Realistic Full-Body Anonymization with Surface-Guided GANs (Supplementary)

Håkon Hukkelås  Morten Smebye  Rudolf Mester  Frank Lindseth
Norwegian University of Science and Technology
hakon.hukkelas@ntnu.no

Contents

A. Additional Experimental Details 1

I. Network Architecture and Hyperparameters . 1
II. Face Quantitative Evaluation . . . . . . . . . 2
III. DeepFashion CSE-Dataset . . . . . . . . . . 2

B. COCO Anonymization Details 3

I. COCO-Body Dataset . . . . . . . . . . . . . 3

C. Additional Quantitative Results 3

I. Truncation Trick . . . . . . . . . . . . . . . 3
II. Effect of Anonymization for Computer Vision 3

D. Additional Qualitative Results 4

I. COCO-Body . . . . . . . . . . . . . . . . . 4
II. DeepFashion . . . . . . . . . . . . . . . . . 4
III. COCO . . . . . . . . . . . . . . . . . . . . 4

A. Additional Experimental Details

We describe the general setup of the COCO-Body model here and state specific changes to DeepFashion in Sec. A.3.

All experiments were performed with Pytorch v1.11 [22] with 2x Tesla V100-32GB GPUs. FID and PPL is computed with torch fidelity [20], where we modify the code to support conditional image synthesis.

I. Network Architecture and Hyperparameters

We use equalized learning rate [8] for all trainable parameters and use the Adam optimizer with a learning rate=0.001, $\beta_1 = 0.0, \beta_2 = 0.99$ and $\epsilon = 10^{-8}$. Note that we set the learning rate to 0.002 for Config E. Each model is trained until the discriminator has seen $12M$ images. All networks are trained with mixed precision using the implementation provided by Pytorch [22].

**Generator Architecture** The generator architecture is a U-Net architecture [24] following design principles of StyleGAN2 [11]. There are 5 residual blocks in the encoder and 5 in the decoder, where each block has two 3x3 convolutions. The residual blocks have the following number of output channels (Config E): [128, 256, 512, 512, 512, 512, 512, 256, 128, 64]. Convolutions of the same block has the same number of output channels. For other configs, we straightforward scale down the reported output channels to match the number of parameters reported in the main paper. Bilinear upsampling/downsampling is performed after each block (except the end/start of the encoder/decoder). We use $1 \times 1$ residual skip connections between the first 4 and last 4 blocks of the encoder/decoder. Each convolution in the residual blocks (except residual/u-net skip connections) has the following order of operations: SAM feature modulation $\rightarrow$ convolution $\rightarrow$ LeakyReLU (slope=.2) $\rightarrow$ normalization. The input image resolution is $288 \times 160$ and the minimum feature map resolution is $36 \times 20$. For inference, we use the exponential moving average [28] of the generator, where we use $\beta = 0.9977$. For all models (except those with V-SAM, INADE [25], Co-mod [30] or StyleGAN [11]), we linearly project the latent variable to a $1 \times 32 \times 32$, then add it to the output of the encoder.

**FPN-Discriminator** The FPN-discriminator consists of 6 residual blocks, with the following number of output channels (Config E) for each: [128, 256, 512, 512, 512, 512], which results in 34M parameters. For the FPN-Head, we add a $1 \times 1$ convolution that linearly transforms the output of each residual block to the number of output channels (16 for CSE embedding, 26 for semantic segmentation). These feature maps are then bilinearly upsampled and added.

