Deep Grouping Model for Unified Perceptual Parsing
-
Supplementary Material

Zhiheng Li[1] Wenxuan Bao[2*] Jiayang Zheng[1] Chenliang Xu[1]
[1]University of Rochester [2]Tsinghua University
{zhiheng.li,jiayang.zheng,chenliang.xu}@rochester.edu bwx16@mails.tsinghua.edu.cn

1. Superpixel Pooling

Here, we give the formal definition of superpixel pooling (For simplicity, superscript denoting the level of the graph or the feature map in the hierarchy is omitted):

$$\mathbf{V}_n = \frac{1}{|\{\mathcal{S}_{ij} = n\}|} \sum_{i}^{H} \sum_{j}^{W} \mathcal{F}_{ij} \mathbb{I}(\mathcal{S}_{ij} = n) \ , \qquad (1)$$

where $\mathcal{S} \in \mathbb{R}^{H \times W}$ is the superpixel map, $\mathcal{F} \in \mathbb{R}^{H \times W}$ is the grid feature map, $\mathbf{V} \in \mathbb{R}^{|\{\mathcal{S}_{ij}=n\}| \times C}$ is vertex feature (C is the channel dimension), and $\mathbb{I}(\cdot)$ denotes the indicator function. $\mathcal{F}_{ij}$ and $\mathcal{S}_{ij}$ represent value at 2D position (i, j) in $\mathcal{F}$ and $\mathcal{S}$, respectively. $\mathcal{S}_{ij}$ stores the superpixel index that pixel at position (i, j) belongs to. $\mathbf{V}_n$ is the n-th vertex feature in $\mathbf{V}$. Thus, the output vertex feature in superpixel pooling comes from the average of features from superpixel region defined by superpixel map.

In practice, since superpixel map $\mathcal{S}$ is pre-computed from the image, the spatial size (H, W) of superpixel map equals to the image size, which is usually larger than the size of feature map $\mathcal{F}$. Therefore, before applying superpixel pooling operaion, we use bilinear interpolation to upsample the feature map $\mathcal{F}$ to the same size of the superpixel map. An alternative way is to downsample the superpixel map to the same size with the feature map, which, however, will make some small superpixels disappear. Thus, we upsample the feature map before using superpixel pooling.

The adjacency matrix $\mathbf{E}$ outputted by superpixel pooling is defined as:

$$\begin{aligned} \mathbf{E}_{mn} = \mathbb{I}(||\mathbf{p}, \mathbf{q}||_1 \leq 2 \mid \\ \exists \mathbf{p} \in \{(i,j) | \mathcal{S}_{ij} = m\} \wedge \\ \exists \mathbf{q} \in \{(i,j) | \mathcal{S}_{ij} = n\}) \ , \end{aligned} \qquad (2)$$

where both $\mathbf{p}$ and $\mathbf{q}$ are 2D positions. $||\cdot, \cdot||_1$ is Manhattan distance between two positions. $\mathbf{E}_{mn}$ denotes the adjacency between the m-th superpixel and the n-th superpixel. Eq. 2 means that superpixels m and n are adjacent if and only if there exists two positions in the respective superpixel regions where such two positions are 8-connected (Manhattan distance is less than or equal to 2).

*The work was performed while Wenxuan Bao was a visiting student at University of Rochester.

Algorithm 1 The Bottom-up Process

Input: $\{\mathcal{F}^l \mid l = 1, ..., L\}, \mathcal{S}$ ▷ $\mathcal{S}$: superpixel
Output: $\{\mathcal{G}^l \mid l = 1, ..., L\}, \mathbf{R}$
▷ // initialization
$\langle \mathbf{V}^1, \mathbf{E}^1 \rangle = \textit{Superpixel\_Pooling}\big(\mathcal{F}^1, \mathcal{S}\big)$
▷ // from the bottom level to the top level
for $l = 1 \rightarrow L - 1$ **do**
  $\langle \bar{\mathbf{V}}^{l+1}, \mathbf{E}^{l+1} \rangle, \mathbf{P}^l = \textit{EMGP}\big(\bar{\mathbf{V}}^l, \mathbf{E}^l\big)$ ▷ EMGP
  $\mathbf{U}^{l+1} = \textit{Superpixel\_Pooling}\big(\mathcal{F}^{l+1}, \mathcal{S}\big)$ ▷ Projection
  $\mathbf{V}^{l+1} = \textit{GConv}\big(\mathbf{U}^{l+1} \cup \bar{\mathbf{V}}_{l+1}, \mathbf{I} + \prod_{k=1}^{l} \mathbf{P}^k\big)$
end for
$\mathbf{R} = \textit{READOUT}\big(\mathbf{V}^L\big)$
Return $\{\mathcal{G}^l = \langle \mathbf{V}^l, \mathbf{E}^l \rangle \mid l = 1, ..., L\} \cup \{\mathbf{R}\}$

