Supplementary for DiGA: *Dis*til to *G*eneralize and then *A*dapt for Domain Adaptive Semantic Segmentation

Fengyi Shen[1,2,3], Akhil Gurram[2], Ziyuan Liu[2], He Wang[3]*, Alois Knoll[1]
[1]Technical University of Munich, [2]Huawei Munich Research Center, [3]EPIC Lab, Peking University
[1]fengyi.shen@tum.de, knoll@in.tum.de, [2]{first.last}@huawei.com, [3]hewang@pku.edu.cn

In this Supplementary, we provide extensive experiments for evaluating our DiGA framework as well as additional results obtained from our trained models of DiGA and its variants.

A. Symmetric Knowledge Distillation against Adversarial Training

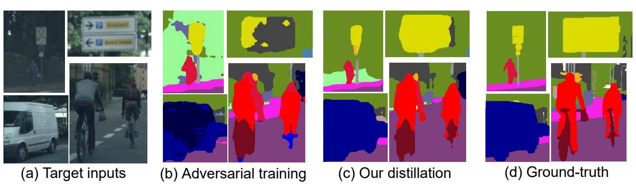

Figure 1. **Visual examples** of the blind alignment issue in adversarial warm-up training and the **comparison of model predictions** between **adversarial training** and our **proposed knowledge distillation** technique given the same target inputs.

Strategy	Adv. [17]	Distil.	Adv. [17]+CrDoMix	Distil.+CrDoMix	Adv. [17]+Distil.+CrDoMix
mIoU	45.2	48.9	47.3	51.1	51.3

Table 1. **Warp-up model comparison** between adversarial training and our knowledge distillation technique w/ and w/o CrDoMix, as well as combining all configurations on GTA5-to-Cityscapes adaptation.

We emphasize and further illustrate that our proposed pixel-wise symmetric knowledge distillation can be a good replacement of adversarial training for warming up UDA segmentation. Without the awareness of the class label of the target data, adversarial learning tries to align target features to their most closely distributed source features but ignores the class-wise relationship. This feature alignment can only ensure that the overall distribution of the features from the source and target domains are indistinguishable, which deviates from the ultimate goal of domain adaptive semantic segmentation, *i.e.*, the feature of one class from the target domain is aligned to the same class from the source domain. Therefore, as presented in Fig. 1(b), the model often gets confused when a prediction should be made between two similar classes, which turn out to be erroneous in many cases. However, as observed in Fig. 1(b), our knowledge distillation technique can effectively alleviate this issue, since the distillation training only involves source domain data and is class-aware.

In Table 1, we quantitatively compare these two warm-up strategies and find out that our pixel-wise symmetric distillation technique is an optimal strategy for warm-up stage training. Out of curiosity, we conduct an additional experiment on GTA5-to-Cityscapes adaptation benchmark to see whether combining our distillation and adversarial training for warm-up can bring a further improvement for the warm-up stage. However, we do not witness a substantial increase, *i.e.*, the mIoU goes up from 51.1 to 51.3.

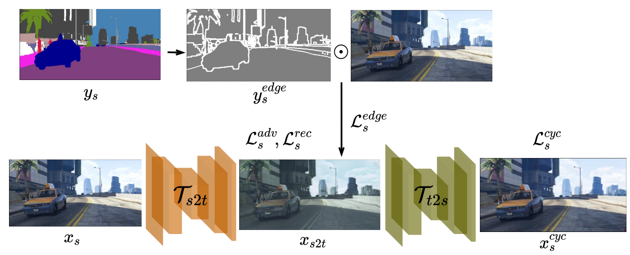

Figure 2. **The training pipeline of our image domain translator for data augmentation**. For clarity, the dual translation path from target to source and back to target domain is omitted.

B. Training of Image Domain Translator

In our work, we take a pretrained image domain translator to help create CrDoMix for data augmentation. As depicted in Fig.2, the training of our image domain translator follows the pipeline of CycleGAN [28]. The input x_s is passed to $\mathcal{T}_{s2t}$, constrained by an adversarial loss $\mathcal{L}_s^{adv}$ and a self-reconstruction loss $\mathcal{L}_s^{rec}$, producing x_{s2t} that is sent to a target-to-source translator for cyclic reconstruction of the input x_s controlled by $\mathcal{L}_s^{cyc}$. Likewise, the dual translation training step is performed for target input x_t, but for simplicity this part is omitted from Fig.2.

*corresponding author

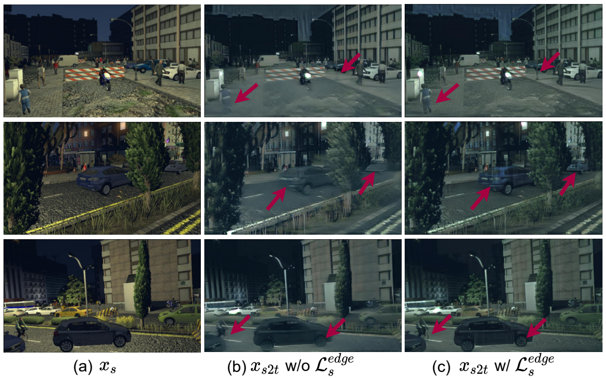

Figure 3. Visual comparison of image domain translator trained w/ and w/o semantic edge reconstruction loss. Red arrows point out the major differences.

