Pastiche Master: Exemplar-Based High-Resolution Portrait Style Transfer

Shuai Yang Liming Jiang Ziwei Liu Chen Change Loy
S-Lab, Nanyang Technological University
{shuai.yang, liming002, ziwei.liu, ccloy}@ntu.edu.sg

Supplementary Material

Contents

1. Implementation Details of DualStyleGAN **2**
1.1. Dataset and Model 2
1.2. Network Architecture 2
1.3. Network Training 3
1.4. Experiment Details of Simulating Fine-Tuning Behavior 4

2. Supplementary Experimental Results of DualStyleGAN **5**
2.1. Comparison with State-of-the-Art Methods 5
2.1.1 Visual comparison 5
2.1.2 User study 8
2.2. Comparison with StyleCariGAN 9
2.3. Arbitrary Artistic Portrait Generation 10
2.4. Performance on Other Styles 13
2.4.1 Performance on Datasets With About One Hundred Images 13
2.4.2 Effect of the Regularization Term on Extremely Limited Data 16
2.5. Effect of Batch Size 17
2.6. Style Blending 18

1. Implementation Details of DualStyleGAN

1.1. Dataset and Model

Dataset. Cartoon dataset [16] has 317 images, which is provided at https://github.com/justinpinkney/toonify without claiming licenses. We use 199 images from WebCaricature [7, 8] to build the Caricature dataset. Figure 1 gives an overview of the Caricature dataset. WebCaricature provides the copyright "The dataset is released for research and educational purposes. We hold no liability for any undesirable consequences of using the dataset. All rights of the WebCaricature Dataset are reserved. Any person or organization is not permitted to distribute, redistribute, publish, copy or disseminate this dataset." We use 174 images from Danbooru Portraits [1] to build the Anime dataset. Figure 2 gives an overview of the Anime dataset. We follow the FFHQ [11] to align caricature and anime faces based on facial landmarks. The landmarks of caricature faces are included in WebCaricature. The landmarks of anime faces are manually labelled by us. Danbooru Portraits are provided at https://www.gwern.net/Crops#danbooru2019-portraits without claiming licenses. CelebA-HQ [10] is under CC BY-NC 4.0 License. FFHQ [11] is made available under CC BY-NC-SA 4.0 License by NVIDIA Corporation.

Figure 1. An overview of Caricature dataset.

Model. We build our model based on the PyTorch version of StyleGAN (https://github.com/rosinality/stylegan2-pytorch) under MIT License. The StyleGAN Encoder of pSp [17], GNR [3] and U-GAT-IT [12] are under MIT License. StarGAN2 [2] is under CC BY-NC 4.0 License. UI2I-style [14] is under CC BY-NC-SA 4.0 License. Toonify [16] is provided at https://github.com/justinpinkney/toonify without claiming licenses. Few-Shot Adaptation (FS-Ada) [15] is under Adobe Research License. The sampling network GLANN [5] is under BSD 3-Clause License.

1.2. Network Architecture

There are 7 ModRes blocks in the extrinsic style path, corresponding to one 4×4, two 8×8, two 16×16 and two 32×32 convolution layers in StyleGAN. Each ModRes has a ResBlock and an AdaIN block. Each ResBlock has two convolution layers with 3×3 kernels. The dimension of the style code of AdaIN is the same as the intermediate style code $\mathbf{w}$ of StyleGAN, *i.e.*, 512. The structure transfer block T_s in the extrinsic style path is made up of two linear layers. There are 11 color transform block T_c in the extrinsic style path, corresponding to two 64×64, two 128×128, two 256×256, two 512×512, two 1024×1024 convolution layers and one 1024×1024 ToRGB layer in StyleGAN. T_c is made up of a linear layer.

Figure 2. An overview of Anime dataset.

1.3. Network Training

Destylization – Stage II uses one NVIDIA Tesla V100 GPU, and optimizes the latent code for 300 iterations, which processes about 45 images per hour.

Progressive fine-tuning – Stage II uses one NVIDIA Tesla V100 GPU and a batch size of 4 per GPU with $\lambda_{\text{adv}} = 0.1, \lambda_{\text{perc}} = 0.5$. To calculate λ_{perc}, we use the conv1_1, conv2_1 and conv3_1 layers of the VGG19 [18] with equal weights of 1. We train on $l = 7, 6, 5$ for 300, 300, 3000 iterations, respectively, Stage II takes about 1.8 hour. Note that once trained, the pre-trained model can be applied to any style dataset.

Progressive fine-tuning – Stage III uses eight NVIDIA Tesla V100 GPUs and a batch size of 4 per GPU with $\lambda_{\text{adv}} = 1, \lambda_{\text{perc}} = 1, \lambda_{\text{CX}} = 0.25, \lambda_{\text{FM}} = 0.25$. To calculate λ_{perc}, we use the conv2_1 and conv3_1 layers of the VGG19 [18] weighted by 0.5 and 1, respectively We set $(\lambda_{\text{ID}}, \lambda_{\text{reg}})$ to $(1, 0.015)$, $(4, 0.005)$, $(1, 0.02)$ and trains for 1400, 1000, 2100 iterations on cartoon, caricature and anime, respectively. Training takes about 0.75 hour on average.

Post-processing. Latent optimization and training sampling network use one GPU. Post-processing optimizes the latent for 100 iterations, taking about 0.46 hour per 100 artistic portrait images. We adopt the Adam optimizer. The first 7 rows of $\mathbf{z}_e^+$ use a small learning rate of 0.005, 0.005, 0.0005 to prevent violent structure changes for cartoon, caricature and anime, respectively. The last 11 rows of $\mathbf{z}_e^+$ use a learning rate of 0.1, 0.01, 0.005 for color refinement for cartoon, caricature and anime, respectively. Training two sampling networks takes about 0.13 hour.