2. Algorithm

We show the complete algorithm of the proposed model for better reproducibility. The bottom-up process including *EMGP* and *Projection* is show in Alg. 1. Details of *EMGP* module are given in Alg. 2. Details of *TDMP* and *Re-projection* are given in Alg. 3.

On line 8 in Alg. 3, *Superpixel_Smear* means copying vertex features to the corresponding feature map locations according to the superpixel map $\mathcal{S}$. Formally, it is defined by (the superscript l denoting hierarchy level is omitted for simplicity):

$$\hat{\mathcal{F}}_{ij} = \hat{\mathbf{U}}_{\mathcal{S}_{ij}} \ . \qquad (3)$$

Thus, the feature at position (i, j) on the feature map $\hat{\mathcal{F}}$ is obtained from the S_{ij}-th feature of vertices $\hat{\mathbf{U}}$.

Algorithm 2 Expectation-Maximization Graph Pooling

Input: $\mathbf{V}^l, \mathbf{E}^l$
Output: $\langle \bar{\mathbf{V}}^{l+1}, \mathbf{E}^{l+1} \rangle, \mathbf{P}^l$

$\bar{\mathbf{V}}^{l+1} = \mathit{Uniform}(\mathbf{V}^l, |\mathbf{V}^{l+1}|)$ ▷ Initialization
for $k = 1 \rightarrow K$ **do**
  $\mathbf{P}^l_{ij} := \frac{1}{\mathbf{Z}^l_j} \exp\left(-\frac{||\mathbf{V}^l_i - \bar{\mathbf{V}}^{l+1}_j||^2}{\sigma^2}\right)$
  $\bar{\mathbf{V}}^{l+1} := (\mathbf{P}^l)^\intercal \mathbf{V}^l$
end for
$\mathbf{P}^l_{ij} := \frac{1}{\mathbf{Z}^l_j} \exp\left(-\frac{||\mathbf{V}^l_i - \bar{\mathbf{V}}^{l+1}_j||^2}{\sigma^2}\right)$
$\mathbf{E}^{l+1} \leftarrow (\mathbf{P}^l)^\intercal \mathbf{E}^l \mathbf{P}^l$
Return $\langle \bar{\mathbf{V}}^{l+1}, \mathbf{E}^{l+1} \rangle, \mathbf{P}^l$

Algorithm 3 Top-down Message Passing and Re-projection

Input: $\{\mathcal{G}^l \mid l = 1, ..., L\} \cup \{\mathbf{R}\}, \mathcal{S}$
Output: $\{\hat{\mathcal{F}}^l \mid l = 1, ..., L\}$
▷ initialization
for $l = L \rightarrow 1$ **do**
  $\tilde{\mathbf{P}}^l_{ij} = \frac{1}{\tilde{\mathbf{Z}}^l_i} \exp\left(-\frac{||\mathbf{V}^l_i - \mathbf{V}^{l+1}_j||^2}{\sigma^2}\right)$
end for
▷ from the top level to the bottom level
for $l = L \rightarrow 1$ **do**
  $\mathbf{V}^l := \mathit{GConv}\left(\mathbf{V}^{l+1} \cup \mathbf{V}^l, \mathbf{I} + \tilde{\mathbf{P}}^l\right)$ ▷ TDMP
  $\mathbf{U}^l = \mathit{Superpixel\_Pooling}\left(\mathcal{F}^l, \mathcal{S}\right)$ ▷ Re-projection
  $\hat{\mathbf{U}}^l = \mathit{GConv}\left(\mathbf{V}^l \cup \mathbf{U}^l, \mathbf{I} + \prod_{k=1}^{l} \tilde{\mathbf{P}}^k\right)$
  $\hat{\mathcal{F}}^l = \mathit{Superpixel\_Smear}\left(\hat{\mathbf{U}}^l, \mathcal{S}\right)$
end for
Return $\{\hat{\mathcal{F}}^l \mid l = 1, ..., L\}$

