# Supplementary Material
# SCAM! Transferring humans between images with Semantic Cross Attention Modulation

Nicolas Dufour[1,2] [0000−0002−1903−5110], David Picard[1] [0000−0002−6296−4222], and Vicky Kalogeiton[2] [0000−0002−7368−6993]

[1] LIGM, Ecole des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France
{nicolas.dufour, david.picard}@enpc.fr
[2] LIX, CNRS, Ecole Polytechnique, IP Paris
vicky.kalogeiton@lix.polytechnique.fr

In this supplementary material, we first discuss the societal and environmental impact of SCAM (Section 1). Then, we give more details of SCAM and additional experimental analysis (Sections 2-3).

## 1 Discussion

### 1.1 Societal impact

This work could allow for multiple applications such as editing images, where we can easily exchange someone by someone else. This can have a negative impact if used with bad intentions. Indeed, one could use our method to create fake news. As of today, deep fakes can still be detected, by either humans or algorithms. However, as research progress in these fields, detection is going to get more complicated and regulation will have to control this.

Our method could also be very useful to create movies. It could allow changing an actor by another one. This could be very helpful for finishing a movie when an actor is impaired. It could also reduce the producing cost when using superstars, where an actor could accept for his image to be used, while not acting physically himself. This however could pose some legal problems since it is easy to create content without permission of the subject.

### 1.2 Environmental impact

For this project, we used 42.3 thousands GPUs hours. The experiments were done on a GPU cluster. The GPUs used are Nvidia V100-32g. Our experiments used 80% of the GPUs maximum power of 250Wh, which amounts to 10.6 MWh of energy used for the whole project. The cluster we used is the Jean Zay cluster, situated in France. France heavily relies on nuclear energy, having a greener energy than average, with 50-80g CO2 for each kWh produced. Considering only the CO2 for the production of the electricity used, this results in 528-846kg of CO2 emitted for this project. Training a single SCAM model takes around 50 GPUs hours, which amounts to 10kWh and 500-800 g of CO2 emitted. As a comparison, the world average per capita CO2 emission is 4.7 ton/year.

### 1.3 Future Work

To improve upon this method, we see three main directions. First, this project is intended for images. For it to work well on videos, we would need to add some temporal information. Adding temporal consistency and smoothing should help have more visually pleasing outputs. Second, another improvement would be to directly train on transferred images. In this project, we train on the reconstructed images, and then we infer subject transfer. We conjecture that training on the transferred images would yield better results. Third, we observe that the subject transfer task is complicated by the quality of the segmentation masks and the disparity between the pose reference subject segmentation and the style reference subject. To prevent this, we could learn a network to map the pose reference subject segmentation to the wanted pose.

## 2 About SCAM

### 2.1 Implementation details

We train all models for 50k steps, with a batch size of 32 on 4 Nvidia V100 GPUs. We use AdamW [4] with a learning rate of 0.0001 for the generator and the encoder, and 0.0004 for the discriminator with $\beta$=$(0.9, 0.999)$. We set $k$=8 latents per label of dimension $d$=256. Each time we use attention on a feature map, we first encode the image using 2D sinusoidal positional encodings, as attention acts on sets that do not include positional information. Note that adding positional encoding to the latents is not useful because latents can be seen as a set and not a sequence. Moreover, since the latents are going to be initialized from a learned vector, this learned vector can incorporate the positional information, if needed, and this information will propagate throughout the architecture. In the discriminator, we use GradNorm [6] to stabilize the training. For the SAT-Encoder, we have $L_E = 6$ SAT-Blocks. In the SAT-Generator, we have $L_G = 7$ SCAM-Blocks.

### 2.2 Details on SAT

Here are the detailed implementations for SAT-Operation (Cross (See Algorithm1) and Self (See Algorithm 2)) and for the SAT Block (See Algorithm 3). For even finer details (Hyperparameters, practical details) see the provided Pytorch code.

### 2.3 Details on SCAM

Here are the detailed implementations for the SCAM Block (See Algorithm 4). For even finer details (Hyperparameters, practical details) see the provided Pytorch code.

