Appendix

A. Implementation Details

In this section, we provide additional implementation details for experiments on CelebA dataset (Liu et al., 2015) and Taskonomy dataset (Zamir et al., 2018).

CelebA.
(a) LearnToBranch-VGG network: we train the network topological distribution for 30 epochs. The global learning rate is set to 10^{-5} and the learning rate for branching operation is set to 10^{-4}. We use exponential learning decay with decay factor 0.97 for every 2.4 epochs. After sampling the final architecture, we train the network for 30 epochs from scratch. We set the learning rate to 0.03, the weight decay to $5e^{-4}$, and the momentum to 0.9. We decay the learning rate by half for every 10 epoch.

(b) LearnToBranch-Deep-Wide network: we train the network topological distribution for 30 epochs. The global learning rate is set to 10^{-4} and the learning rate for branching operation is set to 10^{-2}. We use exponential learning decay with decay factor 0.97 for every 2.4 epochs. After sampling the final architecture, we train the network for 30 epochs from scratch. We set the learning rate to 0.05, the weight decay to $5e^{-4}$, and the momentum to 0.9. We decay the learning rate by half for every 15 epoch.

We visualize both network architectures (a) and (b) in Figure 6. We observe some grouping strategies learned by our method share some similarities with human intuition. For instance, network (a) groups 'Eyeglasses' and 'Narrow Eyes' and groups 'Mustasche' and 'No Beard'. Network (b) groups 'Black Hair' and 'Gray Hair' and groups 'Bald' and 'Receding Hairline'.

Taskonomy.
We train the network topological distribution for 30 epochs. The global learning rate is set to 10^{-3}, the learning rate for branching operations is set to 10^{-1}, and weight decay is set to 10^{-5}. We use exponential learning decay with decay factor 0.97 for every 1 epoch.

After sampling the final architecture, we train the network for 30 epochs from scratch. We set the learning rate to $5e^{-4}$, the weight decay to 10^{-4}, and the momentum to 0.9. We use exponential learning decay with decay factor 0.97 for every 1 epoch.

We follow the work in (Sun et al., 2019) and set the following task weightings: 1.0 for semantic segmentation, 3.0 for surface normal estimation, 2.0 for depth estimation, 7.0 for keypoint prediction, and 7.0 for edge detection. Note that we can further combine the proposed method with other adaptive task weighting methods. We leave this effort for future investigation.

Again following (Sun et al., 2019), for the semantic segmentation task, we ignore uncertain pixels (class 0) and background pixels (class 1). For the monocular depth estimation task, we ignore pixels with depth value larger than 64500 and normalize the disparities by taking the $\log$ operation and downscale by a factor of $\log(2^{16})$. For the surface normal prediction task, we normalize the three-dimensional normal vector from $[0, 255]$ to $[-1, 1]$. For the keypoint estimation and the edge detection tasks, we downscale the original values by a factor of 2^{16}. We then normalize the values from $[0, 0.005]$ to $[-1, 1]$ for keypoints and from $[0, 0.08]$ to $[-1, 1]$ for edges.