1.4. Experiment Details of Simulating Fine-Tuning Behavior

As analyzed in Sec. 3.2 of the paper, StyleGANs before and after fine-tuning have shared latent spaces [14] and closely-related convolution features. Therefore, the difference of these convolution features is also closely-related to the original convolution features. It is possible to keep all other submodules fixed but only learn changes over the convolution features to simulate the changes of the convolution weight matrices in fine-tuning, which naturally corresponding to a residual path. To simulate the unconditional fine-tuning over StyleGAN, we follow three principles to design the residual path:

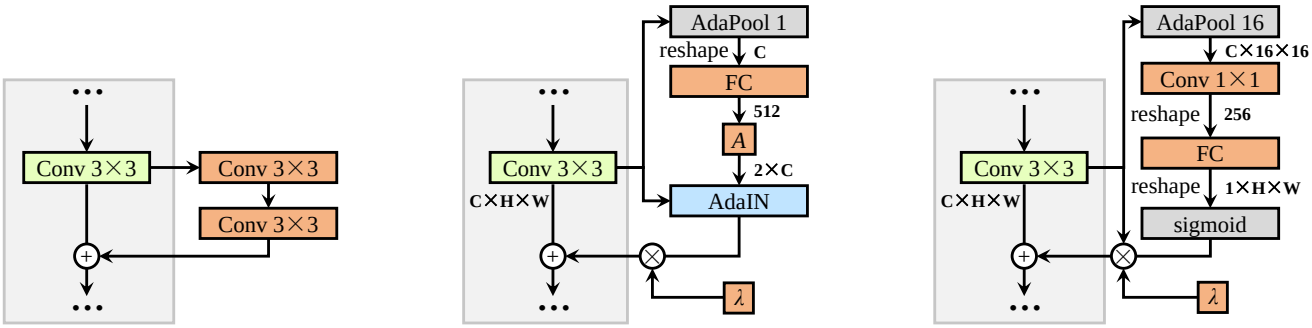
- **Principle I**: The residual path has few impact on the feature at the beginning of fine-tuning in order to make the pre-trained generative space unaltered.
- **Principle II**: The residual path is conditioned by the input convolution features from StyleGAN, considering the residual features should be closely-related to the original convolution features.
- **Principle III**: The residual path is not conditioned by the external style images, since in this experiment we would like to only simulate the unconditional fine-tuning over StyleGAN rather than learning a style transfer task.

Based on the above principles, we compare three modules for the building of the residual path: channel-wise AdaIN [6], spatial-wise Diagonal Attention (DAT) [13] and element-wise ResBlock [4], with the network architectures:

- **ResBlock**: Standard residual blocks with two 3×3 convolution layers. The convolution weight matrices are set to values close to 0 to meet Principle I.
- **AdaIN**: To meet Principles II and III, we extract the channel-wise style code from the input convolution feature by global average pooling and a linear layer rather than using the external style code. The style code then goes through an affine transform block A for adaptive instance normalization. To meet Principle I, the modulated feature is multiplied with a learnable weight λ, which is set to a small value of 0.01 initially, before added to the input feature.
- **DAT**: To meet Principles II and III, we extract the spatial-wise style code from the input convolution feature via an adaptive pooling layer and an 1×1 convolution layer rather than using the external style code. The one-channel style code goes through a linear layer and a sigmoid layer to obtain the attention map as the original DAT. The attention map is multiplied with the input feature as well as the learnable weight λ, which is set to a small value of 0.01 initially, before added to the input feature.

Figure 3 intuitively summarizes the above three network architectures to simulate fine-tuning behavior of StyleGAN.

In the experiment of Sec. 3.2 of the paper, we finetune StyleGAN for 900 iterations on Cartoon dataset. The resulting StyleGAN serves as a ground truth model. Then, we keep the original StyleGAN fixed, and only train three kinds of residual paths for 900 iterations on Cartoon dataset. Finally, we compare the performance of the three model against the ground truth model in cartoon face generation from the same random latent code, and find that channel-wise or spatial-wise modulations alone are not enough to approximate the fine-tuning behavior. ResBlocks achieve the most similar results to those by the ground truth model. Therefore, we choose residual blocks to build the residual path in the paper.



(a) ResBlock (b) AdaIN (c) DAT

Figure 3. Network architectures used to simulate fine-tuning behavior.

2. Supplementary Experimental Results of DualStyleGAN

2.1. Comparison with State-of-the-Art Methods

2.1.1 Visual comparison

In addition to the examples shown in the main paper, we show more visual comparison results in Figs. 4-6.

