of the examples. Source predictions are often not homogeneous with completely wrong regions. After adaptation, CoSMix improves segmentation with more homogeneous regions and correctly assigned classes. In Fig. 4, source predictions are less sparse but wrong for several spatial regions. After adaptation, CoSMix allows better and correct predictions. Additional examples can be found in the Supplementary Material.

## 5 Ablation study

We perform an ablation study of CoSMix by using the SynLiDAR → SemanticPOSS setup. We compare our mixing approach with a recent point cloud mixing strategy [19] by applying it to the synthetic-to-real setting (Sec. 5.2). In Sec. 5.3, we investigate the importance of confidence threshold in CoSMix.

### 5.1 Method components

We analyze CoSMix by organizing its components into three groups: mixing strategies (*mix*), augmentations (*augs*) and other components (*others*). In the

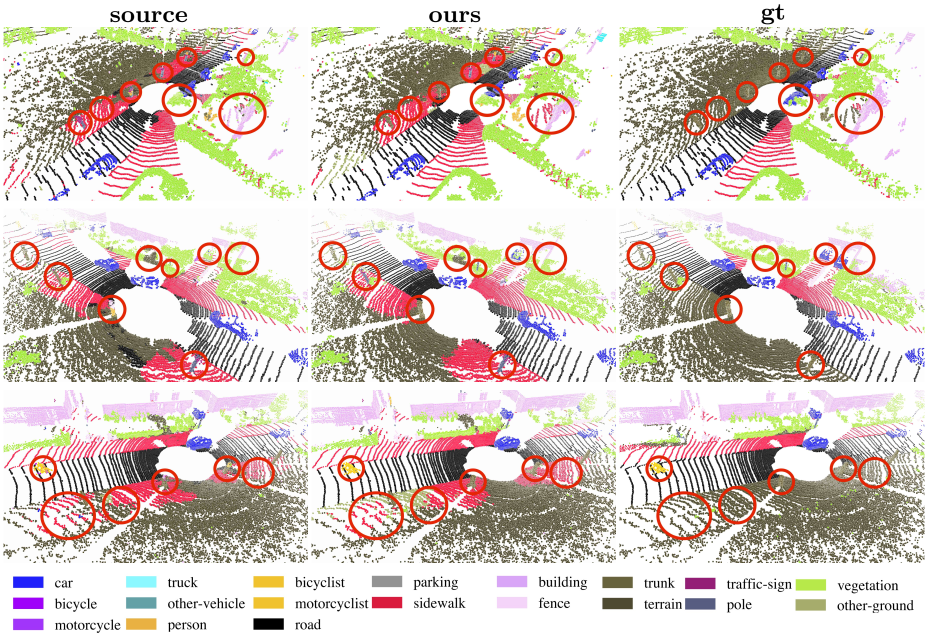


Fig. 4: Results on SynLiDAR→SemanticKITTI. Source predictions are often wrong and mingled in the same region. After adaptation, CoSMix improves segmentation with homogeneous predictions and correctly assigned classes. The red circles highlight regions with interesting results.

*mix* group, we assess the importance of the mixing strategies ($t \rightarrow s$ and $s \rightarrow t$) used in our compositional mix (Sec. 3.2) after semantic selection. In the *augs* group, we assess the importance of the local $h$ and global $r$ augmentations that are used in the compositional mix (Sec. 3.2). In the *others* group, we assess the importance of the mean teacher update ($\beta$) (Sec. 3.3) and of the long-tail weighted sampling $f$ (Sec. 3.1). When the $t \rightarrow s$ branch is active, also the pseudo-label filtering $g$ is utilized, while when $f$ is not active, $\alpha = 0.5$ source classes are selected randomly. With different combinations of components, we obtain different versions of CoSMix which we name CoSMix (a-h). The complete version of our method is named *Full*, where all the components are activated. The Source performance (Source) is also added as a reference for the lower bound. See Tab. 3 for the definition of these different versions.

When the $t \rightarrow s$ branch is used, CoSMix (a) achieves an initial 31.6 mIoU showing that the $t \rightarrow s$ branch provides a significant adaptation contribution over the Source. When we also use the $s \rightarrow t$ branch and the mean teacher $\beta$, CoSMix (b-d) further improve performance achieving a 35.4 mIoU. By introducing local and global augmentations in CoSMix (e-h), we can improve performance up to 39.1 mIoU. The best performance of 40.4 mIoU is achieved with CoSMix Full where all the components are activated.

