Appendix of
On Leveraging Pretrained GANs for Generation with Limited Data

Miaoyun Zhao, Yulai Cong, Lawrence Carin
Department of ECE, Duke University

A. Experimental Settings

For all experiments if not specified, we inherit the experimental settings of GP-GAN (Mescheder et al., 2018). All training images are resized to 128×128 for consistency. For the discriminator, gradient penalty on real samples (R_1-regularizer) is adopted with the regularization parameter $\gamma = 10.0$. We use the Adam optimizer with learning rate 1×10^{-4} and coefficients $(\beta_1, \beta_2) = (0.0, 0.99)$. $60,000$ iterations are used for training. Because of limited computation power, we use a batch size of 16. For the CelebA dataset, the dimension of the latent vector $\boldsymbol{z}$ is set to 256; while for the small datasets (*i.e.,* Flowers, Cars, and Cathedral) and their 1K variants, that dimension is set to 64.

The experimental settings for the extremely limited datasets, *i.e.,* Flowers-25 and FFHQ-25, are provided in Appendix B.

The FID scores are calculated based on $10,000/8, 189/8, 144/7, 350$ real and generated images on CelebA, Flowers, Cars, and Cathedral, respectively. The same FID calculations are employed for experiments on the corresponding 1K variants.

For Scratch, we employ the same architecture as our method without the $\boldsymbol{\gamma}/\boldsymbol{\beta}$ AdaFM parameters, because $\boldsymbol{\gamma}$s and $\boldsymbol{\beta}$s are now redundant if we train the source filter $\mathbf{W}$ (refer to (2) of the main manuscript).

Regarding the training of our method, we fix the scale $\boldsymbol{\gamma} = 1$ and shift $\boldsymbol{\beta} = 0$ in the first 10,000 iterations and only update the tailored specific part for a stable initialization; after that, we jointly train both the $\boldsymbol{\gamma}/\boldsymbol{\beta}$ AdaFM parameters and the specific part to deliver the presented results.

A.1. On Specifying the General Part of the Discriminator for Transfer

To figure out the suitable general part of the discriminator to be transferred from the pretrained GP-GAN model to the target CelebA dataset, we design a series of experiments with increasing number of lower groups included/frozen in the transferred/frozen general part; the remaining high-level specific part is reinitialized and trained with CelebA. The architecture for the discriminator with the D2 general part is shown in Figure 15, as an illustrative example.

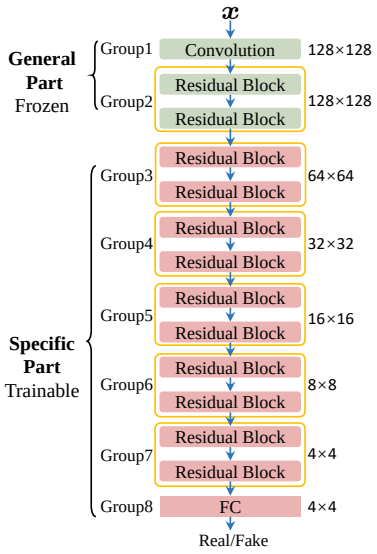

Figure 15. The architecture for the discriminator with the D2 general part. The overall architecture is inherited from the GP-GAN.

Based on the transferred general part of the generator and discriminator, we next reinitialize and train the remaining specific part on CelebA. The employed reinitialization are as follows.

- For all layers except FC in the generator/discriminator, we use the corresponding parameters from the pretrained GP-GAN model as initialization.

- Regarding FC layers in generator/discriminator, since the pretrained GP-GAN model on ImageNet used a conditional-generation architecture (*i.e.,* the input of the generator FC layer consists both the noise $\boldsymbol{z}$ and the label embedding $\boldsymbol{y}$, whereas the discriminator FC layer has multiple heads (each corresponds to one class)), we can not directly transfer the FC parameters therein to initialize our model (without labels). Consequently, we randomly initialize both FC layers in the generator and discriminator.

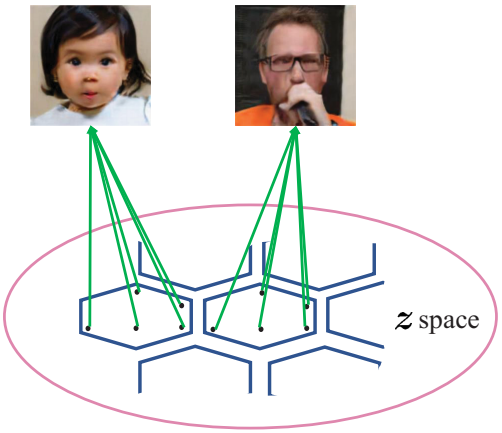

Figure 16. Demonstration of the generative process learned on extremely limited data. Since the continuous latent $\boldsymbol{z}$-space is likely redundant, the generator often maps close latent vectors to similar outputs.

