Shallow Features Guide Unsupervised Domain Adaptation for Semantic Segmentation at Class Boundaries

Adriano Cardace   Pierluigi Zama Ramirez   Samuele Salti   Luigi Di Stefano

Department of Computer Science and Engineering (DISI)

University of Bologna, Italy

{adriano.cardace2, pierluigi.zama}@unibo.it

Abstract

Although deep neural networks have achieved remarkable results for the task of semantic segmentation, they usually fail to generalize towards new domains, especially when performing synthetic-to-real adaptation. Such domain shift is particularly noticeable along class boundaries, invalidating one of the main goals of semantic segmentation that consists in obtaining sharp segmentation masks.

In this work, we specifically address this core problem in the context of Unsupervised Domain Adaptation and present a novel low-level adaptation strategy that allows us to obtain sharp predictions. Moreover, inspired by recent self-training techniques, we introduce an effective data augmentation that alleviates the noise typically present at semantic boundaries when employing pseudo-labels for self-training. Our contributions can be easily integrated into other popular adaptation frameworks, and extensive experiments show that they effectively improve performance along class boundaries.

1. Introduction

Semantic segmentation is the process of assigning a class to each pixel of an image. Recently, convolutional neural networks have proven to be highly effective in solving this challenging visual task [40, 5, 32, 37], leading to ever-increasing interest in the deployment of semantic segmentation models in spaces as diverse as autonomous driving, robotics, and medicine. However, training a semantic segmentation network requires a large amount of pixel-wise annotated data, which are tedious, time-consuming, and expensive to collect. Moreover, current models often fail to generalize toward new domains, an issue that cannot be overlooked in many relevant real-world applications. Indeed, performance often drops when models are tested on new scenarios, especially when there exists a domain gap between the training (source) and test (target) im-
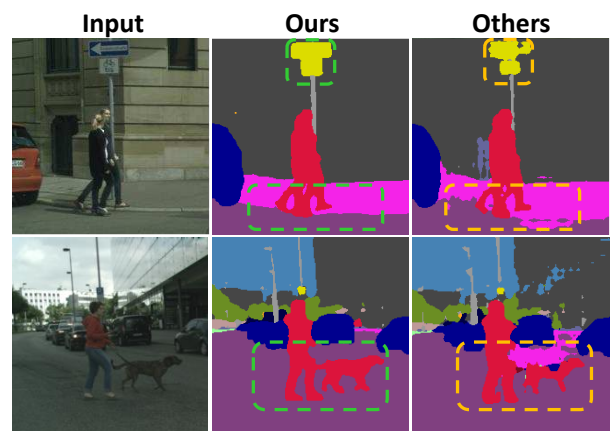

Figure 1: Given in input an RGB image (left-most column), our model produces sharp predictions along class boundaries (central column), while a model trained on translated images (right-most column) exhibits severe noise.

ages. For instance, in autonomous driving settings, object appearances may drastically change when training and testing across different cities, leading to severe segmentation errors. This problem is even more pronounced when relying on synthetic data generated by computer graphics, such as video games [36] or 3D simulations [38], that could otherwise be advantageously exploited to easily obtain large amounts of labeled data.

Unsupervised Domain Adaptation (UDA) [48] aims at minimizing the impact of the domain gap under the assumption that no ground-truth annotations are available for the target domain. In the last few years, several UDA techniques have been proposed for the task of semantic segmentation [20, 44, 35, 19, 50, 2]. However, all these methods ignore the main goal of semantic segmentation, which is to obtain sharp prediction masks and only focus in the feature adaptation part. For this reason, previous works can correctly segment out coarse blobs of large elements in a scene

such as cars or buildings, while they provide inaccurate segmentation masks along class boundaries as shown in Fig. 1.

On the other hand, in the supervised semantic segmentation setting, a large amount of works focus on obtaining sharp predictions [9, 23, 3, 53, 13]. This is commonly done by better integrating low-level features into high-level features since modern segmentation architectures discard spatial information with down-sampling operations such as max-pooling or strided convolution due to memory and time constraints. Following the supervised setting, we argue that this line of research should also be pursued for the UDA case to obtain sharp predictions across domains even though target labels are not available. Our approach, also leverages on low-level features to seek this goal, and we introduce a novel low-level adaptation strategy specifically for the UDA scenario. More precisely, we enforce alignment of low-level features exploiting an auxiliary task that can be solved for both domains in a self-supervised fashion, intending to make them more transferable. By doing this, we enable the possibility to exploit shallow features to refine the coarse segmentation masks for both the source and target domains. To achieve this, we estimate a 2D displacement field from the aligned shallow features that, for each spatial location of the predicted coarse feature map, specifies the direction where the representation for that patch is less ambiguous (i.e. at centre of the semantic object). Our intuition is that when the coarse feature map is bi-linearly up-sampled to regain the target resolution, the feature representation of those patches corresponding to semantic boundaries in the input image is mixed up, as it contains semantic information belonging to different classes. Thanks to the estimated 2D displacement field, however, we refine each patch representation according to the features coming from the center of the object, which are less prone to be influenced by other classes as they lay spatially far from boundaries. This process will be referred later as the feature *warping* process.

Finally, following a recent trend in UDA for semantic segmentation [30, 55, 56, 34], we employ self-training, a technique that foresees the training of a neural network with its own predictions denoted as pseudo-labels. This step allows to implicitly encourage cross-domain feature alignment thanks to the simultaneous training on multiple domains. Yet, differently from previous works that mainly focus on masking incorrect pixels with some heuristics, we propose a novel data augmentation technique aimed at preserving information specifically along class boundaries. In fact, due to the low confidence of the network in the target domain, pixels along edges are usually masked by the aforementioned methods, resulting in a further performance degradation along class boundaries due to the lack of supervision during the self-training process. Thus, we employ a class-wise erosion filtering algorithm that allows us to synthesize new training samples in which only the inner body

of the target objects is preserved and copied into other images. By doing this, all pixels have supervision, and the network is trained to classify correctly edges also in the target domain. Code available at `https://github.com/CVLAB-Unibo/Shallow_DA`. To summarize our contributions are:

- We propose to use shallow features to improve the accuracy of the network along class boundaries in the UDA scenario. This is achieved by computing a displacement field that lets the network use information from the center of semantic blobs.

