Hyperparameters are all you need: Using five-step inference for an original diffusion model to generate images comparable to the latest distillation model

Zilai Li *lizilai2008@163.com*
Independent researcher

Abstract

The diffusion model is a state-of-the-art generative model that samples images by applying a neural network iteratively. However, the original sampling algorithm requires substantial computation cost, and reducing the sampling step is a prevailing research area. To cope with this problem, one mainstream approach is to treat the sampling process as an algorithm that solves an ordinary differential equation (ODE). Our study proposes a training-free inference plugin compatible with most few-step ODE solvers. To the best of my knowledge, our algorithm is the first training-free algorithm to sample a 1024 x 1024-resolution image in 6 steps and a 512 x 512-resolution image in 5 steps, with an FID result that outperforms the SOTA distillation models and the 20-step DPM++ 2m solver, respectively. Based on analyses of the latent diffusion model's structure, the diffusion ODE, and the Free-U mechanism, we explain why specific hyperparameter couplings improve stability and inference speed without retraining. Meanwhile, experimental results also reveal a new design space of the latent diffusion ODE solver. Additionally, we also analyze the difference between the original diffusion model and the diffusion distillation model via an information-theoretic study, which shows the reason why the few-step ODE solver designed for the diffusion model can outperform the training-based diffusion distillation algorithm in few-step inference. The tentative results of the experiment prove the mathematical analysis.

1 Introduction

The diffusion probability model (DM) proposed by Ho et al. (2020a), a state-of-the-art image generation model with solid mathematics instruction on image sampling, is among the most popular research areas in computer vision. A typical diffusion model, such as stable diffusion (SD) (Rombach et al. (2022)), is inspired by the Brownian diffusion process in physics. It consists of two different processes: diffusion and reverse. The diffusion process adds small variants of Gaussian noise to the image step by step to convert an image into a pure Gaussian noise, while the reverse process uses the score matching algorithm (Song et al. (2020b)) to train a neural network studying the gradient of the diffusion process by denoising noisy images (Luo (2022)).

Like the Brownian process in physics, the diffusion process can be described by a diffusion stochastic differential equation (SDE). Moreover, the research proposed by Song et al. (2020b) shows that there exists an ordinary differential equation (ODE) to describe the reverse process. And using the existing mathematical tool for ODE solving enhances the capability of the diffusion model to sample an image.

However, compared to typical algorithms such as generative adversarial networks and variational autoencoders, solving diffusion ODEs still requires substantial computational resources. Hence, coping with this problem remains an active research area.

Research related to it can be categorized into two classes. The first class proposes a training-free algorithm aimed at providing a good SDE and ODE solver tailored to the diffusion process (Ho et al. (2020a); Lu et al. (2025); Song et al. (2020a); Zhao et al. (2023); Karras et al. (2022); Liu et al. (2022)). The second

class deems the whole inference process, which uses a neural network iteratively, as a large neural network, and the algorithm reducing the inference steps is equivalent to train a small distillation model (Luo et al.; Song & Dhariwal (2023); Lin et al. (2024); Yin et al. (2024b;a); Salimans & Ho (2022); Sauer et al. (2024); Kim et al. (2023); Song et al. (2023); Liu et al. (2023)). The training-free method typically necessitates a 20-step inference, like the DPM++ solver proposed by Lu et al. (2025), while algorithms requiring additional training can generate an image within four to eight steps. However, our results show that by selecting the appropriate hyperparameter for the ODE solver and adding a training-free diffusion model decorator to leverage the latent diffusion model's capabilities, the inference speed can exceed expectations.

To that end, we propose a new inference method that generates 512 x 512 images in 5 steps without additional training, and its Frechet Inception Distance (FID) (Heusel et al. (2017)) performance surpasses both the DPM++ 2m solver and the SOTA distillation model in 20 steps and 8 steps, respectively. We also apply our algorithm to 1024 x 1024 image generation, and the FID performance of the eight-step and six-step inference exceeds most of the latest distillation models. We perform extensive experiments using the COCO 2014 (Lin et al. (2014)), COCO 2017 (Lin et al. (2014)), and LAION datasets (Schuhmann et al. (2022)), and exhibit the generation result in Fig 1. We also use information theory to explain why our algorithm achieves a strong FID result.

Figure 1: The comparison between our algorithm and UniPC solver in few step inference

2 Related Work

2.1 Diffusion ODE

The image-generation process of the diffusion model can be deemed as solving a special diffusion ODE. A typical diffusion model has two processes: forward and reverse. The forward process perturbs the image by adding Gaussian noise to it, which can be described by the following SDE:

$$dx = f(t)xdt + g(t)dw, \tag{1}$$

where the w is a standard Wiener process. Choosing different $g(t)$ and $f(t)$ corresponds to applying different algorithms. In this study, we use the variant preserve (VP) SDE proposed by Song et al. (2020b) to describe the diffusion process, which corresponds to the denoised diffusion probability model (DDPM) algorithm (Ho et al. (2020b)). Specifically, its choice is as follows:

$$f(t) = -\frac{1}{2}\beta(t) = \frac{d\log\alpha_t}{dt}, \tag{2}$$

$$g(t) = \sqrt{\beta(t)}, \tag{3}$$

$$\beta(t) = \beta_{min} + t(\beta_{max} - \beta_{min}). \tag{4}$$

Explicitly, in the diffusion process mentioned above, the variant of noise added at step t is β_t. Explicilty, β_{max} and β_{min} can be determined arbitrarily if it can ensure $p(x_T|x_0)$ is a standard Gaussian distribution. In the timestep t, the noisy feature x_t with given x_0 belongs to a Gaussian distribution described in the eq. (5).

$$p(x_t|x_0) = \mathcal{N}(\alpha(t)x_0; \sigma^2(t)I). \tag{5}$$

Typically, to transform an image into a standard Gaussian noise, the DDPM algorithm requires $T = 1000$ steps, and α_T is closed to the zero. Additionally, using the reparameterization algorithm, we can sample x_t in a single step (Ho et al. (2020a)).

In contrast, the reverse process, which corresponds to generating an image from a pure Gaussian noise, can be described by the following time-dependent ODE proposed by Karras et al. (2022):

$$\frac{dx}{dt} = \frac{\dot{s}(t)}{s(t)}x_t + s(t)^2\dot{\sigma}(t)\epsilon_\theta(x_t, t, c), \tag{6}$$

where c is the condition of input, like the caption that describes the image, and the ϵ_θ is modeled by a neural network. The ϵ_θ has a relation between the score function, which is described in the eq. (7):

$$\begin{aligned} \epsilon_\theta(x_t, t, c) &= -\sigma(t)score_\theta(x_t, c), \\ score_\theta(x_t, c) &= \nabla_{x_t} \log p_\theta(x_t, c), \end{aligned} \tag{7}$$

And the $p_\theta(x_t, c)$ is the probability distribution of noisy images x_t modeled implicitly by the neural network with parameters θ. Moreover, the score function normally used in VP SDE is trained by eq. (8)

$$\mathcal{L}_{DSM}(\theta) = \mathbf{E}_{x_0\sim q_0, n\sim\mathcal{N}(0,\sigma^2(t)I)}[||\nabla_{x_t} \log p_\theta(x_t) + \frac{n}{\sigma_t}||_2^2], \tag{8}$$

and $q_t(x)$ is the probability distribution of the random variables sampled by adding noise to the training images in timestep t, and q_0 means the distribution of clean images.