B. On Generation with Extremely Limited Data with 25 Samples

Considering the challenging settings with extremely limited data in quantity (*i.e.,* 25 data samples), we transfer the G4D6 general part from the pretrained GP-GAN model (termed Our-G4D6) and apply GP (gradient penalty) on both real and fake samples with the regularization parameter $\gamma = 20$. The dimension of the latent vector $\boldsymbol{z}$ is set to 4. 60,000 training iterations are used.

The FID scores are calculated following (Noguchi & Harada, 2019). For Flowers-25, 251 real passion images and 251 generated images are used to calculate the FID; for FFHQ-25, 10,000 real face images and 10,000 generated images are used.

Since the target data (25 samples) are extremely limited, we find that the generator managed to learn a generative mapping that captures the generation over the 25 training samples, as illustrated in Figure 16. As the latent $\boldsymbol{z}$-space is continuous and likely redundant for the extremely limited data, the learned generator often maps close latent vectors to a similar output. Regarding the interpolations shown in Figure 12 of the main paper, we use an amplification process (see Figure 17) to get the presented results. Note Figure 17 also empirically verifies the above intuition. The demonstrated results are as expected, because, on one hand, only 25 training images are available, while on the other hand, the gradient penalty applied to discriminator (in addition to regularizations from the proposed techniques) implicitly imposes smoothness to the output space of the generator.

C. More Analysis and Discussions

C.1. On the Worse FID of G2D0 than That of G4D0

The worse FID of G2D0 is believed caused by the insufficiently trained low-level filters, which are time-consuming and data-demanding to train. Specifically, by taking a close look at the generated samples, we find that

- there are generated samples that look similar to each other, indicating a relatively low generative diversity;
- most of the generated samples contain strange textures that look like water spots, as shown in Figure 18.

Such phenomena are deemed to have negative effects on the FID score.

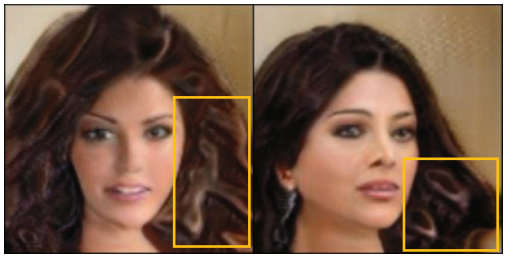

Figure 18. Samples generated from a model with the G2D0 general part. Water-spot shaped textures appear in the hair area (see the yellow boxes).

C.2. On Selecting the Optimal GmDn with the Best Generalization

Supplementing Figures 3 and 4 of the main paper, we evaluate various settings of GmDn for the transfered general part, with the same experimental settings of Section 3 of the main paper. The corresponding FID scores are summarized in Figure 20, where one may observe interesting patterns of the transfer properties of the pretrained GAN model.

- It seems that the performance is, in general, more sensitive to the setting of Gm than that of Dn, meaning that the generator general part may play a more important role in generation tasks.
- Besides, it's clear that compromise arises in both Gm and Dn directions; this is expected as the low-level filters are more generally applicable while the high-level ones are more domain specific.
- Moreover, it seems interesting correlations exist between Gm (generator general part) and Dn (discriminator general part), which might be worthy of future explorations.

