Improvements to Image Reconstruction-Based Performance Prediction for Semantic Segmentation in Highly Automated Driving

—

Supplementary Material

Andreas Bär Daniel Kusuma Tim Fingscheidt
Technische Universität Braunschweig, Braunschweig, Germany
{andreas.baer, d.kusuma, t.fingscheidt}@tu-bs.de

1. Background

For better readability of the Supplement, we repeat a few equations from the main paper. If you are familiar with the main paper, you can skip this part.

First, we assume that encoder $\boldsymbol{E}^{\text{seg}}$ has E layers, while decoders $\boldsymbol{D}^{\text{seg}}$ and $\boldsymbol{D}^{\text{rec}}$ have D layers, with total number of layers $L = E + D$. Next, we introduce the decoder layer identifier $d \in \{1, ..., D-1\}$, which allows us to point to specific decoder layers. Note that we have the relation

$$\ell = E + d \tag{1}$$

between network layer ℓ and decoder layer d. For our *backward* parameter sharing, we first introduce the decoder layer identifiers $d_1, d_2 \in \{1, ..., D-1\}$, $d_1 \leq d_2$, indicating the first and last shared decoder layers. As we have $d_2 = D-1$, we define the number of parameters shared across decoders $\boldsymbol{D}^{\text{seg}}$ and $\boldsymbol{D}^{\text{rec}}$ as

$$\Delta d = D - d_1. \tag{2}$$

We will use Δd in Table 2 to refer to the amount of shared decoder layers. Further, we introduce the set of inter-decoder lateral connections $\mathcal{L}^{\text{IDLC}}$. In particular,

$$d' \in \mathcal{L}^{\text{IDLC}} \subset \{1, ..., D-1\} \tag{3}$$

holds. We will use d' to refer to the decoder layer index, where inter-decoder layer connections are incorporated. The entities of $\mathcal{L}^{\text{IDLC}}$ are then displayed in Table 2.

As task metrics, we employ the mean intersection-over-union defined as

$$mIoU_{\epsilon} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \frac{TP_{s,\epsilon}}{TP_{s,\epsilon} + FP_{s,\epsilon} + FN_{s,\epsilon}}, \tag{4}$$

with class-wise true positives $TP_{s,\epsilon} = \sum_{n \in \mathcal{N}} TP_{n,s,\epsilon}$, false positives $FP_{s,\epsilon} = \sum_{n \in \mathcal{N}} FP_{n,s,\epsilon}$, and false negatives $FN_{s,\epsilon} = \sum_{n \in \mathcal{N}} FN_{n,s,\epsilon}$. Further, ϵ indicates the average distortion strength and $n \in \mathcal{N}$ is an image index from set $\mathcal{N} = \{1, ..., |\mathcal{D}|\}$. With $mIoU_{n,\epsilon}$ being the image-specific mean intersection-over-union, we define its mean as

$$\overline{mIoU}_{\epsilon} = \frac{1}{|\mathcal{N}|} \sum_{n \in \mathcal{N}} mIoU_{n,\epsilon}. \tag{5}$$

We further introduce the image-specific peak signal-to-noise ratio $PSNR_{n,\epsilon}$ and its mean

$$\overline{PSNR}_{\epsilon} = \frac{1}{|\mathcal{N}|} \sum_{n \in \mathcal{N}} PSNR_{n,\epsilon}. \tag{6}$$

As performance prediction metrics, we use the Pearson correlation defined as

$$\rho = \frac{\sum_{n,\epsilon}(a_{n,\epsilon} - \mu_a)(b_{n,\epsilon} - \mu_b)}{\sqrt{\sum_{n,\epsilon}(a_{n,\epsilon} - \mu_a)^2} \sqrt{\sum_{n,\epsilon}(b_{n,\epsilon} - \mu_b)^2}}, \tag{7}$$

with $a_{n,\epsilon} = mIoU_{n,\epsilon}$, $b_{n,\epsilon} = PSNR_{n,\epsilon}$, $\mu_a = \frac{1}{|\mathcal{N}||\mathcal{E}|} \sum_{n,\epsilon} a_{n,\epsilon}$, $\mu_b = \frac{1}{|\mathcal{N}||\mathcal{E}|} \sum_{n,\epsilon} b_{n,\epsilon}$, and $\epsilon \in \mathcal{E}$, set of distortion strengths $\mathcal{E}$, and $\rho \in [-1, 1]$. Further, the mean absolute prediction error is defined as

