STGAN: A Unified Selective Transfer Network for Arbitrary Image Attribute Editing (Supplementary Material)

A. Content

The content of this supplementary material involves:

- Network structure of STGAN in Sec. B.
- Additional facial attribute editing results in Sec. C.
- Additional season translation results in Sec. D.

B. Network Structure

Our STGAN is comprised of two components, *i.e.*, a generator G and a discriminator D.

The generator G has an encoder G_{enc} for abstract latent representation, a decoder G_{dec} for target image generation, and G_{st} consists of a series of selective transfer units (STUs) for selective feature transfer.

The discriminator D has two branches D_{adv} and D_{att}. D_{adv} distinguishes whether an image is a fake image or a real one, and D_{att} predicts an attribute vector.

Fig. A shows the overall architecture of our STGAN, and Table A illustrates the details of the network layers (excluding STU, whose detailed structure and formulation have been discussed in the main text).

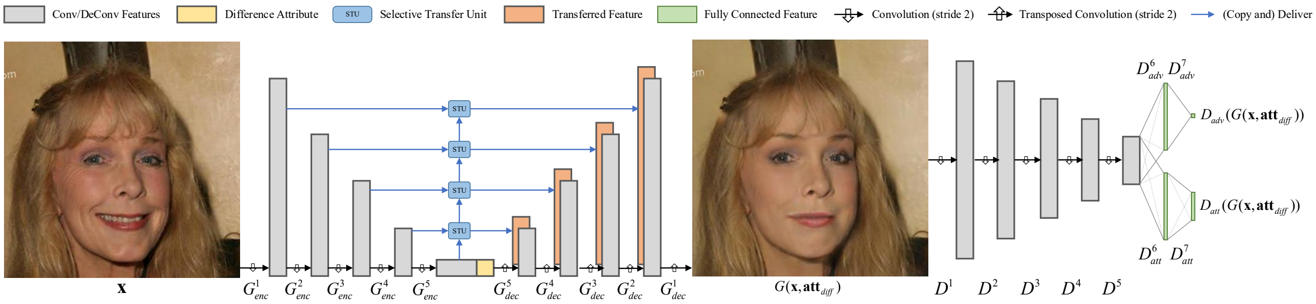

Figure A: Overall architecture of our STGAN. Taking the image above as an example, in the difference attribute vector $\mathbf{att}_{diff}$, *Young* is set to 1, *Mouth Open* is set to -1, and others are set to zeros. The outputs of D_{adv} and D_{att} are the scalar $D_{adv}(G(\mathbf{x}, \mathbf{att}_{diff}))$ and the vector $D_{att}(G(\mathbf{x}, \mathbf{att}_{diff}))$, respectively.

<table>
<tr><th>l</th><th>G^l_{enc}</th><th>G^l_{dec}</th><th>D^l_{adv}</th><th>D^l_{att}</th></tr>
<tr><td>1</td><td>Conv(d,4,2), BN, Leaky ReLU</td><td>DeConv(3,4,2), Tanh</td><td colspan="2">Conv(d,4,2), IN, Leaky ReLU</td></tr>
<tr><td>2</td><td>Conv(d*2,4,2), BN, Leaky ReLU</td><td>DeConv(d*2,4,2), BN, ReLU</td><td colspan="2">Conv(d*2,4,2), IN, Leaky ReLU</td></tr>
<tr><td>3</td><td>Conv(d*4,4,2), BN, Leaky ReLU</td><td>DeConv(d*4,4,2), BN, ReLU</td><td colspan="2">Conv(d*4,4,2), IN, Leaky ReLU</td></tr>
<tr><td>4</td><td>Conv(d*8,4,2), BN, Leaky ReLU</td><td>DeConv(d*8,4,2), BN, ReLU</td><td colspan="2">Conv(d*8,4,2), IN, Leaky ReLU</td></tr>
<tr><td>5</td><td>Conv(d*16,4,2), BN, Leaky ReLU</td><td>DeConv(d*16,4,2), BN, ReLU</td><td colspan="2">Conv(d*16,4,2), IN, Leaky ReLU</td></tr>
<tr><td>6</td><td></td><td></td><td>FC(1024), Leaky ReLU</td><td>FC(1024), Leaky ReLU</td></tr>
<tr><td>7</td><td></td><td></td><td>FC(1)</td><td>FC(c), Sigmoid</td></tr>
</table>

Table A: Architecture of STGAN components excluding G_{st}. Conv(dim, k, s) and DeConv(dim, k, s) denote the convolutional layer and transposed convolutional layer, whose output channel is dim, kernel size is k, stride is s. BN and IN represent batch normalization [3] and instance normalization [6], respectively. d is the base dimension of the network, which is set to 64 for 128 × 128 images and 32 for larger ones. c means the amount of attributes.