In the eq. (6), the $s(t)$ and $\sigma(t)$ is defined by eq. (9) and eq. (10):

$$\sigma(t) = \sqrt{e^{\frac{1}{2}\beta_d t^2 + \beta_{min} t} - 1}, \tag{9}$$

$$s(t) = 1/\sqrt{e^{\frac{1}{2}\beta_d t^2 + \beta_{min} t}}, \tag{10}$$

where $\beta_d = \beta_{max} - \beta_{min}$.

Deeming the loss of the DDPM algorithm as a score function described by the eq. (8) can help exploit the advantage of adapting the original diffusion model to the common diffusion ODE framework proposed by Karras et al. (2022).

2.2 Diffusion ODE Solver

Sampling an image from a diffusion model is equivalent to solving the eq. (6). Via the research from Lu et al. (2022), when given a noisy input x_s and a target timestep t, the correct noisy feature output x_t can be gained by solving the eq. (11) :

$$x_t = \frac{\alpha(t)}{\alpha(s)}x_s - \alpha_t \int_{\lambda_s}^{\lambda_t} e^{-\lambda}\epsilon_\theta(x_{t(\lambda)}, \lambda)d\lambda, \tag{11}$$

where λ_t is the log Signal-to-Noise Ratio (SNR) of the noisy feature x_t.

Sometimes, to cope with the high guidance scale on the classifier free guidance algorithm (CFG) (Ho & Salimans (2022)), researchers tend to replace eq. (11) with the eq. (12) to get a better result (Lu et al. (2025)):

$$x_t = \frac{\sigma(t)}{\sigma(s)} x_s - \sigma_t \int_{\lambda_s}^{\lambda_t} e^{\lambda} \hat{x}_\theta(x_{t(\lambda)}, \lambda) d\lambda, \tag{12}$$

where $\hat{x}_\theta$ is a clean feature predicted indirectly by the neural network.

By using the Taylor Expand in the ϵ_θ or x_θ (Lu et al. (2022; 2025)), we get the following eq. (13):

$$\begin{aligned} x_t = \frac{\alpha(t)}{\alpha(s)} x_s - \alpha_t \Sigma_{n=0}^{k-1} \epsilon_\theta^{(n)}(x_{t(\lambda)}, \lambda) \int_{\lambda_s}^{\lambda_t} e^{-\lambda} \frac{(\lambda - \lambda_s)^n}{n!} d\lambda \\ + \mathcal{O}(h_i^{k+1}), \end{aligned} \tag{13}$$

However, calculating the neural networks' deviation requires substantial computational resources. Researchers typically use the Runge-Kutta method (Lu et al. (2025); Zhao et al. (2023)) to approximate the deviation.

Explicitly, the DPM++ 2s algorithm approximate x_t via the following formula:

$$\begin{aligned} x_t = \frac{\sigma(t)}{\sigma(s)} x_s - \alpha_t (e^{-h_t} - 1) \hat{x}_\theta(x_s, s) \\ + \frac{1}{2r_t} * \alpha_t (e^{-h_t} - 1) * (\hat{x}_\theta(\mu, m) - \hat{x}_\theta(x_s, s)), \end{aligned} \tag{14}$$

in which r_t can be selected arbitarily between 0 and 1, and h_i, m and μ_i is given below:

$$h_t = \lambda_t - \lambda_s, \tag{15}$$

$$r_t = \frac{\lambda_m - \lambda_s}{h_i}, \tag{16}$$

$$\mu_t = \frac{\sigma_m}{\sigma_s} x_s - \alpha_m (e^{-r_t h_t} - 1) \hat{x}_\theta(x_s, s), \tag{17}$$

To get the first-order deviation of the ϵ_θ, the number of function evaluations (NFE) in DPM++ 2s is 2. While DPM++ 2m utilizes the latest cached output in the previous inference to calculate the approximation of the first-order deviation, and hence only requires 1 NFE, which is a typical multistep algorithm in ODE solving. Simultaneously, the UniPC and DPM V3 solver cached all of the previous output to correct and predict a high-order deviation.

The eq. (6) describes a continuous curved trajectory, and the ODE solver approximates it by discretizing the path with a series of lines. Research on reverse ODE solvers aims to precisely estimate the next step, thereby minimizing truncation error from pool discretization in the few-step inference. On the other hand, the discrete method proposed by Karras et al. (2022) does not aim at design a new ODE solving algorithm, but proposes a choice of moving distance h_t to enhance the sample quality. Via the analysis of the Karras' research, the h_t should be large at the beginning of the inference, and then decrease monotonically.

2.3 Diffusion Distillation

The training-free method optimizes the truncation error at each step and thus minimizes the need for additional inference steps for the precise discretization. Unlike this method, the distillation algorithm (Lin et al. (2024); Sauer et al. (2024); Luo et al.; Song & Dhariwal (2023); Salimans & Ho (2022)) considers the whole inference process of applying a neural network iteratively as an extensive neural network. The distillation algorithm, which typically trains a small neural network, reduces its inference steps. However, training a neural network with high-quality content, fast inference, and no additional parameters is difficult. Thus, the

distillation algorithm, through the analysis of information theory (Li & Zhang (2025)) and experiments, can sometimes reduce the diversity of the generated images, and affect the FID performance. This approach may pose a potential risk in the future. AI training requires the use of Internet data, and in some countries, like Japan, all the data can be used in AI training by obeying a series of basic rules. However, when a human draws an image via their craft, they will sometimes be affected by the image they have previously seen. If they generate a similar image unconsciously, without directly using a similar work as a reference during creation, such a generation will not be deemed plagiarism. Compared to this, generative models trained on Internet images are often criticized, with people claiming that if those models generate content similar to their dataset, it should be considered plagiarism. In such a situation, diversity is a metrics that affect people's impression of generative content.

2.4 Latent Diffusion Model

Another method for reducing the computational requirements of inference is to encode images into a low-dimensional latent space. Stable diffusion performs inference in a 64 x 64 x 4 latent space to generate a 512 x 512 x 4 PNG image. One experiment from Rombach et al. (2022) shows that the semantics of the image, which rely on the long-range relationship between different parts of the images, can be modeled by an attention mechanism (Vaswani et al. (2017)). In contrast, its perceptual information, which provides the image with sufficient real details, is strongly tied to local regions and can be modeled by a convolutional neural network (CNN) (e.g. Li et al. (2021)). Hence, the stable diffusion model first encodes the image into a low-dimensional latent space via a β-variational autoencoder (Burgess et al. (2018)) to let the following inference ignore the perceptual detail. Then, it performs diffusion inference in latent space via a U-Net (Ronneberger et al. (2015)) with an attention mechanism to model the semantic information. Experiments show that compared with vector quantify VAE (Wu & Flierl (2020)), the normal β-VAE with a small KL-term constraint generates higher-quality images.

3 Approach

3.1 Algorithm Based on Truncation Error Analysis

To design a training-free, few-step inference algorithm, we should first consider the meaning of the original diffusion models' output, the score function. And then compare it with the distillation model's output.

