Fine-tuning Diffusion Models with Limited Data

Taehong Moon[1] **Moonseok Choi**[1] **Gayoung Lee**[2] **Jung-Woo Ha**[2] **Juho Lee**[1,3]
[1]KAIST AI [2]Naver AI Lab. [3]AITRICS
{taehongmoon, ms.choi, juholee}@kaist.ac.kr
{gayoung.lee, jungwoo.ha}@navercorp.com

Abstract

Diffusion models have recently shown remarkable progress, demonstrating state-of-the-art image generation qualities. Like the other high-fidelity generative models, diffusion models require a large amount of data and computing time for stable training, which hinders the application of diffusion models for limited data settings. To overcome this issue, one can employ a pre-trained diffusion model built on a large-scale dataset and fine-tune it on a target dataset. Unfortunately, as we show empirically, this easily results in overfitting. In this paper, we propose an efficient fine-tuning algorithm for diffusion models that can efficiently and robustly train on limited data settings. We first show that fine-tuning only the small subset of the pre-trained parameters can efficiently learn the target dataset with much less overfitting. Then we further introduce a lightweight adapter module that can be attached to the pre-trained model with minimal overhead and show that fine-tuning with our adapter module significantly improves the image generation quality. We demonstrate the effectiveness of our method on various real-world image datasets.

1 Introduction

Diffusion models [36, 14] have become a mainstream approach in image synthesis, quickly taking over the crown of Generative Adversarial Networks (GANs) [11] in the literature. Compared to Variational Autoencoder (VAE) variants [21, 42, 5] or autoregressive models [43, 9] which are trained to maximize likelihoods instead of directly optimizing image quality, diffusion models can be trained to generate more realistic images with less number of parameters [14, 27]. Similar to the other generative models, diffusion models require tons of data and training times to produce diverse images of high quality. Even a simple diffusion model would require several GPU days to train on CIFAR-10 or ImageNet benchmark. This computational cost acts as a bottleneck, limiting the scalability of diffusion models to applications where data are scarce. To resolve this issue common in high-fidelity generative models, several approaches have been proposed. We offer a detailed review of recent works in Appendix B. However, for diffusion models, except for fine-tuning methods for conditional generation [35, 10], there has been little work on fine-tuning pre-trained unconditional diffusion models for target domain with limited data.

In this paper, we propose a novel fine-tuning algorithm for unconditional diffusion models under limited data settings. This is a challenging task because it is prone to overfitting due to small data and any regularization technique depending on conditional inputs is not available unlike conditional settings [32]. A naı̈ve fine-tuning method of updating all weights easily results in overfitting, so as done in existing transfer learning studies [45, 41, 25], a thorough study on which weights to freeze and tune is required. Through empirical analysis, we first identify that the attention modules common in the diffusion models are crucial for quick adaptation to the target domain, and fine-tuning only the attention modules could significantly alleviate the overfitting. This is somewhat surprising considering that the attention modules take only a small fraction (about 10%) of parameters in the entire network for the diffusion model. Inspired by the adapter approach for language models [16],

NeurIPS 2022 Workshop on Score-Based Methods.

we further design a novel adapter module that can be attached to the pre-trained model and fine-tuned with negligible additional parameters. Unlike the adapter module commonly applied for language models, due to the nature of the diffusion models, our adapter module is time-aware, and we find this time awareness is crucial for successful fine-tuning. Especially, we find that the adapter module plays an important role under a large domain gap between source pre-training and target fine-tuning data. Combining the idea of fine-tuning only the attention modules in the pre-trained model and the fine-tuning with the adapter module, we present **Adapter-Augmented Attention Fine-tuning (A^{3}FT)** which efficiently learns on target tasks with limited data, converging faster with much less overfitting than existing fine-tuning. We empirically demonstrate the effectiveness of our method on real-world image synthesis tasks.

2 Methods

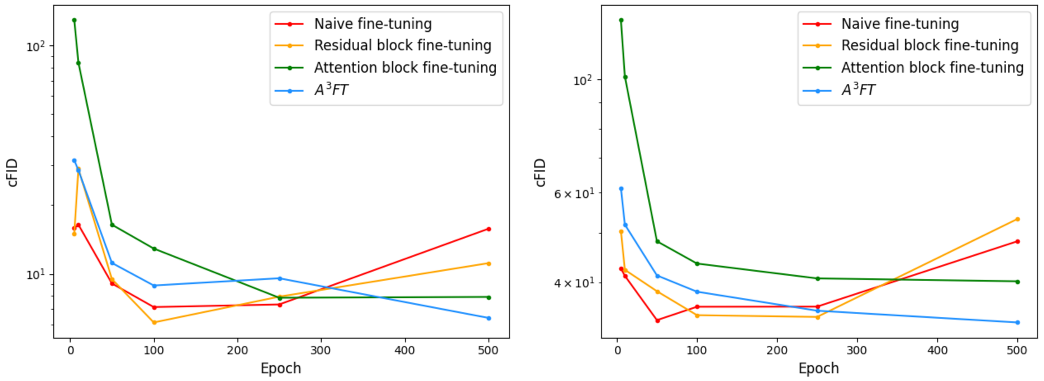

Figure 1: ImageNet pre-trained diffusion models fine-tuned with (left) 1000 CelebA images and (right) 833 Pokemon images. cFID is computed between the whole training dataset and 10K/5K samples from CelebA/Pokemon dataset. While naïve and residual-block fine-tuning exhibit overfitting behavior, our methods, attention-block tuning and A^{3}FT, successfully mitigate overfitting. Further, A^{3}FT achieves the best cFID score.

2.1 Naïve fine-tuning

Naïvely fine-tuning entire parameters of a pre-trained model with limited target data would be a straightforward approach, but it is well-known that this is prone to overfitting [46, 26]. As we empirically validate this for diffusion models in Fig. 1, the naïve fine-tuning causes overfitting in the early stage of training. In Appendix D.1, we further verify that the quality and diversity of the images generated from the overfitted models are seriously damaged, and the models tend to memorize few images in target training data.

