Supplementary Materials
HyperStyle: StyleGAN Inversion with HyperNetworks for Real Image Editing

A. Broader Impact

HyperStyle enables accurate and highly editable inversions of real images. While our tool aims to empower content creators, it can also be used to generate more convincing deep-fakes [25] and aid in the spread of disinformation [27]. However, powerful tools already exist for the detection of GAN-synthesized imagery [15, 28]. These tools continually evolve, which gives us hope that any potential misuse of our method can be mitigated.

Another cause for concern is the bias that generative networks inherit from their training data [17]. Our model was similarly trained on such a biased set, and as a result, may display degraded performance when dealing with images from minority classes [16]. However, we have demonstrated that our model successfully generalizes beyond its training set, and allows us to similarly shift the GAN beyond its original domain. These properties allow us to better preserve minority traits when compared to prior works, and we hope that this benefit would similarly enable fairer treatment of minorities in downstream tasks.

B. Ablation Study: Qualitative Comparisons

In Sec. 4.3 of the main paper, we presented a quantitative ablation study to validate the design choices of our hypernetworks. We now turn to provide visual comparisons to complement this. First, we illustrate the effectiveness of the iterative refinement scheme in Fig. 1. Observe that iteratively predicting the weight offsets results in sharper reconstructions. This is best reflected in the preservation of fine details, most notably in the reconstruction of hairstyle and facial hair. In Fig. 2, we show that altering the toRGB convolutional layers harms the editability of the resulting inversions. This is most noticeable in edits requiring global changes, such as pose and age. For instance, altering the pose in the second row results in blurred edits. Additionally, when modifying age in the bottom two rows, HyperStyle succeeds in realistically altering the clothing (3rd row) and hairstyle (4th row).

C. Additional Quantitative Results

Following the quantitative reconstruction metrics provided in the main paper on the human facial domain, we provide quantitative results on the cars domain and wild animals domain in Tabs. 1 and 2.

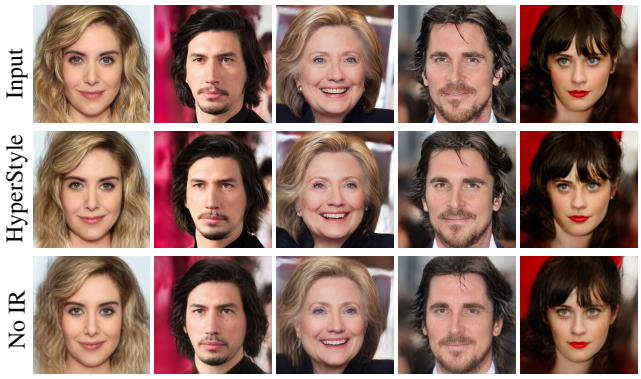


Figure 1. Reconstruction comparison of HyperStyle with and without the iterative refinement (IR) training and inference scheme. As shown, gradually predicting the desired weight offsets results in sharper images with less artifacts, particularly in finer details along the hair, for example. Best viewed zoomed-in.

Figure 2. Comparison of HyperStyle trained with and without altering the toRGB layers of StyleGAN, denoted by "All Layers" and "HyperStyle", respectively. Altering toRGB layers produces more noticeable editing artifacts when making a global change.

Cars				
Method	↑ MS-SSIM	↓ LPIPS	↓ L_2	↓ Time (s)
StyleGAN2 [11]	0.79	0.16	0.06	198.9
PTI [21]	0.93	0.11	0.01	33.71
pSp [20]	0.58	0.29	0.10	0.071
e4e [26]	0.53	0.32	0.12	0.071
ReStyle$_{pSp}$ [1]	0.66	0.25	0.07	0.359
ReStyle$_{e4e}$ [1]	0.60	0.29	0.09	0.359
HyperStyle	0.67	0.27	0.07	0.491

Table 1. Quantitative reconstruction results on the cars domain, computed over the Stanford Cars dataset [12].

AFHQ Wild				
Method	↑ MS-SSIM	↓ LPIPS	↓ L_2	↓ Time (s)
StyleGAN2 [11]	0.82	0.13	0.03	203.0
PTI [21]	0.93	0.08	0.01	33.71
pSp [20]	0.51	0.35	0.13	0.075
e4e [26]	0.47	0.36	0.14	0.075
ReStyle$_{pSp}$ [1]	0.57	0.21	0.05	0.303
ReStyle$_{e4e}$ [1]	0.52	0.25	0.07	0.303
HyperStyle	0.56	0.24	0.06	0.551

Table 2. Quantitative reconstruction results on the wild animals domain, computed over the AFHQ Wild [3] dataset.

D. The HyperStyle Architecture

Given the 6-channel input, the HyperStyle architecture begins with a shared backbone which outputs a single $16 \times 16 \times 512$ feature map. This feature map is then passed to each *Refinement Block*, which further down-samples the feature map using a set of 2-strided 3×3 convolutions with LeakyReLU activations to obtain a $1 \times 1 \times 512$ representation. The standard Refinement Block then directly predicts the $1 \times 1 \times C_\ell^{in} \times C_\ell^{out}$ using a single fully-connected layer.

In addition to the standard Refinement Block, we introduce a *Shared Refinement Block* that is shared between multiple hypernetwork layers. These Shared Refinement Blocks make use of two fully-connected layers whose weights are shared between different output heads. The first fully-connected layer transforms the $1 \times 1 \times 512$ tensor to a 512×512 intermediate representation. This is followed by a per-channel fully-connected layer which maps each 1×512 channel to a $1 \times 1 \times 512$ tensor, resulting in the final $1 \times 1 \times 512 \times 512$ dimensional offsets. We apply these shared blocks to all generator layers with a convolutional dimension of $3 \times 3 \times 512 \times 512$.