B. Learned Branching Features

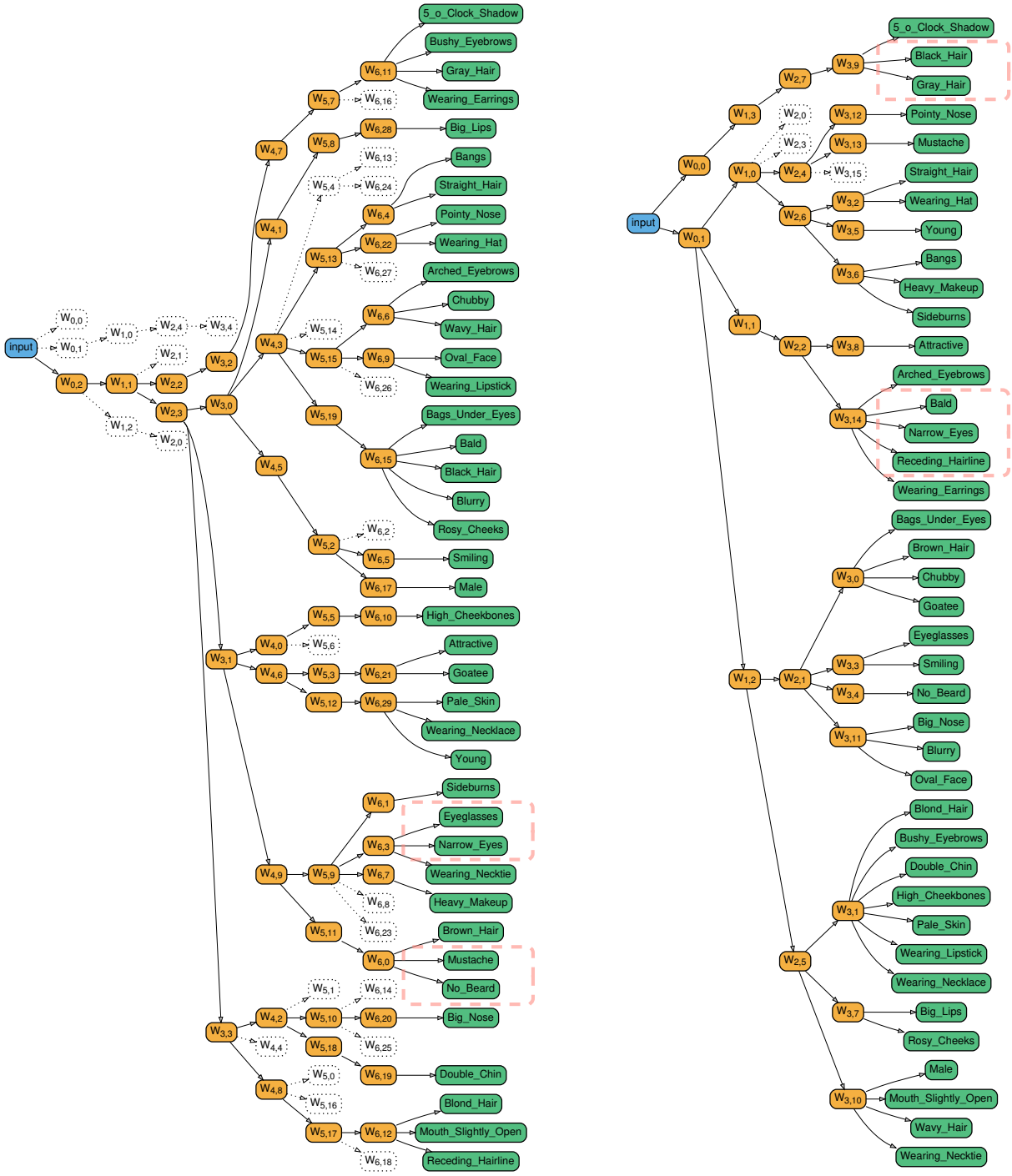
We use Network Dissection (Bau et al., 2017) to examine the features learned from Taskonomy dataset. We found that the SDN {segmentation, depth, normal} branch shows 35% increase in high-level features (object and part detectors) and 20% decrease in low-level features (texture detectors) compared to the shared layer before splitting. On the other hand, the EK {edge, keypoint} branch continues to focus on low-level features, showing no increase in high-level features due to the fact that {edge, keypoint} tasks are generally considered low-level tasks. Table 3 lists the number of detector counts before and after the branching (layer 13).

Table 3. Detector counts for different categories of input images at different layers using Network Dissection (Bau et al., 2017).

LAYER	OBJECT+PART DETECTORS	TEXTURE DETECTORS
LAYER_{13}	116	262
$\text{LAYER}_{14,\text{ SDN}}$	157	208
$\text{LAYER}_{14,\text{ EK}}$	118	253

C. Generalizability of the Learned Branching

We investigate whether the task grouping strategy learned from Tasknomoy dataset can be transferred to NYUv2 dataset on the three shared tasks across the two datasets. Following the metrics in Table 2, for {segmentation, normal, depth} tasks, we found that the grouping learned from Tasknomoy achieves {1.611, 0.739, 0.058} on NYUv2 test set while the grouping learned from NYUv2 training set achieves {1.572, 0.748, 0.058} on NYUv2 test set. The overall performance difference is relatively small at 1.23%. The experiment is performed on the NYUv2 labelled dataset with 795 training images and 654 test images using 256×256 image resolution.



(a) LearnToBranch-VGG (b) LearnToBranch-Deep-Wide

Figure 6. Network architectures learned from CelebA dataset. We observe some grouping strategies learned by our method share some similarities with human intuition. For instance, network (a) groups 'Eyeglasses' and 'Narrow Eyes' and groups 'Mustasche' and 'No Beard'. Network (b) groups 'Black Hair' and 'Gray Hair' and groups 'Bald' and 'Receding Hairline'. The groups are shown in red dotted rectangles. Transparent boxes denote removed nodes because they are not selected by any child nodes.