C. Facial Attribute Editing Results

In Fig. B, we show more images generated by all competing methods (*i.e.*, IcGAN [5], FaderNet [4], AttGAN [2], StarGAN [1] and our STGAN) with single attribute manipulated. The *Hair Color* is set to one of *Black Hair*, *Blond Hair* and *Brown Hair* distinct from the original one, and the others are modified by inversion.

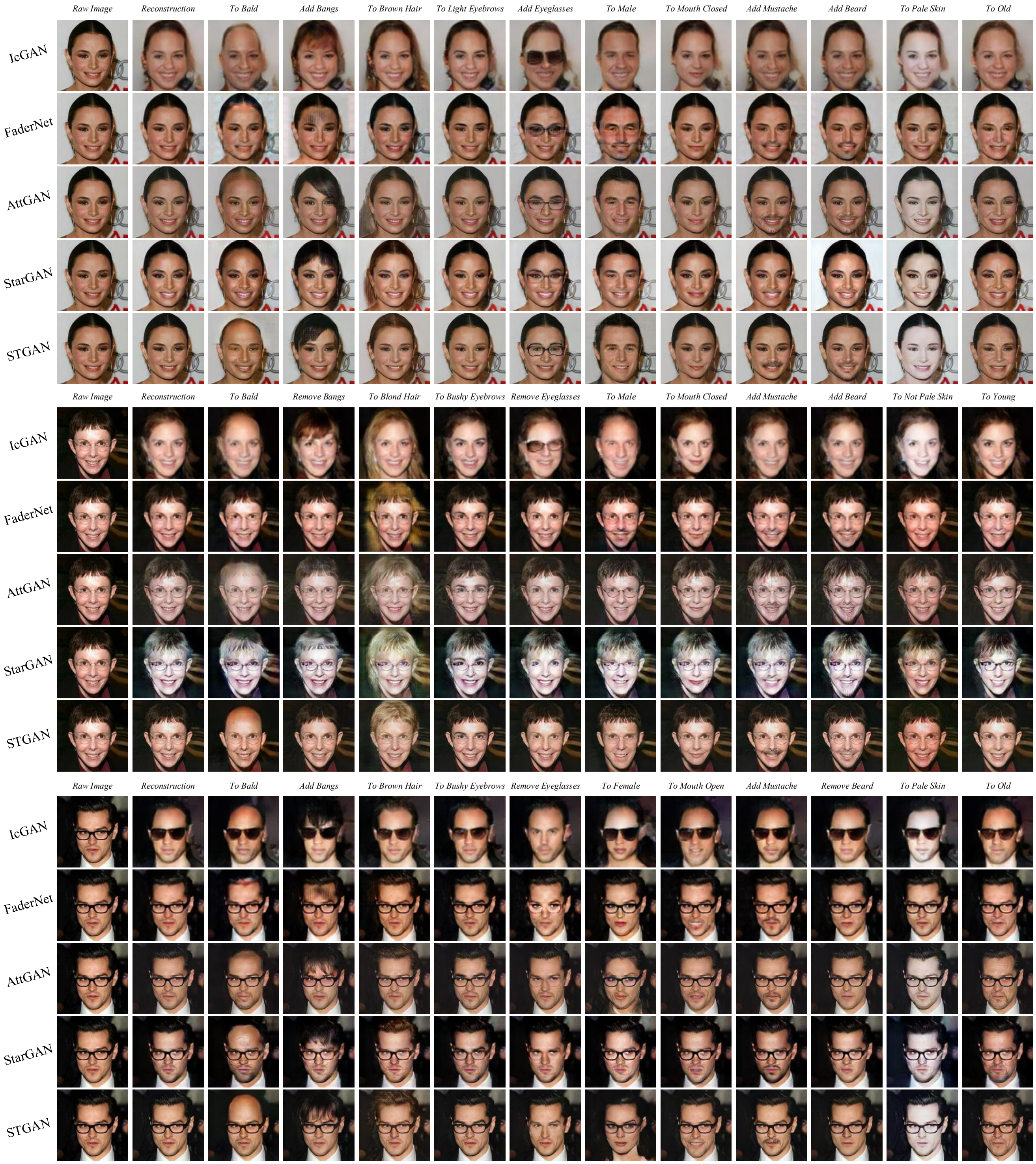

Figure B: Facial attribute editing results generated by IcGAN [5], FaderNet [4], AttGAN [2], StarGAN [1] and our STGAN. Please zoom in for better observation.