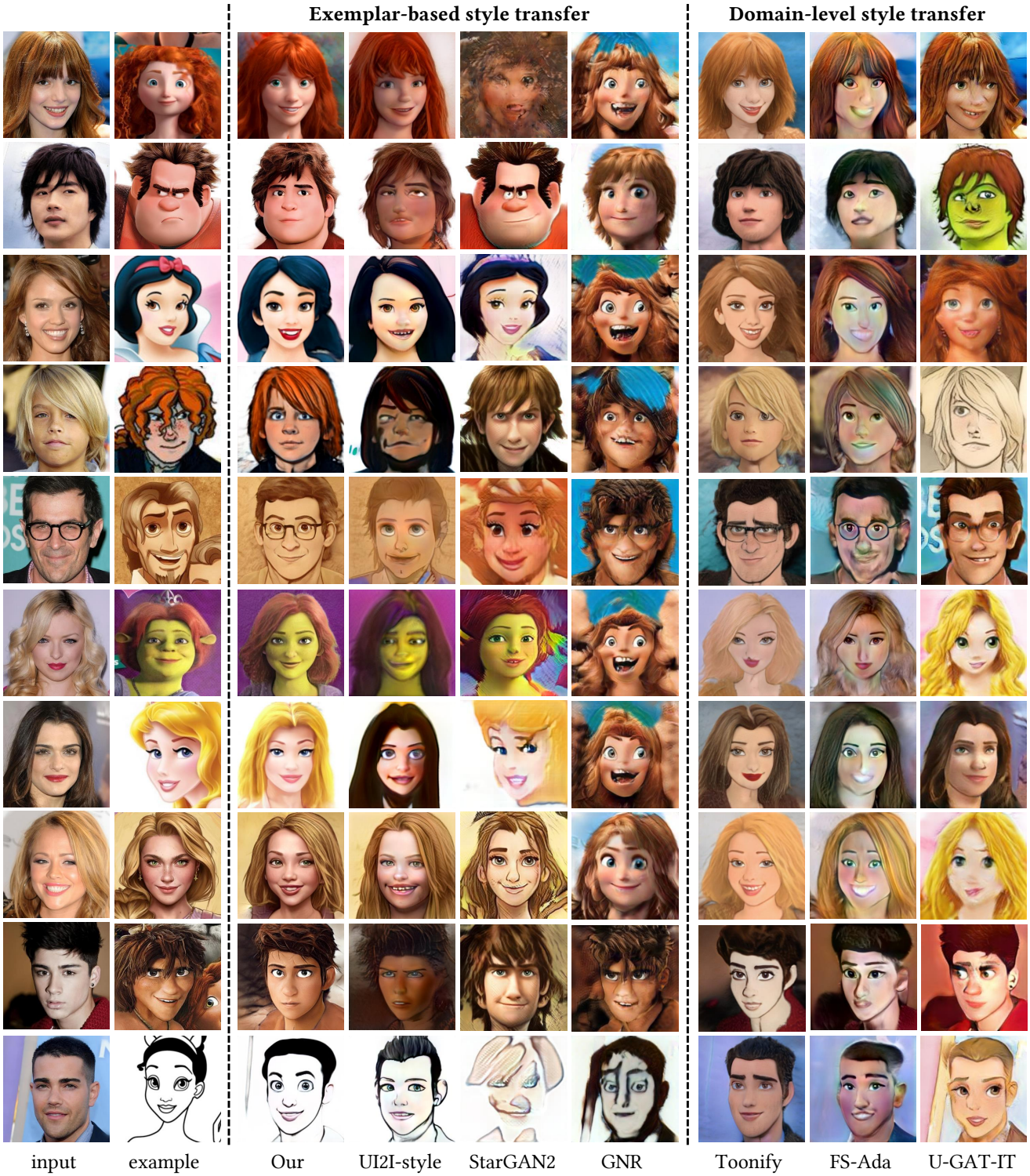


Figure 4. Visual comparison on cartoon face style transfer.

Figure 5. Visual comparison on caricature face style transfer.

Figure 6. Visual comparison on anime face style transfer.

2.1.2 User study

We conduct a user study to evaluate the overall style transfer performance. Users are asked to select the best style transfer result in terms of both content preservation and style consistency with the reference style images. A total of 27 subjects participate in this study to select the best ones from the results of four methods. A total of 810 selections on 30 groups of results (Figs. 4-6) are tallied. Table 1 demonstrates the average user scores of each group and the whole 30 groups, where the proposed method receives notable preference.

Table 1. User preference ratio of state-of-the-art methods. The best score in each row is marked in bold.

Style	ID	DualStyleGAN	UI2I-style [14]	StarGANv2 [2]	GNR [3]
Cartoon	1	**0.96**	0.04	0.00	0.00
	2	**1.00**	0.00	0.00	0.00
	3	**0.93**	0.00	0.07	0.00
	4	**0.96**	0.00	0.04	0.00
	5	**1.00**	0.00	0.00	0.00
	6	**1.00**	0.00	0.00	0.00
	7	**0.89**	0.11	0.00	0.00
	8	**0.96**	0.04	0.00	0.00
	9	**0.85**	0.04	0.04	0.07
	10	**0.74**	0.26	0.00	0.00
Caricature	1	**0.78**	0.19	0.00	0.04
	2	**0.93**	0.00	0.00	0.07
	3	**0.74**	0.04	0.00	0.22
	4	**0.52**	0.48	0.00	0.00
	5	**0.96**	0.00	0.00	0.04
	6	**0.78**	0.19	0.00	0.04
	7	**0.89**	0.07	0.00	0.04
	8	**0.63**	0.33	0.00	0.04
	9	**0.81**	0.11	0.00	0.07
	10	**0.81**	0.07	0.04	0.07
Anime	1	**0.70**	0.07	0.22	0.00
	2	**0.96**	0.04	0.00	0.00
	3	**0.67**	0.26	0.00	0.07
	4	**0.33**	0.30	0.11	0.26
	5	**0.89**	0.07	0.00	0.04
	6	**0.74**	0.22	0.04	0.00
	7	**0.81**	0.19	0.00	0.00
	8	**0.85**	0.15	0.00	0.00
	9	**0.85**	0.15	0.00	0.00
	10	**1.00**	0.00	0.00	0.00
Average		**0.83**	0.11	0.02	0.04

2.2. Comparison with StyleCariGAN

The officially release model of StyleCariGAN [9] is trained on 6K images to produce 256×256 images. The reason might be it is difficult to train cycle translation with limited high-resolution images. As for style control, StyleCariGAN uses style mixing for exemplar-based style control, which is limited to color control like UI2I-style [14]. The cycle translation of StyleCariGAN is not conditioned on example. Thus, it learns an overall structure transfer rather than exemplar-based structure transfer. In [9], StyleCariGAN shows exemplar-based structure transfer results by mixing the structure style at the low-resolution layers, which is however at the cost of lost identity and artifacts.