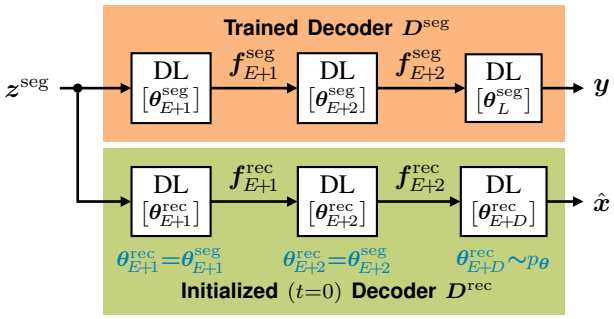
$$\Delta^{\text{M}} = \frac{1}{|\mathcal{N}||\mathcal{E}|} \sum_{n \in \mathcal{N}} \sum_{\epsilon \in \mathcal{E}} \left| \Delta_{n,\epsilon} \right|, \tag{8}$$

with $\Delta_{n,\epsilon} = \widehat{mIoU}_{n,\epsilon} - mIoU_{n,\epsilon}$, where $\widehat{mIoU}_{n,\epsilon}$ is an estimate of $mIoU_{n,\epsilon}$, followed by the root mean squared prediction error

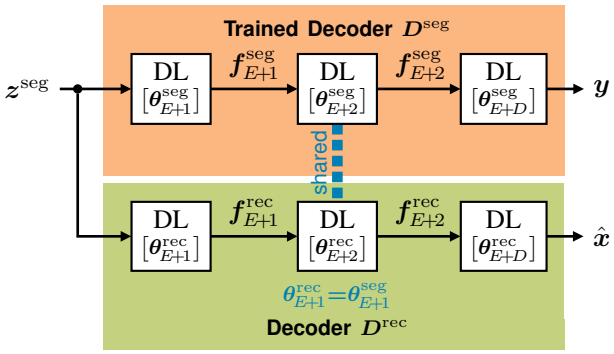
$$\Delta^{\text{R}} = \sqrt{\frac{1}{|\mathcal{N}||\mathcal{E}|} \sum_{n \in \mathcal{N}} \sum_{\epsilon \in \mathcal{E}} \left(\Delta_{n,\epsilon} \right)^2}. \tag{9}$$

2. More Details on Our Method

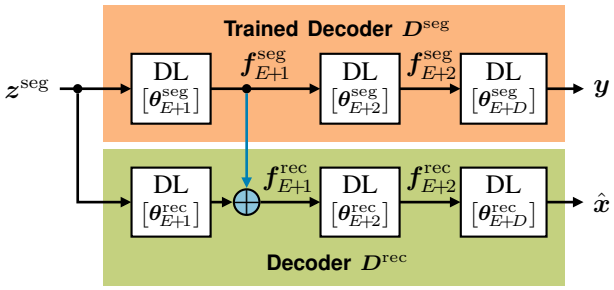
Figure 1 visualizes our proposed methods. We assume that decoders $\boldsymbol{D}^{\text{seg}}$ and $\boldsymbol{D}^{\text{rec}}$ have $D = 3$ layers. Further,