Furthermore, we show images generated by three top-performance methods (*i.e.*, AttGAN [2], StarGAN [1] and STGAN) with complex and/or multiple attributes changed in Fig. C and high resolution results of STGAN in Fig. D.

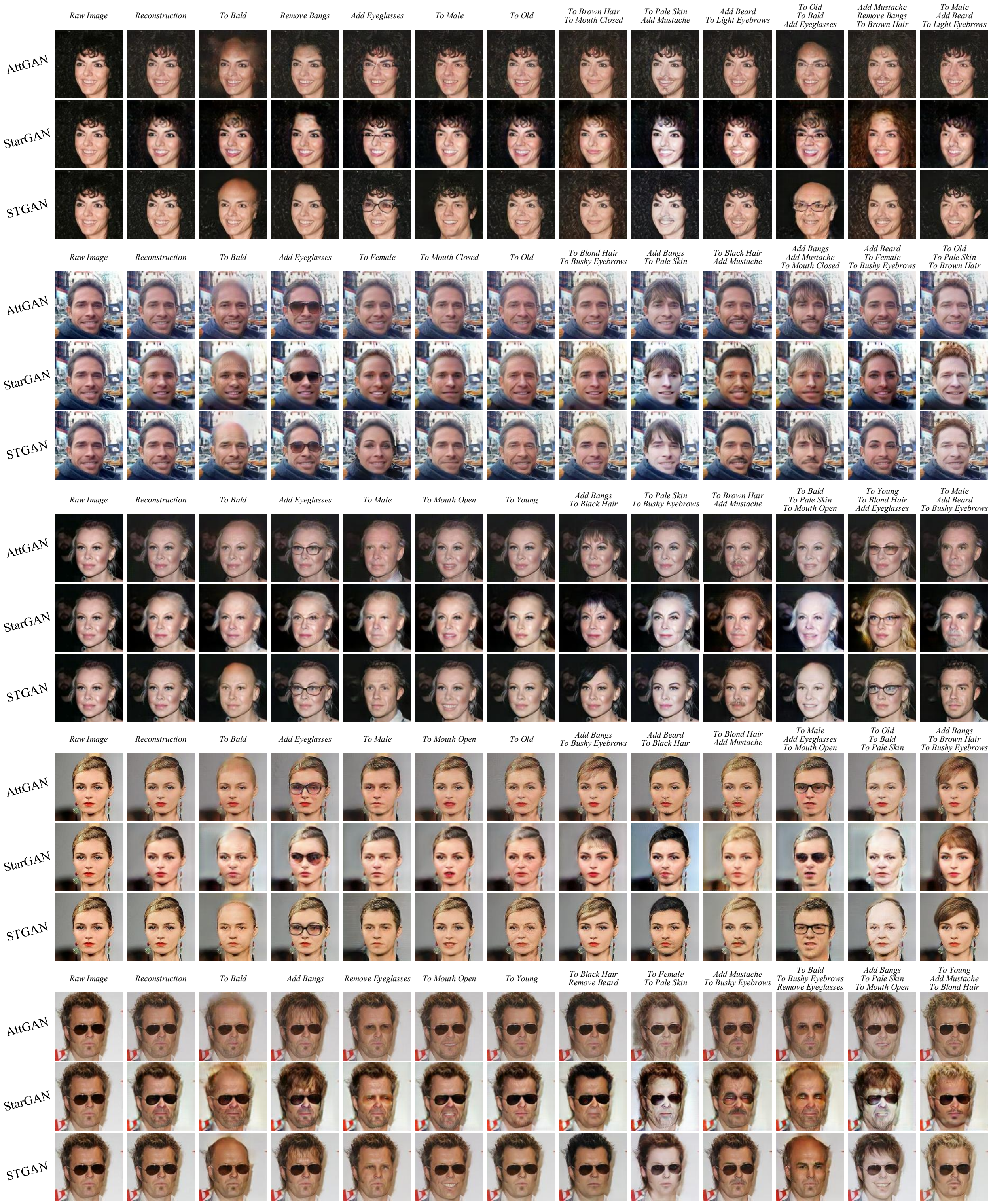

Figure C: Facial attribute editing results generated by AttGAN [2], StarGAN [1] and our STGAN, with complex and/or multiple attributes changed. Please zoom in for better observation.

