Efficient Diffusion Training via Min-SNR Weighting Strategy
Supplementary Material

In the supplementary material, we first provide the proof of Theorem 1 in Section 1. Then we derive the relationship between loss weights of different predicting targets in Section 2. In Section 3, we provide more details on the network architecture, training and sampling settings. Finally, we present more visual results in Section 4.

1. Proof for Theorem 1

First, we introduce the Pareto Optimality mentioned in the paper. Assume the loss for each task is $\mathcal{L}^t(\theta), t \in \{1, 2, \dots, T\}$ and the respective gradient to θ is $\nabla_\theta \mathcal{L}^t(\theta)$. For simplicity, we denote $\mathcal{L}^t(\theta)$ as $\mathcal{L}^t$. If we treat each task with equal importance, we assume each loss item $\mathcal{L}^1, \mathcal{L}^2, \dots, \mathcal{L}^T$ is decreasing or kept the same. There exists one point θ^* where any change of the point will leads to the increase of one loss item. We call the point θ^* "Pareto Optimality". In other words, we cannot sacrifice one task for another task's improvement. To reach Pareto Optimality, we need to find an update direction δ which meet:

$$\left\{ \begin{array}{ll} \left\langle \nabla_\theta \mathcal{L}_\theta^1, \delta \right\rangle & \leq 0 \\ \left\langle \nabla_\theta \mathcal{L}_\theta^2, \delta \right\rangle & \leq 0 \\ \quad \vdots & \\ \left\langle \nabla_\theta \mathcal{L}_\theta^T, \delta \right\rangle & \leq 0 \end{array} \right. \tag{1}$$

$\langle \cdot, \cdot \rangle$ denotes the inner product of two vectors. It is worth noting that $\delta = 0$ satisfies all the above inequalities. We care more about the non-zero solution and adopt it for updating the network parameter θ. If the non-zero point does not exist, it may already achieve the "Pareto Optimality", which is referred as "Pareto Stationary".

For simplicity, we denote the gradient for each loss item $\nabla_\theta \mathcal{L}^t$ as $\mathbf{g}_t$. Suppose we have a gradient vector $\mathbf{u}$ to satisfy that all $\langle \mathbf{g}_t, \mathbf{u} \rangle \geq 0, t \in \{1, 2, \dots, T\}$. Then $-\mathbf{u}$ is the updating direction ensuring a lower loss for each task.

As proposed in [11], $\langle \mathbf{g}_t, \mathbf{u} \rangle \geq 0, \forall t \in \{1, 2, \dots, T\}$ is equivalent to $\min_t \langle \mathbf{g}_t, \mathbf{u} \rangle \geq 0$. And it could be achieved when the minimal value of $\langle \mathbf{g}_t, \mathbf{u} \rangle$ is maximized. Thus the problem is further converted to:

$$\max_{\mathbf{u}} \min_t \langle \mathbf{g}_t, \mathbf{u} \rangle$$

There is no constraint for the vector $\mathbf{u}$, so it may become infinity and make the updating unstable. To avoid it, we add a regularization term to it

$$\max_{\mathbf{u}} \min_t \langle \mathbf{g}_t, \mathbf{u} \rangle - \frac{1}{2} \|\mathbf{u}\|_2^2. \tag{2}$$

And notice that the max function ensures the value is always greater than or equal to a specific value $\mathbf{u} = 0$.

$$\begin{aligned} & \max_{\mathbf{u}} \min_t \langle \mathbf{g}_t, \mathbf{u} \rangle - \frac{1}{2} \|\mathbf{u}\|_2^2 \\ & \geq \min_t \langle \mathbf{g}_t, \mathbf{u} \rangle - \frac{1}{2} \|\mathbf{u}\|_2^2 \Big|_{\mathbf{u}=0} \\ & = 0, \end{aligned}$$

which also means $\max_{\mathbf{u}} \min_t \langle \mathbf{g}_t, \mathbf{u} \rangle \geq \frac{1}{2} \|\mathbf{u}\|_2^2 \geq 0$. Therefore, the solution of Equation 2 satisfies our optimization goal of $\langle \mathbf{g}_t, \mathbf{u} \rangle \geq 0, \forall t \in \{1, 2, \dots, T\}$.

