Simpler Diffusion: 1.5 FID on ImageNet512 with pixel-space diffusion

Emiel Hoogeboom
Google Deepmind
emielh@google.com

Thomas Mensink
Google Deepmind
mensink@google.com

Jonathan Heek
Google Deepmind
jheek@google.com

Kay Lamerigts
Google Research
kaylamerigts@google.com

Ruiqi Gao
Google Deepmind
ruiqig@google.com

Tim Salimans
Google Deepmind
salimans@google.com

Abstract

Latent diffusion models have become the popular choice for scaling up diffusion models for high resolution image synthesis. Compared to pixel-space models that are trained end-to-end, latent models are perceived to be more efficient and to produce higher image quality at high resolution. Here we challenge these notions, and show that pixel-space models can be very competitive to latent models both in quality and efficiency, achieving 1.5 FID on ImageNet512 and new SOTA results on ImageNet128, ImageNet256 and Kinetics600.

We present a simple recipe for scaling end-to-end pixel-space diffusion models to high resolutions. 1: Use the sigmoid loss-weighting [25] with our prescribed hyper-parameters. 2: Use our simplified memory-efficient architecture with fewer skip-connections. 3: Scale the model to favor processing the image at a high resolution with fewer parameters, rather than using more parameters at a lower resolution. Combining these with guidance intervals, we obtain a family of pixel-space diffusion models we call Simpler Diffusion *(SiD2).*

1. Introduction

Diffusion models have been tremendously successful in domains such as image, video and audio generation. For high resolution data, *latent diffusion models* [32] have been particularly successful: Through encoding the data in a lower dimensional space, these models sidestep the difficulties faced by more traditional pixel-space diffusion models in scaling up to high resolution. This makes latent diffusion models also scale better in training compute than other approaches, as shown by [20].

However, latent models have their own challenges: It can be difficult to train the autoencoders that produce the latents, which in practice requires careful tuning of regularization hyperparameters [32]. Even when trained well, the decoder of a latent diffusion model can produce artifacts or otherwise imperfect reconstructions, especially when it is applied to latents that are out of distribution compared to the training set. A single-stage pixel-space diffusion model does not have these issues, and so we ask the question: Can pixel-space diffusion models scale as well as latent models with the right architecture and hyperparameters?

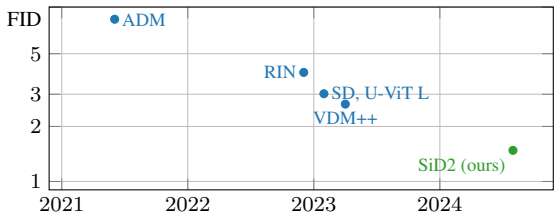


Figure 1. End-to-end pixel diffusion performance on ImageNet512.

Over the years, end-to-end pixel-space diffusion has improved considerably (Figure 1). The main problem was that the standard schedule and loss-weighting [11] work well for small resolution images, like 32^2 or 64^2, but not at higher resolution [3, 14]. It was further shown [25] that it can be helpful to disentangle loss weighting and time sampling for improved control over both, leading to better performance at higher resolution. However, in previous works pixels-space models still did not match the best latent models on standard benchmarks like ImageNet512.

The main contribution of this paper is to improve pixel-space diffusion on high resolution images while keeping as close to the original formulation of diffusion models as possible. Specifically, we analyze and tune the sigmoid loss weighting from [25], and show why it should be preferred over their favored EDM monotonic. We further propose architectural improvements that lead to even better perfor-

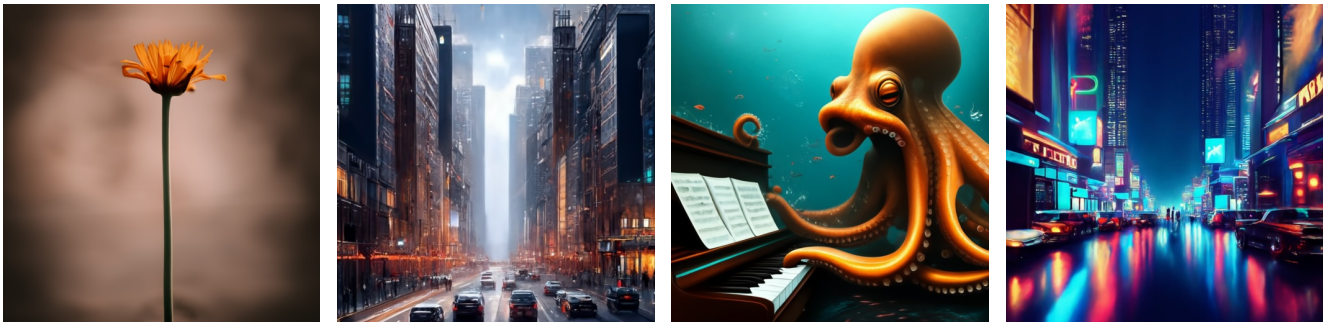


Figure 2. Some text-to-image samples of 512×512 that have been generated by SiD2.

mance, namely flop heavy scaling and removing block skip-connections (Residual U-ViTs). Most notably, we improve the state-of-the-art of pixel-space diffusion from 2.65 to 1.48 on ImageNet512 (see also Figure 1). Further, we also achieve overall state-of-the-art on ImageNet128, ImageNet256 and Kinetics600 without heavy tuning.