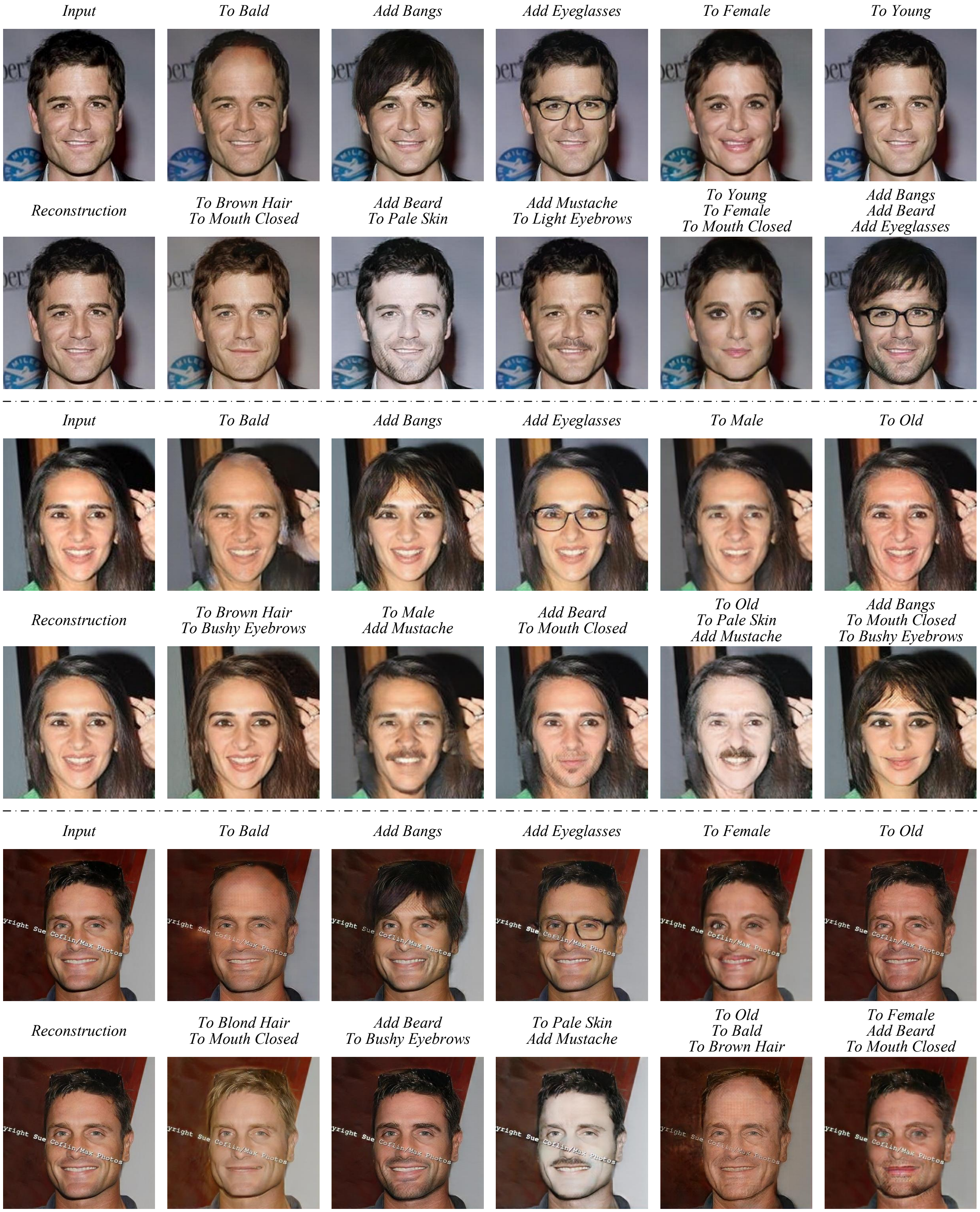

Figure D: High resolution (384 × 384) results of STGAN on facial attribute editing task.

