Supplementary Material

A. Overview

We present implementation setups and additional experiments in this supplementary material. We clarify the architecture of the auto-shot segmenter in Section B. In Section C, we describe experiment details of car and horse part segmentation mentioned in the main paper. Section D compares using logit values vs one-hot vectors as the ground-truth target labels for the dataset generated by few-shot segmenters. Section E describes architectures used in GAN-derived representation analysis (Section 5.2). Section F investigates the segmentation performance using different choices of GAN's layers. Section G tests our method's ability to segment arbitrary parts. Section H explores representation learned from other unsupervised and self-supervised learning methods and compare their few-shot segmentation performance.

B. Auto-shot segmentor architecture

We adopt UNet architecture [10] for our auto-shot segmentation network. The overall architecture is shown in Table A. The network consists of encoder and decoder. For the encoder, there are 5 blocks of a convolutional layer, batch normalization, and a ReLU activation. Max-pooling is used after every 2 blocks to halve the input size. The decoder consists of 4 blocks of bilinear upsampling, 2 convolutional layers, batch normalization, and a ReLU activation. Lastly, a 1x1 convolutional layer is used to map the feature to the segmentation output.

C. Experiment Setups

For our experiments in the paper, we try to match the setups of those baseline methods as much as possible for a fair comparison. Most prior part segmentation methods [11, 14, 7] use bounding boxes to crop the image as they want to focus on only part segmentation not object localization. Some work [7, 14] eliminate objects deemed too small or objects with occlusion. In this section, we clarify the pre-processing step we use for each dataset.

Car part segmentation We follow the setup from [11] and evaluate on PASCAL-Part. We use the provided bounding boxes with class annotations in PASCAL-Part to select and crop images of cars, then use them as our test images. To compare with [14], we discard images whose bounding boxes overlap with other bounding boxes with IOU more than 5 and images that are smaller than 50x50 pixels. We fill the background with black color. Even though our network still predicts the background class, we calculate the average score without the background class.

Horse part segmentation We follow the horse part segmentation's setup from [7] and use the provided bounding boxes with class annotations in PASCAL-Part to extract horse images. We discard horse images smaller than 32x32 pixels.

Table A: Architecture of auto-shot segmentation network.

Layer	Kernel size	Stride	Batch normalization	Activation	Output size
Input	-	-	No	-	H x W x 3
Conv1a	3 x 3	1	Yes	ReLU	H x W x 64
Conv1b	3 x 3	1	Yes	ReLU	H x W x 64
Max Pool	2 x 2	2	No	-	H/2 x W/2 x 64
Conv2a	3 x 3	1	Yes	ReLU	H/2 x W/2 x 128
Conv2b	3 x 3	1	Yes	ReLU	H/2 x W/2 x 128
Max Pool	2 x 2	2	No	-	H/4 x W/4 x 128
Conv3a	3 x 3	1	Yes	ReLU	H/4 x W/4 x 256
Conv3b	3 x 3	1	Yes	ReLU	H/4 x W/4 x 256
Max Pool	2 x 2	2	No	-	H/8 x W/8 x 256
Conv4a	3 x 3	1	Yes	ReLU	H/8 x W/8 x 512
Conv4b	3 x 3	1	Yes	ReLU	H/8 x W/8 x 512
Max Pool	2 x 2	2	No	-	H/16 x W/16 x 512
Conv5a	3 x 3	1	Yes	ReLU	H/16 x W/16 x 512
Conv5b	3 x 3	1	Yes	ReLU	H/16 x W/16 x 512
Upsample Concat(Conv4b)	-	-	No	-	H/8 x W/8 x 1024
Conv6a	3 x 3	1	Yes	ReLU	H/8 x W/8 x 512
Conv6b	3 x 3	1	Yes	ReLU	H/8 x W/8 x 512
Upsample Concat(Conv3b)	-	-	No	-	H/4 x W/4 x 512
Conv7a	3 x 3	1	Yes	ReLU	H/4 x W/4 x 256
Conv7b	3 x 3	1	Yes	ReLU	H/4 x W/4 x 256
Upsample Concat(Conv2b)	-	-	No	-	H/2 x W/2 x 256
Conv8a	3 x 3	1	Yes	ReLU	H/2 x W/2 x 128
Conv8b	3 x 3	1	Yes	ReLU	H/2 x W/2 x 128
Upsample Concat(Conv1b)	-	-	No	-	H x W x 128
Conv9a	3 x 3	1	Yes	ReLU	H x W x 64
Conv9b	3 x 3	1	Yes	ReLU	H x W x 64
Conv10	1 x 1	1	No	-	H x W x Classes

Table B: Comparison between IOU scores of auto-shot segmenter using one-hot masks vs logit values as annotations.

