EDAPS: Enhanced Domain-Adaptive Panoptic Segmentation

Suman Saha*  Lukas Hoyer*  Anton Obukhov  Dengxin Dai  Luc Van Gool
ETH Zurich  ETH Zurich  ETH Zurich  Huawei Zurich Research Center  ETH Zurich, KU Leuven

Abstract

With autonomous industries on the rise, domain adaptation of the visual perception stack is an important research direction due to the cost savings promise. Much prior art was dedicated to domain-adaptive semantic segmentation in the synthetic-to-real context. Despite being a crucial output of the perception stack, panoptic segmentation has been largely overlooked by the domain adaptation community. Therefore, we revisit well-performing domain adaptation strategies from other fields, adapt them to panoptic segmentation, and show that they can effectively enhance panoptic domain adaptation. Further, we study the panoptic network design and propose a novel architecture (EDAPS) designed explicitly for domain-adaptive panoptic segmentation. It uses a shared, domain-robust transformer encoder to facilitate the joint adaptation of semantic and instance features, but task-specific decoders tailored for the specific requirements of both domain-adaptive semantic and instance segmentation. As a result, the performance gap seen in challenging panoptic benchmarks is substantially narrowed. EDAPS significantly improves the state-of-the-art performance for panoptic segmentation UDA by a large margin of 20% on SYNTHIA→Cityscapes and even 72% on the more challenging SYNTHIA→Mapillary Vistas. The implementation is available at https://github.com/susaha/edaps.

1. Introduction

Panoptic segmentation [30] of images is a core computer vision task that jointly solves two related problems – semantic segmentation and instance segmentation. With the rise of robotics and emerging autonomous driving markets, efficient visual perception stacks are in high demand. However, large-scale supervised learning of panoptic segmentation [7, 30, 29, 49, 70, 44] is prohibitively expensive as it requires dense annotations for both semantics and instances, which require time-consuming manual labeling.

*These authors contributed equally to this work.
is much smaller than that of semantic segmentation (88% in DAFormer [20]).

To understand the root cause of the identified performance gap, we revisit the progress of UDA in semantic segmentation, lift the well-performing UDA strategies to panoptic segmentation, and show that they can effectively enhance panoptic segmentation UDA.

Further, we revisit the panoptic network design and conduct a study of principal architecture designs for panoptic segmentation with respect to their UDA capabilities. We show that previous UDA methods took sub-optimal design choices. While separating the networks for both tasks prevents the network from jointly adapting task-shared features from the source to the target domain (see Fig. 2a), a shared encoder-decoder cannot accommodate the different needs (such as task-specific knowledge or architecture design) when adapting both tasks (see Fig. 2b).

To address these problems, we propose EDAPS, a network architecture that is particularly designed for domain-adaptive panoptic segmentation. It uses a shared transformer [65] encoder to facilitate the joint adaptation of semantic and instance features, but task-specific decoders tailored for the specific requirements of both domain-adaptive semantic segmentation and domain-adaptive instance segmentation (see Fig. 2c). Specifically, separate decoders can learn task-specific parameters, mitigating the issue of conflicting gradients [74], and their architectures can be independently chosen to best suit semantic or instance segmentation. Even though this design was used in the supervised setting, we are the first to identify its strong potential specifically for panoptic UDA.

Utilizing the enhanced panoptic UDA and the enhanced domain-adaptive panoptic network design, EDAPS shows significant performance gains over the prior works on a number of standard perception benchmarks (Fig. 1). EDAPS improves the state-of-the-art mPQ from 33.0 to 41.2 on On SYNTHIA → Cityscapes and from 21.3 to 36.6 on SYNTHIA → Mapillary Vistas. EDAPS trains semantic segmentation and instance segmentation tasks together using a joint training optimization. Therefore, EDAPS can be trained end-to-end in a single stage and the training only requires 21 hours on a single RTX 2080 Ti GPU, which improves its applicability for the community.

The main contribution of EDAPS is the novel combination of network components and UDA strategies on a system level, which results in a major relative gain in the state-of-the-art panoptic segmentation UDA performance of 20% on SYNTHIA-to-Cityscapes and even 72% on the more challenging SYNTHIA-to-Mapillary Vistas. In particular, we carefully study various principal panoptic architectures for their UDA capability and identify a network design that is particularly suited for panoptic UDA. It improves the UDA performance over strong baselines, while being parameter-efficient and fast at inference. Further, this is the first paper that lifts recent UDA techniques to panoptic segmentation and systematically studies their effectiveness for panoptic UDA.

2. Related Work

We outline two cornerstone classes of works related to UDA in Semantic and Panoptic Segmentation settings.