(a) Parameter Initialization



(b) Parameter Sharing



(c) Inter-Decoder Lateral Connections

Figure 1. **Our Proposals**. Decoders $\boldsymbol{D}^{\text{seg}}$ and $\boldsymbol{D}^{\text{rec}}$ with $D = 3$ decoder layers (DLs) and $\boldsymbol{z}^{\text{seg}} = \boldsymbol{E}^{\text{seg}}(\boldsymbol{x})$, with encoder $\boldsymbol{E}^{\text{seg}}$ having E layers. Note that each DL may be of arbitrary layer type having its individual set of parameters $\boldsymbol{\theta}$. Both $\boldsymbol{D}^{\text{seg}}$ and $\boldsymbol{D}^{\text{rec}}$ take $\boldsymbol{z}^{\text{seg}}$ as input and produce $\boldsymbol{y} = \boldsymbol{D}^{\text{seg}}(\boldsymbol{z}^{\text{seg}}; \boldsymbol{\theta}^{\text{seg}}_{E+1:E+D})$ and $\hat{\boldsymbol{x}} = \boldsymbol{D}^{\text{rec}}(\boldsymbol{z}^{\text{rec}}; \boldsymbol{\theta}^{\text{rec}}_{E+1:E+D})$, respectively. All proposed adjustments are marked in blue. Best viewed in color. (a) First and second DL ($\ell = E+1, E+2$) of $\boldsymbol{D}^{\text{rec}}$ are initialized, *i.e.*, $t = 0$, with respective DL parameters of $\boldsymbol{D}^{\text{seg}}$, while last DL ($\ell = E + D$) of $\boldsymbol{D}^{\text{rec}}$ is randomly initialized following the distribution $p_{\boldsymbol{\theta}}$. (b) Parameters of second DL ($\ell = E + 2, d_1 = 2$) are shared across $\boldsymbol{D}^{\text{rec}}$ and $\boldsymbol{D}^{\text{seg}}$. This corresponds to $\Delta d = 3 - 2 = 1$ (2). (c) We employ inter-decoder lateral connections after the first DL ($\ell' = E + 1, d' = 1$) resulting in $\boldsymbol{f}^{\text{rec}}_{E+1} \leftarrow \boldsymbol{f}^{\text{seg}}_{E+1} + \boldsymbol{f}^{\text{rec}}_{E+1}$. In this case, the set of inter-decoder lateral connections has one entry, i.e., $\mathcal{L}^{\text{IDLC}} = \{d'\} = \{1\}$ (3).

we assume $\boldsymbol{z}^{\text{seg}} = \boldsymbol{E}^{\text{seg}}(\boldsymbol{x})$, where encoder $\boldsymbol{E}^{\text{seg}}$ has E layers. For visualization purposes we neglect any encoder-decoder lateral connections between encoder $\boldsymbol{E}^{\text{seg}}$ and decoders $\boldsymbol{D}^{\text{seg}}$ and $\boldsymbol{D}^{\text{rec}}$.

In Figure 1a, we provide a visualization for our parameter initialization. In particular, the parameters of layers $\ell = E + 1$ and $\ell = E + 2$ corresponding to the first ($d = 1$) and second ($d = 2$) decoder layer (DL) of $\boldsymbol{D}^{\text{rec}}$ are initialized using the segmentation decoder weights $\boldsymbol{\theta}^{\text{seg}}_{E+d}$ of the segmentation decoder $\boldsymbol{D}^{\text{seg}}$, respectively. As the last layer of each decoder has its individual output space, we randomly initialize $\boldsymbol{\theta}^{\text{rec}}_{E+D}$ following some standard distribution p_θ.

In Figure 1b, we give an overview of our backward parameter sharing scheme. Note that the last shared DL $d_2 = D-1 = 2$ is fixed while the first shared DL d_1 can be freely chosen. In the example, we share the parameters of layer $\ell = E + 2$ which is the second ($d = 2$) DL. This corresponds to setting $d_1 = 2$. As a result, we have $\Delta d = 1$ (2) shared DLs in total. Consequently, if we would like to proceed with our backward parameter sharing scheme and set $d_1 = 1$, we end up with $\Delta d = 2$ shared DLs, where layers $\ell = E + 1$ and $\ell = E + 2$ having the decoder layer identifiers $d = 1$ and $d = 2$, respectively, are shared.