In the main paper, we compare with StyleCariGAN, where the performance of StyleCariGAN degrades when using style codes from example images. The reason might be that the style code is optimized in $\mathcal{W}+$ space as in [9] and is out-of-distribution, while the style codes in official style palette are directly drawn from the standard latent distribution. Figure 7 presents more visual comparison results on the content images from https://github.com/wonjongg/StyleCariGAN. For StyleCariGAN, we only use style codes from official style palette to prevent degradation. For DualStyleGAN, we use the style codes from our Caricature dataset. It can be seen that both methods render diverse and plausible textures and colors. Our method surpasses StyleCariGAN in transferring diverse structure styles.

Figure 7. Random anime faces generated by DualStyleGAN

2.3. Arbitrary Artistic Portrait Generation

In Figs. 8–10, we show artistic portraits generated by DualStyleGAN from random intrinsic and extrinsic style codes. The generative space of DualStyleGAN is of high diversity and high quality.

Figure 8. Random cartoon portraits generated by DualStyleGAN.

Figure 9. Random caricature portraits generated by DualStyleGAN.

Figure 10. Random anime portraits generated by DualStyleGAN

2.4. Performance on Other Styles

2.4.1 Performance on Datasets With About One Hundred Images

In Fig. 11, we show additional style transfer results in styles of Pixar. The training on Pixar dataset of 122 images takes 1000 iterations with $(\lambda_{\text{adv}}, \lambda_{\text{perc}}, \lambda_{\text{CX}}, \lambda_{\text{FM}}, \lambda_{\text{ID}}, \lambda_{\text{reg}}) = (1, 1, 0.25, 0.25, 1, 0.015)$, and does not apply the post-processing of latent optimization to refine the color.