In both cases, we obtain a predicted offset of size $1 \times 1 \times C_\ell^{in} \times C_\ell^{out}$ for a given generator layer ℓ. This tensor is then broadcasted channel-wise to match the $k_\ell \times k_\ell$ kernel dimension of the generator's convolutional filters, resulting in a final offset of size $k_\ell \times k_\ell \times C_\ell^{in} \times C_\ell^{out}$. This final tensor can then be used to update the convolutional weights using Eq. (6) presented in the main paper. We provide a breakdown of the two architectures in Tab. 3 and Tab. 4.

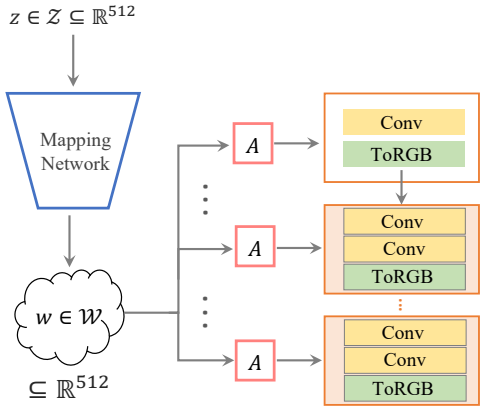


Figure 3. The StyleGAN2 [11] architecture.

E. The StyleGAN2 Architecture

To determine which network parameters are most crucial to our inversion goal, it is important to understand the overall function of the components where these parameters reside. In the case of StyleGAN2 [11], we consider four key components, as illustrated in Fig. 3. First, a mapping network converts the initial latent code $z \sim \mathcal{N}(0,1)^{512}$, into an equivalent code in a learned latent space $w \in \mathcal{W}$. These codes are then fed into a series of affine transformation blocks, one for each of the network's convolutional layers, which in turn predict a series of factors used to modulate the convolutional kernel weights. Lastly, the generator itself is built from two types of convolutional blocks: feature space convolutions, which learn increasingly complex representations of the data in some high-dimensional feature space, and toRGB blocks, which utilize convolutions to map these complex representations into residuals in a planar (or RGB) space. Of these four components, we restrict ourselves to modifying only the feature-space convolutions. We outline the different generator layers and their dimensions in Tab. 5.

F. Additional Qualitative Results

Finally, we provide additional results and comparisons, as follows:

1. Fig. 4 provides additional reconstruction comparisons on the human facial domain.

2. Fig. 5 contains a visual comparison between HyperStyle and IDInvert [30] on the human facial domain.

Layer	Weight Dimension	Output Size
Conv-LeakyReLU	$3 \times 3 \times 512 \times 256$	$8 \times 8 \times 256$
Conv-LeakyReLU	$3 \times 3 \times 256 \times 256$	$4 \times 4 \times 256$
Conv-LeakyReLU	$3 \times 3 \times 256 \times 512$	$2 \times 2 \times 512$
AdaptivePool2D	-	$1 \times 1 \times 512$
Fully-Connected	$512 \times \left(C_\ell^{in} \cdot C_\ell^{out}\right)$	$1 \times 1 \times C_\ell^{in} \times C_\ell^{out}$

Table 3. The breakdown of the standard *Refinement Block* described in Section 3 of the main paper. For a given generator layer ℓ, Refinement Block receives as input the $16 \times 16 \times 512$ feature map extracted from the shared backbone and returns the predicted weight offsets of dimensions $1 \times 1 \times C_\ell^{in} \times C_\ell^{out}$.

Layer	Weight Dimension	Output Size
Conv-LeakyReLU	$3 \times 3 \times 512 \times 128$	$16 \times 16 \times 128$
Conv-LeakyReLU	$3 \times 3 \times 128 \times 128$	$8 \times 8 \times 128$
Conv-LeakyReLU	$3 \times 3 \times 128 \times 128$	$4 \times 4 \times 128$
Conv-LeakyReLU	$3 \times 3 \times 128 \times 128$	$2 \times 2 \times 128$
Conv-LeakyReLU	$3 \times 3 \times 128 \times 512$	$1 \times 1 \times 512$
Fully-Connected	512×512	$1 \times 1 \times 512$
Shared Fully-Connected	$512 \times (512 \cdot 512)$	512×512
Shared Fully-Connected	$512 \times (512 \cdot 1 \cdot 1)$	$1 \times 1 \times 512 \times 512$

Table 4. The breakdown of the *Shared Refinement Block* described in Section 3 of the main paper. The Shared Refinement Block consists of a pair of fully-connected layers that are shared across multiple blocks (namely those tasked with predicting offsets for the generator's largest convolutional layers).

3. Fig. 6 provides reconstruction comparisons on the cars domain.
4. Fig. 7 shows additional editing comparisons on the human facial domain obtained with StyleCLIP [18] and InterFaceGAN [24].
5. Fig. 8 shows additional HyperStyle editing results on the human facial domain obtained with InterFaceGAN [24].
6. Fig. 9 contains additional HyperStyle editing results on the human facial domain obtained with StyleCLIP [18].
7. Fig. 10 provides additional editing comparisons on the cars domain obtained with GANSpace [6].
8. Fig. 11 contains additional HyperStyle editing results on the cars domain obtained with GANSpace [6].
9. Fig. 12 contains additional HyperStyle editing results obtained with StyleCLIP [18] on the AFHQ Wild [3] test set.
10. Fig. 13 and Fig. 14 illustrate HyperStyle's reconstructions and edits on challenging out-of-domain images.
11. Fig. 15 illustrates additional domain adaptation results and comparisons.

G. Implementation Details

All hypernetworks employ a ResNet34 [7] backbone pre-trained on ImageNet. The networks have a modified input layer to accommodate the 6-channel inputs. We train our networks using the Ranger optimizer [29] with a constant learning rate of 0.0001 and a batch size of 8.