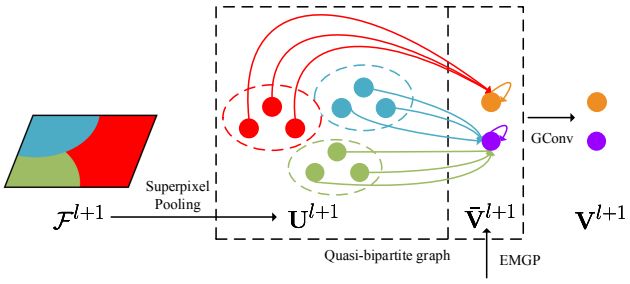

Figure 1: Illustration of the *Projection* module.

3. Projection and Re-projection

To illustrate how the *projection* and *re-projection* modules work in a more intuitive way, Fig. 1 and Fig. 2 give more details of the algorithms in the *projection* module and the *re-projection* module, respectively.

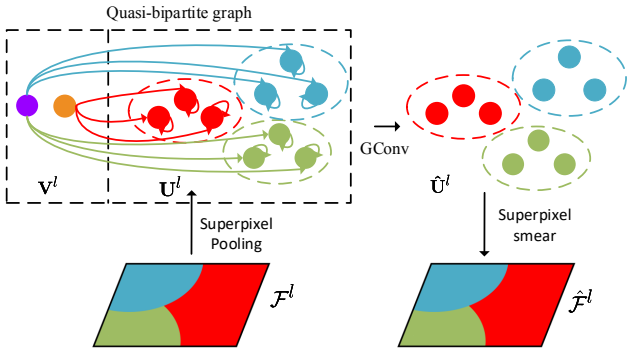

Figure 2: Illustration of the *Re-projection* module.

4. Network Architecture on ADE20k dataset

In this section, we introduce how DGM is incorporated into UPerNet, HRNetv2, DeepLabV3 in Sec. 5.4's **single-task training** of the paper. The backbone of all aforementioned networks is ResNet101 [3]. Similar with the architecture used in UPP task (Sec. 4 in the submitted paper), multi-resolution feature maps are extracted from the backbone, then augmented by DGM, and the sum of the original feature map and the re-projected feature map, ($\{\mathcal{F}_l + \hat{\mathcal{F}}_l | l = 1, ..., L\}$), is used for segmentation task. Thus, for simplicity reason, we just introduce the multi-level grid features of each network that are used as the input to DGM.

- **UPerNet:** We use the four levels of feature map outputted from the feature pyramid networks [5] in UPerNet as the input to DGM.
- **HRNetv2:** The four-scale output from *stage 4*'s *High Resolution Module* is used as the input to DGM.
- **DeepLabV3:** Four-resolution output of atrous spatial pyramid pooling (ASPP) module will be fed to DGM.

5. Qualitative Comparison on Broden+ Dataset

More examples of the qualitative comparison on Broden+ dataset [2] is provided in Fig. 3. Since no pixel-level texture ground-truth is provided in Broden+ dataset, we only show the comparison with other state-of-the-art methods on texture segmentation in Fig. 4. Our model can produce more consistent texture segmentation (e.g., basin on the first row and floor on the second and the third floor in Fig. 4).