1.1. Background

A diffusion process gradually destroys data over a conceptual time t, typically with Gaussian noise as follows:

$$\boldsymbol{z}_t = \alpha_t \boldsymbol{x} + \sigma_t \boldsymbol{\epsilon}, \text{ where } \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}), \tag{1}$$

which defines a distribution $q(\boldsymbol{z}_t|\boldsymbol{x})$ such that $\boldsymbol{z}_0$ is close to clean data and $\boldsymbol{z}_1$ is close to Gaussian noise. In this paper we assume a variance-preserving [41] schedule, which is defined as $\alpha_t = \sqrt{\sigma(\lambda_t)}$ and $\sigma_t = \sqrt{\sigma(-\lambda_t)}$, with $\lambda_t = \log\left(\alpha_t^2/\sigma_t^2\right)$ being the log signal-to-noise ratio, which monotonically decreases with time as $t \to 1$, and $\sigma(\cdot)$ being the sigmoid function. It can be shown that other types of schedules such as the variance-exploding schedule or the flow matching schedule [28] can always be converted to a variance-preserving schedule with a linear scaling of $\boldsymbol{z}_t$ [25]. The key defining property of a noise schedule is the log signal-to-noise ratio λ_t. A neural network is used to predict $\hat{\boldsymbol{x}} = \hat{\boldsymbol{x}}_\theta(\boldsymbol{z}_t, t)$ that effectively estimates $\mathbb{E}[\boldsymbol{x}|\boldsymbol{z}_t]$, which can be done by minimizing a weighted mean squared error loss:

$$L(\boldsymbol{x}) = \mathbb{E}_{t\sim\mathcal{U}(0,1)} -\frac{\mathrm{d}\lambda_t}{\mathrm{d}t} w(\lambda_t) \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2, \tag{2}$$

where $w(\lambda_t)$ is the weighting function. $-\mathrm{d}\lambda_t/\mathrm{d}t$ cancels out the effect of the noise schedule λ_t on the loss. In other words, $L(\mathbf{x})$ can be rewritten as $\int -w(\lambda)\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2 d\lambda$ that is invariant to the choice of λ_t. However, λ_t still impacts the *sampling efficiency* of the Monte Carlo estimator of the expected loss, and thus affects the training dynamics of diffusion models. This decouples *noise schedule* λ_t and *weighting function* $w(\lambda)$ as a function of λ [25].

The above squared error loss can be derived from the variational interpretation of diffusion models [26]. Given two time steps $s < t$, we define the denoising distribution $p(\boldsymbol{z}_s|\boldsymbol{z}_t)$ in the same formulation as the true posterior $q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{x})$, but replace the ground truth $\boldsymbol{x}$ with the learned approximation $\hat{\boldsymbol{x}} \approx \mathbb{E}[\boldsymbol{x}|\boldsymbol{z}_t]$:

$$p(\boldsymbol{z}_s|\boldsymbol{z}_t) = q(\boldsymbol{z}_s|\boldsymbol{z}_t, \boldsymbol{x} = \hat{\boldsymbol{x}}) = \mathcal{N}(\boldsymbol{z}_s|\mu_{t\to s}, \sigma_{t\to s}), \tag{3}$$

where $\sigma^2_{t\to s} = \left(1/\sigma_s^2 + \alpha^2_{t|s}/\sigma^2_{t|s}\right)^{-1}$ and $\mu_{t\to s} = \sigma^2_{t\to s}\left(\alpha_{t|s}/\sigma^2_{t|s}\boldsymbol{z}_t + \alpha_s/\sigma_s^2\boldsymbol{x}\right)$ with $\alpha_{t|s} = \alpha_t/\alpha_s$ and $\sigma^2_{t|s} = \sigma_t^2 - \alpha_{t|s}\sigma_s^2$. Note that during sampling, the standard deviation for $p(\boldsymbol{z}_s|\boldsymbol{z}_t)$ is chosen to be a log-linear interpolation between $\sigma_{t|s}$ and $\sigma_{t\to s}$ of the form $\sigma^\gamma_{t\to s}\sigma^{1-\gamma}_{t|s}$ where $\gamma = 0.3$ was set empirically, which is important as FID is sensitive to over-smooth or grainy images.

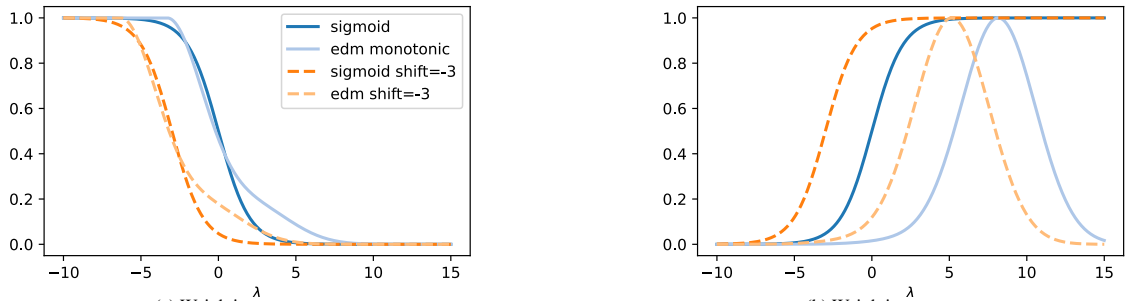
2. Simpler Diffusion

2.1. Revisiting the sigmoid loss

Recall that one of the proposed losses in VDM++ [25] is the sigmoid loss. They expressed the loss with respect to $\boldsymbol{\epsilon}$-mse: $L(\boldsymbol{x}) = \mathbb{E}_{t\sim\mathcal{U}(0,1)} -\frac{\mathrm{d}\lambda_t}{\mathrm{d}t}\tilde{w}(\lambda_t)\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}\|^2$, which is connected to $\boldsymbol{x}$-mse through $\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}\|^2 = \exp(\lambda_t)\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2$ and $\tilde{w}(\lambda_t) = \exp(\lambda_t)w(\lambda_t)$. In VDM++, viewing the loss in $\boldsymbol{\epsilon}$-mse enables a natural connection between the weighted loss and the ELBO and showed that losses should be monotonically decreasing on $\boldsymbol{\epsilon}$-mse. Hence, they propose the sigmoid loss: $\tilde{w}(\lambda_t) = \sigma(b - \lambda_t)$, where b denotes the bias or shift and σ is the sigmoid function. However, they ultimately opted for another loss-weighting named EDM-monotonic for high resolution image generation, see Figure 3a.

Enforcing weighting monotonicity in $\boldsymbol{x}$ Revisiting the sigmoid loss, we derive its $\boldsymbol{x}$-mse form, which turns out to conveniently be defined by a mirrored sigmoid (albeit multiplied by a constant factor):

$$\begin{aligned}\sigma(b - \lambda_t)\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}\|^2 &= \sigma(b - \lambda_t)\exp(\lambda_t)\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2 \\ &= \exp(b)\sigma(\lambda_t - b)\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2.\end{aligned} \tag{4}$$



(a) Weighting over ϵ-mse (b) Weighting over x-mse

Figure 3. Weighting of different losses in both ϵ and x space. Note that the weightings are scaled to have a maximum of one.