When applying the iterative refinement scheme from Alaluf *et al.* [1], our hypernetworks use $T = 5$ iterative steps per batch during training. For each step t, we compute losses between the current reconstructions and inputs. That is, losses are computed T times per batch.

Following recent works [1, 20, 26], we set $\lambda_{\text{LPIPS}} = 0.8$. For the similarity loss human facial domain, we use a pre-trained ArcFace [4] network with $\lambda_{sim} = 0.1$. For the remaining domains, we utilize a MoCo-based [2] loss with $\lambda_{sim} = 0.5$, as done in Tov *et al.* [26]. All experiments were conducted on a single NVIDIA Tesla P40 GPU.

H. Licenses

We provide the licenses of all datasets and models used in our work in Tab. 6.

Group	Layer Index ℓ	Layer Name	Filter Dimension $k \times k \times C_\ell^{in} \times C_\ell^{out}$
Coarse	1	Conv 1	$3 \times 3 \times 512 \times 512$
	2	toRGB 1	$1 \times 1 \times 512 \times 3$
	3	Conv 2	$3 \times 3 \times 512 \times 512$
	4	Conv 3	$3 \times 3 \times 512 \times 512$
	5	toRGB 2	$1 \times 1 \times 512 \times 3$
Medium	6	Conv 4	$3 \times 3 \times 512 \times 512$
	7	Conv 5	$3 \times 3 \times 512 \times 512$
	8	toRGB 3	$1 \times 1 \times 512 \times 3$
	9	Conv 6	$3 \times 3 \times 512 \times 512$
	10	Conv 7	$3 \times 3 \times 512 \times 512$
	11	toRGB 4	$1 \times 1 \times 512 \times 3$
Fine	12	Conv 8	$3 \times 3 \times 512 \times 512$
	13	Conv 9	$3 \times 3 \times 512 \times 512$
	14	toRGB 5	$1 \times 1 \times 512 \times 3$
	15	Conv 10	$3 \times 3 \times 512 \times 256$
	16	Conv 11	$3 \times 3 \times 256 \times 256$
	17	toRGB 6	$1 \times 1 \times 256 \times 3$
	18	Conv 12	$3 \times 3 \times 256 \times 128$
	19	Conv 13	$3 \times 3 \times 128 \times 128$
	20	toRGB 7	$1 \times 1 \times 128 \times 3$
	21	Conv 14	$3 \times 3 \times 128 \times 64$
	22	Conv 15	$3 \times 3 \times 64 \times 64$
	23	toRGB 8	$1 \times 1 \times 64 \times 3$
	24	Conv 16	$3 \times 3 \times 64 \times 32$
	25	Conv 17	$3 \times 3 \times 32 \times 32$
	26	toRGB 9	$1 \times 1 \times 32 \times 3$

Table 5. The breakdown of the StyleGAN2 [11] layer weights and their filter dimensions, split into the coarse, medium, and fine sets. Our final hypernetwork configuration alters the non-toRGB, feature-space convolutions from the medium and fine layers.

Dataset	Source	License
FFHQ	[10]	CC BY-NC-SA 4.0[1]
CelebA-HQ	[8, 14]	Non-commercial Research Purposes
Stanford Cars	[12]	Non-commercial Research Purposes
AFHQ	[3]	CC BY-NC 4.0

(a) Datasets

Model	Source	License
StyleGAN2	[11]	Nvidia Source Code License-NC
pSp	[20]	MIT License
e4e	[26]	MIT License
ReStyle	[1]	MIT License
PTI	[21]	MIT License
IDInvert	[30]	MIT License
InterFaceGAN	[24]	MIT License
StyleCLIP	[18]	MIT License
GANSpace	[6]	Apache 2.0 License
StyleGAN2-pytorch	[23]	MIT License
StyleGAN-ADA	[9]	Nvidia Source Code License
StyleGAN-NADA	[5]	MIT License
Toonify	[19]	No License
Anycost GAN	[13]	MIT License
HopeNet	[22]	Apache 2.0 License

(b) Models

Table 6. Datasets and models used in our work and their respective licenses.

References

[1] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Restyle: A residual-based stylegan encoder via iterative refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021. 2, 3, 4

[2] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 3

[3] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains, 2020. 2, 3, 4, 14

[4] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition, 2019. 3

[5] Rinon Gal, Or Patashnik, Haggai Maron, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators, 2021. 4, 15, 17

[6] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *arXiv preprint arXiv:2004.02546*, 2020. 3, 4, 12, 13

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 3

[8] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017. 4, 7

[9] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data, 2020. 4

[10] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 4, 15

[11] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119, 2020. 2, 4

[12] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013. 2, 4

[13] Ji Lin, Richard Zhang, Frieder Ganz, Song Han, and Jun-Yan Zhu. Anycost gans for interactive image synthesis and editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14986–14996, 2021. 4

[14] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild, 2015. 4, 7

[15] Sara Mandelli, Nicolò Bonettini, Paolo Bestagini, and Stefano Tubaro. Training CNNs in presence of JPEG compression: Multimedia forensics vs computer vision. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, 2020. 1

[16] Sachit Menon, Alexandru Damian, Shijia Hu, Nikhil Ravi, and Cynthia Rudin. Pulse: Self-supervised photo upsampling via latent space exploration of generative models. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 2437–2445, 2020. 1

[17] Michele Merler, Nalini Ratha, Rogerio S. Feris, and John R. Smith. Diversity in faces, 2019. 1

[18] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery, 2021. 3, 4, 9, 14

[19] Justin N M Pinkney and Doron Adler. Resolution Dependant GAN Interpolation for Controllable Image Synthesis Between Domains. *arXiv preprint arXiv:2010.05334*, 2020. 4, 17

[20] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 2, 3, 4

[21] Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *arXiv preprint arXiv:2106.05744*, 2021. 2, 4