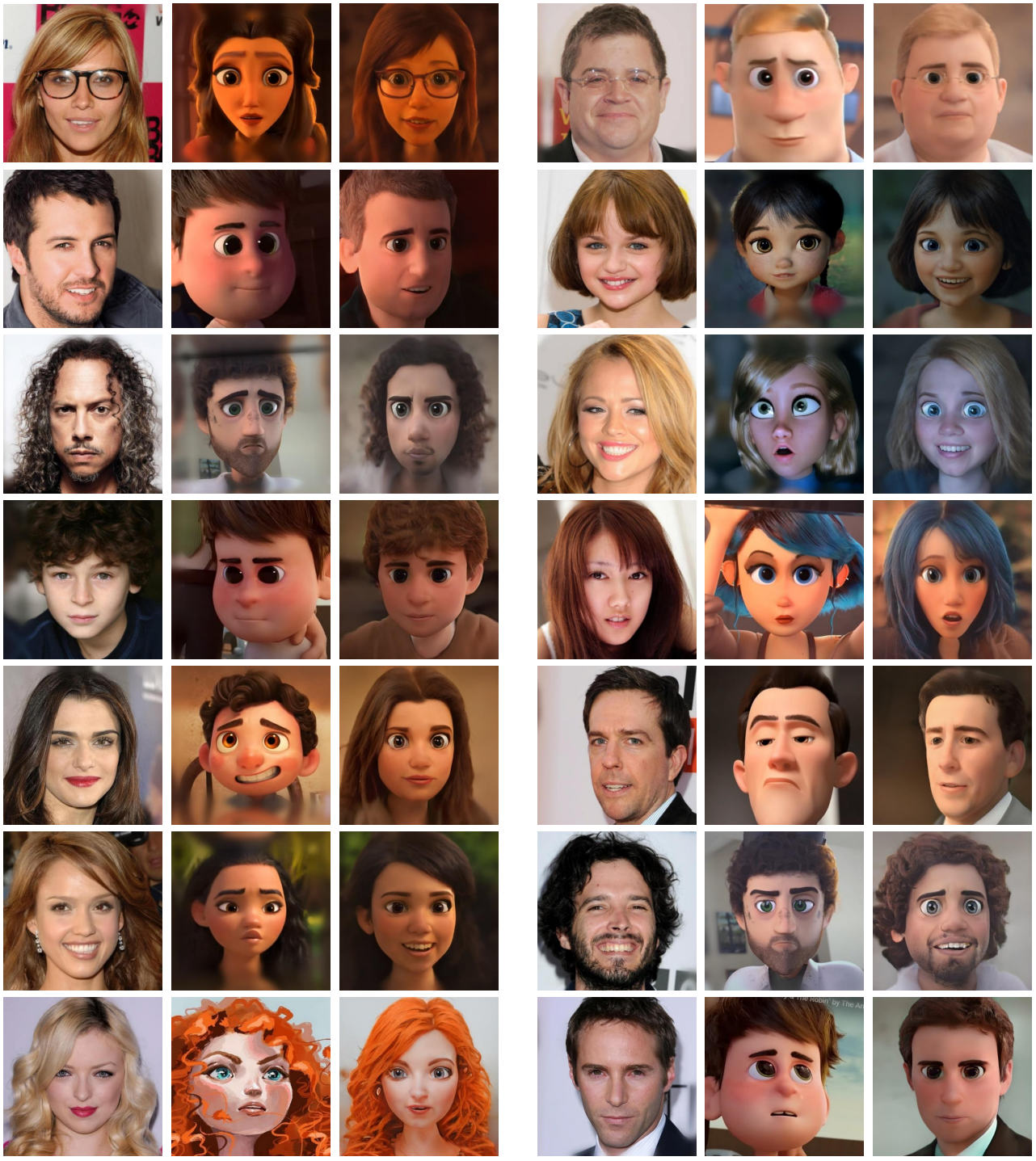


Figure 11. Performance on Pixar styles.

In Fig. 12, we show additional style transfer results in styles of Comic. The training on Comic dataset of 101 images takes 1000 iterations with $(\lambda_{\text{adv}}, \lambda_{\text{perc}}, \lambda_{\text{CX}}, \lambda_{\text{FM}}, \lambda_{\text{ID}}, \lambda_{\text{reg}}) = (1, 1, 0.25, 0.25, 1, 0.015)$, and does not apply the post-processing of latent optimization to refine the color.

Figure 12. Performance on Comic styles.

In Fig. 12, we show additional style transfer results in styles of Slam Dunk. The training on Slam Dunk dataset of 120 images takes 1500 iterations with $(\lambda_{\text{adv}}, \lambda_{\text{perc}}, \lambda_{\text{CX}}, \lambda_{\text{FM}}, \lambda_{\text{ID}}, \lambda_{\text{reg}}) = (1, 1, 0.25, 0.25, 4, 0.02)$, and does not apply the post-processing of latent optimization to refine the color.

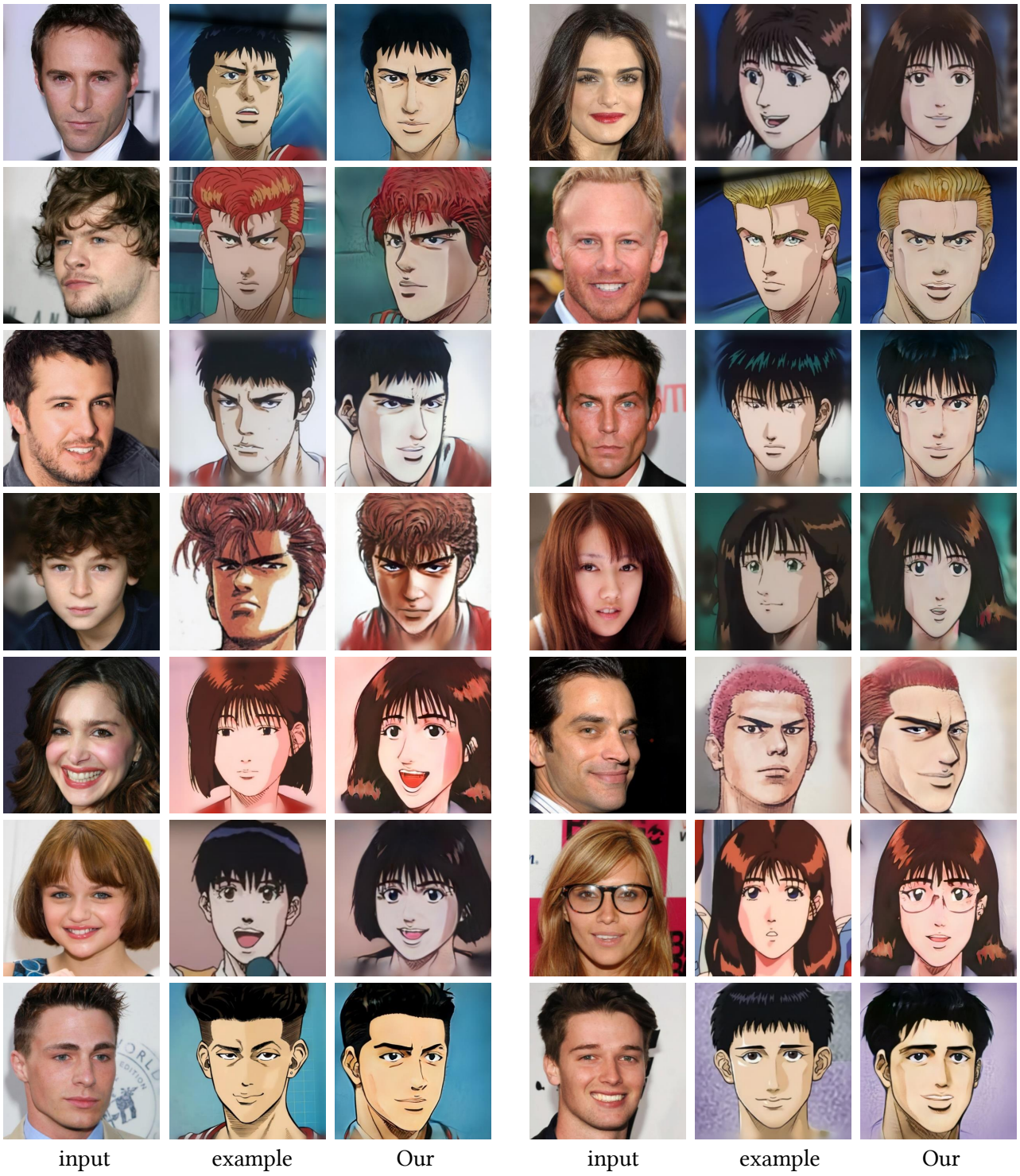


Figure 13. Performance on Slam Dunk styles.

