Exploiting the Complementarity of 2D and 3D Networks to Address Domain-Shift in 3D Semantic Segmentation

Adriano Cardace  Pierluigi Zama Ramirez  Samuele Salti  Luigi Di Stefano
Department of Computer Science and Engineering (DISI)
University of Bologna, Italy
{adriano.cardace2}@unibo.it

3D Network

2D Network

Point Cloud

RGB Image

Effective 3D Receptive Field

Effective 2D Receptive Field

Complementary Information

Corresponding Point n

Re-Projected into 3D

Figure 1. 3D (top) and 2D (bottom) networks processing point clouds and images of the same scene extract features that contain complementary information. Indeed, 2D and 3D effective receptive fields [34] centered on a point \( n \) focus on different portions of the scene, i.e., 2D or 3D neighborhoods respectively. Thus, corresponding features have different content by construction. We exploit this property to reduce the domain gap in 3D semantic segmentation.

Abstract

3D semantic segmentation is a critical task in many real-world applications, such as autonomous driving, robotics, and mixed reality. However, the task is extremely challenging due to ambiguities coming from the unstructured, sparse, and uncolored nature of the 3D point clouds. A possible solution is to combine the 3D information with others coming from sensors featuring a different modality, such as RGB cameras. Recent multi-modal 3D semantic segmentation networks exploit these modalities relying on two branches that process the 2D and 3D information independently, striving to maintain the strength of each modality. In this work, we first explain why this design choice is effective and then show how it can be improved to make the multi-modal semantic segmentation more robust to domain shift. Our surprisingly simple contribution achieves state-of-the-art performances on four popular multi-modal unsupervised domain adaptation benchmarks, as well as better results in a domain generalization scenario.

1. Introduction

3D semantic segmentation is a critical task in many real-world applications, such as autonomous driving and robotics. It involves assigning labels to 3D points in a point cloud based on their semantic meaning. However, this task can be extremely challenging due to ambiguities coming from the unstructured, sparse, and uncolored nature of the 3D point clouds. Fortunately, combining 3D information with others coming from sensors with a different modality, such as RGB cameras, can help to address these shortcomings. Indeed, by combining multi-modal data, we can leverage the strengths of each modality to produce more comprehensive and accurate segmentations. For example, in autonomous driving scenarios, RGB cameras and LiDARs are commonly used together. RGB cameras provide dense, colored, and structured information, but they may fail in dark lighting conditions. On the other hand, LiDARs are robust to light conditions, but the point clouds present the problems highlighted above. By combining these two modalities, we can obtain a richer understanding of the environment and make more robust and precise 3D segmentations.
Several recent approaches for multi-modal 3D semantic segmentation [24, 25, 39, 51, 68] leverage a peculiar two-branch 2D-3D architecture, in which images are processed by a 2D convolutional network, e.g., ResNet [18], while point clouds by a 3D convolutional backbone, e.g., SparseConvNet [15]. By processing each modality independently, each of the two branches focuses on extracting features from its specific signal (RGB colors or 3D structure information) that can be fused effectively due to their inherent complementarity in order to produce a better segmentation score. Indeed, averaging logits from the two branches provides often an improvement in performance, e.g., a mIoU gain from 2% to 4% in almost all experiments in [24]. Although we agree that each modality embodies specific information, such as color for images and 3D coordinates for point clouds, we argue that the complementarity of the features extracted by the two branches is also tightly correlated to the different information processing machinery, i.e., 2D and 3D convolutions, which makes networks focusing on different areas of the scene with different receptive fields. Indeed, in Fig. 1, given a point belonging to the red car, we visualize the effective receptive field [34] of the 2D and 3D networks (red ellipses). As we can clearly see from the receptive fields in the right part of the figure, the features extracted by the 3D network mainly leverage points in a 3D neighborhood, i.e., include points of the car surface. In contrast, the features extracted by the 2D network look at a neighborhood in the 2D projected space, and thus they depend also on pixels of the building behind the car, which are close in image space but far in 3D. We argue that this is one of the main reasons why the features from the two branches can be fused so effectively. Based on the above intuition, we propose to feed 3D and RGB signals to both networks as this should not hinder the complementarity of their predictions, with the goal of making the network more robust to the change of distributions between the training and the test scenarios. This problem is typically referred to Domain shift in the literature. Feeding both branches with both modalities would make: i) the 2D network more robust to domain shifts, as depth information (z coordinates of point clouds projected into image space) is more similar across different domains, as shown in several papers [6, 9, 44, 48, 61, 65]; ii) the 3D network more capable of adapting to new domains thanks to RGB information associated with each point which allows learning better semantic features for the target domain, when this is available, using Unsupervised Domain Adaptation (UDA) approaches. Thus, we propose a simple architecture for multi-modal 3D semantic segmentation consisting of a 2D-3D architecture with each branch fed with both RGB and 3D information. Despite its simplicity, our proposal achieves state-of-the-art results in multi-modal UDA benchmarks, surpassing competitors by large margins, as well as significantly better domain generalization compared to a standard 2D-3D architecture [25]. Code available at https://github.com/CVLAB-Unibo/MM2D3D. Our contributions are:

