Cross-task Attention Mechanism for Dense Multi-task Learning
– Supplementary Material –

Ivan Lopes
Inria
ivan.lopes@inria.fr

Tuan-Hung Vu
Valeo.ai, Inria
tuan-hung.vu@valeo.com

Raoul de Charette
Inria
raoul.de-charette@inria.fr

1. Method details

We provide details for the training of our method and some ablations of our multi-task exchange block (mTEB).

1.1. Training

Considering $T = \{T_1, \dots, T_n\}$ the set of n tasks to be jointly optimized, our general MTL training loss is a weighted combination of individual tasks losses and writes:

$$\mathcal{L}_{\text{tasks}} = \frac{1}{|S|} \sum_{s \in S} \sum_{t \in T} \omega_t \mathcal{L}_t^s + \sum_{t \in T} \omega_t \mathcal{L}_t^{\text{final}}, \tag{1}$$

where ω_t is the task-balancing weight, $\mathcal{L}_t$ is the task-specific supervision loss at intermediate scale s (*i.e.* $\mathcal{L}_t^s$) or at the final prediction stage (*i.e.* $\mathcal{L}_t^{\text{final}}$). See Sec. 4.1.1 of main paper for details on the loss functions used. Additionally, S defines the intermediate scales of supervision considering scale s to be the task intermediate output at resolution $\frac{1}{2^s}$ w.r.t. input resolution. In practice, to enforce cross-task exchange without direct mTEB supervision, we set S equal to the scales at which the mTEB are inserted. In our method, we keep a single mTEB and set $S = \{1\}$.

All our models are trained on a single V100 32g GPU and take between 10 and 20 hours to converge depending on the task set and dataset size.

1.2. Ablations

We study the overall benefit of our multi-task exchange block (mTEB) on VKITTI2 with our complete architecture. To further demonstrate the effect of attention mechanisms in our method, we remove self-attention from the directional features of xTAM, so that Eq. (4) of main paper now writes: $f_{j \to i} = [\text{diag}(\alpha_1, ..., \alpha_c) \times \mathbf{xtask}_{j \to i}]$. We report the performance of 'S-D-N' with spatial cross-task attention. *Without* self-attention we get 97.40/3.556/**14.39**‖30.30 for mIoU/RMSE/mErr‖Δ_{SDN}, and **97.43**/**3.315**/14.83‖**31.08** *with* self-attention. This shows the two types of attention are best combined.

2. Experimental details

2.1. Multi-task setup

2.1.1 Metrics.

In the following paragraph we detail the four task-specific metrics used throughout our paper.

- **Semantic segmentation** uses mIoU as the average of the per-class Intersection over Union (%) between label s and predicted map $\hat{s}$: $m_S = \text{mIoU}(\hat{s}, s)$.

- **Depth regression** uses the Root Mean Square Error computed between label d and predicted map $\hat{d}$: $m_D = \text{RMSE}(\hat{d}, d)$, reporting the RMSE in meters over the evaluated set of images. In the SDE and DA setups, a *per*-image median scaling [12] is applied since the models used are *not* scale-aware.

- **Normals estimation**, we measure the absolute angle error in degrees between the label n and predicted map $\hat{n}$: $m_N = \text{deg}(\hat{n}, n)$. For all datasets we retrieve labels from the depth map [10]: we unproject the pixels using the camera intrinsics and depth values, then compute the cross-product using neighboring points (from 2D perspective) [3] and average over pairs of neighbors [11]. Cityscapes [1] provides disparity maps which we use to compute noisy surface normals labels.

- **Edge estimation**, we apply the F1-score between the predicted and ground-truth maps: $m_E = \text{F1}(\hat{e}, e)$. [6] provides ground-truth semantic edges for NYUDv2.

2.1.2 Task balancing.

Table 1 reports a subset of our grid-search to select an optimal set of weights for both the 'S-D' and 'S-D-N' sets. To avoid favoring a specific task or model, the evaluation is conducted on the 'MTL' baseline model and we select the set of weights from best Δ_{T} metrics.