However, according to [3], in certain occasions, CycleGAN tends to 'hide' information about source images into the translated images in a hardly perceptible way, which is less meaningful to be considered for data augmentation. Hence, considering that the label maps are available for the virtual source domain, we also make use of which to compute a semantic edge reconstruction loss when input comes from the source domain. Specifically, we utilize the source domain label map y_s to get a semantic edge mask y_s^{edge}, which is pixelwisely multiplied with x_s to filter out the non-edge pixels. An element-wise L1-metric semantic edge reconstruction loss $\mathcal{L}_s^{edge}$ is then applied to x_{s2t}. The motivation is simply to take advantage of the available y_s and preserve the semantic edges during translation to avoid the 'steganographical effects' in the resulting outputs. In Fig.3, we visualize and compare the outputs of the domain translators trained w/ and w/o $\mathcal{L}_s^{edge}$. Following the red arrows, we can observe in Fig.3(b) that some objects in certain regions of x_{s2t} get faded without having $\mathcal{L}_s^{edge}$ as a constraint. Reconstructing the semantic edges, however, brings these regions salient again in Fig.3(c). As a quantitative verification, we also utilize $\mathcal{T}_{s2t}$ as image domain translator to train our DiGA warm-up model, witnessing a performance drop by 0.2 mIoU for GTA5-to-Cityscapes adaptation and 0.4 mIoU for Synthia-to-Cityscapes adaptation, respectively. With the pretrained and fixed image domain translator $\mathcal{T}_{s2t}$, we can perform our CrDoMix data augmentation for UDA. Additional examples are provided in Fig.4. We can see that the mixture of $\tilde{x}_s$ and x_{s2t} based on CrDoMix can simultaneously integrate diverse cross-domain effects on every single x_{cdm} without breaking the geometric structure that comes from the original input x_s. Moreover, the inter-domain changes are not fixed to a specific region but randomly appear across the whole image.

Moreover, as Fig. 5 shows, in certain occasions some regions of $\tilde{x}_s$ might appear darker because of the effect from random combination of basic augmentation operations, which means the pixel values are similar and close to zero. In this case, it makes less sense to train the network on these regions. However, our CrDoMix makes the learning of these unobtrusive regions more interesting by making the selected parts more salient while having target-like appearance (See x_{s2t} in Fig. 5).

Phase	warm-up stage			ST stage		
λ_s^{distil}	0.25	0.5	1.0	0.0	0.25	0.5
mIoU	50.8	51.1	50.4	62.1	62.7	62.4

Table 2. The effect of λ_s^{distil} for training DiGA. (GTA5-to-Cityscapes)

α	0.0	0.25	0.5	0.75	1.0
mIoU	49.3	50.9	**51.1**	50.8	50.8

Table 3. **Warm-up performance ablation** using different α values in $\mathcal{L}_s^{distil}$. (GTA5-to-Cityscapes)

C. Feature Discriminability Analysis

In Fig.6, we compare t-SNE [18] visualizations of the source-only model, our warm-up model and ST stage model regarding their class-wise feature distribution based on Cityscapes validation set. We can observe from left to right that the feature distribution changes conform with the improvements of mIoU in Table **??**. As the model gets better adaptable, the feature clusters also demonstrate a clearer momentum of separation class-wisely.

D. Additional Ablative Analysis

In Table 2, we study the impact of re-weighting $\mathcal{L}_s^{distil}$ in DiGA traning pipeline. For warm-up stage training, we set $\lambda_s^{distil} = 0.5$ to emphasize the knowledge distillation loss, achieving 51.1 mIoU on target validation set. On the other hand, reducing λ_s^{distil} to 0.25 makes the effect of $\mathcal{L}_s^{distil}$ a bit insufficient (drops to 50.8 mIoU) but increasing it to 1.0 will overweigh the source domain supervised loss $\mathcal{L}_s^{seg}$, leading to a drop to 50.4 mIoU. In terms of ST stage, however, as pseudo-labels for target domain are produced on-the-fly, the learning should be prioritized on pseudo-supervision instead of knowledge distillation. Therefore, we reduce λ_s^{distil} from 0.5 to 0.25, achieving the best performance for ST stage (62.7 mIoU). As also observed in Table 2, setting λ_s^{distil} to 0.0 means training the ST stage without knowledge distillation, which experiences a performance drop from 62.7 to 62.1 mIoU.

In Table 3, the impact of the scaling factor in $\mathcal{L}_s^{distil}$ is studied. We observe that, even though the symmetric path in our knowledge distillation is beneficial, without which the warm-up mIoU becomes 49.3, yet we need to point out that the choice of α value can influence the warm-up training to certain degree. When α is too large (*e.g.*, set to 1 or

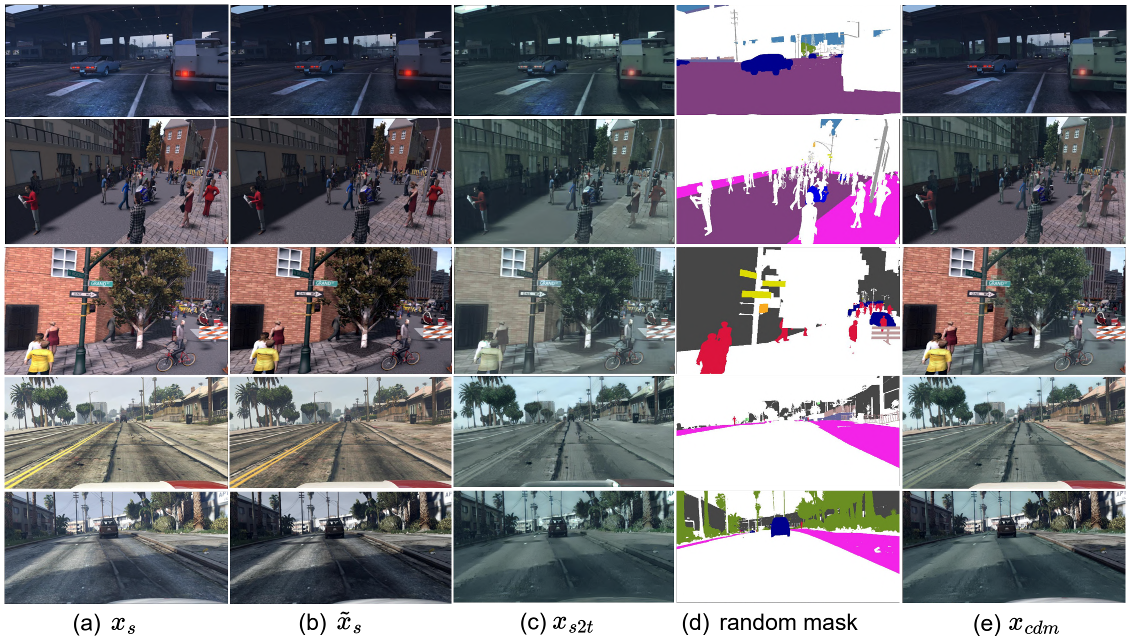

Figure 4. Additional visual examples for creating CrDoMix data augmentation on virtual source domain.

