Supplementary Material – Empirical Generalization Study: Unsupervised Domain Adaptation vs. Domain Generalization Methods for Semantic Segmentation in the Wild

Fabrizio J. Piva   Daan de Geus   Gijs Dubbelman
Eindhoven University of Technology
{f.j.piva, d.c.d.geus, g.dubbelman}@tue.nl

We provide the following supplementary material in addition to the main manuscript:

- More information about the method selection process, to show which existing UDA and DG methods complied to our selection criteria as defined in main manuscript (Sec. 1).
- Specifications of the normalized implementation for the semantic segmentation architecture, that we use for all DG and UDA methods (Sec. 2).
- Intensity histogram analysis for all datasets used in our work, to support our dataset selection process by assessing if they are homogeneous or heterogeneous (Sec. 3).
- Per-class results of UDA and DG methods, to quantify the performance of these methods on unseen domains for each semantic class (Sec. 4).
- Discussion regarding the potential combination of UDA and DG methods (Sec. 5).

1. Method selection process

As reported in the main manuscript, state-of-the-art UDA and DG methods must meet the following requirements to participate in our evaluation framework:

1. The implementation of the method should be publicly available, providing training and evaluation scripts.
2. The method should not require training external networks not available in the published code.
3. The methods should obtain state-of-the-art results in the standard benchmarks for their research domains.
4. Reproducing the reported results should not lead to a performance drop of more than 5%.

Tab. 1 shows the results of the selection process of UDA and DG methods. All methods in this table obtained state-of-the-art results at the time of conducting this research.

Task	Method	Code publicly available	Does not require training external networks	Reported mIoU (avg.)	Obtained mIoU (avg.)	Δ mIoU (%)	Accepted for comparison?
UDA	ProDA [20]	✓	✓	57.5	56.2	-2.3%	✓
	CorDA [15]	✓	✗	-	-	-	✗
	Coarse-to-fine [11]	✗	✓	-	-	-	✗
	CAMix [21]	✗	✓	-	-	-	✗
	DSP [7]	✓	✓	55.0	52.0	-5.5%	✗
	SAC [2]	✓	✓	53.8	52.9	-1.7%	✓
DG	WildNet [9]	✓	✓	50.1	49.0	-2.2%	✓
	RobustNet [5]	✓	✓	51.5	51.1	-0.8%	✓
	DRPC [18]	✗	✓	-	-	-	✗

Table 1. **Result of our method selection process:** four out of nine state-of-the-art candidates could meet the proposed criteria.

Analyzing this table, we observe that several methods are discarded because their code was not released to the public, meaning that the only way to reproduce their results is to re-implement the methods. As this is an extremely time-consuming and error-prone process, we considered this out of scope for this work. Furthermore, one method, CorDA [15], requires training an external self-supervised depth estimation network. As this additional network was not provided with the official code, it would have to be re-implemented and/or retrained for each dataset, which is again outside the scope of this work. Finally, DSP [7] was not considered because the obtained results from the official code deviated more than 5% from the reported results, and thus could not be reproduced.

It is important to consider that our study is not meant to exhaustively benchmark all available UDA and DG methods for the task of generalization to unseen domains. The goal is to take several methods that are representative for the state-of-the-art in UDA and DG research and make a high-level comparison between UDA and DG for the generalization task. The selected UDA and DG methods are such state-of-the-art methods and they employ many techniques that are common to the vast majority of other state-of-the-art methods. Therefore, besides meeting our more practical selection criteria, they most importantly provide a good representation for the state-of-the-art in their respective fields of research.

task	method	original implementation			normalized implementation		
		encoder	decoder	Cityscapes *val* mIoU	encoder	decoder	Cityscapes *val* mIoU
UDA	ProDA [20]	ResNet-101 [8]	DeepLab v2 from [20]	74.4	ResNet-101 [8]	DeepLab v2 from [20]	74.4
	SAC [2]	ResNet-101 [8]	DeepLab v2 [3]	67.6	ResNet-101 [8]	DeepLab v2 from [20]	68.4
DG	WildNet [9]	ResNet-101 [8]	DeepLab v3+ [4]	76.9	ResNet-101 [8]	DeepLab v2 from [20]	70.6
	RobustNet [5]	ResNet-101 from [5]	DeepLab v3+ [4]	76.4	ResNet-101 [8]	DeepLab v2 from [20]	74.8

Table 2. **Original implementation of selected candidates and normalized implementation.** We use a normalized semantic segmentation network for all methods to allow for a fair comparison. We compare the normalized implementation to the original versions, and run them in the training setting of experiment 1A, where Cityscapes [6] is the labeled domain $\mathcal{D}_l$, and BDD-100K [17] and Mapillary Vistas [12] are the unlabeled domains $\mathcal{D}_{nl}$. When the implementations, we find that replacing the decoder of UDA methods from [3] to [20] leads to a performance improvement, while downgrading the decoder of DG methods from [4] to [20] produces a performance drop.

