Appendix for: Unsupervised Part Segmentation through Disentangling Appearance and Shape

A. Implementation Details

A.1. TPS Transformation

We adopt the implementation open-sourced by [3] for TPS transformation. The source control points are from a 10×10 regular grid. All parameters are sampled from a gaussian distribution with zero mean and a given standard deviation. Each control point is perturbed by a parameter with standard deviation 0.001, and then applied with an additional perturbation with a standard deviation 0.005 with 50% probability. For the affine component, we set the standard deviation as 0.1 for the translation parameter and 0 for both the rotation and scale parameters. Note that all the coordinates are normalized to be in the range $[-1, 1]$ during deformation.

A.2. Perceptual Loss

We use the perceptual loss as in previous works [3, 8]. An ImageNet-pretrianed VGG-19 [6] is adopted as a feature extractor in our experiments. Given a pair of images (an input image and a reconstructed image in this work), we extract the output features of input, $\mathrm{conv1\_2}$, $\mathrm{conv2\_2}$, $\mathrm{conv3\_2}$, $\mathrm{conv4\_2}$, $\mathrm{conv5\_2}$ layers from the pretrained VGG-19 for each image, and weighted average the L2 distance of each pair of features. To balance the contribution of each layer, the weight of each layer is set as the reciprocal of its average L2 loss every 100 steps, as used in [1, 3].

A.3. Image Generation

After the *expand* operation, the L dimensional appearance features belonging to the same part are almost the same, which makes reconstruction very challenging. The encoding method proposed in [5] suggests that the coordinates of each pixel can help on reconstruction. Therefore, we normalize the coordinates of each pixel with respect to image width and height and stack all the coordinates to form a $2 \times H \times W$ tensor. Then we concatenate them with the rendered appearance feature maps $\mathbf{A}^{i \to j} \in \mathbb{R}^{L \times H \times W}$ to form a new tensor $\tilde{\mathbf{A}}^{i \to j} \in \mathbb{R}^{(L+2) \times H \times W}$. In our implementation, we use $\tilde{\mathbf{A}}^{i \to j}$ as input to the decoder D instead of $\mathbf{A}^{i \to j}$.

Dataset	K(# of parts)	λ_{rec}	λ_{cls}	λ_{fg}	λ_{bg}
CelebA	4/8	1.5	1.5	0.5	1.0
AFLW	4/8	1.5	1.5	0.5	1.0
CUB-1/2/3	4	1.5	1.5	0.3	1.0
VOC-car	4	1.5	1.0	0.3	0.1
VOC-bus	4	1.5	1.0	0.5	0.1
VOC-horse	4	1.5	3.0	0.3	0.1
VOC-aero	4	1.5	3.0	0.3	0.1
VOC-motor	4	1.0	1.0	0.3	0.1
VOC-cow	4	1.5	2.0	0.3	0.1
VOC-sheep	4	1.5	2.0	0.5	0.1
DeepFashion	9	1.5	3.0	0.5	0.1

Table 1: Settings for different experiments: the number of parts and combination weights in the final loss function.

A.4. Final Loss Function.

In the proposed method, the final loss function is a linear combination of four loss functions, including the reconstruction loss $\mathcal{L}_{rec}$, the part classification loss $\mathcal{L}_{cls}$, the foreground loss $\mathcal{L}_{fg}$, and the background loss $\mathcal{L}_{bg}$, as

$$\mathcal{L}_{sum} = \lambda_{rec}\mathcal{L}_{rec} + \lambda_{cls}\mathcal{L}_{cls} + \lambda_{fg}\mathcal{L}_{fg} + \lambda_{bg}\mathcal{L}_{bg}. \quad (1)$$

We use different settings for different tasks, as shown in Table 1. These combination weights were empirically obtained by coarse grid search. The models were trained on one GeForce RTX 2080 Ti GPU with 32 images per step for 30 epochs. Our model is light and efficient. On average, each model completes training in three hours.