2.4.2 Effect of the Regularization Term on Extremely Limited Data

We investigate the potential of our method in handling extremely limited data. We collected sixteen cartoon portrait images as shown in the top of Fig. 14 and use them with data augmentation of horizontal flip to train DualStyleGAN.

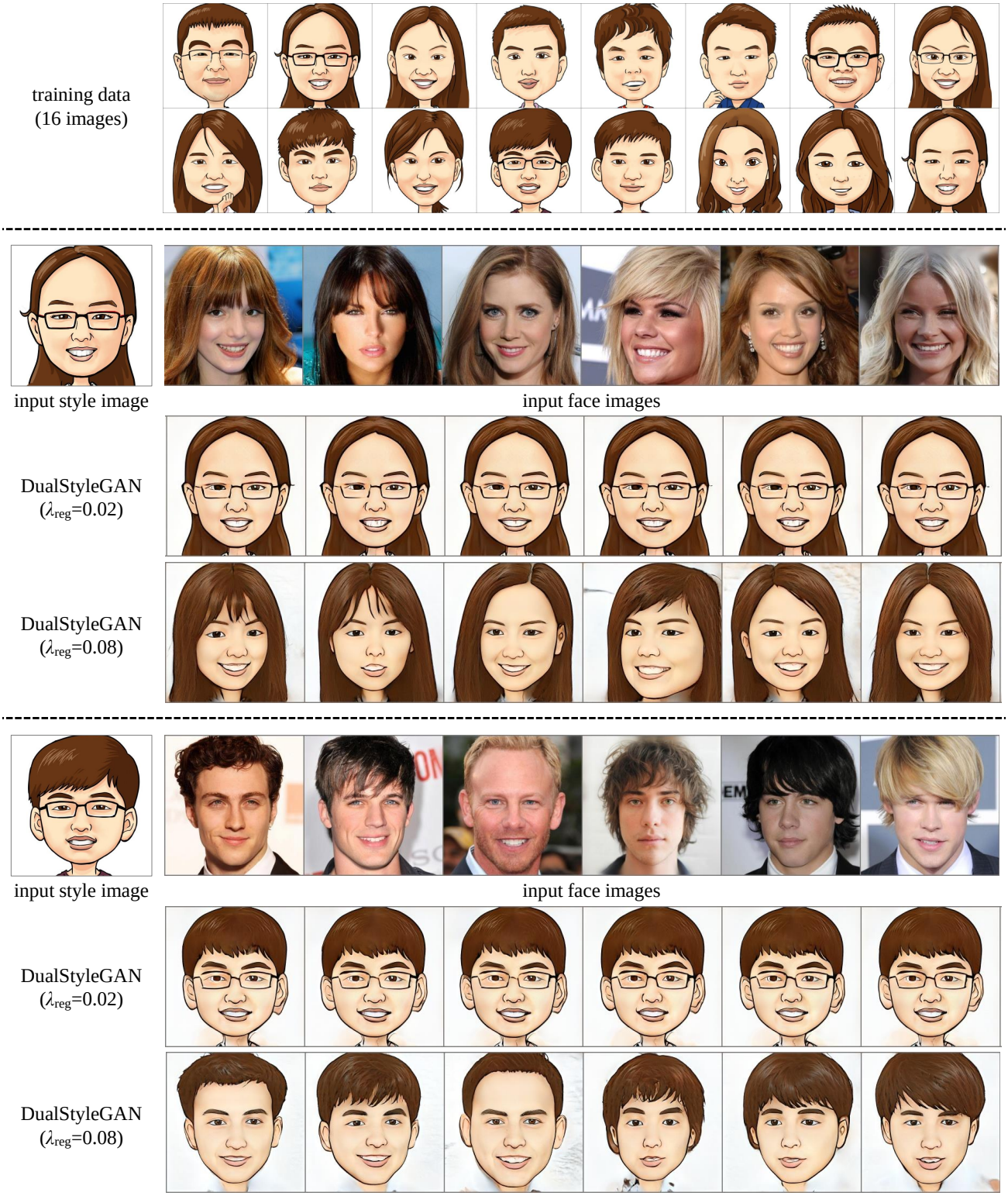


Figure 14. Overview of the sixteen-image cartoon dataset and performance on this dataset with different λ_{reg}.

The extremely limited data makes it very easy to fall into mode collapse. We tested on several hyper-parameter configurations, and found $(\lambda_{\text{adv}}, \lambda_{\text{perc}}, \lambda_{\text{CX}}, \lambda_{\text{FM}}, \lambda_{\text{ID}}, \lambda_{\text{reg}}) = (1, 0.1, 0.05, 0.05, 4, 0.08)$ made the fine-tuning relatively more stable, and prevented DualStyleGAN from just memorizing these sixteen cartoon portrait images. The final results are shown in Fig. 14, where we set $\mathbf{w} = [4 * 1, 3 * 0.5, 11 * 1]$. We also compare with the results with $\lambda_{\text{reg}} = 0.02$. It can be seen that when using a small $\lambda_{\text{reg}} = 0.02$, DualStyleGAN ignores the input face image and just memorizes the training images. By setting $\lambda_{\text{reg}} = 0.08$, the regularization term forces our model to use as few resources to shift the generative space as possible, so that it successfully learns to fully utilize the pre-trained facial features to help transfer and yields reasonable style transfer results. However, the data size still has a large impact on the performance. Compared to datasets with hundreds of images, there are many artifacts when using extremely limited data.