[22] Nataniel Ruiz, Eunji Chong, and James M. Rehg. Fine-grained head pose estimation without keypoints. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018. 4

[23] Kim Seonghyeon. Stylegan2-pytorch. https://github.com/rosinality/stylegan2-pytorch, 2020. 4

[24] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9243–9252, 2020. 3, 4, 9, 10, 11

[25] Supasorn Suwajanakorn, Steven M Seitz, and Ira Kemelmacher-Shlizerman. Synthesizing obama: learning lip sync from audio. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017. 1

[26] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation, 2021. 2, 3, 4

[27] Cristian Vaccari and Andrew Chadwick. Deepfakes and disinformation: Exploring the impact of synthetic political video on deception, uncertainty, and trust in news. *Social Media + Society*, 6(1):2056305120903408, 2020. 1

[28] Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A Efros. Cnn-generated images are surprisingly easy to spot...for now. In *CVPR*, 2020. 1

[29] Less Wright. Ranger - a synergistic optimizer. https://github.com/lessw2020/Ranger-Deep-Learning-Optimizer, 2019. 3

[30] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. *arXiv preprint arXiv:2004.00049*, 2020. 2, 4, 7

Figure 4. Additional reconstruction quality comparisons on the human facial domain between the various inversion techniques. Best viewed zoomed-in.

Figure 5. Reconstruction comparison of HyperStyle and IDInvert [30] on the CelebA-HQ [8, 14] test set.

Figure 6. Additional reconstruction quality comparisons on the cars domain between the various inversion techniques.

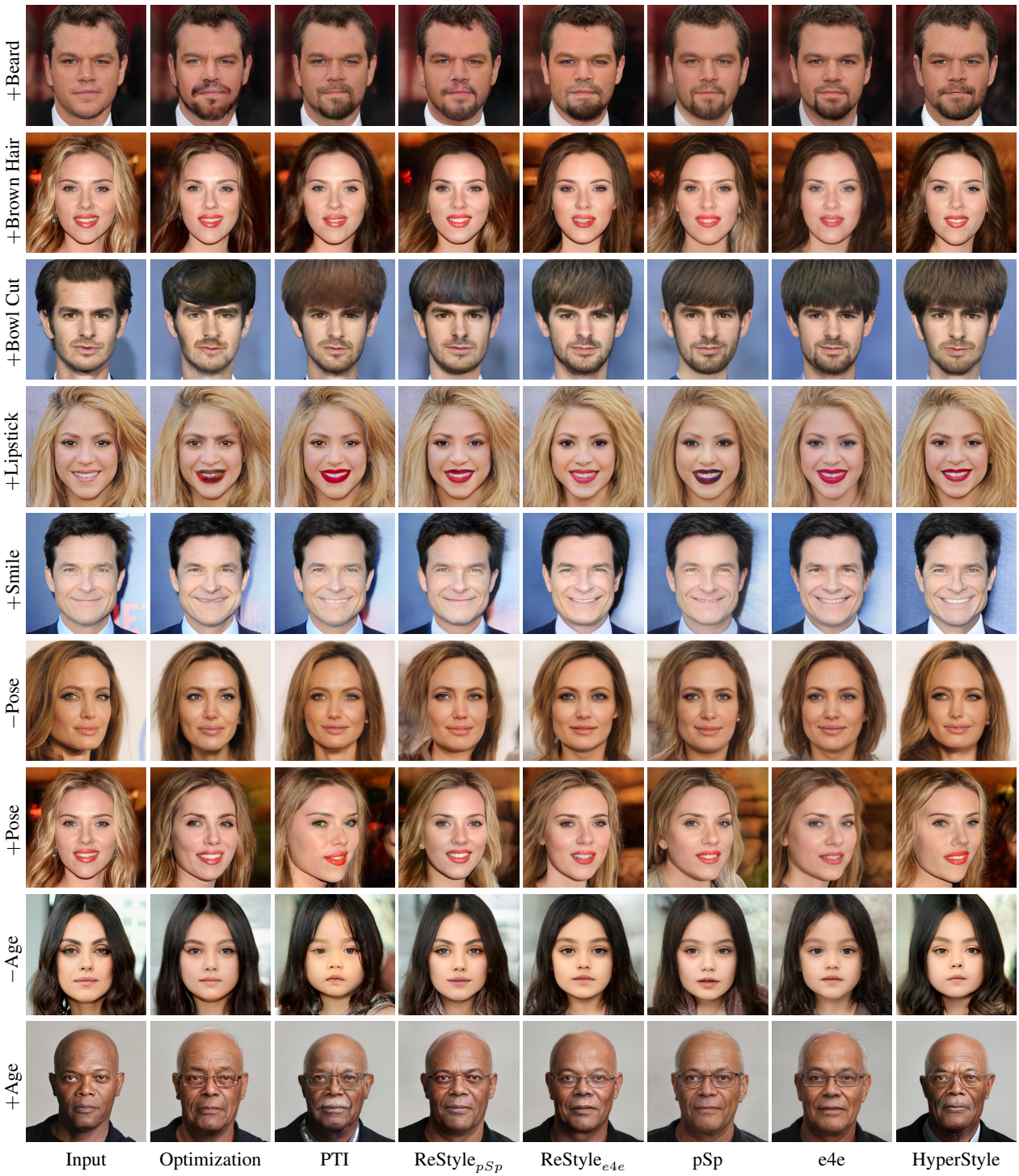


Figure 7. Additional editing comparisons between inversion techniques obtained using StyleClip [18] (first five rows) and InterFaceGAN [24] (bottom four rows).