2.2 Attention block fine-tuning

Most of the modern diffusion models employ U-Net [31] as a backbone architecture, which consists of a stack of residual and attention blocks intertwined with skip connections. At a high level, a U-Net is divided into an encoder and a decoder, where the encoder gradually downsamples images into multi-scale features, and the decoder upsamples them back to the original sizes. Instead of fine-tuning all the parameters, we fine-tune attention blocks while freezing residual blocks. As shown in Fig. 1, the attention block fine-tuning works surprisingly well, significantly outperforming naïve fine-tuning. Especially, attention block fine-tuning is not only more stabilized, but also achieves better image quality in terms of cFID score [29]. Attention block tuning is more memory-efficient since it only takes 10.3% of the total parameters.

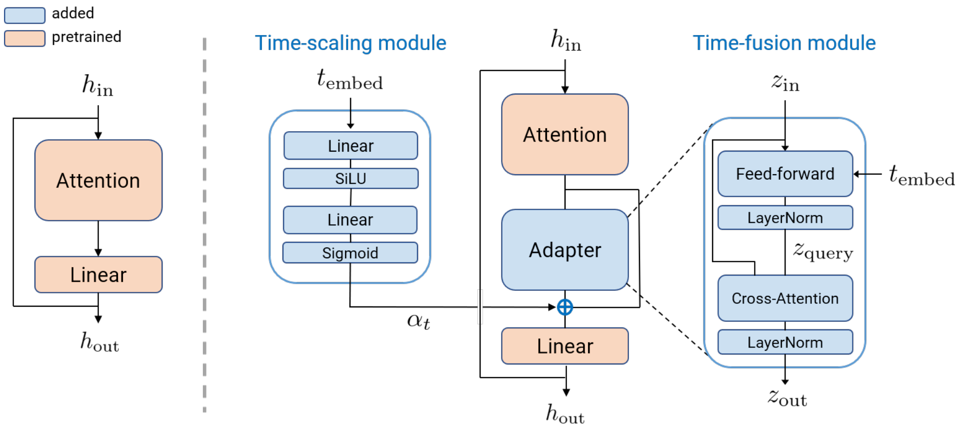

Figure 2: Schematic for the time-aware adapter. Time-aware adapter is designed to fit inside an attention block. We introduce two adapter submodules: a time-fusion module and a time-scaling module. Time-aware adapters take up to 2.2M parameters ($\approx$ 1.8% of the pre-trained diffusion model).

2.3 Time-aware adapter

While the attention block fine-tuning works well, as demonstrated by the adapter-based methods for language models [16, 17], it is often beneficial to attach additional modules to the pre-trained model for better adaptation to target data. Inspired by this, we develop a novel adapter module tailored for diffusion models. Importantly, our adapter is *time-aware*, since the networks being used for diffusion models should produce different outputs with respect to different time steps. As depicted in Fig. 2, the time-aware adapter is placed between the attention block and its successive linear layer so as to modulate the intermediate feature vector given the target data. Compared to the conventional adapters, our adapter additionally takes time embedding vector $t_{\text{embed}} \in \mathbb{R}^{d_t}$ as an input along with feature tensor $z_{\text{in}} \in \mathbb{R}^{c_z \times h_z \times w_z}$, hence the name time-aware. A time-aware adapter is composed of three submodules; a time-fusion module, a cross-attention module, and a time-scaling module.

Time-fusion module Time-fusion network combines the time-embedding vector t_{embed} and an input feature vector z_{in} into another feature vector through feed-forward layers, which is then layer-normalized [1]. Refer to Appendix C.4 for details. The resulting feature vector $z_{\text{query}} \in \mathbb{R}^{c_z \times h_z \times w_z}$ is fed into the cross-attention module as a query. That is,

$$z_{\text{query}} = \text{LN}(\text{FF}(z_{\text{in}}, t_{\text{embed}})), \tag{1}$$

where $\text{LN}(\cdot)$ denotes layer normalization and $\text{FF}(\cdot)$ denotes feed-forward layers. Then, cross-attention module takes z_{query} as a query, and z_{in} as both key and value vectors with the same dimension. We then apply channel-wise cross attention followed by layer normalization to produce an output tensor $z_{\text{out}} \in \mathbb{R}^{c_z \times h_z \times w_z}$.

$$z_{\text{out}} = \text{LN}(\text{CrossAttn}(z_{\text{query}}, z_{\text{in}}, z_{\text{in}})). \tag{2}$$

Time-fusion modules use approximately 1.7M parameters in total ($\approx$ 1.4% of the pre-trained model).

Time-scaling module Motivated by Hong et al. [15], the time-scaling module computes the channel-wise mixture factor $\alpha_t \in [0, 1]^{c_z}$ that regulates the information generated by the time-aware adapter. Here, the mixture factor α_t is also time-aware so that the effect of the adapter output can vary along time steps. The final output of the time-aware adapter is computed as

$$h_{\text{out}} = \text{FF}((1 - \alpha_t) \odot z_{\text{in}} + \alpha_t \odot z_{\text{out}}), \tag{3}$$

where $\odot$ denotes the elementwise multiplication. Time-scaling modules use approximately 0.5M parameters ($\approx$ 0.4% of the pre-trained model).

2.4 Adapter-augmented attention fine-tuning

Combining the attention block fine-tuning and time-aware adapter, we present Adapter-Augmented Attention Fine-tuning (A^{3}FT) which fine-tunes attention blocks along with the time-aware adapter attached to the pre-trained model. Compared with the conventional adapter-based methods [16], we adapt the weight-sharing for the time-aware adapters. Based on the weight-sharing of our modules, we observe that it not only alleviates the overfitting in our case but also accelerates the convergence. We will demonstrate through experiments in Section 3, the time-aware adapter can seamlessly be trained with the attention blocks, leading to a significant performance gain across datasets.