- shining a light on the intrinsic complementarity of recent multi-modal 3D semantic segmentation networks based on 2D-3D branches;
- proposing a simple yet remarkably effective baseline that injects depth cues into the 2D branch and RGB colors into the 3D branch while preserving the complementarity of predictions;
- our network achieves state-of-the-art results in popular UDA benchmarks for multi-modal 3D semantic segmentation and surpasses standard 2D-3D architectures in domain generalization.

2. Related works

Point Cloud Semantic Segmentation. 3D data can be represented in several ways such as point clouds, voxels, and meshes, each with its pros and cons. Similarly to pixels in 2D, voxels represent 3D data as a discrete grid of the 3D space. This representation allows using convolutions as done for images. However, performing a convolution over the whole 3D space is memory intense, and it does not consider that many voxels are usually empty. Some 3D CNNs [45, 54] rely on OctTree [35] to reduce the memory footprint but without addressing the problem of manifold dilation. SparseConvNet [15] and similar implementations [11] address this problem by using hash tables to convolve only on active voxels, allowing the processing of high-resolution point clouds with only one point per voxel. Aside from cubic discretization, some approaches [75, 77] employ cylindrical voxels. Other methods address the problem with sparse point-voxel convolutions [53]. Differently, point-based networks process directly each point of a point cloud. PointNet++ [42] extract features from each point, and then extract global and local features by means of max-pooling in a hierarchical way. Many improvements have been proposed in this direction, such as continuous convolutions [55], deformable kernels [55] or lightweight alternatives [25]. In this work, we select SparseConvNet [15] as our 3D network as done by other works in the field [24, 39, 51, 68] since it is suitable for 3D semantic segmentation of large scenes.

Multi-Modal Learning. Exploiting multiple modalities to learn more robust and performant networks is a well-studied field in the literature [1, 38]. Among them, several approaches address the problem of semantic segmentation exploiting RGB and 3D structure information, either with the final goal of segmenting images, e.g., RGB-D networks [17, 58] or point clouds, e.g., LiDAR + RGB ap-
Figure 2. Framework overview. The RGB image and the sparse depth map obtained from the projection of the corresponding point cloud are fed to a custom 2D architecture to extract point-wise features. The same point cloud and sampled colors from the RGB image are given in input to the 3D Network. Then, two main classifiers output the main predictions to be used at test time. Moreover, two auxiliary classifiers are used at training time only to allow the exchange of information across branches.

proaches [16, 28, 68]. To speed-up research in this promising field, several datasets have been collected [3, 5, 13, 14] with 3D point clouds, images, and annotations for tasks such as 3D object detection or 3D semantic segmentation. Recently, some multi-modal methods [24, 39, 51, 68] show that a framework composed of a 2D and a 3D network can obtain very good performance in popular 3D segmentation benchmarks when averaging the scores coming from the two branches. This result is ascribed to the complementarity of the predictions due to the different modalities processed by each branch (either RGB or point clouds). In this paper, we analyze the improvement obtained by fusing the scores, and we argue that it mainly depends on the fact that the two networks extract complementary features because of the different receptive fields of the 2D and 3D networks. Based on this intuition, we propose a simple yet effective modification of the 2D-3D framework, that consists of providing both modalities as input to both branches.