B. More Visualization Results

We provide more visualization results on wild CelebA [4], CUB [2], and PASCAL VOC [7] from Fig. 1 to Fig. 4. These results indicate that our method keeps a good semantic consistency for objects with large variations.

Figure 1: Visualization results on the wild CelebA dataset with $K = 4$.

Figure 2: Visualization results on the wild CelebA dataset with $K = 8$.

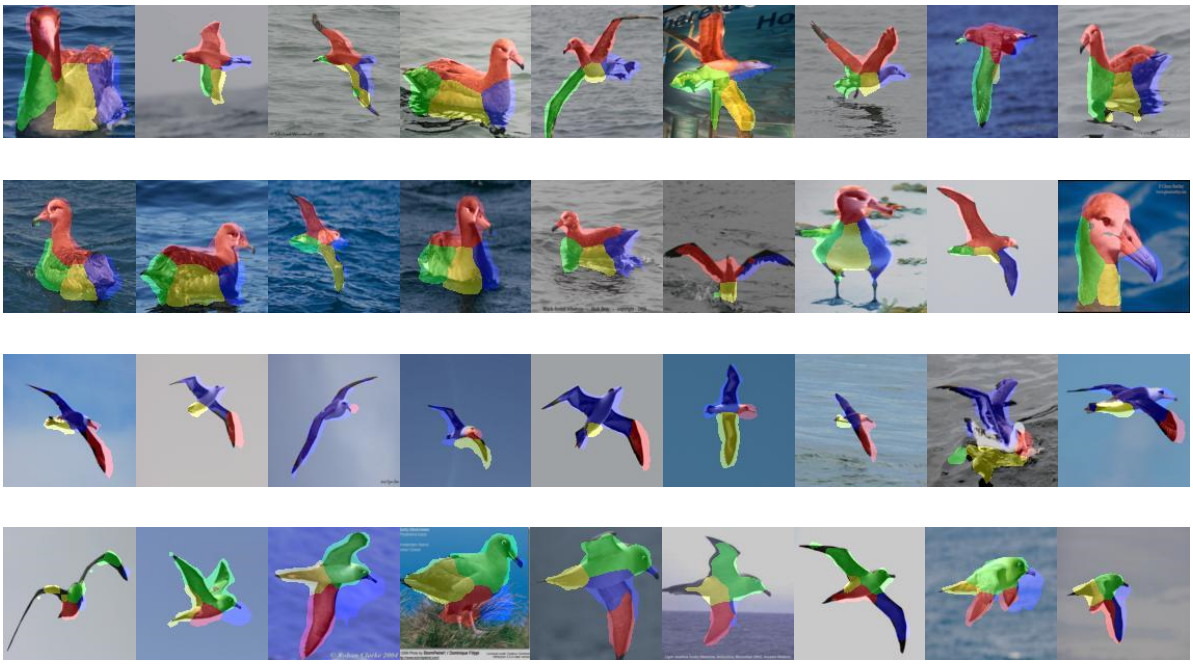

Figure 3: Visualization results on the CUB dataset with $K = 4$.

(a) Motor

(b) Sheep

(c) Bus

(d) Aero

(e) Cow

(f) Horse

(g) Car

Figure 4: Visualization results on PASCAL VOC.

References

[1] Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1520–1529, 2017.

[2] Mark Everingham, Luc Gool, Christopher K. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

[3] Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Unsupervised learning of object landmarks through conditional image generation. In *NIPS 2018: The 32nd Annual Conference on Neural Information Processing Systems*, pages 4016–4027, 2018.

[4] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[5] Dominik Lorenz, Leonard Bereska, Timo Milbich, and Bjorn Ommer. Unsupervised part-based disentangling of object shape and appearance. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10955–10964, 2019.

[6] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[7] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[8] Yinghao Xu, Ceyuan Yang, Ziwei Liu, Bo Dai, and Bolei Zhou. Unsupervised landmark learning from unpaired data. *arXiv preprint arXiv:2007.01053*, 2020.