weights ω_S	ω_D	Semseg ↑ mIoU %	Depth ↓ RMSE m	Delta ↑ Δ_{SD} %
1	1	83.83 $_{\pm 0.15}$	5.713 $_{\pm 0.060}$	-0.35 $_{\pm 0.47}$
1	10	79.87 $_{\pm 0.21}$	<u>5.708</u> $_{\pm 0.036}$	-2.66 $_{\pm 0.40}$
10	1	86.20 $_{\pm 0.71}$	**5.693** $_{\pm 0.055}$	+1.30 $_{\pm 0.22}$
50	1	87.73 $_{\pm 0.12}$	5.720 $_{\pm 0.029}$	**+1.89** $_{\pm 0.21}$
100	1	88.00 $_{\pm 0.20}$	5.754 $_{\pm 0.030}$	<u>+1.75</u> $_{\pm 0.17}$
100	10	86.13 $_{\pm 0.32}$	**5.693** $_{\pm 0.039}$	+1.18 $_{\pm 0.45}$
200	1	<u>88.13</u> $_{\pm 0.12}$	5.790 $_{\pm 0.055}$	+1.52 $_{\pm 0.45}$
500	1	**88.17** $_{\pm 0.15}$	5.847 $_{\pm 0.043}$	+1.04 $_{\pm 0.30}$

(a) 'S-D' gridsearch

weights ω_S	ω_D	ω_N	Semseg ↑ mIoU %	Depth ↓ RMSE m	Normals ↓ mErr. °	Delta ↑ Δ_{SDN} %
1	1	1	83.50 $_{\pm 0.20}$	5.707 $_{\pm 0.058}$	23.03 $_{\pm 0.70}$	-0.17 $_{\pm 0.64}$
10	1	1	86.53 $_{\pm 0.21}$	<u>5.694</u> $_{\pm 0.032}$	22.98 $_{\pm 0.68}$	+1.17 $_{\pm 0.93}$
10	1	10	86.63 $_{\pm 0.21}$	**5.675** $_{\pm 0.050}$	22.61 $_{\pm 0.70}$	+1.85 $_{\pm 0.80}$
50	1	1	87.73 $_{\pm 0.21}$	5.706 $_{\pm 0.051}$	22.90 $_{\pm 0.71}$	+1.69 $_{\pm 0.81}$
50	1	10	87.77 $_{\pm 0.15}$	5.714 $_{\pm 0.065}$	22.56 $_{\pm 0.69}$	+2.15 $_{\pm 0.86}$
50	1	50	87.73 $_{\pm 0.21}$	5.701 $_{\pm 0.062}$	22.37 $_{\pm 0.70}$	+2.49 $_{\pm 0.76}$
100	1	1	88.03 $_{\pm 0.15}$	5.746 $_{\pm 0.030}$	22.95 $_{\pm 0.69}$	+1.49 $_{\pm 0.92}$
100	1	10	87.97 $_{\pm 0.15}$	5.714 $_{\pm 0.048}$	22.59 $_{\pm 0.69}$	+2.19 $_{\pm 0.79}$
100	1	50	88.00 $_{\pm 0.20}$	5.717 $_{\pm 0.048}$	22.40 $_{\pm 0.71}$	+2.45 $_{\pm 0.99}$
100	1	100	88.07 $_{\pm 0.15}$	5.696 $_{\pm 0.038}$	**22.29** $_{\pm 0.70}$	**+2.75** $_{\pm 1.04}$
150	1	10	<u>88.10</u> $_{\pm 0.20}$	5.752 $_{\pm 0.059}$	22.59 $_{\pm 0.70}$	+2.01 $_{\pm 0.86}$
150	1	50	<u>88.10</u> $_{\pm 0.20}$	5.738 $_{\pm 0.039}$	22.41 $_{\pm 0.70}$	+2.35 $_{\pm 0.99}$
150	1	100	**88.13** $_{\pm 0.15}$	5.732 $_{\pm 0.037}$	<u>22.31</u> $_{\pm 0.71}$	<u>+2.54</u> $_{\pm 0.94}$

(b) 'S-D-N' gridsearch

Table 1: Performance of the 'MTL' baseline model (cf. Fig. 1) for different sets of multi-task weights on VKITTI2. There are important remarks. First, uniform weighting is far from optimal. Second, best Δ_T does not always equate to optimal individual metrics as shown by the results in **bold**. Ultimately, to avoid favoring a single task, we use the set of weights with highest Δ_T metric for all models, as highlighted in gray .

2.2. Main results

2.2.1 General architectures.

In Fig. 1 we show the general architectures (*i.e.*, considering depth supervision), including the STL, MTL and PAD-Net models [5], 3-ways$_{\text{PAD-Net}}$ [5] and Ours. Based on the same encoder taken from a pretrained ResNet-101 backbone [4], those multi-task networks differ only in decoder design. We observe improvements in all tasks using both Atrous Spatial Pyramid Pooling (ASPP) and UNet-like connections as done in [5] (cf. 3-ways$_{\text{PAD-Net}}$ *vs.* PAD-Net).