Unsupervised Domain Adaptation. Unsupervised Domain Adaptation is the research field that investigates how to transfer knowledge learned from a source annotated domain to a target unlabelled domain [63]. In the last few years, several UDA approaches have been proposed for 2D semantic segmentation, using strategies such as style-transfer [8, 10, 19, 27, 31, 37, 41, 67, 71, 73], adversarial training to learn domain-invariant representations [4, 20, 56, 57, 59, 62, 64, 69, 74] or self-training [7, 21, 22, 72, 78]. Recently, some works demonstrated the effectiveness of using depth information to boost UDA for 2D semantic segmentation [6, 9, 43, 44, 48, 61, 65]. In our work, we take inspiration from these findings, and we feed the projected point cloud in input to also to the 2D network, considering depth as a rich source of information robust to the domain shift. Recently some works address UDA also for semantic segmentation of point clouds [2, 26, 29, 40, 46, 49, 66, 70, 76]. Very recently, some works have addressed the challenging multi-modal 3D semantic segmentation task [24, 25, 39, 51]. XMUDA [24] is the first work that focuses on UDA in the above setting, it defines a new benchmark and a baseline approach to adapt to a new target domain with an unsupervised cross-modal loss. [25] extend it, by proposing a more solid and comprehensive benchmark. DsCML [39] also extends XMUDA deploying adversarial training to align features across modalities and domains. In our work, we address the same multi-modal UDA scenarios introduced in [25], and we propose a simple yet effective architecture that is more robust to domain shift and can be adapted to new unlabelled target domains. Our framework, depicted in Fig. 2.

3. Method

Setup and Notation. We define input source samples \( \{x_{s}^{2D}, x_{s}^{3D}\} \in S \) and target samples \( \{x_{t}^{2D}, x_{t}^{3D}\} \in T \), with \( x_{s}^{2D} \) being the 2D RGB image and \( x_{s}^{3D} \) the corresponding point cloud, with 3D points in the camera reference frame. Note that \( x_{t}^{3D} \) contains only points visible from the RGB camera, assuming that the calibration of the two sensors is available for both domains and does not change over time. We assume the availability of annotations \( y_{s}^{3D} \) only for the source domain for each 3D point. When tackling the UDA scenario, we also have at our disposal the unlabeled samples from the target domain. Our goal is to obtain a point-wise prediction \( \hat{y}_{t} \) for each 3D point. When addressing UDA, we also have at our disposal the unlabeled samples from the target domain. Our goal is to obtain a point-wise prediction \( \hat{y}_{t} \) for each 3D point. When addressing UDA, we also have at our disposal the unlabeled samples from the target domain.

3.1. Base 2D/3D Architecture

We build our contributions upon the two independent branches (2D and 3D) architecture proposed in [25]. The
2D branch processes images to obtain a pixel-wise prediction given $x^{2D}$ and it consists of a standard 2D U-Net [47]. On the other hand, the 3D branch takes in input point clouds to estimate the class of each point of $x^{3D}$ and it is implemented as a 3D sparse convolutional network [15]. Thanks to the fact that 2D-3D correspondences are known, 3D points can be projected into the image plane to supervise the 2D branch, as supervision is provided only for the sparse 3D points. We denote the 3D semantic labels projected into 2D with the symbol $y^{3D \rightarrow 2D}$. As argued by [25], such design choice allows one to take advantage of the strengths of each input modality, and final predictions can be obtained by averaging the outputs of the two branches to achieve an effective ensemble. In our work, we adopt the same framework, and we give an intuitive explanation of why this design choice is particularly effective. In particular, we reckon that the two predictions are complementary not only for the input signals being different but also for the fact the two branches focus on different things to determine their final predictions. Indeed, 3D convolutions produce features by looking at points that are close in the 3D space, while the 2D counterparts focus on neighboring pixels in the 2D image plane. Therefore, given corresponding 2D and 3D points, the two mechanisms implicitly produce features containing complementary information. In the right part of Fig. 1 we visualize the Effective Receptive Fields (ERF) [34] of a 2D U-Net with backbone ResNet34 [18] and of a 3D U-Net with backbone SparseConvNet [15]. It is worth highlighting that we do not focus on the theoretical yet on the effective receptive field, which is computed by analyzing the real contribution of each input point to the final prediction (the hotter the color intensity in the visualization, the larger the point contribution). Comparing the re-projected 2D ERF to the 3D ERF we can clearly see that the 2D network focuses on sparse 3D regions, i.e., from the car to the building in the background, while the 3D counterpart reasons on a local 3D neighborhood (only car points). With this intuition in mind, we argue that by feeding the RGB signal to the 3D network, and the 3D information to the 2D backbone, we would still obtain complementary features that can be effectively fused together. Moreover, it is well-known that employing depth information as input to 2D segmentation networks can make it more robust to domain shift [6, 17]. At the same time, we posit that the 3D network with RGB information may be able to extract better semantic features. Differently from previous approaches that employ two independent architectures, based on the above considerations, we propose our multi-modal, two-branch framework named MM2D3D. In Sec. 3.2 we show how a point cloud can be used to obtain a stronger and more suitable input signal for the 2D network. Similarly, in Sec. 3.3 we describe our multi-modal 3D network.