6. Grouping Visualization

Generally speaking, the goal of grouping visualization is to map vertex index at the high-level graphs back to the bottom-level superpixel. First, we define hard pooling

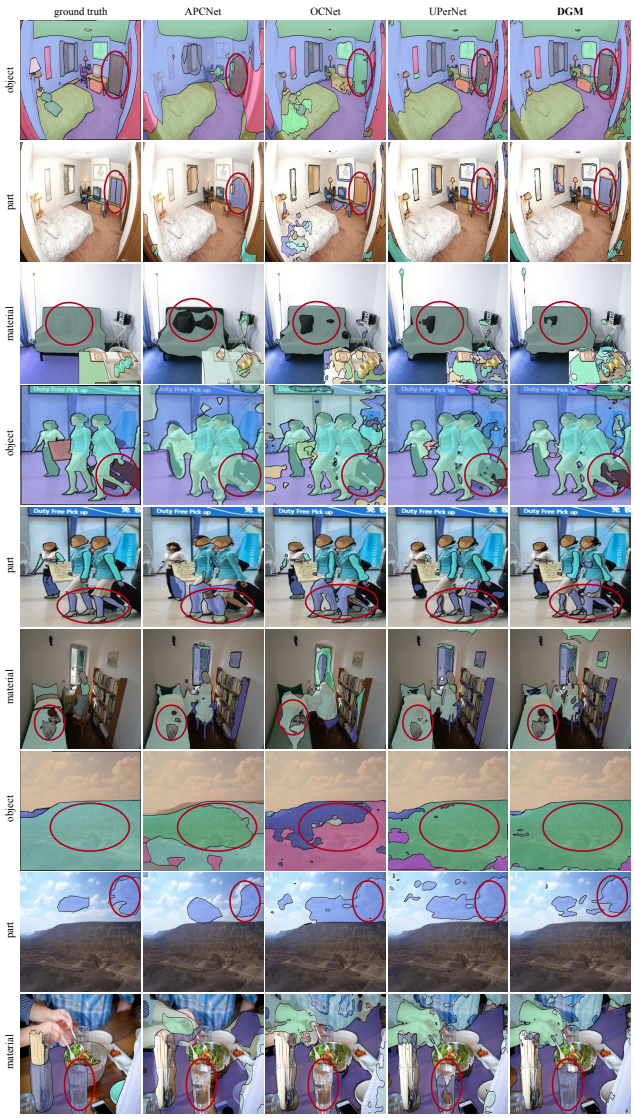

Figure 3: More qualitative comparisons on Broden+ dataset. Differences of segmentation result between different models are pointed out by red circle.

weights $\mathbf{Q}^l \in \mathbb{R}^{|\mathbf{V}^l| \times |\mathbf{V}^{l+1}|}$ by:

$$\mathbf{Q}^l_{ij} = \mathbb{I}(\tilde{\mathbf{P}}^l_{ij} = \max_k(\tilde{\mathbf{P}}^l_{kj})) \ , \tag{4}$$

where $\tilde{\mathbf{P}}^l$ is defined in Eq. 8 of the submitted paper. $\mathbf{Q}^l$ can be regarded as the one-hot version of $\tilde{\mathbf{P}}^l$. Vertex index $\mathbf{T}^l$ is defined by $\{\mathbf{T}^l_i = i | i = 1, ..., |\mathbf{V}^l|\}$. Finally, we can propagte vertex index from the graph at level l to level 1 by:

$$\mathbf{M}^l = Superpixel\_Smear((\prod_{l'=1}^{l} \mathbf{Q}^{l'})\mathbf{T}^l, \mathcal{S}) \ , \tag{5}$$

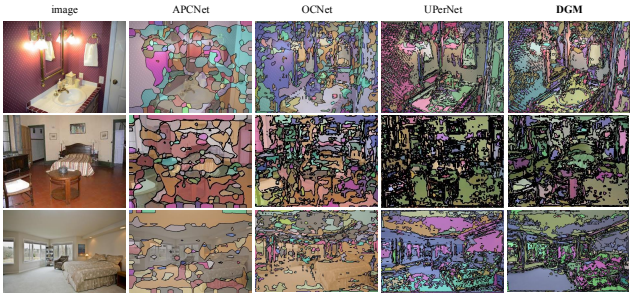

Figure 4: More qualitative comparisons on Broden+ dataset on texture segmentation.

where $\mathbf{M}^l$ is the superpixel map propagated from the graph at level l. In visualization, we randomly assign a unique color for each superpixel index for $\mathbf{M}^l$.

