SDXS: Real-Time One-Step Latent Diffusion Models with Image Conditions

Yuda Song **Zehao Sun** **Xuanwu Yin**
Xiaomi Inc.
https://idkiro.github.io/sdxs/

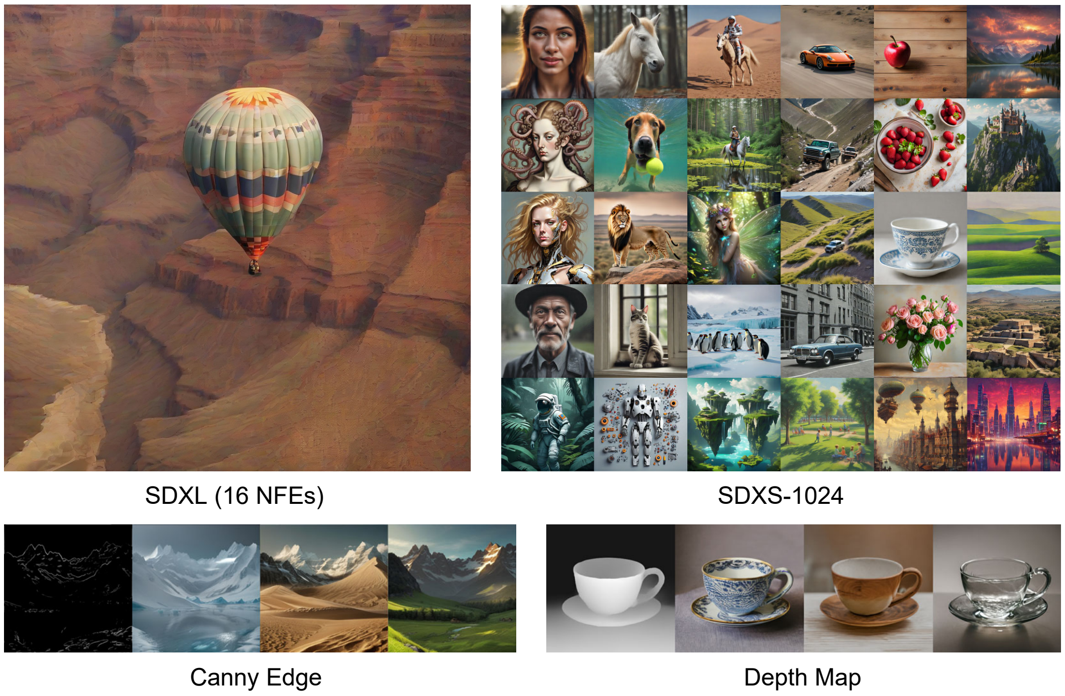

Figure 1: Assuming the image generation time is limited to **1 second**, then SDXL can only use 16 NFEs to produce a slightly blurry image, while SDXS-1024 can generate 30 clear images. Besides, our proposed method can also train ControlNet.

Abstract

Recent advancements in diffusion models have positioned them at the forefront of image generation. Despite their superior performance, diffusion models are not without drawbacks; they are characterized by complex architectures and substantial computational demands, resulting in significant latency due to their iterative sampling process. To mitigate these limitations, we introduce a dual approach involving model miniaturization and a reduction in sampling steps, aimed at significantly decreasing model latency. Our methodology leverages knowledge distillation to streamline the U-Net and image decoder architectures, and introduces an innovative one-step DM training technique that utilizes feature matching and score distillation. We present two models, SDXS-512 and SDXS-1024, achieving inference speeds of approximately **100 FPS** (30$\times$ faster than SD v1.5) and **30 FPS** (60$\times$ faster than SDXL) on a single GPU, respectively. Moreover, our training approach offers promising applications in image-conditioned control, facilitating efficient image-to-image translation.

Technical Report.

1 Introduction

Generative models have recently garnered substantial interest. Leveraging large-scale models and datasets, the text-to-image (T2I) models based on Diffusion Models (DM) [1, 2, 3, 4] have exhibited remarkable capabilities in synthesizing images that are not only realistic but also closely adhere to textual prompts. Based on pretrained diffusion-based text-to-image models, a variety of extended applications have been developed, encompassing areas such as image editing [5, 6], image inpainting [7, 8], image super-resolution [9, 10], video generation [11, 12], and 3D assets synthesis [13, 14]. Despite the rapid and diverse advancements, the deployment of DMs onto low-power devices, such as smartphones, remains a formidable challenge due to the models' large scale and the intrinsic nature of their multi-step sampling processes. Furthermore, even on cloud-based computing platforms equipped with high-performance GPUs, the considerable energy consumption are an aspect that warrants serious consideration.

For common text-to-image DMs, the costs can be approximately calculated as the total latency, which includes the latencies of the text encoder, image decoder, and the denoising model, multiplied by the Number of Function Evaluations (NFEs). Efforts to mitigate the memory, computational, and storage overheads associated with DMs are underway. Similar to other large-scale model deployments, DMs can benefit from pruning [15], distillation [16, 17], and quantization [18, 19, 20] to decrease their costs. Additionally, given the extensive use of Transformer layers within DMs, specific optimizations for Transformer layers [21, 22] are capable of offering considerable efficiency improvements. Importantly, a significant research focus has been placed on minimizing the NFEs, given its profound influence on the overall latency of the model. Techniques employing progressive distillation [23, 24, 25] and consistency distillation [26, 27, 28] have demonstrated the ability to lower NFEs to a range of 4 to 8. Furthermore, innovative strategies Rectified Flow-based [29, 30] and Generative Adversarial Network (GAN)-based [31, 32, 33] methods have achieved reductions of NFEs to as low as 1. These advancements illuminate the promising potential for deploying diffusion models on edge devices. However, the simultaneous exploration of model miniaturization and the transition to one-step operations remains scarcely addressed within the literature. More importantly, although some methods can finetune the model into a few-steps model through LoRA [34] and then directly use the original ControlNet [5] model for image-conditioned generation, this is suboptimal because the distribution of intermediate feature maps in the model updated by LoRA will still differ from that of the original model. Moreover, when the model needs to be finetuned to a one-step model, the low-rank update becomes insufficient for ensuring desired outcomes, and full finetuning would lead to even greater differences in feature map distribution. Therefore, we urgently need a method that can train ControlNet on a one-step model.

Table 1: **Latency Comparison** between SD v2.1 base and our proposed efficient diffusion models on generating 512 × 512 images with batch size = 1.

SD v2.1 base	Text Encoder	U-Net	Image Decoder
#Parameters	0.33B	0.87 B	50 M
Latency (ms)	1.1	16	18
NFEs	2	32	1
Total (ms)	2.2	512	18
SDXS-512	Text Encoder	**U-Net**	**Image Decoder**
#Parameters	0.33B	**0.32 B**	**1.2 M**
Latency (ms)	1.1	**6.3**	**1.7**
NFEs	**1**	**1**	1
Total (ms)	1.1	**6.3**	**1.7**

Table 2: **Latency Comparison** between SDXL and our proposed efficient diffusion models on generating 1024 × 1024 images with batch size = 1.