ResNet-101 from RobustNet [5]	DeepLab v2 from ProDA [20]
+`conv2, bn2, relu2, conv3, bn3, relu3` on `layer0`	+`group norm.` after ASPP module
ResNet bottleneck without dilations	+bottleneck module for affine transformation
	+`group norm., dropout`

Table 3. **Implementation details of the adapted encoder from RobustNet [5] and decoder from ProDA [20].** RobustNet adds custom layers to ResNet-101 [8] on `layer0` and does not use dilations in the bottleneck units, and ProDA adds group normalization [16] and a custom bottleneck module to DeepLab v2 [3].

2. Normalized implementation

As explained in the main manuscript, one of the main issues when trying to compare UDA and DG methods directly using the reported numbers and original implementations, is the fact that there exist significant differences in the network architectures that these models use. To illustrate this, we take the selected UDA and DG methods and perform an additional experiment where we first train them with their original implementation, and then with our proposed normalized implementation. The results of this comparison along with the description of the different architectures of the selected UDA and DG methods are provided in Tab. 2.

From Tab. 2, we find that our two selected UDA methods use different decoders. SAC [2] uses DeepLab v2 [3], as is standard practice for UDA methods [2, 7, 10, 13, 15, 21], together with ResNet-101 [8], whereas ProDA [20] uses a modified version of DeepLab v2 (see Tab. 3 for details). This modified version causes a slight performance boost on Cityscapes, as SAC is able to increase its performance from 67.6 to 68.4 when replacing its standard DeepLabv2 decoder by the modified version proposed by ProDA.

We also observe in Tab. 2 that DG methods typically use a different semantic segmentation network architecture than UDA methods. Specifically, they use either a standard ResNet-101 [8] or a slight modification of it [5] (see Tab. 3), in combination with DeepLab v3+ [4] as decoder. This particular architecture choice impedes a fair comparison with UDA methods. Therefore, in our evaluation framework, we opt for a normalized architecture that all UDA and DG methods use. Concretely, we downgrade the quality of the decoder for DG methods (from DeepLab v3+ to the modified version of DeepLab v2 that ProDA uses) and maintain the standard ResNet-101 as encoder. As shown in Tab. 2, this causes a considerable drop in performance for both WildNet [9] and RobustNet [5]. We also considered using the standard Deeplab v2 or DeepLab v3+ decoder for all methods, as this would yield the best results, but running ProDA with the standard DeepLab with default hyperparameters led to experiment failure, i.e., the loss diverged during training. For this reason, we use the modified version of DeepLab v2 as proposed by ProDA for all methods.

Finally, we note that, even when RobustNet and WildNet are allowed to use DeepLab v3+, they still underperform ProDA with modified DeepLab v2 in terms of generalization to unseen domains. Specifically, RobustNet and WildNet achieve an mIoU avg. of 50.8 and 52.1, respectively, compared to 56.1 for ProDA. In other words, even with a less advanced semantic segmentation network, UDA method ProDA outperforms DG methods on unseen domains.

3. Histogram analysis for all datasets

In Sec. 4.2 of the main manuscript, we aimed to pick one *homogeneous* and one *heterogeneous* dataset from a set of six selected datasets, to use as labeled datasets $\mathcal{D}_l$ in our evaluation framework. As explained, the choice for Cityscapes [6] as *homogeneous* and Mapillary Vistas [12] as *heterogeneous* dataset, is mainly based on the knowledge that we have about the type of images that these datasets contain. Specifically, we know that Cityscapes contains images that are all captured with the same camera, mounted on the same vehicle, in urban environments in the same European country, in similar circumstances. Therefore, by design, this dataset is homogeneous. In contrast, Mapillary