2.2.2 SDE architectures.

To allow monocular depth estimation in 'MTL for segmentation', we adopt the setup of [5] where intermediate depth estimation from pair of consecutive frames is supervised by a photometric reconstruction loss [2]. Fig. 2 shows the architecture used for all 3-ways variants for the semantics training with SDE. Variants consist of swapping the yellow 'Exchange block' with either 'PAD-Net block' (3-ways$_{\text{PAD-Net}}$) or our 'mTEB' (3-ways$_{\text{mTEB}}$).

To train, we use $1.0e{-}5$, $5.0e{-}5$, and $1.0e{-}6$ as learning rates for the encoder, decoder and pose estimation network respectively. The training strategy is similar to our other MTL setups, only this time we initialize all models with weights from a single-branch model trained on self-supervised depth estimation (cf. [5]). Since the depth loss differs from the supervised ones, we do *not* apply the weighting found for 'S-D' but instead resort to uniform weighting for direct comparison to [5].

2.3. MTL for Unsupervised Domain Adaptation

2.3.1 Architecture and training.

Fig. 3 illustrates our adversarial learning scheme with source/target data flows for multi-task UDA. We consider the two-task 'S-D' setup. As explained in our paper, domain alignment is made possible with output-level DA adversarial training. In our work, alignment is done at both intermediate and final output-levels.

We follow the strategies introduced in [7, 8]. Discriminators D are train on the source dataset $\mathcal{X}_{src}$ and target dataset $\mathcal{X}_{trg}$ by minimizing the binary classification loss:

$$\mathcal{L}_{\text{D}} = \frac{1}{|\mathcal{X}_{src}|} \sum_{\boldsymbol{x}_{src} \in \mathcal{X}_{src}} \mathcal{L}_{\text{BCE}}(\text{D}(\mathcal{Q}_{\boldsymbol{x}_{src}}), 1) + \frac{1}{|\mathcal{X}_{trg}|} \sum_{\boldsymbol{x}_{trg} \in \mathcal{X}_{trg}} \mathcal{L}_{\text{BCE}}(\text{D}(\mathcal{Q}_{\boldsymbol{x}_{trg}}), 0), \quad (2)$$

where $\mathcal{L}_{\text{BCE}}$ is the Binary Cross-Entropy loss, and $\mathcal{Q}_{\boldsymbol{x}}$ stands for either segmentation output $\mathcal{Q}^S_{\boldsymbol{x}}$ or depth output $\mathcal{Q}^D_{\boldsymbol{x}}$ of the network. To compete with the discriminators, the main MTL network is additionally trained with the adversarial losses $\mathcal{L}_{adv}$, written as:

$$\mathcal{L}_{adv} = \frac{1}{|\mathcal{X}_{trg}|} \sum_{\boldsymbol{x}_{trg} \in \mathcal{X}_{trg}} \mathcal{L}_{\text{BCE}}(\text{D}(\mathcal{Q}_{\boldsymbol{x}_{trg}}), 1). \quad (3)$$

The final MTL-UDA loss becomes:

$$\mathcal{L}_{\text{MTL-UDA}} = \frac{1}{|S|} \sum_{s \in S} \sum_{t \in T} (\omega_t \mathcal{L}^s_t + \lambda_{adv} \mathcal{L}^{\text{s}}_{adv_t}) + \sum_{t \in T} (\omega_t \mathcal{L}^{\text{final}}_t + \lambda_{adv} \mathcal{L}^{\text{final}}_{adv_t}), \quad (4)$$

where λ_{adv} is used to weight the adversarial losses and is set to $5.0e{-}3$.

For **segmentation** alignment, we use "weighted self-information" map [7] computed from the softmax segmentation output $\mathcal{P}_{\boldsymbol{x}}$ with the formula:

$$\mathcal{Q}^S_{\boldsymbol{x}} = -\mathcal{P}_{\boldsymbol{x}} \odot \log(\mathcal{P}_{\boldsymbol{x}}). \quad (5)$$

For **depth** alignment, we normalize the depth-map outputs using the source's min and max depth values, and directly align the continuous normalized maps $\mathcal{Q}^D_{\boldsymbol{x}}$ [8].