Figure 5. Each row from left to right: examples of x_s, $\tilde{x}_s$ and x_{cdm}. White dashed boxes indicate the major differences from a visual perspective.

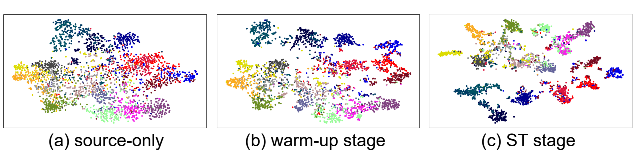

Figure 6. **Visualization of feature distribution** on Cityscapes validation set for each stage based on t-SNE [18] map.

0.75), the symmetric distillation path can balance the supervision signal from y^s in a more dominant manner, leading to a slight performance drop from 51.1 to 50.8. Hence, an optimal empirical choice of α is 0.5. (Note that for Synthia-to-Cityscapes benchmark, however, this value is set to 0.25 acccording to our experience)

E. Discussion of DiGA Variants

We train DiGA variants for warm-up stage as well as ST stage by replacing specific model components with other methods to better understand the effectiveness of them.

What if CrDoMix is performed in Cutmix [24] fashion instead of class-wise exchange?
Variant (1) in Table 4 shows that the warm-up stage mIoU drops from 51.1 to 50.7 after replacing our strategy in CrDoMix with Cutmix [24]. An explanation for this result is that performing CrDoMix in Cutmix fashion makes it possible to insert target domain style onto x_{cdm}, but is limited within a pre-defined square box. However, by class-wise combination according to the source label map y_s, the inter-

Phase	warm-up stage		ST stage		
Variant	(1)CrDoMix→(Cutmix)	(2)$\mathcal{L}_s^{distil}$→($\mathcal{L}_{seg}^{cdm}$)	(3)Λ→(x_s)	(4)Λ→(x_{cdm})	(5)Λ→(x_t)
mIoU	50.7	45.8	60.4	61.3	62.5

Table 4. Model performances of training DiGA based on other variants.

