Supplementary Material

Specialize and Fuse: Pyramidal Output Representation for Semantic Segmentation

A. Visualization of the intermediate results

We show the intermediate results of our approach and illustrate how they are fused in Fig. A.

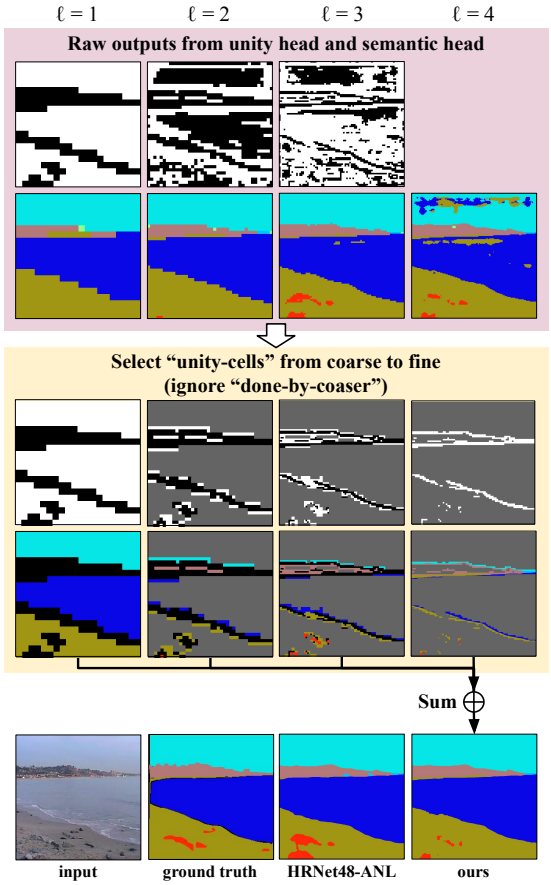

Figure A: An example from the ADE20K validation set showing the intermediate results. **Top**: The raw semantic pyramid outputs and raw unity pyramid outputs. **Middle**: From the coarsest output to the finest output, we select semantic labels only from "unity-cells" but ignore the "done-by-coarser" regions (colored in grey). **Bottom**: Input image, ground truth, HRNet48-ANL prediction, and our final fused prediction.

B. Pyramidal ground truth

Here, we detail how to generate the pyramidal ground truth from standard per-pixel semantic labels when there are "don't care" annotations in the original dataset. For a cell covering $s_\ell \times s_\ell$ pixels, we consider 3 cases for different treatments.

- **All pixels share a semantic class.** In this case, the ground-truth label for the unity pyramid is "unity-cell" (*i.e.*, positive), and the label for the semantic pyramid is the shared class.
- **More than one semantic classes appear.** In this case, the label for the unity pyramid is "mix-cell" (*i.e.*, negative), and the label for the semantic pyramid is "don't care" because it is ambiguous to use one semantic label to represent all underlying pixels of different semantic classes and thus a finer semantic prediction should be referred to.
- **One semantic class and "don't care" appear.** Both the unity and the semantic ground truth are defined as "don't care" as it is ambiguous to determine whether the cell is "unity-cell" or "mix-cell". During the training re-labeling procedure, such a cell is never regarded as true positive, and thus its children cells in the next finer level would never be re-labeled as "done by coarser".

C. Detailed model settings in ablation study

We reorganize the ablation experiment results presented in the main paper into a unified view in Table A and illustrate their network architectures in Fig. B. We label each experiment with an ID (A∼K) and describe the detailed architecture setting below. We call the layers projecting latent dimension D_s to number of classes C as *final projection layer*, which is implemented as Conv1×1 → BN → ReLU → Conv1×1.

- *A* (40.42 %) directly appends the *final projection layer* to $X^{(4)}$.
- *B* (42.02 %) refines $X^{(4)}$ with ANL's APNB block.