- We deploy semantic edge detection as an auxiliary task to enforce the alignment of shallow features, which is key to overcome the domain shift when computing the displacement map.

- We introduce an effective data augmentation that selects objects from target images and filters out noise at class boundaries to obtain sharp pseudo-labels.

- We show that our approach achieves overall on par or even state-of-the-art performance in standard UDA for semantic segmentation benchmarks, and more importantly improves predictions along boundaries when compared to previous works.

2. Related Work

2.1. Pixel-level Domain Adaptation

Pixel-level adaptation aims at reducing the visual gap between source and target images. Typically, style and colors are adapted by deploying CycleGANs[57], a generative model able to capture the target style and injecting it into the source images without altering their content. Early works [50, 19] learn such transformation offline, and employ the translated images during training time. Recent approaches instead [28, 14], fuse the translation process into the training pipeline, obtaining an end-to-end framework. [24] extended this approach to obtain a texture-invariant network by training on source images augmented with textures from other natural images. Following recent works, our approach builds upon these techniques. Indeed, we make use of translated images to obtain strong baseline and extract good pseudo-labels when adapting from synthetic to target.

2.2. Adversarial Learning

The goal of adversarial training in the context of Domain Adaptation is to align the distributions of source and target images so that the same classifier can be seamlessly applied on a shared feature extractor. Adaptation can be forced either in feature space [47] or in output-space [44]. Many extensions of [44] have been introduced. [49] proposed to align differently classes based on their intra-class variability
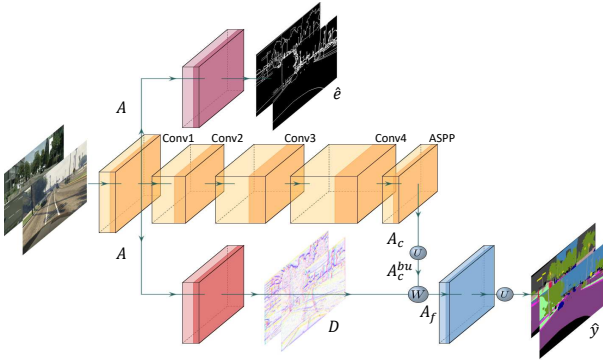
Figure 2: Illustration of our architecture in the adaptation step. Given an RGB input image, the network learns to extract semantic edges from shallow features. From the same feature map, a 2D displacement map is estimated in order to guide the warping of down-sampled deep features, which lacks of fine-grained details.

in their appearance. Other works deploy adversarial learning to minimize the entropy of the target classifier [46] or to perform feature perturbation [52]. In our work, since training a network adversarially is notoriously a difficult and unstable process [39], we avoid it.

2.3. Self-Training

A recent line of research focuses on self-training [27] thanks to its effectiveness and simplicity. This approach is based on the idea of producing pseudo-labels for the target domain and use them to capture domain-specific characteristics. [58] proposes an algorithm to filter out wrong pixels with some confidence thresholds. Similarly, [30] extended the idea by introducing an instance adaptive algorithm to improve the quality of pseudo-label. [55] proposes to use pseudo-labels to minimize the discrepancy between two classifiers, while [31] tries to minimize both the inter-domain and intra-domain gap with the support of the pseudo-labels. Differently, [43] synthesizes new training samples by embedding objects from source images into the target ones. Inspired by these recent trends, we adopt self-training to align shallow features and guide the warping process across domains. Differently from previous approaches, however, we synthesise new training pairs enriching images of both domains with target objects to improve segmentation quality on class boundaries.

3. Method

In UDA for semantic segmentation we are given image-labels pairs $\{x_s^i, y_s^i\}_{i=1}^M$ for a source domain $\mathcal{S}$, while only images $\{x_t^i\}_{i=1}^N$ are available for a target domain $\mathcal{T}$. The goal consists in predicting pixel-wise classification masks

for target images. Our proposed framework comprises several components, as depicted in Fig. 2. A standard backbone (yellow branch) produces a coarse feature map A_c from an image. A semantic edge extractor (top purple branch) estimates semantic edges $\hat{e}$, given the activation map A produced by the first convolutional block of the backbone. The same shallow features are processed by another convolutional block (bottom red branch) to obtain a 2D displacement map, D. Then, A_c is up-sampled to the same size as D and it is refined according to D to produce a fine-grained feature map A_f. Finally, one last convolutional block that acts as a classifier is applied to produce a C-dimensional vector for each pixel, with C being the number of classes, and a final bi-linear up-sampling yields a prediction map of the same size of the input. We detail each component in the following subsections.

3.1. Low-level adaptation

Learning transferable shallow features. We introduce an auxiliary task to push the network to learn domain-invariant features that include details on objects boundaries already from early layers. Given the feature map A, a convolutional block γ is applied to predict an edge map $\hat{e}$. Ground truths e are obtained by the Canny edge detector [1] applied directly on semantic annotations for the source domain and on pseudo-labels for the target domain, so that only semantic boundaries are considered. A binary-cross entropy loss is minimized for batches including images from both domains:

$$
\begin{aligned}
\hat{e} &= \gamma(A), \\
\mathcal{L}_{edge} &= \sum_h^H \sum_w^W e^{(h,w)} \log \hat{e}^{(h,w)} \\
&\quad + (1 - e^{(h,w)}) \log(1 - \hat{e}^{(h,w)})
\end{aligned}
\tag{1}
$$

Hence, we enforce the auxiliary semantic edge detection task for the very first layers of the network only, rather than, as in typical multi-task learning settings such as [16, 10, 42], at a deeper level, where features are more task-dependent. We believe this design choice to be key for a good generalization for three reasons. First, trying to solve this task from shallow layers guides the network to explicitly reason about object shapes from the beginning, rather than solely texture and colors as typically done by CNNs [17]. Second, solving an auxiliary task for both domains forces the network to learn a shared feature representation, which naturally leads to aligned distributions. Consequently, the displacement field generated from the shallow features is effective also in the target domain, and it can be directly exploited at a deeper level to recover fine-grained details. Finally, the peculiar choice of semantic edge detection is directly beneficial to estimate a displacement field that mainly focuses on

edges, making the following warping process more effective where the network is uncertain. We refer to the supplementary material for ablations on the alignment performed at different levels.