Lastly, in Figure 1c we elaborate on inter-decoder lateral connections (IDLCs). In this example, we employ an inter-decoder lateral connection after network layer $\ell' = E + 1$ corresponding to decoder layer $d' = 1$. As a result, we obtain $\boldsymbol{f}^{\text{rec}}_{E+1} \leftarrow \boldsymbol{f}^{\text{seg}}_{E+1} + \boldsymbol{f}^{\text{rec}}_{E+1}$, with the set of inter-decoder lateral connections being $\mathcal{L}^{\text{IDLC}} = \{d'\} = \{1\}$. If we would like to have additional IDLCs, set $\mathcal{L}^{\text{IDLC}}$ is extended by the additional decoder layer identifiers d' that are used, e.g., we have $\mathcal{L}^{\text{IDLC}} = \{1, 2\}$ for the case where IDLCs are employed after the first ($d' = 1$) and second ($d' = 2$) DL.

3. Detailed Experimental Setup

In the following, we extend the experimental setup from the main paper and provide a few more details. In particular, we include decoder layer identifiers d for each decoder type used in our experiments. The decoder layer identifiers are then used as d_1 (2) in our parameter sharing experiments and d' (3) in our inter-decoder lateral connection experiments. Further, all experiments were performed using `PyTorch 1.10.2`, `TorchVision 0.11.3`, `CUDA 10.2`, and a single `NVIDIA GTX 1080Ti`. Code is available at https://github.com/ifnspaml/PerfPredRecV2.

Network architectures: The `SwiftNet (SN)`-based $\boldsymbol{D}^{\text{seg}}$ uses spatial pyramid pooling ($d = 1$) followed by three upsampling layers ($d = 2, 3, 4$), with each having a convolved lateral connection ($d = 5, 6, 7$) from the (pre-activated) encoder outputs $\boldsymbol{f}^{\text{seg}}_{\ell}$, $\ell \in \mathcal{L}^{\text{EDLC}} = \{\ell_{\text{EB1}}, \ell_{\text{EB2}}, \ell_{\text{EB3}}\}$, corresponding to the outputs of the 1st, 2nd, and 3rd `ResNet (RN)`, `Swin (SW)`, or `ConvNeXt (CN)` block. Note that the exact layer indices of $\ell_{\text{EB1}}, \ell_{\text{EB2}}, \ell_{\text{EB3}}$ vary across `ResNet18 (RN18)`, `ResNet50 (RN50)`, `Swin-Tiny (SW-T)`, and

ConvNeXt-Tiny (CN-T). We adapt the SwiftNet decoder from PerfPredRec[1], the ResNet encoders from TorchVision[2], the Swin-Tiny encoder from Transformers[3] and the ConvNeXt-Tiny encoder from ConvNeXt[4]. For completeness, we have $D = 8$ for the SN-based $\boldsymbol{D}^{\text{seg}}$.

On the other hand, the DeepLabv3+ (DL)-based $\boldsymbol{D}^{\text{seg}}$ employs depthwise separable dilated (also called "atrous") spatial pyramid pooling ($d = 2$) followed by a concatenation to the convolved encoder feature representation at ℓ_{EB1} ($d = 1$) and further upsamplings and convolutions ($d = 3, 4$). In addition, DeepLabv3+ varies the encoder's convolution dilation rates with respect to the preset encoder's output stride (we use output stride 16 for RN18, RN50, CN-T and output stride 32 for SW-T). Further, we adapt the DeepLabv3+ decoder from MMSegmentation[5] and follow some design recommendations. In particular, we set the number of feature maps of each dilated spatial pyramid pooling branch to 128, 256, 192, 192 and of the convolved encoder feature representation to 12, 48, 18, 18 for RN18, RN50, SW-T, CN-T, respectively. Note that we do not adjust feature maps in the SN-based decoder. For completeness, we have $D = 5$ for the DL-based $\boldsymbol{D}^{\text{seg}}$.

Lastly, we limit our Monodepth2 (MD) experiments to an RN18 encoder. The MD-based $\boldsymbol{D}^{\text{seg}}$ uses ten consecutive convolutions ($d = 1...10$) decreasing gradually the number of feature maps. Before every second convolution ($d = 2, 4, 6, 8, 10$) an upsampling is performed and the resulting upsampled feature map is concatenated with a lateral connection ($d = 2, 4, 6, 8$). The lateral connections correspond to the outputs of the RN18 stem block as well as the outputs of the 1st, 2nd, and 3rd RN block. Note that different to the SN- and DL-based decoders, the lateral connections are not convolved beforehand. Further, we adapt the Monodepth2 decoder from SGDepth[6]. For completeness, we have $D = 11$ for the MD-based $\boldsymbol{D}^{\text{seg}}$.

