High-Fidelity and Arbitrary Face Editing - Supplementary Material

Yue Gao[1], Fangyun Wei[2], Jianmin Bao[2], Shuyang Gu[3], Dong Chen[2], Fang Wen[2], Zhouhui Lian[1*]

[1]Wangxuan Institute of Computer Technology, Peking University, China
[2]Microsoft Research Asia
[3]University of Science and Technology of China

{gerry, lianzhouhui}@pku.edu.cn, {fawe, jianbao, doch, fangwen}@microsoft.com,
gsy777@mail.ustc.edu.cn

In this supplementary material, we elaborate on the implementation details, some attempts to address the steganography problem of cycle consistency, comparison of different frequency domains for skip-connection in the Generator and more results on wild faces.

1. Implementation Details

Dataset Details. We use CelebA-HQ [4] as the labeled dataset, which contains $30,000$ images with 40 binary attribute annotations for each image. We randomly select $28,000$ images as the training set to train the attribute classifier C, the remaining $2,000$ images are used as the testing set. For the unlabelled dataset FFHQ [5], we use the first $66,000$ images to train the G, D_H and D_I and the remaining $4,000$ images for testing. The image resolution is chosen as 256×256 in our experiments.

Model Details. The detailed architectures of the Generator, Discriminators are shown in Table 1, Table 2 and Table 3 respectively.

Hyper-Parameters Details. The exponential moving average [13] is applied to the Generator G. We use Adam optimizer [6] with $\beta_1 = 0.0$ and $\beta_2 = 0.999$, and utilize TTUR [2] with $lr_G = 5e-4$, $lr_{D_I} = 2e-3$ and $lr_{D_H} = 2e{-}3$. The loss weights are $\lambda^I_{GAN} = 1.0, \lambda^H_{GAN} = 1.0, \lambda_{ar} = 1.0, \lambda_{ac} = 1.0$ and $\lambda_{cyc} = 10.0$. We train the model for 100 epochs and another 100 epochs training with learning rate decaying, where the decaying rate is set to 0.999 for every 10 epochs.

2. Attempts to Address the Steganography

To alleviate the steganography problem caused by cycle consistency, we first tried a few data augmentation techniques to prevent the network from encoding hidden information to satisfy the cycle consistency. Specifically, we update the cycle consistency to

$$\mathcal{L}_{cyc} = \mathbb{E}[\| A(\boldsymbol{x}) - G(A(G(\boldsymbol{x}, \boldsymbol{\Delta})), -\boldsymbol{\Delta}) \|_1], \quad (1)$$

where A stands for data augmentation operations. Horizontal flip, random noise, color jitter (*i.e.*, contrast, saturation, brightness) and affine transformation (*i.e.*, rotation, translation, scaling) are investigated.

As shown in Figure 1 and Table 4, even with data augmentations (*e.g.*, horizontal flip, color jitter and affine transformation), the model can still find a way to hide the information, it still fails to synthesize rich details in the output image. Although adding noise can somehow alleviate the steganography problem, the quality of generated images is far from satisfactory, especially the rich details are missing. On the contrary, our results are high-fidelity keeping all the rich details from the input image. This validates that our proposed approach is effective to solve the steganography problem.

Methods	FID ↓	Acc. ↑	QS ↑	SRE ↓
H-flip	5.49	95.6	0.668	0.078
Noise	6.06	94.4	0.667	0.122
ColorJitter	5.15	95.9	0.703	0.059
Affine	5.34	95.8	0.681	0.071
HifaFace	**4.04**	**97.5**	**0.803**	**0.021**

Table 4: Quantitative comparison of using different data augmentation techniques and our method to solve the steganography problem in cycle consistency.

3. Ablation Studies for the Generator

To validate that the combination of **LH**, **HL** and **HH** frequency components are essential for the *wavelet-based skip-connection*, we perform a few variants of different combinations of frequency components in *wavelet-based*

*Zhouhui Lian is the corresponding author. This work was supported by Beijing Nova Program of Science and Technology (Grant No.: Z191100001119077).