Feature warping. One of the contributions of our method is to refine the bi-linearly up-sampled coarse feature map A_c, hereafter A_c^{bu}, to obtain a fine-grained feature map A_f that better captures the correct class for pixels laying in the boundary regions. The refinement is guided by a 2D displacement field D obtained from the domain-invariant shallow features computed by the first convolutional block of the backbone. The displacement field indicates for each location of A_c^{bu} where the network should look to recover the correct class information, namely the direction that better characterize that patch. We estimate the 2D displacement map D by applying a convolutional block to the aligned shallow features A that are aligned as described above.

Our intuition is that, due to the unavoidable side-effect of the down-sample operations in the forward pass, the representation of those elements in A_c whose receptive field includes regions at class boundaries in the original image, contains ambiguous semantic information. Indeed, when A_c is bi-linearly up-sampled, patches that receive contributions from ambiguous coarse patches inherit such ambiguity. However, in the higher resolution feature map A_c^{bu} it may be possible to compute a better, unambiguous representation for some of the patches, *i.e.* those now laying entirely in a region belonging to one class. The correct semantic information may be available in the nearby high-resolution patches closer to the semantic blob centers. Thus, each feature vector at position p on a standard 2D spatial grid of A_c^{bu}, is mapped to a new position $\hat{p} = p + D(p)$, and we use a differentiable sampling mechanism [22] to approximate the new feature vector representation for that patch:

$$A_f(p) = \sum_{p_l \in \mathcal{N}(\hat{p})} w_{p_l} A_c^{bu}(p_l) \qquad (2)$$

where w_{p_l}, are the bi-linear kernel weights obtained from D and $\mathcal{N}$ the set of neighboring pixels. Hence, Eq. (2) defines a backward warping operation in feature space, where A_f is obtained by warping A_c^{bu} according to D. Finally, the fine-grained feature map A_f is fed to the classifier to obtain the final prediction that is up-sampled by a factor of 2 to regain the input image resolution. We minimize the cross entropy loss using annotations for the source domain and pseudo-labels for the target domain:

$$\mathcal{L}_{sem} = \sum_{h=1}^{H} \sum_{w=1}^{W} \sum_{c=1}^{C} y^{(h,w,c)} \log \hat{y}^{(h,w,c)} \qquad (3)$$

3.2. Data Augmentation for Self-Training

Inspired by [54, 15, 18, 43], we use a pre-trained model to select objects based on predictions on target images and

Figure 3: Given a target image prediction pair (top-left) and a source training pair (top-right), we select classes such as *person* (bottom-left) and apply our class-wise data augmentation pipeline to synthesise a new training pair (bottom-right). The selected shapes are eroded before being pasted.

paste them over source images (see Fig. 3). Peculiarly, our self-training approach relies on a data augmentation process that selects objects from the target scenes rather than the source ones as done [43]. Although selecting source objects may be useful to reduce the unbalanced distributions of classes, it is a sub-optimal choice since the network would be still trained to identify shapes and details peculiar to the source domain, which are different to those found at inference time for the target images. We instead use pseudo-labels to cut objects from the target scenes and paste them into source or target images, forcing the network to look for these patterns on both domains. However, due to the inherent noise of pseudo-labels we need to filter out noisy predictions. In particular, we aim at removing object boundaries as they typically exhibit classification errors and tend to be localized rather inaccurately. Given a target image x_t and its associated predictions $\hat{y}_t$, we compute a binary mask B_c for each class $c \in C^*$, where C^* denotes a random subset of the considered classes. We exclude classes such as *"road"* and *"building"* to avoid occlusion of the whole scene and to counteract the unbalanced distributions of classes, and only use object instances such as as *"car"* and *"poles"*. This categorization is similar to the one used in [49], and can be easily adapted to different datasets. We refer to the supplementary material for the set of classes we used in each experiment. For each spatial location p, B_c has value 1 if p is assigned to class c, 0 otherwise. Then, we apply an erosion operation, $\ominus$, with a 5×5 structuring element k to each class mask B_c. To obtain the set of pixels to be copied from the target image to a randomly selected source image we apply the union set operator to all masks:

$$B = \bigcup_{c \in C^*} B_c \ominus k, \qquad (4)$$

$$x^p = \begin{cases} x_t^p, & B^p = 1 \\ x_s^p, & B^p = 0 \end{cases}, y^p = \begin{cases} \hat{y}_t^p, & B^p = 1 \\ y_s^p, & B^p = 0 \end{cases} \quad (5)$$

The new synthesised training pairs are very often enriched with fine-grained details from the target domain. Indeed, as shown in Fig. 3, thanks to our data augmentation pipeline, only the inner part of an object is preserved while edges are discarded, producing sharp pseudo-labels even at class boundaries. The whole data augmentation process is applied offline before training, therefore it does not have any impact on the training time.

3.3. Training Procedure

The whole pipeline can be summarised in 3 simple steps. We start with the *initialization* step to train our baseline model (i.e. the yellow backbone of Fig. 2) on the source domain only. We follow standard practices [28, 49, 45, 34, 51] and, for synthetic-to-real adaptation, we utilize domain-translated source images provided by [28]. We deploy this baseline to produce pseudo-labels for the target domain and obtain an augmented mixed dataset as detailed in Sec. 3.2.

Then, we perform the *adaptation* step: we train the model illustrated in Fig. 2 that empowers our additional modules for low-level alignment as explained in Sec. 3.1. It is important to highlight that the proposed data augmentation extracts objects from only target images and pastes them on images on both domains. Hence, at this stage, the training is done simultaneously on both domains. The training loss is as follows:

$$\mathcal{L} = \mathcal{L}_{sem} + \lambda \mathcal{L}_{edge} \quad (6)$$

with λ set to 0.1 in all experiments.