**UDA for Semantic Segmentation** Methods in this class take input images from source and target domains along with the source semantic ground truth label. A supervised cross-entropy loss is computed on the source image semantic prediction. A UDA semantic loss (e.g., adversarial [64] or pseudo-label-based self-training [80] loss) is used for domain alignment that operates on the semantic feature space. UDA for semantic segmentation is one of the most popular tasks in dense prediction, and a large number of works in this category exist in the literature [64, 61, 18, 32, 28, 79, 60, 43, 9, 1, 76, 80, 73, 68, 41, 13, 5, 19, 27, 77, 37, 52]. The first approach of using an adversarial loss helps to align the source and target domain distributions at the input [18, 16], at the feature-level [61, 19], at the output [61, 64], or at the patch level [62]. In the second approach of using self-training [80, 32, 77, 81], pseudo-labels are generated on the unannotated target domain images either offline [56, 73, 80, 81] or online [76, 66, 60, 79, 20, 21]. The pseudo-labels can be stabilized with consistency regularization [59, 57] based on pseudo-label prototypes [76], differ-
ent data augmentation schemes [1, 9, 43, 23], cross-domain mixup strategies [60, 79], and multiple resolutions [21, 22].

**UDA for Panoptic Segmentation** Considering the used network architectures, there are distinct shortcomings in both CVRN [25] and UniDAPS [75]. On the one side, CVRN requires the training of two separate networks for semantic segmentation and instance segmentation, which is time-consuming, parameter-inefficient, and slow during inference (Fig. 2). Furthermore, no task-agnostic knowledge can be shared during the adaptation process by separating the semantic from the instance network. The approach relies on expensive multi-stage training (i.e., training, pseudo-label generation, and retraining with fused labels). On the other side, UniDAPS proposed to adapt panoptic segmentation within a unified network [2], which directly predicts panoptic segments instead of separate semantic and instance masks. While this simple concept is intriguing and performs well in a supervised setting [2, 8, 34], unified network architectures are not inherently suited for UDA as shown in [75]. Even with specific UDA strategies to compensate for that, UniDAPS achieves only a small improvement over CVRN.

The authors of [20, 22] pointed out that architectures and training schemes that work for supervised learning might not work as desirably for UDA, and special care is required (in network design and training recipes) to address the domain shift problem effectively. Outside of the synthetic-to-real domain, the authors of [35] proposed a panoptic UDA approach to microscopy image analysis. In this paper, we propose a domain-adaptive panoptic segmentation framework that carefully selects the network design and training recipes tailored explicitly for UDA.

### 3. Method

In this section, we first recap panoptic image segmentation. Second, we present our enhanced panoptic segmentation UDA pipeline. Third, we define principal panoptic architectures for a systematic analysis of their UDA capabilities. And finally, we introduce our EDAPS network architecture, specifically designed for panoptic UDA.

#### 3.1. Supervised Panoptic Segmentation

Panoptic segmentation is commonly approached by decomposing the task into a semantic segmentation and an instance segmentation component so that the panoptic segmentation loss $L_{PS}$ is composed of a semantic and instance loss: $L_{PS} = L_{sem} + L_{inst}$. The semantic segmentation loss typically uses a pixel-wise cross-entropy loss to assign each pixel of an image to one class from a pre-defined set. Instance segmentation further distinguishes instances within classes with countable entities, the so-called thing-classes such as *car* or *person* (as opposed to uncountable stuff-classes such as *road* or *sky*). Instance segmentation can be approached in two ways. In top-down approaches such as Mask R-CNN [17], instances are predicted based on proposals in the form of bounding boxes and instances masks. In bottom-up approaches such as Panoptic-DeepLab [7], the instances are grouped on a pixel-level without proposals, for example using pixel-wise instance center and offset heatmaps. During inference, semantic and instance segmentations are deterministically fused into a panoptic segmentation representation.

#### 3.2. Enhanced Panoptic UDA

In the UDA setting, a neural network $F_{\theta}$ is trained on annotated source domain images $X^{(s)} = \{x^{(s)}_i\}_{i=1}^{N^{(s)}}$ and unannotated target domain images $X^{(t)} = \{x^{(t)}_i\}_{i=1}^{N^{(t)}}$ with $x^{(s)}_i, x^{(t)}_i \in \mathbb{R}^{H \times W \times 3}$ with the objective of achieving good performance on the target domain. As panoptic segmentation ground truth $Y^{(s)} = \{y^{(s)}_i\}_{i=1}^{N^{(s)}}$ is only available on the source domain, the panoptic model can only be trained with source data in a supervised fashion. In particular, the source loss term $L^{(s)} = L_{PS}(\hat{y}^{(s)}, y^{(s)})$.

Models trained using the supervised loss $L^{(s)}$ on the source domain often exhibit poor generalization on the target domain due to the “domain gap” between the distributions of source and target data. To adapt the model to the target domain, an additional unsupervised loss term $L^{(t)}$ is computed on the target domain. The overall loss combines both the source and target loss terms $L_{UDA} = L^{(s)} + L^{(t)}$.