ID	mIoU (%)	Contextual module	Output format	Training procedure
A	40.42	-	single (standard)	-
B	42.02	ANL	single (standard)	-
C	42.00	ours	single (standard)	-
D	40.41	-	pyramidal (ours) {4,8,16,32}	naive
E	41.54	-	pyramidal (ours) {4,8,16,32}	simple fix
F	42.07	-	pyramidal (ours) {4,8,16,32}	our final
G	43.07	ANL	pyramidal (ours) {4,8,16,32}	our final
H	43.45	ANL-multi	pyramidal (ours) {4,8,16,32}	our final
I	**44.31**	ours	pyramidal (ours) {4,8,16,32}	our final
J	42.63	ours	single (standard)	{4,8,16} only for aux.
K	44.20	ours	pyramidal (ours) {4,32}	our final

Table A: The ablation experiment results reorganized in a unified view.

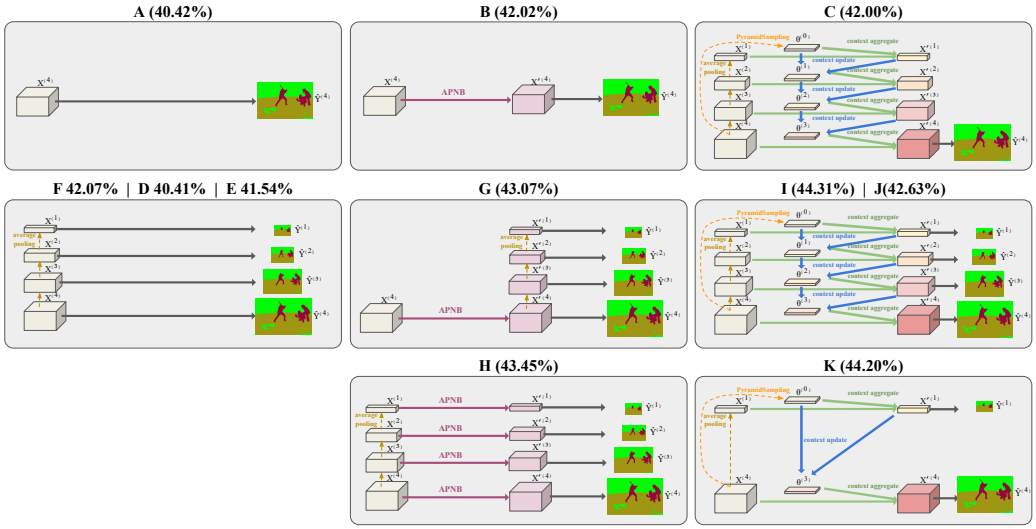

Figure B: The network architectures of the ablation experiments listed in Table A.

- *C* (42.00 %) is our coarse-to-fine contextual module, but only a single finest level semantic prediction is outputted.
- *F* (42.07 %) performs average-pooling on $X^{(4)}$ to get features $X^{(1)}, X^{(2)}, X^{(3)}$ of desired spatial scales, and each of $X^{(1)}, \ldots, X^{(4)}$ has its own *final projection layer*.
- *D* (40.41 %): is a variant of *F* where the training ground-truth is the raw pyramidal ground truth $Y^{(1)}, \ldots, Y^{(4)}$ and $U^{(1)}, \ldots, U^{(3)}$ without the re-labeling procedure to encourage specialization in each pyramid level.
- *E* (41.54 %) is a variant of *F* where the ground-truth unity-cell, instead of the true positive unity-cell, is used for "don't care" re-labeling.
- *G* (43.07 %) refines $X^{(4)}$ of *F* by appending ANL's APNB block to $X^{(4)}$.
- *H* (43.45 %) is similar to *G* but appends APNB block to each of $X'^{(1)}, X'^{(2)}, X'^{(3)}, X'^{(4)}$.
- *I* (44.31 %) is the final version of the proposed **Specialize and Fuse**.
- *J* (42.63 %) is similar to *I*, but the non-finest level predictions $\hat{Y}^{(1)}, \ldots, \hat{Y}^{(3)}$ are only used for auxiliary loss (the loss weight is set to 0.4) in the training phase and discarded in the inference phase.
- *K* (44.20 %) is similar to *I* but with less pyramid levels.