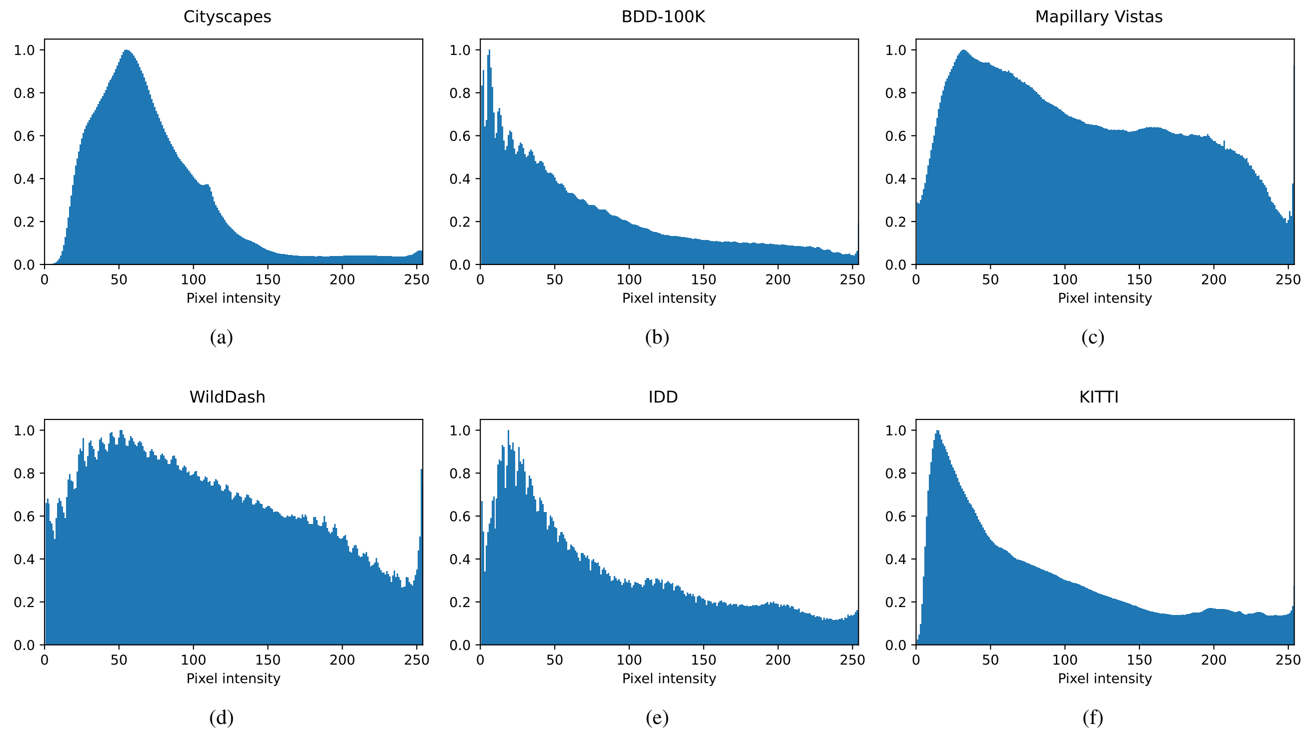

Figure 1. **Pixel intensity histograms for all datasets considered in our work:** Cityscapes [6], BDD-100K [17], Mapillary Vistas [12], WildDash [19], IDD [14] and KITTI [1]. If a dataset has a clear style, and has images captured under similar lighting conditions, we expect to see a distinct peak at certain intensity values. Therefore, this histogram analysis can provide us with some clues about the homogeneity or heterogeneity of a dataset.

Vistas is designed to contain images from all over the world, captured with many different types of cameras, in a plethora of different conditions and situations. In other words, this dataset is designed to be heterogeneous.

To further quantify this, we chose to conduct a histogram analysis of the pixel intensities of images in each dataset. Concretely, we count how often each pixel intensity occurs throughout the dataset, and for visualization purposes we scale the histograms such that the most frequent bin receives a value of 1. In contrast to the histograms provided in the main manuscript, we do not take intensity values 0 and 255 into account during scaling, to make the histograms better interpretable. In the histograms, we expect that, if a dataset is homogeneous and contains images captured under similar lighting conditions, there will be a distinct peak at certain intensity values. If there is no such peak, this indicates that the images in the dataset are captured with varying lighting conditions, meaning that the dataset is more likely to be heterogeneous.