Various unsupervised target loss terms were proposed in the literature, mostly based on adversarial training [61, 62,
Self-Training on Target Domain In this work, we resort to self-training for adapting the network to the target domain similar to previous panoptic UDA methods [71, 75]. In self-training, the model \( F_\theta \) is trained with high-confidence pseudo-labels [31] on the target domain using a weighted cross-entropy loss

\[
L(t) = - \sum_{i,j,c} q_{i,j}^{(t)} \left( p_i^{(t)} \log(\hat{y}_{i,j}^{(t)}) \right)_{i,j,c}
\]

which helps the network gradually adapt to the target domain. Here, \( p_i^{(t)} \) denotes the pseudo-label and \( q_i^{(t)} \) its confidence estimate.

The pseudo-labels can be generated using the predictions of a teacher network \( T_\theta \). As the output of \( T_\theta \) is a map of per-pixel probabilities, a single class assignment is done using the mode of categorical distribution, which could be converted back to one-hot categorical form via the Iverson bracket [1]:

\[
p_{i,j,c}^{(t)} = \left[ c = \arg \max_{c'} T_\theta(x_i^{(t)})_{i,j,c'} \right].
\]

Since the pseudo-labels are the predictions of a network, they are not always correct, and their quality can be estimated by a confidence estimate \( q_i^{(t)} \) [80, 42, 60, 24]

\[
q_i^{(t)} = \frac{\sum_{i,j} \left[ \max_{c'} T_\theta(x_i^{(t)})_{i,j,c'} > \tau \right]}{H \cdot W}.
\]

To stabilize the quality of the pseudo-labels, we resort to the mean teacher framework [59], which is commonly used in semantic segmentation UDA [1, 60, 36, 20]. The parameters of the teacher network \( T_\theta \) are updated with the exponential moving average of the parameters of the student network \( F_\theta \) at every training step \( t \):

\[
\phi_{t+1} \leftarrow \alpha \phi_t + (1 - \alpha) \theta_t.
\]

Alongside self-training, we adopt a consistency training [55, 59, 57], in which the student network \( F_\theta \) is trained on augmented target images, whereas, the mean teacher network \( F_\theta \) predicts pseudo-labels for actual target images. The augmented images are generated by mixing pixels of the source and target images. The mixing is done following the Class-Mix strategy [47], which has been successfully applied to semantic segmentation UDA [60, 36, 20]: first, we randomly select \( N/2 \) semantic classes among \( N \) classes present in the source image. Next, the pixels belonging to these selected \( N/2 \) semantic classes are pasted into the target image, resulting in a new augmented image.

Furthermore, we leverage the recent findings in semantic segmentation UDA [24] to further enhance the performance of the panoptic UDA. More specifically, we adopt rare class sampling and a feature regularization loss based on ImageNet features to learn domain-invariant representations for panoptic UDA.

Rare Class Sampling (RCS) Most datasets are imbalanced, so certain classes are underrepresented. It can hurt the adaptation process if a certain class is sampled with a low frequency [20]. Therefore, we follow DAFormer [20] and sample images with rare classes more frequently. The sampling probability of class \( c \) is

\[
P(c) = \frac{e^{(1-f_c)/T}}{\sum_{c'=1}^C e^{(1-f_{c'})/T}}.
\]

where \( f_c \) denotes the class frequency in the source dataset and \( T \) is a temperature parameter. Given a sampled class \( c \sim P \), a source sample is drawn from the subset containing class \( c \): \( x^{(s)} \sim \text{uniform}(X^{(s,c)}) \).

ImageNet Feature Distance (FD) The encoder network in panoptic segmentation UDA is usually pre-trained using ImageNet classification. Given that ImageNet is a real-world dataset, its features can be valuable when adapting to a real-world target domain. However, it was observed that some relevant features could be forgotten during the adaptation process [20]. Therefore, it can be useful to regularize the adaptation process with an ImageNet feature distance loss term [20]

\[
L_{FD} = \left\| F_{\text{ImageNet}}(x^{(s)}) - F_{\theta}(x^{(s)}) \right\|_2^2,
\]

where \( F_{\text{ImageNet}} \) denotes the frozen ImageNet model. As ImageNet annotates thing classes, it is beneficial to constrain the feature distance to image regions that are labeled as thing classes [20].

3.3. Principal Panoptic Architectures for UDA

To further facilitate panoptic UDA, we aim to design a network architecture particularly tailored for domain-adaptive panoptic segmentation. Several different network architectures have been proposed for panoptic segmentation in the supervised setting. However, the effect of the different designs on the performance in a domain adaptation setting is not well studied. Therefore, we analyze four principal panoptic architectures in a fair comparison using the same network building blocks and the same enhanced panoptic UDA strategy in §4.3. The systematic comparison is the foundation for our EDAPS architecture.

