Appendices

As part of the supplementary materials for this paper, we present our hyper-parameters, provide details on how we implemented and generated results from other works and also show more results as an extension to the ones shown in the main paper.

A Results on CamVid

As shown in Section 4.3 of the paper, we trained various backbones on the Cityscapes dataset. As Cityscapes shares similar classes with the CamVid benchmark [2], we used our models trained on the Cityscapes dataset to generate predictions on CamVid and report domain adaptation mIoU performance on CamVid *test*. This set contains 233 labeled images from two videos in a driving scene. Each scene is annotated with 11 classes. As Cityscapes models report dense predictions from one of 19 semantic classes, we club these predictions from our Cityscapes-trained model using a scheme provided in Table A.1.

CamVid Class	Cityscapes Classes
Building	Building, Wall
Column-Pole	Pole
Road	Road
Sidewalk	Sidewalk
Tree	Vegetation
Sign-Symbol	Traffic Light, Traffic Sign
Sky	Sky, Terrain
Fence	Fence
Car	Car, Truck, Bus, Train, Bicycle
Pedestrian	Person
Bicyclist	Rider

Table A.1: To perform inference on CamVid using a model trained on Cityscapes, we first convert the Cityscapes labels to CamVid labels using this conversion scheme.

We report mIoU scores on CamVid for various models trained on Cityscapes. These include HRNetv2-w18, DeepLab-v3+ and HRNet18-OCR networks trained using the various schemes discussed in our paper. As reported in Table A.2, the HS3 scheme shows a significant improvement in performance over DS with no added inference cost. We show our results on HS3-Fuse in Table A.3. Using our technique of scaling all OCR channels within the HS3-Fuse framework, we measure performance at various operating points for HS3-Fuse and compare it to vanilla OCR. As observed, HS3-Fuse scheme shows a significant improvement in performance over vanilla OCR at the same number of GMacs. We would like to emphasize that none of the above networks have been fine-tuned on the CamVid training set, and the predictions are used directly after training on Cityscapes.

B Implementation details

In this section, we discuss our training setup in detail for every experiment. Depending on the experiment, we use 1 to 8 NVIDIA Tesla-V100 GPUs for training.

Network	DS	HS3(**Ours**)	CamVid mIoU
WRN38-DeepLab-v3+	✓		68.8
WRN38-DeepLab-v3+		✓	**69.4**
HRNetv2-w18	✓		64.2
HRNetv2-w18		✓	**65.0**

Table A.2: Trained on Cityscapes, test on CamVid *test*: Adaptation to CamVid after training on Cityscapes with the proposed HS3 method improves performance compared to various baselines and also outperforms the Deep Supervision (DS) approach with no added inference cost.

Method	Backbone	CamVid mIoU	GMacs
HRNetv2-w18-OCR	HRNetv2-w18	65.9	154
HS3-Fuse	HRNetv2-w18	**67.2**	224
HS3-Fuse (Lighter)	HRNetv2-w18	66.9	154

Table A.3: Trained on Cityscapes, test on CamVid *test*: Adaptation to CamVid after training on Cityscapes with the proposed HS3-Fuse method improves performance compared to the vanilla approach after scaling OCR channels to match the computational cost.

B.1 NYUD-v2 dataset

HRNet Models: We use the HRNet and OCR open-source implementation[1] to produce baseline scores. For all training runs, we augment the input images by random flips(p=0.5), random scaling(ratio in $\{1, 1.2, 1.5\}$) and cropping back to the original dimension of 480x640 pixels. We then train both HRNet and HRNet-OCR backbones with stochastic gradient-descent for 484 epochs with a batch size of 12, setting the initial learning rate to 0.02 and a poly learning rate schedule with exponent 0.9. For DS and HS3 experiments, we apply the supervision at stage 2 and stage 3 of the HRNet backbone(which become stage 1 and stage 2 of our supervision algorithm respectively). The final prediction remains at stage 4 for vanilla HRNet. For HRNet-OCR, the stage 4 prediction is auxiliary, as the final prediction is produced at the output of OCR block. The training loss is obtained from Equation 1 where the loss weights for all intermediate stages are set at $\gamma = 0.1$. For HS3 experiments, we find the proposed number of intermediate stage-wise clusters at $\theta = 76°$ in Figure 2. The spectral clustering algorithm is used to obtain class grouping.