In Fig. 1, we provide the pixel intensity histograms for all six considered datasets. From this figure, it is immediately clear that the pixel intensity distributions for WildDash [19] and Mapillary Vistas are much more uniform than the other datasets. This makes sense, because these datasets contain images captured under a wide range of different conditions, i.e., the datasets are heterogeneous. In contrast, Cityscapes has a very distinct peak at low-to-medium intensity values, and very rarely-occurring high intensity values. Again, this is explainable, because this homogeneous dataset contains images captured with the same camera, under very similar lighting conditions. We can see similar histograms for other homogeneous datasets, IDD [14] and KITTI [1]. Both have a clear peak, and have images captured under similar lighting conditions. Finally, the distribution for BDD-100K [17] is clearly skewed towards low pixel intensities, although it is supposed to be somewhat heterogeneous by design. In this case, this is caused by the fact that it contains many (low-brightness) nighttime images, in addition to the daytime images. This indicates that, although these histograms can provide interesting insights in the style that a dataset could have, they should not be the single resource that is used to determine if a dataset is homogeneous or heterogeneous. Therefore, in our work, we mainly chose the labeled datasets $\mathcal{D}_l$, unlabeled datasets $\mathcal{D}_{nl}$ and unseen datasets $\mathcal{D}_u$ based on the knowledge of how the datasets are collected, and what type of images they contain.

Considering the unlabeled datasets $\mathcal{D}_{nl}$ used in the two experimental settings, it is immediately clear that in ex-