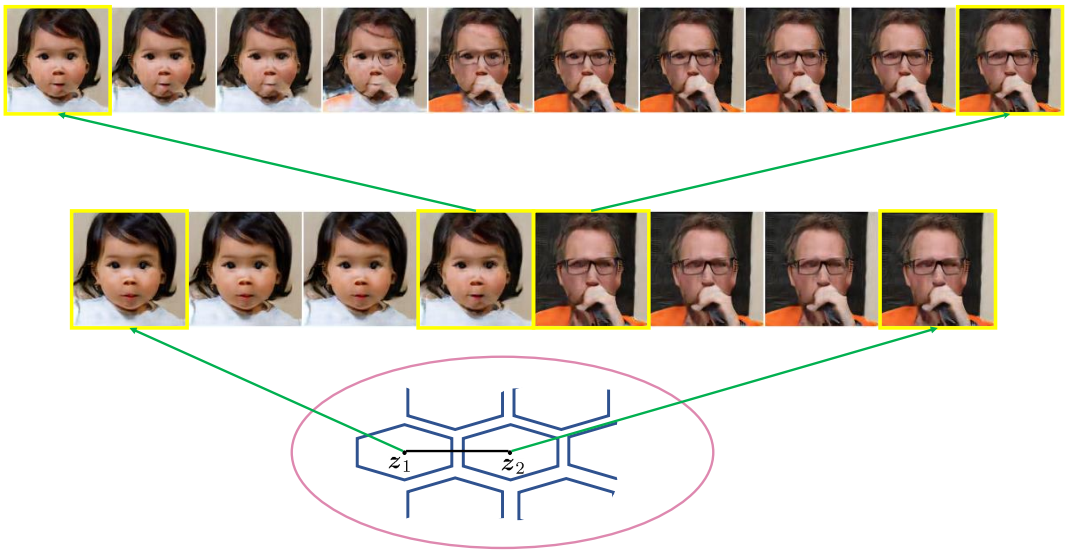

Figure 17. The amplification process employed to yield the smooth interpolations for our method.

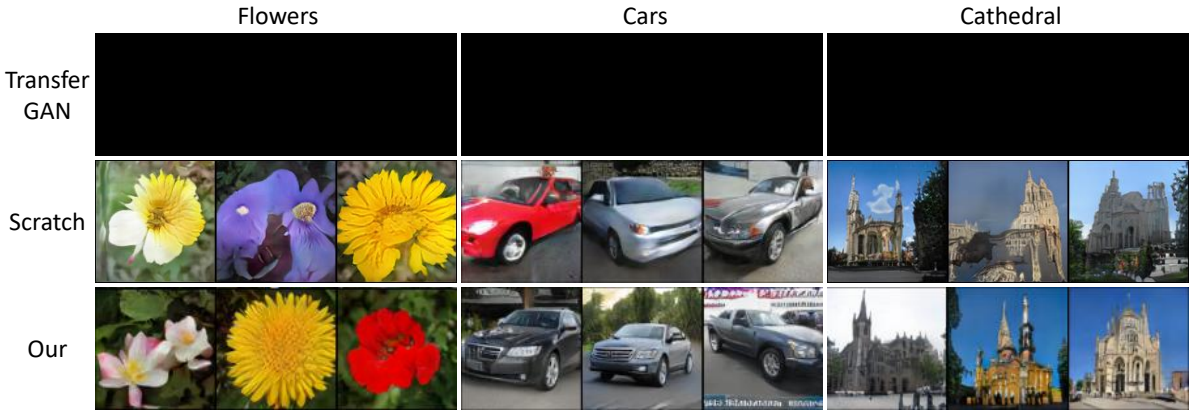

Figure 19. TransferGAN shows mode/training collapse on the three small datasets.

- Finally, the G4D2 general part delivers the best performance among the tested settings, justifying its adoption in the main paper.

It's extremely challenging (if not impossible) to choose the optimal GmDn general part that generalizes well to various target domains (beyond CelebA). To alleviate this concern, we'd like to point out that our AdaFM may greatly relax the requirement of an optimal GmDn, as reflected by our-G4D2's boosted performance on various target datasets. Empirically, we find that the G4 generator general part works reasonably well for most situations; the setting for the discriminator general part (*i.e.,* Dn) is more data-size related, *e.g.,* the D2 setting may be suitable for a target domain with $\geq$7K data samples, D5 for $\approx$1K samples, and D6 for $\approx$25 samples.

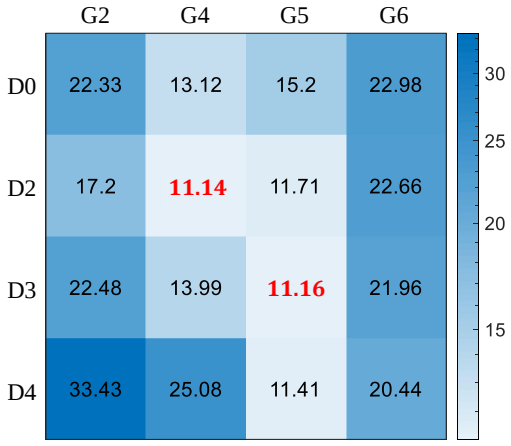

Figure 20. FID scores from various settings for GmDn. GmDn indicates freezing the lower m/n groups as the general part of generator/discriminator. It's clear that the adopted G4D2 general part is a reasonable choice for transfer.

C.3. On Figure 8 of the Main Manuscript

Regarding the FID curves shown in Figure 8 of the main manuscript, one may concern the final performance of Scratch if it's trained for long time. To address that concern, we run Scratch on Flowers for 150,000 iterations and find that Scratch converges to a worse FID=18.1 than our method at 60,000 iterations. Similar phenomena also manifest in Figure 10 of the main manuscript. These results empirically prove that the transferred/frozen general part deliver both improved training efficiency and better final performance.