Method	Venue	road	sdwk	bldng	wall	fence	pole	light	sign	veg	trrn	sky	psn	rider	car	truck	bus	train	moto	bike	mIoU
Source-only	-	83.2	35.7	81.3	29.3	20.8	27.4	26.9	19.4	81.9	32.2	76.6	51.8	15.0	71.9	22.5	28.6	4.0	18.5	0.0	38.3
BDL [13]	CVPR'19	91.0	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
CAG [26]	NeurIPS'19	90.4	51.6	83.8	34.2	27.8	38.4	25.3	48.4	85.4	38.2	78.1	58.6	34.6	84.7	21.9	42.7	41.1	29.3	37.2	50.2
CrCDA [8]	ECCV'20	92.4	55.3	82.3	31.2	29.1	32.5	33.2	35.6	83.5	34.8	84.2	58.9	32.2	84.7	40.6	46.1	2.1	31.1	32.7	48.6
FADA [19]	ECCV'20	91.0	50.6	86.0	43.4	29.8	36.8	43.4	25.0	86.8	38.3	87.4	64.0	38.0	85.2	31.6	46.1	6.5	25.4	37.1	50.1
Seg-Uncer [27]	IJCV'21	90.4	31.2	85.1	36.9	25.6	37.5	48.8	48.5	85.3	34.8	81.1	64.4	36.8	86.3	34.9	52.2	1.7	29.0	44.6	50.3
DACS [16]	WACV'21	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
IAST [15]	ECCV'20	94.1	58.8	85.4	39.7	29.2	25.1	43.1	34.2	84.8	34.6	88.7	62.7	30.3	87.6	42.3	50.3	24.7	35.2	40.2	52.2
UncerDA [22]	ICCV'21	90.5	38.7	86.5	41.1	32.9	40.5	48.2	42.1	86.5	36.8	84.2	64.5	38.1	87.2	34.8	50.4	0.2	41.8	54.6	52.6
CDGA [10]	AAAI'21	91.1	52.8	84.6	32.0	27.1	33.8	38.4	40.3	84.6	42.8	85.0	64.2	36.5	87.3	44.4	51.0	0.0	37.3	44.9	51.5
MetaCorrection [4]	CVPR'21	92.8	58.1	86.2	39.7	33.1	36.3	42.0	38.6	85.5	37.8	87.6	62.8	31.7	84.8	35.7	50.3	2.0	36.8	48.0	52.1
SAC [1]	CVPR'21	90.4	53.9	86.6	42.4	27.3	45.1	48.5	42.7	87.4	40.1	86.1	67.5	29.7	88.5	49.1	54.6	9.8	26.6	45.3	53.8
ProDA [25]	CVPR'21	91.5	52.4	82.9	42.0	35.7	40.0	44.4	43.3	87.0	43.8	79.5	66.5	31.4	86.7	41.1	52.5	0.0	45.4	53.8	53.7
Undoing [14]	CVPR'22	89.1	34.3	83.6	38.3	27.5	28.9	34.7	17.6	84.2	41.0	85.1	57.8	33.7	85.1	38.5	41.3	30.7	31.1	48.0	49.0
CPSL [12]	CVPR'22	91.7	52.9	83.6	43.0	32.3	43.7	51.3	42.8	85.4	37.6	81.1	69.5	30.0	88.1	44.1	59.9	24.9	47.2	48.4	55.7
ProCA [9]	ECCV'22	91.9	48.4	87.3	41.5	31.8	41.9	47.9	36.7	86.5	42.3	84.7	68.4	43.1	88.1	39.6	48.8	**40.6**	43.6	56.9	56.3
CorDA [21]	ICCV'21	94.7	63.1	87.6	30.7	**40.6**	40.2	47.8	51.6	87.6	**47.0**	89.7	66.7	35.9	90.2	48.9	57.5	0.0	39.8	56.0	56.6
BCL [11]	ECCV'22	93.5	60.2	88.1	31.1	37.0	41.9	54.7	37.8	**89.9**	45.5	**89.9**	72.7	38.2	90.7	34.3	53.2	4.4	47.2	58.5	57.1
DiGA (Ours)	CVPR'23	**95.6**	**67.4**	**89.8**	**51.6**	38.1	**52.0**	**59.0**	51.5	86.4	34.5	87.7	**75.6**	**48.8**	**92.5**	**66.5**	**63.8**	19.7	**49.6**	**61.6**	**62.7**
ProDA‡ [25]	CVPR'21	87.8	56.0	79.7	46.3	**44.8**	45.6	53.5	53.5	88.6	45.2	82.1	70.7	39.2	88.8	45.5	59.4	1.0	48.9	56.4	57.5
Undoing+ProDA‡ [14]	CVPR'22	92.9	52.7	87.2	39.4	41.3	43.9	55.0	52.9	**89.3**	**48.2**	**91.2**	71.4	36.0	90.2	67.9	59.8	0.0	48.5	59.3	59.3
CPSL‡ [12]	CVPR'22	92.3	59.9	84.9	45.7	29.7	**52.8**	**61.5**	**59.5**	87.9	41.5	85.0	73.0	35.5	90.4	48.7	**73.9**	**26.3**	53.8	53.9	60.8
DiGA‡ (Ours, ResNet)	CVPR'23	**95.6**	**66.5**	**89.6**	**55.7**	36.5	50.8	59.0	56.9	86.4	35.8	84.4	**76.0**	**48.8**	**93.2**	**70.0**	66.0	18.2	**55.8**	**65.6**	**63.7**
DiGA (Ours, HRNet)	CVPR'23	95.2	65.2	90.7	59.0	57.1	57.8	63.3	54.8	90.0	42.4	89.0	76.8	49.6	91.6	66.8	69.8	59.7	24.0	51.9	66.1
DAFormer [6]	CVPR'22	95.7	70.2	89.4	53.5	**48.1**	49.6	55.8	59.4	89.9	47.9	**92.5**	72.2	44.7	92.3	74.5	78.2	65.1	55.9	61.8	68.3
DiGA (Ours + DAFormer)	CVPR'23	95.7	**70.4**	**89.8**	**54.8**	47.8	**51.3**	**57.8**	**63.9**	**90.3**	**48.8**	91.8	**73.1**	**46.6**	**92.6**	**78.5**	**81.3**	**74.8**	**57.3**	**63.2**	**70.0**
HRDA [7]	ECCV'22	96.4	74.4	91.0	**61.6**	51.5	**57.1**	63.9	69.3	91.3	48.4	**94.2**	79.0	52.9	**93.9**	**84.1**	85.7	75.9	**63.9**	**67.5**	73.8
DiGA (Ours + HRDA)	CVPR'23	**97.0**	**78.6**	**91.3**	60.8	**56.7**	56.5	**64.4**	**69.9**	**91.5**	**50.8**	93.7	**79.2**	**55.2**	93.7	78.3	**86.9**	**77.8**	63.7	65.8	**74.3**

Table 5. **GTA5-to-Cityscapes adaptation results**. We compare our model performance with state-of-the-art methods. For each configuration, bold stands for **best** and underline for second-best. ‡ means an extra distillation stage on target domain using `SimCLRv2` backbone.

domain changes are not fixed to a specific region but randomly appear across the whole image. This makes the data augmentation more diverse, thus creating a more meaningful augmented image.

What if CrDoMix data augmentation is directly adopted for source supervision instead of knowledge distillation? This experiment is conducted as variant (2) in Table 4, which means that we utilize x_{cdm} to compute a segmentation loss $\mathcal{L}_{seg}^{cdm}$ in the same way as we treat x_s. Therefore, in this case, knowledge distillation losses are replaced by a supervised cross-entropy segmentation loss, and we see that the warm-up model performance degrades from 51.1 to 45.8 mIoU. The reason for this decrease is that $\mathcal{T}_{s2t}$ does not ensure a perfect mapping from source to target domain, otherwise the domain gap can be purely closed by image translation. Therefore, applying direct supervision using hard source labels on x_{cdm} might cause the network to overfit on the target-like textures that are faked by $\mathcal{T}_{s2t}$. However, by distilling soft assignments $p_s^\dagger$ to p_{cdm}, we only expect the network to learn a momentum to predict on x_{cdm} in a similar way close to x_s. Thus, the soft distillation signal is not as strong as a hard supervision signal, avoiding the overfitting effect to a certain degree. On the other hand, as discussed in the main paper, our distillation loss prevents the student from learning unwanted source labels, reducing the noise introduced by them. Hence, our CrDoMix-based knowledge distillation demonstrates superiority over direct supervision on x_{cdm}.