This reveals that the sigmoid loss is not only monotonically decreasing in ϵ-mse, it is also monotonically increasing in x-mse. We claim that a reasonable weighting function $w(\lambda_t)$ should be monotonically increasing over λ_t. Observe that with less noise in z_t, there is more information available and it should be strictly easier to predict the original data, meaning that the x-mse generally decreases for higher λ_t. This phenomenon is also observed in practice. If the loss *weighting* in x-mse also decreases, the model would be able to make worse predictions while obtaining *less noisy (better)* conditioning information and still achieve a lower loss. Simply put: Why put less weight on an easier problem?

Studying both the form of the sigmoid and EDM-monotonic weighting functions in Figure 3, we observe that both weighting functions are monotonically decreasing in ϵ-mse, but only the sigmoid weighting is monotonically increasing in x-mse. Therefore, we expect the sigmoid weighting to be a better candidate to scale to high resolution generation than the EDM-monotonic version.

Loss shifting Recall that for a resolution that is two times higher in both height and width, an average (low pass) filter would see $\mathrm{Var}[(x_{11} + x_{12} + x_{21} + x_{22})/4] = 1/4$ of the noise, while signal strength stays the same [14, 25]. Hence, its effective logsnr is expected to increase by about $2 \log 2 \approx 1.386 \approx 1.5$ when viewed through a low-pass filter. Suppose that the optimal setting for images of 128×128 is a sigmoid weighting with $b \approx 0$, then one would expect the optimal setting for images of 512×512 to be roughly $b = -3$.[1]. Empirically we find that slightly more aggressive shifts perform even better.

2.2. Flop heavy scaling

When one deals with a limited data budget, it can be difficult to balance scaling of models and regularization. Our small model version at 512×512 resolution starts out with 4×4 patching, and subsequently runs a U-ViT [14] on the 128×48 input (assuming the raw input is of 3 channels). To make the model more expressive, one of the most common options is to double the number of channels for every layer, leading to $2\times$ model size. However, our experiments show that this results in diminishing returns. We hypothesize that it is because the model suffers from over-fitting when trained with limited data budget such as ImageNet. Indeed, when we train using distribution augmentation [18], the best performance improves from 1.9 to 1.6. This however suggests another way to scale the model: Keep the architecture and parameter count almost the same and instead run on 2×2 patched input (so $256 \times 256 \times 12$ for 512 inputs). Similar to $2\times$ model size, reducing the patching size also increases FLOPS by roughly a factor of 4. However, we found this model performs even better to about 1.5 without the need of distribution augmentation. We call this model *flop heavy*, because it has a much higher flop over parameter ratio.

Flop heavy scaling has another interesting property. Suppose we have a small model trained to fit input of size 256×256 with 2×2 patching. Using the flop heavy scaling at 512×512 resolution enables us to directly finetune from the small model trained at 256×256 with exactly the same amount of parameters, whereas its counterpart version at 512×512 resolution and with 4×4 patching needs to introduce additional parameters to handle the more input channels. This method of finetuning to higher resolutions has already been popularized in the context of text-to-image generation [32]. Here we find that flop heavy scaling is a more favorable way to scale on smaller datasets such as ImageNet. Of course, in principle one could also train a model with more channels and 4×4 patching on a lower resolution, but this is not done in practice to train 'small' models.

2.3. Residual U-ViTs: Removing blockwise skip-connections

U-Nets require that sub-sampled levels (for example via average pooling) are bypassed by skip-connections. However, this makes the architecture very different from large-scale language models such as transformers, where these skip-connections are not used and residual connections suffice.

[1] This holds *under the condition that* the most important information is already contained by the 128×128 image, and this may for example not hold for images without global structure, e.g. that only contain textures.

In this paper we use the U-ViT model of [14], which is a variant of U-Nets with self-attention layers (with convolutional layers) being replaced by transformer blocks. Hence, that part of the architecture is closer to standard transformers.

In this paper, we further simplify model definition with Residual U-ViTs, see Figure 4. The blockwise skip-connections are removed, and in their place only a single skip-connection is used per downsampling operation. To be precise, let f_d, f_u denote a down and up level of a U-ViT and let f_m denote the middle stage. This two-stage Residual U-ViT can now be described by:

$$f(x) = f_u\Big(\mathrm{U}\Big(f_m(\mathrm{D}(h)) - \mathrm{D}(h)\Big) + h\Big) \text{ where } h = f_d(x) \tag{5}$$

where D is typically an average pooling and linear and U is typically a linear and nearest neighbour upsampling. Defining this network to preserve an identity function at initialization is now much easier: As long as f_d, f_u and f_m are initialized to be identity, the entire f is also identity at initialization. The generalization to any-depth U-ViTs is also straightforward:we can define another two levels of U-ViTs in place of f_m recursively, where the nested U-ViTs can also be guaranteed to be initialized as an identity function.

For small networks, it could be helpful to keep the blockwise skip-connections, and expect to incur a slight performance degradation from removing the blockwise skip-connections. Small networks have limited capacity, so limiting skip-connections may create bottleneck issues. Experimentally, this is confirmed for the small architectures, where we observe 0.2 FID performance degradation on ImageNet512 and 0.06 FID performance degradation on ImageNet256. Because of the simplified definition, we choose to accept this performance penalty for small models. Moreover, we find that larger models perform on par or even benefit from removing the blockwise skip-connections.

The final advantage is memory consumption at evaluation. For example, a feature map at 256^2 resolution with 128 channels requires $256^2 \cdot 128 \cdot 2/1024^2 = 16$MB. With 3 blocks per level, this requires holding an extra $3 \cdot 16 = 48$MB per example for the skip-connections, whereas the Residual U-ViT only requires holding h and $D(h)$ (the latter could be cheaply recomputed), so $16 + 8 = 24$MB per example while computing the lower levels. For a hypothetical super-resolution model at 4096^2 resolution, this leads to a difference between $3 \cdot 4 = 12$GB and only $4 + 2 = 6$GB. During training this advantage is usually negligible, because feature maps also need to be retained for the backward pass. A final advantage of Residual U-ViTs is that it is more straightforward to define asymmetric U-ViTs, i.e., with a different number of blocks in the downsampling (encoding) stage from the one in the upsampling (decoding) stage. In certain cases, moving blocks from the downsampling stage to the upsampling stage can increase performance.