Finally, we use the predictions from the model trained in the previous step to synthesise new training pairs by following again the procedure detailed in Sec. 3.2. This allows us to distill the knowledge and the good precision along class boundaries of the previously enhanced model into a lighter segmentation architecture as the one used in the first step. We do this to avoid the introduction of additional modules at inference time. Differently from the adaptation step however, we apply our data algorithm using solely images from the target domain. Indeed, as we are now at the third and final stage, we expect pseudo-labels to be less noisy compared to the previous step, and training only on the target domain allows to capture domain specific characteristic. We denote this third step as the *distillation* step.

4. Implementation

4.1. Architecture

According to standard practice in UDA for semantic segmentation [44, 7, 28, 55, 49, 47, 24], we deploy the

Deeplab-v2 [5] architecture, with a dilated ResNet101 pretrained on ImageNet [12] and output stride 8. The ASPP [5] module acts as classifier. We use this architecture for both the initialization step and the distillation step. For more details on the additional modules of the adaptation step we refer to the supplementary material.

4.2. Training Details

Our pipeline is implemented in PyTorch [33] and trained on a single NVIDIA 2080Ti GPU with 12GB of memory. We train for 20 epochs in the first two steps, while we set the number of epochs to 25 for the final distillation with batch size 4 in all cases. We use random scaling, random cropping at 1024×892, and color jittering in our data augmentation pipeline. Akin to previous works, we freeze Batch-Normalization layers [21] while performing the initialization and adaptation step. For the last step, instead, we activate these layers. We adopt the One Cycle learning rate policy [41] for each training, with maximum learning rate 10^{-3} and SGD as optimizer.

5. Experiments

5.1. Datasets

We test our method on both synthetic-to-real and real-to-real adaptation. We set GTA [36] or SYNTHIA [38] as source datasets and Cityscapes [11] as target for the former setting, while we use Cityscapes as source and the NTHU [8] dataset as target for the latter. GTA5 is a synthetic dataset that contains 24,966 annotated images of 1914×1052 resolution. As for SYNTHIA, we use the SYNTHIA-RAND-CITYSCAPES subset, which is a collection of 9,400 synthetic images with resolution 1280×760. The Cityscapes dataset is a high-quality collection of real images of 2048×1024 resolution. The dataset has 2975 and 500 images for the training and validation split, respectively. For the synthetic-to-real case, we only utilize the training split without labels for training, and test on the validation set as done in previous works [44, 58, 28]. The NTHU dataset is a collection of images taken from four different cities with 2048×1024 resolution: Rio, Rome, Tokyo, and Taipei. For each city, 3200 unlabeled images are available for the adaptation phase, and 100 labeled images for the evaluation. For fair comparison to other models, we compute the mIoU by considering all 19 classes in the GTA5→Cityscapes benchmark, 16 or 13 shared classes for SYNTHIA→Cityscapes, and 13 common classes for the cross-city adaptation setting.

5.2. Synthetic-to-real adaptation

To test our framework, we follow standard practice [44, 58, 28, 55, 46, 7] and report the results for the synthetic-to-real adaptation in the GTA5→Cityscapes and

method	IT	ST	Road	Sidewalk	Building	Walls	Fence	Pole	T-light	T-sign	Vegetation	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	Motorbike	Bicycle	mIoU
AdaptSegNet [44]			86.5	36.0	79.9	23.4	23.3	23.9	35.2	14.8	83.4	33.3	75.6	58.5	27.6	73.6	32.5	35.4	3.9	30.1	28.1	42.4
MaxSquare [7]			88.1	27.7	80.8	28.7	19.8	24.9	34.0	17.8	83.6	34.7	76.0	58.6	28.6	84.1	37.8	43.1	7.2	32.2	34.5	44.3
BDL [28]	✓	✓	88.2	44.7	84.2	34.6	27.6	30.2	36.0	36.0	85.0	43.6	83.0	58.6	31.6	83.3	35.3	49.7	3.3	28.8	35.6	48.5
MRNET [55]	✓	✓	90.5	35.0	84.6	34.3	24.0	36.8	44.1	42.7	84.5	33.6	82.5	63.1	34.4	85.8	32.9	38.2	2.0	27.1	41.8	48.3
Stuff and things [49]	✓	✓	90.6	44.7	84.8	34.3	28.7	31.6	35.0	37.6	84.7	43.3	85.3	57.0	31.5	83.8	42.6	48.5	1.9	30.4	39.0	49.2
FADA [47]	✓	✓	92.5	47.5	85.1	37.6	32.8	33.4	33.8	18.4	85.3	37.7	83.5	63.2	**39.7**	**87.5**	32.9	47.8	1.6	34.9	39.5	49.2
LTIR [24]	✓	✓	92.9	55.0	85.3	34.2	31.1	34.4	40.8	34.0	85.2	40.1	87.1	61.1	31.1	82.5	32.3	42.9	3	36.4	46.1	50.2
Yang et al. [51]	✓	✓	91.3	46.0	84.5	34.4	29.7	32.6	35.8	36.4	84.5	43.2	83.0	60.0	32.2	83.2	35.0	46.7	0.0	33.7	42.2	49.2
IAST [30]		✓	**93.8**	57.8	85.1	**39.5**	26.7	26.2	43.1	34.7	84.9	32.9	88.0	62.6	29.0	87.3	39.2	49.6	**23.2**	34.7	39.6	51.5
DACS† [43]		✓	89.9	39.7	**87.9**	30.7	**39.5**	**38.5**	**46.4**	**52.8**	**88.0**	**44.0**	**88.8**	**67.2**	35.8	84.5	**45.7**	**50.2**	0.0	27.3	34.0	52.1
Ours	✓	✓	91.9	48.9	86.0	38.6	28.6	34.8	45.6	43.0	86.2	42.4	87.6	65.6	38.6	86.8	38.4	48.2	0.0	**46.5**	**59.2**	**53.5**