3 Experiments

Our method is tested on both low and high resolution datasets: 64×64 resolution (CelebA [24], Pokemon [34]) and 256×256 resolution (MetFaces [18]). See Appendix C for further experimental details. Table 1 summarizes our main results. We validate that naïve fine-tuning easily overfits to small-size training datasets yielding high cFID and KID. On the other hand, attention-block fine-tuning largely outperforms naïve fine-tuning on both datasets. Further, with the help of time-aware adapter, A^{3}FT achieves the lowest cFID and KID score. In Fig. 3, we notice the distinct difference in sample quality. While naïvely fine-tuned model produces relatively plain and similar images, A^{3}FT yield both diverse and high-quality images. We provide the sampled images on all datasets in Appendix D.2.

Table 1: Sample quality evaluation on CelebA, Pokemon and MetFaces datasets. We report metrics computed after fine-tuning for 500 epochs on CelebA and Pokemon datasets. With regard to MetFace datasets, we report values computed after fine-tuning for 2000 epochs. A^{3}FT surpasses naïve and attention-block fine-tuning in terms of cFID and KID.

Methods	ImageNet → CelebA		ImageNet → Pokemon		FFHQ → MetFaces	
	cFID	KID	cFID	KID	cFID	KID
Naïve fine-tuning	15.748	0.010	57.505	0.025	54.291	0.011
Attention-block fine-tuning	7.923	0.005	39.835	0.013	32.782	**0.008**
A^{3}FT	**6.412**	**0.004**	**38.398**	**0.012**	**30.985**	0.009

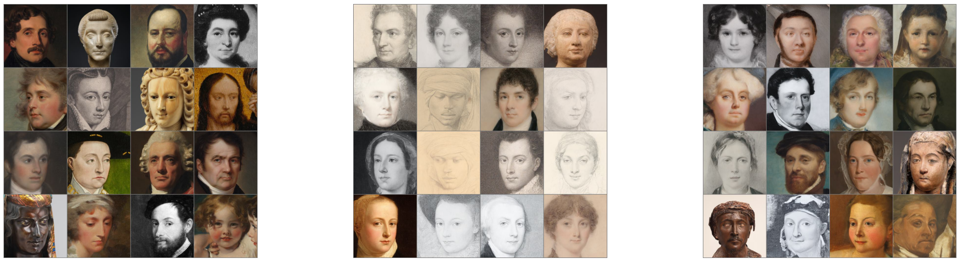

Figure 3: Generated images on MetFaces dataset: (left) training set; (middle) naïve FT; (right) A^{3}FT.

4 Conclusion

In this paper, we empirically showed that naïvely fine-tuning diffusion models incurs overfitting in the early stage of training. As a solution, we discovered that fine-tuning only attention blocks can be a simple yet solid fine-tuning technique. To enhance the fine-tuning performance, we introduced the time-aware adapter that integrates a time-embedding vector so as to fit unconditional diffusion models. We further proposed a novel fine-tuning method, Adapter-Augmented Attention Fine-tuning, which fine-tunes attention blocks in conjunction with the time-aware adapter. Experiments validated that our approach enables faster and more stable fine-tuning with limited data as well as retaining high-quality samples, especially, for a large domain gap between the pre-training and the fine-tuning datasets.

References

[1] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 3

[2] M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton. Demystifying MMD GANs. *arXiv preprint arXiv:1801.01401*, 2018. 10

[3] A. Brock, J. Donahue, and K. Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations (ICLR)*, 2019. 9

[4] S. Chen, C. Ge, Z. Tong, J. Wang, Y. Song, J. Wang, and P. Luo. Adaptformer: adapting vision transformers for scalable visual recognition. *arXiv preprint arXiv:2205.13535*, 2022. 9

[5] R. Child. Very deep VAEs generalize autoregressive models and can outperform them on images. In *International Conference on Learning Representations (ICLR)*, 2021. 1, 9

[6] J. Choi, J. Lee, C. Shin, S. Kim, H. Kim, and S. Yoon. Perception prioritized training of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11472–11481, 2022. 10

[7] P. Dhariwal and A. Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*, 2021. 8, 9

[8] S. Elfwing, E. Uchibe, and K. Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018. 10

[9] P. Esser, R. Rombach, and B. Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 9

[10] G. Giannone, D. Nielsen, and O. Winther. Few-shot diffusion models. *arXiv preprint arXiv:2205.15463*, 2022. 1, 9

[11] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, 2014. 1, 9

[12] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 10

[13] J. Ho and T. Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 8

[14] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020. 1, 8, 9

[15] W. Hong, M. Ding, W. Zheng, X. Liu, and J. Tang. CogVideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022. 3

[16] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of The 36th International Conference on Machine Learning (ICML 2019)*, 2019. 1, 3, 4, 9, 10

[17] R. Karimi Mahabadi, J. Henderson, and S. Ruder. Compacter: Efficient low-rank hypercomplex adapter layers. In *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*, 2021. 3, 9, 10

[18] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila. Training generative adversarial networks with limited data. *Advances in Neural Information Processing Systems*, 33:12104–12114, 2020. 4, 9

[19] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and improving the image quality of styleGAN. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 9

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 10

[21] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014. 1, 8, 9

[22] N. Kumari, R. Zhang, E. Shechtman, and J.-Y. Zhu. Ensembling off-the-shelf models for GAN training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10651–10662, 2022. 9

[23] B. Liu, Y. Zhu, K. Song, and A. Elgammal. Towards faster and stabilized GAN training for high-fidelity few-shot image synthesis. In *International Conference on Learning Representations (ICLR)*, 2020. 9

[24] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015. 4

[25] S. Mo, M. Cho, and J. Shin. Freeze the discriminator: a simple baseline for fine-tuning GANs. In *CVPR AI for Content Creation Workshop*, 2020. 1, 9

[26] J. Ngiam, D. Peng, V. Vasudevan, S. Kornblith, Q. V. Le, and R. Pang. Domain adaptive transfer learning with specialist models. *arXiv preprint arXiv:1811.07056*, 2018. 2