SA-Gates using ResNet-101: For reproducing SA-Gates scores, we use their repository[2] and hyper-parameters to train the baseline ResNet-101 model with DeepLab-V3+. For the HS3-Fuse experiment, we apply the supervision at block 2 and block 3 of the SA-ResNet encoder [□](which become stage 1 and stage 2 of our supervision algorithm respectively). The auxiliary prediction remains at block 4, and the final prediction is produced at the output of OCR block. The training loss is obtained from Equation 1 where the loss weights for all intermediate stages are set at $\gamma = 0.2$, and the auxiliary stage remains at $\gamma = 0.4$. For HS3 experiments, we find the proposed number of intermediate stage-wise clusters at $K_1 = 9$ and $K_2 = 16$. The spectral clustering algorithm is used to obtain class grouping.

[1] https://github.com/HRNet/HRNet-Semantic-Segmentation
[2] SA-Gates Github

B.2 Cityscapes dataset

HRNet-48 with OCR and/or Hierarchical multi-scale attention: To reproduce the numbers for both Hierarchical Multi-Scale Attention [22] and HRNet-48 with OCR [27] models, we use the code made available publicly by the HMS authors[3]. We train the networks using 8 GPUs instead of 16(mentioned in the original paper). Hence, we train with a smaller batch-size of 8. Apart from the batch-size, we use the same hyper-parameters and augmentation schemes used by the HMS paper. We start with a mapillary vistas pre-trained checkpoint and use the refined-coarse annotations provided in the repository for semi-supervised KD training. For DS and HS3 experiments, we apply the supervision at stage 2 and stage 3 of the HRNet backbone(which become stage 1 and stage 2 of our supervision algorithm respectively). The final prediction remains at stage 4 for vanilla HRNet. For HRNet-OCR, the stage 4 prediction is auxiliary, as the final prediction is produced at the output of OCR block. The training loss is obtained from Equation 1 where the loss weights for all intermediate stages are set at $\gamma = 0.1$. For HS3 experiments, we find the proposed number of intermediate stage-wise clusters at $K_1 = 4$ and $K_2 = 10$. The spectral clustering algorithm is used to obtain class grouping. In Section 4.3 of the main paper, we displayed results on both Cityscapes *val* and *test* set. We also showed visual results from our state-of-the-art predictions in in Figure 5, and compare with some recent works. We compare with two recent works, SegFix [28] and the baseline Hierarchical multi-scale attention(HMS) [22]. The authors in [22] made their state-of-the-art HRNet-OCR-HMS model open-source. Hence, we run inference on this model to obtain predictions. The authors of [28] directly shared the predictions from their state-of-the-art HRNet-OCR-SegFix model [4]. We compare both these results with the ones we obtain from our HS3-Fuse candidate.

DeepLab-v3+ with WiderResNet38: For DeepLab-v3+ experiments, we use the same codebase as above made available publicly by the HMS authors. We train the networks using 8 GPUs instead of 16(mentioned in the original paper). Hence, we train with a smaller batch-size of 8. Apart from batch-size, we use the default hyper-parameters set by the repository. We start with a mapillary vistas pre-trained checkpoint and use the refined-coarse annotations provided in the repository for semi-supervised KD training. For DS and HS3 experiments, we apply the supervision at m2 and m3 of the WRN38 backbone(which become stage 1 and stage 2 of our supervision algorithm respectively). The final prediction remains at m7. The training loss is obtained from Equation 1 where the loss weights for all intermediate stages are set at $\gamma = 0.1$. For HS3 experiments, we find the proposed number of intermediate stage-wise clusters at $K_1 = 8$ and $K_2 = 12$. The spectral clustering algorithm is used to obtain class grouping.

B.3 Hyper-parameter search

In Section 3.1, we described how we pick the clustering approach and number of clusters for HS3. To find a suitable set of intermediate γ for the loss function in Equation 1, we performed a hyper-parameter search on a HRNetv2-w18 baseline trained with Deep Supervision(DS) to obtain the intermediate loss weights. The search is on the analysis split and not test-set. This provided the strongest baseline to compare our approach(HS3). For the HS3 results, we use the same weights as obtained for DS ensuring fair comparison. We used these weights for all experiments except for SA-Gates. As the SA-Gates-ResNet baseline already had an

[3] https://github.com/NVIDIA/semantic-segmentation
[4] https://github.com/openseg-group/openseg.pytorch

intermediate prediction and some level of deep supervision(with a different gamma in their code), we re-did the hyper-parameter search for this backbone.