Table 1: Results on GTA5→Cityscapes. † denotes models pre-trained on MSCOCO [29] and ImageNet [12]. IT: Image Translation; ST: Self-Training.

method	IT	ST	Road	Sidewalk	Building	Walls*	Fence*	Pole*	T-light	T-sign	Vegetation	Sky	Person	Rider	Car	Bus	Motorbike	Bicycle	mIoU	mIoU*
AdaptSegNet [44]			84.3	42.7	77.5	-	-	-	4.7	7.0	77.9	82.5	54.3	21.0	72.3	32.2	18.9	32.3	-	46.7
MaxSquare [7]			77.4	34.0	78.7	5.6	0.2	27.7	5.8	9.8	80.7	83.2	58.5	20.5	74.1	32.1	11.0	29.9	39.3	45.8
BDL [28]	✓	✓	86.0	46.7	80.3	-	-	-	14.1	11.6	79.2	81.3	54.1	27.9	73.7	42.2	25.7	45.3	-	51.4
MRNET [55]	✓	✓	83.1	38.2	81.7	9.3	1.0	35.1	30.3	19.9	82.0	80.1	62.8	21.1	84.4	37.8	24.5	**53.3**	46.5	53.8
Stuff and things [49]	✓	✓	83.0	44.0	80.3	-	-	-	17.1	15.8	80.5	81.8	59.9	33.1	70.2	37.3	28.5	45.8	-	52.1
FADA [47]	✓	✓	84.5	40.1	83.1	4.8	0.0	34.3	20.1	27.2	84.8	84.0	53.5	22.6	85.4	43.7	26.8	27.8	45.2	52.5
LTIR [24]	✓	✓	**92.6**	53.2	79.2	-	-	-	1.6	7.5	78.6	84.4	52.6	20.0	82.1	34.8	14.6	39.4	-	49.3
Yang et al. [51]	✓	✓	82.5	42.2	81.3	-	-	-	18.3	15.9	80.6	83.5	61.4	33.2	72.9	39.3	26.6	43.9	-	52.4
IAST [30]		✓	81.9	41.5	83.3	17.7	**4.6**	32.3	**30.9**	28.8	83.4	85.0	65.5	30.8	**86.5**	38.2	**33.1**	52.7	**49.8**	**57.0**
DACS† [43]		✓	80.6	25.1	81.9	**21.5**	2.6	**37.2**	22.7	24.0	83.7	**90.8**	**67.6**	**38.3**	82.9	38.9	28.5	47.6	48.3	54.8
Ours	✓	✓	90.4	**51.1**	**83.4**	3.0	0.0	32.3	25.3	**31.0**	**84.8**	85.5	59.3	30.1	82.6	**53.2**	17.5	45.6	48.4	56.9

Table 2: Results on SYNTHIA→Cityscapes. † denotes models pre-trained with MSCOCO [29] and ImageNet [12]. IT: Image Translation; ST: Self-Training. The 13 classes with * are used to compute mIoU*.

SYNTHIA→Cityscapes benchmarks in Tab. 1 and Tab. 2 respectively. We obtain state-of-the-art performance in the former setting, surpassing also recent methods such as [30] that performs many iterations of self-training. We also improve over [43] for GTA5→Cityscapes, which, differently from all other methods, pre-trains the baseline network not only on ImageNet[12] but also on MSCOCO[29]. We argue that pre-training on more tasks and real annotated data notably improves the baseline performance of the synthetic-to-real benchmark. For GTA5→Cityscapes, we note that, thanks to our low-level adaptation, we can boost performances for fine-detailed classes such as *Bicycle* and *Motorcycle*. Regarding SYNTHIA→Cityscapes, we obtain competitive performance, showing that our method can work also in this challenging scenario in which the source synthetic domain exhibits many bird's-eye views that are very different from the one in Cityscapes. Indeed our method is only slightly inferior to IAST[30] and again superior to [43] that performs a similar data augmentation.

5.3. Cross-city adaptation

We report in Tab. 3 our performance for the real-to-real setting. Our proposal shows great results, confirming the generalization properties of our contributions on diverse set-tings. We improve performance with respect to previous works for all the cities. Our model achieves 60% in mIoU in Rome, which is likely the most similar to the German cities used in the Cityscapes dataset. Nonetheless, we achieve strong results even for more distant domains, e.g. as in the case of Taipei, improving by 7.8% with respect to the model trained only on the source domain. For the Cross-city adaptation setting, differently from the other settings, we use images of both domains in our *distillation* step to exploit the perfect annotations available in the similar source domain.

5.4. Ablation Studies

In this section, we analyze the contribution provided by each component of our framework and motivate our design choices. In Tab. 4 we detail the results for both GTA5→Cityscapes and SYNTHIA→Cityscapes. The first row reports the performance obtained using only translated source domain images. This is nowadays a common building block of many UDA frameworks, and we also consider it our baseline on which we build our pipeline. In the adaptation section instead, we isolate both our contributions and use the model trained in the initialization step to extract pseudo-labels for the target domain as explained in Sec. 3.2 and train on both domains simultaneously. When apply-