2.5. Effect of Batch Size

We have also tried fine-tuning on a singe GPU with a batch size of 4. We find training with one GPU is less stable than that with eight GPUs, and requires about twice or triple the iterations (3500, 3000, 4500 for cartoon, caricature and anime, respectively). We achieve good results with $(\lambda_{\text{adv}}, \lambda_{\text{perc}}, \lambda_{\text{CX}}, \lambda_{\text{FM}}, \lambda_{\text{ID}}, \lambda_{\text{reg}}) = (1, 2.25, 0.25, 0.25, 1, 0.02)$, $(1, 1.0, 0.25, 0.25, 2, 0.01)$ and $(1, 0.25, 0.25, 2.5, 1, 0.04)$ on cartoon, caricature and anime with one GPU, respectively. However, as shown in Fig. 15, the results using a small batch size have more artifacts than that using a large batch size. Therefore, a large batch size is recommended.

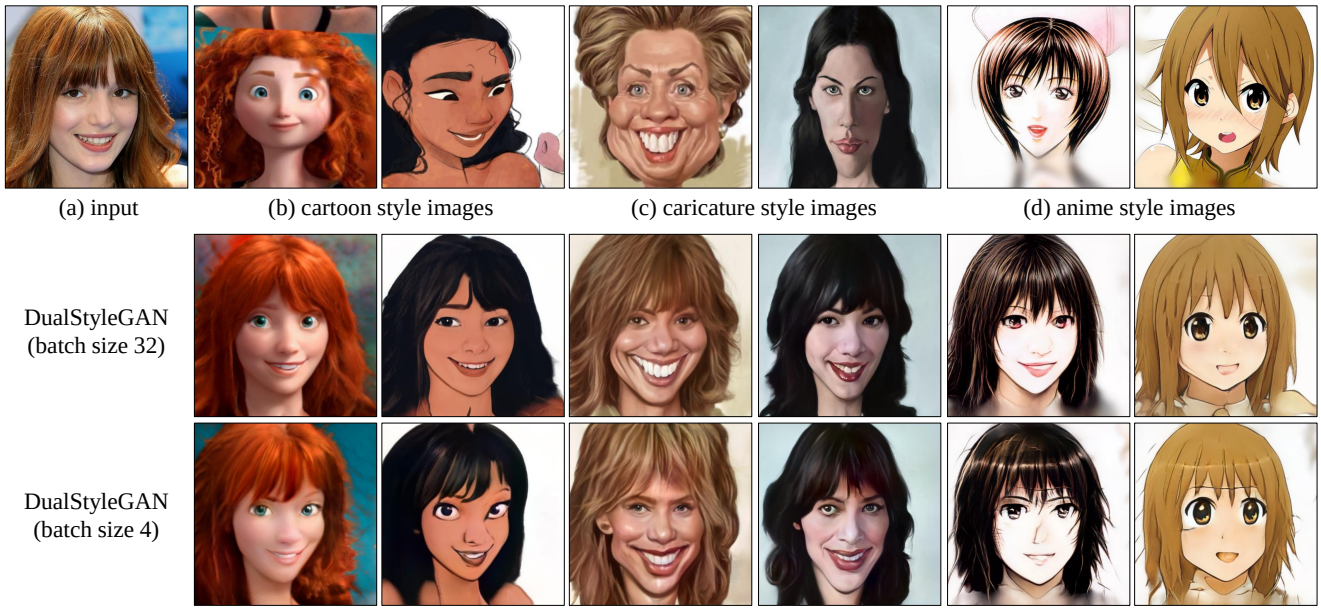


Figure 15. Compare style transfer results of DualStyleGAN trained with a batch size of 32 and 4.

2.6. Style Blending

In Figs. 16-18, we perform a linear interpolation to style feature, and observe smooth changes along with the latent space from one to another.

Figure 16. Style blending on cartoon portraits. The anchor styles are marked by red boxes.

Figure 17. Style blending on caricature portraits. The anchor styles are marked by red boxes.