**Algorithm 1:** SAT-Cross at layer i

**Data:** $Z_i \in \mathbb{R}^{m \times d}$;$X_i \in \mathbb{R}^{n \times C_i}$;$S \in \mathbb{1}^{n \times m}$
**Result:** $\tilde{Z}_i \in \mathbb{R}^{m \times d}$
$Z'_i \leftarrow SCA(Z_i, X_i, S)$;
$Z'_i \leftarrow LN(Z'_i + Z_i)$;
$\tilde{Z}_i \leftarrow FFN(Z'_i)$;
$\tilde{Z}_i \leftarrow LN(\tilde{Z}_i + Z'_i)$;

**Algorithm 2:** SAT-Self at layer i

**Data:** $\tilde{Z}_i \in \mathbb{R}^{m \times d}$;$M \in \mathbb{1}^{m \times m}$
**Result:** $Z_{i+1} \in \mathbb{R}^{m \times d}$
$Z''_i \leftarrow SCA(\tilde{Z}_i, \tilde{Z}_i, M)$;
$Z''_i \leftarrow LN(Z''_i + \tilde{Z}_i)$;
$Z_{i+1} \leftarrow FFN(Z''_i)$;
$Z_{i+1} \leftarrow LN(Z_{i+1} + Z''_i)$;

**Algorithm 3:** SAT-Block at layer i

**Data:** $X_i \in \mathbb{R}^{H_i \times W_i \times C_i}$;$Z_i \in \mathbb{R}^{m \times d}$;$S \in \mathbb{1}^{H_i \times W_i \times m}$
**Result:** $Z_{i+1} \in \mathbb{R}^{m \times d}$; $X_{i+1} \in \mathbb{R}^{H_{i+1} \times W_{i+1} \times C_{i+}}$
$X_i^{flat} \leftarrow \text{Flatten}(X_i, dim = [0, 1])$;
$S_i = \text{Downsample}(S, shape = (H_i, W_i, m))$;
$S_i^{flat} \leftarrow \text{Flatten}(S, dim = [0, 1])$;
$\tilde{Z}_i \leftarrow \text{SAT-Cross}(Z_i, X_i^{flat}, S_i^{flat})$;
$Z_{i+1} \leftarrow \text{SAT-Self}(\tilde{Z}_i, \tilde{Z}_i, M)$;
$X_{i+1} \leftarrow g(X_i)$ ; `/* g a strided convolution */`

**Algorithm 4:** SCAM-Block at layer j

**Data:** $X'_j \in \mathbb{R}^{H_j \times W_j \times C_i}$;$Y'_j \in \mathbb{R}^{H_j \times W_j \times 3}$; $Z'_j \in \mathbb{R}^{m \times d}$;$S \in \mathbb{1}^{H_j \times W_j \times m}$
**Result:** $X'_{j+1} \in \mathbb{R}^{H_{j+1} \times W_{j+1} \times C_{j+1}}$;$Y'_{j+1} \in \mathbb{R}^{H_{j+1} \times W_{j+1} \times 3}$; $Z'_j \in \mathbb{R}^{m \times d}$
$X'_j, Z'_j \leftarrow SCAM(X'_j, Z'_j, S)$;
$X'_j \leftarrow \text{Upsample}(X'_j, upsize = 2)$;
$X'_{j+1}, Z'_{j+1} \leftarrow SCAM(X'_j, Z'_j, S)$;
$X'_{j,RGB}, \leftarrow SCAM(X'_{j+1}, Z'_{j+1}, S)$;
$Y'_j \leftarrow \text{Upsample}(Y'_j, upsize = 2)$;
$Y'_{j+1} \leftarrow Y'_j + X'_{j,RGB}$

### 2.4 Losses

Here, we complement Section 3.4 in the main paper and describe with more details the losses used for training SCAM. We denote by G the SCAM-Generator, E the SAT-Encoder and D the PatchGAN discriminator.
For the **GAN loss**, we use Hinge GAN loss [3] as follows:

$$\begin{aligned}\mathcal{L}_{\mathrm{D,GAN}} &= \mathbb{E}\left[\max(0, 1 - D(X, S))\right] \\ &+ \mathbb{E}\left[\max(0, D(G(E(X, S), S), S) + 1)\right]\end{aligned} \tag{1}$$

$$\mathcal{L}_{\mathrm{F,GAN}} = \mathbb{E}\left[D(G(E(X, S), S), S)\right] \quad . \tag{2}$$