City	Method	ST	road	sidewalk	building	light	sign	veg.	sky	person	rider	car	bus	motor	bike	mIoU (%)
Rome	Source only		85.9	40.0	86.0	9.0	25.4	82.4	90.5	38.8	25.9	81.6	52.0	48.7	6.7	51.9
	CBST [58]	✓	87.1	43.9	89.7	14.8	47.7	85.4	90.3	45.4	26.6	85.4	20.5	49.8	10.3	53.6
	AdaptSegNet [44]		83.9	34.2	88.3	18.8	40.2	86.2	93.1	47.8	21.7	80.9	47.8	48.3	8.6	53.8
	MaxSquare [7]		80.0	27.6	87.0	20.8	42.5	85.1	92.4	46.7	22.9	82.1	53.5	50.8	8.8	53.9
	FADA [47]	✓	84.9	35.8	88.3	20.5	40.1	85.9	92.8	56.2	23.2	83.6	31.8	53.2	14.6	54.7
	Ours	✓	89.4	48.2	87.5	26.3	37.2	83.1	90.7	55.2	42.1	84.8	66.6	59.2	11.1	60.1
Rio	Source only		80.4	53.8	80.7	4.0	10.9	74.4	87.8	48.5	25.0	72.1	36.1	30.2	12.5	47.4
	CBST [58]	✓	84.3	55.2	85.4	19.6	30.1	80.5	77.9	55.2	28.6	79.7	33.2	37.6	11.5	52.2
	AdaptSegNet [44]		76.2	44.7	84.6	9.3	25.5	81.8	87.3	55.3	32.7	74.3	28.9	43.0	27.6	51.6
	MaxSquare [7]		70.9	39.2	85.6	14.5	19.7	81.8	88.1	55.2	31.5	77.2	39.3	43.1	30.1	52.0
	FADA [47]	✓	80.6	53.4	84.2	5.8	23.0	78.4	87.7	60.2	26.4	77.1	37.6	53.7	42.3	54.7
	Ours	✓	86.6	63.3	82.3	10.3	19.8	73.9	88.4	57.5	41.3	78.1	51.5	40.0	19.4	54.8
Tokyo	Source only		86.0	38.8	76.6	11.7	12.3	80.0	89.5	44.9	28.0	71.5	4.7	27.1	42.2	47.2
	CBST [58]	✓	85.2	33.6	80.4	8.3	31.1	83.9	78.2	53.2	28.9	72.7	4.4	27.0	47.0	48.8
	AdaptSegNet [44]		81.5	26.0	77.8	17.8	26.8	82.7	90.9	55.8	38.0	72.1	4.2	24.5	50.8	49.9
	MaxSquare [7]		79.3	28.5	78.3	14.5	27.9	82.8	89.6	57.3	31.9	71.9	6.0	29.1	49.2	49.7
	FADA [47]	✓	85.8	39.5	76.0	14.7	24.9	84.6	91.7	62.2	27.7	71.4	3.0	29.3	56.3	51.3
	Ours	✓	87.8	41.0	79.6	20.3	24.2	80.2	90.0	62.3	30.8	74.0	6.4	32.7	50.0	52.4
Taipei	Source only		85.0	38.1	82.2	17.8	8.9	75.2	91.4	23.9	19.6	69.2	45.9	49.4	16.0	47.9
	CBST [58]	✓	86.1	35.2	84.2	15.0	22.2	75.6	74.9	22.7	33.1	78.0	37.6	58.0	30.9	50.3
	AdaptSegNet [44]		81.7	29.5	85.2	26.4	15.6	76.7	91.7	31.0	12.5	71.5	41.1	47.3	27.7	49.1
	MaxSquare [7]		81.2	32.8	85.4	31.9	14.7	78.3	92.7	28.3	8.6	68.2	42.2	51.3	32.4	49.8
	FADA [47]	✓	86.0	42.3	86.1	6.2	20.5	78.3	92.7	47.2	17.7	72.2	37.2	54.3	44.0	52.7
	Ours	✓	95.6	78.9	94.3	45.9	70.3	93.0	96.2	63.3	51.3	90.5	83.6	84.8	56.5	55.7

Table 3: Results for the Cross-City experiments. ST: Self-Training.

Step	IT	ST	A	W	D	GTA mIoU	Synthia mIoU
Initialization	✓				$\mathcal{S}$	47.3	41.6
Adaptation	✓	✓			$\mathcal{S},\mathcal{T}$	49.8	43.5
	✓	✓	✓		$\mathcal{S},\mathcal{T}$	52.0	46.4
	✓	✓	✓	✓	$\mathcal{S},\mathcal{T}$	52.6	46.9
Distillation	✓	✓	✓		$\mathcal{T}$	53.5	48.4
Oracle					$\mathcal{T}$	63.8	65.1

Table 4: Ablation studies on GTA5→Cityscapes (second-to-last) and SYNTHIA→Cityscapes (last) columns. IT: image translation; ST: Self-Training; W: low-level adaptation; A: Data Augmentation; D: Training domain.

ing a naive self-training strategy (i.e. training directly on pseudo-labels) we already obtain a significant boost (+2.5% and +1.9%) respectively. However, when deploying the proposed data augmentation (row 3), we observe an even greater boost: +4.8% for both settings. This clearly demonstrates the effectiveness of our data augmentation and its applicability to diverse scenarios. Then, applying the proposed low-level adaptation (row 4) also yields an additional contribution overall: about +0.6% on top of the data augmentation version. We argue that is noticeable, especially when performances are already high, as in our case, and the strongest competitors are all within a narrow window. Finally, in row 5, we distill our full model (i.e. row 4) into a simple Deeplab-v2 for efficient inference time and apply once again the proposed data augmentation. Remarkably,

this further improves performance with respect to the distilled model and avoids the typical pseduo-labels overfitting behavior when employing many steps of self-training.

Moreover, to motivate our intuition that shallow features are amenable to guide the warping process, we compare the results obtained by applying our adaptation step in the GTA5→Cityscapes setting at the three different levels of the backbone before the last module achieving 52.6%, 51.6%, and 51.8% mIoU for layers *Conv1*, *Conv2*, and *Conv3* respectively. Thus, the best result is achieved by using the first convolutional block of the architecture, while on *Conv2* and *Conv3* results are comparable (see Fig. 2 for layer names).
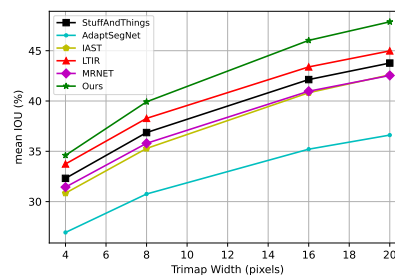
5.5. Performance Along Class Boundaries

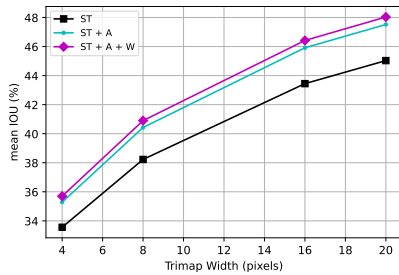Figure 4: mIOU on GTA5→Cityscapes as a function of trimap band width around class boundaries.