### 3.2. Depth-based 2D Encoder

In this section, we focus on how we can use point clouds to make a 2D segmentation architecture more robust to domain shift. Inspired by [6, 17], we propose to use depth maps as an input signal that is less influenced by the domain gap. As we can observe from the two depth maps in Fig. 3, it is hard to understand which one was captured during day or night. At the same time, some objects such as the car can be distinguished by only looking at depths (bottom right of the second depth map). Thus, depth maps provide useful hints to solve the task of semantic segmentation. Given these considerations, we argue that exploiting such invariant information may alleviate the domain shifts and can be used to extract discriminative features for the segmentation task. At a first glance, injecting 3D cues into the 2D branch may seem redundant as the 3D network already has the capability to reason on the full 3D scene. However, given that the two networks have very different receptive fields, we can exploit such additional and useful information without the risk of hindering the complementarity of the two signal streams. Assuming point clouds expressed in the camera reference frame and the availability of the intrinsic camera matrix, we can project the original 3D point cloud to obtain a sparse depth map. In practice, the value of the $z$ axis is assigned to the pixel coordinate $(u, v)$ obtained by pro-

---

**Figure 3.** Depth comparison during daylight or night. Differently, from the RGB image (left column), a sparse depth map obtained by projecting a LiDAR scan into the image plane is not affected by the light conditions.

**Figure 4.** 2D Network of our framework. It is composed of a depth encoder and an RGB encoder to process the two inputs independently. The segmentation decoder leverages the multi-scale features of both encoders to predict semantic segmentation labels.
jecting a 3D point into the image plane. Similarly to [17],
to process both inputs, we modify the 2D encoder of the
2D U-Net architecture by including an additional encoder
to process the sparse depth maps obtained from the point
cloud. As can be seen in Fig. 4, the two streams i.e. one for
the RGB image and the other for the sparse depth map, are
processed independently. Then, the concatenated depth and
RGB features are processed by a decoder, composed of a
series of transposed convolutions and convolutions in order
to obtain semantic predictions of the same size as the input
image. Moreover, features from layers of \( \frac{1}{2} \) to \( \frac{1}{16} \) of the
input resolution are concatenated using skip connections with
the corresponding layer of the decoder. This simple design
choice allows semantic predictions to be conditioned also
on the input depth signal, without altering the RGB encoder
that provides useful classification features. Furthermore,
without altering the RGB encoder, we can take advantage of
a pre-trained architecture on ImageNet [12] as done by
our competitors.

