xMUDA: Cross-Modal Unsupervised Domain Adaptation for 3D Semantic Segmentation

Maximilian Jaritz\textsuperscript{1,2,3}, Tuan-Hung Vu\textsuperscript{3}, Raoul de Charette\textsuperscript{1}, Émilie Wirbel\textsuperscript{2,3}, Patrick Pérez\textsuperscript{3}

\textsuperscript{1}Inria, \textsuperscript{2}Valeo DAR, \textsuperscript{3}Valeo.ai

Abstract

Unsupervised Domain Adaptation (UDA) is crucial to tackle the lack of annotations in a new domain. There are many multi-modal datasets, but most UDA approaches are uni-modal. In this work, we explore how to learn from multi-modality and propose cross-modal UDA (xMUDA) where we assume the presence of 2D images and 3D point clouds for 3D semantic segmentation. This is challenging as the two input spaces are heterogeneous and can be impacted differently by domain shift. In xMUDA, modalities learn from each other through mutual mimicking, disentangled from the segmentation objective, to prevent the stronger modality from adopting false predictions from the weaker one. We evaluate on new UDA scenarios including day-to-night, country-to-country and dataset-to-dataset, leveraging recent autonomous driving datasets. xMUDA brings large improvements over uni-modal UDA on all tested scenarios, and is complementary to state-of-the-art UDA techniques. Code is available at https://github.com/valeoai/xmuda.

1. Introduction

Three-dimensional scene understanding is required in numerous applications, in particular robotics, autonomous driving and virtual reality. Among the different tasks under concern, 3D semantic segmentation is gaining more and more traction as new datasets are being released \cite{1, 5, 6}. We also observe that previous domain adaptation work focuses on single modality, whereas 3D datasets are often multi-modal, consisting of 3D point clouds and 2D images. While the complementarity between these two modalities is already exploited by both human annotators and learned models to localize objects in 3D scenes \cite{18, 21}, we consider it through a new angle, asking the question: If 3D and 2D data are available in the source and target domain, can we capitalize on multi-modality to address Unsupervised Domain Adaptation (UDA)?
We coin our method cross-modal UDA, ‘xMUDA’ in short, and consider 3 real-to-real adaptation scenarios with different lighting conditions (day-to-night), environments (country-to-country) and sensor setup (dataset-to-dataset). It is a challenging task for various reasons. The heterogeneous input spaces (2D and 3D) make the pipeline complex as it implies to work with heterogeneous network architectures and 2D-3D projections. In fusion, if two sensors register the same scene, there is shared information between both, but each sensor also has private (or exclusive) information. Thanks to the latter, one modality can be stronger than the other in a certain case, but it can be the other way around in another, depending on class, context, resolution, etc. This makes selecting the “best” sensor based on prior knowledge unfeasible. Additionally, each modality can be affected differently by the domain shift. For example, camera is deeply impacted by the day-to-night domain change, while LiDAR is relatively robust to it, as shown in Fig. 1.

In order to address these challenges, we propose a cross-modal UDA (‘xMUDA’) framework where information can be exchanged between 2D and 3D in order to learn from each other for UDA (see right side of Fig. 1). We use a disentangled 2-stream architecture to address the domain gap individually in each modality. Our learning scheme allows robust balancing of the cross-modal and segmentation objective. In addition, xMUDA can be combined with existing uni-modal UDA techniques. In this work, we showcase complementarity to self-training with pseudo-labels. Finally, it is common practice in supervised learning to use feature fusion (e.g., early or late fusion) when multiple modalities are available [8, 18, 27]: our framework can be extended to fusion while maintaining a disentangled cross-modal objective.

Our contributions can be summarized as follows:

- We define new UDA scenarios and propose corresponding splits on recently published 2D-3D datasets.
- We design an architecture that enables cross-modal learning by disentangling private and shared information in 2D and 3D.
- We propose a novel UDA learning scheme where modalities can learn from each other in balance with the main objective. It can be applied on top of state-of-the-art self-training techniques to boost performance.
- We showcase how our framework can be extended to late fusion and produce superior results.

On the different proposed benchmarks we outperform the single-modality state-of-the-art UDA techniques by a significant margin. Thereby, we show that the exploitation of multi-modality for UDA is a powerful tool that can benefit a wide range of multi-sensor applications.

