Flow to the Mode: Mode-Seeking Diffusion Autoencoders for State-of-the-Art Image Tokenization

Kyle Sargent[1], Kyle Hsu[1], Justin Johnson[2], Li Fei-Fei[1], Jiajun Wu[1]
[1]Stanford University, [2]University of Michigan

Abstract

Since the advent of popular visual generation frameworks like VQGAN and latent diffusion models, state-of-the-art image generation systems have generally been two-stage systems that first tokenize or compress visual data into a lower-dimensional latent space before learning a generative model. Tokenizer training typically follows a standard recipe in which images are compressed and reconstructed subject to a combination of MSE, perceptual, and adversarial losses. Diffusion autoencoders have been proposed in prior work as a way to learn end-to-end perceptually-oriented image compression, but have not yet shown state-of-the-art performance on the competitive task of ImageNet-1K reconstruction. We propose FlowMo, a transformer-based diffusion autoencoder that achieves a new state-of-the-art for image tokenization at multiple compression rates without using convolutions, adversarial losses, spatially-aligned two-dimensional latent codes, or distilling from other tokenizers. Our key insight is that FlowMo training should be broken into a mode-matching pre-training stage and a mode-seeking post-training stage. In addition, we conduct extensive analyses and explore the training of generative models atop the FlowMo tokenizer. Our code and models are available at https://kylesargent.github.io/flowmo.

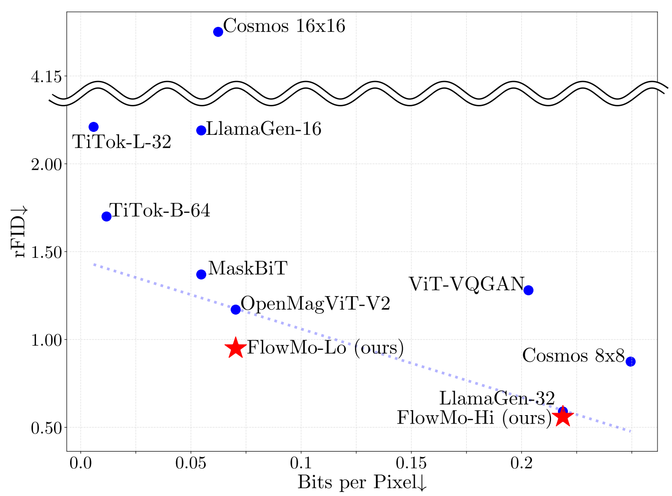
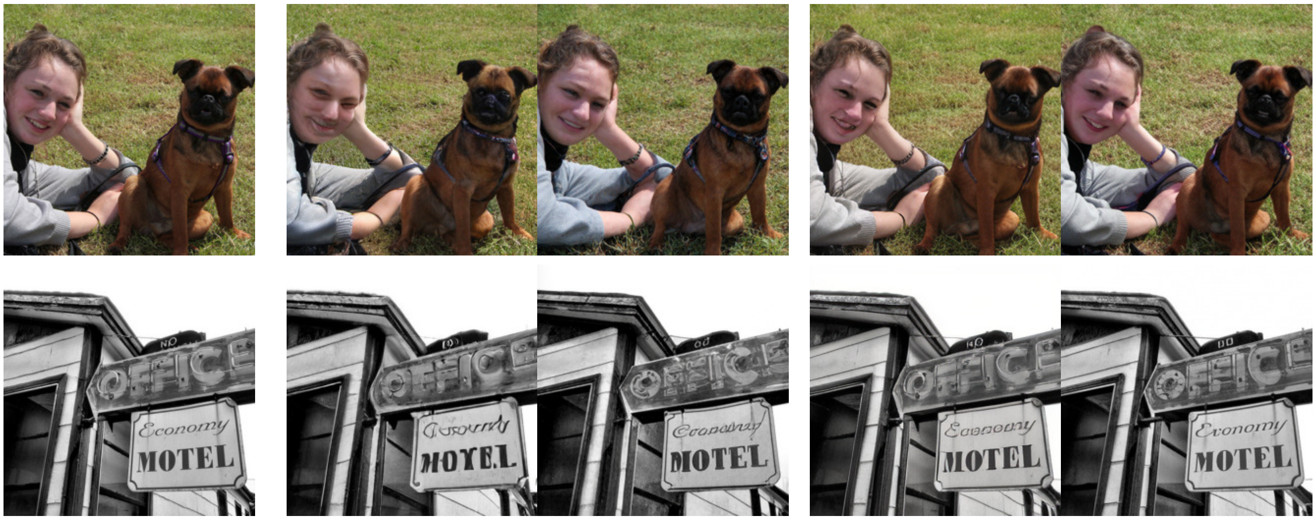


Figure 1. **Discrete tokenizer comparison.** State-of-the-art discrete tokenizers are benchmarked by encoding and reconstructing the ImageNet-1K validation dataset at 256×256 resolution. with performance measured in reconstruction FID (rFID), which trades off against compression rate as measured in bits per pixel (BPP). Whether trained for reconstruction at a low BPP (FlowMo-Lo) or high BPP (FlowMo-Hi), FlowMo achieves state-of-the art image tokenization performance compared with the respective baselines. Moreover, *FlowMo is a transformer-based diffusion autoencoder which does not use convolutions, adversarial losses, or proxy objectives from auxiliary tokenizers.*

1. Introduction

Generative models such as diffusion models [14, 21, 33, 34, 46, 47] and discrete autoregressive models [6, 13] have seen compelling applications in the creation of image and video content [4, 29, 44]. Although less common systems such as cascaded [44] and single-stage [18, 22] systems have been explored, state-of-the-art systems for visual generation generally consist of two stages: first, a "tokenizer" is learned to compress pixel data to a smaller and more tractable discrete or continuous latent space; second, a generative model is trained over this compressed latent space. Due to the dominance of this two-stage paradigm, tokenization has become an active research area in its own right, with multiple influential works focusing heavily on tokenizer training and design [15, 35, 51, 56, 57]. In this work, we focus on tokenizers with a discrete latent space.