3.3. RGB Based 3D Network

In this work, we focus on the 3D convolutional network,
SparseConvNet [15], as it can segment large scenes effi-
ciently. In this network, the initial point cloud is first vox-
elized such that each 3D point is associated with only one
voxel. Then, rather than processing the entire voxel grid,
these models work with a sparse tensor representation ig-
oring empty voxels for the sake of efficiency. The net-
work associates a feature vector to each voxel, and convo-
lutions calculate their results based on these features. A
standard choice for the voxel features is to simply assign to
it a constant value, i.e., 1. Although these strategies have
been shown to be effective [50, 68], the feature vector can
be enriched to make it even more suitable for semantic seg-
mentation. Based on our intuition of the different receptive
fields, we can borrow information from the other modality
to improve the performance of each branch, still preserv-
ing 2D-3D feature complementarity. Thus, we use RGB
colors directly as features for each voxel of the SparseCon-
vNet. Moreover, we design a simple yet effective strategy
to let the 3D network decide whether to use or not this
information. More specifically, the original RGB pixel values
are fed to a linear layer that predicts a scalar value \( \alpha \) to be
multiplied by the color vector. For instance, learning this
scaling could be useful in the UDA scenario, where we can
train on unlabelled target samples, to discard RGB colors in
case they do not provide any useful information, e.g., dark
pixels in images acquired at night time.

3.4. Learning Scheme

Supervised Learning. Given the softmax predictions of
the 2D and 3D networks, \( P_{2D} \) and \( P_{3D} \), we supervise both
branches using the cross-entropy loss on the source domain:

\[
L_{\text{seg}}(x_s, y_s) = - \frac{1}{N} \sum_{n=1}^{N} \sum_{c=1}^{C} y_{s}^{(n,c)} \log P_{x_s}^{(n,c)} \quad (1)
\]

with \( (x_s, y_s) \) being either \((x_{2D}^s, y_{3D}^s)\) or \((x_{3D}^s, y_{3D}^s)\).

Cross-Branch Learning. To allow an exchange of in-
formation between the two branches, [25] and [39] add an
auxiliary classification head to each one. The objective of
these additional classifiers is to mimic the other branch out-
put. The two auxiliary heads estimate the other modal-
ity output: 2D mimics 3D \((P_{2D→3D})\) and 3D mimics 2D
\((P_{3D→2D})\). In practice, this is achieved with the following
objective:

\[
L_{\text{aux}}(x) = D_{\text{KL}}(P_x^{(n,c)} || Q_x^{(n,c)}) \quad (2)
\]

with \( (P, Q) \in \{(P_{2D}, P_{3D→2D}), (P_{3D}, P_{2D→3D})\}\) where \( P \)
is the distribution from the main classification head which
has to be estimated by \( Q \). Note that in Eq. (2), \( x \) can belong
to either \( T \) or \( S \). This means that, in the UDA scenario,
Eq. (2) can also be optimized for \( T \), forcing the two net-
work to have consistent behavior across the two modalities
for the target domain as well without any labels.

Self-Training. Only in the UDA scenario, where unla-
belled target samples are available, as done by [25], we per-
form one round of Self-Training [78] using pseudo-labels
[30]. Specifically, after training the model with Eq. (1) for
the source domain and Eq. (2) on both domains, we gen-
erate predictions on the unlabeled target domain dataset to
be used as pseudo ground truths, \( \hat{y}_t \). Following [25], we
filter out noisy pseudo-labels by considering only the most
confident predictions for each class. Then, we retrain the
framework from scratch the model minimizing the follow-
ing objective function:

\[
L = L_{\text{seg}}(x_s, y_s) + \lambda_t L_{\text{seg}}(x_t, \hat{y}_t)
+ \lambda_{x,s} L_{\text{aux}}(x_s) + \lambda_{x,t} L_{\text{aux}}(x_t) \quad (3)
\]