**Loss function** We use the non-saturating adversarial loss [3] and regularize the discriminator with epsilon penalty [8] and r1-regularization [18]. We mask the r1-regularization by $M$, similar to [29, 7], and apply it every 16th gradient step (known as ”lazy regularization” [11]). The total loss is given by

$$L = L_{GAN} + \lambda_{EP} \cdot L_{EP} + \lambda_{GP} \cdot L_{GP} + \lambda_{CSE} L_{CSE}$$ (1)
where \( \lambda_{EP} = 0.001 \), \( \lambda_{GP} = 5 \), and \( \lambda_{CSE} = 0.1 \). \( \lambda_{EP} \) is identical to Progressive Growing GAN [8]. \( \lambda_{GP} \) and \( \lambda_{CSE} \) are determined by a rough hyperparameter search, where we tested \( \lambda_{CSE} \in [0.1, 0.5, 1, 2, 5, 10, 50] \) and \( \lambda_{GP} \in [0.1, 0.5, 1, 2, 5, 10, 20, 100] \). We did the hyperparameter search only on Config B. No other hyperparameter search is done, unless stated otherwise.

### Data Augmentation

We use a limited amount of data augmentation, but find that it significantly improves quality of generated samples. We adapt the augmentation pipeline of StyleGAN2-ada [9] and modify the Pytorch implementation to support conditional image synthesis. The pipeline includes general geometrical transformations, color transformations, rotation, and horizontal flip. We significantly limit the amount of augmentation done, e.g., we rotate by a maximum of \( 9^\circ \) left/right and translate by a maximum of 5% of the image width/height. This is to prevent augmentations leaking to our generator as we do not use adaptive augmentation training [9]. Following the Pytorch implementation, we use the following parameters: rotate=0.5, rotate_max=0.5, xint=0.5, xint_max=0.5, scale=.5, scale_std=.05, aniso=0.5, aniso_std=.05, xfrac=.5, xfrac_std=.05, brightness=.5, brightness_std=.05, contrast=.5, contrast_std=.1, hue=.5, hue_max=.05, saturation=.5, saturation_std=.5, imgfilter=.5, imgfilter_std=1.

### III. DeepFashion CSE-Dataset

The DeepFashion CSE-dataset is derived from the Inshop Clothes Retrieval Benchmark of DeepFashion [16], where we have annotated each image with a CSE embedding. Each image is automatically annotated with a pre-trained model from detr2on2 [27], specifically R-101 PPN-DL-s6. For each image, we select the highest scoring detected instance. We remove any image that has no detections with a confidence score larger than 0.8. The filtered dataset results in 40,625 training images and 10,275 validation images, where each image is downsamled to \( 384 \times 256 \) using bilinear sampling. We will include the train/val split that we randomly selected.

### II. Face Quantitative Evaluation

The quantitative evaluation of the face region was done by cropping the face region, upsampling the region to \((299, 299)\) and compute FID for every training sample in the dataset. The face region was detected with DSFD [14] using an open source Pytorch implementation [6] where the highest scoring and largest face region was extracted.

#### DeepFashion Decoder-Only Architecture

The decoder-only generator is similar to the generator of StyleGAN2 [11]. The generator consists of 6 residual blocks with two \( 3 \times 3 \) convolutions each, where each block is followed by bilinear upsampling. In total, the generator has 43.5 M parameters, where each residual block has the following number of output channels: \([768, 768, 384, 192, 96, 48]\). The starting resolution is \( 12 \times 8 \) and the output is \( 384 \times 256 \).

We use the same discriminator as for COCO-Body, except that we scale the number of parameters to 42.4M. Otherwise, we use identical hyperparameters as for the COCO-Body dataset.
B. COCO Anonymization Details

The anonymization framework consists of two stages, detection and generation.

The detection network is a pre-trained CSE \[19\] network \[1\] from detectron2 \[27\]. We detect pixel-to-surface correspondences on the entire image (can contain several identities), then crop every person out. For each instance detected, we crop by first finding the minimum bounding box that contains the detected surface, then expand this surface to a rectangular shape that has a similar aspect ratio of the target resolution \((288 \times 160)\). In addition, we ensure that 30% of the resulting crop contains "background" (i.e., not the surface). We zero-pad the image if there is no possible crop that fits in the original image.