Since VQGAN [13], a dominant architecture and training scheme for discrete image tokenizers has emerged. State-of-the-art image tokenizers are typically convolutional autoencoders trained to downsample visual data to a 2-dimensional, spatially-aligned latent code, and then upsample the latent code to a reconstruction regularized by adversarial and perceptual losses. Various deviations from this setup have been proposed: TiTok [57] used a transformer-based architecture and 1-dimensional latent code although relying on a traditional CNN-based tokenizer for an initial



Original image	OpenMagViT-V2 (rFID=1.17)	FlowMo-Lo (Ours) **(rFID=0.95)**	LlamaGen-32 (rFID=0.59)	FlowMo-Hi (Ours) **(rFID=0.56)**

Figure 2. **Example reconstructions.** Comparison of original and reconstructed images of faces and text. OpenMagViT-V2 and FlowMo-Lo are 0.07-bits per pixel tokenizers to be compared against each other. LlamaGen-32 and FlowMo-Hi are 0.22-bits per pixel tokenizers to be compared against each other. Best viewed zoomed-in in the electronic version. More comparisons are available on our website.

distillation stage, and ViT-VQGAN [55] used a transformer-based encoder and decoder. In general, however, the same basic setup for image tokenization has remained dominant.

In this work, we propose FlowMo (**Flow** to the **Mo**de), a significant technical departure from the current state-of-the-art for image tokenization. First, in tackling the task of image tokenization, we propose to model the multimodal reconstruction distribution using a rectified flow objective for the decoder [33, 34]. Second, we use a fully transformer-based architecture to encode to and decode from a 1-dimensional latent code. Third, we optimize FlowMo end-to-end, without distilling from a pre-existing 2-dimensional tokenizer [57] or encoding on top of a latent space from a 2-dimensional tokenizer [1].

Most importantly, FlowMo achieves state-of-the-art performance via a key insight. FlowMo is a diffusion autoencoder, a system which has been explored in prior work [2, 23, 42, 54]. Yet, the state-of-the-art for the most competitive perceptual reconstruction benchmark, ImageNet-1K [11], continues to be dominated by more traditional CNN- or GAN-based tokenizers. FlowMo achieves state-of-the-art tokenization via the following key insight: **for the task of perceptual reconstruction, it is better to sample modes of the reconstruction distribution that are perceptually close to the original image than to try to match all modes.** Hence, we divide FlowMo training into a *mode-matching pre-training stage*, in which the system is trained end-to-end with a diffusion loss on the decoder like typical diffusion autoencoders, and a *mode-seeking post-training stage*, in which the system is trained to selectively drop modes of the reconstruction distribution which are not close to the original image. We explain both stages in Section 3.

Despite its significant methodological departures from prior work, FlowMo is state-of-the-art when compared with the strongest available tokenizers at multiple BPPs. Our main contributions are as follows:

- We present a simple but novel architecture for image tokenization based on diffusion autoencoders [42] and the multimodal diffusion image transformer (MMDiT) [14].
- We present a novel two-stage training scheme for diffusion autoencoders, consisting of *mode-matching pre-training* and *mode-seeking post-training*.
- We set a new state-of-the-art for perceptual image tokenization in the 0.07 bits per pixel and 0.22 bits per pixel regimes, in terms of rFID, PSNR, and other metrics. We also show that a generative model trained with a FlowMo tokenizer can match (though not beat) the performance of a generative model trained with a traditional tokenizer.
- We conduct an extensive analysis of the system design choices in FlowMo, outlining several subtle but critical decisions in the noise schedule, sampler design, model design, quantization strategy, and post-training.

2. Related Work

Image tokenization. State-of-the-art systems for visual generation generally consist of two stages (with a few notable exceptions attempting to learn directly in pixel space [10, 18, 22, 24]). The first stage is a "tokenizer" which reduces the spatial or spatiotemporal dimension of pixel data by projecting to a continuous or discrete latent space. The dominance of this paradigm has led to numerous works that study tokenizers as important components in their own right [7, 35, 48, 51, 56, 57]. In this work, we study tokenizers with discrete latents. Unlike prior work, FlowMo is the first diffusion autoencoder-based architecture to achieve state-of-the-art performance on ImageNet-1K reconstruction.

Diffusion autoencoders. Diffusion models are popular for visual generation [4, 43], and simplified frameworks such as rectified flow [33, 34] have led to further adoption. The idea of learning an autoencoder end-to-end with a diffusion decoder was first proposed in [42]. Various works have followed up on this idea, studying diffusion autoencoders for representation learning [25] and in particular for perceptually-oriented image compression [2, 38, 54]. $\epsilon-$VAE [59] explores a diffusion autoencoder objective in the context of GAN-based tokenizer trained with adversarial and perceptual losses, while DiscoDiff [52] uses a diffusion autoencoder to learning high-level semantic discrete latents for images prior to an autoregressive generation stage.