Figure 5: mIOU on GTA5→Cityscapes as a function of trimap band width around class boundaries. We report results for the three versions of the *adaptation* step of Tab. 4.

In this section, we test the segmentation accuracy with the trimap experiment [6, 26, 4, 25] to quantify the accuracy of the proposed method along semantic edges. Specifically, we evaluate in terms of mIoU pixels within four bandwidths (4, 8, 16, 20 pixels) around class boundaries (trimaps). We first compare our final model against other frameworks in Fig. 4. We observe that our method is more accurate w.r.t. all other competitors in all the tested bandwidths, validating our main goal that is improving precision along class boundaries. We also highlight that although the green line is obtained from a distilled model (row 5 of Tab. 4), that does not include the additional modules presented in Sec. 3.1, it is still able to maintain strong performances at semantic boundaries thanks to the precise pseudo-labels extracted from the adaptation step. We refer to supplementary materials for some qualitative examples. Then, we assess in Fig. 5 how our contributions affect performances on semantic boundaries. To this end, we repeat the same trimap experiment using the intermediate steps of our pipeline i.e. row 2, 3, and 4 of Tab. 4. When applying all our contributions (purple line), we are able to improve by a large margin over the self-training strategy (black line) confirming that the additional modules account for an improvement along semantic edges. Furthermore, activating the low-level adaptation strategy maintains its improvements along semantic edges over the data augmentation only version (cyan line), leading to better pseudo-labels for the distillation step.

5.6. Comparison with other data augmentations

We compare our data augmentation, one of our main contributions, with the one introduced in [43]. More specifically, we apply this data augmentation in the adaptation step as in row 3 of Tab. 4, i.e. without the low-level adaptation modules to isolate the data-augmentation effect. We augment target images randomly pasting objects from the source domain, using the open source implementation of [43]. With this strategy, we only obtain 51.0% in terms of mIoU, while with our technique the mIoU raises to 52.2%,
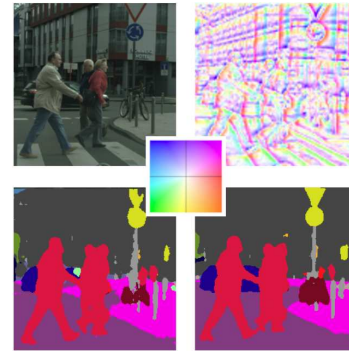

Figure 6: Top left: input target image. Top right: estimated 2D displacement. Bottom left: semantic map from a model trained on translated images. Bottom Right: Our results, improved on class boundaries by using the warping module. Colors and lightness in the middle indicates the warping direction with the corresponding intensity.

confirming our intuition that looking for target instances is more effective than forcing the network to identify source objects as done [43] during the self-training step.

5.7. Displacement map visualization

In this section, we analyze the displacement map learned by the model. As Fig. 6 shows, the 2D map that guides the warping process is consistent with our intuition that the displacement is more pronounced at the boundaries, while areas within regions such as the body of a person, are characterized by a low displacement (i.e. white color). Moreover, we can appreciate that when the warping is applied according to the estimated displacement field (top-right), the contours of small objects such as poles, traffic signs, and persons are better delineated (bottom-right). On the other hand, in the bottom-left mask, these objects are coarsely segmented when using a segmentation model train with translated images only. We also highlight that the displacement field is agnostic to semantic class (it only considers boundaries), and even though it captures other kinds of edges (i.e. not only semantic ones), it leads to computing an average of patches belonging to the same class.

6. Conclusion

In this paper, we have proposed a novel framework for UDA for semantic segmentation that explicitly focuses on improving accuracy along class boundaries. We have shown that we can exploit domain-invariant shallow features to estimate a displacement map used to achieve sharp predictions along semantic edges. Jointly with a novel data augmentation technique that preserves fine edge information during self-training, our approach achieves better accuracy along class boundaries w.r.t. previous methods.

References

[1] J Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698, June 1986. 3

[2] Wei-Lun Chang, Hui-Po Wang, Wen-Hsiao Peng, and Wei-Chen Chiu. All about structure: Adapting structural information across domains for boosting semantic segmentation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2019. 1

[3] Liang-Chieh Chen, Jonathan T. Barron, George Papandreou, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with task-specific edge detection using cnns and a discriminatively trained domain transform. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016. 2

[4] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 8

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, Apr 2018. 1, 5

[6] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018. 8

[7] Minghao Chen, Hongyang Xue, and Deng Cai. Domain adaptation for semantic segmentation with maximum squares loss. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2019. 5, 6, 7

[8] Yi-Hsin Chen, Wei-Yu Chen, Yu-Ting Chen, Bo-Cheng Tsai, Yu-Chiang Frank Wang, and Min Sun. No more discrimination: Cross city adaptation of road scene segmenters. In *ICCV*, 2017. 5

[9] Yifu Chen, Arnaud Dapogny, and Matthieu Cord. Semeda: Enhancing segmentation precision with semantic edge aware loss. *Pattern Recognition*, 108:107557, Dec 2020. 2

[10] Jinwoo Choi, Gaurav Sharma, Samuel Schulter, and Jia-Bin Huang. Shuffle and attend: Video domain adaptation. In *European Conference on Computer Vision*, pages 678–695. Springer, 2020. 3

[11] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 5

[12] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 5, 6

[13] Henghui Ding, Xudong Jiang, Ai Qun Liu, Nadia Magnenat Thalmann, and Gang Wang. Boundary-aware feature propagation for scene segmentation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2019. 2

[14] A. Dundar, M. Y. Liu, Z. Yu, T. C. Wang, J. Zedlewski, and J. Kautz. Domain stylization: A fast covariance matching framework towards domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2020. 2

[15] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 4

[16] Timnit Gebru, Judy Hoffman, and Li Fei-Fei. Fine-grained recognition in the wild: A multi-task domain adaptation approach. In *Proceedings of the IEEE international conference on computer vision*, pages 1349–1358, 2017. 3