2. Related Work

In this section, rather than thoroughly going through the literature, we review representative works for each focus.

Unsupervised Domain Adaptation. The past few years have seen an increasing interest in unsupervised domain adaptation techniques for complex perception tasks like object detection and semantic segmentation. Under the hood of such methods lies the same spirit of learning domain-invariant representations, i.e., features coming from different domains should introduce insignificant discrepancy. Some works promote adversarial training to minimize source-target distribution shift, either on pixel- [11], feature- [12] or output-space [26, 28]. Revisited from semi-supervised learning [14], self-training with pseudo-labels has also been recently proven effective for UDA [16, 34].

While most existing works consider UDA in the 2D world, very few tackle the 3D counterpart. Wu et al. [32] adopted activation correlation alignment [20] for UDA in 3D segmentation from LiDAR point clouds. In this work, we investigate the same task, but differently: our system operates on multi-modal input data, i.e., RGB + LiDAR.

To the best of our knowledge, there are no previous UDA works in 2D/3D semantic segmentation for multi-modal scenarios. Only some consider the extra modality, e.g. depth, solely available at training time on source domain and leverage such privileged information to boost adaptation performance [15, 29]. Otherwise, we here assume all modalities are available at train and test time on both source and target domains.

Multi-Modality Learning. In a supervised setting, performance can naturally be improved by fusing features from multiple sources. The geometrically simplest case is RGB-Depth fusion with dense pixel-to-pixel correspondence for 2D segmentation [8, 27]. It is harder to fuse a 3D point cloud with a 2D image, because they live in different metric spaces. One solution is to project 2D and 3D features into a ‘bird eye view’ [17, 18] or ‘LiDAR front view’ [19] for the task of object detection. Another possibility is to lift 2D features from multi-view images to the 3D point cloud to enable joint 2D-3D processing for 3D semantic segmentation [3, 13, 24]. We are closer to the last series of works: we share the same goal of 3D semantic segmentation. However, we focus on how to exploit multi-modality for UDA instead of supervised learning and only use single view images and their corresponding point clouds.

3D networks for semantic segmentation. While images are dense tensors, 3D point clouds can be represented in multiple ways which leads to competing network families that evolve in parallel. Voxels are very similar to pixels, but very memory intense as most of them are empty. Graham et al. [7] and similar implementation [4] address this
Figure 2: **Overview of our xMUDA framework for 3D semantic segmentation.** The architecture comprises a 2D stream which takes an image as input and uses a U-Net-style 2D ConvNet [23], and a 3D stream which takes the point cloud as input and uses a U-Net-Style 3D SparseConvNet [7]. Feature outputs of both streams have same length \( N \), equal to the number of 3D points. To achieve that, we project the 3D points into the image and sample the 2D features at the corresponding pixel locations. The 4 segmentation outputs consist of the main predictions \( P_{2D}, P_{3D} \) and the mimicry predictions \( P_{2D\rightarrow3D}, P_{3D\rightarrow2D} \). We transfer knowledge across modalities using KL divergence, \( D_{KL}(P_{3D}||P_{2D\rightarrow3D}) \), where the objective of the 2D mimicry head is to estimate the main 3D output and vice versa, \( D_{KL}(P_{2D}||P_{3D\rightarrow2D}) \).

problem by using hash tables to convolve only on active voxels. This allows for very high resolution with typically only one point per voxel. Point-based networks perform computation in continuous 3D space and can thus directly accept point clouds as input. PointNet++ [22] uses point-wise convolution, max-pooling to compute global features and local neighborhood aggregation for hierarchical learning akin to CNNs. Many improvements have been proposed in this direction, such as continuous convolutions [30] and deformable kernels [25]. Graph-based networks convolve on the edges of a point point cloud [31]. In this work, we select SparseConvNet [7] as 3D network which is the state-of-the-art on the ScanNet benchmark [5].

### 3. xMUDA

The aim of cross-modal UDA (xMUDA) is to exploit multi-modality by enabling controlled information exchange between modalities so that they can learn from each other. This is achieved through letting them mutually mimic each other’s outputs, so that they can both benefit from their counterpart’s strengths.