SDXL	Text Encoder	U-Net	Image Decoder
#Parameters	0.80B	2.56 B	50 M
Latency (ms)	1.8	56	73
NFEs	2	32	1
Total (ms)	3.6	1792	73
SDXS-1024	Text Encoder	**U-Net**	**Image Decoder**
#Parameters	0.80B	**0.74 B**	**1.2 M**
Latency (ms)	1.8	**24**	**6.1**
NFEs	**1**	**1**	1
Total (ms)	1.8	**24**	**6.1**

In this paper, we provide a more comprehensive exploration of the aforementioned challenges. Initially, we focus on reducing the size of the VAE [35] decoder and U-Net [36], both are resource-intensive components in DM sampling. We train an extremely lightweight image decoder to mimic the original VAE decoder's output through a combination of output distillation loss and GAN loss. Following this, we leverage the block removal distillation strategy [16] to efficiently transfer the knowledge from the original U-Net to a more compact version, effectively removing the majority of modules that contribute to latency. To reduce the NFEs, we propose a fast and stable training method. First, we suggest straightening the sampling trajectory and quickly finetuning the multi-step model into a one-step model by replacing the distillation loss function with the proposed feature matching loss. Then, we extend the Diff-Instruct training strategy [37], using the gradient of the proposed feature matching loss to replace the gradient provided by score distillation in the latter half of the timestep. We name our proposed method SDXS to admire SDXL [2], and provide two versions: 512×512 and 1024×1024, developed based on SD v2.1 base and SDXL, respectively. As shown in Tables 1 and 2, SDXS demonstrates efficiency far surpassing that of the base models, even achieving image generation at 100 FPS for 512×512 images and 30 FPS for 1024×1024 images on the GPU. Finally, to ease the application of our optimized model to tasks involving image-conditioned generation, we have adapted the block removal distillation strategy for ControlNet [5]. Then, we extend our proposed training strategy to the training of ControlNet, relying on adding the pretrained ControlNet to the score function.

2 Preliminary

2.1 Diffusion Models

The forward process of DMs [38] transforms samples from the real distribution $p_0(\boldsymbol{x})$ into ones that follow a standard Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$ by progressively adding noise. To counter this, the reverse process aims to invert the forward process by training a denoising model:

$$\mathcal{L}_{DM} = \mathbb{E}_{t \in [0,T], \boldsymbol{x}_0 \sim p_0(\boldsymbol{x}), \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \, ||\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(t, \mathbf{x}_t) - \boldsymbol{\epsilon}||_2^2. \tag{1}$$

This framework facilitates the generation of samples $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, which, through the trained model, are transformed into new samples $\mathbf{x} \sim p_{\text{data}}(\mathbf{x})$ using step-by-step denoising.

Although DMs are derived from the Bayesian framework, they can also be considered as a special Variance Preserving (VP) case under the Score SDE framework [39], which simulates the continuous dynamic changes in the data generation process. At its core, the loss function employed is the score matching (SM) loss, aimed at minimizing the difference between the model's estimated score and the true score of the data:

$$\mathcal{L}_{SM} = \int_{t=0}^{T} w(t) \mathbb{E}_{\boldsymbol{x}_0 \sim p_0(\boldsymbol{x}), \boldsymbol{x}_t | \boldsymbol{x}_0 \sim p_t(\boldsymbol{x}_t|\boldsymbol{x}_0)} \| \boldsymbol{s}_\phi(\boldsymbol{x}_t, t) - \nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t|\boldsymbol{x}_0) \|_2^2 \mathrm{d}t. \tag{2}$$

In practice, while we still utilize DM framework and its discrete training process, the underlying principle for the loss function remains consistent with the continuous interpretation offered by the Score SDE framework.

2.2 Diff-Instruct

Although DMs have demonstrated their capability to produce high-quality images, their efficiency is hindered by the requirement for multiple steps during the sampling process. It is necessary to seek a method to distill its accumulated knowledge from pretrained DMs to instruct the training of models capable of generating samples in just one step. This knowledge distillation from pretrained DMs is known as score distillation and was first proposed in the field of 3D assets synthesis [13]. Diff-Instruct [37] brings score distillation back into image generation, relying on the definition of Integral Kullback-Leibler (IKL) divergence between two distributions p, q:

$$\mathcal{D}_{IKL}^{[0,T]}(q, p) = \int_{t=0}^{T} w(t) \mathbb{E}_{\boldsymbol{x}_t \sim q_t(\boldsymbol{x})} \big[\log \frac{q_t(\boldsymbol{x}_t)}{p_t(\boldsymbol{x}_t)} \big] \mathrm{d}t, \tag{3}$$

where q_t and p_t denote the marginal densities of the diffusion process at time t. The gradient of the IKL in (3) between q_0 and p_0 is

$$\mathrm{Grad}(\theta) = \int_{t=0}^{T} w(t) \mathbb{E}_{\boldsymbol{x}_0 = g_\theta(\boldsymbol{z}), \boldsymbol{x}_t|\boldsymbol{x}_0 \sim p_t(\boldsymbol{x}_t|\boldsymbol{x}_0)} \big[\boldsymbol{s}_\phi(\boldsymbol{x}_t, t) - \boldsymbol{s}_{p_t}(\boldsymbol{x}_t) \big] \frac{\partial \boldsymbol{x}_t}{\partial \theta} \mathrm{d}t. \tag{4}$$

where $\boldsymbol{x}_0 = g_\theta(\boldsymbol{z})$ denotes the sample generated by the one-step generator being trained to accept a randomly initialized latent z, and $\boldsymbol{s}_\phi$ and $\boldsymbol{s}_{p_t}$ denote the score functions of the DM trained online on the generated data and the pretrained DM, respectively. Diff-Instruct uses this gradient directly to update the generator, and when the outputs of the two score functions agree, the marginal distribution of the one-step generator output is consistent with the marginal distribution of the pretrained DM. It's worth noting that the core concept of Diff-Instruct closely aligns with that of VSD [14], and this concept has also been recently expanded to text-to-image generation tasks [40, 41].