D. Architecture of our ResNet-decoder

The standard ResNet produces coarse features of output stride 32. To obtain better results, recent state-of-the-art methods employ the dilated version of ResNet, which generates features of output stride 8. However, we find such a modification leads to a lower speed and more memory footprint as shown in Table B (the 2nd row), so we adopt the standard ResNet with a lightweight decoder instead. Specifically, we use the features from the four stages of a standard ResNet, which respectively have output strides $4, 8, 16, 32$, and $256, 512, 1024, 2048$ latent channels. To reduce the computational cost, the numbers of latent channels are decreased from $[256, 512, 1024, 2048]$ to $[48, 96, 192, 384]$ with Conv3x3 layers. We then fuse the four features by applying the last stage of HRNet, which produces a high-resolution feature for the following semantic segmentation model head. The comparison of computational efficiency between ResNet variants is shown in Table B. Note that the constructed ResNet-decoder runs faster and consumes less GPU memory than ResNet101-dilated-os8. Moreover, the overall computational efficiency of adding our *specialize and fuse* head upon the ResNet-decoder is still better than that of ResNet101-dilated-os8 (which is the optimal bound for many recent ResNet-based state-of-the-art methods).

Method	Testing FPS ↑	Training Memory ↓
ResNet101	82	4.5G
ResNet101-dilated-os8 †	24	9.6G
ResNet101-decoder	32	8.4G
ResNet101-decoder + our head	26	8.9G

†Optimal bound for many recent works built upon dilated ResNet

Table B: Comparing the model efficiency measured on a GeForce RTX 2080 Ti with image size 512×512. FPS is averaged for processing 50 images. GPU memory consumption in training is monitored with a batch size of 4.

E. Do different pyramid levels specialize in different classes?

To demonstrate our intuition that different pyramid levels have their specializations in different classes, we show the per-class IoUs predicted by each level of the semantic pyramid for the 150 classes in the ADE20K dataset in Fig. C. For clearer visualization, we show the IoU difference between the prediction of a single level $\hat{Y}^{(\ell)}$ and the final fused $\hat{Y}$ instead of the original IoU of $\hat{Y}^{(\ell)}$, and we clip the values in the heatmaps to the range from -15% to 1%. A few entries in the heatmaps are larger than 0, which means that $\hat{Y}^{(\ell)}$ performs better than the fused $\hat{Y}$ on that class; it also implies that improving the performance of unity pyramid would provide opportunities for further enhancement. In Fig. C, the coarser pyramid levels (*e.g.* $\ell = 1$) generally look inferior to the finer levels, which could be caused by the fact that the single-level evaluation setting disadvantage the coarser levels since the per-class IoUs for each level are evaluated on the same per-pixel scale. However, this does not conflict with our intention to demonstrate that each pyramid level specializes in different classes. In addition, the finest level performs significantly worst for some classes (*e.g.*, sky, water, tower) since it is not trained to predict the central parts of these large-area classes. Some visual results presented in Fig. A provide more cues about this.

F. Qualitative results

In Fig. D, we show some visual comparisons of results between our approach and a baseline method—ANL with HRNet48 backbone.