Specifically, we investigate xMUDA using point cloud (3D modality) and image (2D modality) on the task of 3D semantic segmentation. An overview is depicted in Fig. 2. We first describe the architecture in Sec. 3.1, our learning scheme in Sec. 3.2, and later showcase its extension to the special case of fusion.

In the following, we consider a source dataset \( S \), where each sample consists of 2D image \( x_{s}^{2D} \), 3D point cloud \( x_{s}^{3D} \) and 3D segmentation labels \( y_{s}^{3D} \) as well as a target dataset \( T \), lacking annotations, where each sample only consists of image \( x_{t}^{2D} \) and point cloud \( x_{t}^{3D} \). Images \( x^{2D} \) are of spatial size \((H, W, 3)\) and point clouds \( x^{3D} \) of spatial size \((N, 3)\), with \( N \) the number of 3D points in the camera field of view.

#### 3.1. Architecture

To allow cross-modal learning, it is crucial to extract features specific to each modality. Opposed to 2D-3D architectures where 2D features are lifted to 3D [18], we use a 2-stream architecture with independent 2D and 3D branches that do not share features (see Fig. 2).

We use SparseConvNet [7] for 3D and a modified version of U-Net [23] with ResNet34 [9] for 2D. Even though each stream has a specific network architecture, it is important that the outputs are of same size to allow cross-modal learning. Implementation details are provided in Sec. 4.2.

**Dual Segmentation Head.** We call segmentation head the last linear layer in the network that transforms the output features into logits followed by a softmax function to produce the class probabilities. For xMUDA, we establish a link between 2D and 3D with a ‘mimicry’ loss between the
output probabilities, i.e., each modality should predict the other modality’s output. This allows us to explicitly control the cross-modal learning.

In a naive approach, each modality has a single segmentation head and a cross-modal optimization objective aligns the outputs of both modalities. Unfortunately, this leads to only using information that is shared between the two modalities, while discarding private information that is exclusive to each sensor (more details in the ablation study in Sec. 5.1). This is an important limitation, as we want to leverage both private and shared information, in order to obtain the best possible performance.

To preserve private information while benefiting from shared knowledge, we introduce an additional segmentation head to uncouple the mimicry objective from the main segmentation objective. This means that the 2D and 3D streams both have two segmentation heads: one main head for the best possible prediction, and one mimicry head to estimate the other modality’s output.

The outputs of the 4 segmentation heads (see Fig. 2) are of size \((N, C)\), where \(C\) is equal to the number of classes such that we obtain a vector of class probabilities for each 3D point. The two main heads produce the best possible predictions, \(P_{2D}\) and \(P_{3D}\) respectively for each branch. The two mimicry heads estimate the other modality’s output: 2D estimates 3D \((P_{2D} \rightarrow 3D)\) and 3D estimates 2D \((P_{3D} \rightarrow 2D)\).

3.2. Learning Scheme

The goal of our cross-modal learning scheme is to exchange information between the modalities in a controlled manner to teach them to be aware of each other. This auxiliary objective can effectively improve the performance of each modality and does not require any annotations which enables its use for UDA on target dataset \(T\). In the following we define the basic supervised learning setup, our cross-modal loss \(L_{XM}\), and the additional pseudo-label learning method. The loss flows are depicted in Fig. 3a.

**Supervised Learning.** The main goal of 3D segmentation is learned through cross-entropy in a classical supervised fashion on the source data. We can write the segmentation loss \(L_{seg}\) for each network stream (2D and 3D) as:

\[
L_{seg}(x_s, y_s^{3D}) = -\frac{1}{N} \sum_{n=1}^{N} \sum_{c=1}^{C} y_s^{(n,c)} \log P_x^{(n,c)},
\]

where \(x_s\) is either \(x_s^{2D}\) or \(x_s^{3D}\).

**Cross-Modal Learning.** The objective of unsupervised learning across modalities is twofold. Firstly, we want to transfer knowledge from one modality to the other on the target dataset. For example, let one modality be sensitive and the other more robust to the domain shift, then the robust modality should teach the sensitive modality the correct class in the target domain where no labels are available. Secondly, we want to design an auxiliary objective on source and target, where the task is to estimate the other modality’s prediction. By mimicking not only the class with maximum probability, but the whole distribution, more information is exchanged, leading to softer labels.