For the **reconstruction loss**, we use the perceptual loss as in [7]. This uses a pretrained VGG network, and tries to make the intermediate feature maps between the input and the reconstructed input as close as possible. It is defined as:

$$\mathcal{L}_{\mathrm{VGG}} = \mathbb{E}\left[\sum_{i=1}^{L} \|F_i(X) - F_i(G(E(X, S)))\|_1\right] \quad , \tag{3}$$

with $L$ the number of VGG hidden layers and $F_i$ the $i^{th}$ layer feature map of the VGG feature map.

We also use a $\mathcal{L}_1$ loss between the input and the reconstruction.

$$\mathcal{L}_1 = \mathbb{E}\left[\|X - G(E(X, S)))\|_1\right] \quad , \tag{4}$$

### 2.5 Metrics

Here we give more details on the metrics we introduce: REIDSim and REIDAcc. Let's call $I^S$ the subject images, $I^B$ the background images and $I^{ST}$ the subject transfer images. Note that the subject images/background images couples are fixed to be able to compare on the same Now if REID is the REID network introduced in [1], we now have $X^S = REID(I^S)$, $X^B = REID(I^B)$ and $X^B = REID(I^B)$ the respective embeddings. Now we can compute the REID metric:

$$REIDSim(I^S, I^{ST}) = \frac{1}{n}\sum_{k=1}^{n} \frac{(X_k^S)^T X_k^{ST}}{\|X_k^S\|\|X_k^{ST}\|} \tag{5}$$

$$REIDAcc(I^S, I^B, I^{ST}) = \frac{1}{n}\sum_{k=1}^{n} \mathbb{1}\left(\frac{(X_k^S)^T X_k^{ST}}{\|X_k^S\|\|X_k^{ST}\|} > \frac{(X_k^B)^T X_k^{ST}}{\|X_k^B\|\|X_k^{ST}\|}\right) \tag{6}$$

With $n$ the number of subject transfer images we want to evaluate.

| Method | # parameters ↓ | Training speed ↓ | Inference speed ↓ |
|---|---|---|---|
| SPADE | 109M | 95ms | 39ms |
| CLADE | 84M | **86ms** | **38ms** |
| SEAN-CLADE | 240M | 384ms | 83ms |
| SEAN | 265M | 280ms | 92ms |
| INADE | 85M | 179ms | 50ms |
| SCAM | **95M** | 180ms | 61ms |

Table 1: **Comparison** of model characteristics. Speeds are given in ms/samples. We evaluate on CelebAMask-HQ[2] 20 semantic labels. SCAM has $k = 8$.

### 2.6 Model characteristics

Thanks to the SCAM-Block, SCAM results in 2x fewer number of parameters compared to SEAN (i.e., 95M vs 265M). SEAN is bigger than our model, as it comprises a big FFN for each style code at each block. Instead, with the SCAM-Block, we have parameter sharing for every token in SCAM blocks, removing this constraint. Moreover, SCAM trains faster and can infer values 50% faster than SEAN. Measurements are made on a single NVIDIA RTX 3090.

## 3 Additional experimental analysis

In this section, we display complementary experiments on the task of pose transfer. We also showcase some qualitative ablations of SCAM.

### 3.1 Pose Transfer

Here, we illustrate qualitative results for the pose transfer task for SCAM on the iDesigner [5] dataset. Given two images, one being the style image $X_{Style}$ and another of $X_{Pose}$ being the pose reference. $S_{Pose}$ the associated segmentation mask. The goal of pose transfer is to generate an image matching the style of $X_{Style}$ with the semantics of $X_{Pose}$. We extract the latent codes $Z_{Style}$ with the SAT-Encoder. We then can generate $X_{PT} = G(Z_{Style}, S_{Pose})$ the generated image that share the style of $X_{Style}$ and the pose of $X_{Pose}$.