[27] A. Q. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *Proceedings of The 38th International Conference on Machine Learning (ICML 2021)*, 2021. 1, 8, 10

[28] U. Ojha, Y. Li, J. Lu, A. A. Efros, Y. J. Lee, E. Shechtman, and R. Zhang. Few-shot image generation via cross-domain correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 9

[29] G. Parmar, R. Zhang, and J.-Y. Zhu. On aliased resizing and surprising subtleties in GAN evaluation. In *CVPR*, 2022. 2, 10

[30] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 9

[31] O. Ronneberger, P. Fischer, and T. Brox. U-Net: convolutional networks for biomedical image segmentation. In *Proceedings of the 18th Medical Image Computing and Computer-Assisted Intervention (MICCAI 2015)*, 2015. 2

[32] N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, and K. Aberman. DreamBooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arXiv preprint arxiv:2208.12242*, 2022. 1, 9

[33] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022. 9

[34] A. Sauer, K. Chitta, J. Müller, and A. Geiger. Projected GANs converge faster. In *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*, 2021. 4, 9

[35] A. Sinha, J. Song, C. Meng, and S. Ermon. D2C: Diffusion-decoding models for few-shot conditional generation. In *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*, 2021. 1

[36] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015. 1, 8

[37] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2020. 9

[38] Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019. 8

[39] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021. 8

[40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56): 1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html. 10

[41] Y.-L. Sung, J. Cho, and M. Bansal. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5227–5237, 2022. 1, 9

[42] A. Vahdat and J. Kautz. NVAE: A deep hierarchical variational autoencoder. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020. 1, 9

[43] A. Van Den Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. In *Proceedings of The 33rd International Conference on Machine Learning (ICML 2016)*, 2016. 1, 9

[44] T. Wang, T. Zhang, B. Zhang, H. Ouyang, D. Chen, Q. Chen, and F. Wen. Pretraining is all you need for image-to-image translation. *arXiv preprint arXiv:2205.12952*, 2022. 9

[45] L. Xuhong, Y. Grandvalet, and F. Davoine. Explicit inductive bias for transfer learning with convolutional networks. In *Proceedings of The 35th International Conference on Machine Learning (ICML 2018)*, 2018. 1

[46] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014. 2

[47] M. Zhao, Y. Cong, and L. Carin. On leveraging pretrained GANs for generation with limited data. In *Proceedings of The 37th International Conference on Machine Learning (ICML 2020)*, 2020. 9

[48] S. Zhao, Z. Liu, J. Lin, J.-Y. Zhu, and S. Han. Differentiable augmentation for data-efficient GAN training. *Advances in Neural Information Processing Systems*, 33:7559–7570, 2020. 9

A Background

We provide a brief summary of diffusion models. We follow the standard formulation and notations [36, 38, 14].

A diffusion model is a class of latent variable models with a Markov chain of latent variables describing the generative process. Given a data $\mathbf{x}_0 \sim p_{\text{data}}(\mathbf{x}) := q(\mathbf{x})$, a forward process injects a series of Markovian Gaussian noises with a sequence of controlled variances $\{\beta_t\}_{t=1}^{T}$ over T steps, until the perturbed data $\mathbf{x}_T$ is nearly isotropic Gaussian distributed:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \text{ for } t = 1, \dots, T-1, \quad q(\mathbf{x}_T|\mathbf{x}_0) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}). \tag{4}$$

Reversing the forward process, a backward diffusion process will bring an isotropic Gaussian noise $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ to a sample from the data distribution $\mathbf{x}_0 \sim q(\mathbf{x}_0)$. Since the backward process $q(\mathbf{x}_t|\mathbf{x}_{t-1})$ is intractable, we introduce an approximate backward process defined with the model $p_\theta(\mathbf{x}_t|\mathbf{x}_{t-1})$ parameterized by θ:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)) \text{ for } t = 1, \dots, T. \tag{5}$$

Then we can optimize the parameter θ by maximizing the variational lower-bound, or minimizing the negative of the variational lower bound.

$$\mathcal{L}_{\text{vlb}} := \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] \geq \mathbb{E}_{q(\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0)\right]. \tag{6}$$

By Bayes' rule and Markov property, one can easily derive that

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t\mathbf{I}) \tag{7}$$

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t, \ \tilde{\beta}_t := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t, \ \bar{\alpha}_t := \prod_{i=1}^{t}(1-\beta_i), \tag{8}$$

and the variational objective (6) boils down to

$$\mathcal{L}_{\text{vlb}} \equiv \mathbb{E}_{t,\mathbf{x}_0}\left[\frac{1}{2||\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)||^2}||\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)||^2\right], \tag{9}$$

where $\equiv$ denotes the equality up to a constant. Employing reparameterization for Gaussians [21], Ho et al. [14] proposes to further simplify the objective as

$$\mathcal{L}_{\text{simple}} \equiv \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}_t}\left[||\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)||^2\right], \tag{10}$$

where we parameterize the noise $\boldsymbol{\epsilon}_\theta$ instead of the mean $\boldsymbol{\mu}_\theta$ and the scaling factors $\|\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\|^2$ are discarded to have uniform loss weights for all time steps.[1]

While Ho et al. [14] demonstrated that using fixed variance schedules for the model distributions empirically working well, that is, setting $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t) = \sigma_t^2\mathbf{I}$, Nichol and Dhariwal [27] later showed that one can improve upon fixed variance with the variances parameterized by weighting vector v, $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t) = \exp(v \log \beta_t + (1-v)\log \tilde{\beta}_t)$. Since the simplified objective (10) dropping the scaling factors does not contain the variance terms, Nichol and Dhariwal [27] proposed to optimize a hybrid objective defined as

$$\mathcal{L}_{\text{hybrid}} = \mathcal{L}_{\text{simple}} + \lambda\mathcal{L}_{\text{vlb}}, \tag{11}$$

for some $\lambda > 0$, together with cosine annealing variance scheduling which is specifically tailored for the low-resolution images (e.g., 64×64 or 32×32 images). We follow the training scheme and model architecture described in Nichol and Dhariwal [27] throughout our experiments.