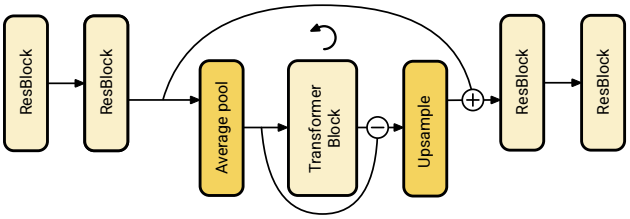


Figure 4. Residual U-ViT with levelwise skip-connections (instead of blockwise). When the network blocks are initialized to be identity, the entire network is also an identity function at initialization.

2.4. Distillation

Sampling from a pixel-space diffusion model naively often requires more function evaluations than sampling from a latent-space diffusion model. For example, the popular Stable Diffusion model [32] is usually sampled using 20-50 sampling steps, while pixel-space models are sampled with 100+ steps [14]. To reduce the number of sampling steps, we could use either more advanced solvers or distillation. Recently *distillation* methods show that pixel-space models can be adapted into few-step samplers that maintain high sample quality [23, 29, 36, 44, 45, 47]. In this work, we use *moment matching distillation* [37] which is currently one of the best distillation methods available in the few step regime. We distill our pixel-space models for use with 8-16 sampling steps, which empirically we find maintains or even improves the sample quality of our models.

3. Related Work

The first end-to-end diffusion work that directly generated high resolution images was [4]. Given the poor performance of end-to-end models, two popular alternatives were introduced: cascaded diffusion [12] and latent diffusion [32].

Cascaded diffusion approaches have been shown to scale to large datasets and models to generate high quality and high resolution imagery, albeit in multiple stages [12, 40, 42, 52], or with fine-tuning [43]. Although the focus of this paper has been single stage generation up to a resolution of 1024^2, the loss weightings may also be used for cascaded models.

Significant improvements were made to end-to-end diffusion by shifting noise schedules [3, 14] and subsequently loss weightings [25]. Other approaches such as multi-resolution losses (and possibly architecture adaptions) [6, 7, 14] added marginal gains. However, the current best known latent model EDM2 [20] outperforms these prior end-to-end approaches by a large margin. In contrast, our models are competitive with latent models, outperforming DiT-XL/2 [31] and only performing slightly worse than EDM2.

Skip connections The U-Net architecture is a fully convolutional network with long-range skip-connections introduced for (medical) image segmentation [33, 39]. The main

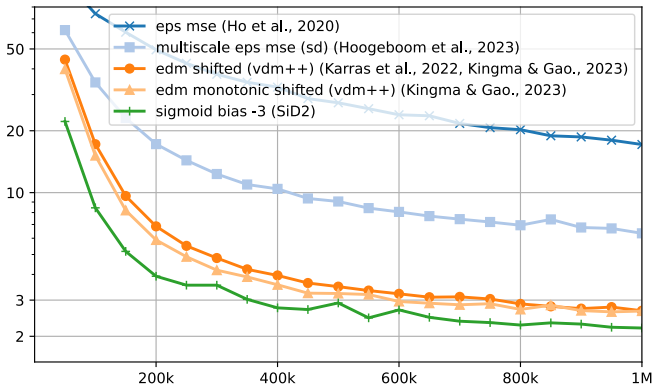


Figure 5. Effect of weighting functions in the loss on the FID score of ImageNet512 over the training iterations. The proposed sigmoid weighting with $b = -3$ shows best performance consistent across all training iterations.

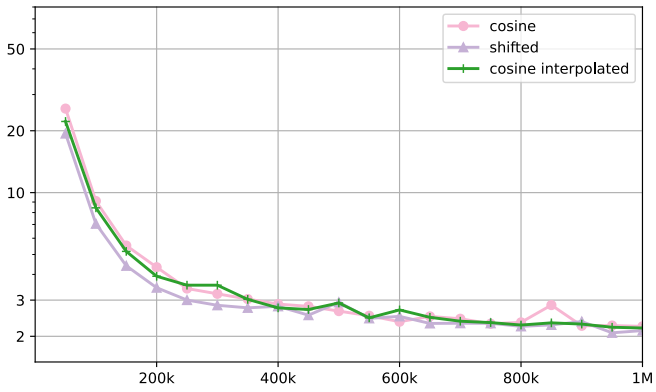


Figure 6. Effect of *training* noise schedules on the FID score of ImageNet512 over the training iterations. Fortunately, with our proposed sigmoid weighting ($b = -3$), the exact choice of the noise schedules has little effect on the *training* process.

idea of U-Net is to combine a contracting path (which encodes the image semantics) with long skip-connections at the same resolution to enable precise localization.

With the introduction of ViTs [5] as an alternative for convolutional layers, the U-Net has been converted to U-ViTs using a similar network but replacing (some) ResNet blocks with ViTs, either with spatial down-sampling [14] or without [2]. Long-range skip-connections play an pivotal role in these networks, for example for fine-tuning controllable diffusion models [17] and for the stability and training speed [15, 30]. In these latter works, it is shown that multiplying the skip-connections with a (constant) scalar (< 1) improves training stability and speed, because of the propagation of noise throughout the network. In our work, we reduce the number of skip-connections significantly and initialise the network such that it resembles an identify function.

Flop heavy fine-tuning Fine-tuning on a higher resolution at large scale was effectively done in Stable Diffusion [32] to limit the training cost, which is a form of flop heavy scaling over different resolutions. In this paper we show that flop heavy scaling is also preferred from a regularization perspective, which can be done either by finetuning from a lower resolution, or by lowering the patching size.