4. Experiments

4.1. Datasets

To evaluate our method, we follow the benchmark in-
trouded in [25] because it comprehends several interesting
domain shift scenarios. The datasets used in the benchmark
are nuScenes [5] A2D2 [14], SemanticKITTI [3], and Virtu-
alkITTI [13] in which LiDAR point clouds and camera are
synchronized and calibrated so that the projection between
a 3D point and its corresponding 2D image pixel can always
be computed. It is important to note that only 3D points visible from the camera are used for both training and testing. NuScenes consists of 1000 driving scenes in total, each of 20 seconds, with 40k annotated point-wise frames taken at 2Hz, and it is deployed to implement two adaptation scenarios: day-to-night and country-to-country. The former exhibits severe light changes between the source and the target domain, while the latter covers changes in the scene layout. In both settings adaptation is performed on six classes: vehicle, driveable surface, sidewalk, terrain, manmade, vegetation. The third challenging benchmark foresees adaptation from synthetic to real data, and it is implemented by adapting from VirtualKITTI to SemanticKITTI. Since VirtualKITTI only provides depth maps, we use the same simulated LiDAR scans from our competitor [25] for a fair comparison. Note also that to accommodate for the different classes in the two datasets, a class mapping is required and we use the same defined in [25]. The last adaptation scenario involves A2D2 and SemanticKITTI. The A2D2 dataset is composed of 20 drives, with a total of 28,637 frames. As the LiDARs sensor is very sparse (16 layers), all three front LiDARs are used. All frames of all sequences are used for training, except for the sequence 20180807_145028 which is left out for testing. The SemanticKITTI dataset is composed of 20 drives, with a total of 28,637 frames. It is important to note that augmentations are done independently for each branch. We implement our framework in PyTorch using two NVIDIA 3090 GPU with 24GB of RAM. We train with a batch size of 16, alternating batches of source and target domain for the UDA case and source only in DG. The smaller dataset is repeated to match the length of the other. We rely on the AdamW optimizer [33] and the One Cycle Policy as a learning rate scheduler [52]. We train for 50, 35, 15, and 30 epochs for USA → Singapore, Day → Night, v. KITTI → Sem. KITTI, and A2D2 → Sem. KITTI respectively. As regards the hyper-parameters, we follow [25] and set $\lambda_s = 0.8, \lambda_t = 0.1, \lambda_{xs} = 0.1, \lambda_{xt} = 0.01$ in all settings without performing any fine-tuning on these values.

### 4.3. UDA results

Following previous works in the field [25, 39], we evaluate the performance of a model on the target test set using the standard Intersection over Union (IoU) and select the best checkpoint according to a small validation set on the target domain. In Tab. 1, we report our results on the four challenging UDA benchmarks explained in Sec. 4.1. For each experiment, we report two reference methods: a model trained only on the source domain, named Baseline (Source Only); a model trained only on the target data using annotations, representing the upper bound than can be obtained with real ground-truth, namely Oracle. We note that these two models employ the two independent stream architecture of [25]. In the columns Avg, we report the results obtained by the mean of the 2D and 3D outputs after softmax which is the final output of our multi-modal framework. For the sake of completeness, we also report the results of each individual branch (2D and 3D only). We compare our method with both Uni-modal and Multi-Modal approaches. In particular, we mainly focus on a comparison with xMUDA [25] and DsCML [39], as they are the current s.o.t.a. methods for UDA in our multi-modal setting. In particular, for the latter, we use the official code provided by the authors to retrain the model on the new more exhaustive benchmark defined by [25]. Overall, we note how our contributions largely improve results over competitors across all settings and modalities. In USA → Singapore, we observe a large boost in both branches, and on average we report a +3% improvement across the board.

<table>
<thead>
<tr>
<th>Modality</th>
<th>Method</th>
<th>USA → Singapore</th>
<th>Day → Night</th>
<th>v.KITTI → Sem.KITTI</th>
<th>A2D2 → Sem.KITTI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>2D</td>
<td>3D</td>
<td>Avg</td>
<td>2D</td>
</tr>
<tr>
<td>Baseline (Source only)</td>
<td>58.4</td>
<td>62.8</td>
<td>68.2</td>
<td>54.8</td>
<td>68.8</td>
</tr>
<tr>
<td>Uni-modal</td>
<td>MinEnt [60]</td>
<td>57.6</td>
<td>61.5</td>
<td>66.0</td>
<td>47.1</td>
</tr>
<tr>
<td></td>
<td>Deep logCORAL [36]</td>
<td>64.4</td>
<td>63.2</td>
<td>69.4</td>
<td>47.7</td>
</tr>
<tr>
<td></td>
<td>PL [33]</td>
<td>62.0</td>
<td>64.8</td>
<td>70.4</td>
<td>47.9</td>
</tr>
<tr>
<td>Multi-modal</td>
<td>xMUDA [25]</td>
<td>64.4</td>
<td>63.2</td>
<td>69.4</td>
<td>55.5</td>
</tr>
<tr>
<td></td>
<td>DsCML [39]</td>
<td>52.9</td>
<td>52.3</td>
<td>56.9</td>
<td>51.2</td>
</tr>
<tr>
<td></td>
<td>MM2D3D (Ours)</td>
<td>71.7</td>
<td>66.8</td>
<td>72.4</td>
<td>70.5</td>
</tr>
<tr>
<td>Oracle</td>
<td>75.4</td>
<td>76.0</td>
<td>79.6</td>
<td>61.5</td>
<td>69.8</td>
</tr>
</tbody>
</table>
Table 2. Results for 3D for semantic segmentation in the Domain Generalization setting. We report performance for each network stream in terms of mIoU. ‘Avg’ column denotes the obtained by taking the mean of the 2D and 3D predictions. * indicates trained by us using official code.