We choose KL divergence for the cross-modal loss \(L_{XM}\) and define it as follows:

\[
L_{XM}(x) = D_{KL}(P^{(n,c)}_x || Q^{(n,c)}_x) \tag{2}
\]

\[
= -\frac{1}{N} \sum_{n=1}^{N} \sum_{c=1}^{C} P^{(n,c)}_x \log \frac{P^{(n,c)}_x}{Q^{(n,c)}_x} \tag{3}
\]

with \((P, Q) \in \{ (P_{2D}, P_{3D \rightarrow 2D}), (P_{3D}, P_{2D \rightarrow 3D}) \}\) where \(P\) is the target distribution from the main prediction which is to be estimated by the mimicking prediction \(Q\). This loss is applied on the source and the target domain as it does not require ground truth labels and is the key to our proposed domain adaptation framework. For source, \(L_{XM}\) can be seen as an auxiliary mimicry loss in addition to the main segmentation loss \(L_{seg}\).
The complete objective for each network stream (2D and 3D) is the combination of the segmentation loss $L_{\text{seg}}$ on source and the cross-modal loss $L_{\text{CM}}$ on source and target:

$$
\min_{\theta} \left[ \frac{1}{|S|} \sum_{x_s \in S} \left( L_{\text{seg}}(x_s, y_s^{3D}) + \lambda_s L_{\text{CM}}(x_s) \right) + \frac{1}{|T|} \sum_{x_t \in T} \lambda_t L_{\text{CM}}(x_t) \right],
$$

where $\lambda_s, \lambda_t$ are hyperparameters to weight $L_{\text{CM}}$ on source and target respectively and $\theta$ are the network weights of either the 2D or the 3D stream.

There are parallels between the cross-modal learning and model distillation which also adopts KL divergence as mimicry loss, but with the goal to transfer knowledge from a large network to a smaller one in a supervised setting [10]. Recently Zhang et al. introduced Deep Mutual Learning [33] where an ensemble of uni-modal networks are jointly trained to learn from each other in collaboration. Though to some extent, our cross-modal learning is of similar nature to those strategies, we tackle a different distillation angle, i.e. across modalities (2D/3D) and not in the supervised, but in the UDA setting.

### Additional self-training with Pseudo-Labels

Cross-modal learning is complementary to pseudo-labeling [14] used originally in semi-supervised learning and recently in UDA [16, 34]. In details, once having optimized a model with Eq. 4, we extract pseudo-labels offline, selecting highly confident labels based on the predicted class probability. Then, we train again from scratch using the produced pseudo-labels for an additional segmentation loss on the target training set. The optimization problem writes:

$$
\min_{\theta} \left[ \frac{1}{|S|} \sum_{x_s \in S} \left( L_{\text{seg}}(x_s, \hat{y}_s^{3D}) + \lambda_s L_{\text{CM}}(x_s) \right) + \frac{1}{|T|} \sum_{x_t \in T} \lambda_t L_{\text{CM}}(x_t) \right],
$$

where $\lambda_{PL}$ is weighting the pseudo-label segmentation loss and $\hat{y}^{3D}$ are the pseudo-labels. For clarity, we will refer to the xMUDA variant that uses additional self-training with pseudo-labels as xMUDA$_{PL}$.

### 3.3. Discussion

A central contribution of our work is the formulation of cross-modal learning via KL-divergence minimization in multi-modal scenarios which helps us not only to benefit from multiple sensors but also to mitigate domain shift. Indeed, as computing KL-divergence between 2D and 3D predictions does not require ground-truth, our learning scheme allows extra regularization on the target set—bringing adaptation effects. Fig. 3b visualizes the 4 data subsets considered in our task and shows which dimension different UDA techniques operate on. Opposed to previous UDA methods which only act on a single modality, xMUDA introduces a new way of cross-modal self-training and is thus orthogonal and complementary to existing adaptation techniques.