4. Experiments

4.1. Sigmoid weighting versus others

Loss weighting In this experiment, we want to understand the effect of loss weighting on the final image quality of a model. For the noise schedule, we use an interpolated cosine schedule because it has relatively broad coverage over logsnr values. We compare the weightings corresponding to the original ϵ-mse [11] and the multiscale ϵ-mse [14], the EDM shifted weightings [19], including the original and monotonic versions [19, 26], and the proposed sigmoid weighting

Table 1. Sigmoid weighting performance (*left*) with the *flop heavy* architecture. Larger resolutions require more negative bias. Power loss performance (*right*) on ImageNet512 with the *small model*. Increasing downsampling has a similar effect as decreasing bias, hence we prefer the more simple sigmoid weighting.

(a) Sigmoid weighting

	resolution			
bias	128	256	512	1024
-5	-	-	-	1.82
-4	-	-	1.63	**1.75**
-3	-	1.47	**1.48**	2.32
-2	1.49	1.42	1.55	2.17
-1	1.35	**1.38**	1.75	-
0	1.30	1.46	1.94	-
1	**1.26**	1.56	-	-

(b) Power loss weighting

	downsample levels			
bias	0	1	2	3
-4	**2.19**	2.58	3.62	5.11
-3	**2.19**	2.32	2.85	4.02
-2	2.29	**2.15**	2.34	3.13
-1	2.64	2.32	2.25	2.54
0	3.22	2.74	**2.07**	2.22
1	-	2.69	2.29	**2.16**
2	-	3.26	2.53	2.19

with shift $b = -3$. As can be seen in Figure 5, the sigmoid weighting outperforms all other weightings, with EDM monotonic shifted being the closest alternative. Appendix C contains additional experiments, where we also evaluate the sigmoid weighting against the closest competitor without guidance. Here again sigmoid weighting is consistently better. We also evaluate a time-shifted variant, where the bias depends on the progress in training.

Training efficiency of noise schedules Since the training noise schedule can be decoupled from the loss weighting and it only affects the training efficiency but not the final performance, one can measure the convergence speed over the training process caused by leveraging particular noise schedules. In this experiment we test the cosine schedule [11], the shifted cosine and the interpolated cosine schedules [14] (Figure 6). For fair comparison, we fix the sampling noise schedule as the cosine interpolated schedule. Interestingly, when the sigmoid weighting is used, the choice of noise schedule does not affect the training efficiency significantly. Arguably, the shifted schedule is a bit more sample effi-

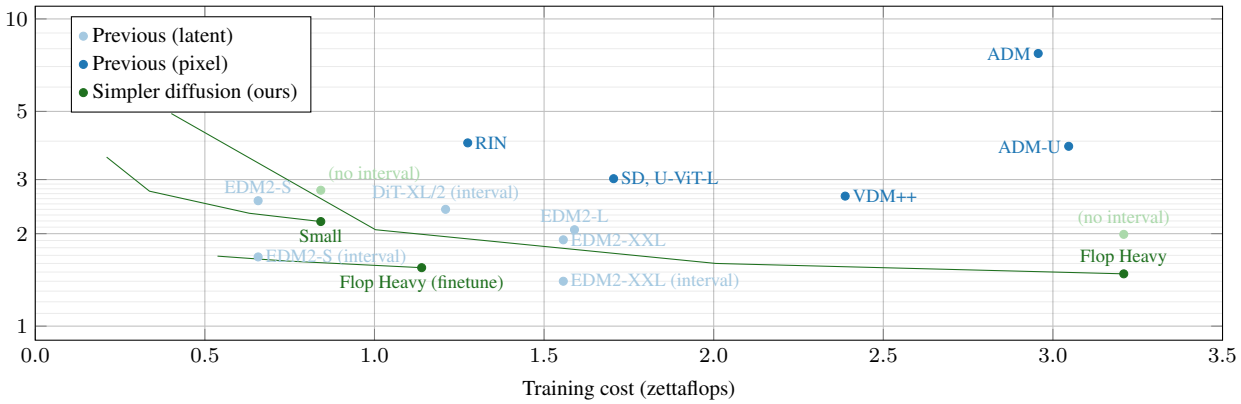


Figure 7. FID versus training cost on ImageNet512. Note that the EDM-2 training curves are without guidance, but the best performance with guidance intervals is reported with a separate marker, the figure source adapted has been adapted from EDM2 [20]. Following their assumptions, one training iteration is three times as expensive as evaluating the model. Our Flop Heavy (finetune) is trained by first training a small model on ImageNet256 for 400k iterations and then finetuning the same model on ImageNet512.

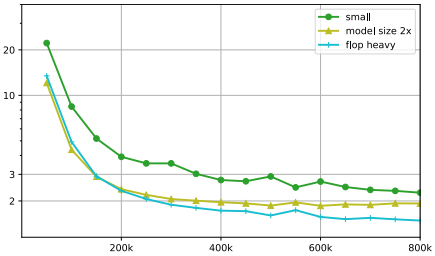


Figure 8. The effect of scaling either channels or tokens, compared to the small model variant. FLOPS between model size 2x and flop heavy are similar.

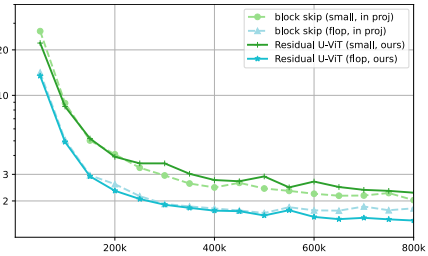


Figure 9. The effect of Residual-UViTs (removing block skip connections). Although smaller models can be effected, larger models are unaffected by the simplification.

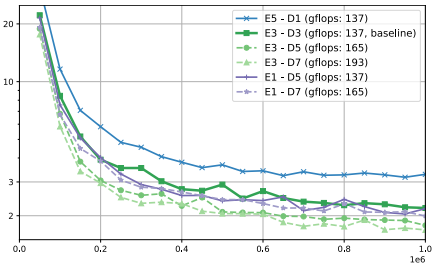


Figure 10. Changing the encoder decoder architecture. Decoder blocks are more important than encoder blocks, and symmetric architectures have decent performance.

cient initially but all schedules perform similarly towards the end. This suggests that there is a lot of transfer across different logsnr values. We also tried the EDM training noise schedule with the sigma power sampling noise schedule [19] (not plotted), with both shifted or the original version, but its narrower logsnr sampling schedule led to performance degradation, achieving about 9 FID at best. This suggests that for high resolution, a wide logsnr range is crucial to cover the important noise levels.