Components	Input → Output Shape	Layer Information
From RGB	(3, H, W) → (64, H, W)	Conv(F64)
Downsample ResBlock	(64, H, W) → (128, H/2, W/2)	IN-LReLU-Conv(F64)-Downsample-IN-LReLU-Conv(F128)
Downsample ResBlock	(128, H/2, W/2) → (256, H/4, W/4)	IN-LReLU-Conv(F64)-Downsample-IN-LReLU-Conv(F128)
ResBlock	(256, H/4, W/4) → (256, H/4, W/4)	IN-LReLU-Conv(F256)-IN-LReLU-Conv(F256)
ResBlock	(256, H/4, W/4) → (256, H/4, W/4)	IN-LReLU-Conv(F256)-IN-LReLU-Conv(F256)
ResBlock	(256, H/4, W/4) → (256, H/4, W/4)	IN-LReLU-Conv(F256)-IN-LReLU-Conv(F256)
AdaIN ResBlock	(256, H/4, W/4) → (256, H/4, W/4)	AdaIN-LReLU-Conv(F256)-AdaIN-LReLU-Conv(F256)
ResBlock	(256, H/4, W/4) → (256, H/4, W/4)	IN-LReLU-Conv(F256)-IN-LReLU-Conv(F256)
ResBlock	(256, H/4, W/4) → (256, H/4, W/4)	IN-LReLU-Conv(F256)-IN-LReLU-Conv(F256)
Upsample ResBlock	(256 × 4, H/4, W/4) → (128, H/2, W/2)	IN-LReLU-Conv(F256)-Upsample-IN-LReLU-Conv(F128)
Upsample ResBlock	(128 × 4, H/2, W/2) → (64, H, W)	IN-LReLU-Conv(F64)-Upsample-IN-LReLU-Conv(F3)
To RGB	(64 × 4, H, W) → (3, H, W)	LReLU-Conv(F3)

Table 1: The network architecture of the generator G. For all convolution (Conv) layers, the kernel size, stride and padding are $3, 1,$ and 1, respectively, Fx is the channel number of feature maps. "IN" denotes the Instance Normalization [11], "LReLU" denotes the LeakyReLU activation function. "AdaIN" [3] is used to inject the attribute vector. Since we used the wavelet-base skip-connection in G, the number of input channels in decoding layers are multiplied by 4.

Components	Input → Output Shape	Layer Information
D_{I0}	(3, H, W) → (32, H/2, W/2)	Conv(F32, K=4, S=2, P=1)-LReLU
	(32, H/2, W/2) → (64, H/4, W/4)	Conv(F64, K=4, S=2, P=1)-LReLU
	(64, H/4, W/4) → (128, H/8, W/8)	Conv(F128, K=4, S=2, P=1)-LReLU
	(128, H/8, W/8) → (256, H/16, W/16)	Conv(F256, K=4, S=2, P=1)-LReLU
	(256, H/16, W/16) → (512, H/32, W/32)	Conv(F512, K=4, S=2, P=1)-LReLU
	(512, H/32, W/32) → (512, H/64, W/64)	Conv(F512, K=4, S=2, P=1)-LReLU
	(512, H/64, W/64) → (1, 1, 1)	Conv(F1, K=4, S=1)
D_{I1}	(3, H/2, W/2) → (32, H/4, W/4)	Conv(F32, K=4, S=2, P=1)-LReLU
	(32, H/4, W/4) → (64, H/8, W/8)	Conv(F64, K=4, S=2, P=1)-LReLU
	(64, H/8, W/8) → (128, H/16, W/16)	Conv(F128, K=4, S=2, P=1)-LReLU
	(128, H/16, W/16) → (256, H/32, W/32)	Conv(F256, K=4, S=2, P=1)-LReLU
	(256, H/32, W/32) → (512, H/64, W/64)	Conv(F512, K=4, S=2, P=1)-LReLU
	(512, H/64, W/64) → (1, 1, 1)	Conv(F1, K=4, S=1)

Table 2: The network architecture of the multi-scale image-level Discriminators: D_{I0} and D_{I1}.

skip-connection. For concreteness, we qualitatively and quantitatively compare the following three variants with different choices of frequency components in the *wavelet-based skip-connection*, we have three variants: (1) the **HifaFace**, skip-connecting **LH**, **HL** and **HH**; (2) the Low-Freq, which use the low-frequency **LL** in the skip-connection; (3) the All-Freq, skip-connecting all the four frequency components **LL**, **LH**, **HL** and **HH**. As shown in Figure 2 and Table 5, we observe that the model can not synthesize rich details well without explicitly knowing high-frequency domain information. And if we skip-connecting all the low and high-frequency information, the model can produce rich details. The overall performance is slightly worse than our proposed HifaFace.