Diffusion post-training. Various work has attempted to improve the quality of diffusion model samples with dedicated post-training strategies, the goal of which is generally to instill desired attributes such as aesthetic quality. DDPO [3] and DPOK [16] explore reinforcement learning objectives, while DRAFT [9] and AlignProp [41] explore backpropagation through the sampling chain. FlowMo uses a backpropagation-based post-training strategy, adapted to the continuous noise schedule of rectified flow and the unique setting of image tokenization.

Concurrent work. In work concurrent to ours, DiTo [8] studies diffusion autoencoders for the continuous image tokenization task. Unlike DiTo, we focus on discrete image tokenization and compare against the strongest state-of-the-art discrete tokenizers available [35, 48]. Meanwhile, FlexTok [1] proposes a system in which diffusion tokenizers are learned on top of a latent space from a traditional continuous variational autoencoder (VAE) trained with perceptual, adversarial, and reconstruction losses. FlowMo does not rely on an auxiliary VAE. The reader may consult these works for a more complete picture of this developing field.

3. Method

Existing state-of-the-art tokenizers have enabled considerable progress in visual generation. However, they have a variety of drawbacks. First, they require adversarial losses [13, 35, 56], which can be unstable and difficult to tune. They almost all require CNNs in at least one training stage, making it difficult to leverage the hardware efficiency and well-understood scaling behavior of transformers [49, 53]. Finally, they often leverage distillation from previously trained tokenizers [1, 57] in order to achieve state-of-the-art performance.

FlowMo is motivated by several key goals. We propose a tokenizer that is

- **Diffusion-based.** Prior discrete tokenizers leverage adversarial losses to sample a plausible mode of the reconstruction distribution, which necessitates adaptive gradient scale computation [13], LeCam regularization [50, 56], or careful loss weight tuning for stability. Instead, we will use a diffusion decoder, given that diffusion models have proven simple, well-suited for multi-modal modeling, and reliable at scale [4, 14].
- **Transformer-only, with 1-dimensional latent code.** Nearly every state-of-the-art image tokenization framework uses a CNN-based architecture or predicts a spatially aligned 2-dimensional latent code, with the exception of TiTok [57], which still relies on distilling from a pre-trained CNN-based tokenizer with a 2-dimensional latent code. While these choices may provide a useful inductive bias, we feel transformer-based tokenizers with 1-dimensional latent codes may eventually provide more efficiency and flexibility at large data and model scales. FlowMo learns 1-dimensional latent codes in a transformer-based architecture following MMDiT.
- **State-of-the-art.** There has already been considerable work in diffusion autoencoders [2, 23, 25, 42], but FlowMo is the first to achieve state-of-the-art perceptual reconstruction on ImageNet-1K.

FlowMo achieves these goals, as we will now show now, via an exposition of the system, and later (in Section 4), via extensive experiments, ablations, and analyses. As a diffusion autoencoder, the distribution of reconstructed images x given the latent code c, denoted $p(x|c)$, is necessarily multimodal given the limited information in c. Our key insight is that in order to achieve state-of-the-art tokenization with a diffusion autoencoder, multiple steps should be taken to bias $p(x|c)$ towards modes of high perceptual similarity to the original image. We achieve this by using two key ingredients: (1) *mode-seeking post-training*, explained in 3.3, and (2) a *shifted sampler*, explained in Section 3.4.

We will now describe FlowMo in detail. First, we will go over the FlowMo architecture, which is a simple transformer-based architecture based on MMDiT [14]. Then, we will explain the two training stages: mode-matching pre-training (Stage 1A) and mode-seeking fine-tuning (Stage 1B). Finally, we will discuss generative modeling over the FlowMo latent space (Stage 2) and sampling.

3.1. Architecture

The architectural diagram of FlowMo is given in Figure 3. FlowMo is a diffusion autoencoder. The encoder encodes images x to a quantized latent c. The decoder is a conditional diffusion model which learns the conditional distribution of reconstructed images $p(x|c)$.

FlowMo consists of an encoder e_θ and decoder d_θ. Both are transformers based on the multimodal diffusion image

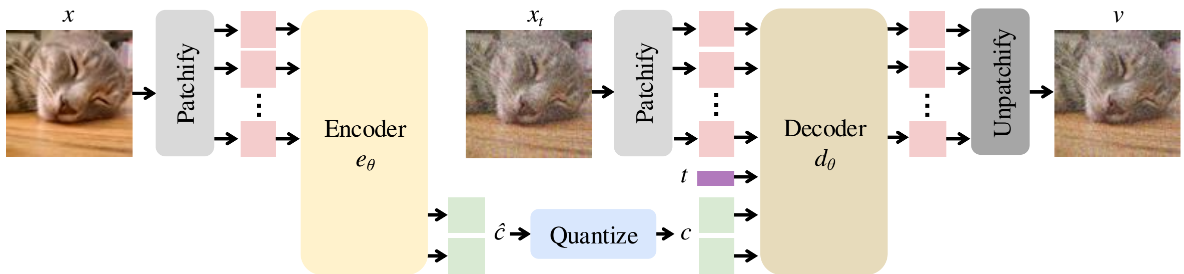


Figure 3. **FlowMo architecture**. FlowMo is a diffusion autoencoder which encodes images x to a latent $\hat{c}$ which is quantized to c. Then, the model decodes a rectified flow velocity v conditioned on c as well as a noise level t and noised image x_t.