To evaluate difference attribute vector, in addition to attribute generation accuracy reported in the main text, we also give qualitative results of AttGAN/AttGAN-diff, StarGAN/StarGAN-diff and STGAN-dst/STGAN in Fig. E. All three methods can better manipulate the attributes with difference attribute vector, and less other attributes are wrongly edited. Besides, the difference attribute vector also benefits the reconstruction quality as shown in Table B. And the qualitative results of STU variants are given in Fig. F.

Furthermore, the training and inference time of AttGAN and our STGAN are given in Table C.

Method	AttGAN	StarGAN	STGAN-dst
PSNR/SSIM	24.07/0.841	22.80/0.819	30.22/0.942
Method	AttGAN-diff	StarGAN-diff	STGAN
PSNR/SSIM	25.36/0.858	22.88/0.818	31.67/0.948

Table B: Reconstruction evaluation on difference attribute vector.

Method	Training	Inference
AttGAN	20 min/epoch	7.21 ms/image
STGAN	27 min/epoch	11.78 ms/image

Table C: Comparison of training and inference time. Note that we count inference time of image generation without preprocessing.

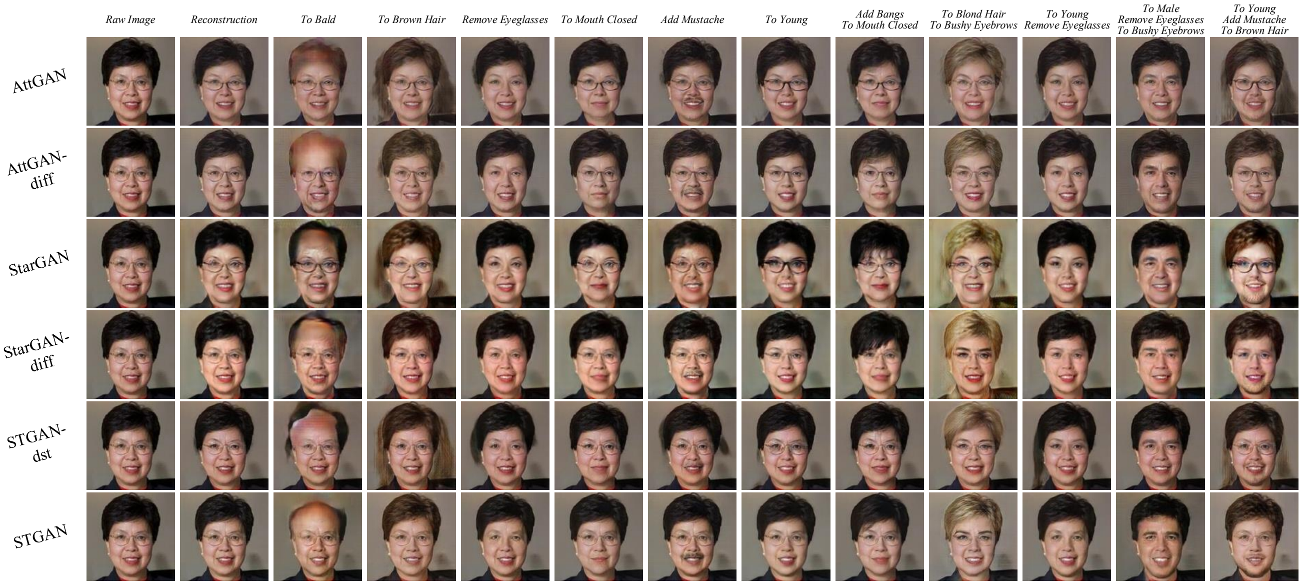

Figure E: Qualitative results on facial attribute editing task. AttGAN [2], StarGAN [1] and STGAN-dst use target attribute vector, AttGAN-diff, StarGAN-diff and STGAN are trained with difference attribute vector. Please zoom in for better observation.