By rewriting the eq. (8) as follows:

$$\mathcal{L}_\theta = \frac{E_{x_0}[q(x_t|x_0)||s_\theta - \nabla_x \log q(x_t|x_0)||^2]}{E_{x_0}[q(x_t|x_0)]}, \tag{18}$$

We find out that the score function is pointing to the average of all possible images that can cause this noisy image. In an extreme few-step inference case, like one or two NFE inference, the final output will be blurred, since in that step, the score function only points to the centroid of images. Meanwhile, the distillation algorithm, like rectify flow proposed by Liu (2022), will tend to modify the output and let it point to a special sample.

Although the task of modifying the score function to allow it to point to a special sample without additional training is hard, one can presume that the score function $\nabla_x \log q(x_t)$ can be approximated by the conditional score function $\nabla_x \log q(x_t|x_0)$ in the least noisy stage, since the possible images are very limited. In this situation, the inference task can be approximated by a pure denoising task, and we do not need further inference if we can modify the U-Net to perform the denoising task without additional training.

Since the changing $\nabla_x \log p_\theta(x_t)$ to the $\nabla_x \log p_\theta(x_t|x_0)$ means removing the blurred part of the result caused by the average, one of the natural method is to enhance the denoise capability of U-Net, which can be achieved by modifying the skip connection and the backbone feature of the U-Net.

Previous study (Si et al. (2024)) adds a scalar, fast Fourier transformation (FFT) and inverse fast Fourier transformation (iFFT) modules to the skip connection, which the backbone feature being modified by the scalar can use to enhance denoising capability. Moreover, the skip connection being modified by the FFT

and iFFT can prevent further smoothing and add details by amplifying the low-frequency part of the noisy backbone feature map. The decorator only applies to the first two skip connections. The scaler that affects the backbone feature is b_1 and b_2, while the scaler that affects the FFT and iFFT is s_1 and s_2.

However, we don't know which step could be changed simply from the score function to the conditional score function, and one of the vanilla ways is to apply free-U at all time steps. This method has been seen as helpful to slightly modify the score function and make it more similar to the distillation's output. The previous research from Ma et al. (2024) also shows that compared to the original diffusion model, the diffusion distillation model has small skip connection scalers, since the skip connection itself will abate the result in the block below that layer, while the few-step inference tends to excute a huge transportation from a standard Gaussian distribution to a complex probability distribution, and hence require the skip connection reduce its' effect that abate the block below it. However, via the mathematical analysis of the diffusion ODE and the experimental result. This vanilla method in the extreme few-step inference of high-resolution images, such as 6 NFE, would not work properly.

Explicitly, consider that the normal ResNet (He et al. (2016)) can be viewed as an ODE (Chen et al. (2018)) by using the following formula:

$$h_{t+1} = h_t + f_\theta(h_t, t), \tag{19}$$

$$\frac{dh}{dt} = F_\theta(h_t, t), \tag{20}$$

where h_t is the feature output in layer t.

The U-Net, as a black box ODE solver, can be regarded as another ODE. Thus, modifying the skip connection and scale backbone feature will further increase the norm of the output, increasing the moving distance of the diffusion reverse ODE. In the situation of the few-step inference, it will be a serious problems, since the weakness of the poor discretization method will be amplified by this behaviour. The related mathematic details will be mentioned below. We also visualize the decoded features along the trajectories of different samplers in fig. 2.

To further show the influence of the Free-U on the black box ODE solver, we then visualize the decoded trajectory using different discrete methods by applying the same sampler DPM++ 1s sampler in fig. 3. The standard time schedule using Free-U generates a trajectory similar to that of Karras et al. (2022) time schedule without Free-U, providing evidence that Free-U increases the moving distance at the beginning of the inference, which determines the image layout composition.

Another method is to parameterize the start step of using the U-Net decorator in an existing solver. However, experimental results show that this Vallina method doesn't work either. We visualize the 6-step inference result combining the free-U and UniPC solver in fig. 4.

To address black-box ODE-solving problems, we should identify the characteristics of the diffusion ODE.

The basic logic of the ODE solver is to discretize a continuous curve trajectory into a series of lines. And to achieve the best performance, the ODE solver should determine the length of each discrete line for different ODEs, such as the Fehlberg method (see e.g. Atkinson et al. (2009)) that adjusts the discrete length based on the error per unit stepsize.

For the diffusion ODE, when the moving distance in step t to $t+1$ is large during inference, the weakness of the poor discretion in the noisy stage will not be as fatal as the non-noisy steps. These insights have been proved by Karras et al. (2022) via an experiment that discretizes the inference process with a uniform gap between each timestep. And it obtains an array of steps $\{t_i\}$. Then, the researchers calculate the root mean square error (RMSE) between the one-step inference from t_i to t_{i+1} and the 200-step inference from t_i to t_{i+1}. The experimental results show that the truncation error increases monotonically as the noise variance add in the input feature at the time step t_i decreases.

A previous study from Karras et al. (2022) shows that the global truncation error is bound by $||e_N|| \leq Emax_i||\mathcal{T}_i||$, where $||\mathcal{T}_i|| \leq Ch_i$. The E depends on the discrete step N, the start time step t_0, the ending timestep t_N, and the existing constant C. The meaning of this upper bound is that if we discretize the

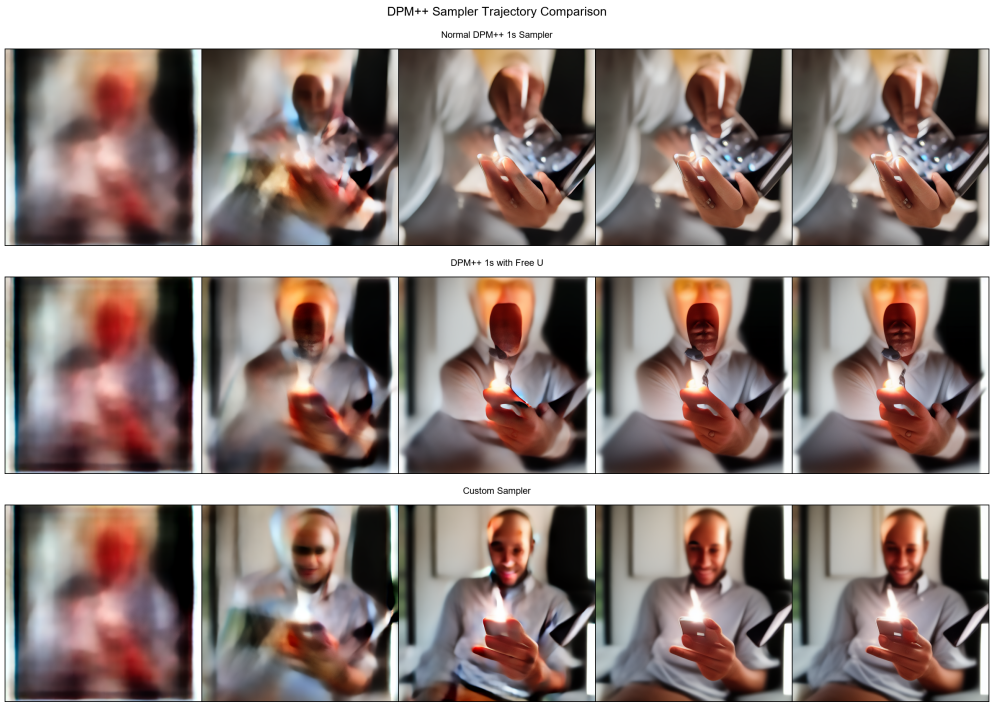

Figure 2: The comparison between our custom DPM++1s sampler and the original DPM++1s sampler. The first line is the result trajectory of the original DPM++1s sampler with Karras' schedule, the second one is Karras' schedule + Free-U, and the last one is our sampler. The results show that if we apply Free-U in a few-step sampling without involving additional tricks, the generation result will be degraded.