Regarding more generated samples for comparing our method with Scratch, we show in Figure 21, 22 and 23 more randomly generated samples from our method and Scratch as a supplementary for Figure 8 of the main manuscript. Thanks to the transferred low-level filters and the better adaption to target domain via AdaFM, our method shows a much higher generation quality than Scratch.

C.4. On the Failing of TransferGAN on the Three Small Datasets

Since the FC layers from the pretrained GP-GAN model on ImageNet is not directly applicable to the target domain (as discussed in Section A.1), we implement the TransferGAN method by initializing parameters of all layers (except the FC layer, whose parameters are randomly initialized) with the corresponding parameters from the pretrained GP-GAN model. A similar architecture of the pretrained GP-GAN model is therefore employed for TransferGAN.

When only a small (or limited) amount of training data are available, (*e.g.,* on the three small datasets: Flowers, Cars, and Cathedral), TransferGAN is prone to overfitting because of its large amount of trainable parameters. Specifically, the number of the trainable parameters within the TransferGAN generator is 96.1M; by comparison, the generator of our method only contains 24.4M trainable parameters. Accordingly, TransferGAN suffers from mode/training collapse on the three small datasets, as shown in Figure 19.

C.5. Early Stopping for Generation with Limited Data

Concerning early stopping for generation with limited data, we find that the discriminator loss may be leveraged for that goal, as overfitting empirically manifests as a decreasing discriminator loss in our setup.

Specifically, with the GP-GAN settings, we empirically find that the discriminator loss stables roughly within $[0.8, 1.3]$ when the training is successful without clear overfitting. However, if the discriminator loss falls into $[0.5, 0.7]$ and remains there for a period, overfitting likely starts arising; accordingly, that may be a proper time for early stopping.

C.6. On Figure 13(c) of the Main Manuscript

Figure 13(c) shows the sorted demonstration of the learned $\boldsymbol{\gamma}$ from the last convolutional layer in Group2. We sort the $\boldsymbol{\gamma}$'s learned on different target datasets as follows.

1. Reshape each $\boldsymbol{\gamma}$ matrix into a vector; stack these vectors into a matrix $\mathbf{M}$, so that each row represents the $\boldsymbol{\gamma}$ from a specific target dataset;

2. Clip all the values of $\mathbf{M}$ to $[0.9, 1.1]$ and then re-scale to $[0, 1]$ for better contrast;

3. For the i-th row/target-dataset, find the set of column indexes $\boldsymbol{s}'_i = \{j | \forall k \neq i, \mathbf{M}_{i,j} - \mathbf{M}_{k,j} > 0.03\}$; sort $\boldsymbol{s}'_i$ according to the values $\mathbf{M}_{i,\boldsymbol{s}_i}$ to yield $\boldsymbol{s}_i$;

4. Concatenate $\{\boldsymbol{s}_i\}$ with the remaining column indexes to yield the sorted indexes $\boldsymbol{t}$; sort the columns of the matrix $\mathbf{M}$ according to $\boldsymbol{t}$ to deliver the presented matrix in Figure 13(c).

C.7. Comparison of AdaFM and Weight Demodulation

Supplementing Section 3.3 of the main manuscript, we further compare the weight demodulation (Karras et al., 2019b) with our AdaFM below.

Recall that AdaFM uses learnable matrix parameters $\{\boldsymbol{\gamma}, \boldsymbol{\beta}\}$ to modulate/adapt a transfered/frozen convolutional filter (see (2) of the main manuscript); by comparison, the weight demodulation uses $\boldsymbol{\beta} = 0$ and rank-one $\boldsymbol{\gamma} = \boldsymbol{\eta}\boldsymbol{s}^T$, where $\boldsymbol{s}$ is parametrized as a neural network to control style and $\boldsymbol{\eta}$ calculated based on $\boldsymbol{s}$ and the convolutional filter $\mathbf{W}$ (see (3) of the main manuscript). Therefore, a direct comparison of AdaFM and the weight demodulation is not feasible.

On one hand, it's interesting to consider generalizing our AdaFM with neural-network-parameterized $\{\boldsymbol{\gamma}, \boldsymbol{\beta}\}$ for better adaptation of the transfered general part, for introduction of conditional information, or for better generation like in StyleGAN2; we leave that as future research. On the other hand, if we degrade the weight demodulation by setting $\boldsymbol{s}$ as a learnable vector parameter, the weight demodulation may work similar to FS (see the comparison between FS and AdaFM in Figure 7 of the main manuscript), because both of them have almost the same flexibility.