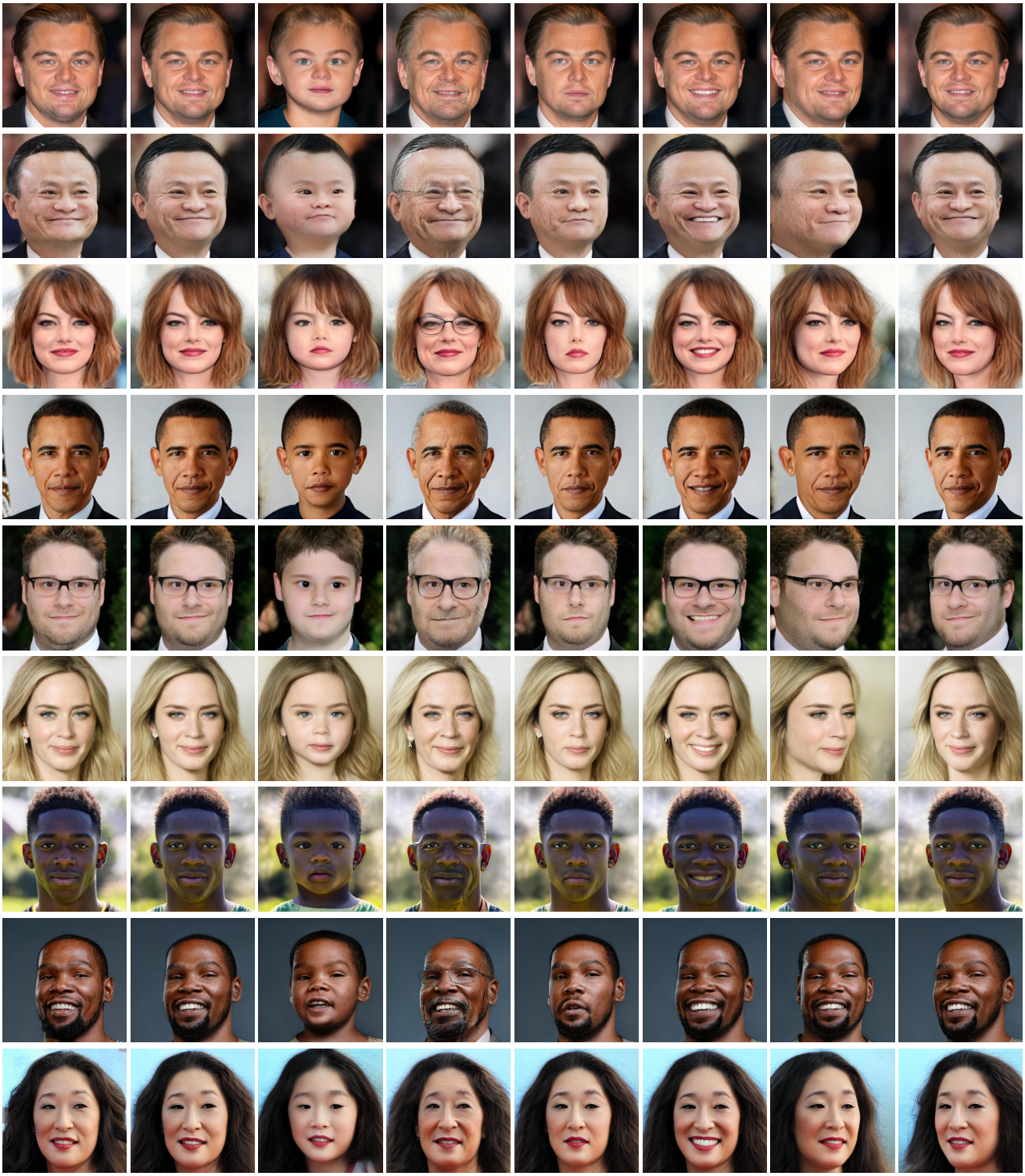


Figure 8. Additional reconstruction and editing results obtained by HyperStyle over the facial domain using InterFaceGAN [24].