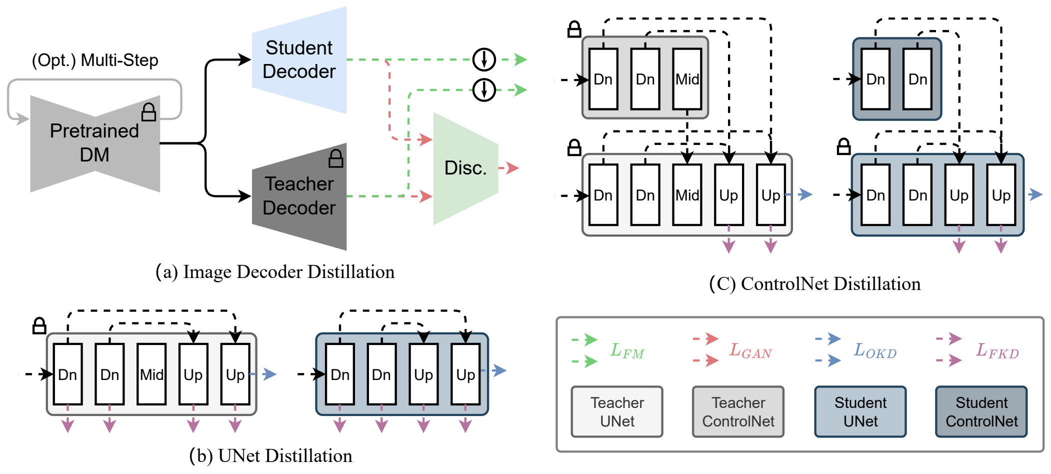

Figure 2: Network architecture distillation, including image decoder, U-Net and ControlNet.

3 Method

3.1 Architecture Optimizations

The image generation process in the Latent Diffusion Model (LDM) framework consists of three key elements: a text encoder, an image decoder, and a denoising model that requires multiple iterations for a clear image. Given the relatively low overhead associated with the text encoder, optimizing its size has not been deemed a priority.

VAE. The LDM framework significantly improves the training efficiency for high-resolution image diffusion models by projecting samples to a more computationally efficient lower-dimensional latent space. This is facilitated through high-ratio image compression using pretrained models, such as the Variational AutoEncoder (VAE) [35, 42] or the Vector Quantised-Variational AutoEncoder (VQ-VAE) [43, 44]. The VAE, in particular, includes an encoder to map images into latent space and a decoder to reconstruct images. Its training is optimized by balancing three losses: reconstruction, Kullback-Leibler (KL) divergence, and GAN loss. However, equally treating all samples during training introduces redundancy. Utilizing a pretrained diffusion model F to sample latent codes $\boldsymbol{z}$ and a pretrained VAE decoder to reconstruct images $\tilde{\boldsymbol{x}}$, we introduce a VAE Distillation (VD) loss for training a tiny image decoder G:

$$\mathcal{L}_{VD} = \|G(\boldsymbol{z})_{\downarrow 8\times} - \tilde{\boldsymbol{x}}_{\downarrow 8\times}\|_1 + \lambda_{GAN}\mathcal{L}_{GAN}(G(\boldsymbol{z}), \tilde{\boldsymbol{x}}, D), \tag{5}$$

where D is the GAN discriminator, λ_{GAN} is used to balance two loss terms, and $\|G(\boldsymbol{z})_{\downarrow\times 8} - \boldsymbol{x}_{\downarrow\times 8}\|_1$ means the L_1 loss is measured on $8\times$ downsampled images . Figure 2 (a) illustrates the training strategy for distilling tiny image decoder. We further advocate for a streamlined CNN architecture devoid of complex components like attention mechanisms and normalization layers, focusing solely on essential residual blocks and upsampling layers.

U-Net. LDMs employ U-Net architectures [36], incorporating both residual and Transformer blocks, as their core denoising model. To leverage the pretrained U-Nets' capabilities while reducing computational demands and parameter numbers, we adopt a knowledge distillation strategy inspired by the block removal training strategy from BK-SDM [16]. This involves selectively removing residual and Transformer blocks from the U-Net, aiming to train a more compact model that can still reproduce the original model's intermediate feature maps and outputs effectively. Figure 2 (b) illustrates the training strategy for distilling tiny U-Net. The knowledge distillation is achieved through output knowledge distillation (OKD) and feature knowledge distillation (FKD) losses:

$$\mathcal{L}_{OKD} = \int_{t=0}^{T} \mathbb{E}_{\boldsymbol{x}_0 \sim p_0(\boldsymbol{x}), \boldsymbol{x}_t|\boldsymbol{x}_0 \sim p_t(\boldsymbol{x}_t|\boldsymbol{x}_0)} \|\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) - \boldsymbol{s}_\phi(\boldsymbol{x}_t, t)\|_2^2 \mathrm{d}t, \tag{6}$$

$$\mathcal{L}_{FKD} = \int_{t=0}^{T} \mathbb{E}_{\boldsymbol{x}_0 \sim p_0(\boldsymbol{x}), \boldsymbol{x}_t|\boldsymbol{x}_0 \sim p_t(\boldsymbol{x}_t|\boldsymbol{x}_0)} \sum_l \|\boldsymbol{f}_\theta^l(\boldsymbol{x}_t, t) - \boldsymbol{f}_\phi^l(\boldsymbol{x}_t, t)\|_2^2 \mathrm{d}t, \tag{7}$$

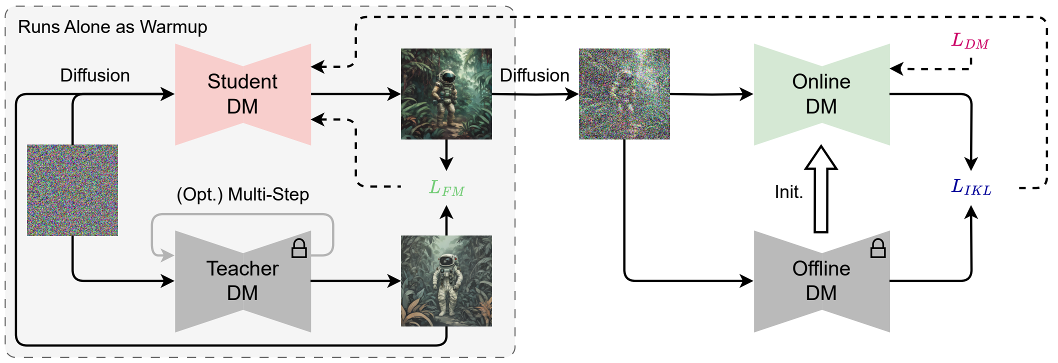

Figure 3: The proposed one-step U-Net training strategy based on feature matching and score distillation. The dashed lines indicate the gradient backpropagation.

with the overarching loss function being a combination of the two:

$$\mathcal{L}_{KD} = \mathcal{L}_{OKD} + \lambda_F \mathcal{L}_{FKD}, \tag{8}$$

where λ_F balances the two loss terms. Different from BK-SDM, we exclude the original denoising loss. Our models are miniaturized based on SD-2.1 base and SDXL-1.0 base. For SD-2.1 base, we remove the middle stage, the last stage of the downsampling stages and the first stage of the upsampling stages, and remove the Transformer blocks of the highest resolution stages. For SDXL-1.0 base, we remove most of the Transformer blocks.