We define $\mathcal{C}^T$ as a set of n-dimensional variables

$$\mathcal{C}^T = \left\{ (w_1, w_2, \dots, w_T) | w_1, w_2, \dots, w_T \geq 0, \sum_{t=1}^{T} w_t = 1 \right\}, \tag{3}$$

It is easy to verify that

$$\min_t \langle \mathbf{g}_t, \mathbf{u} \rangle = \min_{w \in \mathcal{C}^T} \left\langle \sum_t w_t \mathbf{g}_t, \mathbf{u} \right\rangle. \tag{4}$$

We can also verify the above function is concave with respect to $\mathbf{u}$ and α. According to Von Neumann's Minmax theorem [12], the objective with regularization in Equation 2 is equivalent to

$$\max_{\mathbf{u}} \min_{w \in \mathcal{C}^T} \left\{ \left\langle \sum_t w_t \mathbf{g}_t, \mathbf{u} \right\rangle - \frac{1}{2} \|\mathbf{u}\|_2^2 \right\} \tag{5}$$

$$= \min_{w \in \mathcal{C}^T} \max_{\mathbf{u}} \left\{ \left\langle \sum_t w_t \mathbf{g}_t, \mathbf{u} \right\rangle - \frac{1}{2} \|\mathbf{u}\|_2^2 \right\} \tag{6}$$

$$= \min_{w \in \mathcal{C}^T} \left\{ \left\langle \sum_t w_t \mathbf{g}_t, \mathbf{u} \right\rangle - \frac{1}{2} \|\mathbf{u}\|_2^2 \right\} \Bigg|_{\mathbf{u}=\frac{1}{2}\sum_t w_t \mathbf{g}_t} \tag{7}$$

$$= \min_{w \in \mathcal{C}^T} \frac{1}{2} \left\| \sum_t w_t \mathbf{g}_t \right\|_2^2. \tag{8}$$

Finally, we achieved Theorem 1 in the main paper.

2. Relationship between Different Targets

The most common predicting target is in ϵ-space. Loss for prediction in $\mathbf{x}_0$-space and ϵ-space can be transformed by the SNR loss weight.

$$\begin{aligned}\mathcal{L}_\theta &= \|\epsilon - \hat{\epsilon}_\theta(\mathbf{x}_t)\|_2^2 \\ &= \left\| \frac{1}{\sigma_t}(\mathbf{x}_t - \alpha_t \mathbf{x}_0) - \frac{1}{\sigma_t}(\mathbf{x}_t - \alpha_t \hat{\mathbf{x}}_\theta(\mathbf{x}_t)) \right\|_2^2 \\ &= \frac{\alpha_t^2}{\sigma_t^2} \|\mathbf{x}_0 - \hat{\mathbf{x}}_\theta(\mathbf{x}_t))\|_2^2 \\ &= \text{SNR}(t) \|\mathbf{x}_0 - \hat{\mathbf{x}}_\theta(\mathbf{x}_t))\|_2^2 \,,\end{aligned}$$

where $\hat{\epsilon}_\theta$ is the network to predict the noise and $\hat{\mathbf{x}}_\theta$ is to predict the clean data.

Prediction target $\mathbf{v} = \alpha_t \epsilon - \sigma_t \mathbf{x}_0$ is proposed in [9], we can derive the related loss

$$\begin{aligned}\mathcal{L}_\theta &= \|\mathbf{v}_t - \mathbf{v}_\theta(\mathbf{x}_t)\|_2^2 \\ &= \|(\alpha_t \epsilon - \sigma_t \mathbf{x}_0) - (\alpha_t \hat{\epsilon}_\theta(\mathbf{x}_t) - \sigma_t \hat{\mathbf{x}}_\theta(\mathbf{x}_t))\|_2^2 \\ &= \|\alpha_t (\epsilon - \hat{\epsilon}_\theta(\mathbf{x}_t)) - \sigma_t (\mathbf{x}_0 - \hat{\mathbf{x}}_\theta(\mathbf{x}_t))\|_2^2 \\ &= \left\| \alpha_t \frac{\alpha_t}{\sigma_t} (\hat{\mathbf{x}}_\theta(\mathbf{x}_t) - \mathbf{x}_0) - \sigma_t (\mathbf{x}_0 - \hat{\mathbf{x}}_\theta(\mathbf{x}_t)) \right\|_2^2 \\ &= \left\| \frac{\alpha_t^2 + \sigma_t^2}{\sigma_t} (\mathbf{x}_0 - \hat{\mathbf{x}}_\theta(\mathbf{x}_t)) \right\|_2^2 \\ &= \frac{1}{\sigma_t^2} \|(\mathbf{x}_0 - \hat{\mathbf{x}}_\theta(\mathbf{x}_t))\|_2^2 \\ &= \frac{\alpha_t^2 + \sigma_t^2}{\sigma_t^2} \|(\mathbf{x}_0 - \hat{\mathbf{x}}_\theta(\mathbf{x}_t))\|_2^2 \\ &= (\text{SNR}(t) + 1) \|(\mathbf{x}_0 - \hat{\mathbf{x}}_\theta(\mathbf{x}_t))\|_2^2\end{aligned}$$