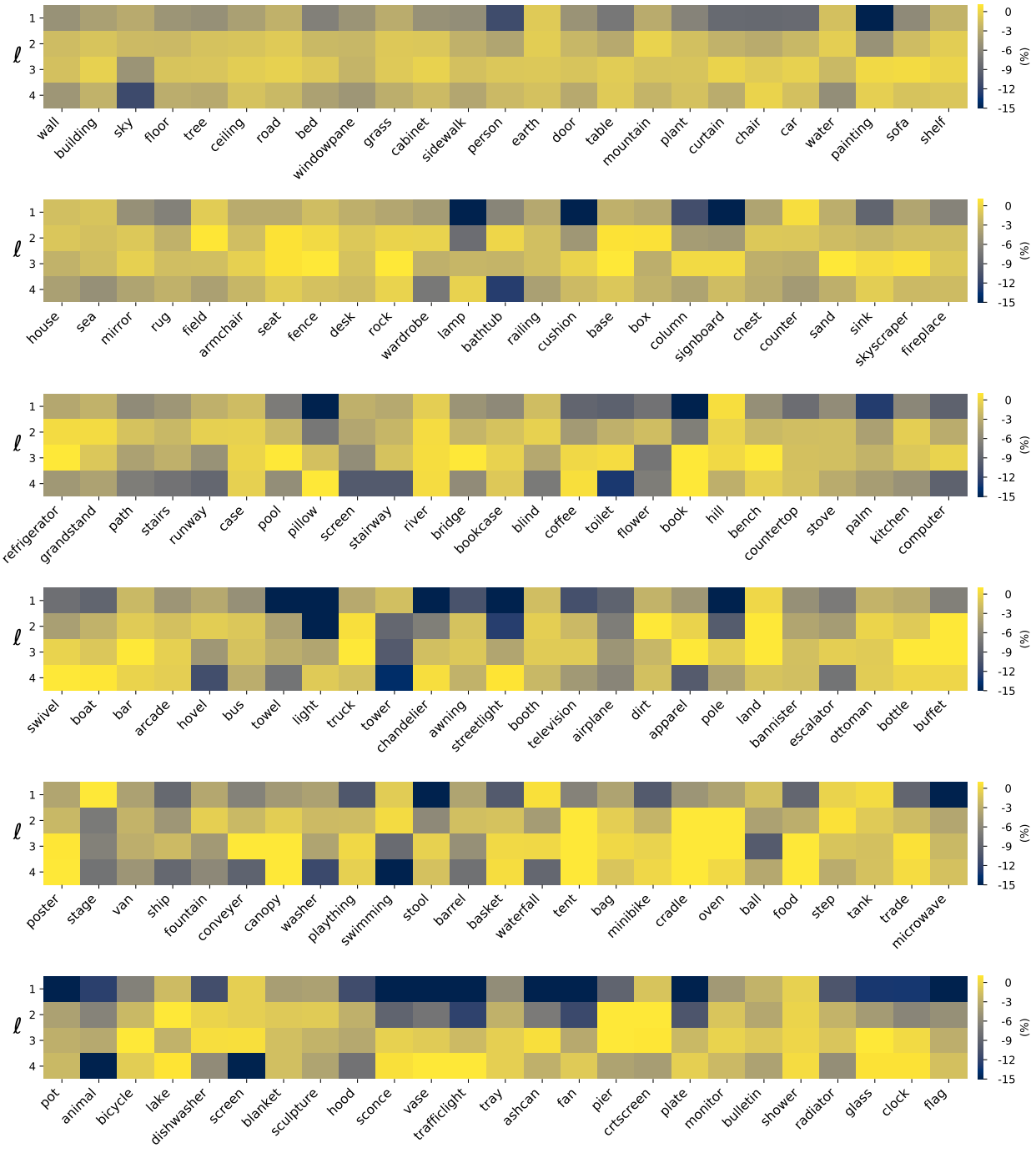

Figure C: Semantic segmentation performance at each level $\{\hat{Y}^{(\ell)}\}_{\ell=1,\ldots,4}$ on different classes. We show the IoU difference between using prediction of a level $\hat{Y}^{(\ell)}$ and using the fused prediction $\hat{Y}$. A brighter cell indicates that the pyramid level is more specialized in the class.