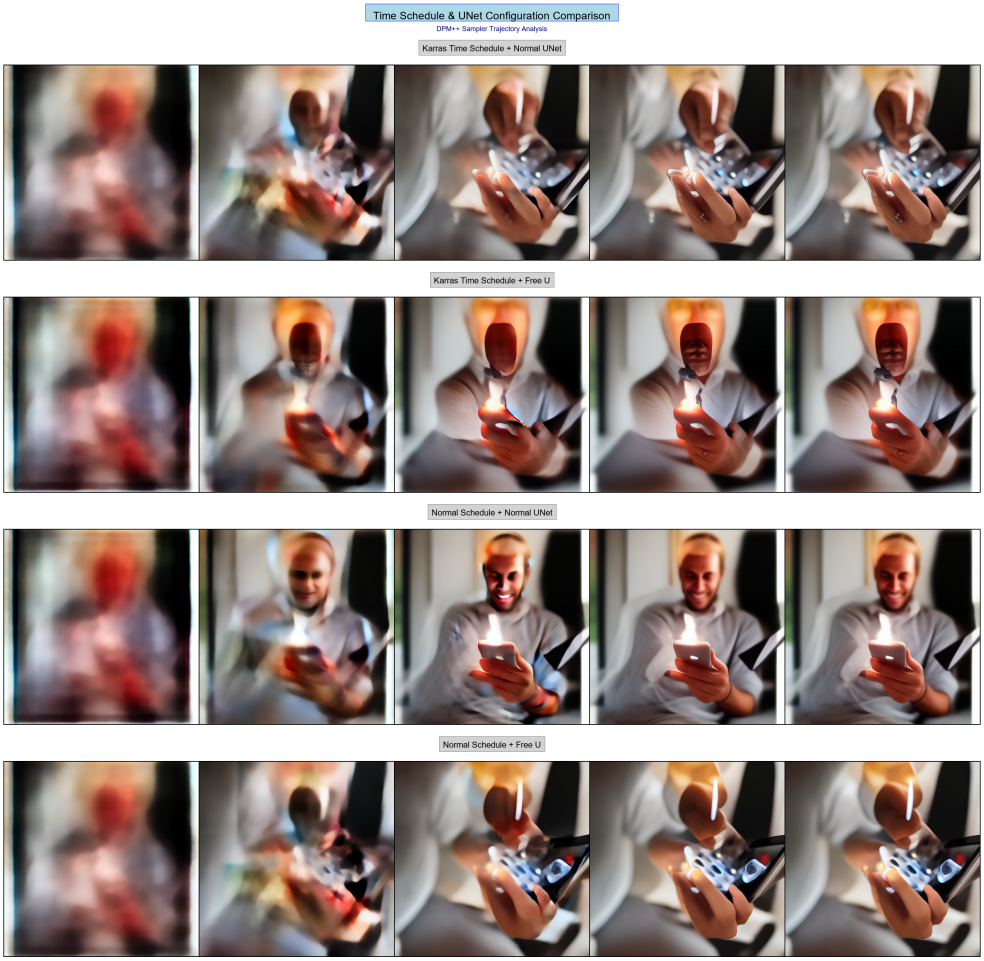

Figure 3: The comparison between different discrete methods when using the same DPM++ 1s Sampler in a few-step inference. The first line exhibits the trajectory of the normal Karras discrete method. The second one exhibits the trajectory of the Karras discrete method + Free-U. The third exhibits the normal discrete method with a normal U-Net, and the final one exhibits the normal discrete method with Free-U. And the final line is similar to the first line.

Figure 4: The result that starts using free-U in the different steps of the UniPC solver.

solving process uniformly, and the discretizing method is not sufficiently precise, the global truncation error will be limited by the less noisy part, which contributes the highest h_i.

Based on this observation, Karras et al. (2022) proposes a following discrete method:

$$\sigma_i = (\sigma_{min}^{\frac{1}{p}} + \frac{i}{N-1}(\sigma_{min}^{\frac{1}{p}} - \sigma_{max}^{\frac{1}{p}}))^p. \tag{21}$$

eq. (21) controls the variant σ_i via p, and if the p increases, the $\sigma_i - \sigma_{i-1}$ will increase in the noisier part and decrease in the less noisy part. Moreover, according to the observation from the Li & Zhang (2025), if we choose $p = 7$ and $N = 20$, the fourth-to-last step corresponds to the first $\frac{6}{1000}$ perturbation steps, and it will cost $\frac{1}{5}$ inference budget. Considering the U-Net has 860 million parameters, and the variance of the $p(x_t|x_0)$ is approximately $\frac{6}{1000}$, performing four-step ODE inference at this point is illogical. Moreover, the β-VAE used for decoding is also trained with a small KL-term constraint, which enables it to handle the noise in the latent variable.

Based on that observation, Li & Zhang (2025) proposes another discrete method:

$$t_i = (t_{min}^{\frac{1}{p}} + \frac{i}{N-1}(t_{min}^{\frac{1}{p}} - t_{max}^{\frac{1}{p}}))^p. \tag{22}$$

When choosing $p = 1.2$, the h_i in both the noisy part and the less noisy part are larger than those in Karras's original method. In such a situation, $t_i - t_{i-1}$ will increase in the noisier part. The eq. (4) shows that t_i is proportional to the variance of noise added in x_t, which can be deemed as the speed of movement at timestep t_i. The increase of $t_i - t_{i-1}$ in the noisy part aligns with the rise in speed, while the rise of $\sigma_i - \sigma_{i-1}$ aligns with the increment of the moving distance, hence it can cope with the problem of the original Karras schedule mentioned above.

However, both of those discrete methods have been commonly used in all kinds of images. Our research shows that, even with the same time schedule and UniPC solver, perceptual quality in the few-step inference remains substantially different across image resolution and image type. Hence, the design space of hyperparameters for different ODE solvers has not yet been totally exploited. We find a discrete method, which changes the eq. (23), a formula with a special design reason mentioned later, by adding one more step to get clean denoised images. And use it to generate images at different resolutions and of different types in a few-step inference. We find that the result in fig. 5 has a noticeable gap in perceptual quality.

Figure 5: The comparison between the different resolution images and different type images. The first line contains 512 x 512 images generated by Stable Diffusion 1.5, the second line contains 1024 x 1024 images generated by Dreamshaper XL. The last lines contain high-resolution ACGN images generated by Illustrij Evo.

Based on the experimental results, designing a customized training-free algorithm for different image types requires exploring the design space of the ODE solver for the latent diffusion model.