What if x_s (or $\mathcal{X}_s$) instead of x_{cdm} (or $\mathcal{X}_{cdm}$) and x_t (or $\mathcal{X}_s$) is adopted for computing and updating class centroids Λ? What about x_{cdm} or x_t alone? This experiment is conducted as variant (4), (5) and (3) in Table 4, showing that using x_s instead of $\tilde{x}_{cdm}$ to compute Λ results in a mIoU decrease from 62.7 to 60.4 in ST stage. Similarly, the ST stage result decreases to 61.3 (using x_{cdm}) and 62.3 (using x_t) mIoU respectively. This result is self-explanatory since x_{cdm} contains target-like characteristics at pixel-level, which builds up a smoother transition

Method	Venue	road	sdwk	bldng	wall*	fence*	pole*	light	sign	veg	sky	psn	rider	car	bus	mcycl	bcycl	mIoU	mIoU*
Source-only	-	65.1	25.6	77.1	10.4	0.0	28.5	0.0	10.1	76.0	71.7	52.2	18.6	69.4	20.8	15.2	28.2	35.6	40.8
BDL [13]	CVPR'19	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
CAG [26]	NeurIPS'19	84.7	40.8	81.7	7.8	0.0	35.1	13.3	22.7	84.5	77.6	64.2	27.8	80.9	19.7	22.7	48.3	44.5	51.5
CrCDA [8]	ECCV'20	86.2	44.9	79.5	8.3	0.7	27.8	9.4	11.8	78.6	86.5	57.2	26.1	76.8	39.9	21.5	32.1	42.9	50.0
FADA [19]	ECCV'20	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
Seg-Uncer [27]	IJCV'21	84.3	37.7	79.5	5.3	0.4	24.9	9.2	8.4	80.0	84.1	57.2	23.0	78.0	38.1	20.3	36.5	41.7	48.9
DACS [16]	WACV'21	80.6	25.1	81.9	21.5	2.9	37.2	22.7	24.0	83.7	<u>90.8</u>	67.6	38.3	82.9	38.9	28.5	47.6	48.3	54.8
IAST [15]	ECCV'20	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
UncerDA [22]	ICCV'21	87.6	41.9	83.1	14.7	1.7	36.2	31.3	19.9	81.6	80.6	63.0	21.8	86.2	40.7	23.6	53.1	47.9	54.9
CDGA [10]	AAAI'21	90.7	49.5	84.5	-	-	-	33.6	<u>38.9</u>	84.6	84.6	59.8	33.3	80.8	51.5	37.6	45.9	-	54.1
MetaCorrection [4]	CVPR'21	<u>92.6</u>	52.7	81.3	8.9	2.4	28.1	13.0	7.3	83.5	85.0	60.1	19.7	84.8	37.2	21.5	43.9	45.1	52.5
SAC [1]	CVPR'21	89.3	47.2	<u>85.5</u>	26.5	1.3	43.0	45.5	32.0	87.1	89.3	63.6	25.4	86.9	35.6	30.4	53.0	52.6	59.3
ProDA [25]	CVPR'21	87.1	44.0	83.2	26.9	0.0	42.0	45.8	34.2	86.7	81.3	68.4	22.1	87.7	50.0	31.4	38.6	51.9	58.5
Undoing [14]	CVPR'22	83.6	36.2	80.9	10.3	0.1	27.4	17.6	22.8	81.5	81.2	54.6	20.1	80.3	38.1	11.1	42.9	43.0	50.1
CPSL [12]	CVPR'22	87.3	44.4	83.8	25.0	0.4	42.9	47.5	32.4	86.5	83.3	69.6	29.1	89.4	52.1	<u>42.6</u>	54.1	54.4	61.7
ProCA [9]	ECCV'22	90.5	52.1	84.6	**29.2**	3.3	40.3	37.4	27.3	86.4	85.9	<u>69.8</u>	28.7	88.7	<u>53.7</u>	14.8	<u>54.8</u>	53.0	59.6
CorDA [21]	ICCV'21	**93.3**	**61.6**	85.3	19.6	<u>5.1</u>	37.8	36.6	**42.8**	**94.9**	90.4	69.7	<u>41.8</u>	85.6	38.4	32.6	53.9	55.0	62.8
BCL [11]	ECCV'22	83.8	42.2	85.3	16.4	**5.7**	<u>43.1</u>	<u>48.3</u>	30.2	<u>89.3</u>	**92.1**	68.2	**43.1**	<u>89.7</u>	47.2	42.2	54.2	<u>55.6</u>	<u>62.9</u>
DiGA (Ours, ResNet)	CVPR'23	89.1	<u>53.4</u>	**86.1**	<u>28.7</u>	3.0	**49.6**	**50.6**	34.9	88.2	84.9	**71.3**	40.9	**91.6**	**75.1**	**50.3**	**65.8**	**60.2**	**67.9**
ProDA‡ [25]	CVPR'21	<u>87.8</u>	<u>45.7</u>	84.6	**37.1**	<u>0.6</u>	44.0	54.6	37.0	<u>88.1</u>	84.4	<u>74.2</u>	24.3	88.2	51.1	40.5	45.6	55.5	62.0
Undoing+ProDA‡ [14]	CVPR'22	82.5	37.2	81.1	23.8	0.0	45.7	<u>57.2</u>	**47.6**	87.7	<u>85.8</u>	74.1	28.6	88.4	<u>66.0</u>	47.0	55.3	56.7	64.5
CPSL‡ [12]	CVPR'22	87.2	43.9	<u>85.5</u>	<u>33.6</u>	0.3	<u>47.7</u>	**57.4**	37.2	87.8	**88.5**	**79.0**	<u>32.0</u>	<u>90.6</u>	49.4	<u>50.8</u>	<u>59.8</u>	<u>57.9</u>	<u>65.3</u>
DiGA ‡(Ours, ResNet)	CVPR'23	**89.2**	**53.4**	**86.1**	33.0	**2.5**	**49.8**	54.0	<u>37.5</u>	<u>88.3</u>	**88.5**	72.5	**43.3**	**92.0**	**77.7**	**51.5**	**67.2**	**61.8**	**69.4**
DiGA (Ours, HRNet)	CVPR'23	90.6	56.3	87.4	38.8	6.4	57.7	59.3	50.4	87.9	86.4	76.1	47.9	89.0	54.2	47.2	69.1	62.8	69.4
DAFormer [6]	CVPR'22	84.5	40.7	**88.4**	41.5	6.5	50.0	55.0	**54.6**	86.0	89.8	73.2	48.2	87.2	53.2	53.9	61.7	60.9	67.4
DiGA (Ours + DAFormer)	CVPR'23	**85.2**	**41.4**	88.2	**42.6**	**7.5**	**52.1**	**57.5**	47.7	**87.8**	**90.8**	**75.0**	**50.8**	**87.8**	**58.0**	**58.5**	**63.0**	**62.1**	**68.6**
HRDA [7]	ECCV'22	85.2	47.7	88.8	49.5	4.8	**57.2**	**65.7**	60.9	85.3	92.9	**79.4**	**52.8**	89.0	**64.7**	**63.9**	64.9	65.8	72.4
DiGA (Ours + HRDA)	CVPR'23	**88.5**	**49.9**	**90.1**	**51.4**	**6.6**	55.3	64.8	**62.7**	**88.2**	**93.5**	78.6	51.8	**89.5**	62.2	61.0	**65.8**	**66.2**	**72.8**