In Fig. 5 of the submitted paper, we use MCG [6] as the superpixel initialization at the bottom graph. Here, we show more examples of grouping visualization in Fig. 5 where SLIC [1] is used as superpixel extraction algorithm. The first five rows are successful cases and the last row is a failure case where all nodes are grouped into one node on level 4.

7. Click Propagation

First, we denote the propagated click at level l as click vector $\mathbf{C}^l \in \mathbb{R}^{|\mathbf{V}^l|}$, where $\mathbf{C}^l_i$ indicates the probability of the i-th vertex being a positive click or a negative click. The bottom level click $\mathbf{C}^1$ is initialized by the user's click (selection on the superpixel), which is a one-hot vector:

$$\mathbf{C}^1_i = \begin{cases} 1 & \text{vertex } i \text{ is clicked,} \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Remember that the goal of click propagation in interactive segmentation is to augment the user's click to related areas. Therefore, at each level l, we use two steps to propagate the user's clicks.

Step 1: Propagation via grouping Given the lower-level click $\mathbf{C}^l$, we use the pooling weights $\mathbf{P}^l$ to obtain the preliminary propagated click $\hat{\mathbf{C}}$ at level $l+1$:

$$\bar{\mathbf{C}}^{l+1} = (\mathbf{P}^l)^\intercal \mathbf{C}^l \ . \tag{7}$$

This step is very similar to EMGP (Eq. 2 in the submitted paper), where we replace the vertices feature $\mathbf{V}^l$ with the click vector $\mathbf{C}^l$.

Step 2: Propagation via adjacency matrix Given the preliminary click vector $\bar{\mathbf{C}}^{l+1}$, we use the adjacency matrix $\mathbf{E}^{l+1}$ predicted in *EMGP* with self-loop to propagate

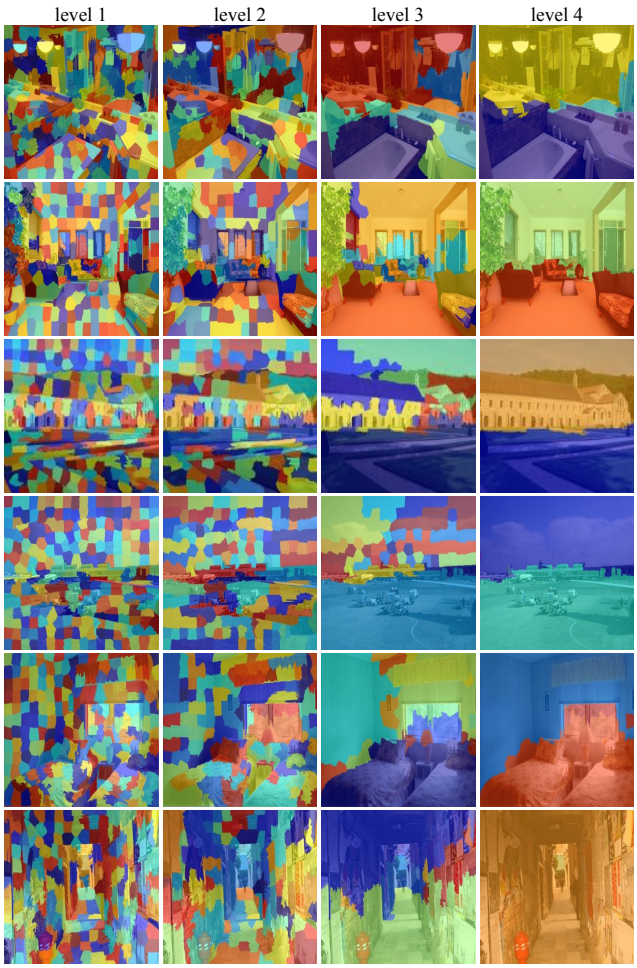

Figure 5: More examples of grouping visualization.

clicks at level $l+1$:

$$\mathbf{C}^{l+1} = (\mathbf{I} + \mathbf{E}^{l+1})\bar{\mathbf{C}}^{l+1} \quad . \tag{8}$$

Since the adjacency matrix $\mathbf{E}$ can capture local relationships between different vertices at the same level, step 2 can propagate clicks to other vertices more locally compared with step 1.