M-Net uses two separate encoder-decoder networks for semantic and instance segmentation (see Fig. 2a). This design was deployed in CVRN [25]. The two encoders utilize the same architecture but the decoder architecture are task-specific. A bottom-up decoder is used for semantic segmentation while a top-down decoder is used for instance segmentation. The design of the network building blocks is further detailed in §3.4. A potential disadvantage of M-Net is the use of separate encoders, which prevents the network from jointly adapting task-shared features from the source to the target domain.
S-Net shares the encoder and decoder for both tasks (see Fig. 2b), which is similar to the architecture of UniDAPS [75]. As the decoder is shared, it is not possible to have task-specific designs, which can be disadvantageous if different decoder architectures have better domain adaptation properties for semantic and instance segmentation. Also, shared decoders can favor negative transfer of task-specific knowledge.

M-Dec-BU shares the encoder but splits the decoders to enable task-specific knowledge. Both decoders use a bottom-up design.

M-Dec-TD (EDAPS) shares the encoder but splits the decoders to enable task-specific knowledge. The instance decoder uses a top-down approach in contrast to the bottom-up semantic decoder (see Fig. 2c). We adopt this design in our EDAPS architecture. In the following section, we motivate and detail the design choices of the EDAPS architecture.

3.4. EDAPS Network Architecture

An overview of the proposed EDAPS architecture is presented in Fig. 3. On the one side, EDAPS utilizes a shared transformer backbone, which has an improved domain-robustness compared to CNNs [45, 20]. Sharing the encoder is advantageous compared to the separate networks as it can facilitate the joint adaptation of semantic and instance features from the source to the target domain. On the other side, EDAPS uses task-specific decoders with different design tailored for the specific requirements of domain-adaptive semantic segmentation and domain-adaptive instance segmentation. In that way, EDAPS combines the strength of previous methods (see Fig. 2).

Shared Hierarchical Transformer Encoder Transformers have shown domain-robust properties beneficial for UDA [20, 72, 58]. Therefore, EDAPS uses a Mix Transformer (MiT-B5) [69] as its encoder network, which is tailored for dense predictions by generating coarse (high-resolution) and fine-grained (low-resolution) features at different levels. Unlike 16 patches of ViT [12], MiT divides an image into relatively smaller patches of shape 4 × 4; this facilitates preserving finer details helpful in improving semantic segmentation. The patches are processed with four transformer blocks into multi-level features at {1/4, 1/8, 1/16, 1/32} scales of the original image shapes. Moreover, an efficient self-attention is used to cope with the computation bottleneck of transformer encoders, i.e., the computational complexity of the self-attention operation is minimized by reducing the input sequence length \((H \times W)\) using a reduction ratio as in [67]. Overlapping patch merging is used to downsample feature maps which helps preserve local continuity. These properties of MiT make it possible to learn robust domain-invariant features for UDA panoptic segmentation using limited computing resources (i.e., on a single GPU).

Bottom-Up Semantic Decoder We adopt a domain-robust context-aware feature fusion decoder [24] as our semantic decoder, which has two main benefits over traditional DeepLab decoders [3, 15] or MLP-based decoders [67, 69, 78]. Firstly, it allows exploiting context information alongside local information in the decoder, which is beneficial for domain-robust semantic segmentation [26]. Secondly, instead of only using the context information coming from the bottleneck features [3, 15], it fuses context encoded by features at different levels of the backbone network. The high-resolution features from the earlier layers encode low-level concepts helpful in better semantic segmentation. The multi-level context-aware feature fusion is learned using a series of 3×3 depthwise separable convolutions [10] with different dilation rates.

Top-Down Instance Decoder Based on the findings of the network study in §4, EDAPS resorts to a proposal-based top-down instance decoder, which exhibits better domain-adaptive properties compared to a bottom-up decoder.

Specifically, we follow Mask R-CNN [17] and use a Region Proposal Network (RPN) to predict candidate object bounding boxes. It is trained using a binary cross-entropy (CE) loss for bounding box classification (object or no object), and an \(L_1\) loss for box regression. Given the bounding boxes predicted by RPN, a Region-of-Interest Alignment (RoIAlign) layer is used to extract features from each box. The extracted features are used as inputs to the box and mask heads. The box head is trained to predict the bounding boxes and class labels, and the mask head is trained to predict the class-agnostic binary instance masks. The box head is trained using CE loss for object classification and \(L_1\) loss for box regression over all thing classes. The mask head predictions are penalized using the binary CE loss.