transformer (MMDiT) [14] adapted from Flux [32]. Given a patchified [12] image $x \in \mathbb{R}^n$ and an initial latent code $c_0 \in \mathbb{R}^d$ (a vector of all zeroes), the encoder produces a latent token sequence

$$\hat{c} = e_\theta(x, c_0), \tag{1}$$

by interacting the concatenated sequence of latent and image tokens with self-attention while maintaining separate streams for each modality. $\hat{c}$ is then binarized elementwise via a quantization operation q following lookup-free quantization (LFQ) [56], yielding

$$c = q(\hat{c}) = 2 \cdot \mathbb{1}[\hat{c} \geq 0] - 1. \tag{2}$$

Following rectified flow [34], the decoder is trained to model a velocity field v from noise to data defined as

$$v = d_\theta(x_t, c, t). \tag{3}$$

The decoder processes x_t and c identically to how the encoder processes x and c_0, but additionally accepts the time (or noise level) parameter t to condition each MMDiT block via AdaLN modulation [40]. The encoder e_θ and decoder d_θ are architecturally symmetric but differently sized, with the decoder being considerably larger and deeper. We use the μP parameterization [53] for all models in order to simplify hyperparameter transfer between exploratory and scaled-up configurations.

3.2. Stage 1A: Mode-matching pre-training

In the first training stage, our goal is to train the encoder e_θ and decoder d_θ jointly so that the quantized c is maximally informative about x, and so that $p_\theta(x|c)$ matches the true distribution $p(x|c)$, which is necessarily multimodal since the quantized c contains only limited information. A full diagram of Stage 1A is given in Figure 4. For theoretical justification, we appeal to prior work in diffusion autoencoders [54], which notes that the end-to-end diffusion objective corresponds to a modified variational lower bound with non-Gaussian decoder $p(x|c)$.

FlowMo is trained end-to-end as a diffusion autoencoder [42]. Specifically, the FlowMo encoder and decoder are trained end-to-end to optimize the rectified flow loss $\mathcal{L}_{\text{flow}}$ on the decoder output. Given noise $z \sim \mathcal{N}(0, I)$, data $x \sim p_x$, and time (or noise level) $t \sim p_t$, $t \in [0, 1]$, we define

$$x_t = tz + (1-t)x \tag{4}$$

and we optimize the flow-matching objective

$$\mathcal{L}_{\text{flow}} = \mathbb{E}\left[\left\|x - z - d_\theta(x_t, q(e_\theta(x)), t)\right\|_2^2\right]. \tag{5}$$

We also use a learned perceptual distance d_{perc} [26] to supervise the 1-step denoised prediction of the network via

$$\mathcal{L}_{\text{perc}} = \mathbb{E}\left[d_{\text{perc}}(x, x_t + t d_\theta(x_t, q(e_\theta(x)), t))\right]. \tag{6}$$

Finally, on the latent code c we use the entropy and commitment losses of LFQ following [56], with

$$\mathcal{L}_{\text{ent}} = \mathbb{E}\big[H(q(\hat{c})\big] - H(\mathbb{E}\big[q(\hat{c})\big]), \tag{7}$$

$$\mathcal{L}_{\text{commit}} = \mathbb{E}\left[\left\|\hat{c} - q(\hat{c})\right\|_2^2\right]. \tag{8}$$

For simplicity, these two losses could be replaced with finite scalar quantization (FSQ) [36], but we find LFQ to perform slightly better. Our training loss in Stage 1A is

$$\mathcal{L}_{\text{flow}} + \lambda_{\text{perc}}\mathcal{L}_{\text{perc}} + \lambda_{\text{commit}}\mathcal{L}_{\text{commit}} + \lambda_{\text{ent}}\mathcal{L}_{\text{ent}}, \tag{9}$$

with loss weights and further details given in the supplementary material. Images are rescaled to lie in $[-1, 1]$. The noise level t is sampled from a thick-tailed logit-normal distribution following Stable Diffusion 3 [14].

3.3. Stage 1B: Mode-seeking post-training

In this stage, our goal is to optimize the decoder distribution $p_\theta(x|c)$ to seek modes which are perceptually of high similarity to the original image. We accomplish this by freezing the encoder and co-training the decoder on $\mathcal{L}_{\text{flow}}$ along with a post-training objective $\mathcal{L}_{\text{sample}}$ inspired by prior work on diffusion model post-training [9, 41].

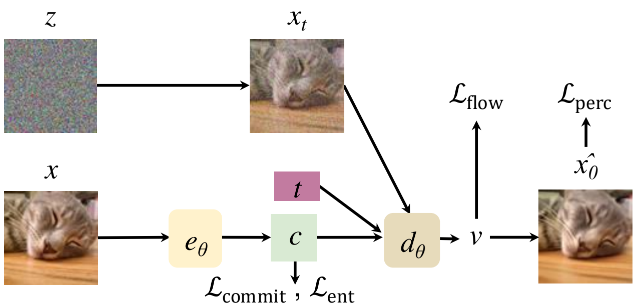


Figure 4. **Stage 1A**. The encoder and decoder are trained end-to-end with output losses $\mathcal{L}_{\text{perc}}, \mathcal{L}_{\text{flow}}$ and latent losses $\mathcal{L}_{\text{commit}}, \mathcal{L}_{\text{ent}}$.