In Figure 1, we observe how SPADE, SEAN, SEAN++ and SCAM perform pose transfer. In this figure, we observe similar trends that we observe with subject transfer.

SCAM captures more details than the competing methods. This is particularly visible on the backgrounds of images a) b) and d), where the backgrounds on SCAM display more coherent outputs with the style reference image than other methods.

Similarly, we can observe that SCAM presents better subject reconstruction like in a), where given the segmentation map, SCAM generates a coherent subject. The skirt gets converted into pants because of the segmentation masks, but

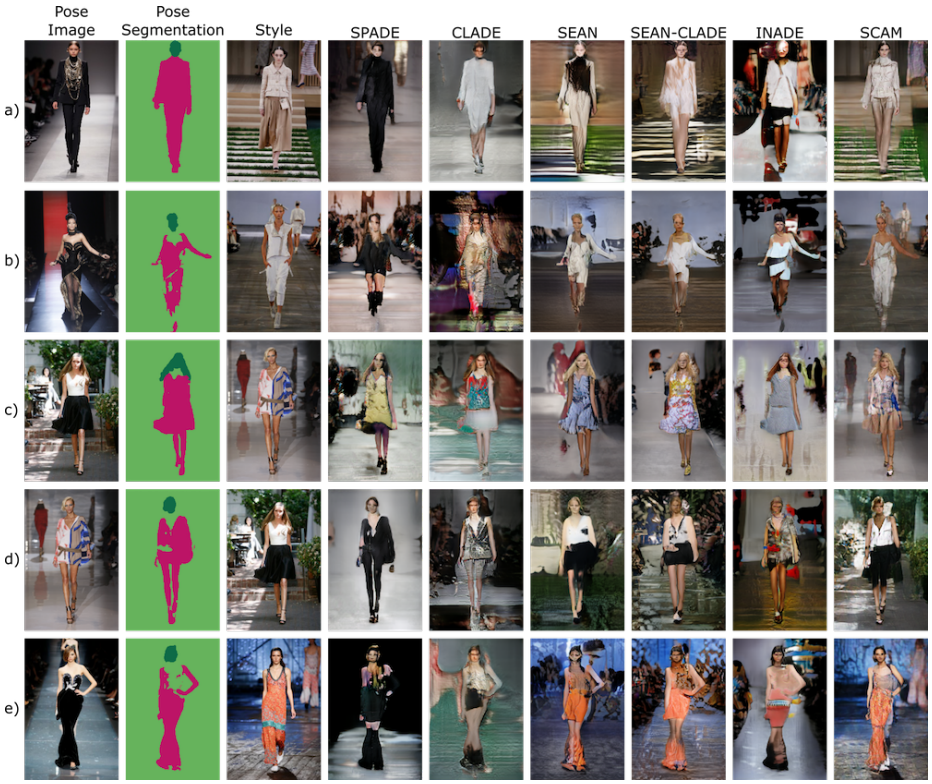


Fig. 1: **Pose Transfer in idesigner**. We compare SCAM to competing methods on the pose transfer task.

the texture remains the one of the style image. For SEAN and SEAN-CLADE, we do not see it generating coherent clothes that match the style of the style image.

### 3.2 Ablations

In Figure 2 we showcase some qualitative ablations to complement the main paper quantitative ablations. This additional visual ablation help us understand what each block of SCAM brings perceptually.

We observe that when removing the convolutions (fourth column) in the SAT encoder, we end-up with a very simple texture that lacks details. This is particularly true in row c), where the dress texture is very repetitive and simplistic. This confirms our hypothesis that using information at multiple resolutions gives higher quality encodings.