### 4. Experiments

#### 4.1. Datasets

To evaluate xMUDA, we identified 3 real-to-real adaptation scenarios. In the day-to-night case, LiDAR has a small domain gap, as it is an active sensor sending out laser beams which are mostly invariant to lighting conditions. In contrast, camera has a large domain gap as its passive sensing suffers from lack of light sources, leading to drastic changes in object appearance. The second scenario is country-to-country adaptation, where the domain gap can be larger for LiDAR or camera: for some classes the 3D shape might change more than the visual appearance or vice versa. The third scenario, dataset-to-dataset, comprises changes in the sensor setup, such as camera optics, but most importantly a higher LiDAR resolution on target. 3D networks are sensitive to varying point cloud density and the image could help to guide and stabilize adaptation.

We leverage recently published autonomous driving datasets nuScenes [2], A2D2 [6] and SemanticKITTI [1] in which LiDAR and camera are synchronized and calibrated allowing to compute the projection between a 3D point and its corresponding 2D image pixel. The chosen datasets contain 3D annotations. For simplicity and consistency across datasets, we only use the front camera image and the LiDAR points that project into it.

For nuScenes, the annotations are 3D bounding boxes and we obtain the point-wise labels for 3D semantic segmentation by assigning the corresponding object label if a point lies inside a 3D box; otherwise the point is labeled as background. We use the meta data to generate the splits for two UDA scenarios: Day/Night and USA/Singapore.

A2D2 and SemanticKITTI provide segmentation labels. For UDA, we define 10 shared classes between the two datasets. The LiDAR setup is the main difference: in A2D2, there are 3 LiDARs with 16 layers which generate a rather sparse point cloud and in SemanticKITTI, there is one high-resolution LiDAR with 64 layers.

We provide the data split details in the supplementary.

### 4.2. Implementation Details

#### 2D Network

We use a modified version of U-Net [23] with a ResNet34 [9] encoder where we add dropout after the 3rd and 4th layer and initialize with ImageNet pretrained weights provided by PyTorch. In the decoder, each layer consists of a transposed convolution, concatenation with encoder features of same resolution (skip connection) and another convolution to mix the features. The network takes
an image $x^{2D}$ as input and produces an output feature map with equal spatial dimensions $(H, W, F_{2D})$, where $F_{2D}$ is the number of feature channels. In order to lift the 2D features to 3D, we sample them at sparse pixel locations where the 3D points project into the feature map, and obtain the final two-dimensional feature matrix $(N, F_{2D})$.

**3D Network.** For SparseConvNet [20] with network. We use a batch size of 8, the Adam optimizer accumulate gradients computed on source and target batch. We jointly train the 2D and 3D stream and at each iteration, 10 at 80k and 90k iterations; the training finishes at 100k.

**Training.** For data augmentation we employ horizontal flipping and color jitter in 2D, and x-axis flipping, scaling and rotation in 3D. Due to the wide angle image in SemanticKITTI, we crop a fixed size rectangle randomly on the horizontal image axis to reduce memory during training. Log-smoothed class weights are used in all experiments to address class imbalance. For the KL divergence for the cross-modal loss in PyTorch, we detach the target variable to only backpropagate in either the 2D or the 3D network. We use a batch size of 8, the Adam optimizer with $\beta_1 = 0.9, \beta_2 = 0.999$, and an iteration based learning schedule where the learning rate of 0.001 is divided by 10 at 80k and 90k iterations; the training finishes at 100k. We jointly train the 2D and 3D stream and at each iteration, accumulate gradients computed on source and target batch. All trainings fit into a single GPU with 11GB RAM.

For xMUDA, we train with Eq. 4, where we apply the segmentation loss using ground truth labels on source and cross-modal loss on source and target. For xMUDA$_{PL}$, we generate pseudo-labels offline as in [16] with the previously trained xMUDA model and train again from scratch, now with additional segmentation loss on target with pseudo-labels (Eq. 5). Note, that we do not select the best weights on the validation set, but rather use the last checkpoint to generate the pseudo-labels in order to prevent any supervised learning signal. The 2D and 3D network are trained jointly and optimized on source and target at each iteration.