We refer to [10], [2], and [6] for further details about SwiftNet, DeepLabv3+, and the adapted Monodepth2, respectively. Finally, the image reconstruction decoder $\boldsymbol{D}^{\text{rec}}$ follows the architecture of the employed $\boldsymbol{D}^{\text{seg}}$, with the adaptation of layer L as described in the main paper.

Training details: We train the SwiftNet-based and Monodepth2-based models by following the SwiftNet training protocol [10] and train for 200 epochs using the Adam [4] (RN18, RN50) or AdamW [9] (SW-T, CN-T) optimizer with learning rate $4 \cdot 10^{-4}$ and weight decay 10^{-4}. A cosine annealing schedule is applied with minimum learning rate 10^{-6}. For all pretrained model we adjust the training parameters to learning rate 10^{-4}, weight decay $0.25 \cdot 10^{-4}$, and minimum learning rate 10^{-7}. Further, during training we augment the images by random horizontal flipping, random resizing in the range [0.5, 2.0], and random cropping to 768×768. We set the batch size to 12, 7, 4, or 4 for models with RN18, RN50, SW-T, or CN-T encoder, respectively. Note that we use the AdamW optimizer for SW-T- and CN-T-based models as we found out it yields significantly better performance than Adam.

The DeepLabv3+-based models are trained by combining the DeepLabv3+ protocol from [2] with parts of the SwiftNet training protocol as well as MMsegmentation[5] training protocols. In particular, we train for 200 epochs using the SGD optimizer with momentum of 0.9 [11] (RN18, RN50) or the AdamW [9] (SW-T, CN-T) optimizer. Further, we follow a polynomial learning rate schedule and set the starting learning rate to either 10^{-2} (SGD with momentum) or 10^{-4} (AdamW) and reduce the learning rate down to 10^{-4} or 10^{-5}, respectively. During training we augment the images similar to the SwiftNet protocol [10] described above. We set the batch size to 12, 5, 4, or 4 for models with RN18, RN50, SW-T, or CN-T encoder, respectively.

4. Additional Results

Baseline performance: We report performance of baselines for models with RN50-, SW-T-, or CN-T-based encoders and SN- or DL-based decoders on clean ($\epsilon = 0$) validation and test datasets in Table 1.

Best combination: We report results on the best configuration for each of our proposed methods, i.e., parameter initialization, parameter sharing, and inter-decoder lateral connections, as well as a combination of all our proposed methods on mixed clean/distorted validation datasets in Table 2. The results are reported for models with RN50-, SW-T-, or CN-T-based encoders and SN- or DL-based decoders, i.e., six model architectures in total. Further, the best model out of all configurations for each out of the six possible model architectures is highlighted in gray.

The results mostly align with the observations in the main paper. For three model architectures, i.e., SN+RN50, SN+CN-T, and DL+SW-T, the combination of our proposed methods leads to the best results. For two model architectures, i.e, SN+SW-T and DL+CN-T, using inter-decoder lateral connections alone leads to the best results. For one model architecture, i.e., DL+RN50, using our parameter sharing scheme alone leads to the best results. Moreover, with the exception of two model architectures evaluated on two different datasets, i.e., DL+CN-T on $\mathcal{D}_{\text{val}}^{\text{CS}}$ and SN+CN-T on $\mathcal{D}_{\text{val}}^{\text{KIT}}$, using parameter sharing alone, inter-decoder lateral connections alone, or the combination of all methods is better than using parameter initialization alone

[1] https://github.com/ifnspaml/PerfPredRec
[2] https://github.com/pytorch/vision/tree/main/torchvision/models
[3] https://github.com/huggingface/transformers
[4] https://github.com/facebookresearch/ConvNeXt
[5] https://github.com/open-mmlab/mmsegmentation
[6] https://github.com/ifnspaml/SGDepth

Table 1. **Baseline performance on clean ($\epsilon = 0$) validation and test datasets**. Metrics $mIoU_{\epsilon=0}$ [%] (4), $\overline{mIoU}_{\epsilon=0}$ [%] (5), and $\overline{PSNR}_{\epsilon=0}$ [dB] (6) of the SN/DL-based $\boldsymbol{D}^{\mathrm{seg}}$ and $\boldsymbol{D}^{\mathrm{rec}}$ and of the RN50/SW-T/CN-T-based $\boldsymbol{E}^{\mathrm{seg}}$. All models trained on $\mathcal{D}^{\mathrm{CS}}_{\mathrm{train}}$.