task	method	road	sidewalk	building	wall	fence	pole	t. light	t. sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorcycle	bicycle
	Experiment 1.A: homogeneous labeled domain, heterogeneous unlabeled domain																			
	Labeled-domain-only	81.4	30.6	61.9	27.5	34.3	30.7	33.9	34.5	78.2	40.4	88.5	35.2	23.5	66.1	49.9	36.9	0.9	20.5	23.1
UDA	ProDA [20]	$\underline{90.1}^{\uparrow+8.7}$	$\underline{43.0}^{\uparrow+12.4}$	$\underline{75.7}^{\uparrow+13.8}$	$\underline{40.8}^{\uparrow+13.3}$	$\underline{38.5}^{\uparrow+4.2}$	$35.2^{\uparrow+4.5}$	$\underline{48.0}^{\uparrow+14.1}$	$\underline{58.7}^{\uparrow+24.2}$	$\underline{84.4}^{\uparrow+6.2}$	$48.8^{\uparrow+8.4}$	$92.1^{\uparrow+3.6}$	$\underline{\mathbf{63.2}}^{\uparrow+28.0}$	$\underline{44.9}^{\uparrow+21.4}$	$\underline{81.7}^{\uparrow+15.6}$	$\underline{70.1}^{\uparrow+20.2}$	$\underline{56.5}^{\uparrow+19.6}$	$4.1^{\uparrow+3.2}$	$\underline{55.9}^{\uparrow+35.4}$	$\underline{\mathbf{34.1}}^{\uparrow\mathbf{+11.0}}$
	SAC [2]	$88.2^{\uparrow+6.8}$	$37.7^{\uparrow+7.1}$	$71.0^{\uparrow+9.1}$	$37.8^{\uparrow+10.3}$	$36.0^{\uparrow+1.7}$	$\underline{37.4}^{\uparrow+6.7}$	$29.4^{\downarrow-4.5}$	$49.6^{\uparrow+15.1}$	$81.2^{\uparrow+3.0}$	$48.1^{\uparrow+7.7}$	$\underline{92.4}^{\uparrow+3.9}$	$49.1^{\uparrow+13.9}$	$34.5^{\uparrow+11.0}$	$73.9^{\uparrow+7.8}$	$60.6^{\uparrow+10.7}$	$51.8^{\uparrow+14.9}$	$4.4^{\uparrow+3.5}$	$38.2^{\uparrow+17.7}$	$16.5^{\downarrow-6.6}$
DG	WildNet [9]	$87.1^{\uparrow+5.7}$	$36.7^{\uparrow+6.1}$	$62.3^{\uparrow+0.4}$	$28.7^{\uparrow+1.2}$	$37.9^{\uparrow+3.6}$	$17.7^{\downarrow-13.0}$	$30.7^{\downarrow-3.2}$	$43.5^{\uparrow+9.0}$	$80.2^{\uparrow+2.0}$	$45.7^{\uparrow+5.3}$	$84.0^{\downarrow-4.5}$	$50.9^{\uparrow+15.7}$	$35.7^{\uparrow+12.2}$	$75.8^{\uparrow+9.7}$	$64.4^{\uparrow+14.5}$	$52.7^{\uparrow+15.8}$	$\underline{6.4}^{\uparrow+5.5}$	$45.2^{\uparrow+24.7}$	$21.0^{\downarrow-2.1}$
	RobustNet [5]	$83.2^{\uparrow+1.8}$	$37.6^{\uparrow+7.0}$	$63.5^{\uparrow+1.6}$	$27.3^{\downarrow-0.2}$	$37.0^{\uparrow+2.7}$	$27.3^{\downarrow-3.4}$	$34.4^{\uparrow+0.5}$	$49.9^{\uparrow+15.4}$	$80.4^{\uparrow+2.2}$	$\underline{52.4}^{\uparrow+12.0}$	$86.7^{\downarrow-1.8}$	$50.9^{\uparrow+15.7}$	$33.2^{\uparrow+9.7}$	$70.7^{\uparrow+4.6}$	$52.2^{\uparrow+2.3}$	$52.1^{\uparrow+15.2}$	$3.2^{\uparrow+2.3}$	$34.6^{\uparrow+14.1}$	$21.0^{\downarrow-2.1}$
	Fully sup. training domains	$89.9^{\uparrow+8.5}$	$39.1^{\uparrow+8.5}$	$76.4^{\uparrow+14.5}$	$43.4^{\uparrow+15.9}$	$42.3^{\uparrow+8.0}$	$44.2^{\uparrow+13.5}$	$43.1^{\uparrow+9.2}$	$57.3^{\uparrow+22.8}$	$86.0^{\uparrow+7.8}$	$56.8^{\uparrow+16.4}$	$94.4^{\uparrow+5.9}$	$55.6^{\uparrow+20.4}$	$43.1^{\uparrow+19.6}$	$83.4^{\uparrow+17.3}$	$69.8^{\uparrow+19.9}$	$59.2^{\uparrow+22.3}$	$4.0^{\uparrow+3.1}$	$41.6^{\uparrow+21.1}$	$25.0^{\uparrow+1.9}$
	Fully sup. unseen domains	$90.9^{\uparrow+9.5}$	$44.9^{\uparrow+14.3}$	$74.4^{\uparrow+12.5}$	$43.2^{\uparrow+15.7}$	$35.2^{\uparrow+0.9}$	$39.8^{\uparrow+9.1}$	$30.7^{\downarrow-3.2}$	$45.6^{\uparrow+11.1}$	$85.1^{\uparrow+6.9}$	$61.5^{\uparrow+21.1}$	$94.2^{\uparrow+5.7}$	$49.7^{\uparrow+14.5}$	$35.3^{\uparrow+11.8}$	$82.5^{\uparrow+16.4}$	$65.7^{\uparrow+15.8}$	$58.3^{\uparrow+21.4}$	$3.2^{\uparrow+2.3}$	$38.4^{\uparrow+17.9}$	$2.7^{\downarrow-20.4}$
	Fully sup. all domains	$\mathbf{93.3}^{\uparrow\mathbf{+11.9}}$	$\mathbf{58.4}^{\uparrow\mathbf{+27.8}}$	$\mathbf{80.4}^{\uparrow\mathbf{+18.5}}$	$\mathbf{56.4}^{\uparrow\mathbf{+28.9}}$	$\mathbf{49.7}^{\uparrow\mathbf{+15.4}}$	$\mathbf{49.8}^{\uparrow\mathbf{+19.1}}$	$\mathbf{48.9}^{\uparrow\mathbf{+15.0}}$	$\mathbf{61.3}^{\uparrow\mathbf{+26.8}}$	$\mathbf{87.5}^{\uparrow\mathbf{+9.3}}$	$\mathbf{66.2}^{\uparrow\mathbf{+25.8}}$	$\mathbf{95.5}^{\uparrow\mathbf{+7.0}}$	$62.8^{\uparrow+27.6}$	$\mathbf{50.0}^{\uparrow\mathbf{+26.5}}$	$\mathbf{88.1}^{\uparrow\mathbf{+22.0}}$	$\mathbf{79.3}^{\uparrow\mathbf{+29.4}}$	$\mathbf{77.4}^{\uparrow\mathbf{+40.5}}$	$\mathbf{6.9}^{\uparrow\mathbf{+6.0}}$	$\mathbf{57.0}^{\uparrow\mathbf{+36.5}}$	$\mathbf{34.1}^{\uparrow\mathbf{+11.0}}$