To visualize click propagation on the original image, we use $\tilde{\mathbf{P}}^l$ (see Eq. 8 in the submitted paper) to acquire the click map $\mathbf{N}^l \in \mathbb{R}^{H \times W}$ ($H \times W$ is the image size) corresponding to the click vector $\mathbf{C}^l$:

$$\mathbf{N}^l = Superpixel\_Smear((\prod_{l'=1}^{l} \tilde{\mathbf{P}}^{l'})\mathbf{C}^l, \mathcal{S}) \quad . \tag{9}$$

Finally, $\mathcal{N}^l$ is applied with min-max normalization for visualization, where values from 0 to 1 are visualized via black-to-white colors.

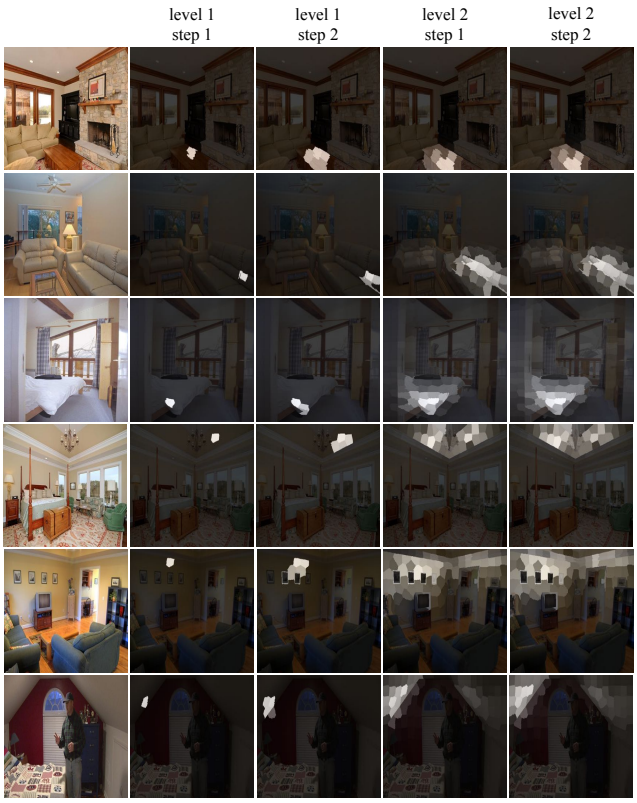

Figure 6: More examples of visualization of click propagation.

More examples of click propagation, including $\bar{\mathbf{C}}^l$ and $\mathbf{C}^l$, can be seen in Fig. 6. Since we observed that clicks will be propagated to larger contextual region (covering multiple objects and background) starting from level 3, we only show the visualization at level 1 and level 2.

8. Explainability with Grad-CAM

Following [7], the Grad-CAM $\mathbf{G}$ on level l graph's n-th vertex is defined by:

$$\mathbf{G}_n^l = \text{ReLU}(\sum_k \alpha_k^l \mathbf{V}_{k,n}^l) \quad , \tag{10}$$

where k denotes the channel dimension. The class specific weights α_k^l is defined by:

$$\alpha_k^l = \frac{1}{|\mathbf{V}^l|} \sum_{n=1}^{|\mathbf{V}^l|} \frac{\partial y^{gt}}{\partial \mathbf{V}_{k,n}^l} \quad , \tag{11}$$

where y^{gt} is the ground-truth class score outputted by the scene classification layer (before softmax). In this experiment, only the ground-truth scene label gt is used. However, it can be generalized to any class c.