$\boldsymbol{D}^{\mathrm{seg}}$, $\boldsymbol{D}^{\mathrm{rec}}$	$\boldsymbol{E}^{\mathrm{seg}}$	$mIoU_{\epsilon=0}$ $\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	$mIoU_{\epsilon=0}$ $\mathcal{D}^{\mathrm{CS}}_{\mathrm{test}}$	$mIoU_{\epsilon=0}$ $\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	$mIoU_{\epsilon=0}$ $\mathcal{D}^{\mathrm{KIT}}_{\mathrm{test}}$	$\overline{mIoU}_{\epsilon=0}$ $\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	$\overline{mIoU}_{\epsilon=0}$ $\mathcal{D}^{\mathrm{CS}}_{\mathrm{test}}$	$\overline{mIoU}_{\epsilon=0}$ $\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	$\overline{mIoU}_{\epsilon=0}$ $\mathcal{D}^{\mathrm{KIT}}_{\mathrm{test}}$	$\overline{PSNR}_{\epsilon=0}$ $\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	$\overline{PSNR}_{\epsilon=0}$ $\mathcal{D}^{\mathrm{CS}}_{\mathrm{test}}$	$\overline{PSNR}_{\epsilon=0}$ $\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	$\overline{PSNR}_{\epsilon=0}$ $\mathcal{D}^{\mathrm{KIT}}_{\mathrm{test}}$
SN	RN50	65.18	76.08	38.22	36.52	50.49	63.24	32.98	31.33	31.99	31.39	21.54	21.85
	SW-T	68.98	76.04	53.37	46.84	55.95	63.31	40.27	39.00	21.83	20.09	16.27	16.49
	CN-T	74.76	79.43	56.49	58.46	57.08	66.43	45.25	47.08	31.37	30.83	20.39	21.04
DL	RN50	64.18	77.47	42.46	40.13	51.99	65.03	36.68	34.89	32.56	31.67	22.03	22.16
	SW-T	64.65	75.53	49.73	47.56	54.91	63.17	43.90	42.00	31.40	30.87	20.64	21.13
	CN-T	69.02	79.02	54.23	58.10	55.71	66.41	46.42	46.70	31.49	30.71	20.47	20.96

Table 2. **Best combination (ours) on mixed clean/distorted validation datasets**. Metrics ρ (7), Δ^{M} (8), and Δ^{R} (9) of the SN/DL-based $\boldsymbol{D}^{\mathrm{seg}}$ and $\boldsymbol{D}^{\mathrm{rec}}$ and of the RN50/SW-T/CN-T-based $\boldsymbol{E}^{\mathrm{seg}}$. 'Init' = 'initialization mode' (random weights (r) or segmentation decoder weights (s)), 'Δd' = 'amount of shared decoder layers' (2) & '$\mathcal{L}^{\mathrm{IDLC}}$' = 'set of inter-decoder later connections' (3). A dash (-) indicates that this feature is disabled. Best results in boldface, second best underlined.