We crop the CSE-embedding similarly. The cropped image and the embedding is then resampled to \(288 \times 160\) with bilinear up/downsampling. The segmentation map \(S\) outputted by CSE is 1 for every pixel belonging to the surface. We equally resample \(S\) and dilate \(S\) depending on the number of pixels in the image that belongs to the surface. This is to ensure that we remove clothing and other accessories on the human body, as CSE primarily detects the body. Then, we zero-out all pixels that are belongs to the surface, indicated by the dilated segmentation \(S\). The resulting partial image is the input of our surface-guided generator.

The final image is naively stitched together. For each instance, the generated image is resampled to the original resolution, then all pixels in the original image belonging to the dilated surface is replaced by with the generated ones.

I. COCO-Body Dataset

We generate the COCO-Body dataset by using the detection procedure described above on COCO \[15\] train2014 and val2014. Specifically, for every image in the COCO dataset, we detect instances in the image. For each instance, we find the cropped image (described above) and include the image in the dataset if either; (1) the detected surface has more than 80% geodisc point similarity \[4\] to the ground truth DensePose-CSE dataset \[19\], or (2) the instance has a confidence score higher than 99.5%. Out of these detections, we filter out all images that are not in the aspect ratio range \([0.4, 4]\) (height/width), images that cover an area smaller than \(144 \times 80\) pixels, or images that contains more than 25% zero-padding (in image area).

After filtering the detections, we are left with 43,053 images from COCO train2014, and 10,777 in val2014. We use this train/validation split for our experiments. The dataset is published open source at github.com/hukkelas/full_body_anonymization

C. Additional Quantitative Results

Table \[1\] includes all metrics for every model that is reported in the main paper.

I. Truncation Trick

The truncation trick is an established technique to sample latent vectors from a truncated distribution to improve generated image quality at the cost of diversity of samples \[1\]. A similar strategy is adopted for the intermediate latent space of StyleGAN \[10\], which is shown to be more effective than truncating in \(z\) \[13\].

We adopt a similar strategy \[10\] \[13\] for the conditional latent space of V-SAM. We find the mass of center; \(\bar{\omega}_i = \mathbb{E}_{z \sim P(z)} [f_\omega (\omega_i, z)]\). Then, we evaluate the effect of interpolation and clamping the latent by the following approaches:

- **Interpolate** \(\omega\): \(\bar{\omega} + t_i \cdot (\omega - \bar{\omega})\),
- **Clamp** \(\omega\): \(\text{clamp}(\omega, \bar{\omega} - t_c, \bar{\omega} + t_c)\),
- **Interpolate** \(z\): \(t_i \cdot z\),
- **Clamp** \(z\): \(\text{clamp}(z, -t_c, t_c)\)

where \(t_i \in \mathbb{R}, t_i \in [0, 1]\), and \(\text{clamp}(x, x_{\text{lower}}, x_{\text{upper}})\) sets all elements of \(x\) that are outside the range of \([x_{\text{lower}}, x_{\text{upper}}]\) to \(x_{\text{lower}}\) or \(x_{\text{upper}}\).

We observe that truncating the intermediate latent space \(\omega\) yields a better tradeoff between quality and diversity, compared to truncating in \(z\) \[1\].

Interpolating in \(\omega\) The disentangled representation of V-SAM allows for natural transitions between different synthesized styles. Figure \[2\] reflects that linearly interpolating between two randomly sampled \(\omega\) yields a natural transition. Observe that the interpolation path is smooth, where different features of the synthesized person is gradually changed. Note that we observe a discontinuous transition (e.g. Row 5 in Figure \[2\]) for some combinations of conditional pose and sampled \(\omega\). We speculate that a deeper mapping network can further improve the disentanglement and mitigate this non-smooth transition.

II. Effect of Anonymization for Computer Vision

Anonymizing Data for Evaluation (PASCAL VOC)

Following the same approach as for COCO (described in the main paper), we show that surface-guided anonymization strongly improves over traditional techniques for evaluation purposes (Table \[2\]). Furthermore, we note a slight gap between surface-guided anonymization and the original dataset (50.3 vs 51.8 AP). We believe this originates primarily from errors in detection, as we use a confidence threshold of 0.1.

\[1\] Specifically, R-101-FPN-DL-s1x
Figure 1: Analysis of truncation techniques for V-SAM. (a) Compares different truncation strategies applied to the intermediate latent space $\omega$ or $z$. Truncation in $\omega$ yields a better trade off between diversity and image quality. (b) Evaluates the effect of $\omega$ interpolation, where we interpolate a sampled latent towards $\bar{\omega}$. Increasing the truncation (t) gradually improves image quality at the cost of sample diversity.