### 4.3. Main Experiments

We evaluate our method on the 3 proposed cross-modal UDA scenarios and compare against uni-modal UDA methods: Deep logCORAL [20], entropy minimization (MinEnt) [28] and pseudo-labeling (PL) [16]. Regarding PL, we apply [16] as follows: we generate pseudo-labels offline with a first training without UDA, and discard unconfident labels through class-wise thresholding. Then, we run a second training from scratch adding PL loss on target. The image-2-image translation part was excluded due to its instability, high training complexity and incompatibility with LiDAR data, thus limiting reproducibility. Regarding the other two uni-modal techniques, we adapt the published implementations to our settings. For all, we searched for the best respective hyperparameters.

We report mean Intersection over Union (mIoU) results for 3D segmentation in Tab. 1 on the target test set for the 3 UDA scenarios. We evaluate on the test set using the checkpoint that achieved the best score on the validation set. In addition to the scores of the 2D and 3D model, we show the ensembling result (‘softmax avg’) which is obtained by taking the mean of the predicted 2D and 3D probabilities after softmax. The baseline is trained on source only and the oracle on target only, except the Day/Night oracle, where we used batches of 50%/50% Day/Night to prevent overfitting. The uni-modal UDA baselines [16, 20, 28] are applied separately on each modality.

xMUDA – using the cross-modal loss but not PL – brings a significant adaptation effect on all 3 UDA scenarios compared to ‘Baseline (source only)’ and often outperforms the uni-modal UDA baselines. We observe, that xMUDA consistently improves both modalities (2D and 3D), i.e. even the strong modality can learn from the weaker one. xMUDA$_{PL}$ achieves the best score everywhere with the only exception of Day/Night softmax avg. Further, cross-modal learning and self-training with pseudo-labels (PL) are complementary as their combination in xMUDA$_{PL}$ consistently

### Table 1: mIoU on the respective target sets for 3D semantic segmentation in different cross-modal UDA scenarios. We report the result for each network stream (2D and 3D) as well as the ensembling result (‘softmax avg’).

<table>
<thead>
<tr>
<th>Method</th>
<th>USA/Singapore</th>
<th>Day/Night</th>
<th>A2D2/SemanticKITTI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2D</td>
<td>3D</td>
<td>softmax avg</td>
</tr>
<tr>
<td>Baseline (source only)</td>
<td>53.4</td>
<td>46.5</td>
<td>61.3</td>
</tr>
<tr>
<td>Deep logCORAL [20]</td>
<td>52.6</td>
<td>47.1</td>
<td>59.1</td>
</tr>
<tr>
<td>MinEnt [28]</td>
<td>53.4</td>
<td>47.0</td>
<td>59.7</td>
</tr>
<tr>
<td>PL [16]</td>
<td>55.5</td>
<td>51.8</td>
<td>61.5</td>
</tr>
<tr>
<td>xMUDA</td>
<td>59.3</td>
<td>52.0</td>
<td>62.7</td>
</tr>
<tr>
<td>xMUDA$_{PL}$</td>
<td>61.1</td>
<td>54.1</td>
<td>63.2</td>
</tr>
<tr>
<td>Oracle</td>
<td>66.4</td>
<td>63.8</td>
<td>71.6</td>
</tr>
</tbody>
</table>

* Trained with batch size 6 instead of 8 to fit into GPU memory.
Figure 4: Architectures for fusion. (a) In Vanilla Fusion the 2D and 3D features are concatenated, fed into a linear layer with ReLU to mix the features and followed by another linear layer and softmax to obtain a fused prediction $P_{\text{fuse}}$. (b) In xMUDA Fusion, we add two uni-modal outputs $P_{2D\rightarrow\text{fuse}}$ and $P_{3D\rightarrow\text{fuse}}$ that are used to mimic the fusion output $P_{\text{fuse}}$. yea yields a higher score than each separate technique.

Qualitative results are presented in Fig. 6 and show the versatility of xMUDA across all proposed UDA scenarios. We provide additional qualitative results in the supplementary and a video of the A2D2 to SemanticKITTI scenario at http://tiny.cc/xmuda.

4.4. Extension to Fusion

In Sec. 4.3 we show how each 2D and 3D modality can be improved with xMUDA. However, can we obtain even better results with fusion?