$\boldsymbol{D}^{\mathrm{seg}}$, $\boldsymbol{D}^{\mathrm{rec}}$	$\boldsymbol{E}^{\mathrm{seg}}$	Init	Δd	$\mathcal{L}^{\mathrm{IDLC}}$	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$ Δ^{M}	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$ Δ^{R}	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$ Δ^{M}	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$ Δ^{R}
SN	RN50	r	-	-	8.46	11.62	7.30	9.36
SN	RN50	r	3	-	7.41	10.51	**5.97**	**7.83**
SN	RN50	r	-	4	8.37	11.50	6.80	8.72
SN	RN50	r	3	4	**6.93**	**9.97**	6.25	8.09
SN	SW-T	s	-	-	13.76	17.03	11.21	14.14
SN	SW-T	r	3	-	13.72	16.84	10.58	13.29
SN	SW-T	r	-	2,3,4	**11.96**	**14.64**	**9.59**	**11.84**
SN	SW-T	s	3	2,3,4	12.71	16.08	10.91	13.94
SN	CN-T	r	-		15.67	19.27	13.23	16.46
SN	CN-T	r	4	-	14.34	18.30	10.35	13.51
SN	CN-T	r	-	2,3,4	13.00	**16.18**	13.37	16.50
SN	CN-T	r	4	2,3,4	**12.56**	**16.18**	**9.15**	**11.85**
DL	RN50	r	-	-	10.51	13.31	8.77	10.61
DL	RN50	r	1	-	**9.54**	12.44	**7.84**	**9.60**
DL	RN50	r	-	3,4	9.61	**12.27**	8.58	10.40
DL	RN50	r	1	3,4	9.76	12.48	8.37	10.26
DL	SW-T	r	-	-	13.97	16.82	12.94	15.25
DL	SW-T	r	2	-	12.27	15.30	11.64	14.10
DL	SW-T	r	-	4	10.98	13.78	10.03	12.29
DL	SW-T	r	2	4	**10.53**	**13.29**	**9.41**	**11.50**
DL	CN-T	r	-	-	14.92	18.31	13.60	16.83
DL	CN-T	r	3	-	13.97	17.09	**11.77**	**14.61**
DL	CN-T	r	-	4	**13.09**	**16.24**	12.12	14.71
DL	CN-T	r	3	4	17.86	21.39	12.78	15.90

on both $\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$ and $\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$. Note that the baseline approach in [1] simply uses random parameter initialization ("r") and does not incorporate any of our proposed approaches. Assuming that the difference between random parameter initialization ("r") and segmentation weights initialization ("s") is not large (as was also shown in the main paper), we can conclude that even with this large variety of model architectures our proposed methods are better than the baseline approach [1].

Table 3. **State of the art comparison on mixed clean/distorted test datasets**. Metrics ρ (7), Δ^{M} (8), and Δ^{R} (9) for state-of-the-art methods [1,5,6] and ours. Ours and [1] use $\mathcal{D}^{\mathrm{CS}}_{\mathrm{train}}$ for training, while [5, 6] also use video data $\mathcal{D}^{\mathrm{CS}}_{\mathrm{vid}}$, $\mathcal{D}^{\mathrm{KIT}}_{\mathrm{vid}}$. We extend Table 7 from the main paper with additional encoders (Enc.) and decoders (Dec.) for benchmarking reasons. 'Cal.' = 'regression calibration' & '*' = 'slightly modified $\mathcal{D}^{\mathrm{CS}}_{\mathrm{test}}$'.

Eval	Video	Cal.	Method	Dec.	Enc.	ρ	Δ^{M}	Δ^{R}
$\mathcal{D}^{\mathrm{CS}}_{\mathrm{test}}$	-	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	Ours	SN	RN50	**0.91**	**8.88**	**12.13**
	-	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	Ours	SN	SW-T	0.77	13.27	15.94
	-	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	Ours	SN	CN-T	0.69	16.27	19.94
	-	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	Ours	DL	RN50	0.86	14.11	16.96
	-	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	Ours	DL	SW-T	0.87	12.53	15.10
	-	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	Ours	DL	CN-T	0.67	16.60	19.50
	-	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	Ours	SN	RN18	**0.92**	**9.47**	**12.85**
	-	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	[1]	SN	RN18	0.90	10.12	13.18
	-	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	Ours	MD	RN18	**0.88**	**9.14**	**12.25**
	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{vid}}$	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	[6]	MD	RN18	0.58*	12.19*	15.71*
	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{vid}}$	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{val}}$	[6]	MD	RN18	0.43*	13.38*	16.12*
$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{test}}$	-	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	Ours	SN	RN50	**0.76**	**6.67**	**8.42**
	-	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	Ours	SN	SW-T	0.54	10.11	12.59
	-	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	Ours	SN	CN-T	0.63	11.44	14.57
	-	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	Ours	DL	RN50	0.68	8.94	11.18
	-	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	Ours	DL	SW-T	0.65	10.81	13.98
	-	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	Ours	DL	CN-T	0.37	13.15	16.34
	-	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	Ours	SN	RN18	**0.74**	**7.80**	**10.10**
	-	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	[1]	SN	RN18	0.73	8.00	10.24
	-	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	Ours	MD	RN18	0.70	7.92	9.79
	$\mathcal{D}^{\mathrm{CS}}_{\mathrm{vid}}$	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	[6]	MD	RN18	0.54	7.81	9.79
	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{vid}}$	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	[6]	MD	RN18	0.77	6.01	7.70
	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{vid}}$	$\mathcal{D}^{\mathrm{KIT}}_{\mathrm{val}}$	[5,6]	MD	RN18	**0.86**	**4.45**	**6.16**