D. Medical/Biological Applications with Gray-Scale Images

Concerning medical/biological applications with gray-scale images, we conduct experiments on a gray-scale variant of Cathedral, termed gray-Cathedral. The randomly generated samples are shown in Figure 24. Obviously, without AdaFM, worse (blurry and messy) details are observed in the gener-

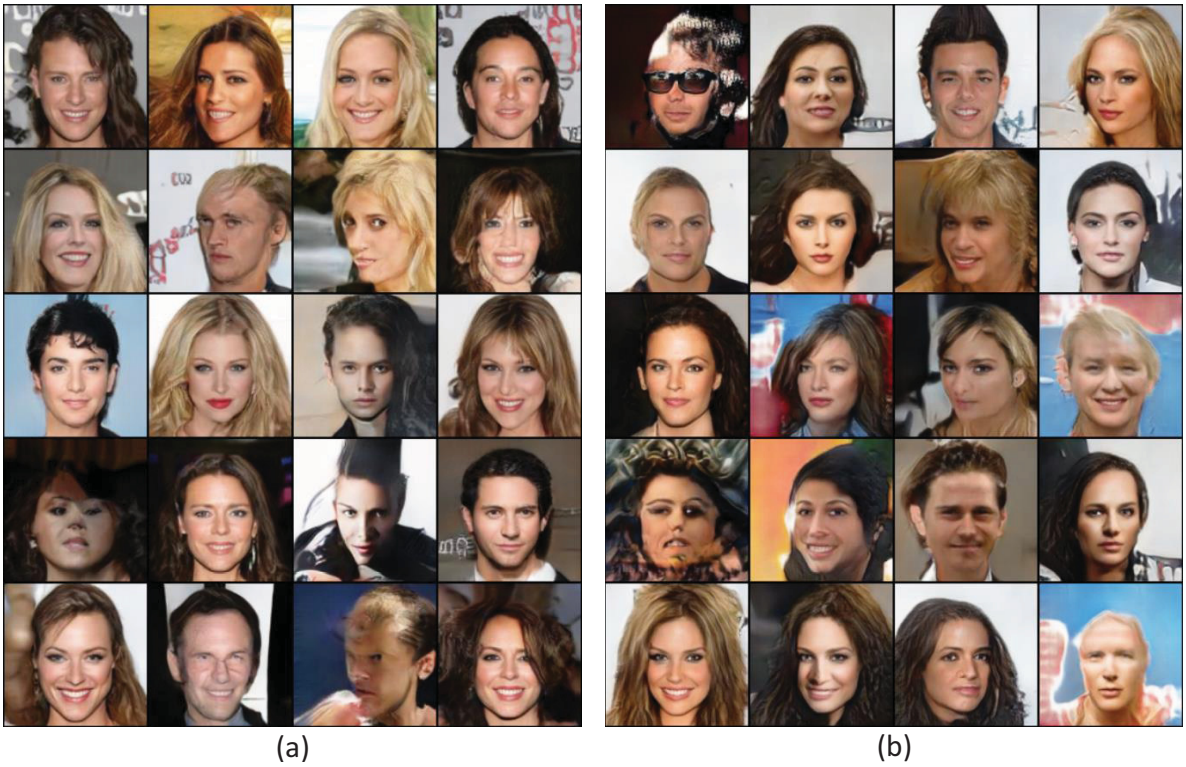

Figure 21. More generated samples on CelebA, supplementing Figure 8 of the main manuscript. (a) Our; (b) Scratch.

ated images, likely because of the mismatched correlation among channels between source and target domains.

E. Contributions of the Proposed AdaFM and the Transferred General Part

To demonstrate the contribution of the proposed AdaFM, the randomly generated samples from our method (with AdaFM) and SmallHead (without AdaFM) on different target domains are shown in Figures 25, 26, and 27, respectively. Note that the only difference between our method and SmallHead is the use of AdaFM. It's clear that, with AdaFM, our method delivers significantly improved generation quality over the Smallhead.

It is worth noting that on all these perceptually-distinct target datasets (*e.g.,* CelebA, Flowers, Cathedral), the proposed SmallHead with the transferred general part has proven to train successfully and delivers diverse and relatively-realistic generations, despite without modulations from AdaFM. Such phenomena prove that the G4D2 general part discovered in Section 3.1 of the main manuscript generalizes well to various target domains.

F. Style Mixing on Flowers and CelebA

The style-mixing results shown in Figure 12 of the main manuscript are obtained as follows.

Following (Karras et al., 2019a), given the generative process of a "source" image, we replace its style input of Group 5[5] (the arrow on the left of the specific part of our model; see Figure 1(h)) with that from a "Destination" image, followed by propagating through the rest of the generator to generate a new image with mixed style.