3. Hyper-parameter

Here we list more details about the architecture, training and evaluation setting.

3.1. Architecture Settings

The ViT setting adopted in the paper are as follows,

Model	Layers	Hidden Size	Heads	Params
ViT-Small	13	512	6	43M
ViT-Base	12	768	12	88M
ViT-Large	21	1024	16	269M
ViT-XL	28	1152	16	451M

Table 1: Configurations of our used ViTs.

We use ViT-Small for face generation on CelebA 64×64. Besides, we adopt ViT-Base as the default backbone for the ablation study. To make relative fair comparison with U-ViT, we use a 21-layer ViT-Large for ImageNet 64×64 benchmark. To compare with former state-of-the-art method DiT [7] on ImageNet 256×256, we adopt the similar setting ViT-XL with the same depth, hidden size, and patch size.

In the paper, we also evaluate our method's robustness to model architectures using the UNet backbone. For ablation study, we adjust the setting based on ADM [2] to make the parameters and FLOPs close to ViT-B. The setting is

- Base channels: 192
- Channel multipliers: 1, 2, 2, 2
- Residual blocks per resolution: 3
- Attention resolutions: 8, 16
- Attention heads: 4

We also conduct experiments with the same architecture (296M) in ADM [2] on ImageNet 64×64. After 900K training iterations with batch size 1024, it could achieve an FID score of 2.11.

For high resolution generation on ImageNet 256×256. We use the 395M setting from LDM [8], which operates on the $32 \times 32 \times 4$ latent space.

3.2. Training Settings

The training iterations and learning rate have been reported in the paper. We use AdamW [5, 4] as our default optimizer. (β_1, β_2) is set to $(0.9, 0.999)$ for UNet backbone. Following [1], we set (β_1, β_2) to $(0.99, 0.99)$ for ViT backbone.

3.3. Sampling Settings

If not otherwise specified, we only use EDM's [3] Heun sampler. We only adjust the sampling steps for better results. For ablation study with ViT-B and UNet, we set the number of steps to 30. For ImageNet 64×64 in Table 4, the number of steps is set to 20. For ImageNet 256×256 in Table 5, the number of sampling steps is set to 50.

4. Additional Results

4.1. Ablation Study on Pixel Space

In the paper, most of the ablation study is conducted on ImageNet 256×256's latent space. Here, we present the results on ImageNet 64×64 pixel space. We adopt a ViT-B model as our backbone and train the diffusion model for 800K iterations with batch size 512. Our predicting targets are $\mathbf{x}_0$ and ϵ and they are equipped with our proposed simple

Min-SNR-γ loss weight ($\gamma = 5$). We adopt the pre-trained noisy classifier at 64×64 from ADM [2] as conditional guidance. We can see that the loss weighting strategy contributes to the faster convergence for both $\mathbf{x}_0$ and ϵ.

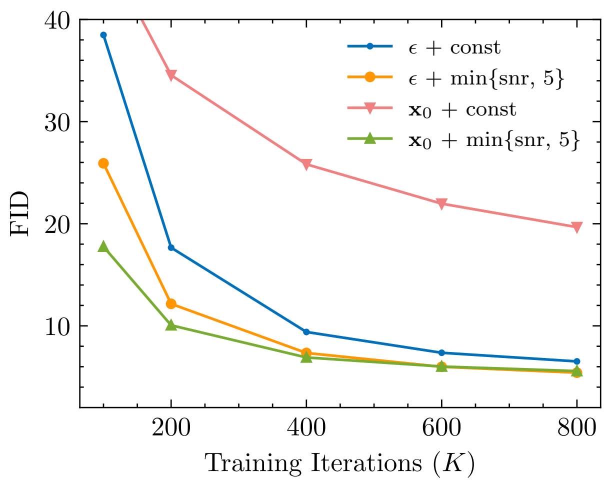

Figure 1: Ablate loss weight design in pixel space (ImageNet 64×64). We adopt DPM Solver [6] to sample $50k$ images to calculate the FID score with classifier guidance.