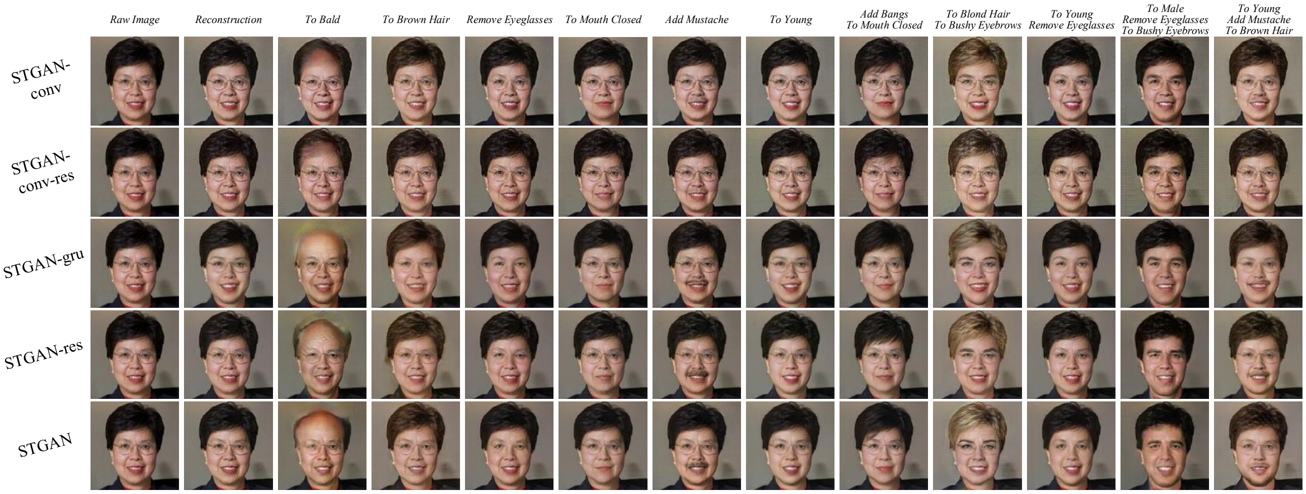

Figure F: Qualitative results on facial attribute editing task of STGAN variants. Please zoom in for better observation.

To further explore the selective mechanism and verify $z^l(l = 1, \dots, 4)$ in STUs can learn to correctly select the edited attributes-irrelevant features, we show the visualization of representative z^1s in Fig. H. We can know that z^l is able to learn to selectively transfer features, especially when the edited attributes are local ones (*e.g.*, *Bald*, *Mustache*).

We also show two representative failure cases in Fig. G. In the first row, even the sunglasses are removed, there is nothing about the occluded eyes to transfer from input image, and STGAN fails to generate high-quality eyes. In the second row, STGAN fails to recognize the hat and falsely intends to edit it to blond hair. Actually, these failure cases may be tackled by deploying hierarchical noise vectors and introducing extra attributes (*i.e.*, hat), which will be further investigated in our future work.

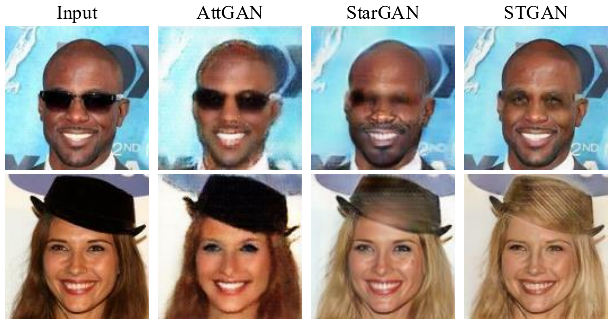

Figure G: Failure cases on removing eyeglasses and changing to blond hair.

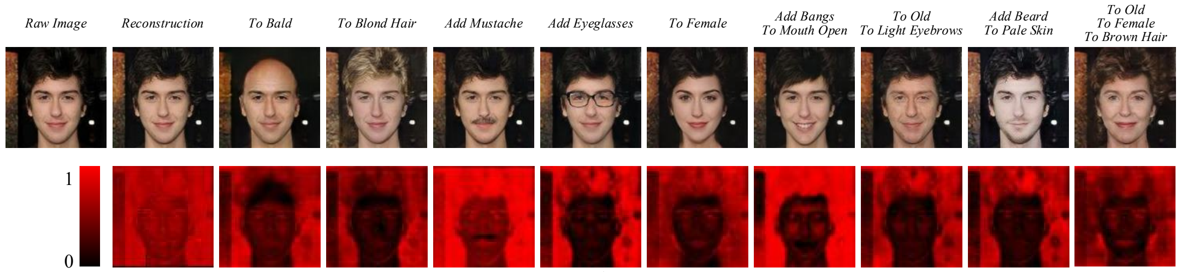

Figure H: Visualization of representative z^1s, each column represents a specific task.

D. Season Translation Results

Fig. I shows more season translation results.