Model	Face	Horse	Car
One-hot	82.1	69.9	70.6
Logits	84.5	72.3	72.6

D. Logit vs One-hot Labels for Auto-shot Segmentation

As explained in the main paper, we use our few-shot segmenter along with a trained GAN to generate a labeled dataset for our auto-shot segmenter. Each training example in this dataset consists of a generated image and its corresponding segmentation map predicted from our few-shot segmenter. For this dataset, each pixel in each segmentation map is represented as a set of logit values corresponding to the probabilities of different part classes, as opposed to a standard one-hot encoding of the part class. The motivation is to keep the class confidence scores that could provide useful information for the auto-shot segmenter and help prevent over-confident predictions based on spurious or ambiguous target labels.

In this experiment, we compare our auto-shot segmenter trained with our proposed logit values to the standard one-hot target labels. Table B shows that using logit labels outperforms one-hot labels on all three object categories.

Table C: Architecture of S-network.

Layer	Kernel size	Dilation rate	Padding	Output channel size
Conv1	3 x 3	1	1	128
Conv2	3 x 3	2	2	64
Conv3	3 x 3	1	1	64
Conv4	3 x 3	2	2	32
Conv5	3 x 3	1	1	number of classes

E. Various architectures used in GAN-derived Representation Analysis

In this section, we describe architectures used in GAN-derived representation analysis (Section 5.2). The small, medium, and large architectures are shown in Table C, Table D, and Table E respectively.

F. Effects of GAN's Layer Selection

A study on style mixing from StyleGAN [3] suggests that information in earlier layers of the GAN's generator controls the higher-level appearance of the output image, whereas late layers control the subtle details. In this experiment, we explore whether choosing different subsets of layers from the generator can affect the performance. Similarly to the study in StyleGAN, we roughly split the layers into 3 groups: (A) the coarse style (from resolution $4^2 - 8^2$), (B) the middle style ($16^2 - 32^2$), and (C) the fine style ($64^2 - 1024^2$). Then, we test our one-shot segmenter by feeding different combinations of these groups shown in Table F. The result shows that the representation from group B yields the highest IOU with a slight increase from using all layers, and group A performs the worst. This suggests that the middle layers that control the variation and appearance of facial features are more useful for few-shot face segmentation and that layer selection could play an important role.

G. Arbitrary Segmentation

We have shown that features from GANs are effective for part segmentation when those parts, so far, correspond to some natural semantic regions such as eyes and mouth. In this experiment, we test whether features from GANs are restricted to those parts and whether our method can generalize to any arbitrary segmented shapes. We manually create random shaped annotations and use them to train our few-shot network. Figure A shows that our method can still handle semantic-less parts and produce consistent segmentation across people and head poses.

H. Comparison on Representation Learned from Other Tasks

In our paper, we demonstrate the effectiveness of representation from GANs on few-shot semantic part segmentation. However, apart from GANs and generative tasks, there are other tasks and networks that can be used for representation learning. In this section, we compare how other representation tasks perform on few-shot human face segmen-

Table D: Architecture of M-network.

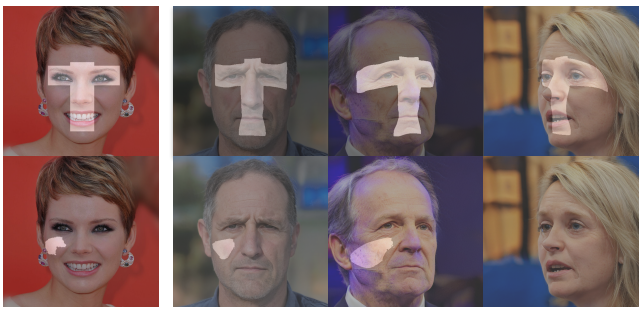
Layer	Kernel size	Dilation rate	Padding	Output channel size
Conv1	3 x 3	1	1	128
Conv2	3 x 3	2	2	64
Conv3	3 x 3	4	4	64
Conv4	3 x 3	1	1	64
Conv5	3 x 3	2	2	64
Conv6	3 x 3	4	4	32
Conv7	3 x 3	1	1	number of classes

Table E: Architecture of L-network.

Layer	Kernel size	Dilation rate	Padding	Output channel size
Conv1	3 x 3	1	1	128
Conv2	3 x 3	2	2	64
Conv3	3 x 3	4	4	64
Conv4	3 x 3	8	8	64
Conv5	3 x 3	1	1	64
Conv6	3 x 3	2	2	64
Conv7	3 x 3	4	4	64
Conv8	3 x 3	8	8	32
Conv9	3 x 3	1	1	number of classes

Table F: Comparison of 1-shot segmentation performance with representation from different layers of GANs.

Layers	Resolution	weighted mean IOU
A	$4^2 - 8^2$	59.6
B	$16^2 - 32^2$	**79.1**
C	$64^2 - 512^2$	69.0
A-B	$4^2 - 32^2$	75.2
B-C	$16^2 - 512^2$	75.0
A-B-C (all)	$4^2 - 512^2$	**77.9**



Annotation One-shot Results

Figure A: Given segmentation masks with arbitrary, meaningless shapes that have no clear boundary, our approach can infer the same regions across different face images and can even recognize it when the regions are stretched or out of view.

Table G: Weighted IOU scores on 10-shot face segmentation with representation learned from different tasks / networks.