ControlNet. ControlNet [5] boosts diffusion models by embedding spatial guidance in existing text-to-image frameworks, enabling image-to-image tasks like sketch-to-image translation, inpainting, and super-resolution. It copys U-Net's encoder architecture and parameters, adding extra convolutional layers to incorporate spatial controls. Despite inheriting U-Net's parameters and employing zero convolutions for enhanced training stability, ControlNet's training process remains expensive and is significantly affected by the dataset quality. To address these challenges, we propose a distillation approach that distills the ControlNet of the original U-Net into the corresponding ControlNet of the tiny U-Net. As illustrated in Figure 2 (b), instead of directly distilling the output of ControlNet's zero convolutions, we combine ControlNet with U-Net and then distill the intermediate feature maps and output of U-Net, which allows the distilled ControlNet and the tiny U-Net to work better together. Considering that ControlNet does not affect the feature map of U-Net's encoder, feature distillation is only applied to U-Net's decoder.

3.2 One-Step Training

While DMs excel in image generation, their reliance on multiple sampling steps introduces significant inference latency even with advanced samplers [45, 46, 47, 48, 49]. To address this, prior studies have introduced knowledge distillation techniques, such as progressive distillation [23, 24, 25] and consistency distillation [26, 27, 28], aiming to reduce the sampling steps and accelerate inference. However, these approaches typically can only produce clear images with 4∼8 sampling steps, which starkly contrasts the one-step generation process seen in GANs. Exploring the integration of GANs into the DM training regime has shown promise for enhancing image quality [31, 32]. However, GANs come with their own set of challenges, including sensitivity to hyperparameters and training instabilities. It is necessary to seek a more stable training strategy for one-step generation models.

Feature Matching Warmup. A straightforward approach to training a one-step model involves initializing noises $\boldsymbol{\epsilon}$ and employing an Ordinary Differential Equation (ODE) sampler ψ to sample and obtain the generated images $\hat{\boldsymbol{x}}_0$, thereby constructing noise-image pairs. These pairs then serve as inputs and ground truth for the student model during training. This method, however, often results in the production of low-quality images. The underlying issue is the crossings in the sampling trajectories of noise-image pairs generated using the ODE sampler from a pretrained DM, leading to an ill-posed problem. Rectified Flow [29, 30] tackles this challenge by straightening the sampling trajectories. It replaces the training objective and proposes a 'reflow' strategy to refine the pairing,

Algorithm 1: Segmented Score Distillation Algorithm

```
Input: offline DM $\boldsymbol{s}_{p_t}$, online DM $\boldsymbol{s}_\phi$, ODE sampler $\psi$, prior distribution $p_z$, one-step DM $\mathbf{x}_\theta$,
        hyperparameters $\lambda_{FM}$ and $\alpha$.
while not converge do
    update $\phi$ using SGD with gradient
        $\mathrm{Grad}(\phi) = \frac{\partial}{\partial \phi} \int_{t=0}^{T} w(t) \mathbb{E}_{\substack{\boldsymbol{z} \sim p_z, \boldsymbol{x}_0 = \mathbf{x}_\theta(\boldsymbol{z}), \\ \boldsymbol{x}_t | \boldsymbol{x}_0 \sim p_t(\boldsymbol{x}_t | \boldsymbol{x}_0)}} \| \boldsymbol{s}_\phi(\boldsymbol{x}_t, t) - \nabla_{\boldsymbol{x}_t} \log p_t(\boldsymbol{x}_t | \boldsymbol{x}_0) \|_2^2 \mathrm{d}t.$
    update $\theta$ using SGD with the gradient
        $\mathrm{Grad}(\theta) = \int_{t=0}^{\alpha T} w(t) \mathbb{E}_{\boldsymbol{x}_0 = \mathbf{x}_\theta(\boldsymbol{z}), \boldsymbol{x}_t | \boldsymbol{x}_0 \sim p_t(\boldsymbol{x}_t | \boldsymbol{x}_0)} \left[ \boldsymbol{s}_\phi(\boldsymbol{x}_t, t) - \boldsymbol{s}_{p_t}(\boldsymbol{x}_t) \right] \frac{\partial \boldsymbol{x}_t}{\partial \theta} \mathrm{d}t$
            $+ \lambda_{FM} \frac{\partial \mathcal{L}_{FM}(\mathbf{x}_\theta(\boldsymbol{x}_t), \psi(\mathbf{x}_\phi(\boldsymbol{x}_t)))}{\partial \theta}.$
    adjust hyperparameters $\lambda_{FM}$ and $\alpha$.
end
return $\theta, \phi$.
```

thus minimizing trajectory crossings. Instead, we note that the crossing of sampling trajectories can lead to a one noise input corresponding to multiple ground-truth images, causing the trained model generating an image that is a weighted sum of multiple feasible outputs with weights $w(y)$:

$$\min_{\hat{y}} \mathbb{E}[\mathcal{L}(y, \hat{y})] \Rightarrow \hat{y} = \int w(y) \cdot y \, dy. \tag{9}$$

For the most commonly used mean square error (MSE) loss, while consisting of multiple feasible targets, the model tends to output the average of multiple feasible solutions to minimize the overall error, which in turn leads to the blurriness of the generated images. To address this, we explore alternative loss functions that alter the weighting scheme to prioritize sharper images. At the most cases, we can use L1 loss, perceptual loss [50], and LPIPS loss [51] to change the form of weighting. We build upon the method of feature matching [52], which involves computing the loss on the intermediate feature maps generated by a encoder model. Specifically, we draw inspiration from the DISTS loss [53] to apply the Structural Similarity Index (SSIM) on these feature maps for a more refined feature matching loss:

$$\mathcal{L}_{FM} = \sum_l w_l \cdot \mathrm{SSIM}(\boldsymbol{f}_\theta^l(\mathbf{x}_\theta(\boldsymbol{\epsilon})), \boldsymbol{f}_\theta^l(\psi(\mathbf{x}_\phi(\boldsymbol{\epsilon})))). \tag{10}$$

where w_l is the weight of the SSIM loss calculated on the l-th intermediate feature map encoded by the encoder $\boldsymbol{f}_\theta$, $\mathbf{x}_\theta(\boldsymbol{\epsilon})$ is the image generated by tiny U-Net $\mathbf{x}_\theta$, and $\psi(\mathbf{x}_\phi(\boldsymbol{\epsilon}))$ is the image generated by original U-Net $\mathbf{x}_\phi$ with ODE sampler ψ. In practice, we find that using the pretrained CNN backbone [54], ViT backbone [55], and the encoder of DM U-Net all yield favorable results, with a comparison to MSE loss shown in Figure 6. Besides, we also straighten the model's trajectories to narrow the range of feasible outputs using existing finetuning methods like LCM [27, 28] or directly use the publicly available few-step models [32, 33]. We will use $\mathcal{L}_{FM}$ alone to train the one-step model as a warmup, relying only on a small number of training steps.