[17] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019. 3

[18] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2918–2928, 2021. 4

[19] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. CyCADA: Cycle-consistent adversarial domain adaptation. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1989–1998, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR. 1, 2

[20] Judy Hoffman, Dequan Wang, Fisher Yu, and Trevor Darrell. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv preprint arXiv:1612.02649*, 2016. 1

[21] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015. 5

[22] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and koray kavukcuoglu. Spatial transformer networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. 4

[23] Tsung-Wei Ke, Jyh-Jing Hwang, Ziwei Liu, and Stella X. Yu. Adaptive affinity fields for semantic segmentation. *Lecture Notes in Computer Science*, page 605–621, 2018. 2

[24] Myeongjin Kim and Hyeran Byun. Learning texture invariant representation for domain adaptation of semantic segmentation. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2020. 2, 5, 6

[25] Pushmeet Kohli, Philip HS Torr, et al. Robust higher order potentials for enforcing label consistency. *International Journal of Computer Vision*, 82(3):302–324, 2009. 8

[26] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *Advances in Neural Information Processing Systems*, 24:109–117, 2011. 8

[27] D. Lee. Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, 2013. 3

[28] Yunsheng Li, Lu Yuan, and Nuno Vasconcelos. Bidirectional learning for domain adaptation of semantic segmentation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2019. 2, 5, 6

[29] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. *Lecture Notes in Computer Science*, page 740–755, 2014. 6

[30] Ke Mei, Chuang Zhu, Jiaqi Zou, and Shanghang Zhang. Instance adaptive self-training for unsupervised domain adaptation. *Lecture Notes in Computer Science*, page 415–430, 2020. 2, 3, 6

[31] Fei Pan, Inkyu Shin, Francois Rameau, Seokju Lee, and In So Kweon. Unsupervised intra-domain adaptation for semantic segmentation through self-supervision. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2020. 3

[32] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *CoRR*, abs/1606.02147, 2016. 1

[33] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS 2017 Workshop on Autodiff*, 2017. 5

[34] Sujoy Paul, Yi-Hsuan Tsai, Samuel Schulter, Amit K. Roy-Chowdhury, and Manmohan Chandraker. Domain adaptive semantic segmentation using weak labels. In *European Conference on Computer Vision (ECCV)*, 2020. 2, 5

[35] Pierluigi Zama Ramirez, Alessio Tonioni, and Luigi Di Stefano. Exploiting semantics in adversarial training for image-level domain adaptation. In *2018 IEEE International Conference on Image Processing, Applications and Systems (IPAS)*, pages 49–54. IEEE, 2018. 1

[36] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. *Lecture Notes in Computer Science*, page 102–118, 2016. 1, 5

[37] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, page 234–241, 2015. 1

[38] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1, 5

[39] Tim Salimans, Ian J Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *NIPS*, 2016. 3

[40] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651, Apr 2017. 1

[41] Leslie N. Smith and Nicholay Topin. Super-convergence: very fast training of neural networks using large learning rates. *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, May 2019. 5

[42] Yu Sun, Eric Tzeng, Trevor Darrell, and Alexei A. Efros. Unsupervised domain adaptation through self-supervision. *arXiv preprint arXiv:1909.11825*, 2019. 3

[43] Wilhelm Tranheden, Viktor Olsson, Juliano Pinto, and Lennart Svensson. Dacs: Domain adaptation via cross-domain mixed sampling. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1379–1389, January 2021. 3, 4, 6, 8

[44] 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. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun 2018. 1, 2, 5, 6, 7

[45] Yi-Hsuan Tsai, Kihyuk Sohn, Samuel Schulter, and Manmohan Chandraker. Domain adaptation for structured output via discriminative patch representations. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2019. 5

[46] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Perez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2019. 3, 5

[47] Haoran Wang, Tong Shen, Wei Zhang, Lingyu Duan, and Tao Mei. Classes matter: A fine-grained adversarial approach to cross-domain semantic segmentation. In *The European Conference on Computer Vision (ECCV)*, August 2020. 2, 5, 6, 7

[48] Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, Oct 2018. 1

[49] Zhonghao Wang, Mo Yu, Yunchao Wei, Rogerio Feris, Jinjun Xiong, Wen-mei Hwu, Thomas S. Huang, and Honghui Shi. Differential treatment for stuff and things: A simple unsupervised domain adaptation method for semantic segmentation. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2020. 2, 4, 5, 6

[50] Zuxuan Wu, Xintong Han, Yen-Liang Lin, Mustafa Gökhan Uzunbas, Tom Goldstein, Ser Nam Lim, and Larry S. Davis. Dcan: Dual channel-wise alignment networks for unsupervised scene adaptation. *Lecture Notes in Computer Science*, page 535–552, 2018. 1, 2

[51] Jinyu Yang, Weizhi An, Chaochao Yan, Peilin Zhao, and Junzhou Huang. Context-aware domain adaptation in semantic segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 514–524, January 2021. 5, 6

[52] Jihan Yang, Ruijia Xu, Ruiyu Li, Xiaojuan Qi, Xiaoyong Shen, Guanbin Li, and Liang Lin. An adversarial perturbation oriented domain adaptation approach for semantic segmentation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 12613–12620. AAAI Press, 2020. 3

[53] Jianlong Yuan, Zelu Deng, Shu Wang, and Zhenbo Luo. Multi receptive field network for semantic segmentation. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Mar 2020. 2

[54] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6023–6032, 2019. 4

[55] Zhedong Zheng and Yi Yang. Unsupervised scene adaptation with memory regularization in vivo. *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, Jul 2020. 2, 3, 5, 6

[56] Zhedong Zheng and Yi Yang. Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation. *International Journal of Computer Vision*, Jan 2021. 2

[57] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 2

[58] Yang Zou, Zhiding Yu, Xiaofeng Liu, B. V. K. Vijaya Kumar, and Jinsong Wang. Confidence regularized self-training. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2019. 3, 5, 7