Methods	FID ↓	Acc. ↑	QS ↑	SRE ↓
Low-Freq	5.37	95.9	0.707	0.060
All-Freq	4.18	97.4	0.792	0.022
HifaFace	**4.04**	**97.5**	**0.803**	**0.021**

Table 5: Quantitative comparison of results of using different frequency components in *wavelet-based skip-connection*.

Components	Input $\rightarrow$ Output Shape	Layer Information
D_{H0}	(3 × 3, H/2, W/2) $\rightarrow$ (32, H/4, W/4)	Conv(F32, K=4, S=2, P=1)-LReLU
	(32, H/4, W/4) $\rightarrow$ (64, H/8, W/8)	Conv(F64, K=4, S=2, P=1)-LReLU
	(64, H/8, W/8) $\rightarrow$ (128, H/16, W/16)	Conv(F128, K=4, S=2, P=1)-LReLU
	(128, H/16, W/16) $\rightarrow$ (256, H/32, W/32)	Conv(F256, K=4, S=2, P=1)-LReLU
	(256, H/32, W/32) $\rightarrow$ (512, H/64, W/64)	Conv(F512, K=4, S=2, P=1)-LReLU
	(512, H/64, W/64) $\rightarrow$ (1, 1, 1)	Conv(F1, K=4, S=1)
D_{H1}	(3 × 3, H/4, W/4) $\rightarrow$ (64, H/8, W/8)	Conv(F64, K=4, S=2, P=1)-LReLU
	(64, H/8, W/8) $\rightarrow$ (128, H/16, W/16)	Conv(F128, K=4, S=2, P=1)-LReLU
	(128, H/16, W/16) $\rightarrow$ (256, H/32, W/32)	Conv(F256, K=4, S=2, P=1)-LReLU
	(256, H/32, W/32) $\rightarrow$ (512, H/64, W/64)	Conv(F512, K=4, S=2, P=1)-LReLU
	(512, H/64, W/64) $\rightarrow$ (1, 1, 1)	Conv(F1, K=4, S=1)

Table 3: The network architecture of the multi-scale high-frequency Discriminators: D_{H0} and D_{H1}.