Segmented Score Distillation. Although feature matching loss can produce almost clear images, it falls short of achieving an true distribution match, so the trained model can only be used as an initialization for formal training. To address this gap, we elaborate on the training strategy utilized in Diff-Instruct, which aims to align the model's output distribution more closely with that of a pretrained model by matching marginal score functions over timestep. However, because it requires adding high levels of noise at $t \to T$ for the target score to be calculable, the score function estimated at this time is inaccurate [56, 57]. We note that the sampling trajectory of diffusion models from coarse to fine, which means that $t \to T$, the score function provides gradients of low-frequency information, while $t \to 0$, it offers gradients of high-frequency information. Therefore, we divide the timestep into two segments: $[0, \alpha T]$ and $(\alpha T, T]$, with the latter being replaced by $\mathcal{L}_{FM}$ because it

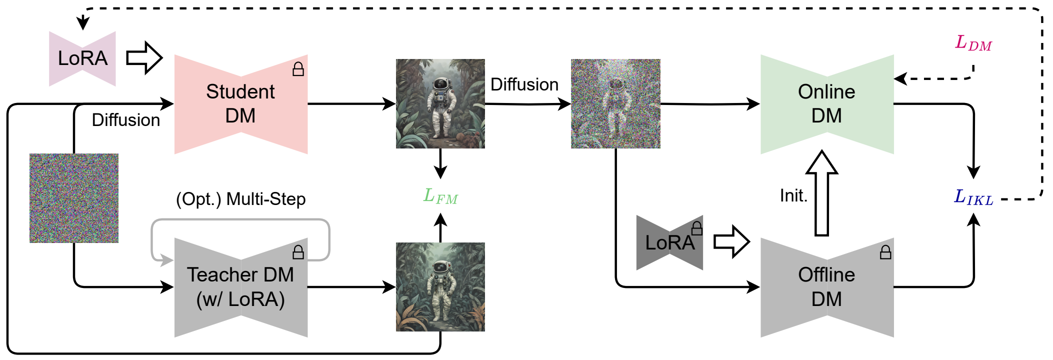

Figure 4: The proposed LoRA training strategy based on feature matching and score distillation. The dashed lines indicate the gradient backpropagation.

can provide sufficient low-frequency gradients. This strategy can be formally represented as:

$$\begin{aligned}\text{Grad}(\theta) = \frac{\partial \mathcal{L}_{IKL}(t, \boldsymbol{x}_t, \boldsymbol{s}_\phi)}{\partial \theta} &= \int_{t=0}^{\alpha T} F(t, \boldsymbol{x}_t)\text{d}t + \int_{t=\alpha T}^{T} F(t, \boldsymbol{x}_t)\text{d}t \\ &\approx \int_{t=0}^{\alpha T} F(t, \boldsymbol{x}_t)\text{d}t + \lambda_{FM} \frac{\partial \mathcal{L}_{FM}(\mathbf{x}_\theta(\boldsymbol{x}_t), \psi(\mathbf{x}_\phi(\boldsymbol{x}_t)))}{\partial \theta},\end{aligned} \tag{11}$$

where

$$F(t, \boldsymbol{x}_t) = w(t)\mathbb{E}_{\boldsymbol{x}_0=\mathbf{x}_\theta(\boldsymbol{z}), \boldsymbol{x}_t|\boldsymbol{x}_0 \sim p_t(\boldsymbol{x}_t|\boldsymbol{x}_0)} \left[\boldsymbol{s}_\phi(\boldsymbol{x}_t, t) - \boldsymbol{s}_{p_t}(\boldsymbol{x}_t)\right] \frac{\partial \boldsymbol{x}_t}{\partial \theta}, \tag{12}$$

λ_{FM} is used to balance gradients of the two segments, and $\alpha \in [0, 1]$. we intentionally set α close to 1 and λ_{FM} at a high value to ensure the model's output distribution smoothly aligns the predicted distribution by the pretrained score function. After achieving significant overlap in the probability densities, we gradually lower both α and λ_{FM}. Figure 3 visually depicts our training strategy, where the offline DM represents the U-Net of a pretrained DM, and the online DM is initialized from the offline DM and finetuned on the generated images through Eq. (1). In practice, the online DM and student DM are trained alternately as in Algorithm 1.

LoRA. Once the one-step DM is trained, it can be finetuned like other DMs to adjust the style of the generated images. We utilize LoRA [34] in conjunction with the proposed Segmented Score Distillation to finetune an one-step DM, as illustrated in Figure 4. Specifically, we insert the pretrained LoRA into the offline DM, and if it is also compatible with the Teacher DM, it is inserted there as well. It is important to note that we do not insert LoRA into the online DM, as it corresponds to the output distribution of the one-step DM. Then, we use the same training procedure as we do for one-step training, but skip the feature matching warmup, since LoRA finetune is much more stable than full finetune. Besides, when the Teacher DM cannot incorporate the pretrained LoRA, we use a reduced λ_{FM}. In this way, the pretrained LoRA can be distilled into the SDXS's LoRA.

ControlNet. Our approach can also be adapted for training ControlNet, enabling the tiny one-step model to incorporate image conditions into its image generation process, as depicted in Figure 5. Compared to the base model for text-to-image generation, the model trained here is the distilled ControlNet that accompanies the tiny U-Net mentioned earlier, and the parameters of the U-Net are fixed during training. Importantly, we need to extract the control images from the images sampled by the teacher model, rather than from the dataset images, to ensure that the noise, target image, and control image form a pairing triplet. Furthermore, the original multi-step U-Net's accompanying pretrained ControlNet is integrated with both the online U-Net and offline U-Net but does not participate in training. Similar to the text encoder, the function is confined to serving as a pretrained feature extractor. In this way, to further reduce $\mathcal{L}$, the trained ControlNet is to learn to utilize the control images extracted from the target images. At the same time, the score distillation encourages the model to match the marginal distributions, enhancing the contextual relevance of the generated images. Notably, we found that replacing a portion of the noise input to U-Net with freshly reinitialized noise can enhance control capabilities.