Bias over resolutions In Table 1a, we show the effect of the bias term b on the final performance when trained using a flop heavy model. All models use the same guidance and similar guidance intervals, where the lower bound of the guidance interval has been shifted according to resolution (again approximately 1.5 per 2x resolution change), see Appendix B for more details.

Sigmoid weighting versus the *power loss* In literature, high resolution generation can greatly benefit from having a multiscale loss [6, 7, 14]. To that end we create a multiscale variant that we name the *power loss*. In addition to the sigmoid weighting that effectively upweights low frequency components of the data and downweights high frequency components, the power loss aims to amplify the loss of low sub-band (smooth) signals in the spatial domain via Haar wavelets. For details on this loss see Appendix B.2.

In Table 1b we see that the bias should be adjusted based on the number of Haar transform downsampling levels. Despite being competitive, the performance of power loss is roughly on par with the one with sigmoid weighting but without multiscaling (i.e. downsample levels $= 0$), if we tune the bias of the sigmoid weighting more aggressively. Interestingly, lowering the bias has almost the same effect as increasing the scale. Therefore, building the multiscale variant is not necessary with the sigmoid weighting.

4.2. Scaling channels vs tokens on ImageNet

In this section we show the difference between scaling a model 4x FLOPS, either via channels or by creating larger resolution feature maps (or put differently, feature maps with a higher token count). Note that our small model on

ImageNet512 is patched 4 x 4. This model is scaled either by increasing channels by two (named model size 2x), or by reducing patching to 2 x 2 (named flop heavy, because FLOPS over params is much higher). Indeed as can be seen in Figure 8, the flop heavy model performs much better than the model size 2x, when trained in a standard way on ImageNet512. Note that unlike most experiments, in these experiments it was necessary to use very different optimal guidance settings, more details in Appendix B.

We confirm this is an issue related to overfitting, by training the model size 2x with distribution augmentation [18] the model size 2x variant performs much better, with an FID of 1.6 instead of 1.9 (latter in Figure 8). Still, without any augmentation beyond the standard horizontal flips, the flop heavy model achieves a score of 1.54.

4.3. Removing block skip connection

In this section we study the difference between the residual skip-connections and the blockwise skip-connections. As hypothesized, the small model performance drops a bit because of this. For ImageNet512 a deficit of about 0.2 FID (in Figure 9) and for Imagenet256 a deficit of about 0.05 FID. That the task of fitting ImageNet256 is less affected by removing skip-connections can be explained by the task being easier: The task of fitting ImageNet256 is a subset of fitting ImageNet512.

For larger models, removing blockwise skip-connections in favor of a single residual connection does not degrade performance, and may even lead to slightly increased performance (see Figure 9). This is somewhat unexpected, and the phenomenon persists with different guidance settings as well. The FID curve reminds of the model size 2x curve, which at first improves but flattens out over iterations. Therefore our best hypothesis is that the block skip (flop) baseline may have similar problems such as overfitting. Regardless, it shows that Residual U-ViTs scale well.

Asymmetric U-ViTs Since our U-ViT models encompass only a single skip connection for each level, we can use a different number of ResNet or ViT blocks per level. This allows to move compute from the encoder stage (the downsampling half of the model) to the decoder stage (the upsampling half of the model). From Figure 10 we observe that (a) for a fixed compute budget, it is better to add more parameters to the decoder, the encoder heavy model (E5-D1) is significantly worse than the baseline (E3-D3) and the decoder heavy model (E1-D5); (b) there needs to be some balance between the encoder and the decoder, E1-D7 works less good than E3-D5. (c) by a slightly heavier model (E3-D7, only 10% more parameters than our small (E3-D3) model, the performance significantly increases.

Table 2. Literature Comparison on ImageNet on FID. Note that even though certain NFEs are high (e.g. 4100 for CDM) typically these suffer minor performance degradation for lower NFEs and should not be taken too literally.

Method	NFE	FID
Imagenet 128 x 128		
RIN [16]	1000	2.75
SD, U-ViT-L [14]	512	1.94
StyleGAN-XL [38]	1	1.81
VDM+++ [25]	250	1.75
DisCo-Diff [46]	414	1.73
PaGoDA [22]	1	1.48
SiD2, Flop Heavy (ours)	512	**1.26**
Imagenet 256 x 256		
ADM-G [4]	250	4.59
ADM-U [4]	250	3.94
CDM [12]	4100	4.88
RIN [3, 16]	1000	3.52
SD, U-ViT-L [14]	512	2.44
MDM [7]	250	3.51
StyleGAN-XL [38]	1	2.30
DiT-XL/2 [31]	250	2.27
VDM++ [25]	250	2.12
RDM [42]	356	1.89
PaGoDA [22]	1	1.56
SiD2, small (ours)	512	1.72
SiD2, Flop Heavy (ours)	512	**1.38**
Imagenet 512 x 512		
ADM-G [4]	250	7.72
RIN [3, 16]	1000	3.95
ADM-U [4]	250	3.85
LDM-4-G [32]	250	3.60
SD, U-ViT-L [14]	512	3.02
DiffiT [9]	250	2.67
StyleGAN-XL [38]	1	2.40
DiT-XL/2 [31]	250	3.04
VDM++ [25]	250	2.65
MAGVIT-v2 [51]	64	1.91
PaGoDA [22]	1	1.80
EDM2-S (interval) [27]	63	1.68
EDM2-XXL (interval) [27]	63	**1.40**
SiD2, small (ours)	512	2.19
SiD2, Flop Heavy (ours)	512	1.48
SiD2, Flop Heavy distilled (ours)	16	1.50
Imagenet 1024 x 1024		
RIN [16]	1000	8.72
SiD2, Flop Heavy (ours)	512	**1.75**

Table 3. Zero-shot text-to-image performance evaluated on MSCOCO in FID_{30k}. Our distilled model, evaluated on the more difficult 512 setting, outperforming other works.