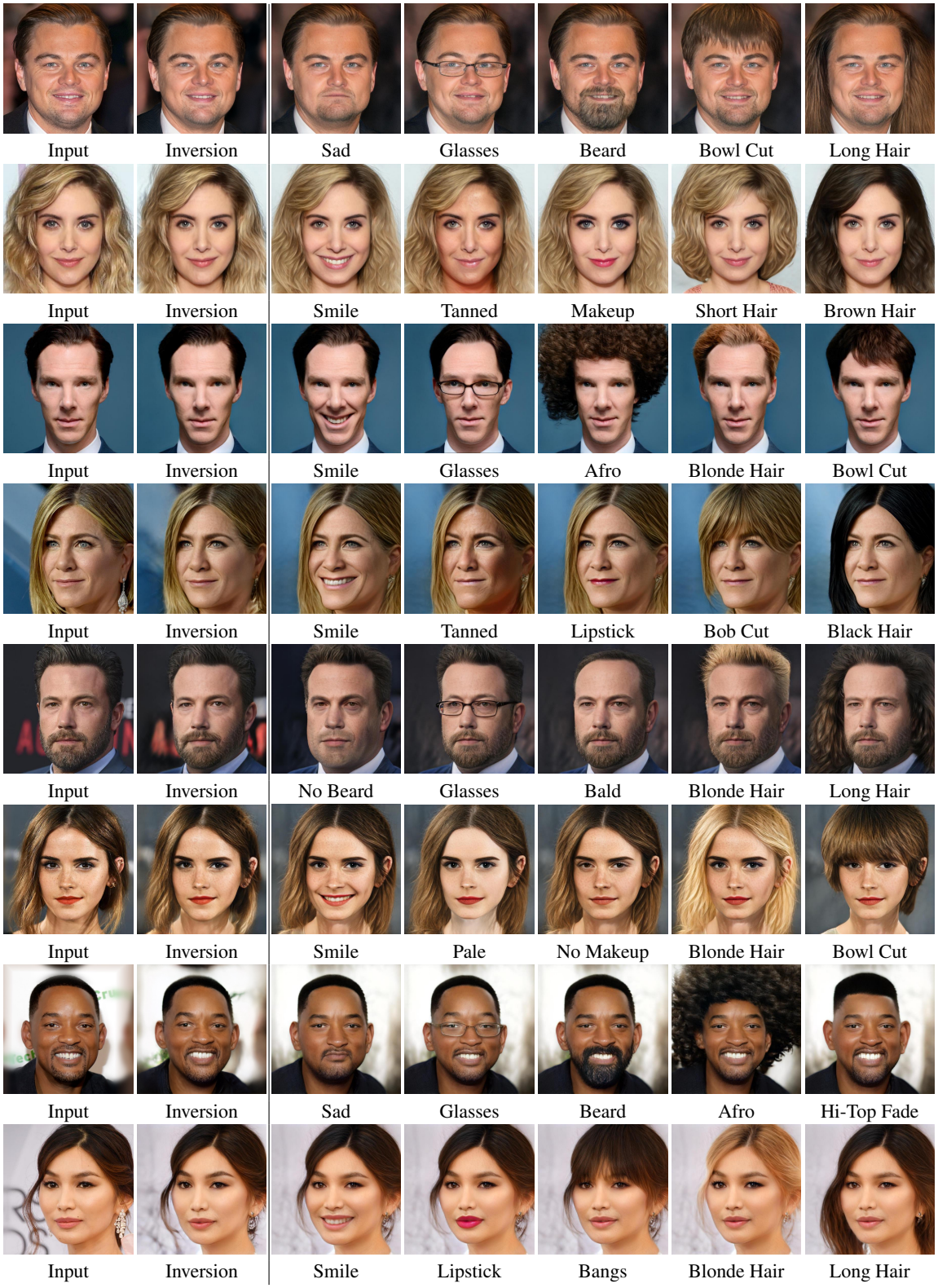


Figure 9. Additional reconstruction and editing results obtained by HyperStyle over the facial domain using StyleCLIP's [24] global direction approach. We illustrate a wide range of edits including changes to expression, facial hair, makeup, and hairstyle.

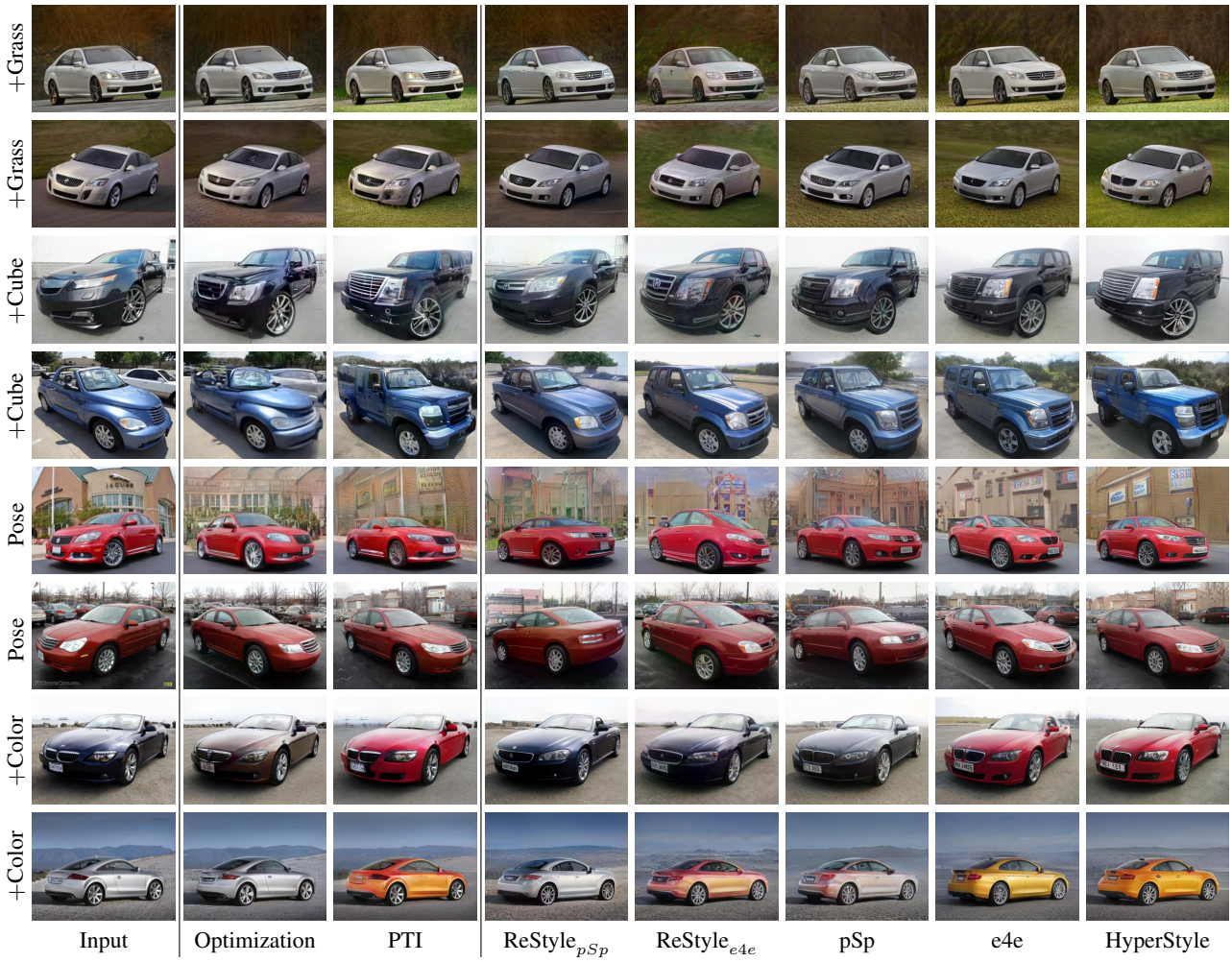


Figure 10. Additional editing comparisons over the cars domain obtained using GANSpace [6].