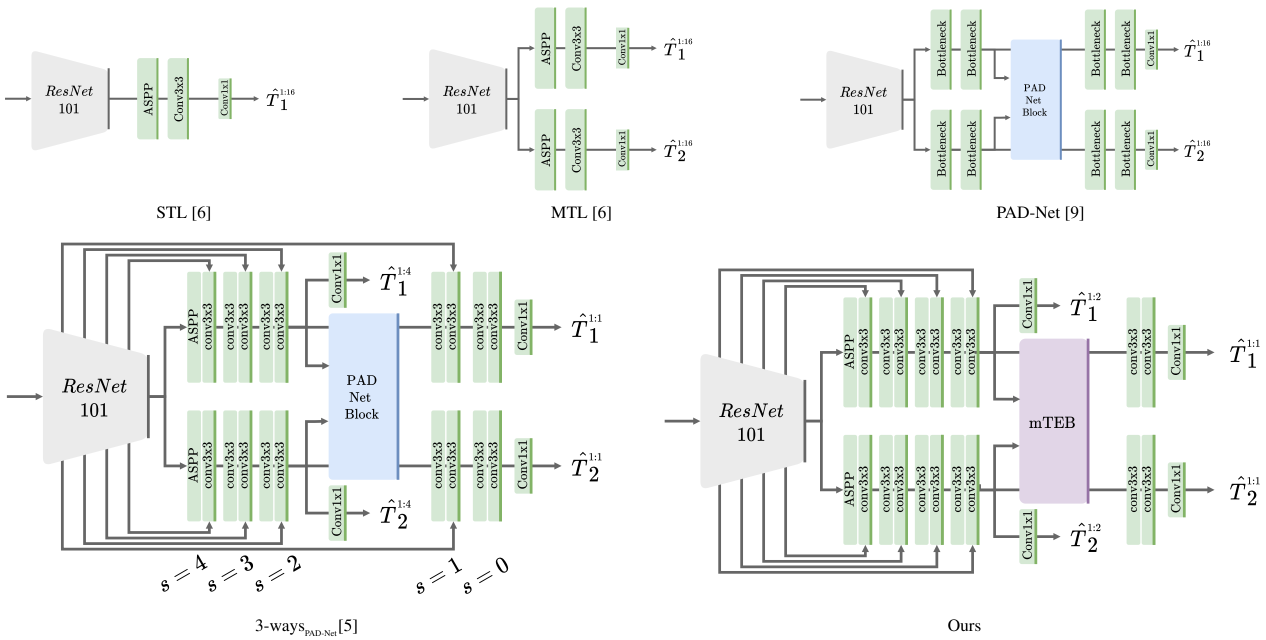

Figure 1: **General architectures.** For clarity, we only visualize two tasks in the multi-task networks. While the encoder is identical, models differ in their decoder architecture, with PAD-Net, 3-ways$_{\text{PAD-Net}}$ and Ours using dedicated tasks exchange blocks.

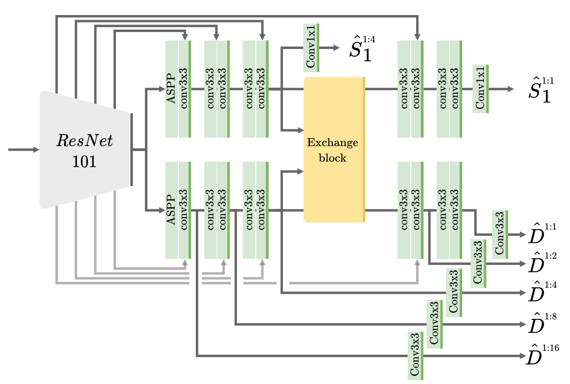

Figure 2: **Architecture for Self-supervised Depth Estimation (SDE).** To accommodate monocular depth on 'Cityscapes SDE', we follow the setup of [5] with added intermediate depth supervision ($\hat{D}^{1:\mathbf{x}}$). For the two variants in the SDE setup, we use the above architecture, replacing the 'exchange block' with the desired one.

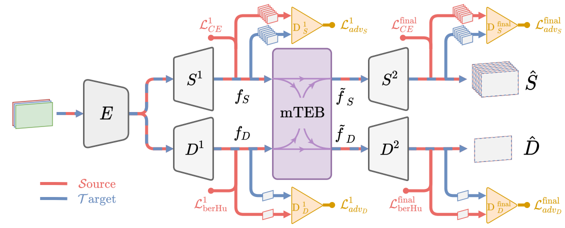

Figure 3: **Multi-task UDA.** Arrows indicating data flows are drawn in either red (source), blue (target) or a mix (both). Additional discriminators (shown as yellow triangles) are jointly trained with our multi-task model.