Firstly, the decoder of the latent diffusion model, a β-VAE trained with a small KL-term constraint, can handle a small amount of noise; thereby, if we choose a time schedule that does not fully denoise the image, we can save further inference budget. Although the latent diffusion model theoretically does not require using the KL term to train an autoencoder, this optimization trick is very prevalent in the training of the diffusion model. The latest representation autoencoders (RAE) also add noise during the training to augment the decoder Zheng et al. (2025). The experimental results show that the decoder's capacity to handle noisy features varies with image resolution. In the fig. 6, the second line uses the same time schedule as the first line, which doesn't denoise totally, but those images in the second line have some blurred parts that harm the details of the images, while the first line doesn't confront this problem. And the third line output feature with less noise, and has a better results.

Figure 6: The comparison between the different resolution images generated by the same time schedule and ODE solver utilizing the same β-VAE.

Based on the aforementioned idea and the experiment results in Section 4, we first involve the following discrete method that aims to correct the output in the less noisy stage:

$$t_i = (t_{max}^{\frac{1}{p_1}} + \frac{i}{N}(t(\sigma_{stop})^{\frac{1}{p_1}} - t_{max}^{\frac{1}{p_1}}))^{p_1}, \tag{23}$$

where σ_{stop} is defined via the following formula:

$$\sigma_{stop} = (\sigma_{max}^{\frac{1}{p_2}} + \frac{stop}{M}(\sigma_{min}^{\frac{1}{p_2}} - \sigma_{max}^{\frac{1}{p_2}}))^{p_2}, \tag{24}$$

which could be nonzero to save the one-step inference budget. By applying this special time schedule, the problems of using Free-U while preventing an increase in moving distance could be addressed, since we could

only apply Free-U at timesteps that move a short distance. By leveraging the robustness of the β-VAE, the total moving distance has also been constrained.

The visual difference between our discrete method and the method proposed by the Karras' paper is in fig. 7

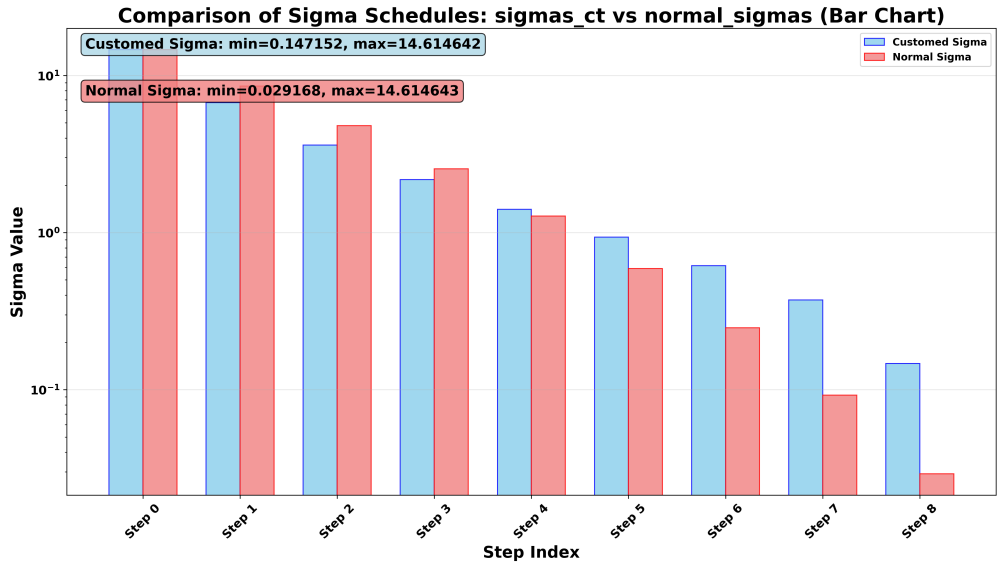

Figure 7: The comparison between our time schedule and the time schedule proposed by Karras

Meanwhile, since the total moving distance should be as limited as possible, we also apply the analytical first step (AFS) proposed by Dockhorn et al. (2022) during inference.

Our algorithm can be described by the algorithm 1

Algorithm 1: custom Karras scheduler in the common framework of Diffusion ODE solving

Data: number of inference step N, initiated noise ϵ, stop step in Karras schedule t_{stop}, the total step M of the original discrete method, discretion hyper parameter p_1 and p_2, U-Net augment step t_{aug}, ODE solver F_θ, augment U-Net method $freeU()$, a bool value afs to determine whether using afs, and an analytical first step solver without using neural network F

Result: final result image

$\sigma_{stop} = (\sigma_{max}^{\frac{1}{p_2}} + \frac{t_{stop}}{N+t_{stop}+1}(\sigma_{min}^{\frac{1}{p_2}} - \sigma_{max}^{\frac{1}{p_2}}))^{p_2}$;

$ts = \{t_i = (t_{max}^{\frac{1}{p_1}} + \frac{i}{N}(t(\sigma_{stop})^{\frac{1}{p_1}} - t_{max}^{\frac{1}{p_1}}))^{p_1}; 0 \leq i < N\}$

if afs **then**

  $ts.insert((\frac{t_0+t_1}{2}), at : 1)$

end

$x = \epsilon$

while t *in* ts **do**

  if t *equal to* t_{aug} **then** $freeU(F_\theta)$;

  ;

  if afs *and* t *equal to* 999 **then** $x = F(x)$;

  else $x = F_\theta(x)$;

end

3.2 Information Theory Analysis of the Different Algorithms

The experimental results in the section 4 show that the diversity of the generative images sampled from a distillation model is the main bottleneck of its FID performance. Given that all images are generated from the same prompts, and the distillation algorithm changes the score function from pointing to the centroid of all possible images to a special image, we assume the distillation model limits the diversity of images' low-frequency components, which is caused by modification of the score function at the start stage of the inference. Explicitly, some images with totally different semantics could have a similar, fuzzy structure with the same color block in a similar position. And the limitation on the possible fuzzy structure of images may also cause diversity problems. And we use information theory to explain why applying Free-U only in the final stage of the inference helps address this problem.

To introduce the relationship between information theory and machine learning, we should first review the historical confusion about machine learning's generalization capability.

A common point of confusion in deep learning at an early stage is how neural networks with so many parameters can still generalize (Breiman (2018)). The standard statistical model uses a limited number of parameters to concisely describe the phenomenon and capture the essence of the observed data, whereas deep learning is fundamentally different. For this problem, the study theory (Harman & Kulkarni (2012)) claims that limiting the number of functions that a model can express helps ensure that the model approaches the best functions it can have, and that it also bounds the distance between the final function a neural network study about and the average of functions.

Generally, a possible function expressed by a model is a hypothesis $\mathcal{H}$, and if the number of input data points is N, the number of possible hypotheses is less than 2^N. According to information theory, for a dataset $\mathcal{S}$ with data D, the typical set that a model can access is $2^{H(D)}$ (Yeung (2012)). For example, in a high-dimensional standard Gaussian distribution, a pure black image, although with the highest possible density, is not in the typical set. Considering the amount of data in the typical set, and the mutual information between the output and the middle feature of the neural network $I(y, z)$, we can further bound the models' capacity for potential hypotheses.