If we remove self attention in SAT (fifth column), we observe that the reconstruction lack details in the coarser label, such as the background. For example, in row d), we can observe that the 2 person in the background features the same

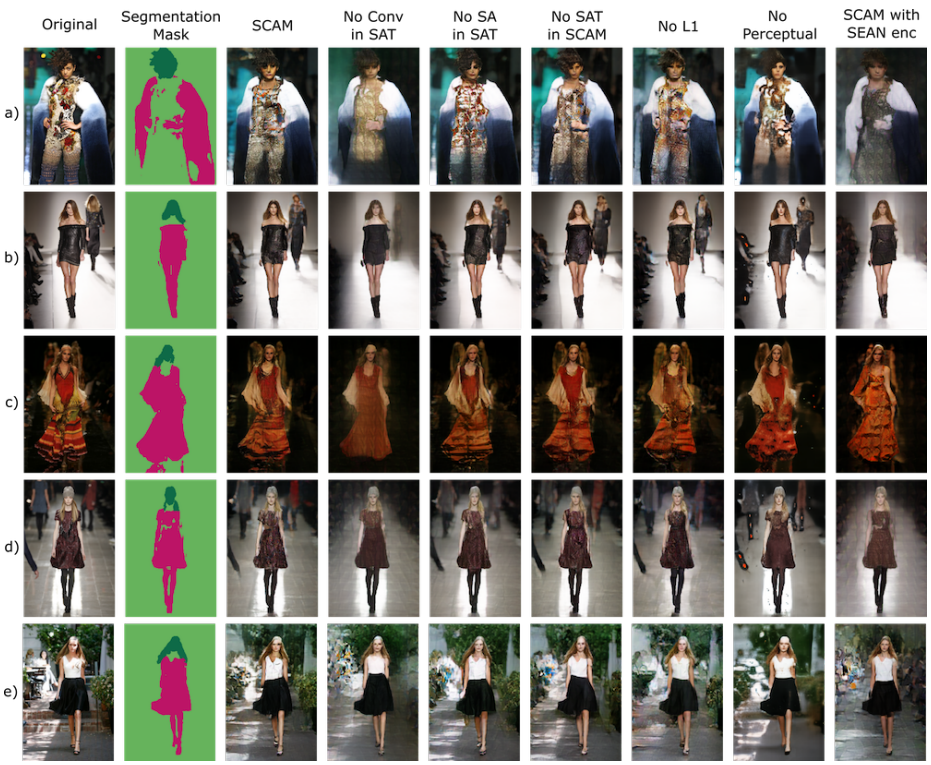


Fig. 2: **Qualitative ablations on idesigner**. We perform the same ablations as in the quantitative ablations section of the main paper.

cloths colour where in SCAM we can see 2 different colours. Allowing latents from the same region interacting together in the encoder enables refining such details.

Removing the SAT in SCAM (sixth column) yields similar results. For instance, we observe in row b) that the background is less sharp than in SCAM. The background person seems to be floating, whereas SCAM has a more coherent interaction between the floor and the background person. Refining the latents in the SCAM-Generator also yields improves the semantic knowledge of SCAM.

When removing the $\mathcal{L}_1$ loss (seventh column) we have outputs similar to the ones in SCAM (third column). However, there are some colourized artefacts, like in row a) (No L1). The $\mathcal{L}_1$ loss allows removing artefacts that arise from the perceptual loss.

When removing the perceptual loss (eighth column), we see some salt and pepper noise appearing in the output, like in row b). These artefacts are linked to the $\mathcal{L}_1$ loss and are stabilized by the perceptual loss.

When replacing the SAT encoder with the SCAM encoder (eighth column), we study the benefits of our encoder compared to the SEAN encoder. We observe that the SEAN Encoder fails to capture details in the background, and hence it limits the generator to only simplistic texture generation. This is particularly the case in row b) and d) where the method totally misses the person in the background

### 3.3 User Study

We provide in Figure 3 and 4 the samples we used for the user study and the score for each individual question.

### 3.4 Additional visual results

We provide in Figure 5 and 6 additional results on CelebAMask-HQ and ADE20K.