4.1.1 Min-SNR-γ on EDM

We also apply our Min-SNR-γ weighting strategy on the SoTA "denoiser" framework EDM. We find that our strategy can also help converge faster in such framework in Figure 2. The specific implementation is to multiply $\frac{\min\{\text{SNR},5\}}{\text{SNR}}$ in `EDMLoss` from official code[1]. We keep the same setting as official ImageNet-64 training setting, including batch size and optimizer. Due to the limit of compute budget, we did not train the model as long as that in EDM [3] (about 2k epochs on ImageNet). We use 2^{nd} Heun approach with 18 steps (NFE=35). The curve in Figure 2 reflects the FID's changing with training images.

4.2. Comparison with UGD

We compare our methods with UGD under the same computation cost in Figure 8. For UGD, first, it requires computing the gradients among all the timesteps (1000 by default), then it needs hundreds of optimization steps (250 steps following [10]) to compute the optimal loss weight. Thus it takes about 3.3 minutes per iteration on 16G-V100 GPUs. However, our method needs about 0.2 seconds for each iteration. Though we can speed up UGD with some implementation improvements, it's still 20 times slower than our method. Thus it's infeasible for practical use.

[1]https://github.com/NVlabs/edm.git

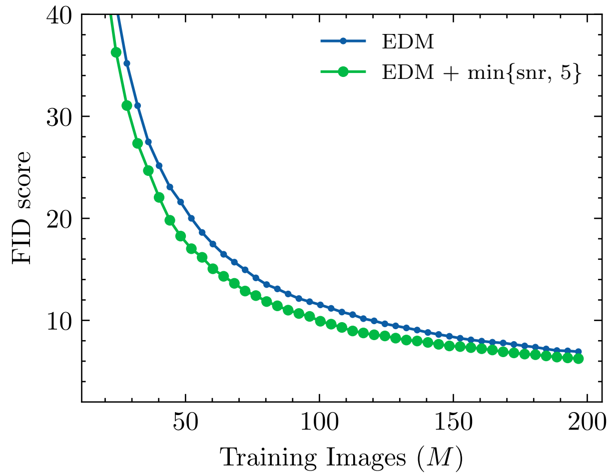

Figure 2: Effect of Min-SNR-γ on EDM [3].

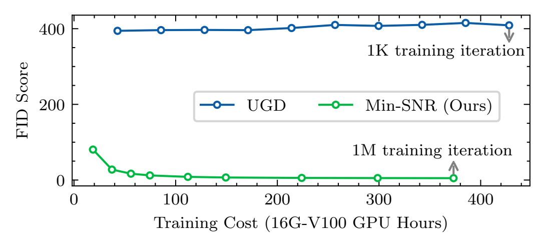

Figure 8: UGD trains 1000 times slower than Min-SNR.

4.3. Visual Results on Different Datasets

We provide additional generated results in Figure 9-12. Figure 9 shows the generated samples with UNet backbone on CelebA 64×64. Figure 10 and Figure 11 demonstrate the generated samples on conditional ImageNet 64×64 benchmark with ViT-Large and UNet backbone respectively. The visual results on CelebA 64×64 and ImageNet 64×64 are randomly synthesized without cherry-pick.

We also present some visual results on ImageNet 256×256 with our model which can achieve the FID 2.06 in Figure 12.

4.4. Variance of Our Results

We evaluated the performance of our model using 3 different random seeds and calculated the mean $\pm$ standard deviation to be 2.06 ± 0.01. This demonstrates that our reported results are robust to randomness.

4.5. Consistency of Sampler

We opt for the 2^{nd} Heun Sampler owing to its robustness and efficiency. Each method picked different samplers to achieve their best performance, *e.g.*, U-ViT in Tab.3 adopts EM-1000 sampling and uses 50-step DPM-solver in Tab.5. ADM-G leverages an additional noisy classifier for sam-

Figure 9: Additional generated samples on CelebA 64×64. The samples are from UNet backbone with 1.60 FID.