Explicitly, we can consider the probability of middle feature z given output y, and the $p(z|y)$ provides a match between y and z. The information bottleneck theory proposed by Tishby et al. (2000) shows that $2^{H(Y|Z)}$ denotes the volume of Y projected to the Z. Hence, $2^{I(Y,Z)} = \frac{2^{H(Y)}}{2^{H(Y|Z)}}$ further constraints the possible hypotheses by limiting the possible middle feature (Tishby et al. (2000)). As the mutual information bounds the distance between the study hypothesis and the average hypothesis, generative adversarial networks (GANs) (Goodfellow et al. (2020)), which face mode collapse (Creswell et al. (2018)) that only generate a subset of the dataset, can benefit from it by adding a mutual information constraint Jeon et al. (2021). One study (Li & Zhang (2025)) shows that this phenomenon also appears in the diffusion distillation, which consistently changes the score function and triggers an increase in mutual information. The basic idea of the proof in that study is related to β-variation auto-encoder (β-VAE). When β-VAE decreases β to let VAE generate high-quality images, the upper bound of the mutual information constraint will be abated. Meanwhile, the peak signal-to-noise ratio (PSNR) of the encoding feature Z will increase. From this perspective, the diffusion model can be deemed as a hierarchy VAE (Luo (2022)), while progressive distillation proposed by Salimans & Ho (2022) increases the mutual information by letting the model directly use $X_\theta(t+2)$ to replace the original output $X_\theta(t+1)$. Where X_θ is a noisy feature output by solving ODE. Hence, the PSNR of the middle output feature also increases.

However, it is intuitive that changing the direction of the score function to allow it to point to the final result, rather than the centroid of the possible image, will also affect the mutual information, even if the output does not have an increased PSNR. We provide further analysis regarding this.

Solver	FID	CLIP-Score
custom UniPC(6) apply Free-U on the every steps $w = 5.5$	28.88	30.96
custom UniPC(8) apply Free-U on the every steps $w = 5.5$	33.70	30.50
custom UniPC(6) don't apply Free-U $w = 5.5$	27.13	31.49
custom UniPC(8) don't apply Free-U $w = 5.5$	25.47	31.39
custom UniPC(6) start apply Free-U at 4th step $w = 5.5$	25.20	31.42
custom UniPC(8) start apply Free-U at 6th step $w = 5.5$	24.38	31.38
custom UniPC(6) apply Free-U on the every steps $w = 7.5$	39.62	30.33
custom UniPC(8) apply Free-U on the every steps $w = 7.5$	54.26	29.63
custom UniPC(6) don't apply Free-U $w = 7.5$	25.34	31.53
custom UniPC(8) don't apply Free-U $w = 7.5$	24.61	31.22
custom UniPC(6) start apply Free-U at 4th step $w = 7.5$	25.29	31.44
custom UniPC(8) start apply Free-U at 6th step $w = 7.5$	24.58	31.21

Table 1: Results of applying Free-U in different steps of inference.

The diffusion ODE can be regarded as a Markov process $Z_\theta(T) - Z_\theta(T-1) - ... - Z_\theta(t) - ...Z_\theta(0)$, in which $Z_\theta(t)$ is the noisy input of the diffusion model, and we have the following eq. (25):

$$\begin{aligned} &I(Z_\theta(0), Z_\theta(t)) = H(Z_\theta(0)) - H(Z_\theta(0)|Z_\theta(t)) \\ &\leq H(X(0)) - E_{Z_\theta(t)}[p(Z_\theta(0)|Z_\theta(t)) \\ &* \log p_\theta(Z_\theta(0)|Z_\theta(t)))], \end{aligned} \tag{25}$$

in which $H(X(0))$ is the entropy of the real $X(0)$ sample from the dataset.

In β-VAE, when given the random variable Z, the output Y is assumed to be sampled from a Gaussian distribution using Y as its mean, and abating the β makes the middle feature Z more similar to Y, enhancing the log likelihood of the Y. And in the distillation algorithm, eq. (25) shows that if we modify the $Z_\theta(t)$ to make it similar to the final output to enhance the log-likelihood by rectifying the trajectory, then the upper bound of the mutual information will also increase. Most distillation methods that do not increase the PSNR of the output will still face this problem. We evaluate the Precision and Recall of Distribution (PRD) (Sajjadi et al. (2018)) of flash diffusion and our sampler to further validate this claim.

4 Experiments

4.1 Ablation Study

We conduct an ablation Study on a customized UniPC solver that aims to generate a 1024 x 1024 image in a few-step inference guided by 5000 prompts sampling from the COCO 2017 dataset. Explicitly, for the 8-step inference, we employ $p_1 = 12$, $p_2 = 1.2$, $stop = 10$, $M = 12$, and $N = 8$ as the discrete setting. And for the 6-step inference, the discrete setting is $p_1 = 12$, $p_2 = 1.2$, $stop = 10$, $M = 12$, and $N = 6$. Then, we try the following setting: 1. apply Free-U on every step of inference, 2. apply Free-U only on the last few steps of inference, 3. Don't apply Free-U. The experimental result is in the table 1. The w in the table 1 is the guidance scale of inference, and the number in the parentheses is the total number of function evaluations (NFEs) of the model. Meanwhile, for the 8-step UniPC solver, we use a 2nd-order multistep algorithm, whereas the 6-step solver is 1st-order solver.

4.2 Performance

In this study, we employ our algorithm on the DPM++ 2m solver and UniPC solver. We employ $p_1 = 7$, $p_2 = 1.2$, $stop = 8$, $M = 12$, and $N = 8$ as the hyperparameters of the discrete setting in the eight-step inference when generating 1024 x 1024 and 512 x 512 images via DPM++ 2m. Additionally, $p_1 = 12$, $p_2 = 1.2$, $stop = 10$, $M = 12$, and $N = 8$ are applied on the custom UniPC solver in the eight-step inference

when generating 1024 x 1024 images. And for the six-step inference for generating 1024 x 1024-resolution images, the hyperparameters of the UniPC solver are $p_1 = 12$, $p_2 = 1.2$, $stop = 10$, $M = 12$, and $N = 6$. Meanwhile, for 512 x 512 images, the hyperparameters of the 8-step UniPC solver are $p_1 = 7$, $p_2 = 1.2$, $stop = 9$, $M = 12$, and $N = 8$. And for the customized 5-step UniPC solver for 512 x 512 images, the hyperparameters are $p_1 = 5$, $p_2 = 1.2$, $stop = 6$, $M = 8$, and $N = 5$. Then, the 6-step UniPC solver of the 512 x 512-resolution images has hyperparameters $p_1 = 5$, $p_2 = 1.2$, $stop = 6$, $M = 8$, and $N = 6$.

For the eight-step inference that uses DPM++ 2m to generate 512x512-resolution images, we avoid using free-U. And for the customized DPM++ 2m solver that generates 1024x1024-resolution images, the t_{aug} apply free-U is 8. For all UniPC solver customized to generate 512x512 resolution images, $t_{aug} = 4$. And $t_{aug} = 6$, $t_{aug} = 4$ for 6-step UniPC solver and 8-step UniPC solver customeized for 1024 x 1024-resolution images, respectively. And we only use AFS on the UniPC solver customized for the 1024 x 1024-resolution image. We don't guarantee we provide the best hyperparameters; we just ensure we provide hyperparameters that can sample images with FID performance comparable to that of the inference result from the 20-step DPM++ solver and the SOTA distillation algorithm.