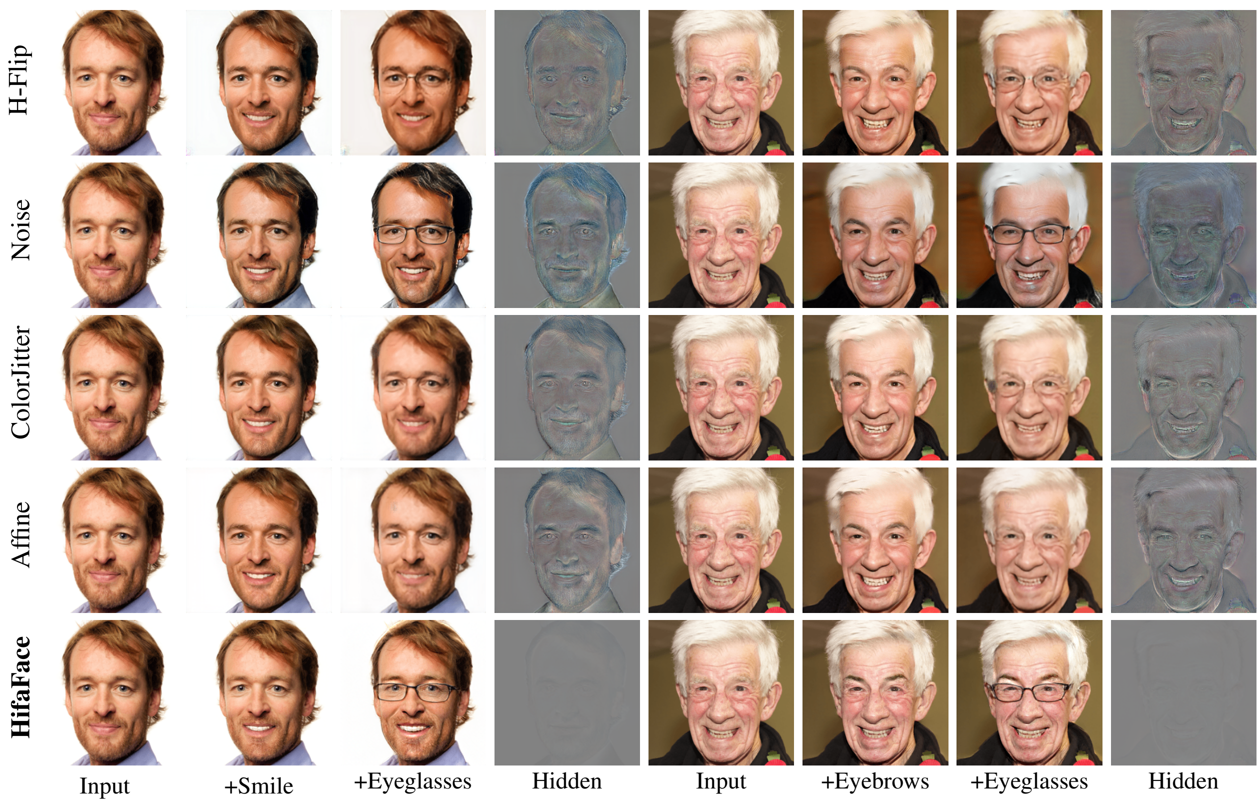

Figure 1: Comparison of methods using different data augmentation techniques and our methods to solve the steganography problem in cycle consistency.

4. Additional Visual Results

In this section, more visual results are provided to demonstrate the superiority of our model. The figures to be presented and their corresponding subjects are listed as follows:

- In Figure 3, we present the comparison of attribute-based face editing results obtained by our model and other existing methods, including GANimation [9], STGAN [8], RelGAN [12], InterFaceGAN [10] and StyleFlow [1]. We also provide the results by an industrial app, FaceApp [7].
- In Figure 4 and Figure 5, we show the comparison of arbitrary face editing results obtained by our HifaFace, our model without the attribute regression loss $\mathcal{L}_{ar}$, RelGAN [12] and InterFaceGAN [10].
- In Figure 6, we demonstrate that our method HifaFace can handle face images under various poses, races and expressions.

Figure 2: Comparison of methods with different combinations of frequency components in the wavelet-based skip-connection.