	Experiment 1.B: heterogeneous labeled domain, homogeneous unlabeled domain																			
	Labeled-domain-only	89.7	39.2	74	38.6	39.7	42.7	40.6	51.4	84.4	50.6	93.5	46.5	36.6	80.8	68.2	56.5	0.2	33.9	21.6
UDA	ProDA [20]	$\underline{92.3}^{\uparrow+2.6}$	$\underline{47.8}^{\uparrow+8.6}$	$\underline{77.8}^{\uparrow+3.8}$	$\underline{43.1}^{\uparrow+4.5}$	$\underline{47.2}^{\uparrow+7.5}$	$34.3^{\downarrow-8.4}$	$\underline{47.3}^{\uparrow+6.7}$	$\underline{58.3}^{\uparrow+6.9}$	$\underline{85.4}^{\uparrow+1.0}$	$\underline{54.0}^{\uparrow+3.4}$	$93.7^{\uparrow+0.2}$	$\underline{61.0}^{\uparrow+14.5}$	$\underline{35.6}^{\downarrow-1.0}$	$\underline{83.6}^{\uparrow+2.8}$	$\underline{76.3}^{\uparrow+8.1}$	$\underline{67.4}^{\uparrow+10.9}$	$\underline{3.2}^{\uparrow+3.0}$	$\underline{44.5}^{\uparrow+10.6}$	$\underline{25.9}^{\uparrow+4.3}$
	SAC [2]	$88.9^{\downarrow-0.8}$	$43.0^{\uparrow+3.8}$	$74.9^{\uparrow+0.9}$	$38.5^{\downarrow-0.1}$	$40.5^{\uparrow+0.8}$	$\underline{45.7}^{\uparrow+3.0}$	$43.0^{\uparrow+2.4}$	$52.2^{\uparrow+0.8}$	$84.8^{\uparrow+0.4}$	$52.2^{\uparrow+1.6}$	$\underline{94.1}^{\uparrow+0.6}$	$51.6^{\uparrow+5.1}$	$32.2^{\downarrow-4.4}$	$78.2^{\downarrow-2.6}$	$66.7^{\downarrow-1.5}$	$56.5^{\downarrow0.0}$	$0.0^{\downarrow-0.2}$	$34.4^{\uparrow+0.5}$	$21.0^{\downarrow-0.6}$
DG	WildNet [9]	$88.3^{\downarrow-1.4}$	$35.4^{\downarrow-3.8}$	$73.8^{\downarrow-0.2}$	$32.0^{\downarrow-6.6}$	$30.3^{\downarrow-9.4}$	$24.2^{\downarrow-18.5}$	$31.6^{\downarrow-9.0}$	$40.8^{\downarrow-10.6}$	$82.0^{\downarrow-2.4}$	$52.8^{\uparrow+2.2}$	$90.9^{\downarrow-2.6}$	$47.6^{\uparrow+1.1}$	$31.9^{\downarrow-4.7}$	$74.1^{\downarrow-6.7}$	$58.6^{\downarrow-9.6}$	$55.7^{\downarrow-0.8}$	$0.0^{\downarrow-0.2}$	$27.9^{\downarrow-6.0}$	$11.9^{\downarrow-9.7}$
	RobustNet [5]	$89.3^{\downarrow-0.4}$	$39.4^{\uparrow+0.2}$	$73.8^{\downarrow-0.2}$	$34.8^{\downarrow-3.8}$	$36.9^{\downarrow-2.8}$	$33.0^{\downarrow-9.7}$	$41.2^{\uparrow+0.6}$	$47.3^{\downarrow-4.1}$	$83.3^{\downarrow-1.1}$	$53.9^{\uparrow+3.3}$	$92.4^{\downarrow-1.1}$	$46.2^{\downarrow-0.3}$	$30.1^{\downarrow-6.5}$	$75.9^{\downarrow-4.9}$	$57.8^{\downarrow-10.4}$	$51.9^{\downarrow-4.6}$	$0.0^{\downarrow-0.2}$	$24.9^{\downarrow-9.0}$	$9.6^{\downarrow-12.0}$
	Fully sup. training domains	$89.9^{\uparrow+0.2}$	$39.1^{\downarrow-0.1}$	$76.4^{\uparrow+2.4}$	$43.4^{\uparrow+4.8}$	$42.3^{\uparrow+2.6}$	$44.2^{\uparrow+1.5}$	$43.1^{\uparrow+2.5}$	$57.3^{\uparrow+5.9}$	$86.0^{\uparrow+1.6}$	$56.8^{\uparrow+6.2}$	$94.4^{\uparrow+0.9}$	$55.6^{\uparrow+9.1}$	$43.1^{\uparrow+6.5}$	$83.4^{\uparrow+2.6}$	$69.8^{\uparrow+1.6}$	$59.2^{\uparrow+2.7}$	$4.0^{\uparrow+3.8}$	$41.6^{\uparrow+7.7}$	$25.0^{\uparrow+3.4}$
	Fully sup. unseen domains	$90.9^{\uparrow+1.2}$	$44.9^{\uparrow+5.7}$	$74.4^{\uparrow+0.4}$	$43.2^{\uparrow+4.6}$	$35.2^{\downarrow-4.5}$	$39.8^{\downarrow-2.9}$	$30.7^{\downarrow-9.9}$	$45.6^{\downarrow-5.8}$	$85.1^{\uparrow+0.7}$	$61.5^{\uparrow+10.9}$	$94.2^{\uparrow+0.7}$	$49.7^{\uparrow+3.2}$	$35.3^{\downarrow-1.3}$	$82.5^{\uparrow+1.7}$	$65.7^{\downarrow-2.5}$	$58.3^{\uparrow+1.8}$	$3.2^{\uparrow+3.0}$	$38.4^{\uparrow+4.5}$	$2.7^{\downarrow-18.9}$
	Fully sup. all domains	$\mathbf{93.3}^{\uparrow\mathbf{+3.6}}$	$\mathbf{58.4}^{\uparrow\mathbf{+19.2}}$	$\mathbf{80.4}^{\uparrow\mathbf{+6.4}}$	$\mathbf{56.4}^{\uparrow\mathbf{+17.8}}$	$\mathbf{49.7}^{\uparrow\mathbf{+10.0}}$	$\mathbf{49.8}^{\uparrow\mathbf{+7.1}}$	$\mathbf{48.9}^{\uparrow\mathbf{+8.3}}$	$\mathbf{61.3}^{\uparrow\mathbf{+9.9}}$	$\mathbf{87.5}^{\uparrow\mathbf{+3.1}}$	$\mathbf{66.2}^{\uparrow\mathbf{+15.6}}$	$\mathbf{95.5}^{\uparrow\mathbf{+2.0}}$	$\mathbf{62.8}^{\uparrow\mathbf{+16.3}}$	$\mathbf{50.0}^{\uparrow\mathbf{+13.4}}$	$\mathbf{88.1}^{\uparrow\mathbf{+7.3}}$	$\mathbf{79.3}^{\uparrow\mathbf{+11.1}}$	$\mathbf{77.4}^{\uparrow\mathbf{+20.9}}$	$\mathbf{6.9}^{\uparrow\mathbf{+6.7}}$	$\mathbf{57.0}^{\uparrow\mathbf{+23.1}}$	$\mathbf{34.1}^{\uparrow\mathbf{+12.5}}$