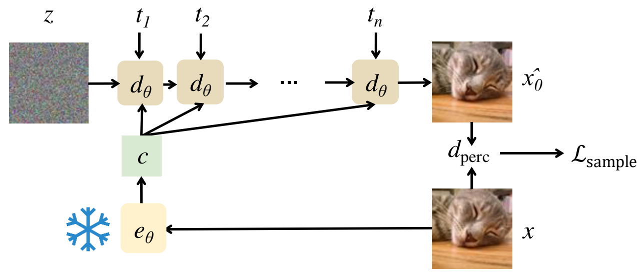


Figure 5. **Stage 1B**. The frozen encoder e_θ encodes the input image to c to condition the decoder d_θ, which is trained via back-propagation through the entire sampling chain. We also co-train with $\mathcal{L}_{\text{flow}}$, which is the same as in Figure 4.

$\mathcal{L}_{\text{sample}}$ is simply a perceptual loss on the n-step sample from integrating the probability flow ODE, which we differentiate through. We sample the timesteps for integration $t_1, ..., t_n$ randomly to enable experimentation with different sampling schedules at test time, and use gradient checkpointing in order to backpropagate through the sampling chain. A full diagram of Stage 1B is given in Figure 5.

Let $d_{t_i}(x_t)$ denote the flow sample update function, i.e.,

$$d_{t_i}(x_t) = x_t + (t_{i+1} - t_i) d_\theta(x_t, c, t_i). \tag{10}$$

Then, we define

$$\mathcal{L}_{\text{sample}} = \mathbb{E}\left[d_{\text{perc}}\left(x, d_{t_n} \circ d_{t_{n-1}} \circ \cdots \circ d_{t_1}(z)\right)\right], \tag{11}$$

and our complete loss for Stage 1B is

$$\mathcal{L}_{\text{flow}} + \lambda_{\text{sample}} \mathcal{L}_{\text{sample}}. \tag{12}$$

We find the value of λ_{sample} in this stage to be particularly important, and use $\lambda_{\text{sample}} = 0.01$. Too small a weight on $\mathcal{L}_{\text{sample}}$ results in poor rFID as the network forgets the perceptual features acquired in Stage 1A, whereas too large a weight results in either so-called "reward-hacking" as the decoder d_θ overfits to d_{perc}, or training divergence. We use an LPIPS-VGG network [58] as the perceptual network in Stage 1A, but, following [51], we use a ResNet as the perceptual network in Stage 1B. This and other design choices are explained in the supplementary material, and we justify there via ablation that simply using the ResNet network on the 1-step denoised prediction as in $\mathcal{L}_{perc}$ is ineffective; the post-training stage presented is necessary. This stage is computationally expensive as it requires backpropagation through the full sampling chain. We use $n = 8$ and find it generalizes well to other n at sampling time.

3.4. Sampling

Given a quantized latent code c, the multimodal distribution of reconstructed images given c, denoted $p(x|c)$, is sampled by solving the probability flow ODE

$$dx_t = v(x_t, t) = d_\theta(x_t, c, t), \tag{13}$$

given an initial $x_1 \sim \mathcal{N}(0, I)$. As an aside, FlowMo can also be used to compute sample log-likelihoods by solving the flow ODE in reverse following flow matching [33].

At inference, we integrate the rectified flow ODE with a timestep spacing given by a tunable shift hyperparameter ρ which signifies the concentration of sampling timesteps towards lower noise levels. For a timestep spacing

$$(t_1, ..., t_n) = \left(\left(\frac{n}{n}\right)^\rho, \left(\frac{n-1}{n}\right)^\rho, \ldots, \left(\frac{1}{n}\right)^\rho\right), \tag{14}$$

setting $\rho = 1$ reduces to the usual linearly-spaced rectified flow ODE sampler. In the extreme, letting $\rho \to \infty$ means taking a single large step at $t = 1$, which corresponds to regressing x given c [5]. We use $\rho = 4$, which corresponds to taking large steps closer to $t = 1$ and concentrating the sample towards the mean of $p(x|c)$, while still spending significant sampling FLOPs at the low noise levels, a choice which prior work has shown to be critical for rFID [23]. Our sampler significantly improves both rFID and PSNR.

3.5. Stage 2: Latent generative modeling

As in other work training autoencoders with discrete latent spaces [13, 35, 48, 56], we verify that our tokenizer can be used to train a high-quality second-stage generative model. We use MaskGiT [6] and our settings for this stage are largely taken from MaskGiT and TiTok [6, 57]. Additional details are in the supplementary material.

4. Experiments

4.1. Main results

Tokenization. For the main task of tokenization, all tokenizers take an image input, encode it to a quantized latent, and then reconstruct the image. All tokenizers are trained on ImageNet-1K [11]. Reconstruction quality is measured in rFID [19], PSNR, SSIM, and LPIPS [58]. We evaluate on the ImageNet-1K validation set, at 256×256 resolution.

The amount of information contained in a tokenizer's quantized latent code is measured in bits per pixel (BPP).