Method	Size	NFE	FID_{30k}
DeepFloyd [40]	256		6.7
eDiff-I [1]	256		6.9
Parti [49]	256		7.2
Imagen [35]	256		7.3
simple diffusion [14]	256		8.3
simple diffusion [14]	512	256	9.6
SDv1.5 (eval DMD2) [32]	512	200	7.2
DMD2 [48]	512	4	8.3
SiD2 (ours, bias=-3)	512	256	8.1
SiD2 16-step distilled (ours)	512	16	**6.7**

Table 4. Conditional Video generation performance on Kinetics600 in FVD (resized to 64). For video generation the loss specification is important to obtain good sample quality (see *ablations*).

Method	FVD_{50k}
pixel diffusion	
Video Diffusion Models [13]	16.6
RIN [16]	10.8
Rolling Diffusion [34]	5.2
latent diffusion	
MAGVIT [50]	9.9
MAGVITv2 [51]	4.3
W.A.L.T [8]	3.3
SiD2 (ours, sigmoid bias = 0)	**2.3**
ablations	
eps-mse	5.7
sigmoid bias = +2	3.2
sigmoid bias = -2	2.5

4.4. Literature comparison

Here we compare different models from the literature. In the first section end-to-end diffusion models are compared, and in the second section we compare between all types of generative models. In Table 2 it can be seen that our SiD2 is very competitive with existing generative models: On ImageNet128 and ImageNet256 it achieves state-of-the-art performance, on ImageNet512 it is a little worse but still competitive with the best latent diffusion model and outperforms all other models, pixel-based and latent.

Comparing diffusion models on compute The training cost SiD2 is another helpful way to see how the model scales, as depicted in Figure 7). Importantly, the performance of our pixel-based approach outperforms other significantly. Observe for example the performance of our small variant (using no guidance interval to keep this comparison fair), which roughly the same performance as the VDM++ while only using a third of the training cost, and a fourth of the FLOPS in a single forward pass (see Figure 16 in Appendix C). It is also competitive with the best currently known latent diffusion approach EDM2 [20], but in fairness EDM2 scales a little bit better. These models attain even better performance for a smaller training cost, and a similar story holds for forward pass complexity. End-to-end diffusion can work much better than previously demonstrated, but it is still somewhat outperformed by latent diffusion on quality versus compute comparisons for both sampling and training.

Text to image generation Finally, we show that our method can be used for end-to-end training of text-to-image models. Here, evaluation is more difficult as MSCOCO zero-shot performance depends a lot on the dataset used and is known to not necessarily correspond to actual perceived performance. Nevertheless, it is important that the model achieves performance in the right ballpark. Indeed, when we train SiD2 it performs comparable to other models in the literature on MSCOCO FID_{30K}, and detailed samples have only a small number of artifacts (Figure 2). Furthermore, the moment matching distilled version using [37] outperforms the baseline model as previously observed. This model also outperforms all other zero-shot approaches at 512 resolution as shown in Table 3.

Video Generation To test whether our approach transfers to video, we test our model on Kinetics600 following [13], by conditioning on 5 frames and generating (sometimes referred to as predicting) the subsequent 11 frames. We then compute the FVD between the training set and the generated samples both on the subsequent 11 frames. Even though our method generates at 128×128, we follow [8] and resize samples to have resolution 64×64. Our model achieves SOTA performance and it turns out that tuning the loss is very important to achieve this as shown in Table 4. Note that the latent approaches ([8, 50, 51]) use much less training compute than our models in pixel-space, which we discuss in more detail in the Appendix B.

5. Discussion

In this paper we present Simpler Diffusion (SiD2) and show how one can train end-to-end diffusion and achieve competitive performance with latent models: (1) use the sigmoid loss, (2) with a simpler U-ViT architecture, and (3) favour a flop heavy model design. As a result, our method outperforms existing end-to-end models by a large margin. Nevertheless, at this moment latent diffusion methods still seem to have slightly better scaling properties and it remains to be seen whether this gap can be closed any further. Importantly, for many applications training a separate autoencoder may not be either practical or desired. Therefore, we believe this paper is an important contribution that allows end-to-end modelling directly in the pixel-space.

References

[1] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, Tero Karras, and Ming-Yu Liu. ediffi: Text-to-image diffusion models with an ensemble of expert denoisers. *CoRR*, abs/2211.01324, 2022. 8

[2] Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *CVPR*, 2023. 5

[3] Ting Chen. On the importance of noise scheduling for diffusion models. *arxiv*, 2023. 1, 4, 7

[4] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. *CoRR*, abs/2105.05233, 2021. 4, 7

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. 5

[6] Jiatao Gu, Shuangfei Zhai, Yizhe Zhang, Miguel Ángel Bautista, and Josh M. Susskind. f-dm: A multi-stage diffusion model via progressive signal transformation. *CoRR*, abs/2210.04955, 2022. 4, 6

[7] Jiatao Gu, Shuangfei Zhai, Yizhe Zhang, Josh M. Susskind, and Navdeep Jaitly. Matryoshka diffusion models. *CoRR*, abs/2310.15111, 2023. 4, 6, 7

[8] Agrim Gupta, Lijun Yu, Kihyuk Sohn, Xiuye Gu, Meera Hahn, Fei-Fei Li, Irfan Essa, Lu Jiang, and José Lezama. Photorealistic video generation with diffusion models. In *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part LXXIX*, pages 393–411. Springer, 2024. 8

[9] Ali Hatamizadeh, Jiaming Song, Guilin Liu, Jan Kautz, and Arash Vahdat. Diffit: Diffusion vision transformers for image generation. *CoRR*, abs/2312.02139, 2023. 7

[10] Jonathan Heek, Emiel Hoogeboom, and Tim Salimans. Multistep consistency models. *CoRR*, abs/2403.06807, 2024. 3

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS*, 2020. 1, 5

[12] Jonathan Ho, Chitwan Saharia, William Chan, David J. Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *J. Mach. Learn. Res.*, 23:47:1–47:33, 2022. 4, 7

[13] Jonathan Ho, Tim Salimans, Alexey A. Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS*, 2022. 8

[14] Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *International Conference on Machine Learning, ICML*, pages 13213–13232. PMLR, 2023. 1, 3, 4, 5, 6, 7, 8