A similar style mixing is conducted on CelebA, with the results shown in Figure 28. We observe that the style inputs from the "Source" images control the identity, posture, and hair type, while the style inputs from the "Destination" images control the sex, color, and expression.

[5]We choose Group 5 for example demonstration; one can of course control the input to other groups, or even hierarchically control all of them.

Figure 22. More generated samples on Flowers, supplementing Figure 8 of the main manuscript. (a) Our; (b) Scratch.

Figure 23. More generated samples on Cathedral, supplementing Figure 8 of the main manuscript. (a) Our; (b) Scratch.

(a) Our method with AdaFM

(b) SmallHead without AdaFM

Figure 24. Randomly generated samples on gray-Cathedral, supplementing Section 4.4 in the main manuscript. Better viewed with zoom in.

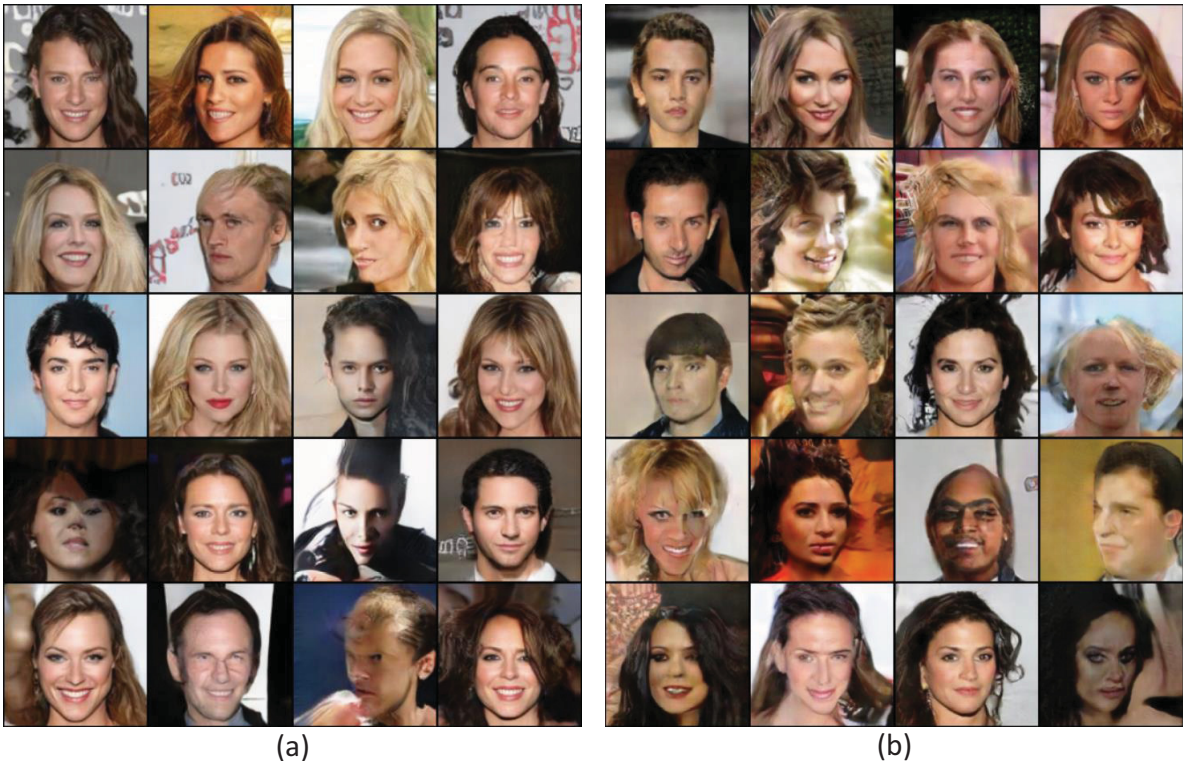

Figure 25. Randomly generated samples from our method and SmallHead on CelebA. (a) Our (with AdaFM); (b) SmallHead (without AdaFM).