Figure 11. Additional editing results obtained with HyperStyle over the cars domain obtained using GANSpace [6].

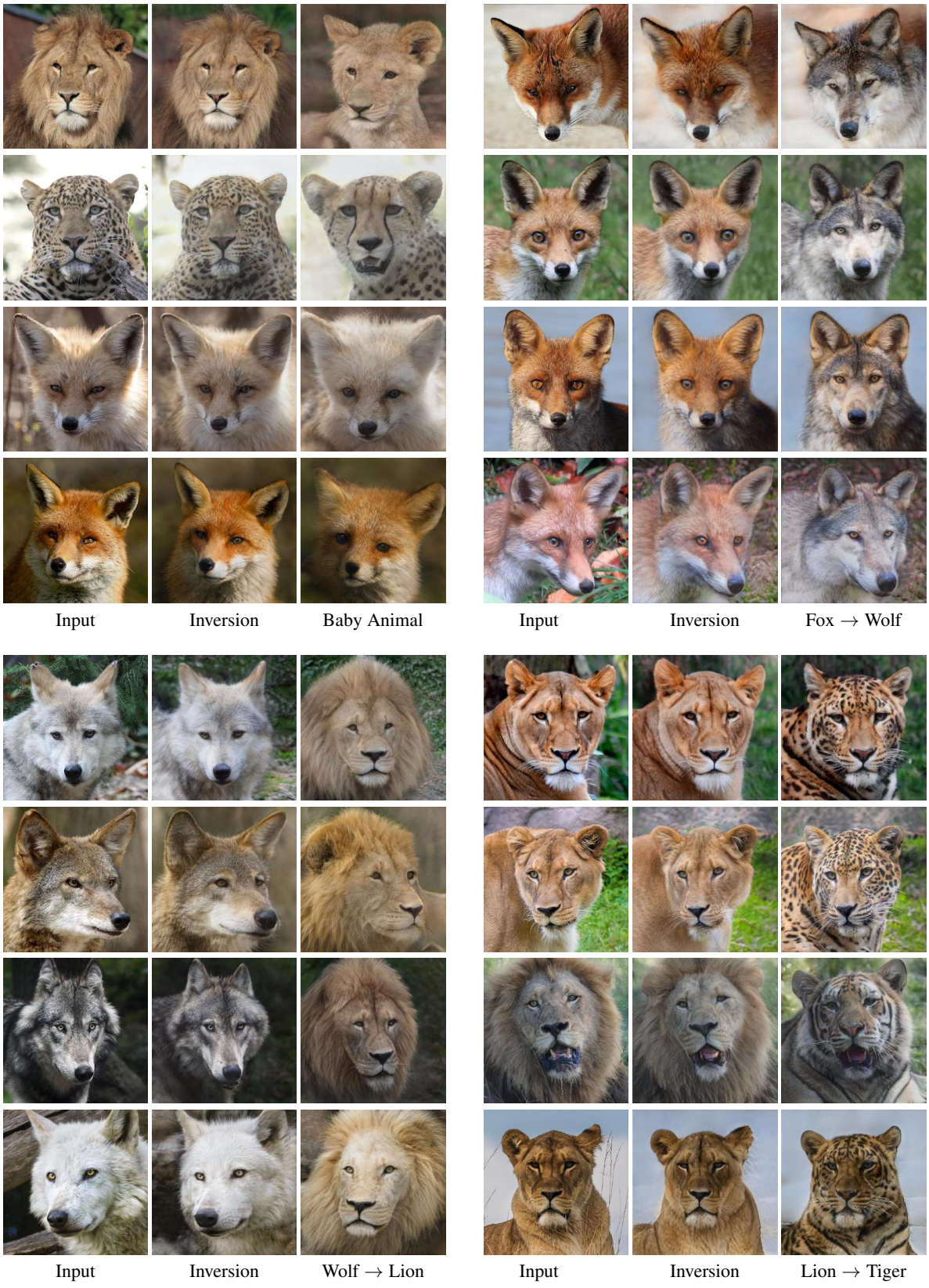


Figure 12. Reconstruction and editing results obtained with HyperStyle on the AFHQ Wild [3] test set. Edits were obtained using StyleCLIP's [18] global directions approach.