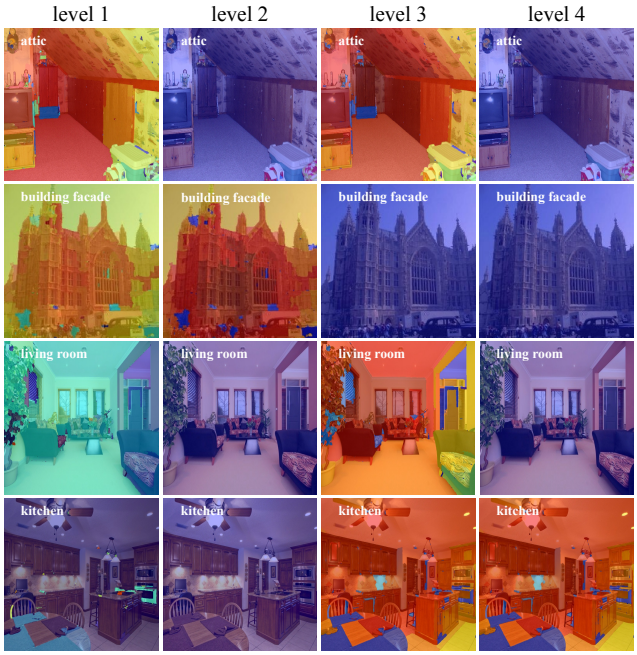

Figure 7: More examples of visualization of Grad-CAM.

In order to visualize $\mathbf{G}$, we use reversed pooling weights $\tilde{\mathbf{P}}^l$ (see Eq. 8 in the original paper) to obtain Grad-CAM on the original image $\mathbf{H}$:

$$\mathbf{H}^l = Superpixel\_Smear((\prod_{l'=1}^{l} \tilde{\mathbf{P}}^{l'})\mathbf{G}^l, \mathcal{S}) \ , \qquad (12)$$

where $\mathbf{H^1}$ denotes Grad-CAM on the original image obtained from $\mathbf{G}^l$. Finally, in the visualization, min-max normalization is applied on $\mathbf{H}$ to rescale the values in range of 0 to 1, visualized by blue-to-red colors.

More Grad-CAM visualizations can be seen in Fig. 7. One interesting finding from the visualization is that for the same image Grad-CAM only focuses on some vertices at one or two levels of the hierarchy. For example, the Grad-CAM visualization only focuses on the sofa on level 3's graph for the living room image on the 3rd row. We suspect that Grad-CAM may detect discriminative region only at the level when vertices can represent the discriminative object (*e.g.*, sofa) through grouping.

9. Comparison with state-of-the-art on UPP task

We compare DGM with GCU in terms of object segmentation, part segmentation, and scene classification tasks on Broden+ dataset [2]. Following the setting in [4], we add four GCU [4] modules on top of the ResNet [3] and concatenate the outputs. Following the hyperparameters used in [4], each one of the four GCUs has $(2, 4, 8, 32)$ vertices and output dimension is 256. The result in Tab. 1 shows that DGM outperforms GCU.

Method	Object		Part		Scene
	mIoU	P.A.	mIoU	P.A.	Top-1 Acc.
ResNet [3]+GCU [4]	22.49	72.71	23.34	41.49	71.55
ResNet+**DGM**	**24.76**	**75.15**	**31.26**	**50.55**	**71.87**
HRNet [8]	23.93	74.77	**31.79**	**51.12**	70.73
HRNet+**DGM**	**25.19**	**75.95**	31.72	50.92	**72.39**

Table 1: Comparing with GCU, HRNet on Broden+ dataset in terms of object segmentation, part segmentation, and scene classification tasks.

Furthermore, we add DGM on top of HRNet [8] backbone instead of ResNet. The result is shown in Tab. 1. Our method has similar performance with HRNet on part segmentation task and achieves much better result on object segmentation task and scene classification task.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, Nov 2012. 3

[2] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2, 5

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2, 5

[4] Yin Li and Abhinav Gupta. Beyond grids: Learning graph representations for visual recognition. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 9225–9235. Curran Associates, Inc., 2018. 5

[5] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2

[6] J. Pont-Tuset, P. Arbeláez, J. T. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping for image segmentation and object proposal generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(1):128–140, Jan 2017. 3

[7] Phillip E. Pope, Soheil Kolouri, Mohammad Rostami, Charles E. Martin, and Heiko Hoffmann. Explainability methods for graph convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 4

[8] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, March 2020. 5