We report the results of the inference that generate 512 x 512-resolution images in the COCO 2014, COCO 2017, and LAION dataset in table 2, table 3, and table 4. Moreover, the result of the 1024 x 1024 images is given in table 5, table 6, and table 7. Explicitly, the UniPC solver is a 2nd-order multistep solver for eight-step inference and a 1st-order multistep solver for five and six-step inference. The inference model that uses our algorithm for synthesizing 512 x 512 images and 1024 x 1024 images is Stable Diffusion v1.5 and Dreamshaper XL, respectively. Furthermore, all the synthesized images in table 2 and table 5 employs the random 10k COCO caption as a guidance prompt. In table 3 and table 6, for each image in the COCO 2017 dataset, we fetch one particular caption of it as guidance. Furthermore, we use 10k captions in LAION for table 4 and table 7. We also compare the FID performance and the clip score of AMED-Pulging solver, progressive distillation, SDXL turbo, SDXL-Lightning, Flash DiffusionXL, Flash Diffusion, and DMD2 (Yin et al. (2024a); Chadebec et al. (2025); Lin et al. (2024); Sauer et al. (2024)). Our approach outperforms state-of-the-art models for most of the datasets. However, some of the models (Yin et al. (2024a); Chadebec et al. (2025); Lin et al. (2024)) are trained in LAION, which do not separate the validation and the training sets, and may affect the comparison. The w in the table 2 and table 5 is the guidance scale of inference, and the number in the parentheses is the total number of function evaluations (NFEs) of the model. The b_1, b_2, s_1, and s_2 in the Free-U decorator is 1.1, 1.1, 0.9, 0.2.

We also calculate the PRD in a 6-step inference by sampling 10k captions from COCO 2014 and getting the corresponding image in the COCO dataset as a reference. PRD was first used to evaluate GAN. Considering two distributions Q and P, the first one is the distribution of the generative images, and the second one is the distribution of the original images, the precision shows how much of Q can be generated by P, and the recall exhibits how much P can be generated by Q. For example, if P is a subset of MNIST that only contains numbers ranging from 1 to 4, adding 5 and 6 to the generative distribution reduces the precision, while lacking modes like 3 and 4 will reduce the recall rate. Our experiment shows that the recall rate of the flash diffusion in the 6-step is less than that of us. The guidance scale of our sampler is 5.

5 Conclusion

In this study, we propose a plug-in method to enhance the sampling algorithm. By fully harnessing the potential capacity of the latent diffusion model—such as modifying the skip-connection at the right time, using the property of the decoder, which is a β-VAE trained with small KL-constraints, ignoring a small amount of noise in the final feature to reschedule the discretizing method, and choosing the proper ODE solver via truncation error analysis—we can sample a high-quality image with limited inference budget and avoid the need for additional training.

5.1 Limitation and the future work

Firstly, a diffusion model is a robust generative model (Chen et al. (2024)) that can prevent adversarial attacks. Most attacks that hack the latent diffusion model attack its VAE decoder. Thus, it remains unclear

Solver	FID	CLIP-Score
(Our) custom DPM++1s(5) $w = 7.5$	19.18	30.92
(Our) custom DPM++1s(6) $w = 7.5$	18.54	31.16
(Our) custom DPM++2m(8) $w = 7.5$	15.92	31.32
(Our) custom DPM++1s(5) $w = 5.5$	17.54	31.0
(Our) custom DPM++1s(6) $w = 5.5$	17.04	31.16
(Our) custom DPM++2m(8) $w = 5.5$	15.7	31.07
(Our) custom UniPC(5) $w = 7.5$	19.35	30.63
(Our) custom UniPC(6) $w = 7.5$	16.41	30.97
(Our) custom UniPC(8) $w = 7.5$	16.00	31.15
(Our) custom UniPC(5) $w = 5.5$	15.19	30.93
(Our) custom UniPC(6) $w = 5.5$	14.72	31.08
(Our) custom UniPC(8) $w = 5.5$	15.26	31.12
Original DPM++2m(20) $w = 7.5$	17.3	30.97
Original DPM++2m(20) $w = 5.5$	18.68	30.97
Flash Diffusion(5)	17.34	30.45
Flash Diffusion(6)	17.94	30.59
Flash Diffusion(8)	19.03	30.37
AMED Pulgin(8)	19.07	31.15
SDXL turbo(4)	23.24	31.66

Table 2: Results of 512 x 512 image generation. COCO 2014

Solver	FID	CLIP-Score
(Our) custom DPM++1s(5) $w = 5.5$	24.22	31.06
(Our) custom DPM++1s(6) $w = 5.5$	23.24	31.12
(Our) custom DPM++2m(8) $w = 5.5$	22.40	31.23
(Our) custom DPM++1s(5) $w = 7.5$	25.58	30.93
(Our) custom DPM++1s(6) $w = 7.5$	24.71	31.13
(Our) custom DPM++2m(8) $w = 7.5$	22.35	31.35
(Our) custom UniPC(5) $w = 7.5$	22.70	30.99
(Our) custom UniPC(6) $w = 7.5$	24.02	31.04
(Our) custom UniPC(8) $w = 7.5$	24.55	30.99
(Our) custom UniPC(5) $w = 5.5$	21.31	30.92
(Our) custom UniPC(6) $w = 5.5$	20.82	31.04
(Our) custom UniPC(8) $w = 5.5$	21.71	31.07
Original DPM++2m(20) $w = 7.5$	25.97	31.21
Original DPM++2m(20) $w = 5.5$	23.75	31.10
Flash Diffusion(5)	23.56	30.64
Flash Diffusion(6)	24.16	30.60
Flash Diffusion(8)	25.35	30.55
AMED Pulgin(8) $w = 7.5$	25.50	31.17
SDXL turbo(4)	30.92	31.66

Table 3: Results of 512 x 512 image generation. COCO 2017

Solver	FID	CLIP Score
(Our) custom DPM++1s(5) $w = 5.5$	19.61	31.81
(Our) custom DPM++1s(6) $w = 5.5$	18.45	32.10
(Our) custom DPM++2m(8) $w = 5.5$	17.74	32.42
(Our) custom DPM++1s(5) $w = 7.5$	19.88	31.45
(Our) custom DPM++1s(6) $w = 7.5$	18.60	31.97
(Our) custom DPM++2m(8) $w = 7.5$	17.52	32.41
(Our) custom UniPC(5) $w = 7.5$	20.43	30.87
(Our) custom UniPC(6) $w = 7.5$	17.56	31.63
(Our) custom UniPC(8) $w = 7.5$	17.38	32.08
(Our) custom UniPC(5) $w = 5.5$	17.44	31.73
(Our) custom UniPC(6) $w = 5.5$	16.84	32.08
(Our) custom UniPC(8) $w = 5.5$	17.32	32.26
Original DPM++2m(20) $w = 7.5$	18.30	32.67
Original DPM++2m(20) $w = 5.5$	17.33	32.47
Flash Diffusion(5)	15.40	31.60
Flash Diffusion(6)	15.80	31.63
Flash Diffusion(8)	16.87	31.57
AMED Pulgin(8) $w = 7.5$	18.06	32.44
SDXL turbo(4)	22.25	33.02