Bottom-Up Instance Decoder (Baseline) For the network studies with a bottom-up instance decoder, we adapt the strong domain-robust decoder from [24] from semantic segmentation to instance segmentation by replacing the classification head (for semantic prediction) with two \(1 \times 1\) convolutional layers that predict the instance centers and offsets. It is trained using the losses proposed in [7].

Feature Fusion We follow Panoptic-DeepLab [7] to fuse semantic and instance predictions into the final panoptic segmentation upon inference. Class-agnostic instance segmentation maps are generated by selecting the top-k instance predictions with a detection score above a certain threshold. The resulting instance segmentation is fused with the predicted semantic map by a majority-voting rule to generate the final panoptic segmentation.
4. Experiments

4.1. Implementation Details

Datasets  We evaluate EDAPS for the common setting of synthetic-to-real adaptation. As a source dataset, we use Synthia [51], which contains 9,400 synthetic images and panoptic labels. As target datasets, we use Cityscapes [11] and Mapillary Vistas [46]. Cityscapes [11] consists of 2,975 training and 500 validation images of European street scenes. Mapillary Vistas [46] is a large-scale autonomous driving dataset consisting of 18,000 training and 2,000 validation images.

Training  We follow DAFormer [20] and train EDAPS with AdamW [40] for 40k iterations, a batch size of 2, a crop size of 512×512, a learning rate of 6×10⁻⁵ with a linear warmup for 1.5k iterations and polynomial decay afterward, and a weight decay of 0.01. The experiments are conducted on a RTX 2080 Ti GPU with 11 GB memory. We use the same data augmentation parameters as in DACS [66]. To encourage the under-represented classes (in the source domain) to be sampled more frequently, we use RCS [20] and set the RCS temperature to 0.01. Following DAFormer [20], we also use the ImageNet feature distance loss to preserve information about certain thing classes encoded in the ImageNet features. We set this loss weight \( \lambda_{FD} = 0.005 \). For the bottom-up instance head, we use the center loss weight \( \lambda_{heatmap} = 10.0 \), offset loss weight \( \lambda_{offs} = 0.1 \). For the top-down instance head, we set the following loss weights to 1.0: \( \lambda_{RPN-cls} \), \( \lambda_{RPN-box} \), \( \lambda_{RoI-cls} \), \( \lambda_{RoI-box} \), \( \lambda_{RoI-mask} \).

Evaluation Metrics  For evaluating panoptic segmentation, we use the panoptic quality (PQ) metric [30] that captures performance for all classes (including stuff and things) in an interpretable and unified manner. PQ can be seen as the multiplication of a segmentation quality (SQ) term and a recognition quality (RQ) term, i.e., \( PQ = SQ \times RQ \). We report PQ for each category. Mean SQ (mSQ), mean RQ (mRQ), and mean PQ (mPQ) are average scores over all categories. For evaluating instance and semantic segmentation, we use the mIoU and mAP metrics respectively.

4.2. Comparison with the State of the Art

First, we compare our EDAPS with the state-of-the-art methods for panoptic segmentation UDA on SYNTHIA \( \rightarrow \) Cityscapes (Tab. 1), SYNTHIA \( \rightarrow \) Mapillary (Tab. 2), Cityscapes \( \rightarrow \) Foggy Cityscapes, and Cityscapes \( \rightarrow \) Mapillary Vistas (Tab. 3). It can be seen that EDAPS consistently outperforms previous methods with a large margin in all aggregated metrics (mSQ, mRQ, mPQ). Specifically, EDAPS improves the mPQ from 34.2 to 41.2 on SYNTHIA \( \rightarrow \) Cityscapes and from 21.3 to 36.6 SYNTHIA \( \rightarrow \) Mapillary, which is a respective improvement of remarkable 20% and 72% over previous works. Tab. 1 shows that EDAPS significantly improves both the recognition (+9.3 mRQ) as well as the segmentation quality (+5.8 mSQ) over previous SOTA. Both can be also observed qualitatively in Fig. 4. EDAPS particularly improves the performance of classes that previous methods struggled with, such as wall, pole, traffic light, traffic sign, person, rider on Cityscapes (Tab. 1), and road, sidewalk, building, wall, pole, traffic light, traffic sign, vegetation, sky, bus on Mapillary (Tab. 2). Additional benchmarks for other domain gaps are presented in Tab. 3. EDAPS also improves the SOTA on Cityscapes \( \rightarrow \) Foggy Cityscapes and Cityscapes \( \rightarrow \) Mapillary Vistas with respective improvements of 47% and 23% over UniDAPS and CVRN with fewer parameters and faster inference speed (Tab. 5). As the different panoptic UDA methods resort to different network architectures, it has to be considered that they have a varying inherent capability to learn panoptic segmentation. Therefore, we compare both the UDA performance and the supervised performance (as indicators of the capability of a network architecture in Tab. 4. Further, we normalize the UDA performance by the supervised performance (mPQ_rel) for a more fair comparison. It can be seen that even though UniDAPS uses a more powerful network than CVRN, as indicated by the higher supervised performance, this does not translate well to UDA performance.
Table 1: Comparison with state-of-the-art methods on SYNTHIA → Cityscapes benchmark for UDA panoptic segmentation. For clarity, per class PQs are reported. The results of EDAPS are averaged over 3 random seeds.