[1] Of note, not only straight-forward, the simple loss resembles the score matching loss in noise-conditioned score networks [38]. Forward and reverse processes each can be mapped to noise conditional score matching and Langevin dynamics. Such connection laid the groundwork for score-based continuous diffusion models [7, 39, 13]. The uniform weighting is reported to enhance image quality while sacrificing the log-likelihood values of generated images [14].

B Related works

Generative models for image synthesis Generative models for image synthesis are largely categorized into likelihood-based approaches [21, 42, 5, 43, 9] and GANs [11, 19, 3]. Diffusion models have recently gained popularity due to their ability to generate high-quality images [37, 14, 7]. Most modern generative models use deep neural networks requiring a large amount of training data, so they usually suffer from overfitting when trained with limited data.

Image synthesis with limited data Multiple works have been proposed to improve the performance of image synthesis when the number of data is limited. For GAN-based methods, Karras et al. [18], Zhao et al. [48] propose data augmentation techniques to prevent discriminator overfitting. Liu et al. [23] propose a lightweight network architecture suitable for small datasets. Meanwhile, leveraging pre-trained models on larger data and fine-tuning them on smaller data sets is becoming as popular as in other fields. Mo et al. [25] show that freezing discriminators in GAN fine-tuning gives stable results. Ojha et al. [28] constrain the adapted model to share latent space with the source model. Zhao et al. [47] shows that the low-level filters of pre-trained GANs are helpful in learning limited training data even when they are perceptually distant from the target data. Sauer et al. [34], Kumari et al. [22] used pre-trained models as feature extractors for a discriminator and show the effectiveness of faster and more stable convergence. However, fine-tuning with limited data for unconditional diffusion models is still less explored yet.

Fine-tuning for diffusion models Recently, a high-performance pre-trained diffusion model has been published [30, 33]. As of now, few pioneering works investigate fine-tuning diffusion models; Giannone et al. [10] proposing few-shot learning diffusion models that can quickly learn conditional generative model for a set of images, and Ruiz et al. [32] proposing a prompt-like approach for text-to-image diffusion models. Wang et al. [44] propose two-stage finetuning scheme for image-to-image translation. However, these works focus on conditional generation and fine-tune the pre-trained model without network modification. To the best of our knowledge, our method is the first one to study module-level effects for fine-tuning an unconditional diffusion model with limited data.

Adapter-based method for transfer learning In a natural language processing task, Houlsby et al. [16] proposed a memory-efficient method for transformers that only updates adapter modules while freezing other weights during fine-tuning. The adapter-based method requires only a small number of additional parameters, so it is efficient when adapting to a variety of tasks. Karimi Mahabadi et al. [17] developed a more compact and faster adapter module based on matrix factorization and weight sharing. Chen et al. [4], Sung et al. [41] adopt the adapter method for the vision task and showed effectiveness on reducing memory and training time.

C Experimental Details

C.1 Configurations

Our algorithm is implemented in Python 3.9.5 and Pytorch 1.10.2. All experiments are performed on four NVIDIA Tesla V100s.

C.2 Setting

We present detailed experimental settings for experiment results reproduction. For the low-resolution datasets, we use the ImageNet 64×64 pre-trained model ($\approx$ 120M parameters) in Nichol and Dhariwal [27]. Our model consists of a total of 15 attention blocks and 30 residual blocks. Input images go under four downsampling steps, each with $[2C, 2C, 2C, 4C]$ channels where $C = 128$, and then upsampled via four symmetric steps. The pre-trained model employs multi-head attention blocks with 4 heads. For the high-resolution datasets, we use the FFHQ 256×256 pre-trained model ($\approx$ 90M parameters) presented in [6], which is composed of a total of 4 attention blocks and 31 residual blocks.

We employ Adam [20] for all experiments, with a learning rate of $2 \cdot 10^{-4}$ and epsilon of $1 \cdot 10^{-8}$. We set the batch size to 128 and 5 for fine-tuning and sampling, respectively. Opposed to Nichol and Dhariwal [27], we do not use neither dropout [40] nor exponential moving average over model parameters.

To fine-tune diffusion models, we use the hybrid loss in (11) with cosine beta scheduling such that $(\beta_1, \beta_2) = (1 \cdot 10^{-4}, 2 \cdot 10^{-2})$. We set diffusion timesteps as $T = 1000$.

We fine-tune the pre-trained model with 1000 CelebA images, 833 Pokemon images and 1336 MetFaces images. Evaluation metrics are computed between the whole training dataset and 10K/5K/1.4K samples from CelebA/Pokemon/MetFaces dataset.

C.3 Metrics

Since quantifying image generation performance remains a challenge, metrics must be carefully chosen. We use two different evaluation metrics: (i) clean Fréchet Inception Distance (cFID) [29] and (ii) Kernel Inception Distance (KID) [2]. While FID [12] is a popular choice for capturing both image fidelity and diversity, Parmar et al. [29] points out that FID is inconsistent with image resizing and quantization. To resolve such an issue, we use cFID by default. Due to our relatively small dataset size, we also implement KID which is proven to be unbiased and asymptotically normal. Metrics are computed between the whole training dataset and generated images.

C.4 Time-aware adapter

In this section, we review a detailed structure of the time-aware adapter implemented in low-resolution experiment. In conventional adapter-based methods [16, 17], independent adapters are employed across all blocks. In Houlsby et al. [16] and Karimi Mahabadi et al. [17], every single transformer layer needs two separate adapters so that the number of adapters needed grows as to the number of layers. We alleviate such an issue using a weight-sharing technique. We let the same adapter submodule fit into several attention blocks.