Figure 5. Qualitative examples of the proposed framework in the UDA scenario. From left to right: RGB images, point cloud segmentations projected into 2D for visualization purpose of the baseline source only model, our method, and the ground truth respectively. From top to bottom: the four different adaptation scenarios. Comparisons are provided for the target domain.

(3rd row of the Multi-modal section). The large improvement (+7.3%) for the 2D model, suggests that the depth cues injected into a common 2D decoder can be quite useful even if the light conditions are similar across domains. In Day → Night, we observe a remarkable +15% for the 2D branch, which in turn rises the average score to +4.7% when compared with the previous best model. We attribute this boost in performance to the depth encoder, which is able to provide useful hints when the RBG encoder has to deal with large changes in light conditions. Remarkably, our network surpasses even the performance of the two independent streams Oracle. Indeed, as discussed in Sec. 3.2, the sparse depth is able to give useful details for the task of semantic segmentation. Moreover, thanks to the fact that the cross-modal loss Sec. 3.4 is optimized for both domains, the network lean to use both encoders to make the final predictions, leading to more robust performance when the encoder receives a less informative RGB signal. In the challenging synthetic-to-real case (v. KITTI → Sem. KITTI), we also notice consistent improvements in both branches. We highlight that even though RGB colors are here likely the main source of the domain gap, they are still useful to obtain a stronger 3D model (+3.6%). In the A2D2 → Sem. KITTI setting, where the sensors setup is different, we still benefit from the depth hints provided to the 2D network, and on average, our method surpasses by 2.2% xMUDA. In general, we highlight that though we employed both modalities in the 2D and 3D branches, the Avg performances are better than those of each individual branch, supporting our core intuition. In Fig. 5, we report some qualitative results obtained with our framework.

4.4. Domain Generalization results

In this section, we test our contributions in the Domain Generalization setting, in which the target data cannot be used at training time. For this study we consider XMUDA [25] as our baseline two-branch 2D-3D method, and we show that our simple contribution can boost generalization performances. Results are reported in Tab. 2. To implement this experiment we keep the same hyper-parameters as used in the UDA scenario. We retrain [24] using the official code, but without the target data. Also in this set-
Table 3. Modality-wise ablation of the proposed framework in the UDA scenario. Depth indicates the usage of the additional sparse depth encoder, while RGB denotes the introduction of the RGB information in the 3D network.

<table>
<thead>
<tr>
<th>Method</th>
<th>Depth</th>
<th>RGB</th>
<th>USA → Singapore</th>
<th>Day → Night</th>
<th>v. KITTI → Sem. KITTI</th>
<th>A2D2 → Sem. KITTI</th>
</tr>
</thead>
<tbody>
<tr>
<td>xMUDA [25]</td>
<td></td>
<td></td>
<td>64.4</td>
<td>61.2</td>
<td>69.4</td>
<td>55.9</td>
</tr>
<tr>
<td>MM2D3D (Ours)</td>
<td>✓</td>
<td></td>
<td>69.5</td>
<td>64.0</td>
<td>69.6</td>
<td>71.3</td>
</tr>
<tr>
<td>MM2D3D (Ours)</td>
<td>✓</td>
<td>✓</td>
<td>71.7</td>
<td>66.8</td>
<td>72.4</td>
<td>70.5</td>
</tr>
</tbody>
</table>

Table 4. Self-training Analysis. Results with different self-training strategies in the UDA scenario.