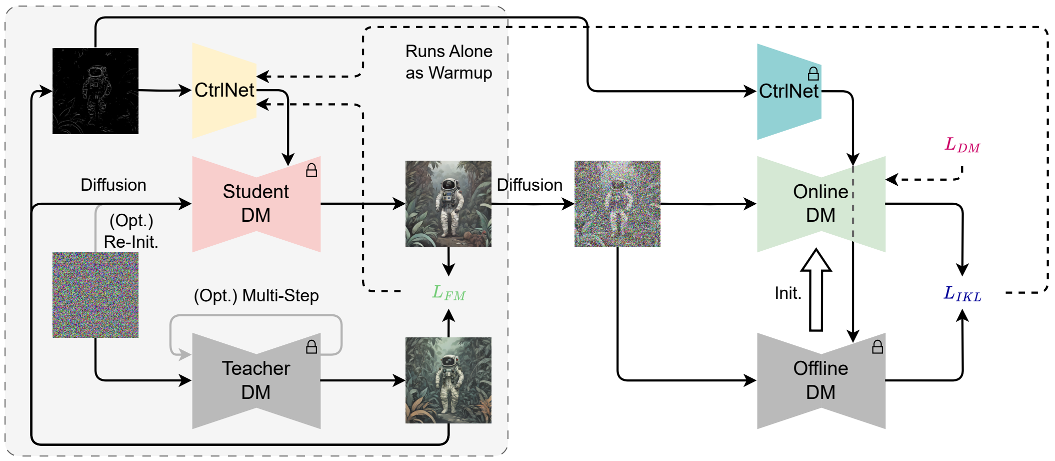

Figure 5: The proposed one-step ControlNet training strategy based on feature matching and score distillation. The dashed lines indicate the gradient backpropagation.

4 Experiment

Implementation Details. Our code is developed based on diffusers library[1]. Because we cannot access the training datasets of SD v2.1 base and SDXL, the entire training process is almost data-free, relying solely on prompts available from publicly accessible dataset [58]. When necessary, we use open-source pretrained models in conjunction with these prompts to generate corresponding images. To train our model, we configure the training mini-batch size to range from 1,024 to 2,048. To accommodate this batch size on the available hardware, we strategically implement gradient accumulation when necessary. It's important to note that we found that the proposed training strategy results in models generating images with less texture. Therefore, after training, we utilize GAN loss in conjunction with extremely low-rank LoRA for a short period of fine-tuning. When GAN loss is needed, we use Projected GAN loss from StyleGAN-T [59], and the basic settings are consistent with ADD [32]. For the training of SDXS-1024, we use Vega [17], a compact version of SDXL, as the initialization for both online DM and offline DM to reduce training overhead.

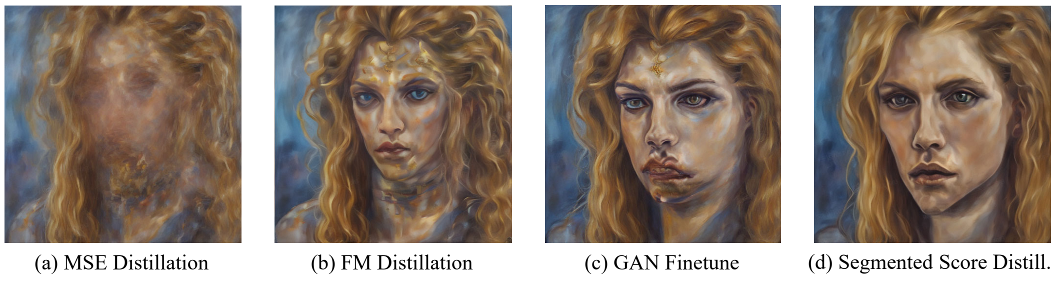

Figure 6: Comparison of images generated by models trained with different distillation strategies. Prompt: Self-portrait oil painting, a beautiful woman with golden hair.

4.1 Text-to-Image Generation

We report the quantitative results, *i.e.*, FID [61] and CLIP scores [62], on MS-COCO 2017 validation set [63] for evaluation. Due to the strong assumption of Gaussian distribution, FID is not a good indicator for measuring image quality [2], as it is significantly affected by the diversity of generated samples, but we still report it as prior works do. Table 3 shows the performance comparison on MS-COCO 2017 5K subset and Figure 7 shows some examples. Despite a noticeable downsizing in both the sizes of the models and the number of sampling steps required, the prompt-following

[1]https://github.com/huggingface/diffusers

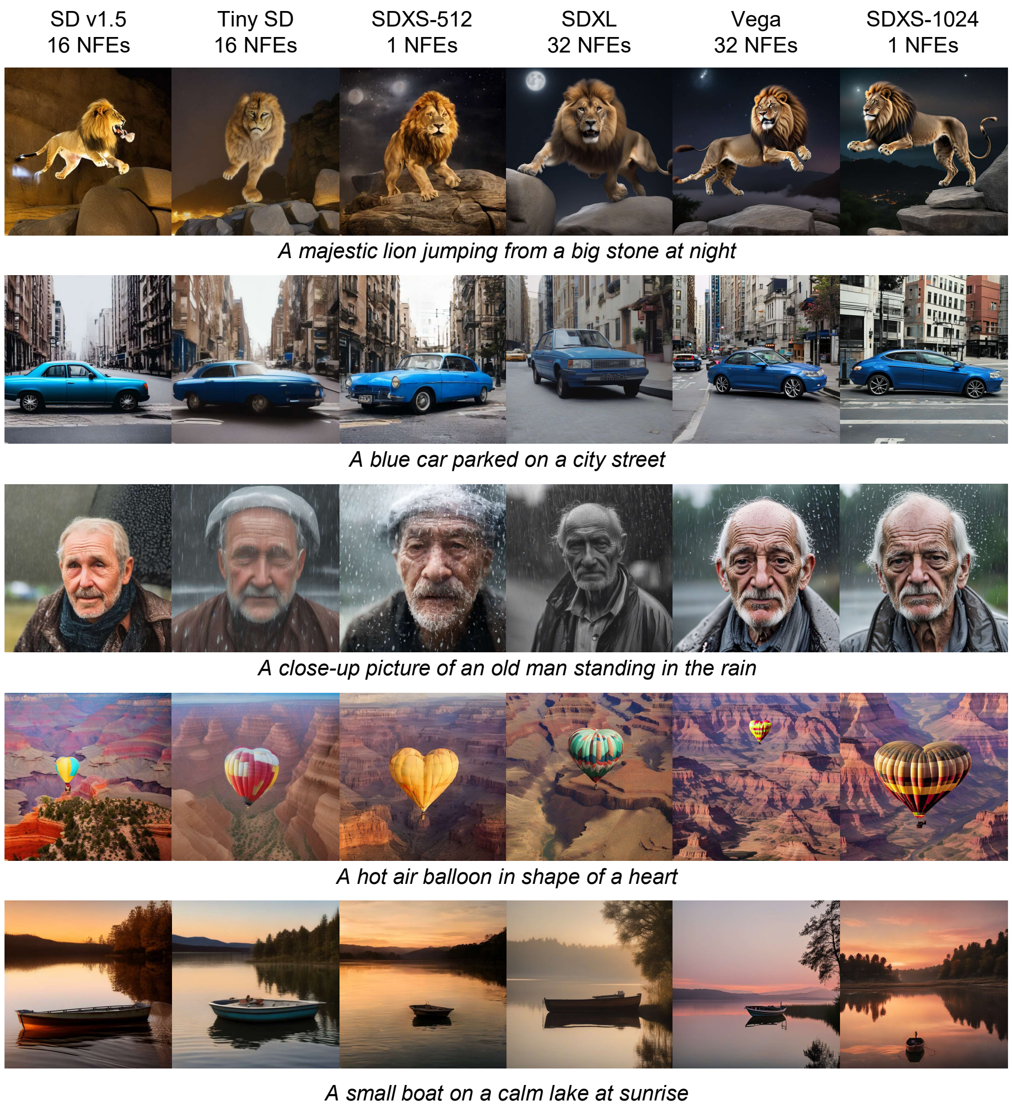
SD v1.5
16 NFEs
Tiny SD
16 NFEs
SDXS-512
1 NFEs
SDXL
32 NFEs
Vega
32 NFEs
SDXS-1024
1 NFEs
A majestic lion jumping from a big stone at night
A blue car parked on a city street
A close-up picture of an old man standing in the rain
A hot air balloon in shape of a heart
A small boat on a calm lake at sunrise