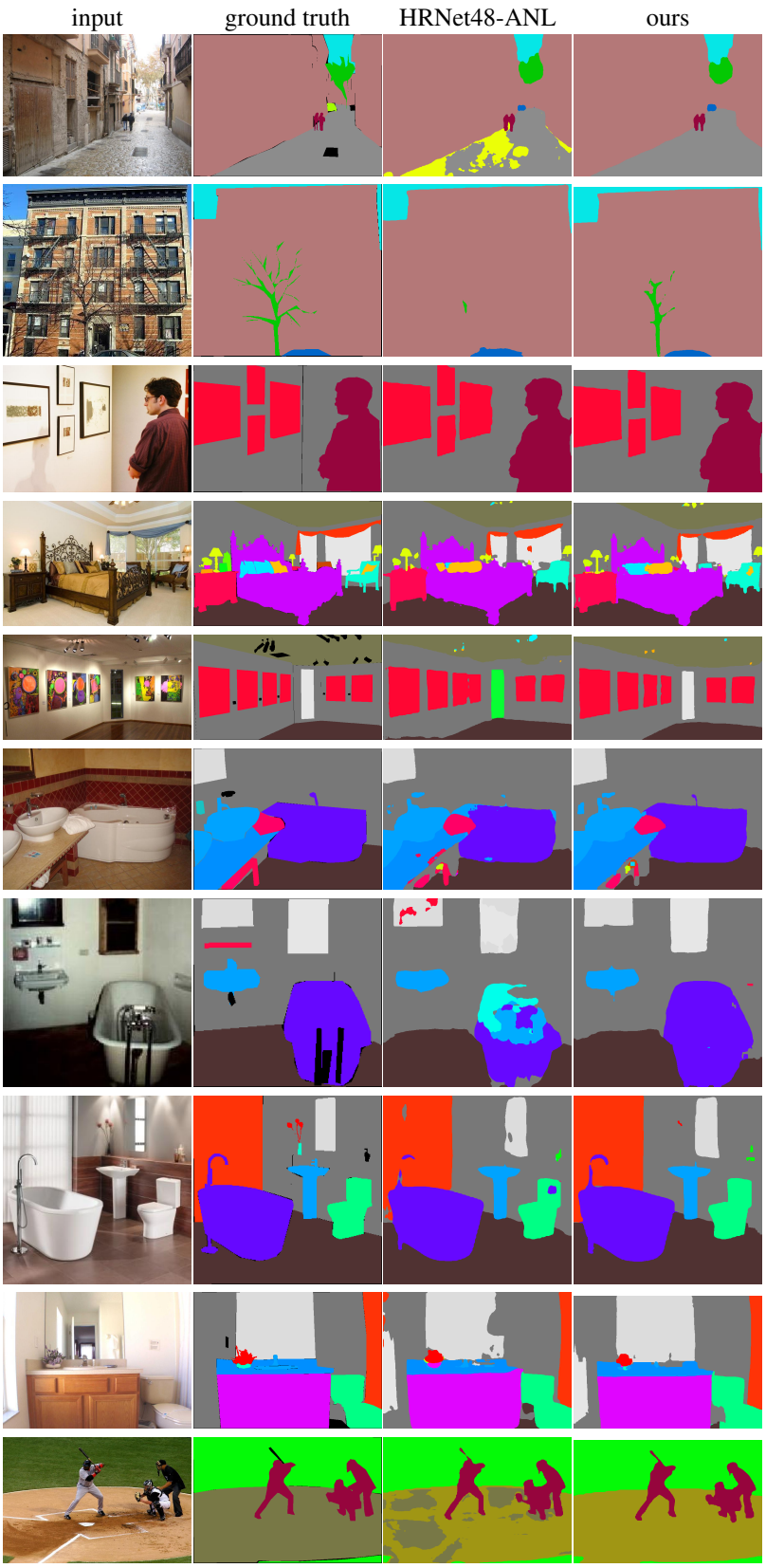

Figure D: Qualitative comparison. In the above examples, predictions by our approach are more consistent within an instance and also provide finer details.