<table>
<thead>
<tr>
<th>UDA Method</th>
<th>road</th>
<th>sidewalk</th>
<th>building</th>
<th>wall</th>
<th>fence</th>
<th>pole</th>
<th>light</th>
<th>sign</th>
<th>veg</th>
<th>sky</th>
<th>person</th>
<th>rider</th>
<th>car</th>
<th>bike</th>
<th>mSQ</th>
<th>mRQ</th>
<th>mPQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>FDA [73]</td>
<td>79.0</td>
<td>22.0</td>
<td>61.8</td>
<td>1.0</td>
<td>0.0</td>
<td>5.6</td>
<td>5.5</td>
<td>9.5</td>
<td>51.6</td>
<td>70.7</td>
<td>23.4</td>
<td>16.3</td>
<td>34.1</td>
<td>31.0</td>
<td>5.2</td>
<td>8.8</td>
<td>65.0</td>
</tr>
<tr>
<td>CRST [81]</td>
<td>75.4</td>
<td>19.0</td>
<td>70.8</td>
<td>1.4</td>
<td>0.0</td>
<td>7.3</td>
<td>0.0</td>
<td>5.2</td>
<td>74.1</td>
<td>69.2</td>
<td>23.7</td>
<td>19.9</td>
<td>33.4</td>
<td>26.6</td>
<td>2.4</td>
<td>4.8</td>
<td>60.3</td>
</tr>
<tr>
<td>AdvEnt [64]</td>
<td>87.1</td>
<td>32.4</td>
<td>69.7</td>
<td>1.1</td>
<td>0.0</td>
<td>3.8</td>
<td>0.7</td>
<td>2.3</td>
<td>71.7</td>
<td>72.0</td>
<td>28.2</td>
<td>17.7</td>
<td>31.0</td>
<td>21.1</td>
<td>6.3</td>
<td>4.9</td>
<td>65.6</td>
</tr>
<tr>
<td>CVRN [25]</td>
<td>86.6</td>
<td>33.8</td>
<td>74.6</td>
<td>3.4</td>
<td>0.0</td>
<td>10.0</td>
<td>5.7</td>
<td>13.5</td>
<td>80.3</td>
<td>76.3</td>
<td>26.0</td>
<td>18.0</td>
<td>34.1</td>
<td>26.6</td>
<td>6.3</td>
<td>4.9</td>
<td>65.3</td>
</tr>
<tr>
<td>UniDAPS [75]</td>
<td>73.7</td>
<td>26.5</td>
<td>71.9</td>
<td>1.0</td>
<td>0.0</td>
<td>7.6</td>
<td>9.9</td>
<td>5.2</td>
<td>74.1</td>
<td>69.2</td>
<td>23.7</td>
<td>19.9</td>
<td>33.4</td>
<td>26.6</td>
<td>6.3</td>
<td>4.9</td>
<td>65.6</td>
</tr>
<tr>
<td>w/ CVRN Net [75]</td>
<td>87.7</td>
<td>34.0</td>
<td>73.2</td>
<td>1.3</td>
<td>0.0</td>
<td>8.1</td>
<td>9.9</td>
<td>6.7</td>
<td>78.2</td>
<td>74.0</td>
<td>37.6</td>
<td>15.0</td>
<td>18.8</td>
<td>2.8</td>
<td>72.7</td>
<td>53.6</td>
<td>41.2</td>
</tr>
<tr>
<td>EDAPS (Ours)</td>
<td>77.5</td>
<td>36.9</td>
<td>80.1</td>
<td>17.2</td>
<td>1.8</td>
<td>29.2</td>
<td>33.1</td>
<td>40.9</td>
<td>82.6</td>
<td>80.4</td>
<td>43.5</td>
<td>33.8</td>
<td>45.6</td>
<td>35.6</td>
<td>12.6</td>
<td>15.4</td>
<td>64.7</td>
</tr>
</tbody>
</table>

Table 2: Comparison with state-of-the-art methods on SYNTHIA → Mapillary Vistas benchmark for UDA Panoptic Segmentation. For clarity, per class PQs are reported. The results of EDAPS are averaged over 3 random seeds.