Task / Network	4 class			10 class		
	1-shot	5-shot	10-shot	1-shot	5-shot	10-shot
GANs	71.7	82.1	83.5	77.9	83.9	85.2
VAE	55.1	69.7	72.8	51.6	58.4	65.5
Jigsaw Solving	23.3	46.3	60.0	41.6	54.9	60.4
Colorization	32.1	39.7	51.7	49.1	55.5	66.1
HED	38.2	48.5	60.9	48.9	67.2	70.3
Bilat Filtering	10.9	22.4	49.9	29.2	45.3	54.5

Figure B: Comparison on 10-shot human face segmentation with representation learned from different tasks / networks.

tation. We focus on unsupervised or self-supervised tasks in this study because they require no manual labels and can be applied to any new unseen class.

We select networks learned to solve 5 different tasks for this experiment: a) VAE [4], a well-known approach used to synthesize images or find a compact representation of an image through auto encoding with regularized latent distribution, b) jigsaw solving network [9], a successful representation learning approach for ImageNet classification that achieves comparable results to a fully-supervised baseline, c) holistically-nested edge detection (HED) network whose nature of the task is closely related to image segmentation, e) colorization network whose representation displays good results in a segmentation task in [5], and f) bilateral filtering network which solves a simple task but has to be edge-aware.

Upstream Networks and Feature Extraction

We train the following baseline networks with human facial images from CelebA dataset [6] which comprises 160,000 training images.

Variational Autoencoder (VAE) We use ResNet architecture [1] for both encoder and decoder and train the network with a perceptual loss from all layers of VGG19 [13]. The generated images are realistic but blurrier compared to those generated from state-of-the-art GANs.

We extract a pixel-wise representation from VAE by feeding an input image into the encoder through the decoder and extracting all activation maps from all convolutional layers in the decoder of VAEs. Then, similarly to GANs feature extraction, all activation maps are upsampled into the dimension of the biggest activation maps and concatenated together in the channel dimension.

Jigsaw Solving Network Setup We follow the setup and network architecture from [9] to implement a jigsaw solving

network. This task asks the network to predict one of the 1,000 predefined permutations of shuffled image patches. To explain the process, first we randomly crop a big square patch from an image and divide the patch into a 3x3 grid. All 9 partitioned squares are then cropped again into slightly smaller squares. The 9 squares are then shuffled into one of 1,000 predefined permutations, and the network is trained to predict the 1,000 predefined permutation from the nine squares as input.

We extract features from all convolutional layers of the jigsaw solving network using the same method as in our VAE representation extraction.

Images Translation with Pix2Pix we use Pix2Pix [2] framework to create networks that take an image as input and predict some transformed version of that image. We use three types of image transformations: colorization, holistically-nested edge detection (HED) [12], and bilateral filtering. For colorization, we transform the images to Lab space, then use L channel as input and train the network to predict values in a channel and b channel. For HED, we use HED implementation and pretrained weights from [8].

Pix2Pix is a conditional GAN, and a latent optimizer is not needed for embedding the input image because it can take images as input directly to its UNet-based architecture. We feed an input image into Pix2Pix's encoder but only use activation maps from all convolutional layers of the generator (or decoder) to construct a pixel-wise representation.

Results

The segmentation results are shown in Figure B and Table G. Representation acquired from GANs produces the best results among all networks. For VAE, the segmentation results are good for 3-class segmentation, second only to GANs. However, the results are noticeably worse in 10-class segmentation because of the bad results in hair class. The segmentation results with representation from the jigsaw solving task can locate facial features, but the quality of contour is poor. HED representation has comparatively good segmentation results which could be because segmentation and edge detection problems are closely related, both requiring locating part boundaries. However, since HED often fails to find the nose boundary, nose segmentation is worse than other three previous tasks. Colorization is another task that cannot find the nose boundary as the nose and all facial skin share the same color, and there is no need for the colorization network to learn to discriminate noses from faces. The bilateral filtering task has the worst segmentation results as the network may only learn to find objects' edges and a kernel that can blur images.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[2] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017. 4

[3] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 2

[4] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 3

[5] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6874–6883, 2017. 3

[6] 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. 3

[7] Shujon Naha, Qingyang Xiao, Prianka Banik, Md Alimoor Reza, and David J Crandall. Pose-guided knowledge transfer for object part segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 906–907, 2020. 1

[8] Simon Niklaus. A reimplementation of HED using PyTorch. https://github.com/sniklaus/pytorch-hed, 2018. 4

[9] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016. 3

[10] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 1

[11] Stavros Tsogkas, Iasonas Kokkinos, George Papandreou, and Andrea Vedaldi. Deep learning for semantic part segmentation with high-level guidance. *arXiv preprint arXiv:1505.02438*, 2015. 1

[12] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1395–1403, 2015. 4

[13] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 3

[14] Yifan Zhao, Jia Li, Yu Zhang, Yafei Song, and Yonghong Tian. Ordinal multi-task part segmentation with recurrent prior generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 1