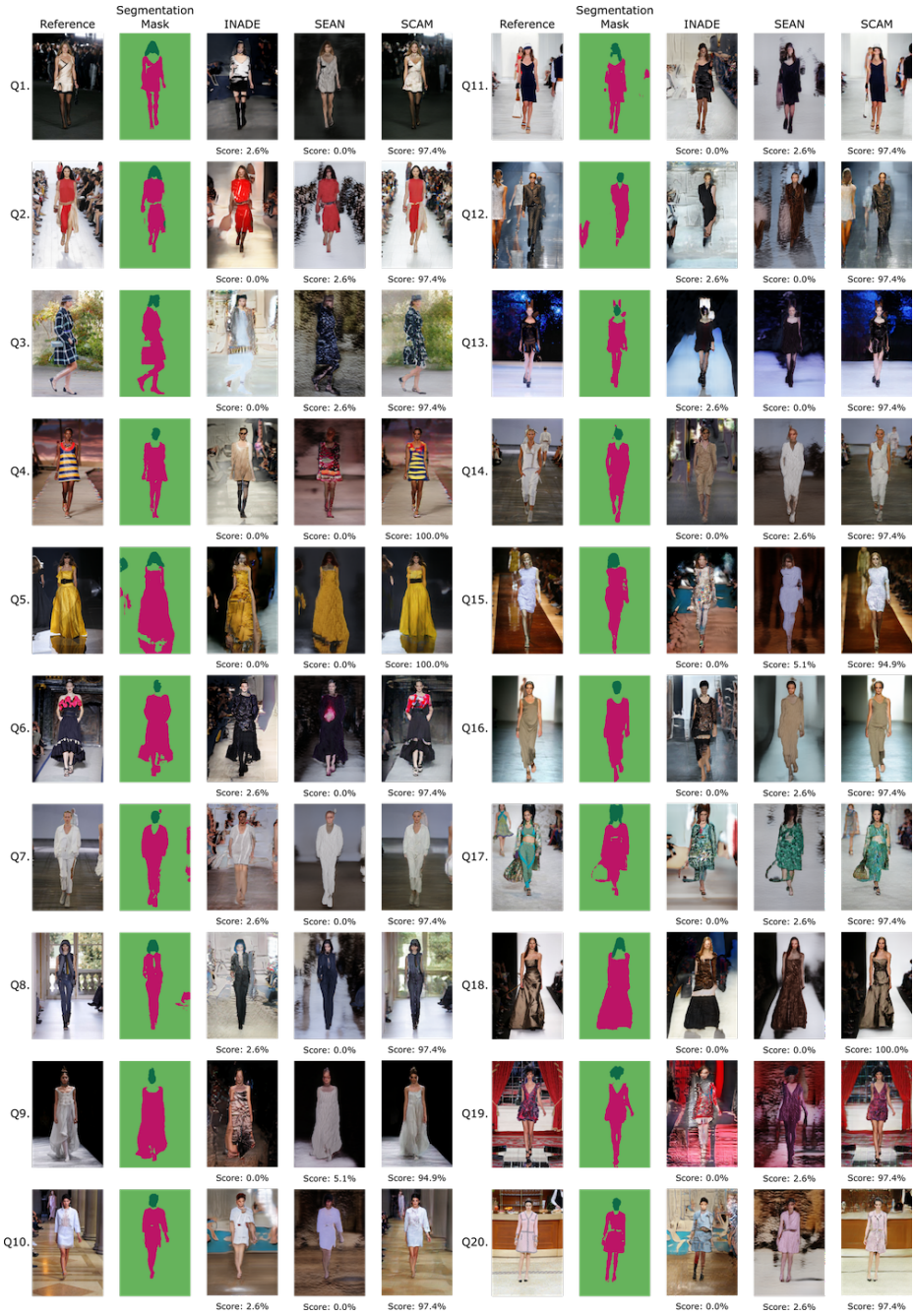


Fig. 3: **User study on reconstruction**. We provide here the samples that were used to evaluate the reconstruction quality on SCAM, SEAN and INADE in a user study. Note that we provide here the segmentation mask for reference but it was not provided to users.