<table>
<thead>
<tr>
<th>UDA Method</th>
<th>road</th>
<th>sidewalk</th>
<th>building</th>
<th>wall</th>
<th>fence</th>
<th>pole</th>
<th>light</th>
<th>sign</th>
<th>veg</th>
<th>sky</th>
<th>person</th>
<th>rider</th>
<th>car</th>
<th>bike</th>
<th>mSQ</th>
<th>mRQ</th>
<th>mPQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>FDA [73]</td>
<td>44.1</td>
<td>7.1</td>
<td>26.6</td>
<td>1.3</td>
<td>0.0</td>
<td>5.5</td>
<td>45.2</td>
<td>61.3</td>
<td>30.1</td>
<td>13.9</td>
<td>39.4</td>
<td>12.1</td>
<td>8.5</td>
<td>1.0</td>
<td>7.0</td>
<td>63.8</td>
<td></td>
</tr>
<tr>
<td>CRST [81]</td>
<td>36.0</td>
<td>6.4</td>
<td>29.1</td>
<td>0.2</td>
<td>0.0</td>
<td>4.6</td>
<td>47.7</td>
<td>68.9</td>
<td>28.3</td>
<td>13.0</td>
<td>42.4</td>
<td>13.6</td>
<td>5.1</td>
<td>2.0</td>
<td>63.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AdvEnt [64]</td>
<td>27.7</td>
<td>6.1</td>
<td>28.1</td>
<td>0.3</td>
<td>0.0</td>
<td>5.2</td>
<td>48.1</td>
<td>66.5</td>
<td>28.4</td>
<td>28.4</td>
<td>40.5</td>
<td>14.6</td>
<td>5.2</td>
<td>3.3</td>
<td>63.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CVRN [25]</td>
<td>33.4</td>
<td>7.4</td>
<td>32.9</td>
<td>1.6</td>
<td>0.0</td>
<td>5.2</td>
<td>48.1</td>
<td>66.5</td>
<td>28.4</td>
<td>28.4</td>
<td>40.5</td>
<td>14.6</td>
<td>5.2</td>
<td>3.3</td>
<td>63.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EDAPS (Ours)</td>
<td>77.5</td>
<td>25.3</td>
<td>59.9</td>
<td>14.9</td>
<td>0.9</td>
<td>11.6</td>
<td>16.4</td>
<td>47.5</td>
<td>71.7</td>
<td>46.1</td>
<td>36.6</td>
<td>14.9</td>
<td>7.9</td>
<td>11.6</td>
<td>47.5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Comparison on Cityscapes → Foggy Cityscapes and Cityscapes → Mapillary Vistas.

<table>
<thead>
<tr>
<th>UDA Method</th>
<th>Cityscapes → Foggy Cityscapes</th>
<th>Cityscapes → Mapillary Vistas</th>
</tr>
</thead>
<tbody>
<tr>
<td>UniDAPS [75]</td>
<td>58.7 M</td>
<td>185.5 M</td>
</tr>
<tr>
<td>EDAPS-MiT-B2</td>
<td>47.6 M</td>
<td>104.9 M</td>
</tr>
</tbody>
</table>

Table 4: Comparison of the mPQ for source-only training, UDA (SYNTHIA → Cityscapes), and supervised oracle training (Cityscapes) along with the relative UDA performance $mPQ_{Rel} = mPQ_{UDA} / mPQ_{Sup}$.

<table>
<thead>
<tr>
<th>UDA Method</th>
<th>mPQ_{Src-Only}</th>
<th>mPQ_{UDA}</th>
<th>mPQ_{Sup}</th>
<th>mPQ_{Rel}</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVRN [25]</td>
<td>20.1</td>
<td>32.1</td>
<td>47.7</td>
<td>67.3%</td>
</tr>
<tr>
<td>UniDAPS [75]</td>
<td>18.3</td>
<td>33.0</td>
<td>51.9</td>
<td>63.6%</td>
</tr>
<tr>
<td>EDAPS (Ours)</td>
<td>21.6</td>
<td>41.2</td>
<td>56.6</td>
<td>72.7%</td>
</tr>
</tbody>
</table>

Table 5: Efficiency comparison on an RTX 2080 Ti.

<table>
<thead>
<tr>
<th>Network architecture</th>
<th>#Parameters</th>
<th>Inference Speed</th>
<th>mPQ_{SYN→CS}</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVRN [25]</td>
<td>185.5 M</td>
<td>0.36 fps</td>
<td>32.1</td>
</tr>
<tr>
<td>UniDAPS [75]</td>
<td>58.7 M</td>
<td>7.24 fps</td>
<td>33.0</td>
</tr>
<tr>
<td>EDAPS w/ MiT-B2</td>
<td>47.6 M</td>
<td>9.03 fps</td>
<td>37.1</td>
</tr>
<tr>
<td>EDAPS w/ MiT-B3</td>
<td>67.5 M</td>
<td>7.65 fps</td>
<td>38.9</td>
</tr>
<tr>
<td>EDAPS w/ MiT-B5</td>
<td>104.9 M</td>
<td>5.84 fps</td>
<td>41.2</td>
</tr>
</tbody>
</table>

