Supplementary material for
“Unsupervised Creation of Parameterized Avatars”

1 Summary of Notations
Tab. 1 itemizes the symbols used in the submission. Fig. 2,3,4 of the main text illustrate many of these symbols.

2 DANN results
Fig. 1 shows side by side samples of the original image and the emoji generated by the method of [1]. As can be seen, these results do not preserve the identity very well, despite considerable effort invested in finding suitable architectures.

3 Multiple Images Per Person
Following [4], we evaluate the visual quality that is obtained per person and not just per image, by testing TOS on the Facescrub dataset [3]. For each person \( p \), we considered the set of their images \( X_p \), and selected the emoji that was most similar to their source image, i.e., the one for which:

\[
\arg\min_{x \in X_p} ||f(x) - f(e(G(x))))||.
\] (1)

Fig. 2 depicts the results obtained by this selection method on sample images from the Facescrub dataset (it is an extension of Fig. 7 of the main text). The figure also shows, for comparison, the DTN [4] result for the same image.

4 Detailed Architecture of the Various Networks
In this section we describe the architectures of the networks used in for the emoji and avatar experiments.

4.1 TOS
Network \( g \) maps DeepFace’s 256-dimensional representation [5] into \( 64 \times 64 \) RGB emoji images. Following [4], this is done through a network with 9 blocks, each consisting of a convolution, batch-normalization and ReLU, except the last layer which employs Tanh activation. The odd blocks 1,3,5,7,9 perform upscaling convolutions with 512-256-128-64-3 filters respectively of spatial size \( 4 \times 4 \). The even ones perform \( 1 \times 1 \) convolutions [2]. The odd blocks use a stride of 2 and padding of 1, excluding the first one which does not use stride or padding.

Network \( e \) maps emoji parameterization into the matching \( 64 \times 64 \) RGB emoji. The parameterization is given as binary vectors in \( \mathbb{R}^{811} \) for emojis; Avatar parameterization is in \( \mathbb{R}^{354} \). While there are dependencies among the various dimensions (an emoji cannot have two hairstyles at once), the binary representation is chosen for its simplicity and generality. \( e \) is trained in a fully supervised way, using pairs of matching parameterization vectors and images in a supervised manner.

The architecture of \( e \) employs five upscaling convolutions with 512-256-128-64-3 filters respectively, each of spatial size \( 4 \times 4 \). All layers except the last one are batch normalized followed by a ReLU activation. The last layer is followed by Tanh activation, generating an RGB image with values in range \( [-1, 1] \). All the layers use a stride of 2 and padding of 1, excluding the first one which does not use stride or padding.
Network $d$ takes $152 \times 152$ RGB images (either natural or scaled-up emoji) and outputs log-probabilities predicting if the image is fake or real. It consists of 6 blocks, each containing a convolution with stride 2, batch normalization, and a leaky ReLU with leakiness coefficient of 0.2. Each block contains 64-128-256-512-512-3 filters respectively. As before, the last layer does not employ batch normalization and ReLU.

Network $c$ maps a $64 \times 64$ emoji to parameterization vector. It contains five convolutional layers, each followed by batch normalization and a leaky ReLU with a leakiness coefficient of 0.2. Each layer contains 64-128-256-512-813 filters respectively. The last layer is followed by Tanh activation, generating a parameterization vector with values in range $[-1, 1]$.

The networks used for the synthetic polygon experiment are somewhat simpler: $g$ has the same structure of as in the emoji experiment excluding the even convolutions i.e., it does not contain the $1 \times 1$ convolutions. The architecture of $d$ is unchanged. Finally, the architectures of $e$ and $c$ are updated to match the synthetic experiment parameterization. $e$ is changed to map a parameterization vectors in $\mathbb{R}^3$ to RGB images, and $c$ is trained to predict such a vector.

### 4.2 DANN

In the domain adaptation experiments, network $p$ extracts 2048-dimensional feature vectors from $64 \times 64$ RGB images. It resembles the structure of network $c$ - with 4 convolution layers. Each convolution is with 64-128-256-512 filters respectively. The last convolutional layer employs a stride of 1 instead of 2 and does not use batch-normalized or leaky ReLU. Finally, the network output is flattened to 1-dimensional feature vector.

The label prediction network $l$ accepts as input feature vectors generated by $p$ and outputs emoji param-
Figure 1: Shown, side by side are sample images from the CelebA dataset and the results obtained by the DANN domain adaptation method [1]. These results are not competitive.

...ization vectors matching the input image. It consists of 3 fully connected layers. Each hidden layer is followed by batch-normalization and leaky ReLU activation. The last layer is followed by Tanh activation. The hidden layers contain 1024 and 512 units respectively.

The discriminator $d$ predicts the input image domain given its feature vector. It consists of two fully connected layers with 512 hidden units. The hidden layer is followed by batch normalization and leaky ReLU activations. It is preceded by a gradient reversal layer to ensure that the feature distributions of both domains are similar. The last layer is followed by Sigmoid activation, predicting the input image domain.

References

Figure 2: The results obtained by the TOS method for a sample of individuals from the Facescrub dataset. Shown, side by side, are the image used to create the TOS and the DTN emoji, the DTN emoji, and the TOS emoji, obtained by $\left(\alpha\circ\alpha\circ\alpha\circ f\right)$. The image that represents a person maximizes, out of all images for this person, $f$-constancy for the TOS method.