BPP	Model	Tokens per image	Vocab size	rFID↓	PSNR↑	SSIM↑	LPIPS↓
0.006	TiTok-L-32 [57]	32	2^{12}	2.21	15.60	0.359	0.322
0.012	TiTok-B-64 [57]	64	2^{12}	1.70	16.80	0.407	0.252
0.023	TiTok-S-128 [57]	128	2^{12}	1.71	17.52	0.437	0.210
0.055	LlamaGen-16 [48]	256	2^{14}	2.19	20.67	0.589	0.132
	MaskBiT† [51]	256	2^{14}	1.37	21.5	0.56	-
0.062	Cosmos DI-16x16 [57]	256	$\approx 2^{16}$	4.40	19.98	0.536	0.153
0.070	OpenMagViT-V2 [35]	256	2^{18}	1.17	21.63	0.640	**0.111**
	FlowMo-Lo (ours)	256	2^{18}	**0.95**	**22.07**	**0.649**	0.113
0.203	ViT-VQGAN† [55]	1024	2^{13}	1.28	-	-	-
0.219	LlamaGen-32 [48]	1024	2^{14}	0.59	24.44	0.768	**0.064**
	FlowMo-Hi (ours)	1024	2^{14}	**0.56**	**24.93**	**0.785**	0.073
0.249	Cosmos DI-8x8 [57]	1024	$\approx 2^{16}$	0.87	24.82	0.763	0.072

Table 1. **Tokenization results**. Horizontal lines separate tokenizers trained at different BPPs. FlowMo achieves state-of-the-art performance at multiple BPPs compared to existing state-of-the-art tokenizers. †*Results from the original paper.*.

Tokenizer	FID ↓	IS ↑	sFID ↓	Prec. ↑	Rec. ↑
OpenMagViT-V2	**3.73**	241	10.66	0.80	**0.51**
FlowMo-Lo (ours)	4.30	**274**	**10.31**	**0.86**	0.47

Table 2. **Generation results**. We compare two MaskGiT transformers trained atop two tokenizers at the same BPP.

Model	rFID ↓	PSNR ↑	SSIM ↑	LPIPS ↓
DiTo† [8]	0.78	24.10	0.706	0.102
FlowMo (ours)	**0.65**	**26.61**	**0.791**	**0.054**

Table 3. **DiTo comparison.** Comparison with concurrent work DiTo [8]. †*Results from the original paper.*

Where S is the latent sequence length and V is the token vocabulary size, the BPP of a tokenizer is computed as $\frac{S \log_2(V)}{256^2}$. Tokenizers trained at different BPP cannot be compared apples-to-apples since access to more bits will improve performance. Therefore, we train two models in order to match existing state-of-the-art tokenizers. FlowMo-Lo is trained at 0.0703 BPP to match the BPP of OpenMagViT-V2 [35]. FlowMo-Hi is trained at 0.219 BPP to match the BPP of LlamaGen-32 [48]. Both FlowMo-Lo and FlowMo-Hi are exactly the same FlowMo architecture except for the BPP. To match the BPP of the tokenizer against which we compare, we modify the FlowMo token vocabulary size or number of tokens.

We achieve state-of-the-art results at both BPPs in terms of rFID [19], PSNR, and SSIM. The only exception to FlowMo's strong performance is with respect to the LPIPS metric, for which FlowMo still tends to underperform. Extended visual comparisons are available in the supplementary material and on our website.

Generation. For the task of generation, we train the generative model MaskGiT [6] over the encoded token sequences produced by different tokenizers. We then decode the generated token sequences via the respective tokenizers, and measure the image quality in terms of image generation metrics, namely FID [19], sFID [37], Inception Score [45], and Precision and Recall [30]. The generation metrics are evaluated on the ImageNet-1K benchmark. Despite achieving stronger tokenization, using FlowMo as a tokenizer for a downstream generative model improves some but not all generation metrics. There is an interesting and complicated interplay between tokenizer quality and generation quality, and we hope to improve the results in future work.

Training hyperparameters such as model size, batch size, and training length were identical in Table 2 save the tokenizer. Specifically, we train for 300 epochs atop both tokenizers with batch size 128 and learning rate 1×10^{-4}. Further hyperparameter settings are given in the supplementary material. While we lack the resources to train state-of-the-art generative models, which require e.g. $1,000 - 1,080$ epochs of training with batch size $1,024 - 2,048$ [51, 57], our goal here is not to set a state of the art for generation, but to fairly compare tokenizers from the perspective of the latent space they provide for generation.

Other comparisons. We also conduct a tokenization comparison between FlowMo and DiTo [8], a similar concurrent work, in Table 3. Exclusively for this comparison, we train a tokenizer with a continuous latent space, with LayerNorm as the final encoder layer and the "noise sync" augmentation as proposed in DiTo. We also equalize the overall latent size, using 256 tokens with a token dimension of 16.

Experimental details. All our models are parameterized in μP [53] so that hyperparameters can be "μ-transferred" be-