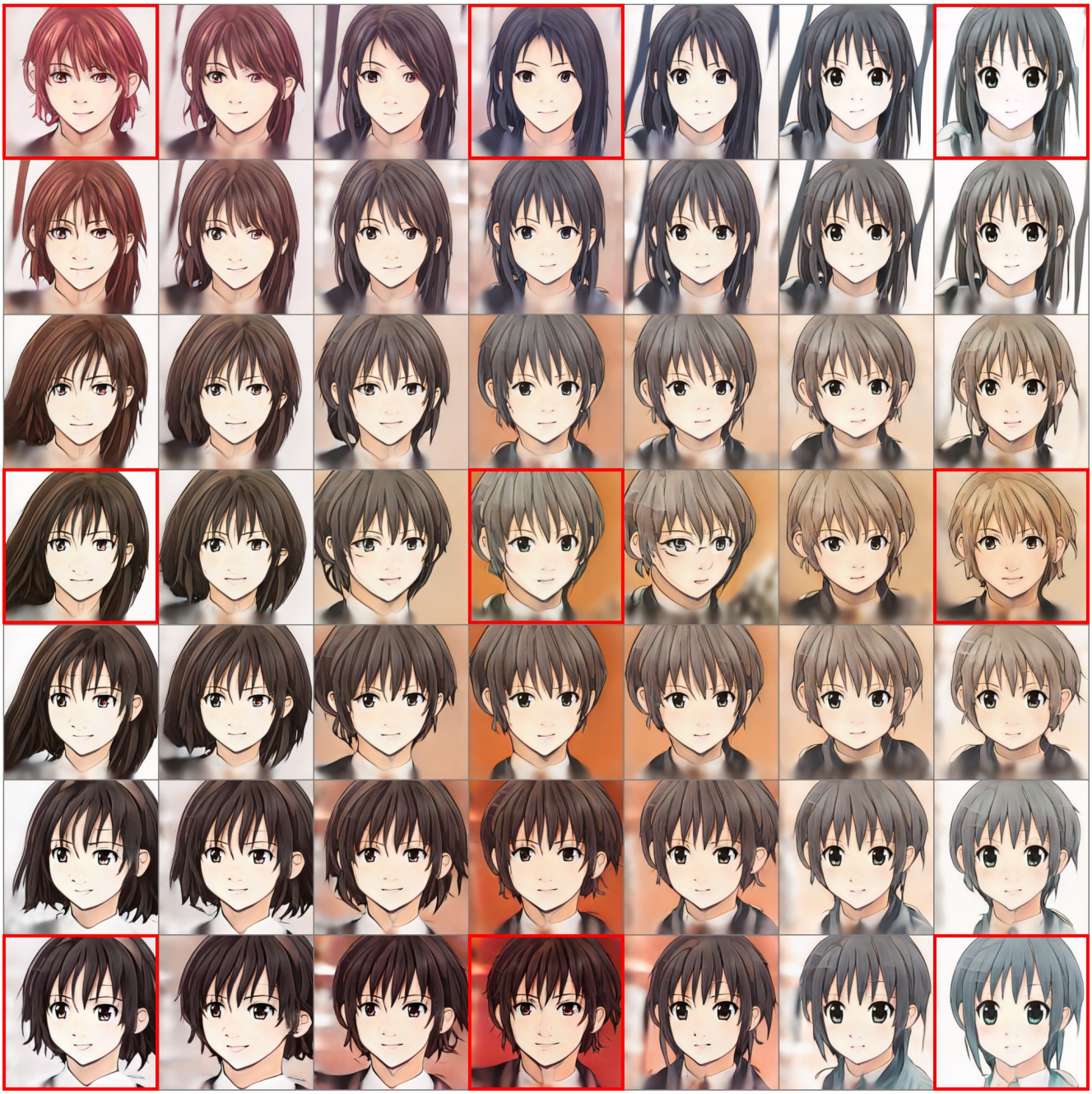


Figure 18. Style blending on anime portraits. The anchor styles are marked by red boxes.

References

[1] Gwern Branwen, Anonymous, and Danbooru Community. Danbooru2019 portraits: A large-scale anime head illustration dataset. https://www.gwern.net/Crops#danbooru2019-portraits, March 2019. 2

[2] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, pages 8188–8197, 2020. 2, 8

[3] Min Jin Chong and David Forsyth. GANs N' Roses: Stable, controllable, diverse image to image translation. *arXiv preprint arXiv:2106.06561*, 2021. 2, 8

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, pages 770–778, 2016. 4

[5] Yedid Hoshen, Ke Li, and Jitendra Malik. Non-adversarial image synthesis with generative latent nearest neighbors. In *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, pages 5811–5819, 2019. 2

[6] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proc. Int'l Conf. Computer Vision*, pages 1510–1519, 2017. 4

[7] Jing Huo, Yang Gao, Yinghuan Shi, and Hujun Yin. Variation robust cross-modal metric learning for caricature recognition. In *Proc. Thematic Workshops of ACM Int'l Conf. Multimedia*, pages 340–348, 2017. 2

[8] Jing Huo, Wenbin Li, Yinghuan Shi, Yang Gao, and Hujun Yin. Webcaricature: a benchmark for caricature recognition. In *Proc. British Machine Vision Conference*, 2018. 2

[9] Wonjong Jang, Gwangjin Ju, Yucheol Jung, Jiaolong Yang, Xin Tong, and Seungyong Lee. StyleCariGAN: caricature generation via stylegan feature map modulation. *ACM Transactions on Graphics*, 40(4):1–16, 2021. 9

[10] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *Proc. Int'l Conf. Learning Representations*, 2018. 2

[11] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, pages 4401–4410, 2019. 2

[12] Junho Kim, Minjae Kim, Hyeonwoo Kang, and Kwang Hee Lee. U-GAT-IT: Unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation. In *Proc. Int'l Conf. Learning Representations*, 2019. 2

[13] Gihyun Kwon and Jong Chul Ye. Diagonal attention and style-based gan for content-style disentanglement in image generation and translation. In *Proc. Int'l Conf. Computer Vision*, 2021. 4

[14] Sam Kwong, Jialu Huang, and Jing Liao. Unsupervised image-to-image translation via pre-trained stylegan2 network. *IEEE Transactions on Multimedia*, 2021. 2, 4, 8, 9

[15] Utkarsh Ojha, Yijun Li, Jingwan Lu, Alexei A Efros, Yong Jae Lee, Eli Shechtman, and Richard Zhang. Few-shot image generation via cross-domain correspondence. In *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, pages 10743–10752, 2021. 2

[16] Justin NM Pinkney and Doron Adler. Resolution dependent gan interpolation for controllable image synthesis between domains. *arXiv preprint arXiv:2010.05334*, 2020. 2

[17] 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 *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 2021. 2

[18] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proc. Int'l Conf. Learning Representations*, 2015. 3