<table>
<thead>
<tr>
<th>Validation Dataset</th>
<th>AP$_{50:95}$ $\uparrow$</th>
<th>AP$_{50}$ $\uparrow$</th>
<th>AP$_{75}$ $\uparrow$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>51.8</td>
<td>80.3</td>
<td>56.5</td>
</tr>
<tr>
<td>Mask Out</td>
<td>47.9</td>
<td>75.2</td>
<td>51.4</td>
</tr>
<tr>
<td>8 × 8 Pixelation</td>
<td>47.2</td>
<td>75.1</td>
<td>50.8</td>
</tr>
<tr>
<td>16 × 16 Pixelation</td>
<td>48.5</td>
<td>76.7</td>
<td>52.0</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>50.3</strong></td>
<td><strong>78.9</strong></td>
<td><strong>54.2</strong></td>
</tr>
</tbody>
</table>

Table 2: Object Detection AP on the PASCAL VOC validation set \cite{Everingham:2010}. The results are from a **pre-trained** Faster R-CNN \cite{Ren:2015} R50-C4 from detectron2 \cite{Wu:2019}.

D. Additional Qualitative Results

I. COCO-Body

Random examples from the COCO-Body dataset are given in Figure 3, Figure 4, Figure 5, and Figure 6.

II. DeepFashion

Random examples from the DeepFashion-CSE dataset are given in Figure 7, Figure 8, and Figure 9.

III. COCO

Random examples from the COCO dataset are given in Figure 10, Figure 11, Figure 12, Figure 13, Figure 14 and Figure 15.

Anonymizing data for Training

Surface-guided anonymization slightly improves over traditional anonymization when using the anonymized data for training (Table 3, Table 4). Our naive anonymization of COCO without modifying object labels (e.g. not removing the class "tie") introduces ambiguities in the training objective, which we believe significantly degrades training performance. This issue is not as severe for pixelation, as these objects are still present in the image. Furthermore, we note that the CSE detector has several false positives and negatives which directly impacts the training objective. For example, false positives yields highly corrupted images with surface-guided anonymization.
Table 3: Instance segmentation mask AP on the COCO validation set [15]. The results are from a Mask R-CNN [5] R50-FPN-3x from detectron2 [27] trained on different anonymized datasets. The validation set is not anonymized.

<table>
<thead>
<tr>
<th>Training Dataset</th>
<th>AP_{50:95}↑</th>
<th>AP_{50}↑</th>
<th>AP_{75}↑</th>
<th>AP_{m}↑</th>
<th>AP_{r}↑</th>
<th>AP_{person}↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>37.1</td>
<td>58.7</td>
<td>39.8</td>
<td>18.8</td>
<td>39.6</td>
<td>53.3</td>
</tr>
<tr>
<td>Mask Out</td>
<td>34.6</td>
<td>55.2</td>
<td>36.7</td>
<td>16.7</td>
<td>36.5</td>
<td>50.0</td>
</tr>
<tr>
<td>8 × 8 Pixelation</td>
<td>34.8</td>
<td>55.7</td>
<td>37.0</td>
<td>16.9</td>
<td>36.7</td>
<td>50.3</td>
</tr>
<tr>
<td>16 × 16 Pixelation</td>
<td><strong>35.0</strong></td>
<td><strong>55.9</strong></td>
<td><strong>37.3</strong></td>
<td>17.3</td>
<td>37.1</td>
<td><strong>50.4</strong></td>
</tr>
<tr>
<td>Ours</td>
<td><strong>35.0</strong></td>
<td><strong>56.0</strong></td>
<td><strong>37.5</strong></td>
<td><strong>17.7</strong></td>
<td><strong>36.7</strong></td>
<td><strong>50.6</strong></td>
</tr>
</tbody>
</table>

Table 4: Object Detection AP on the PASCAL VOC validation set [2]. The results are from a Faster R-CNN [23] R50-C4 from detectron2 [27] trained on different anonymized datasets. The validation set is not anonymized.