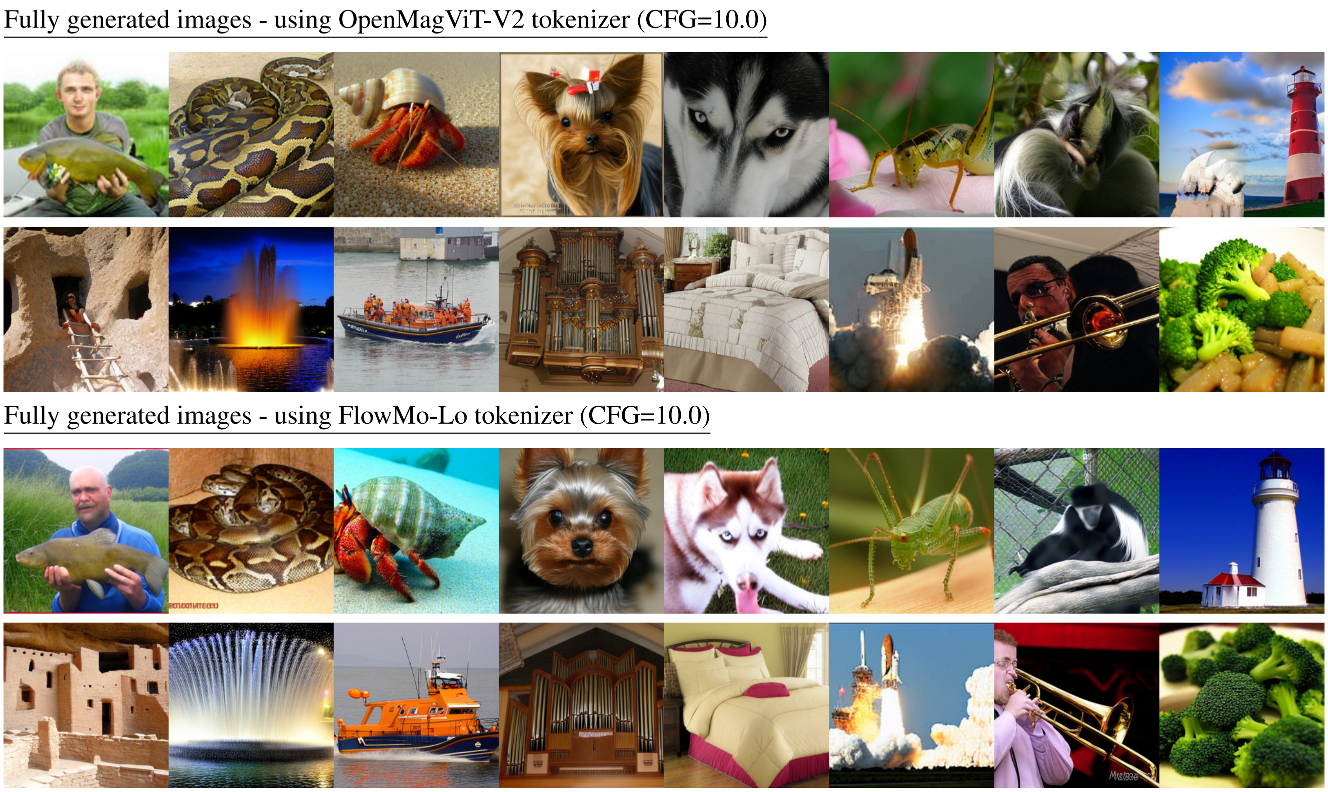


Figure 6. **Generated images**. Example generated images from MaskGiT trained with different tokenizers. FlowMo can be used to train high-quality second-stage generative models. The corresponding class indices are identical for ease of comparison.

tween exploratory and scaled-up configurations. The majority of our experiments were conducted on mixed hardware (A6000, L40S, A100, H100, or H200 GPUs as available) with ViT patch size 8 and hidden dimension 768; this is also the configuration used for ablation studies in Table 4.

Our final experiments (FlowMo-Lo, FlowMo-Hi) were conducted on an 8×H100 node using a scaled-up model with patch size 4 and hidden dimension increased to 1152 for the decoder only. All other hyperparameters were $\mu-$transferred directly. We pre-train FlowMo-Lo for 130 epochs and FlowMo-Hi for 80 epochs, and then post-train both for approximately 1 epoch. In pre-training, performance in rFID does not saturate. Longer training would require more resources but likely further benefit performance.

We train both FlowMo-Lo and FlowMo-Hi with batch size 128 and learning rate 10^{-4}. We forcibly normalize the weight matrices in the MLP blocks of the encoder and decoder per step, following EDM2 [27], in order to counteract exploding activations and weight matrices. All models are trained in PyTorch [39] with bfloat16 precision. We use the Adam optimizer [28] with $(\beta_1, \beta_2) = (0.9, 0.95)$ since we noticed higher β_2 was unstable, possibly due to bfloat16 precision or the long transformer sequence length. We set the encoder learning rate to 0 after $200,000$ training steps. We use an exponential moving average with rate 0.9999.

For the full model configuration and hyperparameters, please consult the supplementary material.

4.2. Analysis

Ablation studies. We conduct ablation studies on tokenization performance, analyzing the impact of different decisions on rFID and other metrics. The setup is the same as our main ImageNet-1K tokenization experiments.

We first conduct an ablation study for design decisions in Stage 1A, in Table 4. We analyze these decisions and reference prior diffusion autoencoder works where applicable. For these experiments, we used a configuration with reduced μP width for efficient experimentation.

Doubled patch size. In vision transformers [12] and derived architectures [14], "patch size" determines the length of the sequence to which an image is converted, with smaller patch sizes resulting in more computationally intensive models. Prior work on transformer-based diffusion models in pixel space has noted the importance of small patch size [24]. We corroborate this: increasing patch size compromises all metrics.

Encoder trained with MSE. Prior diffusion autoencoder works have explored first training an autoencoder with an MSE- or LPIPS-based regression objective, potentially with an adversarial loss as well, and then using the result-

Figure 7. **Noise schedule ablation.** Training FlowMo with a logit-normal noise schedule [14] results in discoloration.

ing frozen features as conditioning for a diffusion decoder [2, 23, 38]. Though this style of training may boost PSNR, for best rFID it is essential to train the entire system end-to-end at all noise levels to ensure the latent code contains helpful features for velocity estimation at all noise levels.

No perceptual loss. Prior work in diffusion autoencoders [54] has noted the importance of a perceptual loss on the 1-step denoised prediction of the diffusion decoder. Without a perceptual loss, performance degrades.