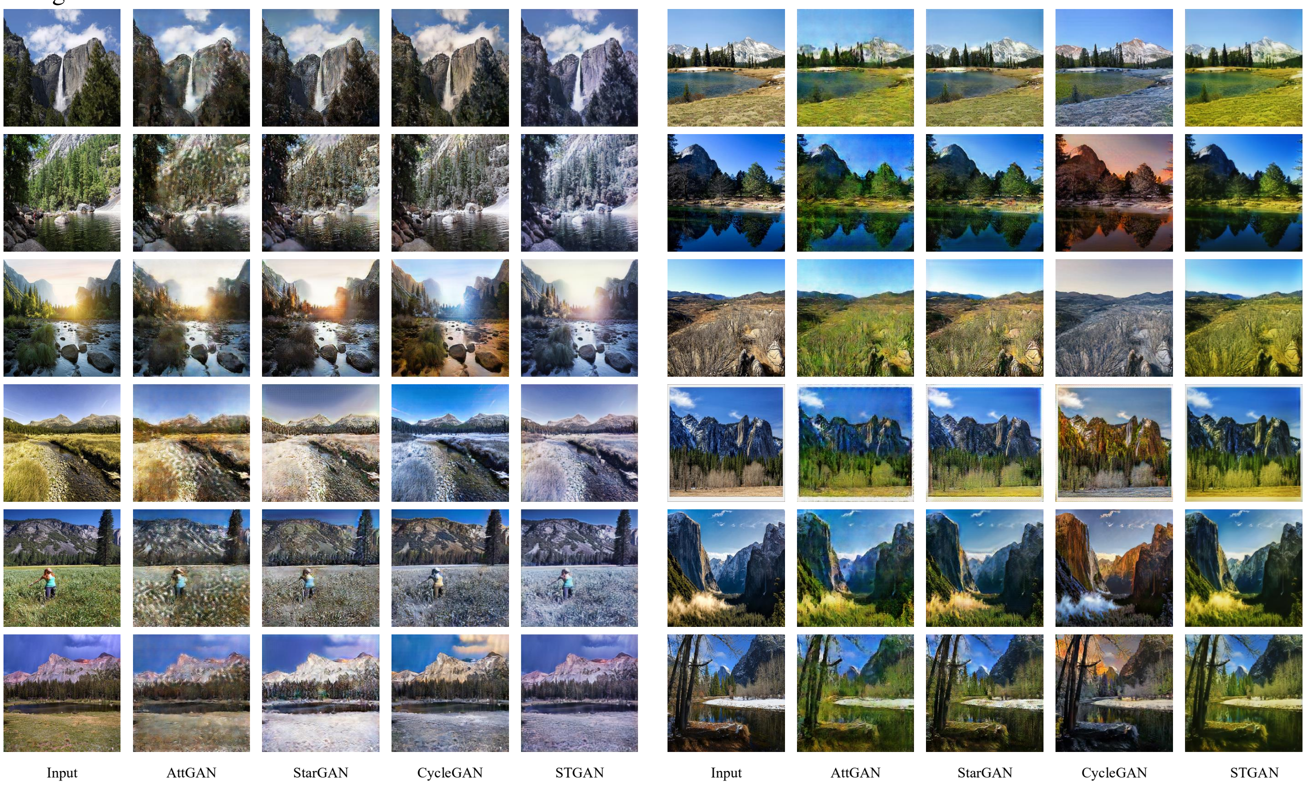

Figure I: Season translation results by AttGAN [2], StarGAN [1], CycleGAN [7] and STGAN. On the left are results on *summer→winter* task, and on the right are results on *winter→summer* task. Please zoom in for better observation.

References

[1] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018. 2, 3, 5, 6

[2] Zhenliang He, Wangmeng Zuo, Meina Kan, Shiguang Shan, and Xilin Chen. Arbitrary facial attribute editing: Only change what you want. *arXiv preprint arXiv:1711.10678*, 2017. 2, 3, 5, 6

[3] Sergey Ioffe and Christian Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015. 1

[4] Guillaume Lample, Neil Zeghidour, Nicolas Usunier, Antoine Bordes, Ludovic Denoyer, et al. Fader networks: Manipulating images by sliding attributes. In *Advances in Neural Information Processing Systems*, pages 5967–5976, 2017. 2

[5] Guim Perarnau, Joost van de Weijer, Bogdan Raducanu, and Jose M Álvarez. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*, 2016. 2

[6] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 1

[7] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision*, pages 2223–2232, 2017. 6