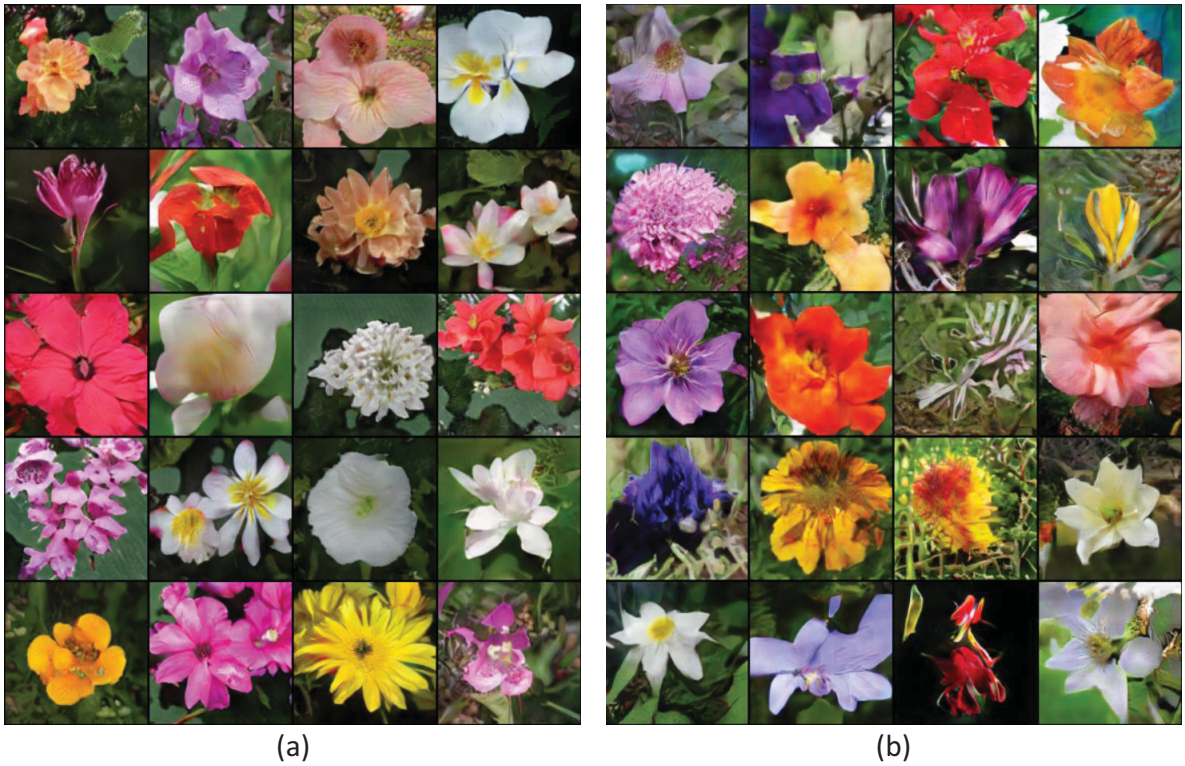

Figure 26. Randomly generated samples from our method and SmallHead on Flowers. (a) Our (with AdaFM); (b) SmallHead (without AdaFM).