Figure 7: Qualitative comparison between SD v1.5, Tiny SD, SDXL, Vega, and our SDXS.

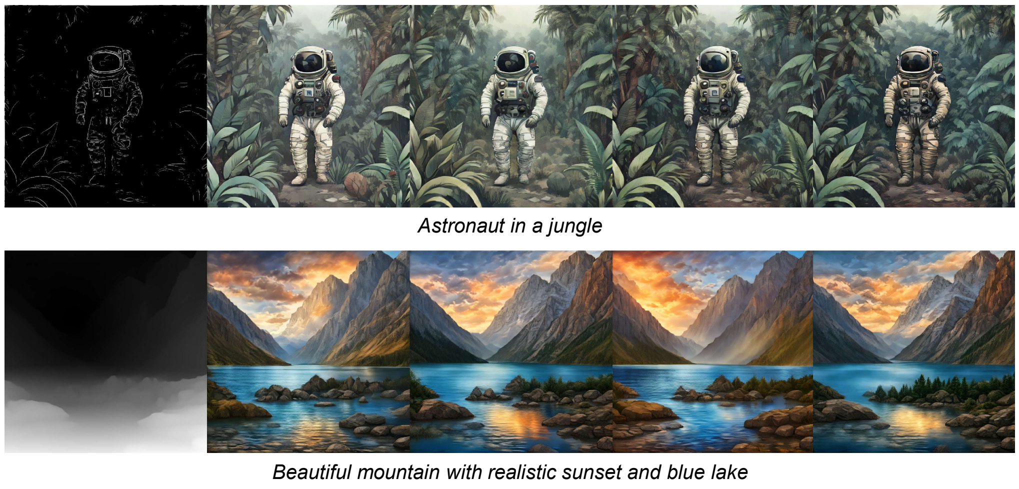

Figure 8: Two examples of ControlNet with SDXS-512.

Table 3: **Performance Comparison** on MS-COCO 2017 5K subset. If CFG [60] is enabled, then the scale will be set to 7.5, and NFEs will be twice the number of sampling steps. Latency is tested under the conditions of enabling float16, with a batch size of 1, and after compiling the model.

Method	Resolution	#Params of U-Net	Sampler	NFEs	Latency (ms) ↓	FID ↓	CLIP Score ↑
SD v1.5 [1]	512×512	860 M	DPM-Solver++	16	276	24.28	31.84
SD v1.5-LCM [28]	512×512	860 M	LCM	4	84	34.74	30.85
SD Turbo [32]	512×512	865 M	-	1	35	26.50	33.07
Tiny SD [16]	512×512	324 M	DPM-Solver++	16	146	31.16	31.06
SDXS-512 (ours)	512×512	319 M	-	1	9	28.21	32.81
Method	Resolution	#Params of U-Net	Sampler	NFEs	Latency (ms) ↓	FID ↓	CLIP Score ↑
SDXL [2]	1024×1024	2.56B	Euler	32	1869	24.60	33.77
SDXL-LCM [28]	1024×1024	2.56B	LCM	4	299	28.87	32.45
SDXL Lightning [33]	1024×1024	2.56B	-	1	131	29.26	32.09
Vega [17]	1024×1024	0.74B	Euler	32	845	29.24	32.99
SDXS-1024 (ours)	1024×1024	0.74B	-	1	32	30.92	32.32

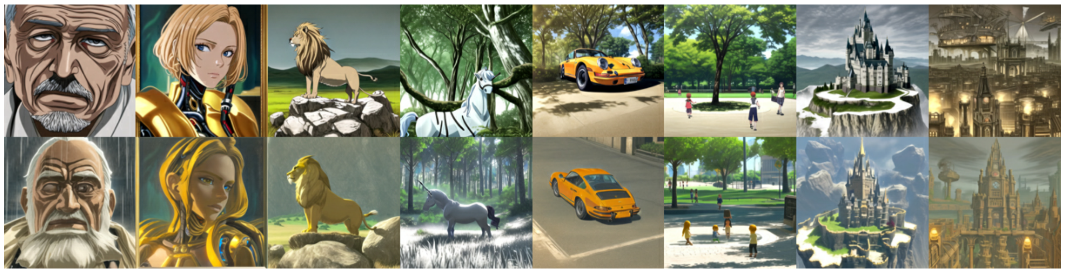
Figure 9: Two samples of two different styles of LoRA with top and bottom images generated using the same prompts.

capability of SDXS-512 remains superior to that of SD v1.5. Moreover, when compared to Tiny SD, another model designed for efficiency, the superiority of SDXS-512 becomes even more pronounced. This observation is consistently validated in the performance of SDXS-1024 as well. Besides, we demonstrate the sample of training LoRA using the proposed method, as shown in Figure 9. It is evident that the style of the generated images by the model can be effectively transferred to match the style of the offline DM into which the style-oriented LoRA has been integrated, while generally maintaining the consistency of the scene's layout.

4.2 Image-to-Image Translation

As we have illustrated earlier, our introduced one-step training approach is versatile enough to be applied to image-conditioned generation. Here, we demonstrate its efficacy in facilitating image-to-image conversions utilizing ControlNet, specifically for transformations involving canny edges and depth maps. Figure 8 illustrates a representative example from each of two distinct tasks, highlighting the capability of the generated images to closely adhere to the guidance provided by control images. However, it also reveals a notable limitation in terms of image diversity. As shown in Figure 1, while the problem can be mitigated by replacing the prompt, it still underscores an area ripe for enhancement in our subsequent research efforts.