<table>
<thead>
<tr>
<th>Method</th>
<th>Depth</th>
<th>RGB</th>
<th>USA → Singapore</th>
<th>Day → Night</th>
<th>v. KITTI → Sem. KITTI</th>
<th>A2D2 → Sem. KITTI</th>
</tr>
</thead>
<tbody>
<tr>
<td>xMUDA [25] + PL</td>
<td></td>
<td></td>
<td>67.0</td>
<td>65.4</td>
<td>71.2</td>
<td>57.6</td>
</tr>
<tr>
<td>MM2D3D (Ours) + PL</td>
<td></td>
<td>✓</td>
<td>74.3</td>
<td>68.3</td>
<td>74.9</td>
<td>71.3</td>
</tr>
<tr>
<td>MM2D3D (Ours) + Fusion</td>
<td>x</td>
<td>x</td>
<td>74.0</td>
<td>x</td>
<td>x</td>
<td>71.0</td>
</tr>
</tbody>
</table>

4.5. Ablation Studies

Modality-wise analysis. In Tab. 3, we ablate our contributions starting from the model proposed by [24] in the UDA scenario. We start by activating our depth-based network, introduced in Sec. 3.3. The performance boost given by our proposal is remarkable across all settings. In cases such as Day → Night, where the RGB gap is larger, the depth cues injected with skip connections to the semantic decoder greatly enhance performances in the target domain (+15.8% for 2D and +5.4% in “Avg”). We note also a consistent improvement for the remaining settings, in particular, we highlight +10.5% for the 2D scores on the challenging synthetic-to-real adaptation benchmark (v. KITTI → Sem. KITTI). Furthermore, when feeding RGB colors to the 3D network (last row of Tab. 3), we observe improved performances in almost all settings. The largest improvement is observed in the synthetic-to-real setting, where we achieve a +10% in terms of mIoU for the 3D, which in turn increased the average score from 53.7% to 56.5%. Better performance is also achieved for both the 3D network and the average score for A2D2 → Sem. KITTI.

Self-Training. In this section, we compare different self-training strategies and report results in Tab. 4. As explained in Sec. 3.4, for the self-training protocol we first need a model trained on the source domain to produce the pseudo-labels for the target domain in the second round. We report in the first row of Tab. 4 the performance of this starting model to better appreciate the effectiveness of self-training. First, we note how thanks to our contributions, for USA → Singapore, Day → Night, and v. KITTI → Sem. KITTI we already surpass xMUDA [24] on the Avg column even without the usage of pseudo-labels. When pseudo-labels from the 2D and the 3D branches are used to supervise the 2D and the 3D network respectively, we establish new state-of-the-art performances for all four settings in the average predictions (third row). Furthermore, in the fourth row of Tab. 4, we deploy the strategy proposed in [24], where point-wise features from the two networks are concatenated and used to train a unique classifier. In this case, we observe mixed results, indicating that this self-training strategy is not necessarily better across all settings when compared to the standard self-training protocol.

5. Conclusions

In this paper, we shed light on the complementarity of recent and emerging 3D-2D architectures for 3D semantic segmentation. We provide an intuitive explanation based on the notion of effective receptive field of why processing data with these two networks grants orthogonal predictions that can be effectively fused together. Based on this, we propose to feed both modalities to both branches. Despite the simplicity of our approach, we establish new state-of-the-art results in four common UDA scenarios and demonstrate superior generalization performance over the baseline 2D-3D architecture. A limitation of our work is that our method is purely multi-modal, and it requires both modalities and a valid calibration across sensors at test time. An interesting future direction is to investigate how our approach may generalize to other multi-modal 2D-3D architectures for semantic segmentation.
References


[56] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schultuer, Kihyuk Sohn, Ming-Hsuan Yang, and Mannmohan Chandraker. Learning to adapt structured output space for semantic segmentation. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Jun 2018. 3


[67] Zuxuan Wu, Xintong Han, Yen-Liang Lin, Mustafa Gökhun Uzunbas, Tom Goldstein, Ser Nam Lim, and Larry S. Davis. Dcan: Dual channel-wise alignment networks for supervised scene adaptation. Lecture Notes in Computer Science, page 535–552, 2018. 3