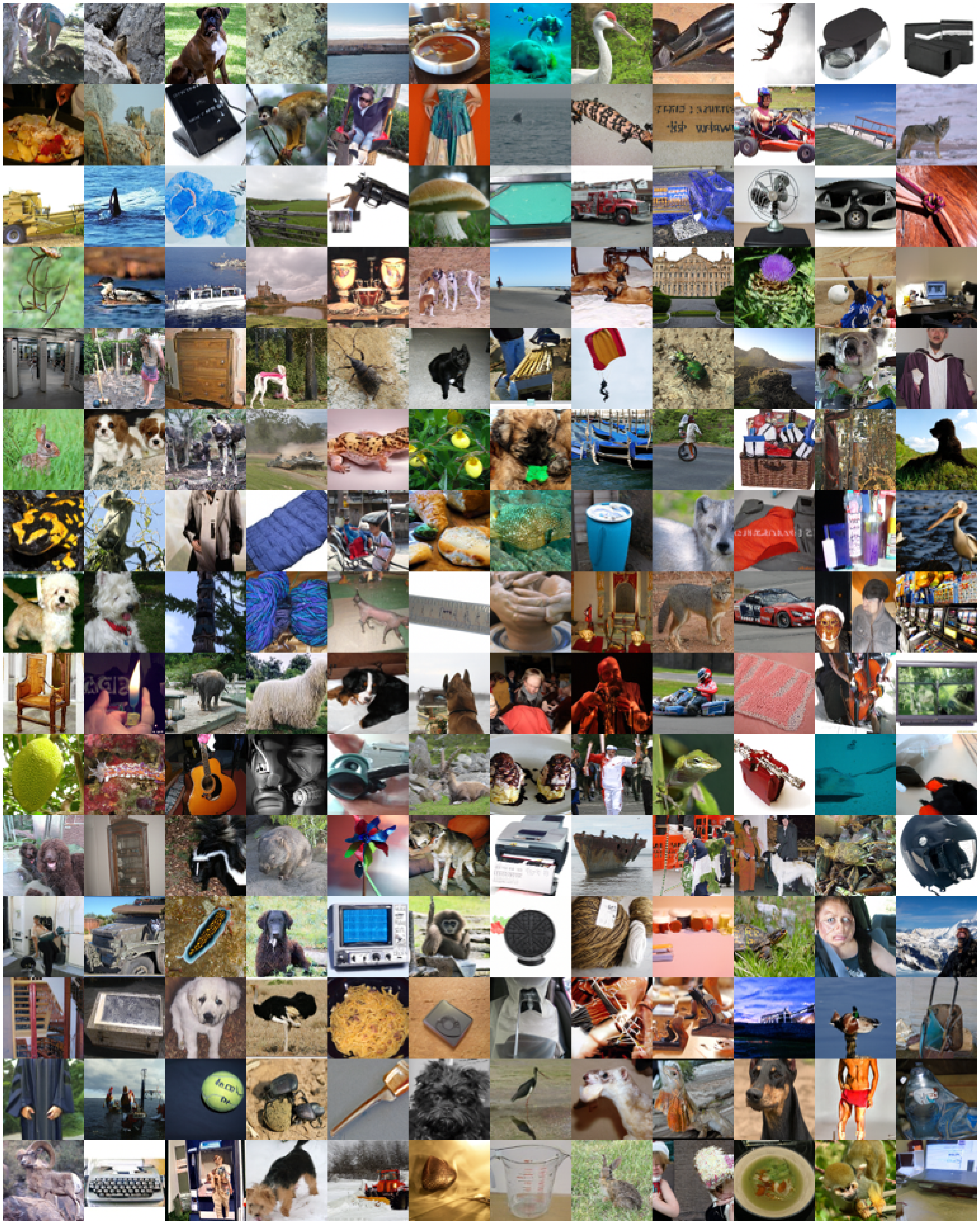

Figure 10: Additional generated samples on ImageNet 64×64. The samples are from ViT backbone with 2.28 FID.

Figure 11: Additional generated samples on ImageNet 64×64. The samples are from UNet backbone with 2.14 FID.