5 Conclusion

This paper explores the distillation of large-scale diffusion-based text-to-image generation models into efficient versions that enable real-time inference on GPUs. Initially, we employ knowledge distillation to compress both the U-Net architecture and the image decoder. Subsequently, we introduce a novel training strategy that leverages feature matching and score distillation to reduce the sampling process to one step. This approach allows for the real-time generation of 1024×1024 images on a single GPU, maintaining quality comparable to original models. Moreover, the training methodology we propose can also adapt to tasks involving image-conditioned generation, eschewing the direct adaptation of the pretrained ControlNet. We believe that the deployment of efficient image-conditioned generation on edge devices represents a promising avenue for future research, with plans to explore additional applications such as inpainting and super-resolution.

References

[1] 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, 10

[2] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023. 2, 3, 8, 10

[3] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 2

[4] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022. 2

[5] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023. 2, 3, 5

[6] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2021. 2

[7] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471, 2022. 2

[8] Tao Yu, Runseng Feng, Ruoyu Feng, Jinming Liu, Xin Jin, Wenjun Zeng, and Zhibo Chen. Inpaint anything: Segment anything meets image inpainting. *arXiv preprint arXiv:2304.06790*, 2023. 2

[9] Jianyi Wang, Zongsheng Yue, Shangchen Zhou, Kelvin CK Chan, and Chen Change Loy. Exploiting diffusion prior for real-world image super-resolution. *arXiv preprint arXiv:2305.07015*, 2023. 2

[10] Zongsheng Yue, Jianyi Wang, and Chen Change Loy. Resshift: Efficient diffusion model for image super-resolution by residual shifting. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[11] Yingqing He, Tianyu Yang, Yong Zhang, Ying Shan, and Qifeng Chen. Latent video diffusion models for high-fidelity long video generation. *arXiv preprint arXiv:2211.13221*, 2022. 2

[12] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. 2

[13] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *International Conference on Learning Representations*, 2022. 2, 3

[14] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 3

[15] Gongfan Fang, Xinyin Ma, and Xinchao Wang. Structural pruning for diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[16] Bo-Kyeong Kim, Hyoung-Kyu Song, Thibault Castells, and Shinkook Choi. Bk-sdm: Architecturally compressed stable diffusion for efficient text-to-image generation. In *Workshop on Efficient Systems for Foundation Models@ ICML2023*, 2023. 2, 4, 10

[17] Yatharth Gupta, Vishnu V Jaddipal, Harish Prabhala, Sayak Paul, and Patrick Von Platen. Progressive knowledge distillation of stable diffusion xl using layer level loss. *arXiv preprint arXiv:2401.02677*, 2024. 2, 8, 10

[18] Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. Post-training quantization on diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1972–1981, 2023. 2

[19] Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-diffusion: Quantizing diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17535–17545, 2023. 2

[20] Yefei He, Luping Liu, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. Ptqd: Accurate post-training quantization for diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[21] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022. 2

[22] Benjamin Lefaudeux, Francisco Massa, Diana Liskovich, Wenhan Xiong, Vittorio Caggiano, Sean Naren, Min Xu, Jieru Hu, Marta Tintore, Susan Zhang, Patrick Labatut, Daniel Haziza, Luca Wehrstedt, Jeremy Reizenstein, and Grigory Sizov. xformers: A modular and hackable transformer modelling library. `https://github.com/facebookresearch/xformers`, 2022. 2

[23] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2021. 2, 5

[24] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14297–14306, 2023. 2, 5

[25] Yanyu Li, Huan Wang, Qing Jin, Ju Hu, Pavlo Chemerys, Yun Fu, Yanzhi Wang, Sergey Tulyakov, and Jian Ren. Snapfusion: Text-to-image diffusion model on mobile devices within two seconds. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 5

[26] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023. 2, 5

[27] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023. 2, 5, 6

[28] Simian Luo, Yiqin Tan, Suraj Patil, Daniel Gu, Patrick von Platen, Apolinário Passos, Longbo Huang, Jian Li, and Hang Zhao. Lcm-lora: A universal stable-diffusion acceleration module. *arXiv preprint arXiv:2311.05556*, 2023. 2, 5, 6, 10

[29] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International conference on learning representations (ICLR)*, 2023. 2, 5

[30] 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 *International Conference on Learning Representations*, 2024. 2, 5

[31] Yanwu Xu, Yang Zhao, Zhisheng Xiao, and Tingbo Hou. Ufogen: You forward once large scale text-to-image generation via diffusion gans. *arXiv preprint arXiv:2311.09257*, 2023. 2, 5

[32] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. *arXiv preprint arXiv:2311.17042*, 2023. 2, 5, 6, 8, 10

[33] Shanchuan Lin, Anran Wang, and Xiao Yang. Sdxl-lightning: Progressive adversarial diffusion distillation. *arXiv preprint arXiv:2402.13929*, 2024. 2, 6, 10

[34] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2021. 2, 7

[35] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2, 4

[36] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICAI*, 2015. 2, 4

[37] Weijian Luo, Tianyang Hu, Shifeng Zhang, Jiacheng Sun, Zhenguo Li, and Zhihua Zhang. Diff-instruct: A universal approach for transferring knowledge from pre-trained diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 3

[38] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 3

[39] 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*, 2020. 3

[40] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. *arXiv preprint arXiv:2311.18828*, 2023. 3

[41] Thuan Hoang Nguyen and Anh Tran. Swiftbrush: One-step text-to-image diffusion model with variational score distillation. *arXiv preprint arXiv:2312.05239*, 2023. 3

[42] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pages 1278–1286. PMLR, 2014. 4

[43] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in Neural Information Processing Systems*, 30, 2017. 4

[44] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883, 2021. 4

[45] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2020. 5

[46] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *International Conference on Learning Representations*, 2022. 5

[47] 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. 5

[48] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022. 5

[49] 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, 2024. 5

[50] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016. 6

[51] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018. 6

[52] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in Neural Information Processing Systems*, 29, 2016. 6

[53] Keyan Ding, Kede Ma, Shiqi Wang, and Eero P Simoncelli. Image quality assessment: Unifying structure and texture similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(5):2567–2581, 2020. 6

[54] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6

[55] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2023. 6

[56] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019. 6

[57] Thiemo Alldieck, Nikos Kolotouros, and Cristian Sminchisescu. Score distillation sampling with learned manifold corrective. *arXiv preprint arXiv:2401.05293*, 2024. 6

[58] 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. *arXiv preprint arXiv:2210.08402*, 2022. 8

[59] Axel Sauer, Tero Karras, Samuli Laine, Andreas Geiger, and Timo Aila. Stylegan-t: Unlocking the power of gans for fast large-scale text-to-image synthesis. In *International Conference on Machine Learning*, pages 30105–30118. PMLR, 2023. 8

[60] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 10

[61] 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. 8

[62] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 8

[63] 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*, pages 740–755. Springer, 2014. 8