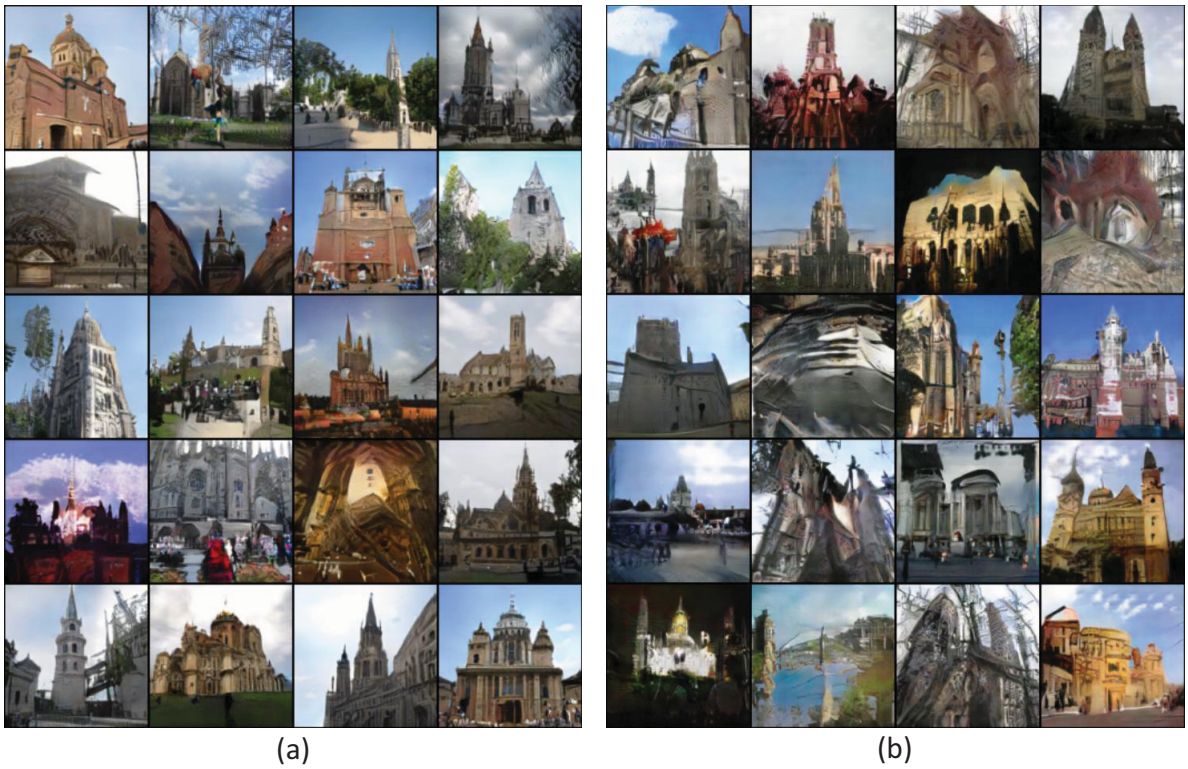

Figure 27. Randomly generated samples from our method and SmallHead on Cathedral. (a) Our (with AdaFM); (b) SmallHead (without AdaFM).

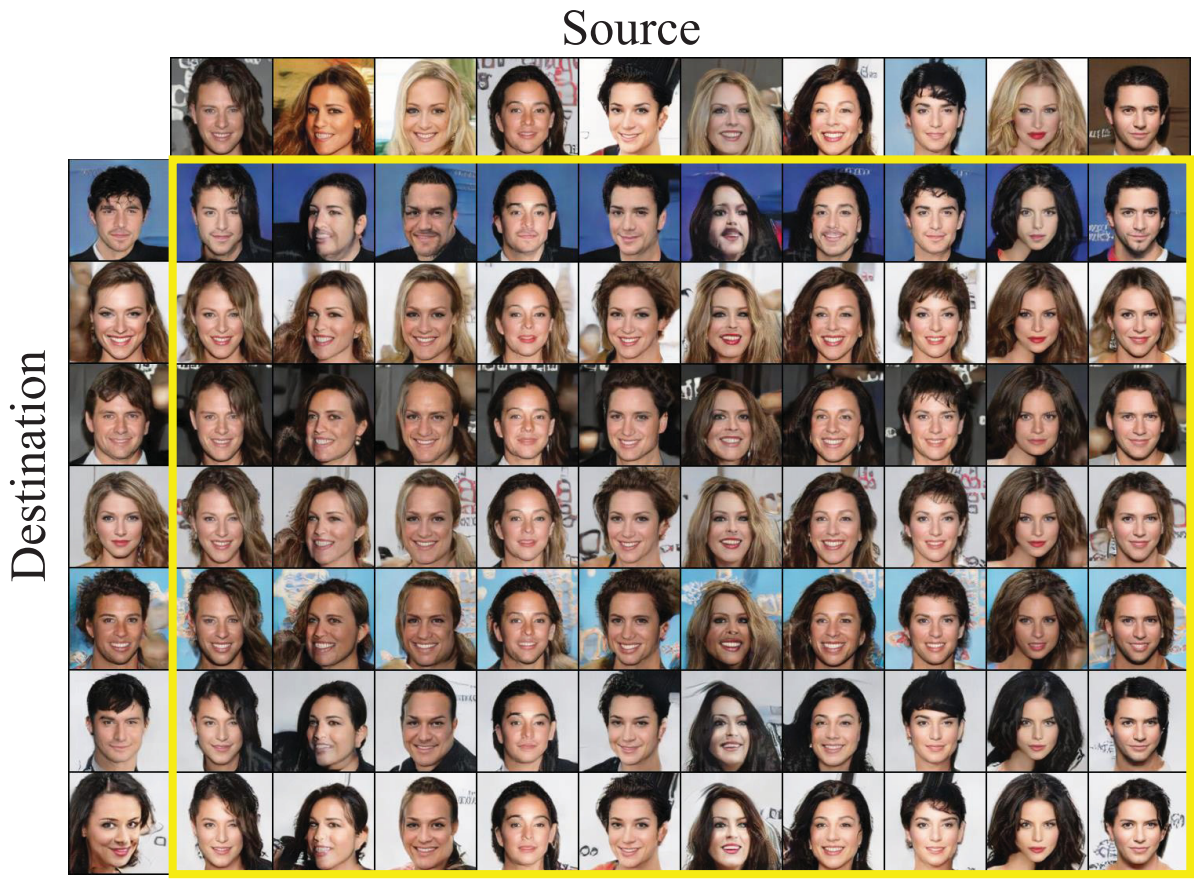

Figure 28. Style mixing on CelebA via the tailored specific part of our method. The "Source" sample controls the identity, posture, and hair type, while the "Destination" sample controls the sex, color, and expression.