We compare EDAPS with previous works from an efficiency viewpoint in Tab. 5. EDAPS with a MiT-B5 backbone (default) increases the inference speed by a factor of 16 compared to CVRN, demonstrating the efficiency of EDAPS. Compared to UniDAPS, EDAPS w/ MiT-B5 reaches 80% of its inference speed, which is due to the additional instance decoder of EDAPS. However, given the large gains in the quality of the adapted panoptic segmentation, this is an acceptable trade-off. Even with a smaller MiT-B2 backbone, EDAPS outperforms UniDAPS by +4.1 mPQ on SYNTHIA → Cityscapes while requiring fewer parameters and with faster inference time (9.0 vs. 7.2 fps).

4.3. Study of Panoptic Architectures for UDA

To gain insights into the influence of the network architecture on the UDA performance, we compare the four principal architectures from §3.3. For all network variants, we use the enhanced network components of EDAPS, i.e., the transformer encoder and the domain-robust decoder as well.
as the same enhanced UDA strategy for a fair comparison.

Tab. 6 shows that S-Net achieves the lowest mPQ of 34.0. In contrast, M-Net achieves 38.1 mPQ, which is +4.1 mPQ better. This demonstrates the importance of task-specific networks (components) to learning task-specific knowledge in the UDA setting. Still, sharing features across tasks in a common encoder for UDA is useful, as shown by M-Dec-BU, which gains a performance improvement of +0.9 mPQ over M-Net. However, the symmetric architecture with bottom-up decoders for both semantics and instances is sub-optimal, as can be seen when comparing M-Dec-BU with M-Dec-TD. With a top-down decoder for instance segmentation, a significant improvement of +2.2 mPQ can be achieved over M-Dec-BU. In particular, the instance mAP improves by +16.8. This shows that the top-down approach is more domain-robust for instance segmentation than the bottom-up approach.

The network study in Tab. 6 also allows a fair comparison of EDAPS and CVRN as M-Net uses the CVRN architecture with the enhanced components of EDAPS, i.e., Transformer encoder, Mean Teacher, RCS, and FD. Therefore, M-Net has a higher 38.1 mPQ compared to the original CVRN 32.1 mPQ. Still, EDAPS outperforms M-Net by +3.1 mPQ. These improvements come from the joint training of EDAPS for semantic and instance segmentation (compared to disjoint M-Net training), allowing the shared encoder to learn better domain-robust instance and semantic features. In particular, EDAPS improves the instance mAP by +11.1 over M-Net demonstrating that particularly the instance head benefits from the shared features. The class-wise comparison in Tab. 7 reveals that EDAPS mostly benefits difficult thing-classes such as rider or motorbike, showing that shared features are particularly important for instances that are difficult to adapt.

### 4.4. UDA Ablation Study

To better understand the influence of the UDA components that we newly introduced to panoptic UDA, we ablate them in Tab. 8. While the source-only model performs at 21.6 mPQ, the UDA baseline with basic self-training achieves a performance of 37.5 mPQ. This is already higher than previous state-of-the-art methods, emphasizing the strength of the proposed EDAPS architecture. The self-training can be improved by +1.4 mPQ using an EMA teacher for pseudo-label generation. ImageNet Thing-Class Feature Distance (FD) further increases performance by +0.8 mPQ. When further integrating Rare Class Sampling (RCS), the performance gains another +1.5 mPQ. This shows that the EMA teacher, FD and RCS are valuable components for panoptic UDA. Empirically, we found that additional instance pseudo-labels do not improve the performance of EDAPS. We assume that semantic self-training is sufficient to bridge the domain gap at the level of the shared encoder features. For simplicity, we therefore only use semantic pseudo-labels.

### 5. Conclusions

Previous approaches to domain-adaptive panoptic segmentation either follow inefficient and expensive adaptation techniques, or their network architecture and training strategies are influenced by supervised learning. In this work, we addressed these issues by carefully selecting the network design and training schemes tailored for UDA and proposed EDAPS, an efficient domain-adaptive panoptic segmentation network that surpasses prior art by a large margin. Furthermore, we provided a detailed analysis of the various design choices for enhancing the panoptic UDA performance. When compared with previous methods on UDA panoptic segmentation, EDAPS shows significant performance gains of 20% on SYNTHIA-to-Cityscapes and even 72% on the more challenging SYNTHIA-to-Mapillary Vistas. We believe that EDAPS as a network architecture will facilitate benchmarking future UDA strategies in panoptic segmentation, making domain-adaptive panoptic segmentation more usable in practice.
References


[50] Lorenzo Porzi, Samuel Rota Bulo, Aleksander Colovic, and Peter Kontschieder. Seamless scene segmentation. In *Proc-