<table>
<thead>
<tr>
<th>Training Dataset</th>
<th>AP_{50:95}↑</th>
<th>AP_{50}↑</th>
<th>AP_{75}↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td><strong>51.3</strong></td>
<td><strong>80.1</strong></td>
<td><strong>56.0</strong></td>
</tr>
<tr>
<td>Mask Out</td>
<td>51.8</td>
<td>80.0</td>
<td>55.7</td>
</tr>
<tr>
<td>8 × 8 Pixelation</td>
<td>51.2</td>
<td>79.9</td>
<td>55.03</td>
</tr>
<tr>
<td>16 × 16 Pixelation</td>
<td>51.2</td>
<td>79.8</td>
<td>55.7</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>51.3</strong></td>
<td><strong>80.1</strong></td>
<td><strong>56.0</strong></td>
</tr>
</tbody>
</table>

References


Figure 3: Diverse synthesized images from COCO-Body. We truncate the normal of SPADE and Config D, and show diverse results for Config E. Note that the images are randomly sampled and the latent variable for (d-g) is randomly sampled with no latent truncation.
Figure 4: Diverse synthesized images from COCO-Body. We truncate the normal of SPADE and Config D, and show diverse results for Config E. Note that the images are randomly sampled and the latent variable for (d-g) is randomly sampled with no latent truncation.
Figure 5: Diverse synthesized images from COCO-Body. We truncate the normal of SPADE and Config D, and show diverse results for Config E. Note that the images are randomly sampled and the latent variable for (d-g) is randomly sampled with no latent truncation.
Figure 6: Diverse synthesized images from COCO-Body. We truncate the normal of SPADE and Config D, and show diverse results for Config E. Note that the images are randomly sampled and the latent variable for (d-g) is randomly sampled with no latent truncation.
Figure 7: Diverse synthesized images from DeepFashion-CSE. (b) shows generated results with a truncated latent, and all other columns use no truncation. Each column is synthesized with the same latent variable $z$. All images and latent variables are randomly sampled. Zoom in for details.
Figure 8: Diverse synthesized images from DeepFashion-CSE. (b) shows generated results with a truncated latent, and all other columns use no truncation. Each column is synthesized with the same latent variable \( z \). All images and latent variables are randomly sampled. Zoom in for details.
Figure 9: Diverse synthesized images from DeepFashion-CSE. (b) shows generated results with a truncated latent, and all other columns use no truncation. Each column is synthesized with the same latent variable $z$. All images and latent variables are randomly sampled. Zoom in for details.
Figure 10: Randomly sampled images from the anonymized COCO [15] validation dataset. Each image is sampled with latent truncation (t=0), see Appendix C for details. Zoom in for details.
Figure 11: Randomly sampled images from the anonymized COCO [15] validation dataset. Each image is sampled with latent truncation (t=0), see Appendix C for details. Zoom in for details.
Figure 12: Randomly sampled images from the anonymized COCO [15] validation dataset. Each image is sampled with latent truncation (t=0), see Appendix C for details. Zoom in for details.
Figure 13: Randomly sampled images from the anonymized COCO [15] validation dataset. Each image is sampled with latent truncation (t=0), see Appendix C for details. Zoom in for details.
Figure 14: Randomly sampled images from the anonymized COCO [15] validation dataset. Each image is sampled with latent truncation ($t=0$), see Appendix C for details. Zoom in for details.
Figure 15: Randomly sampled images from the anonymized COCO [15] validation dataset. Each image is sampled with latent truncation (t=0), see Appendix C for details. Zoom in for details.