Table 6. **Synthia-to-Cityscapes adaptation results**. mIoU, mIoU* refer to 16-class and 13-class experiment settings, respectively. For each configuration, bold stands for **best** and underline for <u>second-best</u>. ‡ means an extra distillation stage using `SimCLRv2` backbone.

from source to target domain. Updating together with x_t contributes to the computation of more domain-robust class centroids, which is beneficial when it comes to the step of feature-centroid based voting for target pseudo-supervision. Therefore, we prefer to adopt x_{cdm} together with x_t rather than x_s along to obtain the feature class centroids in ST stage.

What if $\hat{y}_t^{warm}$ is updated at each iteration during the self-training stage?
In our paper, we update $\hat{y}_t^{warm}$ only once at epoch 50 (half of the training). But there are more options for updating $\hat{y}_t^{warm}$. Here we report the performances (mIoU) of three updating strategies: (1) **60.5** (*no update*); (2) **56.4** (*update each iteration*); (3) **62.7** (*ours*). This shows that updating $\hat{y}_t^{warm}$ is useful but tricky. When $\hat{y}_t^{warm}$ is updated at each iteration, we observed that the validation accuracy would first go up, but then fall forever due to the instability when $\hat{y}_t^{warm}$ and $\hat{y}_t^{feat}$ both change. Although our self-training strategy shows nice potential in label selection, but there is still no guarantee that every pseudo-label is correct. If there is a wrong pseudo-label selected, this wrong classification will be mistakenly 'encouraged' in following iterations and get further amplified during training. Therefore, we choose to keep $\hat{y}_t^{warm}$ relatively static and $\hat{y}_t^{feat}$ dynamic. We believe further strategy for updating $\hat{y}_t^{warm}$ can be an interesting future direction.

When distilling to p_s, why do we use $\tilde{p}_s^{\dagger}$ instead of $p_s^{\dagger}$?
As mentioned in our paper, our symmetric knowledge distillation enables bidirectional alignment and between the input source image and its augmented view. In the additional symmetric distillation path, p_s is supervised by y^s while being balanced by the more smoother $\tilde{p}_s^{\dagger}$. In this way, p_s is pulled to a distribution that is close to $\tilde{p}_s^{\dagger}$ (augmented output) while still being encouraged to make correct semantic predictions because of the supervision signal from y^s. In order to enforce the student to learn to produce domain-invariant outputs, it is actually helpful to inject out-of-distribution or target-aware perturbations to p_s. Compared with $p_s^{\dagger}$, $\tilde{p}_s^{\dagger}$ carries out-of-distribution and target-aware property, preventing the soft label to be source-specific and thus improves domain generalization. We experimented to use $p_s^{\dagger}$ but found that the warm-up mIoU drops by **0.4**.

F. Quantitative Comparison with SOTA

We provide more detailed quantitative comparison of DiGA with state-of-the-art methods for domain adaptive semantic segmentation. From Table 5 and Table 6, we ob-