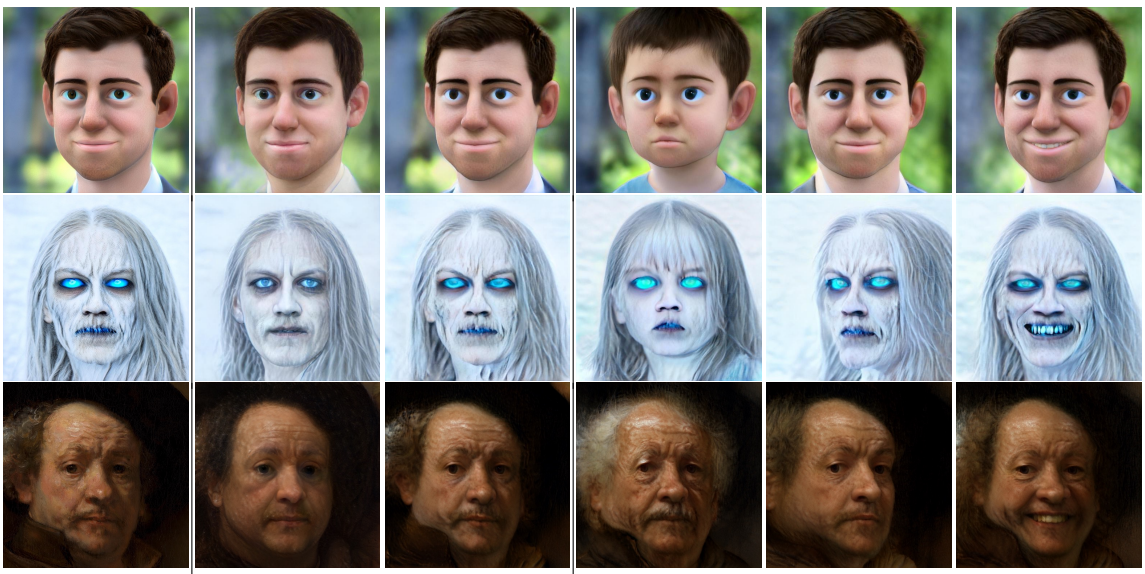
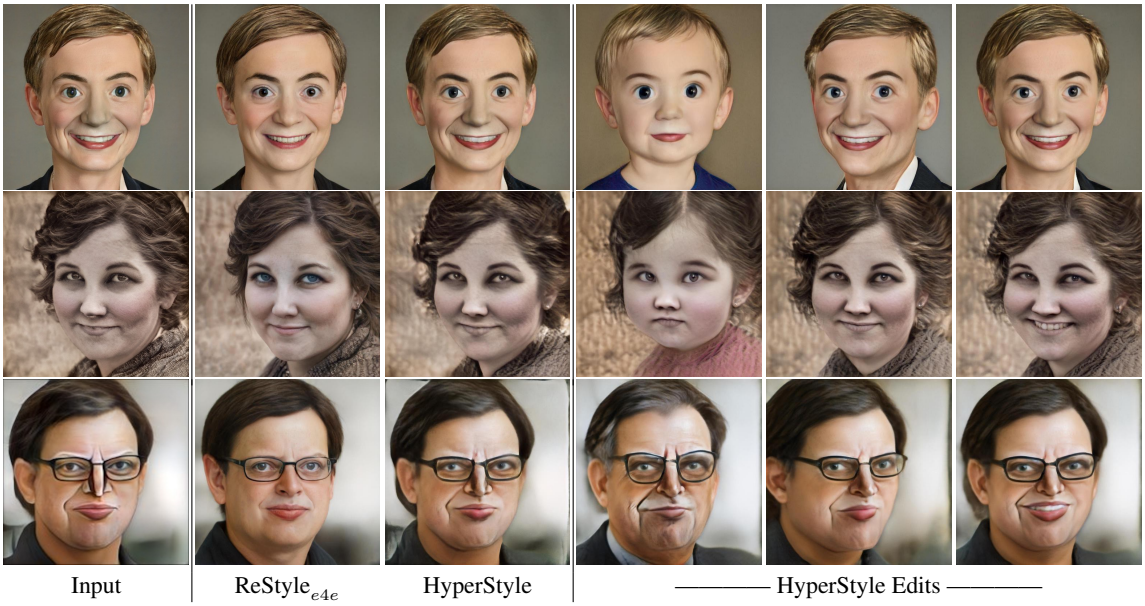


Figure 13. Reconstruction and editing results obtained on challenging input styles not observed during training. All reconstructions and editing results are obtained with a hypernetwork and StyleGAN generator trained on the FFHQ [10] dataset. Note, some of the input images were generated from a StyleGAN model fine-tuned with StyleGAN-NADA [5]. Even in such cases, prior encoders struggle in accurately reconstructing the input.



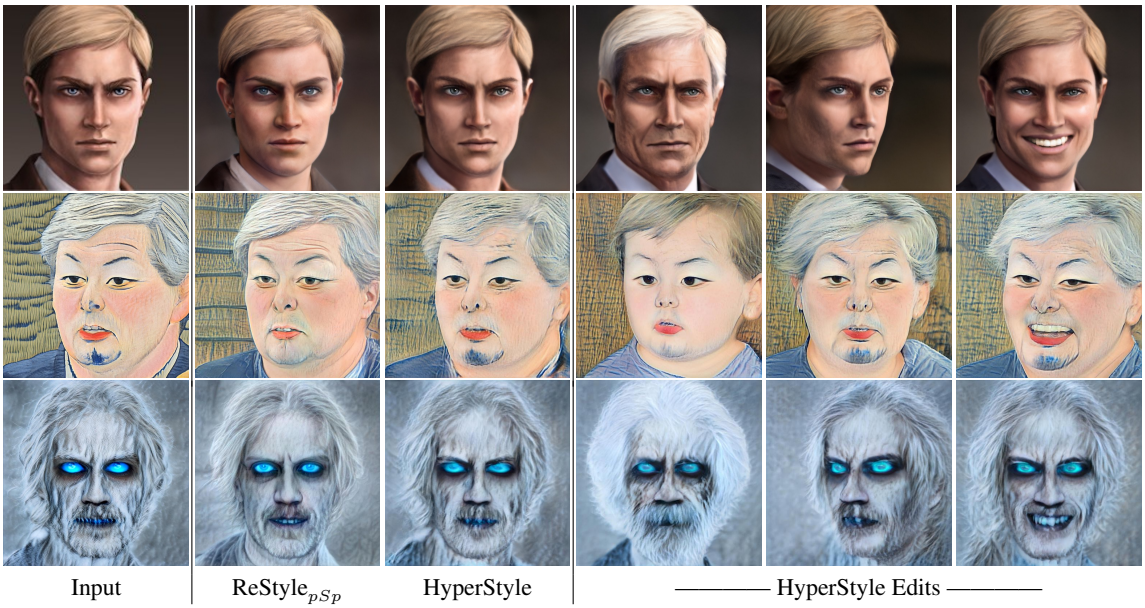


Figure 14. Reconstruction and editing results obtained on challenging input styles unobserved during training, using the same settings as in Fig. 13.

Figure 15. Additional domain adaptation comparisons for various fine-tuned models such as Toonify [19] and those obtained using StyleGAN-NADA [5].