We build two time-fusion modules which integrate into attention blocks with 384 and 512 channels. Time-fusion module first match the dimension of two inputs, t_{embed} and z_{in}. The time-embedding vector t_{embed} goes through two linear layers to match the dimension of z_{in}, and we add such two vectors before passing on to the final linear layer. Similarly, we build only two cross-attention modules which are attached to attention blocks with 384 and 512 channels. We implement vanilla single-head attention operation. Importantly, we construct separate time-scaling modules for all attention blocks analogous to conventional NLP adapters. This is because we want to regulate the information from time-aware adapter in a block-wise manner. Time-scaling module is merely an MLP with two fully connected layers with SiLU activation [8].

D Additional results

D.1 Sample diversity

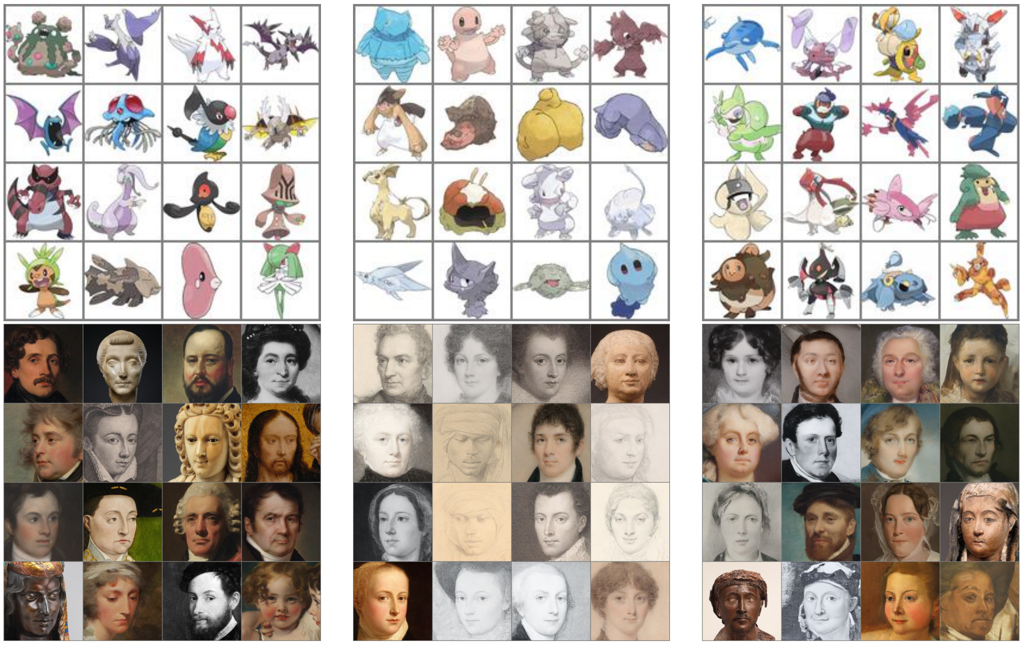

Figure 4: Image samples on Pokemon and MetFaces datasets: (left) training set; (middle) naïve fine-tuning; (right) A^{3}FT. While naïvely fine-tuned model produces low-quality samples close to the training set, A^{3}FT yields both diverse and high-quality images.

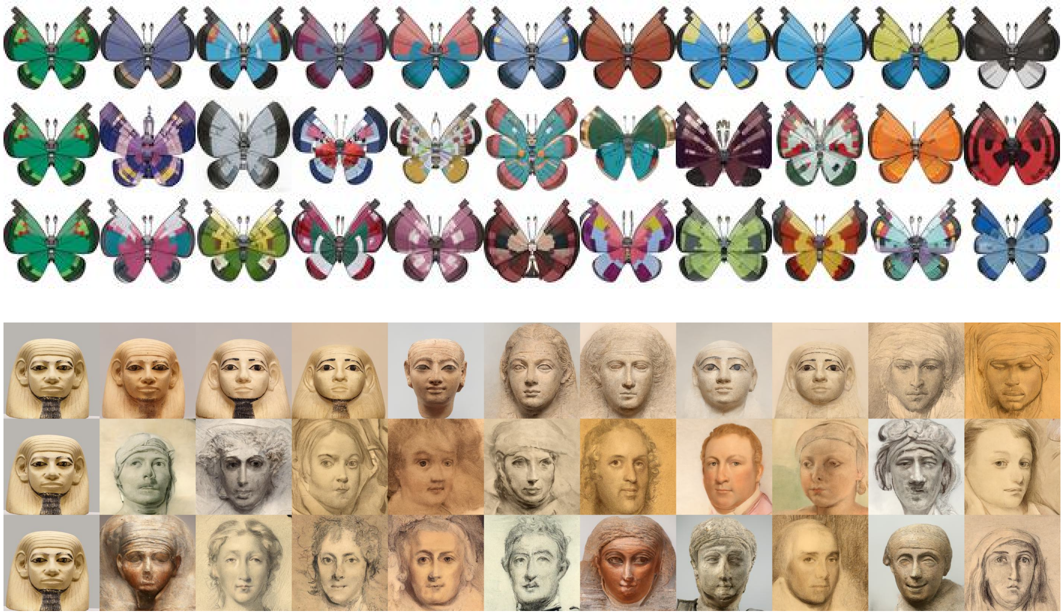

Figure 5: Pixel-space nearest neighbors on Pokemon and MetFaces dataset. The training data is in the leftmost column, and the corresponding 10 closest generated images are presented in the remaining columns. (top) naïve fine-tuning; (middle) attention-block fine-tuning; (bottom) A^{3}FT. Compared to the naïve fine-tuning method, our methods produce a broader set of images.