A common fusion architecture is late fusion where the features from different sources are concatenated (see Fig. 4a). However, when merging the main 2D/3D branches into a unique fused head, we can no longer apply cross-modal learning (as in Fig. 5a). To address this problem, we propose ‘xMUDA Fusion’ where we add an additional segmentation head to both 2D and 3D network streams prior to the fusion layer, with the purpose of mimicking the central fusion head (see Fig. 4b). Note that this idea could also be applied on top of other fusion architectures.

In Tab. 2 we show results for different fusion approaches where we specify which architecture was used (Vanilla late fusion from Fig. 4a or xMUDA Fusion from Fig. 4b). While ‘xMUDA$_{PL}$ Fusion’ outperforms all other uni-modal baselines, ‘xMUDA’ already achieves better performances than ‘Deep logCORAL’ and ‘MinEnt’.

5. Ablation Studies

5.1. Segmentation Heads

In the following we justify our design choice of two segmentation heads per modality stream as opposed to a single one in a naive approach (see Fig. 5a). In the single head architecture the mimicking could also be applied on top of other fusion architectures.

Figure 5: Single vs. Dual segmentation head. (a) Main and mimicry prediction are not uncoupled as in xMUDA of Fig. 2. (b) Curves of mIoU w.r.t. $\lambda_t$ of single- vs. dual-head architectures. USA/Singapore scenario.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>mIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline (source only)</td>
<td>Vanilla 59.9</td>
</tr>
<tr>
<td>Deep logCORAL [29]</td>
<td>Vanilla 58.2</td>
</tr>
<tr>
<td>MinEnt [28]</td>
<td>Vanilla 60.8</td>
</tr>
<tr>
<td>PL [16]</td>
<td>Vanilla 65.2</td>
</tr>
<tr>
<td>Distillation</td>
<td>Vanilla 65.8</td>
</tr>
<tr>
<td>xMUDA Fusion</td>
<td>xMUDA 61.9</td>
</tr>
<tr>
<td>xMUDA$_{PL}$ Fusion</td>
<td>xMUDA 66.6</td>
</tr>
<tr>
<td>Oracle</td>
<td>xMUDA 72.2</td>
</tr>
</tbody>
</table>

Table 2: mIoU for fusion methods, USA/Singapore scenario. In ‘Distillation’ we use the xMUDA$_{PL}$ model of the main experiments reported in Tab. 1 to generate pseudo-labels from the softmax average and use those to train the Vanilla Fusion network.
head is more robust because it disentangles the segmentation from the mimicking objective.

5.2. Cross-Modal Learning on Source

In (4), cross-modal loss $L_{XM}$ is applied on source and target, although we already have supervised segmentation loss $L_{seg}$ on source. We observe a gain of 4.8 mIoU on 2D and 4.4 on 3D when adding $L_{XM}$ on source as opposed to applying it on target only. This shows that it is important to train the mimicking head on source, stabilizing the predictions, which can be exploited during adaptation on target.

5.3. Cross-Modal Learning for Oracle Training

We have shown that cross-modal learning is very effective for UDA. However, it can also be used in a purely supervised setting. When training the oracle with cross-modal loss $L_{XM}$, we can improve over the baseline, see Tab. 3. We conjecture that $L_{XM}$ is a beneficial auxiliary loss and can help to regularize training and prevent overfitting.

6. Conclusion

We propose xMUDA, Cross-Modal Unsupervised Domain Adaptation, where modalities learn from each other to improve performance on the target domain. For cross-modal learning we introduce mutual mimicking between the modalities, achieved through KL divergence. We design an architecture with separate main and mimicking head to disentangle the segmentation from the cross-modal learning objective. Experiments on 3D semantic segmentation on new UDA scenarios using 2D/3D datasets, show that xMUDA largely outperforms uni-modal UDA and is complementary to the pseudo-label strategy. An analog performance boost is observed on fusion.

We think that cross-modal learning could be useful in a wide variety of settings and tasks, not limited to UDA. Particularly, it should be beneficial for supervised learning and other modalities than image and point cloud.
References


[16] Yunseng Li, Lu Yuan, and Nuno Vasconcelos. Bidirectional learning for domain adaptation of semantic segmentation. In CVPR, 2019. 1, 2, 5, 6, 7


[26] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In CVPR, 2018. 2


[31] 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. 3