VKITTI2	mapped	Cityscapes	mapped
terrain	*ignore*	road	road
sky	sky	sidewalk	*ignore*
tree	vegetation	building	building
vegetation	vegetation	wall	vegetation
building	building	fence	*ignore*
road	road	pole	pole
guardrail	*ignore*	light	light
sign	sign	sign	sign
light	light	vegetation	vegetation
pole	pole	sky	sky
misc	*ignore*	person	*ignore*
truck	vehicle	rider	*ignore*
car	vehicle	car	vehicle
van	vehicle	bus	vehicle
		mbike	*ignore*
		bike	*ignore*

Table 2: Class mapping for VKITTI2↦Cityscapes DA setup.

2.3.2 Class mapping.

To allow compatible semantics in the VKITTI2↦ Cityscapes setup, we adopt the mapping of Table 2.

3. Additional results

Figs. 4 to 6 show additional qualitative results for Synthia, VKITTI2, and Cityscapes, respectively. Comparing *Ours* with *PAD-Net* show an evident segmentation improvement on thin elements such as poles or pedestrians in Figs. 4 and 6 with significantly sharper results for depth and normals across setups. Comparing against *3-ways$_{PAD\text{-}Net}$* is harder due to their high scores (cf. main paper Tab. 1).

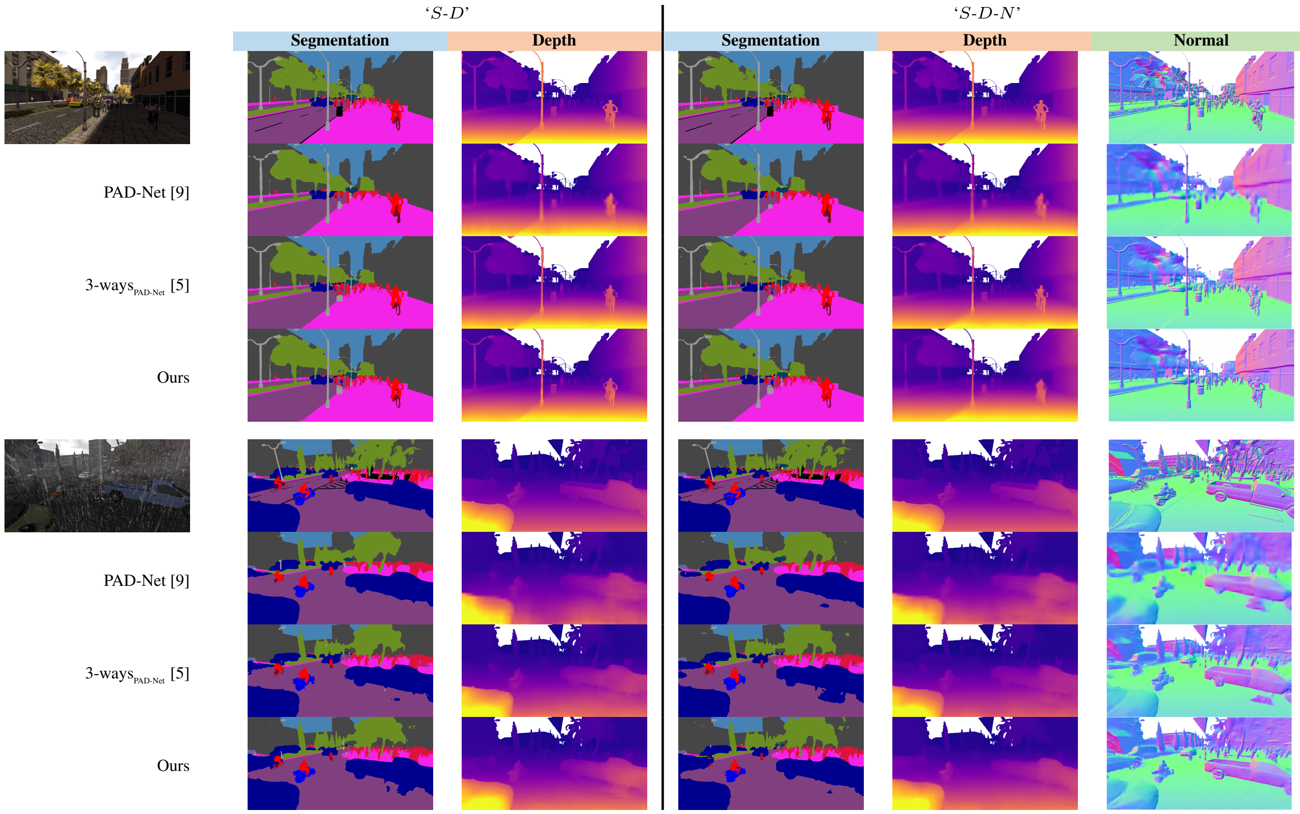

Figure 4: **Qualitative results on Synthia.** Overall, *Ours* produces better and sharper. Comparing visually against 3-ways$_{\text{PAD-Net}}$ is harder due to high scores.

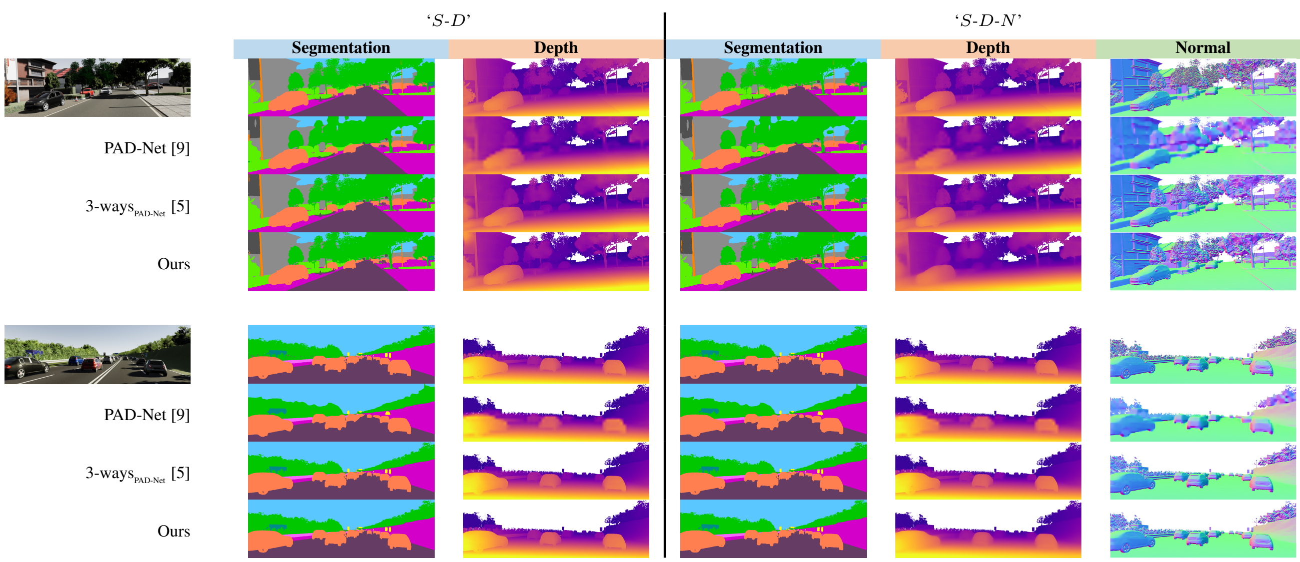

Figure 5: **Qualitative results on VKITTI2**. Overall, *Ours* produces better and sharper. Comparing against 3-ways$_{\text{PAD-Net}}$ is harder visually due to high scores.