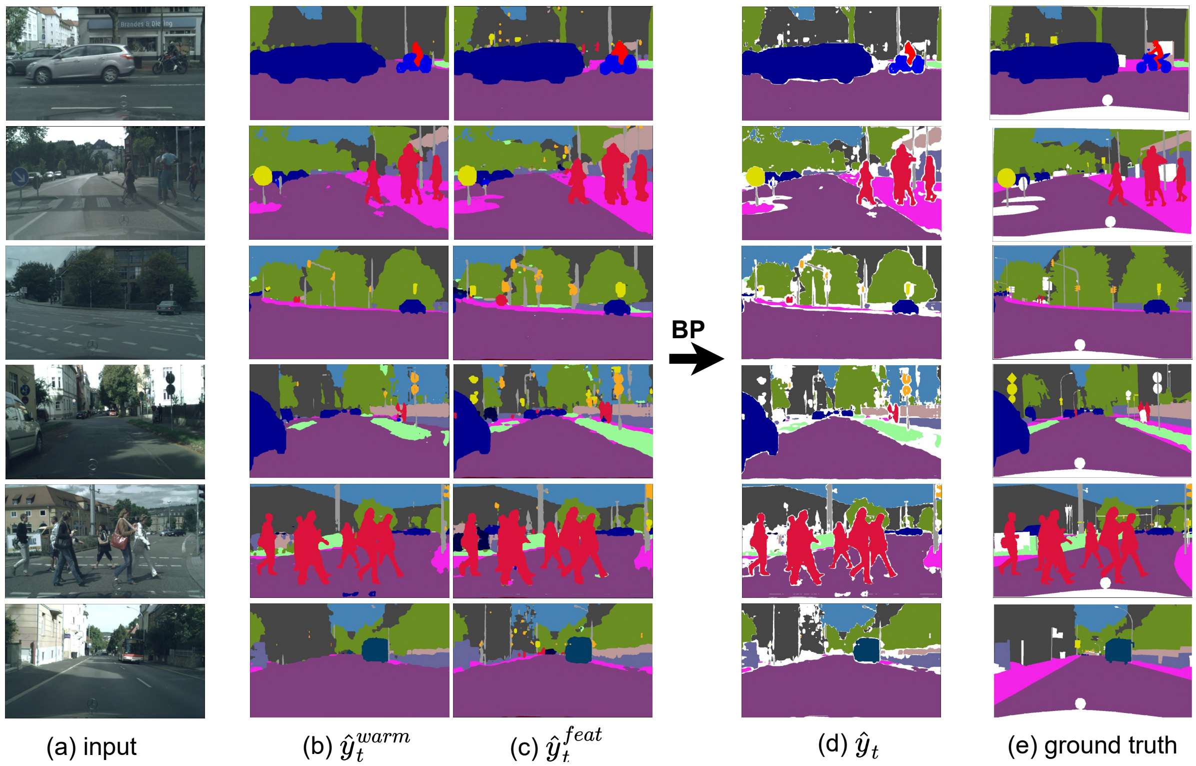

Figure 7. Additional examples showing our bilateral-consensus pseudo-supervision procedure.

serve that our approach outperforms other methods by considerable margins in terms of mIoU when trained on the same network architecture, *i.e.*, `ResNet-101` [5] backbone plus `Deeplab-V2` [2] for segmentation. Likewise, we also reproduced an extra knowledge distillation stage using `SimCLRv2` backbone and observe that our previous results get improved from 62.7 to 63.7 mIoU for GTA5-to-Cityscapes adaptation and from 60.2 to 61.8 mIoU for Synthia-to-Cityscapes adaptation. To test the architectural generalizability of our method, we also train on `OCRNet` [23] with `HRNet-W48` [20] backbone and obtain higher mIoU scores on both benchmarks. For some specific classes, the `HRNet` results do not outperform the `ResNet` ones. This is owing to the randomness in data sampling, for example, if some long-tail classes are rarely seen by the network during training, the worst case is that the network is much less accurate when predicting those classes. We retrain our framework twice and can observe that the class-wise IoU varies a lot even though the mIoU is close. This also explains why model assembling works to further improve the performance for deep neural networks.

G. Additional Qualitative Results of DiGA

In this section we provide additional examples show the pseudo-labelling procedure and results of our proposed BP strategy (Fig. 7), and more model inference examples of DiGA on the Cityscapes validation set (Fig. 8).

References

[1] Nikita Araslanov and Stefan Roth. Self-supervised augmentation consistency for adapting semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15384–15394, 2021. iv, v

[2] 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. vi

[3] Casey Chu, Andrey Zhmoginov, and Mark Sandler. Cyclegan, a master of steganography. *arXiv preprint arXiv:1712.02950*, 2017. ii

[4] Xiaoqing Guo, Chen Yang, Baopu Li, and Yixuan Yuan. Metacorrection: Domain-aware meta loss correction for unsupervised domain adaptation in semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3927–3936, 2021. iv, v

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. vi

[6] Lukas Hoyer, Dengxin Dai, and Luc Van Gool. Daformer: Improving network architectures and training strategies for