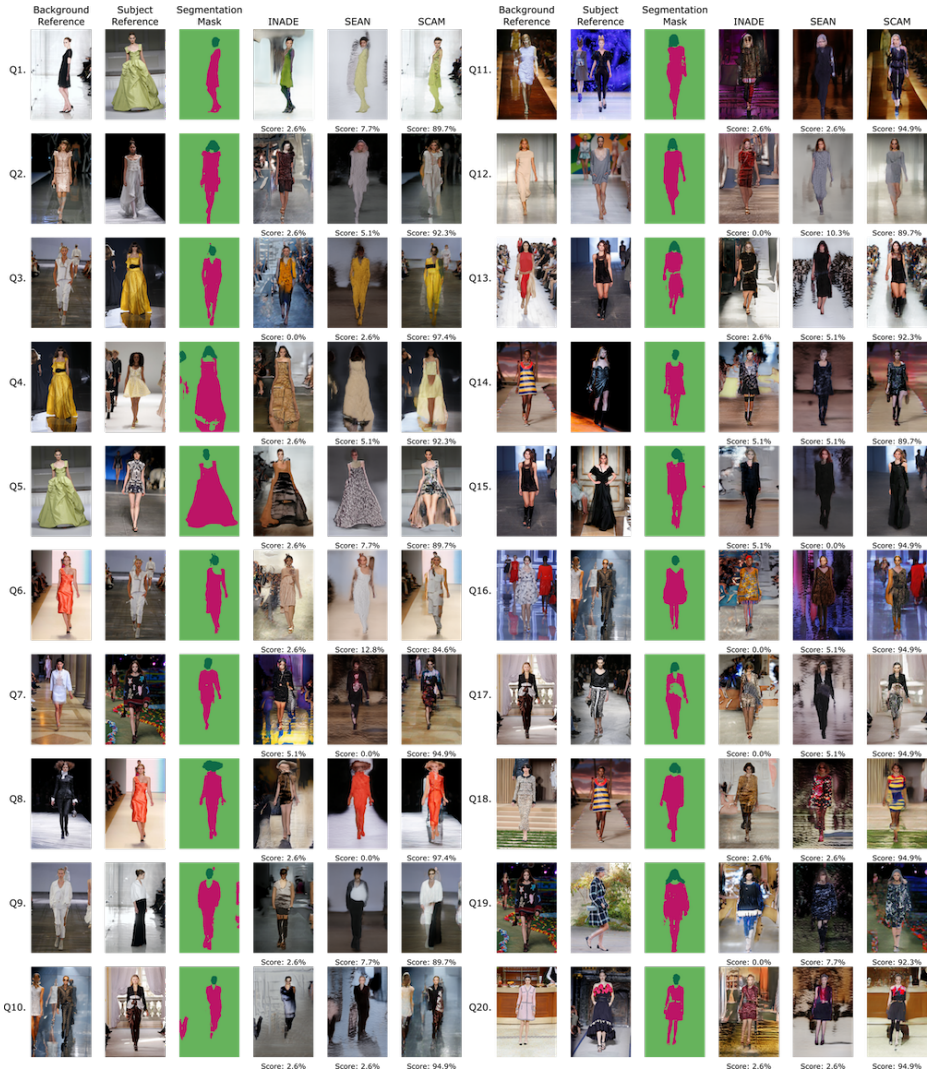


Fig. 4: **User study on subject transfer**. We provide here the samples that were used to evaluate the subject quality on SCAM, SEAN and INADE in a user study. Note that we provide here the segmentation mask for reference but it was not provided to users.