D.2 More images

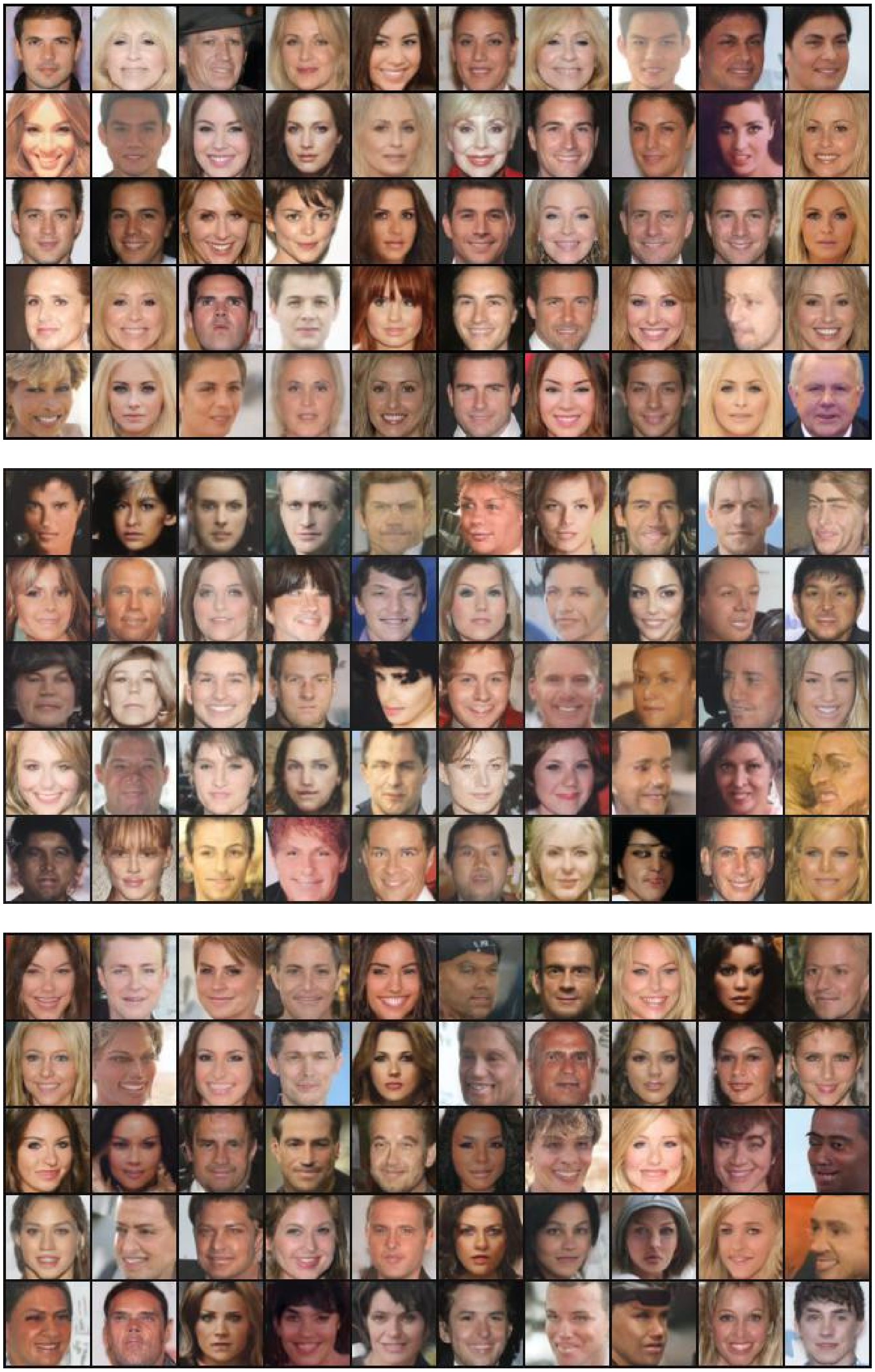

Figure 6: Generated images after fine-tuning for 1500 epochs on CelebA dataset: (top) naïve fine-tuning, (middle) attention-block fine-tuning, (bottom) A^{3}FT.

