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.

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.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>mIoU</td>
<td>45.2</td>
<td>46.9</td>
<td>47.3</td>
<td>51.1</td>
<td>51.3</td>
</tr>
</tbody>
</table>

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.

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 $T_{s,t}$, constrained by an adversarial loss $L_{adv}$ and a self-reconstruction loss $L_{rec}$, producing $x_{st}$ that is sent to a target-to-source translator for cyclic reconstruction of the input $x_s$ controlled by $L_{cyc}$. Likewise, the dual translation training step is performed for target input $x_t$, but for simplicity this part is omitted from Fig. 2.
However, according to [3], in certain occasions, Cycle-GAN 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 $L_s^{edge}$ is then applied to $x_s$. 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 semantic edge reconstruction loss. Red arrows point out the major differences.

In Table 2, we study the impact of re-weighting $L_{distil}$ in DiGA training pipeline. For warm-up stage training, we set $\lambda_{distil} = 0.5$ to emphasize the knowledge distillation loss, achieving 51.1 mIoU on target validation set. On the other hand, reducing $\lambda_{distil}$ to 0.25 makes the effect of $L_{distil}$ a bit insufficient (drops to 50.8 mIoU) but increasing it to 1.0 will overweigh the source domain supervised loss $L_s^{seg}$, 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 $\lambda_{distil}$ from 0.5 to 0.25, achieving the best performance for ST stage (62.7 mIoU). As also observed in Table 2, setting $\lambda_{distil}$ 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 $L_{distil}$ is too large (e.g., set to 1 or

<table>
<thead>
<tr>
<th>Phase</th>
<th>warm-up stage</th>
<th>ST stage</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda_{distil}$</td>
<td>0.25</td>
<td>0.5</td>
</tr>
<tr>
<td>mIoU</td>
<td>50.8</td>
<td>51.1</td>
</tr>
</tbody>
</table>

Table 2. The effect of $\lambda_{distil}$ for training DiGA. (GTA5-to-Cityscapes)

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>0.0</th>
<th>0.25</th>
<th>0.5</th>
<th>0.75</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>mIoU</td>
<td>49.3</td>
<td>50.9</td>
<td>51.1</td>
<td>50.8</td>
<td>50.8</td>
</tr>
</tbody>
</table>

Table 3. Warm-up performance ablation using different $\alpha$ values in $L_{distil}$. (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 2. 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 $L_{distil}$ in DiGA training pipeline. For warm-up stage training, we set $\lambda_{distil} = 0.5$ to emphasize the knowledge distillation loss, achieving 51.1 mIoU on target validation set. On the other hand, reducing $\lambda_{distil}$ to 0.25 makes the effect of $L_{distil}$ a bit insufficient (drops to 50.8 mIoU) but increasing it to 1.0 will overweigh the source domain supervised loss $L_s^{seg}$, 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 $\lambda_{distil}$ from 0.5 to 0.25, achieving the best performance for ST stage (62.7 mIoU). As also observed in Table 2, setting $\lambda_{distil}$ 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 $L_{distil}$ 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 $\alpha$ value can influence the warm-up training to certain degree. When $\alpha$ 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.

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-
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 $L_{cdm}$ 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 $T_{cdm}$ 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 $T_{cdm}$. However, by distilling soft assignments $p_s$ 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 $x'_s$) instead of $x_{cdm}$ (or $x'_{cdm}$) and $x_t$ (or $x'_t$) is adopted for computing and updating class centroids $\Lambda$? 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 $x_{cdm}$ to compute $\Lambda$ 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.
DiGA (Ours, ResNet) 2021 89.1 53.4 86.1 26.7 50.0 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] 2021 87.8 55.7 84.6 37.1 0.6 44.0 34.6 37.0 88.1 84.4 72.4 24.3 88.2 51.1 40.5 45.8 55.5 62.0

 UndoingProDA† [14] 2021 82.5 37.2 81.1 28.8 0.0 45.7 27.2 47.6 87.7 35.8 74.1 28.6 88.4 66.0 47.0 59.3 56.7 64.5

DACS [16] 2021 89.7 54.0 85.5 33.6 0.3 47.2 57.4 37.2 87.8 88.5 79.0 32.0 90.6 49.4 50.8 59.8 57.9 65.5

CorDA [21] 2021 93.3 61.6 85.3 19.6 5.1 37.8 36.6 42.8 94.9 90.4 69.7 41.8 85.6 38.4 32.6 53.9 55.0 62.8

BCL [11] 2022 83.8 42.2 85.3 16.4 5.7 43.1 48.3 30.2 89.3 92.1 68.2 43.1 89.2 47.2 42.2 54.2 55.6 62.9

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 second-best. † 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 $x_{edm}$ together with $x_t$ rather than $x_s$ along to obtain the feature class centroids in ST stage.

What if $\hat{y}_t$ is updated at each iteration during the self-training stage?

In our paper, we update $\hat{y}_t$ only once at epoch 50 (half of the training). But there are more options for updating $\hat{y}_t$. 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 $\hat{y}_t$ is useful but tricky. When $\hat{y}_t$ is updated at each iteration, we observed that the validation accuracy would first go up, but then fall forever due to the instability when $\hat{y}_t$ and $\hat{y}_{\text{feat}}$ 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 $\hat{y}_t$ relatively static and $\hat{y}_{\text{feat}}$ dynamic. We believe a further strategy for updating $\hat{y}_t$ can be an interesting future direction.

When distilling to $p_s$, why do we use $\hat{p}_s$ instead of $p_s^*$? 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 $\hat{p}_s$. In this way, $p_s$ is pulled to a distribution that is close to $\hat{p}_s$ (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 $\hat{p}_s$, $p_s^*$ 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 $p_s^*$ 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-

---

**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 second-best. † means an extra distillation stage using SimCLRv2 backbone.
serve that our approach outperforms other methods by considerable margins in terms of mIoU when trained on the same network architecture, \textit{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


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.