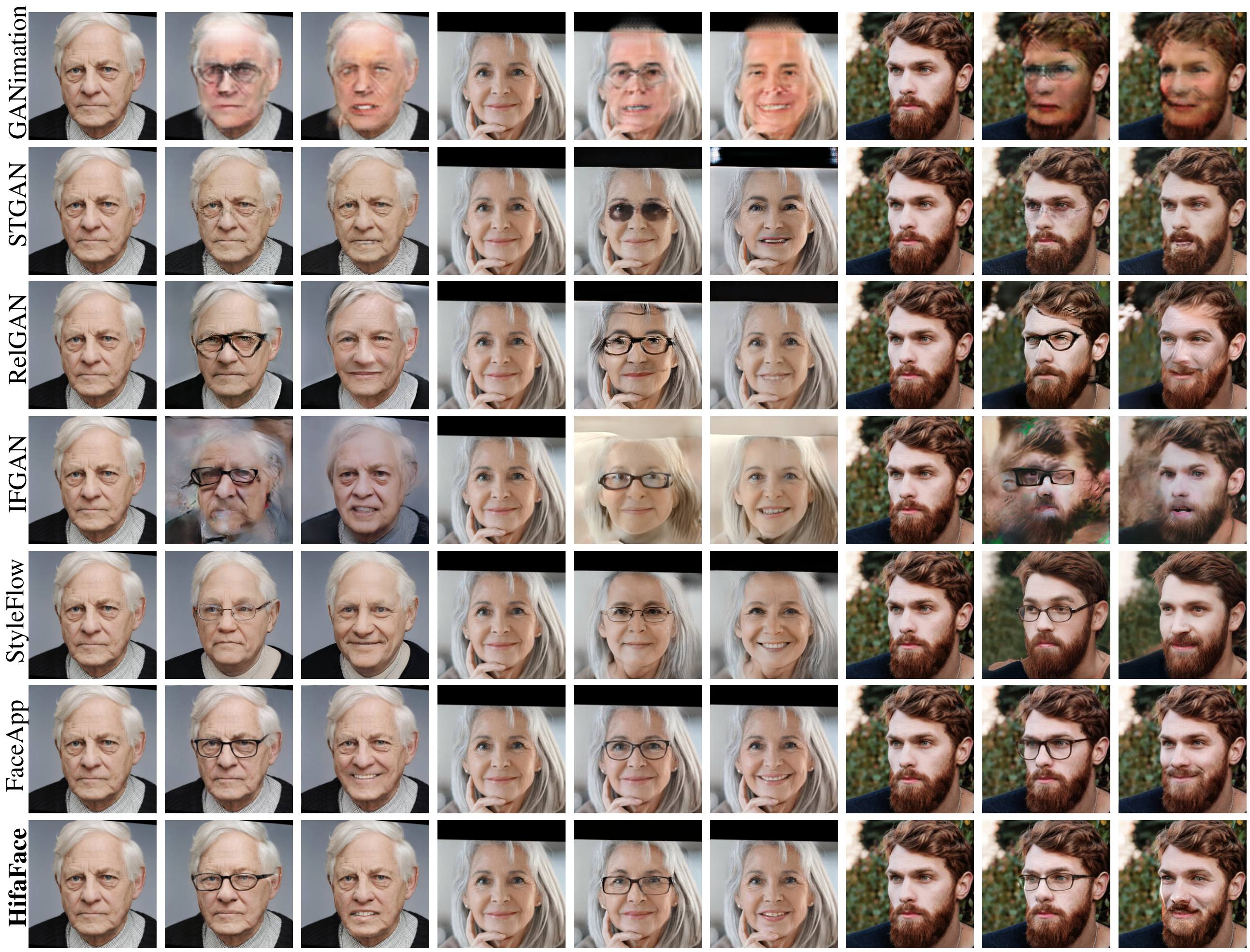

Figure 3: Comparison of results obtained by our HifaFace and other state-of-the-art methods.

References

[1] Rameen Abdal, Peihao Zhu, Niloy Mitra, and Peter Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ArXiv*, abs/2008.02401, 2020. 3