Table 4: Results of 512 x 512 image generation. Laion

Solver	FID	CLIP Score
custom DPM++2m (8) $w = 7.5$	17.84	31.62
custom DPM++2m (8) $w = 5.5$	19.62	31.58
custom UniPC (8) $w = 7.5$	18.18	31.23
custom UniPC (8) $w = 5.5$	17.87	31.40
custom UniPC (6) $w = 7.5$	18.80	31.42
custom UniPC (6) $w = 5.5$	19.07	31.45
Flash DiffusionXL (6)	22.23	31.00
Flash DiffusionXL (8)	23.25	30.89
SDXL-Lightning (8)	21.07	31.11
DMD2 (4)	19.64	31.64

Table 5: Results of 1024 x 1024 image generation. COCO 2014

Solver	FID	CLIP Score
custom DPM++2m (8) $w = 7.5$	24.42	31.71
custom DPM++2m (8) $w = 5.5$	26.04	31.56
custom UniPC (8) $w = 7.5$	24.58	31.21
custom UniPC (8) $w = 5.5$	24.38	31.38
custom UniPC (6) $w = 7.5$	25.29	31.44
custom UniPC (6) $w = 5.5$	25.20	31.42
Flash DiffusionXL (6)	28.74	30.96
Flash DiffusionXL (8)	29.63	30.88
SDXL-Lightning (8)	27.89	31.08
DMD2 (4)	26.19	31.65

Table 6: Results of 1024 x 1024 image generation. COCO 2017

Solver	FID	CLIP Score
custom DPM++2m (8) $w = 7.5$	19.25	32.22
custom DPM++2m (8) $w = 5.5$	20.82	32.07
custom UniPC (8) $w = 7.5$	21.59	31.73
custom UniPC (8) $w = 5.5$	20.41	32.06
custom UniPC (6) $w = 7.5$	22.51	31.82
custom UniPC (6) $w = 5.5$	22.33	31.87
Flash DiffusionXL (6)	20.88	31.64
Flash DiffusionXL (8)	21.30	31.61
SDXL-Lightning (8)	20.45	32.50
DMD2 (4)	16.51	33.28

Table 7: Results of 1024 x 1024 image generation.Laion

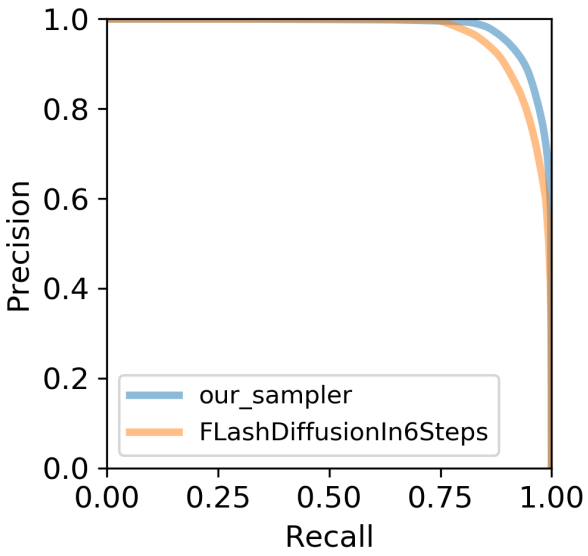

Figure 8: The comparison of PRD between our 6-step inference sampler and the FLash diffusion.

whether the robustness of the diffusion model will be affected when using our algorithm in a scenario that largely depends on the property of the VAE. Moreover, in future research, we will explore whether it is possible to find a better scheduler in a low-step inference process via an automated algorithm, and whether this new time scheduler is available to different latent diffusion models. Last, the hyperparameters of the ACGN model need further exploration.

References

Kendall Atkinson, Weimin Han, and David E Stewart. *Numerical solution of ordinary differential equations*. John Wiley & Sons, 2009.

Leo Breiman. Reflections after refereeing papers for nips. In *The Mathematics of Generalization*, pp. 11–15. CRC Press, 2018.

Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in backslash beta-vae. *arXiv preprint arXiv:1804.03599*, 2, 2018.

Clement Chadebec, Onur Tasar, Eyal Benaroche, and Benjamin Aubin. Flash diffusion: Accelerating any conditional diffusion model for few steps image generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 15686–15695, 2025.

Huanran Chen, Yinpeng Dong, Shitong Shao, Zhongkai Hao, Xiao Yang, Hang Su, and Jun Zhu. Your diffusion model is secretly a certifiably robust classifier. *arXiv preprint arXiv:2402.02316*, 2024.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018.

Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Genie: Higher-order denoising diffusion solvers. *Advances in Neural Information Processing Systems*, 35:30150–30166, 2022.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11): 139–144, 2020.

Gilbert Harman and Sanjeev Kulkarni. *Reliable reasoning: Induction and statistical learning theory*. MIT press, 2012.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020a.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020b.

Insu Jeon, Wonkwang Lee, Myeongjang Pyeon, and Gunhee Kim. Ib-gan: Disentangled representation learning with information bottleneck generative adversarial networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 7926–7934, 2021.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. *arXiv preprint arXiv:2310.02279*, 2023.

Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems*, 33(12): 6999–7019, 2021.

Zilai Li and Rongkai Zhang. Direct distillation: A novel approach for efficient diffusion model inference. *Journal of Imaging*, 11(2):66, 2025.

Shanchuan Lin, Anran Wang, and Xiao Yang. Sdxl-lightning: Progressive adversarial diffusion distillation. *arXiv preprint arXiv:2402.13929*, 2024.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022.

Qiang Liu. Rectified flow: A marginal preserving approach to optimal transport. *arXiv preprint arXiv:2209.14577*, 2022.

Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, et al. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. In *The Twelfth International Conference on Learning Representations*, 2023.

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. *Advances in neural information processing systems*, 35:5775–5787, 2022.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, pp. 1–22, 2025.

Calvin Luo. Understanding diffusion models: A unified perspective. *arXiv preprint arXiv:2208.11970*, 2022.

S Luo, Y Tan, S Patil, D Gu, P Von Platen, A Passos, L Huang, J Li, and H Zhao. Lcm-lora: A universal stable-diffusion acceleration module. arxiv 2023. *arXiv preprint arXiv:2311.05556*.

Jiajun Ma, Shuchen Xue, Tianyang Hu, Wenjia Wang, Zhaoqiang Liu, Zhenguo Li, Zhi-Ming Ma, and Kenji Kawaguchi. The surprising effectiveness of skip-tuning in diffusion sampling. *arXiv preprint arXiv:2402.15170*, 2024.

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*, pp. 10684–10695, 2022.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. *Advances in neural information processing systems*, 31, 2018.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *European Conference on Computer Vision*, pp. 87–103. Springer, 2024.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in neural information processing systems*, 35:25278–25294, 2022.

Chenyang Si, Ziqi Huang, Yuming Jiang, and Ziwei Liu. Freeu: Free lunch in diffusion u-net. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4733–4743, 2024.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020a.

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. *arXiv preprint arXiv:2310.14189*, 2023.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. 2023.

Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Hanwei Wu and Markus Flierl. Vector quantization-based regularization for autoencoders. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 6380–6387, 2020.

Raymond W Yeung. *A first course in information theory*. Springer Science & Business Media, 2012.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and Bill Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in neural information processing systems*, 37:47455–47487, 2024a.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6613–6623, 2024b.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *Advances in Neural Information Processing Systems*, 36: 49842–49869, 2023.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. *arXiv preprint arXiv:2510.11690*, 2025.