[15] Zhongzhan Huang, Zhou Pan, Shuicheng Yan, and Liang Lin. Scalelong: Towards more stable training of diffusion model via scaling network long skip connection. In *NeurIPS*, 2023. 5

[16] Allan Jabri, David J. Fleet, and Ting Chen. Scalable adaptive computation for iterative generation. *CoRR*, abs/2212.11972, 2022. 7, 8

[17] Zeyinzi Jiang, Chaojie Mao, Yulin Pan, Zhen Han, and Jingfeng Zhang. Scedit: Efficient and controllable image diffusion generation via skip connection editing. Technical Report 2312.11392, arXiv, 2023. 5

[18] Heewoo Jun, Rewon Child, Mark Chen, John Schulman, Aditya Ramesh, Alec Radford, and Ilya Sutskever. Distribution augmentation for generative modeling. In *Proceedings of the 37th International Conference on Machine Learning, ICML*, 2020. 3, 7

[19] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS*, 2022. 5, 6

[20] Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and improving the training dynamics of diffusion models. *CoRR*, abs/2312.02696, 2023. 1, 4, 6, 8, 2, 5

[21] Tero Karras, Miika Aittala, Tuomas Kynkäänniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself. *CoRR*, abs/2406.02507, 2024. 2

[22] Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Yuhta Takida, Naoki Murata, Toshimitsu Uesaka, Yuki Mitsufuji, and Stefano Ermon. Pagoda: Progressive growing of a one-step generator from a low-resolution diffusion teacher. *CoRR*, abs/2405.14822, 2024. 7, 5

[23] 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. In *The Twelfth International Conference on Learning Representations*, 2024. 4

[24] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018. 1

[25] Diederik P. Kingma and Ruiqi Gao. Understanding the diffusion objective as a weighted integral of elbos. *CoRR*, abs/2303.00848, 2023. 1, 2, 3, 4, 7

[26] Diederik P. Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *CoRR*, abs/2107.00630, 2021. 2, 5

[27] Tuomas Kynkäänniemi, Miika Aittala, Tero Karras, Samuli Laine, Timo Aila, and Jaakko Lehtinen. Applying guidance in a limited interval improves sample and distribution quality in diffusion models. *CoRR*, abs/2404.07724, 2024. 7, 2

[28] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda*. OpenReview.net, 2023. 2

[29] 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. 4

[30] 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. In *Proceedings of the 41st International Conference on Machine Learning*, pages 34053–34074, 2024. 5

[31] William Peebles and Saining Xie. Scalable diffusion models with transformers. *CoRR*, abs/2212.09748, 2022. 4, 7, 5

[32] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 10674–10685. IEEE, 2022. 1, 3, 4, 5, 7, 8

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. Technical report, ArXiV, 2015. 4

[34] David Ruhe, Jonathan Heek, Tim Salimans, and Emiel Hoogeboom. Rolling diffusion models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. 8

[35] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. *CoRR*, abs/2205.11487, 2022. 8

[36] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *The Tenth International Conference on Learning Representations, ICLR*. OpenReview.net, 2022. 4

[37] Tim Salimans, Thomas Mensink, Jonathan Heek, and Emiel Hoogeboom. Multistep distillation of diffusion models via moment matching. *arXiv preprint arXiv:2406.04103*, 2024. 4, 8

[38] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *SIGGRAPH '22: Special Interest Group on Computer Graphics and Interactive Techniques Conference*, pages 49:1–49:10. ACM, 2022. 7, 5

[39] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651, 2016. 4

[40] Alex Shonenkov, Misha Konstantinov, Daria Bakshandaeva, Christoph Schuhmann, Ksenia Ivanova, and Nadiia Klokova. Deepfloyd if: A text-to-image model with a high degree of photorealism and language understanding, 2023. 4, 8

[41] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. 2

[42] Jiayan Teng, Wendi Zheng, Ming Ding, Wenyi Hong, Jianqiao Wangni, Zhuoyi Yang, and Jie Tang. Relay diffusion: Unifying diffusion process across resolutions for image synthesis. In *The Twelfth International Conference on Learning Representations, ICLR*. OpenReview.net, 2024. 4, 7

[43] Haoning Wu, Shaocheng Shen, Qiang Hu, Xiaoyun Zhang, Ya Zhang, and Yanfeng Wang. Megafusion: Extend diffusion models towards higher-resolution image generation without further tuning. *CoRR*, abs/2408.11001, 2024. 4

[44] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion gans. In *International Conference on Learning Representations*, 2021. 4

[45] 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. 4

[46] Yilun Xu, Gabriele Corso, Tommi S. Jaakkola, Arash Vahdat, and Karsten Kreis. Disco-diff: Enhancing continuous diffusion models with discrete latents. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. 7

[47] 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. 4

[48] Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Frédo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. In *NeurIPS*, 2024. 8

[49] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, Ben Hutchinson, Wei Han, Zarana Parekh, Xin Li, Han Zhang, Jason Baldridge, and Yonghui Wu. Scaling autoregressive models for content-rich text-to-image generation. *CoRR*, abs/2206.10789, 2022. 8

[50] Lijun Yu, Yong Cheng, Kihyuk Sohn, José Lezama, Han Zhang, Huiwen Chang, Alexander G. Hauptmann, Ming-Hsuan Yang, Yuan Hao, Irfan Essa, and Lu Jiang. MAGVIT: masked generative video transformer. *CoRR*, abs/2212.05199, 2022. 8

[51] Lijun Yu, José Lezama, Nitesh Bharadwaj Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Agrim Gupta, Xiuye Gu, Alexander G. Hauptmann, Boqing Gong, Ming-Hsuan Yang, Irfan Essa, David A. Ross, and Lu Jiang. Language model beats diffusion - tokenizer is key to visual generation. In *The Twelfth International Conference on Learning Representations, ICLR*. OpenReview.net, 2024. 7, 8

[52] Wendi Zheng, Jiayan Teng, Zhuoyi Yang, Weihan Wang, Jidong Chen, Xiaotao Gu, Yuxiao Dong, Ming Ding, and Jie Tang. Cogview3: Finer and faster text-to-image generation via relay diffusion. In *Computer Vision - ECCV Proceedings, Part LXXVII*, pages 1–22. Springer, 2024. 4