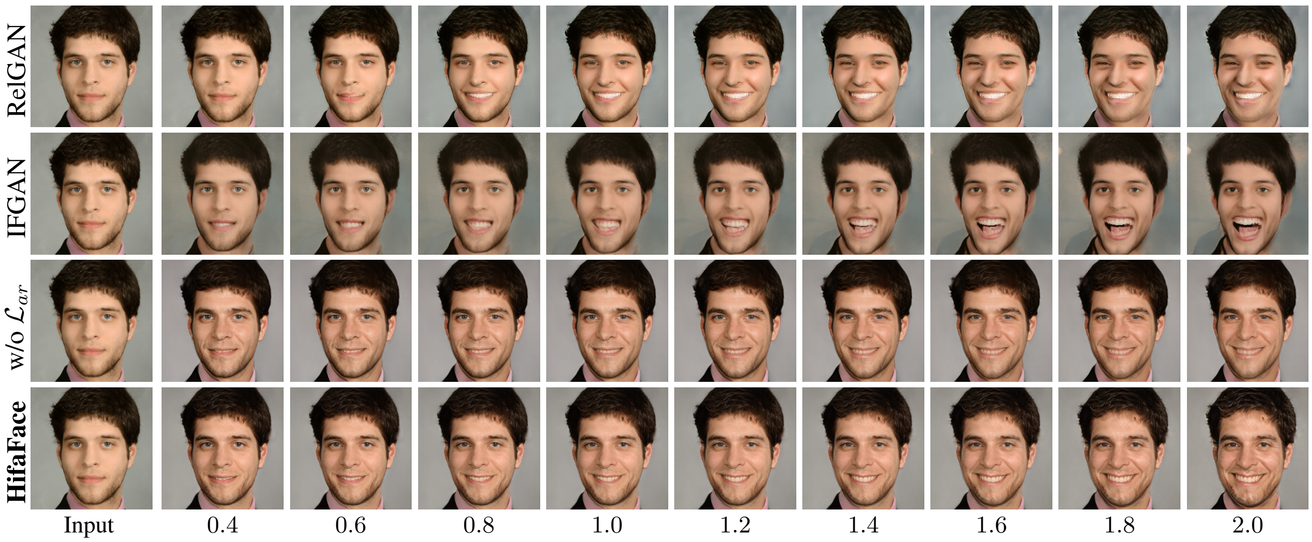

Figure 4: Interpolation results on attribute "smile" obtained by RelGAN [12], InterFaceGAN(IFGAN) [10], HifaFace without the $\mathcal{L}_{ar}$ and our HifaFace.

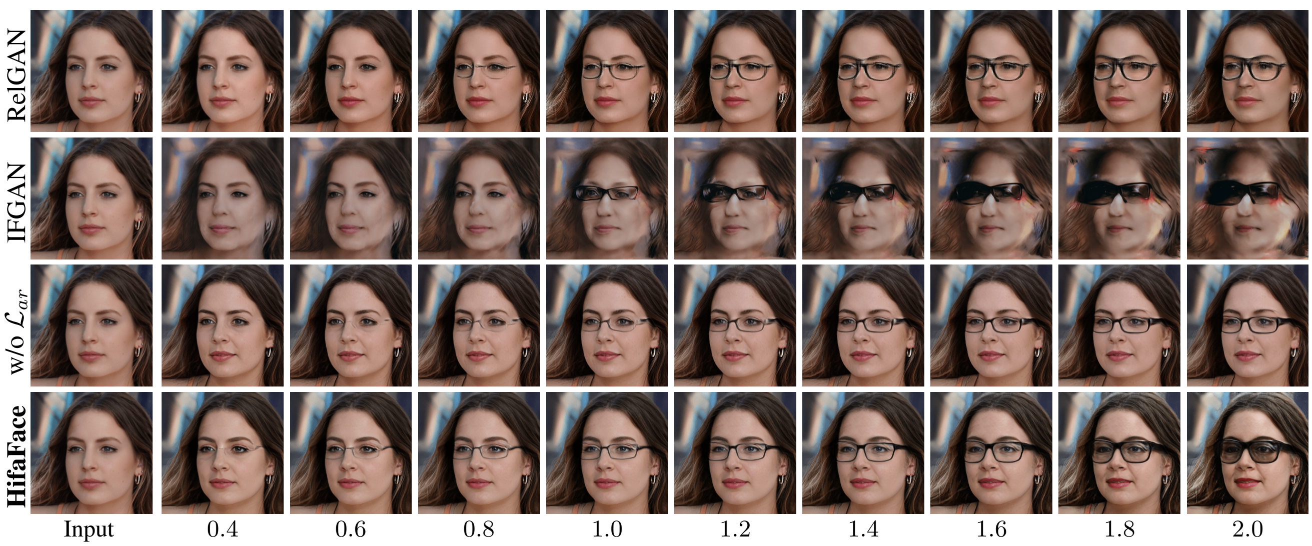

Figure 5: Interpolation results on attribute "eyeglasses" obtained by RelGAN [12], InterFaceGAN(IFGAN) [10], HifaFace without the $\mathcal{L}_{ar}$ and our HifaFace.