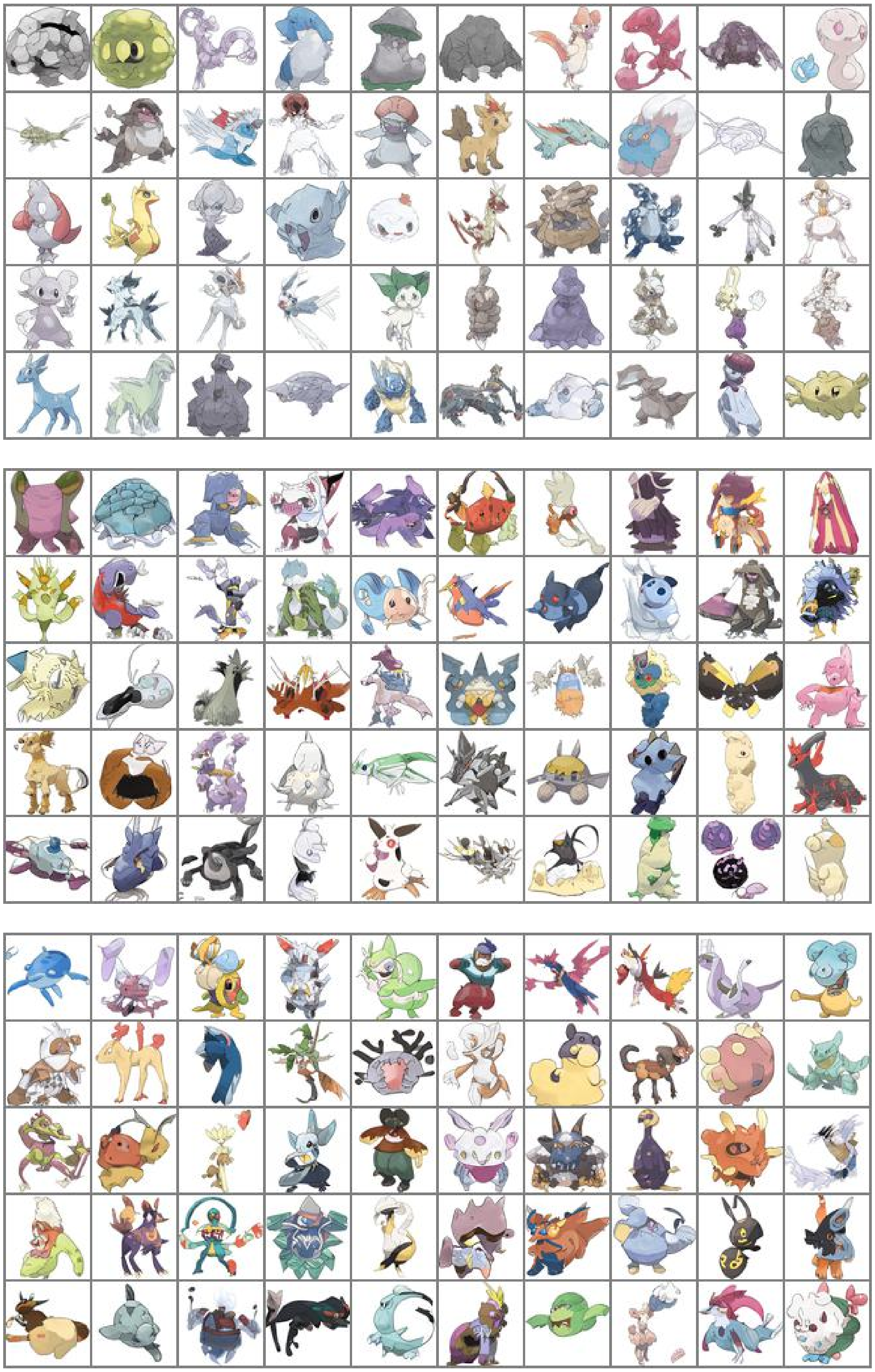

Figure 7: Generated images after fine-tuning for 500 epochs on Pokemon dataset: (top) naïve fine-tuning, (middle) attention-block fine-tuning, (bottom) A^{3}FT.

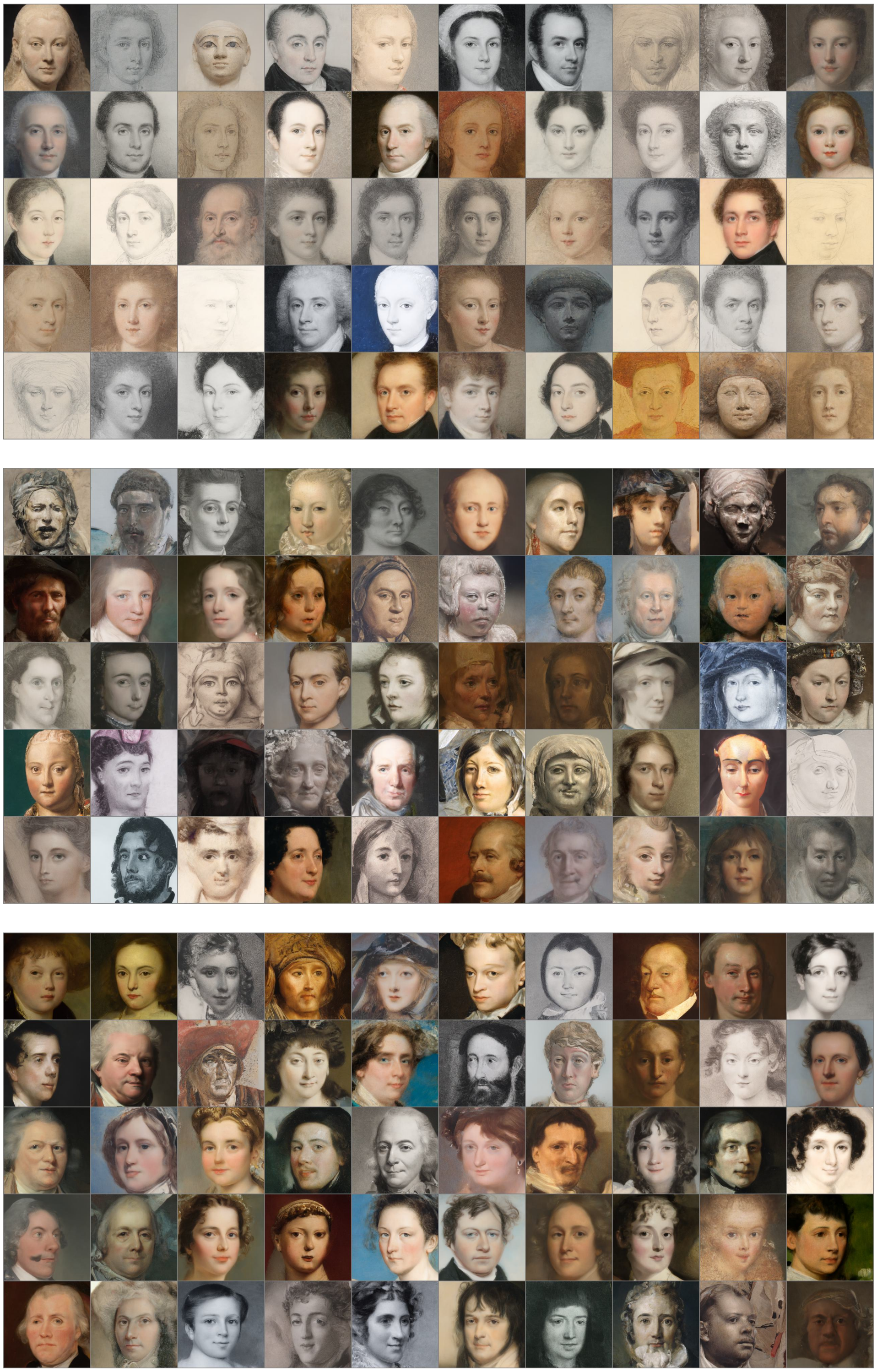

Figure 8: Generated images after fine-tuning for 2000 epochs on MetFaces dataset: (top) naïve fine-tuning, (middle) attention-block fine-tuning, (bottom) A^{3}FT.