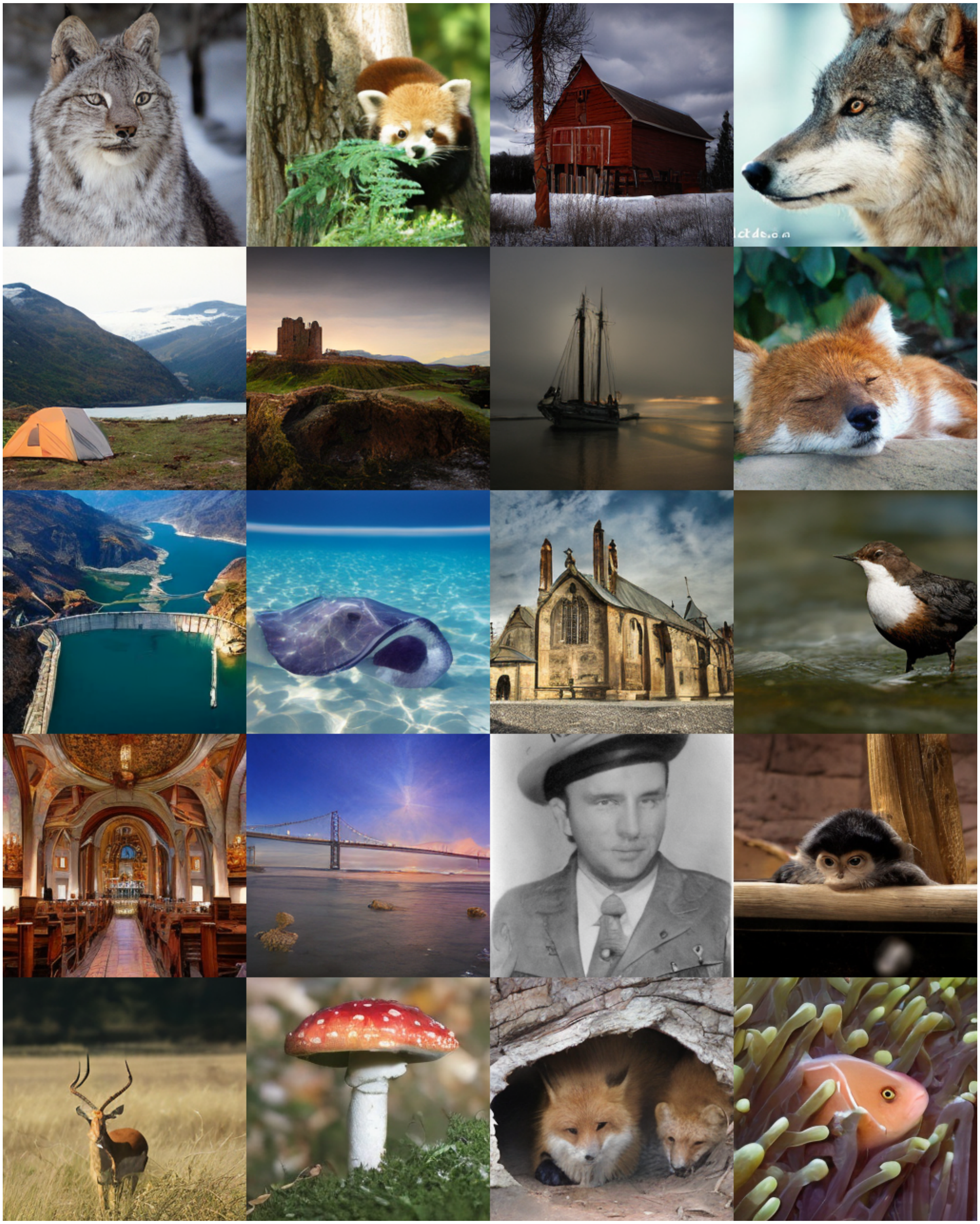

Figure 12: Additional generated samples on ImageNet 256×256. The samples are from ViT backbone with 2.06 FID.

pling. iDDPM even adopts a two-stage non-uniform sampling. We report the best results from their paper. These tables can prove our method's strong results on different datasets. Meanwhile, Tab.1 and Tab.2 adopt the same sampler for a fair comparison to demonstrate the effectiveness.

References

[1] Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. *arXiv preprint arXiv:2209.12152*, 2022. 2

[2] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021. 2, 3

[3] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. 2, 3

[4] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2014. 2

[5] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018. 2

[6] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *ArXiv*, abs/2206.00927, 2022. 3

[7] William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022. 2

[8] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 2

[9] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022. 2

[10] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31, 2018. 3

[11] Jianlin Su. Talking about multi-task learning (2): By the way of gradients, Feb 2022. 1

[12] John Von Neumann and Oskar Morgenstern. Theory of games and economic behavior, 2nd rev. 1947. 1