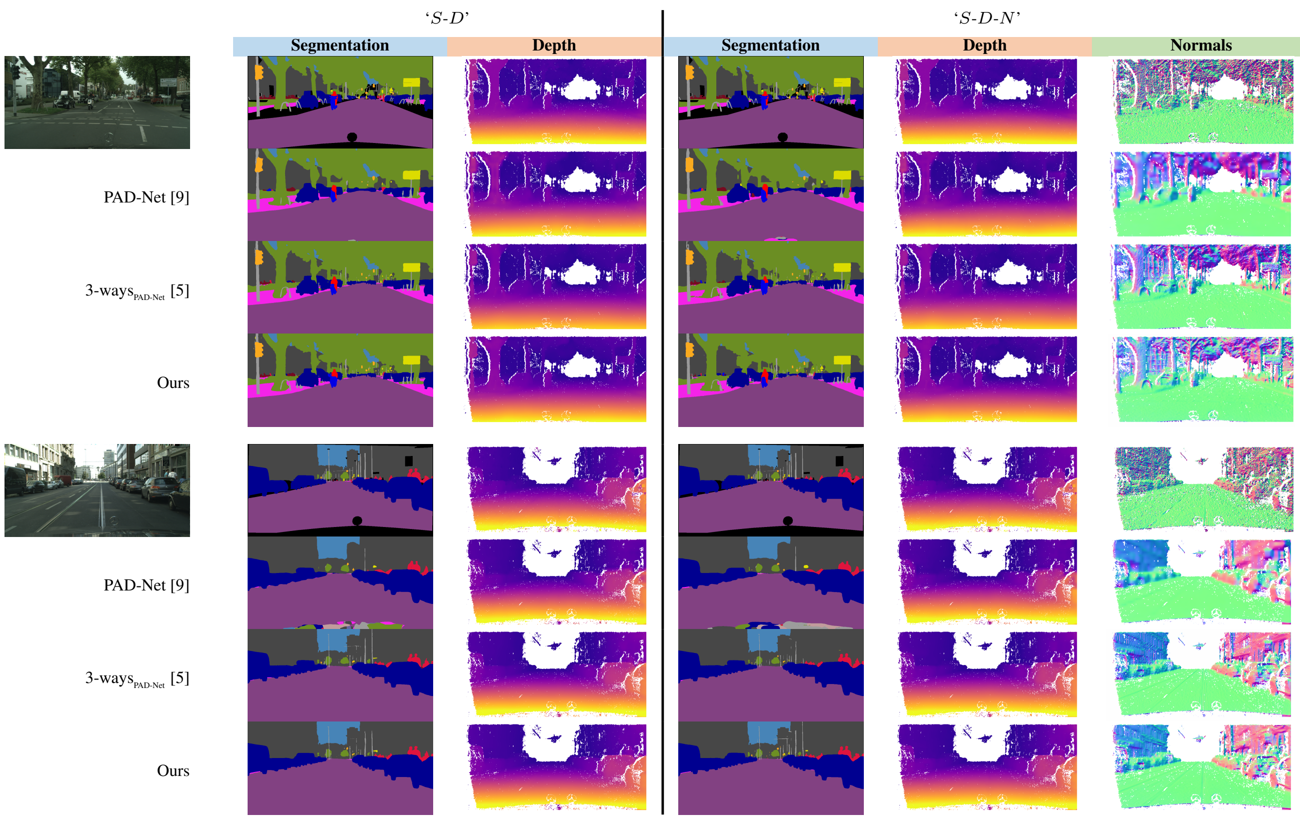

Figure 6: **Qualitative results on Cityscapes**. Overall, *Ours* produces better and sharper. Comparing visually against 3-ways$_{\text{PAD-Net}}$ is harder due to high scores.

References

[1] 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, 2016.

[2] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel Brostow. Digging into self-supervised monocular depth estimation. In ICCV, 2019.

[3] Vitor Guizilini, Jie Li, Rares Ambrus, and Adrien Gaidon. Geometric unsupervised domain adaptation for semantic segmentation. In ICCV, 2021.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In CVPR, 2016.

[5] Lukas Hoyer, Dengxin Dai, Yuhua Chen, Adrian Köring, Suman Saha, and Luc Van Gool. Three ways to improve semantic segmentation with self-supervised depth estimation. In CVPR, 2021.

[6] Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-task learning for dense prediction tasks: A survey. TPAMI, 2021.

[7] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Pérez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. In CVPR, 2019.

[8] Yufeng Wang, Yi-Hsuan Tsai, Wei-Chih Hung, Wenrui Ding, Shuo Liu, and Ming-Hsuan Yang. Semi-supervised multi-task learning for semantics and depth, 2021.

[9] Dan Xu, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing. In CVPR, 2018.

[10] Zhenheng Yang, Peng Wang, Wei Xu, Liang Zhao, and Ramakant Nevatia. Unsupervised learning of geometry with edge-aware depth-normal consistency, 2017.

[11] Zhenheng Yang, Peng Wang, Wei Xu, Liang Zhao, and Ramakant Nevatia. Unsupervised learning of geometry from videos with edge-aware depth-normal consistency. 2018.

[12] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. In CVPR, 2017.