Table 4. **Quantitative comparison on unseen domains WildDash [19], IDD [14] and KITTI [1], averaged per class.** The reported deltas (in green and red) are with respect to each experiment's baseline. The highest mIoU per category is highlighted as follows: 1) bold, when comparing all methods, and 2) underlined, excluding fully supervised models in the comparison (UDA and DG methods only). Best viewed digitally.

periment 1.A, mixing BDD-100K with Mapillary Vistas leads to a highly heterogeneous unlabeled domain, as it consists of Mapillary Vistas, which is already heterogeneous by itself. In experiment 1.B, mixing BDD-100K with Cityscapes leads to a dataset with a lower variability, as both datastets contain images captured at very specific geographical locations. As a result, there are two distinct settings that could provide interesting results, as presented in Sec. 5 of our main manuscript.

Finally, the unseen domain $\mathcal{D}_u$, contains one clear heterogeneous dataset with WildDash, complemented by two homogeneous datasets, IDD and KITTI. Because these different types of datasets are used as $\mathcal{D}_u$, it is very challenging for UDA or DG methods to generalize well to all these unseen datasets, which is a desired property, as these datasets should represent *the wild* in our evaluation framework.

4. Generalization to unseen domains: class analysis

Tab. 4 shows the per-class mIoU scores of UDA, DG and fully supervised methods on the unseen domains for experiments 1.A and 1.B. Results are averaged over the three datasets representing the unseen domains to obtain a single value per class.

From this table, it can be noticed that overall, the fully supervised model trained on all domains provides a strong 'upper bound' for both experiments 1.A and 1.B, as it outperforms all methods over the majority of the classes. Furthermore, comparing just the UDA and DG methods (best results underlined), it can be seen that the best performing UDA method ProDA [20], outperforms the other UDA and DG methods on 15 classes in experiment 1.A, and on 17 classes in experiment 1.B. These results show shows that domain adaptation methods can improve generalization to unseen domains for most of the classes, without overfitting on any specific group of categories such as foreground or background classes. Finally, it is worth noting that, in experiment 1.A, ProDA even outperforms the fully supervised model trained on all domains for a single class (*person*). Although this happens for only one class, it is important to remember that ProDA achieves this by using only 7.7% of the amount of labeled data that the fully supervised model has access to during training, highlighting the benefit of using abundant unlabeled data.