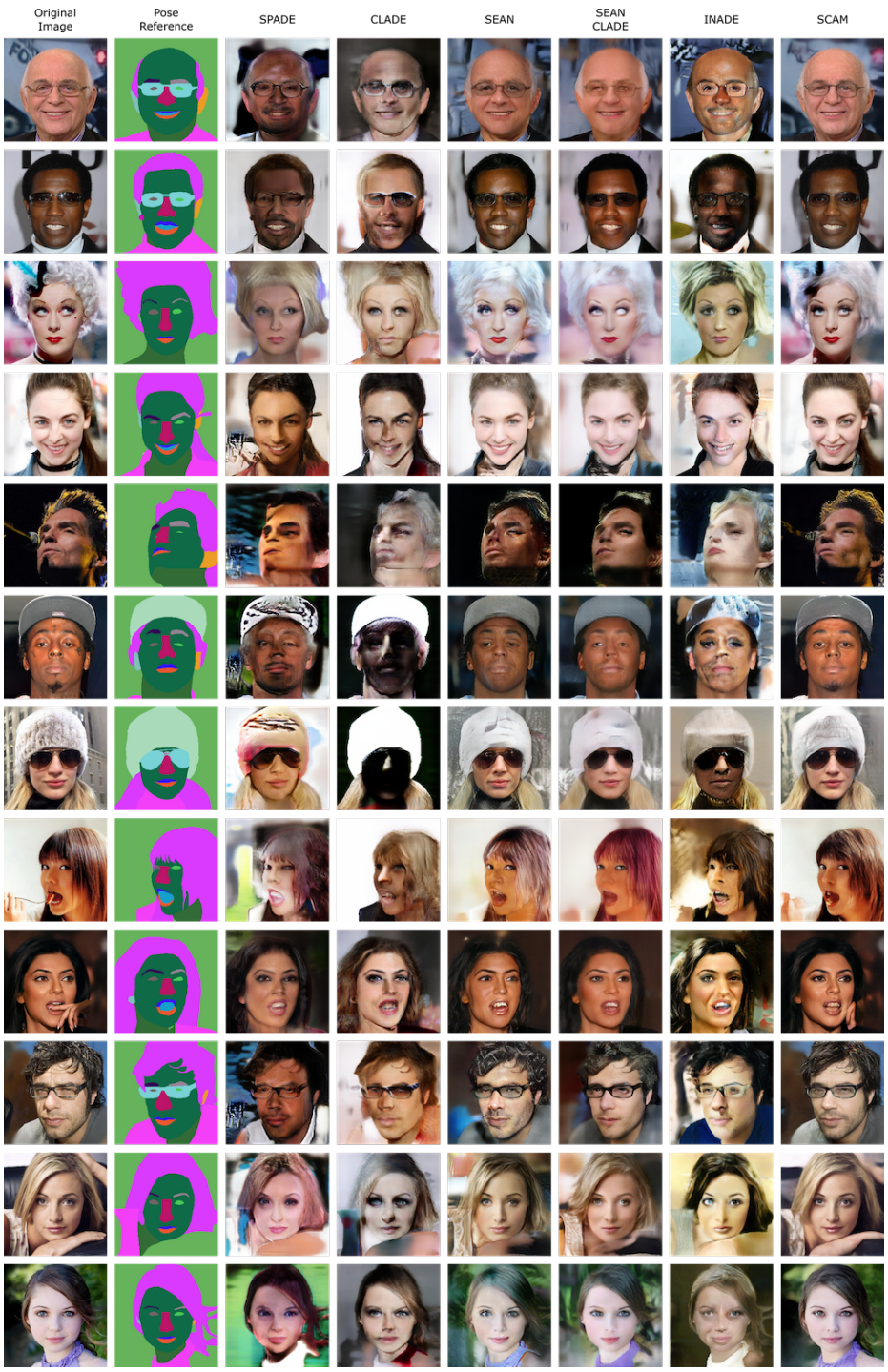


Fig. 5: **Reconstruction on CelebAMask-HQ**.


Fig. 6: **Reconstruction on ADE20K**.

## References

1. Fu, D., Chen, D., Bao, J., Yang, H., Yuan, L., Zhang, L., Li, H., Chen, D.: Unsupervised pre-training for person re-identification. In: CVPR (2021)
2. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of GANs for improved quality, stability, and variation. In: ICLR (2018)
3. Lim, J.H., Ye, J.C.: Geometric gan. In: arXiv (2017)
4. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv (2017)
5. R.J. Lehman, H.H.: idesigner 2019. In: FGVC6 (2019)
6. Wu, Y.L., Shuai, H.H., Tam, Z.R., Chiu, H.Y.: Gradient normalization for generative adversarial networks. In: arXiv (2021)
7. Zhu, P., Abdal, R., Qin, Y., Wonka, P.: Sean: Image synthesis with semantic region-adaptive normalization. In: CVPR (2020)