More results on test data: We report results for models with RN50-, SW-T-, or CN-T-based encoders and SN-

or DL-based decoders on our mixed clean/distorted test datasets in Table 3. In particular, we use the best models in Table 2 (highlighted in gray) and just perform an additional evaluation on the respective test dataset to establish further benchmark results. Note that Table 3 can be seen as an extended version of Table 7 from the main paper. First of all, note that that all three encoders are comparable with regard to number of parameters and number of floating point operations [3, 7, 8], however, all three being more complex than RN18 [3]. We observe that the SN-based decoder with a RN50-based encoder shows the best results for image-only performance prediction with $\Delta^{\mathrm{M}} = 8.88$ and $\Delta^{\mathrm{R}} = 12.13$ on $\mathcal{D}_{\mathrm{test}}^{\mathrm{CS}}$ and with $\Delta^{\mathrm{M}} = 6.67$ and $\Delta^{\mathrm{R}} = 8.42$ on $\mathcal{D}_{\mathrm{test}}^{\mathrm{KIT}}$.

References

[1] Andreas Bär, Marvin Klingner, Jonas Löhdefink, Fabian Hüger, Peter Schlicht, and Tim Fingscheidt. Performance Prediction for Semantic Segmentation by a Self-Supervised Image Reconstruction Decoder. In *Proc. of CVPR - Workshops*, pages 4399–4408, New Orleans, LA, USA, June 2022. 4

[2] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-Decoder With Atrous Separable Convolution for Semantic Image Segmentation. In *Proc. of ECCV*, pages 801–818, Munich, Germany, Sept. 2018. 3

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proc. of CVPR*, pages 770–778, Las Vegas, NV, USA, June 2016. 5

[4] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *Proc. of ICLR*, pages 1–15, San Diego, CA, USA, May 2015. 3

[5] Marvin Klingner, Andreas Bär, Marcel Mross, and Tim Fingscheidt. Improving Online Performance Prediction for Semantic Segmentation. In *Proc. of CVPR - Workshops*, pages 1–11, virtual, June 2021. 4

[6] Marvin Klingner and Tim Fingscheidt. Online Performance Prediction of Perception DNNs by Multi-Task Learning with Depth Estimation. *IEEE Transactions on Intelligent Transportation Systems (T-ITS)*, 22(7):4670–4683, July 2021. 3, 4

[7] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *Proc. of ICCV*, pages 10012–10022, virtual, Oct. 2021. 5

[8] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *Proc. of CVPR*, pages 11976–11986, New Orleans, LA, USA, June 2022. 5

[9] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *Proc. of ICLR*, pages 1–18, New Orleans, LA, USA, May 2019. 3

[10] Marin Oršić, Ivan Krešo, Petra Bevandić, and Siniša Šegvić. In Defense of Pre-Trained ImageNet Architectures for Real-Time Semantic Segmentation of Road-Driving Images. In *Proc. of CVPR*, pages 12607–12616, Long Beach, CA, USA, June 2019. 3

[11] Boris T. Polyak. Some Methods of Speeding Up the Convergence of Iteration Methods. *USSR Computational Mathematicsand Mathematical Physics*, 4(5):1–17, Nov. 1964. 3