5. Discussion on combining UDA and DG strategies

In Sec. 5 of the main manuscript, we found that UDA methods consistently outperform existing DG methods on unseen domains. This is especially true when the labeled domain is heterogeneous (i.e., in experiment 1.B), as DG methods even underperform the labeled-domain-only baseline in that setting. However, DG methods do considerably outperform the baseline in experiment 1.A, where the labeled domain is homogeneous. Although they do not outperform UDA methods, they still achieve a considerable performance boost. Moreover, UDA and DG methods achieve their performance boosts with different methods, by approaching the problem from different perspectives. Therefore, these types of methods could be complementary to each other, and combining UDA and DG strategies to achieve even better results may have potential. Specifically, the efforts of DG methods to enlarge the training domain could be combined with the techniques that UDA methods use to leverage unlabeled data. We hope that our work sparks ideas for follow-up research in this direction.

References

[1] Hassan Alhaija, Siva Mustikovela, Lars Mescheder, Andreas Geiger, and Carsten Rother. Augmented reality meets computer vision: Efficient data generation for urban driving scenes. *IJCV*, 2018.

[2] Nikita Araslanov and Stefan Roth. Self-supervised augmentation consistency for adapting semantic segmentation. In *CVPR*, pages 15384–15394, 2021.

[3] 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. *TPAMI*, 2018.

[4] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *ECCV*, 2018.

[5] Sungha Choi, Sanghun Jung, Huiwon Yun, Joanne Taery Kim, Seungryong Kim, and Jaegul Choo. Robustnet: Improving domain generalization in urban-scene segmentation via instance selective whitening. In *CVPR*, pages 11580–11590, 2021.

[6] 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 *CVPR*, pages 3213–3223, 2016.

[7] Li Gao, Jing Zhang, Lefei Zhang, and Dacheng Tao. DSP: dual soft-paste for unsupervised domain adaptive semantic segmentation. In *ACM Multimedia*, pages 2825–2833, 2021.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[9] Suhyeon Lee, Hongje Seong, Seongwon Lee, and Euntai Kim. Wildnet: Learning domain generalized semantic segmentation from the wild. *CoRR*, abs/2204.01446, 2022.

[10] Y. Li, L. Yuan, and N. Vasconcelos. Bidirectional learning for domain adaptation of semantic segmentation. *CVPR*, pages 6929–6938, 2019.

[11] Haoyu Ma, Xiangru Lin, Zifeng Wu, and Yizhou Yu. Coarse-to-fine domain adaptive semantic segmentation with photometric alignment and category-center regularization. In *CVPR*, pages 4051–4060, 2021.

[12] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *ICCV*, pages 5000–5009, 2017.

[13] Fabrizio J. Piva and Gijs Dubbelman. Exploiting image translations via ensemble self-supervised learning for unsupervised domain adaptation. *CoRR*, abs/2107.06235, 2021.

[14] Girish Varma, Anbumani Subramanian, Anoop M. Namboodiri, Manmohan Chandraker, and C. V. Jawahar. IDD: A dataset for exploring problems of autonomous navigation in unconstrained environments. In *WACV*, pages 1743–1751, 2019.

[15] Qin Wang, Dengxin Dai, Lukas Hoyer, Olga Fink, and Luc Van Gool. Domain adaptive semantic segmentation with self-supervised depth estimation. *CoRR*, abs/2104.13613, 2021.

[16] Yuxin Wu and Kaiming He. Group Normalization. In *ECCV*, 2018.

[17] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. BDD100K: A diverse driving dataset for heterogeneous multitask learning. In *CVPR*, pages 2633–2642, 2020.

[18] Xiangyu Yue, Yang Zhang, Sicheng Zhao, Alberto L. Sangiovanni-Vincentelli, Kurt Keutzer, and Boqing Gong. Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data. In *ICCV*, pages 2100–2110. IEEE, 2019.

[19] Oliver Zendel, Katrin Honauer, Markus Murschitz, Daniel Steininger, and Gustavo Fernández Domínguez. Wilddash - creating hazard-aware benchmarks. In *ECCV*, pages 407–421, 2018.

[20] 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 *CVPR*, pages 12414–12424, 2021.

[21] Qianyu Zhou, Zhengyang Feng, Qiqi Gu, Jiangmiao Pang, Guangliang Cheng, Xuequan Lu, Jianping Shi, and Lizhuang Ma. Context-aware mixup for domain adaptive semantic segmentation. *CoRR*, abs/2108.03557, 2021.