Table 3: Ablation study of the CoSMix components: mixing strategy ($t \to s$ and $s \to t$), compositional mix augmentations (local $h$ and global $r$), mean teacher update ($\beta$) and, weighted class selection in semantic selection ($f$). Each combination is named with a different version (a-h). Source performance are added as lower bound and highlighted in gray to facilitate the reading.

| CoSMix | mix | | augs | | others | | |
|---|---|---|---|---|---|---|---|
| version | $t \to s$ | $s \to t$ | $h$ | $r$ | $\beta$ | $f$ | **mIoU** |
| Source | - | - | - | - | - | - | 20.7 |
| (a) | ✓ | | | | | | 31.6 |
| (b) | ✓ | | | | ✓ | | 31.9 |
| (c) | ✓ | ✓ | | | | | 35.0 |
| (d) | ✓ | ✓ | | | ✓ | | 35.4 |
| (e) | ✓ | ✓ | ✓ | | ✓ | | 36.8 |
| (f) | ✓ | ✓ | | ✓ | ✓ | | 37.3 |
| (g) | ✓ | ✓ | ✓ | ✓ | ✓ | | 39.0 |
| (h) | ✓ | ✓ | ✓ | ✓ | | ✓ | 39.1 |
| Full | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | **40.4** |

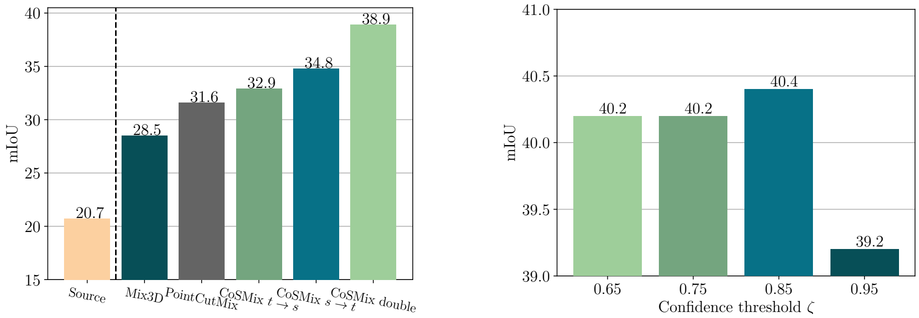
### 5.2 Point Cloud Mix

We compare CoSMix with Mix3D [19] and PointCutMix [39] to show the effectiveness of the different mixing designs. As per our knowledge, Mix3D [19] is the only mixup strategy designed for 3D semantic segmentation, while PointCutMix is the only strategy for mixing portions of different point clouds. We implement Mix3D [19] and PointCutMix [39] based on authors descriptions: we concatenate point clouds (random crops for PointCutMix) of the two domains, i.e., $\mathcal{X}^s$ and $\mathcal{X}^t$, as well as their labels and pseudo-labels, i.e., $\mathcal{Y}^s$ and $\hat{\mathcal{Y}}^t$, respectively. CoSMix double is our two-branch network with sample mixing. For a fair comparison, we deactivate the weighted sampling and the mean teacher update. We keep local and global augmentations ($h$ and $r$) activated.

Fig. 5 shows that Mix3D [19] outperforms the Source model, achieving 28.5 mIoU, while PointCutMix [5] achieves 31.6 mIoU. When we use the $t \to s$ branch alone we can achieve 32.9 mIoU and when we use the $s \to t$ branch alone, CoSMix can further improve the results, achieving 34.8 mIoU. This shows that the supervision from the source to target is effective for adaptation on the target domain. When we use the contribution from both branches simultaneously, CoSMix achieves the best result with 38.9 mIoU.

### 5.3 Pseudo label filtering

We investigate the robustness of CoSMix to increasingly noisier pseudo-labels and study the importance of setting the correct confidence threshold $\zeta$ for pseudo-label distillation in $g$ (Sec. 3.1). We repeat the experiments with a confidence threshold from 0.65 to 0.95 and report the obtained adaptation performance in Fig. 5. CoSMix is robust to noisy pseudo-labels reaching a 40.2 mIoU with the



(a)

(b)

Fig. 5: Comparison of the adaptation performance with (a) different point cloud mix up strategies and (b) on confidence threshold values. (a) Compared to the recent mixing strategy Mix3D [19], our mixing strategy and its variations achieve superior performance. (b) Adaptation results show that $\zeta$ should be set such that to achieve a trade-off between pseudo-label correctness and object completeness.

low threshold of 0.65. The best adaptation performance of 40.4 mIoU is achieved with a confidence threshold of 0.85. By using a high confidence threshold of 0.95 performance is affected reaching 39.2 mIoU. With this configuration, too few pseudo-labels are selected to provide an effective contribution for the adaptation.