[2] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NIPS*, 2017. 1

[3] X. Huang and Serge J. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1510–1519, 2017. 2

[4] Tero Karras, Timo Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *ArXiv*, abs/1710.10196, 2018. 1

[5] Tero Karras, S. Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4396–4405, 2019. 1

[6] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015. 1

[7] Wireless Lab. Faceapp. `https://www.faceapp.com`. 3

[8] Ming Liu, Yukang Ding, Min Xia, Xiao Liu, E. Ding, W. Zuo, and Shilei Wen. Stgan: A unified selective transfer network for arbitrary image attribute editing. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3668–3677, 2019. 3

[9] Albert Pumarola, Antonio Agudo, Aleix M Martinez, Alberto Sanfeliu, and Francesc Moreno-Noguer. Ganimation: Anatomically-aware facial animation from a single image. In *Proceedings of the ECCV*, 2018. 3

[10] Yujun Shen, Ceyuan Yang, X. Tang, and B. Zhou. Interfacegan: Interpreting the disentangled face representation

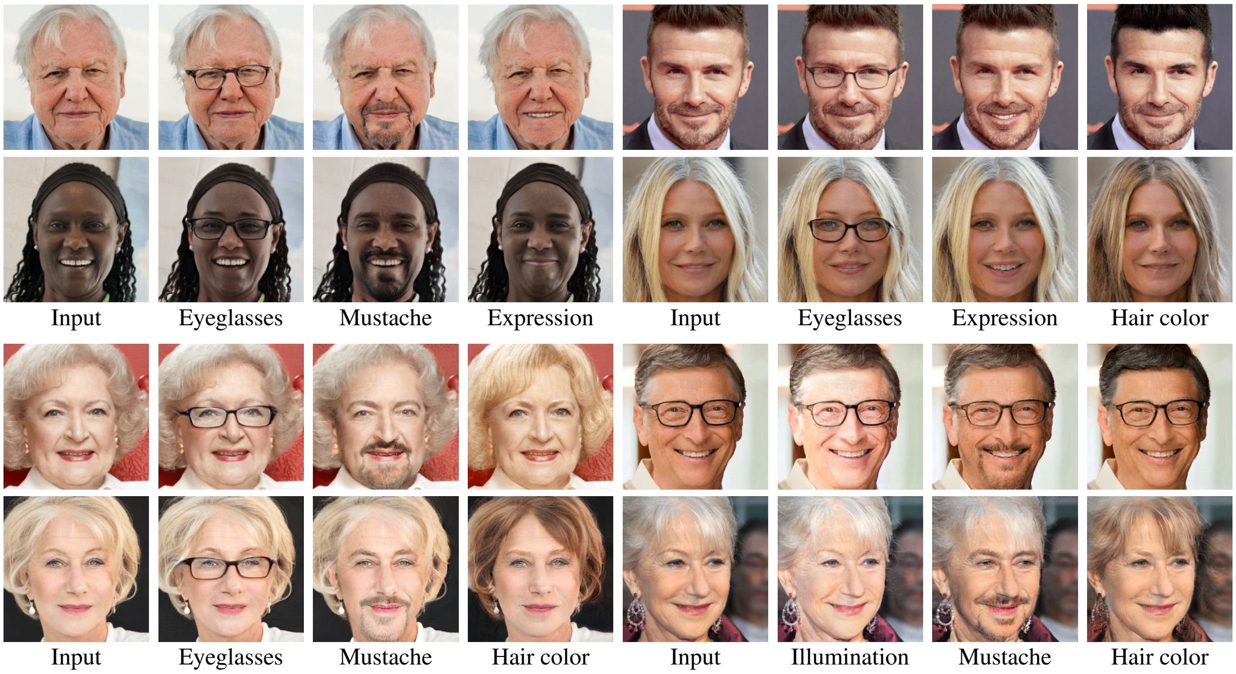

Figure 6: Face editing results obtained by our **HifaFace** on wild images.

learned by gans. *IEEE transactions on pattern analysis and machine intelligence*, PP, 2020. 3, 5

[11] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *ArXiv*, abs/1607.08022, 2016. 2

[12] P. Wu, Yu-Jing Lin, Che-Han Chang, E. Chang, and S. Liao. Relgan: Multi-domain image-to-image translation via relative attributes. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5913–5921, 2019. 3, 5

[13] Yasin Yazici, C. S. Foo, S. Winkler, Kim-Hui Yap, G. Piliouras, and V. Chandrasekhar. The unusual effectiveness of averaging in gan training. *ArXiv*, abs/1806.04498, 2019. 1