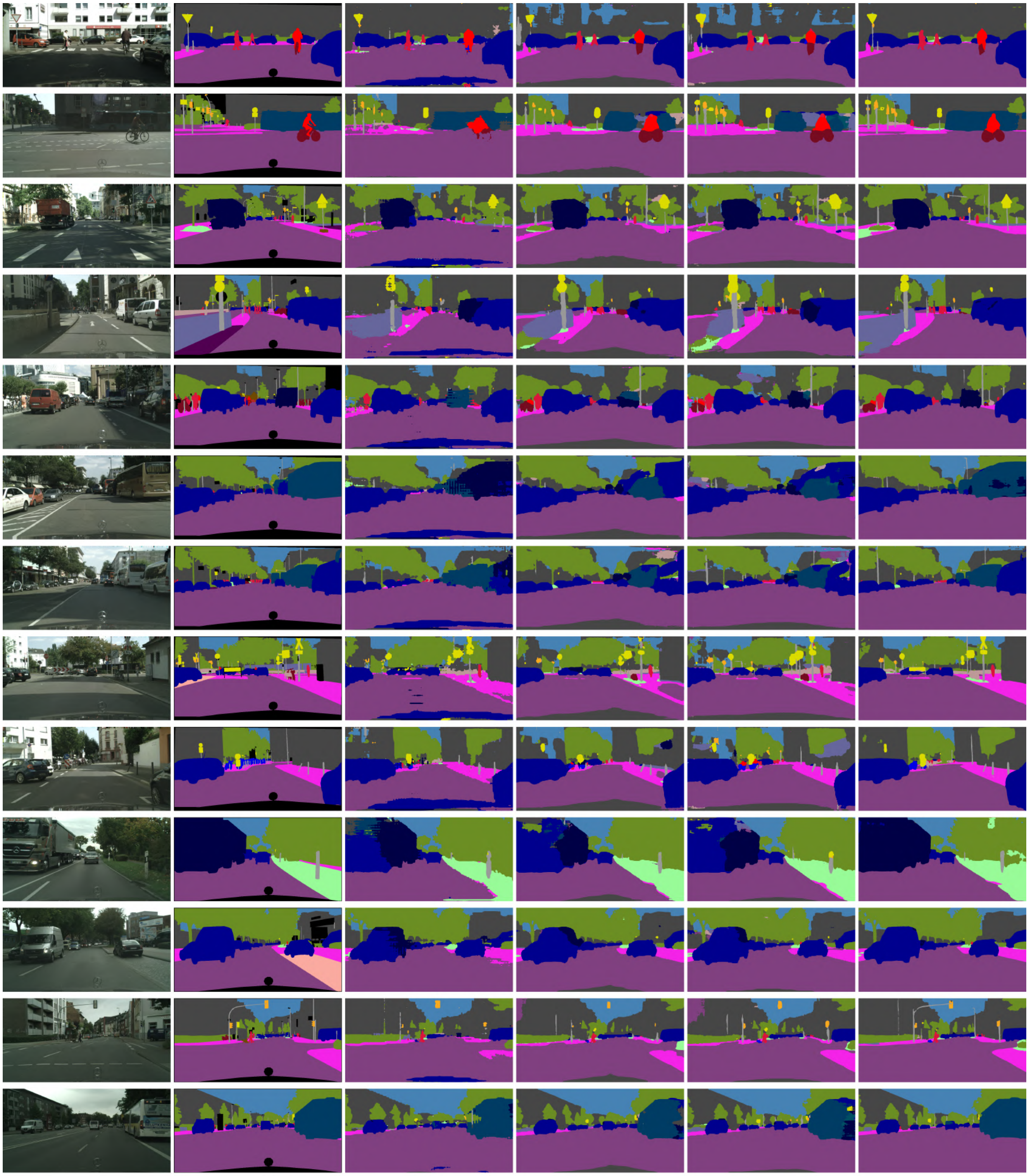

Figure 8. **Qualitative results of GTA5-to-Cityscapes adaptation on Cityscapes validation set**. Columns from left to right are: target domain inputs; ground-truth labels; segmentation predictions of BDL [13], ProDA [25], CPSL [12] and DiGA.

domain-adaptive semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern*

Recognition, pages 9924–9935, 2022. iv, v

[7] Lukas Hoyer, Dengxin Dai, and Luc Van Gool. Hrda: Context-aware high-resolution domain-adaptive semantic segmentation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXX*, pages 372–391. Springer, 2022. iv, v

[8] Jiaxing Huang, Shijian Lu, Dayan Guan, and Xiaobing Zhang. Contextual-relation consistent domain adaptation for semantic segmentation. In *European conference on computer vision*, pages 705–722. Springer, 2020. iv, v

[9] Zhengkai Jiang, Yuxi Li, Ceyuan Yang, Peng Gao, Yabiao Wang, Ying Tai, and Chengjie Wang. Prototypical contrast adaptation for domain adaptive semantic segmentation. *arXiv preprint arXiv:2207.06654*, 2022. iv, v

[10] Minsu Kim, Sunghun Joung, Seungryong Kim, JungIn Park, Ig-Jae Kim, and Kwanghoon Sohn. Cross-domain grouping and alignment for domain adaptive semantic segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1799–1807, 2021. iv, v

[11] Geon Lee, Chanho Eom, Wonkyung Lee, Hyekang Park, and Bumsub Ham. Bi-directional contrastive learning for domain adaptive semantic segmentation. In *European Conference on Computer Vision*, pages 38–55. Springer, 2022. iv, v

[12] Ruihuang Li, Shuai Li, Chenhang He, Yabin Zhang, Xu Jia, and Lei Zhang. Class-balanced pixel-level self-labeling for domain adaptive semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11593–11603, 2022. iv, v, vii

[13] Yunsheng Li, Lu Yuan, and Nuno Vasconcelos. Bidirectional learning for domain adaptation of semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6936–6945, 2019. iv, v, vii

[14] Yahao Liu, Jinhong Deng, Jiale Tao, Tong Chu, Lixin Duan, and Wen Li. Undoing the damage of label shift for cross-domain semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7042–7052, 2022. iv, v

[15] Ke Mei, Chuang Zhu, Jiaqi Zou, and Shanghang Zhang. Instance adaptive self-training for unsupervised domain adaptation. *arXiv preprint arXiv:2008.12197*, 2020. iv, v

[16] 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*, pages 1379–1389, 2021. iv, v

[17] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7472–7481, 2018. i

[18] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. ii, iii

[19] Haoran Wang, Tong Shen, Wei Zhang, Ling-Yu Duan, and Tao Mei. Classes matter: A fine-grained adversarial approach to cross-domain semantic segmentation. In *European conference on computer vision*, pages 642–659. Springer, 2020. iv, v

[20] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3349–3364, 2020. vi

[21] Qin Wang, Dengxin Dai, Lukas Hoyer, Luc Van Gool, and Olga Fink. Domain adaptive semantic segmentation with self-supervised depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8515–8525, 2021. iv, v

[22] Yuxi Wang, Junran Peng, and ZhaoXiang Zhang. Uncertainty-aware pseudo label refinery for domain adaptive semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9092–9101, 2021. iv, v

[23] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. In *European conference on computer vision*, pages 173–190. Springer, 2020. vi

[24] 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. iii

[25] Pan Zhang, Bo Zhang, Ting Zhang, Dong Chen, Yong Wang, and Fang Wen. Prototypical pseudo label denoising and target structure learning for domain adaptive semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12414–12424, 2021. iv, v, vii

[26] Qiming Zhang, Jing Zhang, Wei Liu, and Dacheng Tao. Category anchor-guided unsupervised domain adaptation for semantic segmentation. *arXiv preprint arXiv:1910.13049*, 2019. iv, v

[27] Zhedong Zheng and Yi Yang. Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation. *International Journal of Computer Vision*, 129(4):1106–1120, 2021. iv, v

[28] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. i