## 6 Conclusions

In this paper, we proposed the first UDA method for 3D semantic segmentation based on a novel 3D point cloud mixing strategy that exploits semantic and structural information concurrently. We performed an extensive evaluation in the synthetic-to-real UDA scenario by using large-scale publicly available LiDAR datasets. Experiments showed that our method outperforms all the compared state-of-the-art methods by a large margin. Furthermore, in-depth studies highlighted the importance of each CoSMix component and that our mixing strategy is beneficial for solving domain shift in 3D LiDAR segmentation. Future research directions may include the introduction of self-supervised learning tasks and the extension of CoSMix to source-free adaptation tasks.

**Acknowledgements.** This work was partially supported by OSRAM GmbH, by the Italian Ministry of Education, Universities and Research (MIUR) "Dipartimenti di Eccellenza 2018-2022", by the EU JPI/CH SHIELD project, by the PRIN project PREVUE (Prot. 2017N2RK7K), the EU ISFP PROTECTOR (101034216) project and the EU H2020 MARVEL (957337) project and, it was carried out in the Vision and Learning joint laboratory of FBK and UNITN.

## References

1. A.Geiger, P.Lenz, C.Stiller, R.Urtasun: Vision meets robotics: The kitti dataset. IJRR (2013) 8
2. Alonso, I., Riazuelo, L., Montesano, L., Murillo, A.: 3d-mininet: Learning a 2d representation from point clouds for fast and efficient 3d lidar semantic segmentation. In: IROS (2020) 3
3. Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., Gall, J.: SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In: ICCV (2019) 1, 3, 8
4. Caesar, H., Bankiti, V., Lang, A., Vora, S., Liong, V., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuScenes: A multimodal dataset for autonomous driving. In: CVPR (2020) 1
5. Chen, Y., Hu, V., Gavves, E., Mensink, T., Mettes, P., Yang, P., Snoek, C.G.: Pointmixup: Augmentation for point clouds. In: ECCV (2020) 2, 3, 4, 13
6. Choy, C., Gwak, J., Savarese, S.: 4d spatio-temporal convnets: Minkowski convolutional neural networks. In: CVPR (2019) 1, 3, 4, 9
7. Csurka, G.: Domain adaptation for visual applications: A comprehensive survey. arXiv (2017) 1
8. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: An open urban driving simulator. In: ACRL (2017) 1, 4, 8
9. Gao, L., Zhang, J., Zhang, L., Tao, D.: Dsp: Dual soft-paste for unsupervised domain adaptive semantic segmentation. In: ACMM (2021) 2, 4, 6, 8
10. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The KITTI vision benchmark suite. In: CVPR (2012) 8
11. Graham, B., Engelcke, M., van der Maaten, L.: 3d semantic segmentation with submanifold sparse convolutional networks. In: CVPR (2018) 4
12. Graham, B., van der Maaten, L.: Submanifold sparse convolutional networks. arXiv (2017) 3, 4
13. Hu, Q., Yang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., Trigoni, N., Markham, A.: RandLA-Net: Efficient semantic segmentation of large-scale point clouds. In: CVPR (2020) 3
14. Jadon, S.: A survey of loss functions for semantic segmentation. In: CIBCB (2020) 8, 9
15. Jaritz, M., Vu, T.H., de Charette, R., Wirbel, E., Pérez, P.: xMUDA: Cross-modal unsupervised domain adaptation for 3d semantic segmentation. In: CVPR (2020) 4
16. Langer, F., Milioto, A., Haag, A., Behley, J., Stachniss, C.: Domain transfer for semantic segmentation of LiDAR data using deep neural networks. In: IROS (2021) 4
17. Mancini, M., Akata, Z., Ricci, E., Caputo, B.: Towards recognizing unseen categories in unseen domains. In: ECCV (2020) 2
18. Milioto, A., Vizzo, I., Behley, J., Stachniss, C.: Rangenet++: Fast and accurate lidar semantic segmentation. In: IROS (2019) 3
19. Nekrasov, A., Schult, J., Litany, O., Leibe, B., Engelmann, F.: Mix3D: Out-of-Context Data Augmentation for 3D Scenes. In: 3DV (2021) 2, 3, 4, 7, 11, 13, 14
20. Pan, Y., Gao, B., Mei, J., Geng, S., Li, C., Zhao, H.: SemanticPOSS: A Point Cloud Dataset with Large Quantity of Dynamic Instances. arXiv (2020) 1, 3, 8
21. Qi, C., Su, H., Mo, K., Guibas, L.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: CVPR (2017) 3

22. Qi, C., Yi, L., Su, H., Guibas, L.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. arXiv (2017) 3
23. Rahman, M., Wang, Y.: Optimizing intersection-over-union in deep neural networks for image segmentation. In: ISVC (2016) 9
24. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI (2015) 3
25. Tranheden, W., Olsson, V., Pinto, J., Svensson, L.: Dacs: Domain adaptation via cross-domain mixed sampling. In: WACV (2021) 2, 4
26. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In: CVPR (2017) 8, 10
27. Vu, T., Jain, H., Bucher, M., Cord, M., Pérez, P.: Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. In: CVPR (2019) 8, 10
28. Wang, Q., Dai, D., Hoyer, L., Gool, L.V., Fink, O.: Domain adaptive semantic segmentation with self-supervised depth estimation. In: ICCV (2021) 2
29. Wu, B., Wan, A., Yue, X., Keutzer, K.: Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. In: ICRA (2018) 3, 4
30. Wu, B., Zhou, X., Zhao, S., Yue, X., Keutzer, K.: Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud. In: ICRA (2019) 2, 3, 4
31. Wu, Y., Inkpen, D., A. El-Roby, A.: Dual mixup regularized learning for adversarial domain adaptation. In: ECCV (2020) 4
32. Xiao, A., Huang, J., Guan, D., Zhan, F., Lu, S.: Synlidar: Learning from synthetic lidar sequential point cloud for semantic segmentation. AAAI (2022) 2, 3, 4, 8, 9, 10
33. Xu, M., Zhang, J., Ni, B., Li, T., Wang, C., Tian, Q., Zhang, W.: Adversarial domain adaptation with domain mixup. In: AAAI (2020) 2, 4
34. Yang, Y., Soatto, S.: FDA: Fourier Domain Adaptation for Semantic Segmentation. In: CVPR (2020) 4
35. Yi, L., Gong, B., Funkhouser, T.: Complete & label: A domain adaptation approach to semantic segmentation of lidar point clouds. arXiv (2021) 4
36. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: CutMix: Regularization strategy to train strong classifiers with localizable features. In: ICCV (2019) 2, 4
37. Y.Zhou, O.Tuzel: Voxelnet: End-to-end learning for point cloud based 3d object detection. In: CVPR (2018) 3, 4
38. Zhang, H., Cisse, M., Dauphin, Y., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. In: ICLR (2018) 2, 4
39. Zhang, J., Chen, L., Ouyang, B., Liu, B., Zhu, J., Chen, Y., Meng, Y., Wu, D.: Pointcutmix: Regularization strategy for point cloud classification. arXiv (2021) 2, 4, 13
40. Zhang, Y., Zhou, Z., David, P., Yue, X., Xi, Z., Gong, B., Foroosh, H.: Polarnet: An improved grid representation for online lidar point clouds semantic segmentation. In: CVPR (2020) 3
41. Zhao, S., Wang, Y., Li, B., Wu, B., Gao, Y., Xu, P., Darrell, T., Keutzer, K.: epointda: An end-to-end simulation-to-real domain adaptation framework for lidar point cloud segmentation. arXiv (2020) 2, 4
42. Zhu, X., Zhou, H., Wang, T., Hong, F., Ma, Y., Li, W., Li, H., Lin, D.: Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In: CVPR (2021) 1, 4

43. Zou, L., Tang, H., Chen, K., Jia, K.: Geometry-aware self-training for unsupervised domain adaptation on object point clouds. In: CVPR (2021) 2, 3, 4
44. Zou, Y., Yu, Z., Kumar, B., Wang, J.: Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In: ECCV (2018) 2
45. Zou, Y., Yu, Z., Liu, X., Kumar, B., Wang, J.: Confidence regularized self-training. In: ICCV (2019) 2, 8, 10