FSQ quantization. Compared with our default quantization strategy LFQ [56], we observe that FSQ [36] achieves slightly better final training loss and pairwise metric performance, but at the cost of rFID, which is most critical. We experimented with alternative quantization techniques as well [17, 60], but achieved best performance with LFQ.

Logit-normal noise. The logit-normal noise schedule for rectified flow training [14] assigns 0 probability mass to $t \in \{0, 1\}$. Since our conditioning signal is extremely strong, the estimate of $v(x_t, t)$ at $t = 1$ (pure noise) is critical. Using the usual logit-normal schedule causes PSNR and rFID to collapse. Instead, we use a thick-tailed logit-normal noise schedule: we sample the noise level from a uniform distribution on $[0, 1]$ 10% of the time. Without this modification, the result is low PSNR and discoloration. See Figure 7 for an illustrative example.

Unshifted sampler. We integrate the rectified flow ODE using a spacing of timesteps t, which is proportionately more concentrated towards $t = 0$, as explained in Section 3.4. Using linearly-spaced timesteps degrades performance.

No guidance. One advantage of FlowMo over a GAN tokenizer is the ability to leverage techniques such as classifier-free guidance [20] and guidance intervals [31] to improve perceptual quality. Without applying classifier guidance in a limited interval, rFID is worsened. We enable guidance by dropping out the quantized latent code c 10% of the time during tokenizer training. We do not leverage any class information in the tokenizer.

Table 5 is an ablation study for the inclusion of Stage 1B. Without this critical stage, performance on all metrics drops. Intriguingly, despite only training on $\mathcal{L}_{\text{sample}}$, this stage also improves PSNR. We illustrate in Figure 8 how the FlowMo decoder remains multimodal after post-training.

Model	rFID↓	PSNR↑	LPIPS↓
FlowMo (fewer params.)	**2.87**	20.71	0.15
with doubled patch size	6.39	19.94	0.17
with MSE-trained encoder	3.82	21.40	0.15
without perceptual loss	13.86	**22.11**	0.21
with FSQ quantization	3.14	21.31	**0.14**
with logit-normal schedule	4.08	16.45	0.21
without shifted sampler	3.42	20.25	0.16
without guidance	3.28	20.67	0.16

Table 4. **Stage 1A Ablation**. Deviating from FlowMo design choices compromises either PSNR or rFID. We prioritize rFID in our model due to its correlation with perceptual quality.

Model	rFID↓	PSNR↑	LPIPS↓
FlowMo-Lo	1.10	21.38	0.134
FlowMo-Lo (post-trained)	**0.95**	**22.07**	**0.113**
FlowMo-Hi	0.73	24.02	0.086
FlowMo-Hi (post-trained)	**0.56**	**24.93**	**0.073**

Table 5. **Stage 1B ablation**. Without the post-training stage, performance is considerably worse.

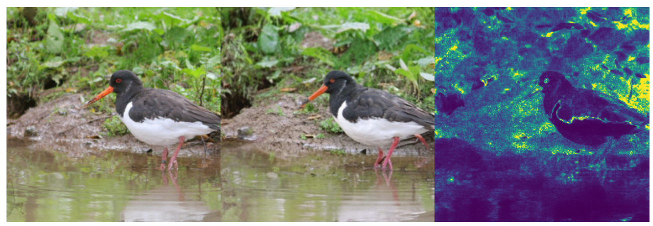


Figure 8. **Multimodal reconstruction.** After post-training, FlowMo reconstruction remains multimodal, but biased towards preserving the perceptually relevant details of the image, which manifests here by the variance concentrating in the background.

Limitations. The primary limitation of FlowMo is inference time. FlowMo requires multiple model forward passes (we use $n = 25$ steps in our work) to generate a sample from the decoder given a quantized latent code.

5. Conclusion

We have presented FlowMo, a transformer-based diffusion autoencoder that achieves state-of-the-art image tokenization. We have shown state-of-the-art performance on the competitive ImageNet-1K benchmark at 256×256 resolution *without using 2D spatially aligned latent codes, adversarial losses, proxy objectives from auxiliary tokenizers, or convolutions*, unlike much prior work.

Acknowledgments. This work is in part supported by ONR MURI N00014-22-1-2740.

References

[1] Roman Bachmann, Jesse Allardice, David Mizrahi, Enrico Fini, Oğuzhan Fatih Kar, Elmira Amirloo, Alaaeldin El-Nouby, Amir Zamir, and Afshin Dehghan. FlexTok: Resampling Images into 1D Token Sequences of Flexible Length. *arXiv preprint arXiv:2502.13967*, 2025. 2, 3

[2] Vighnesh Birodkar, Gabriel Barcik, James Lyon, Sergey Ioffe, David Minnen, and Joshua V. Dillon. Sample What You Can't Compress. *arXiv preprint arXiv:2409.02529*, 2024. 2, 3, 8

[3] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2024. 3

[4] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video Generation Models as World Simulators, 2024. 1, 3

[5] Eric R. Chan, Koki Nagano, Matthew A. Chan, Alexander W. Bergman, Jeong Joon Park, Axel Levy, Miika Aittala, Shalini De Mello, Tero Karras, and Gordon Wetzstein. GeNVS: Generative Novel View Synthesis with 3D-Aware Diffusion Models. In *In IEEE Conference on Computer Vision and Pat tern Recognition (ICCV)*, 2023. 5

[6] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. MaskGIT: Masked Generative Image Transformer. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 5, 6

[7] Junyu Chen, Han Cai, Junsong Chen, Enze Xie, Shang Yang, Haotian Tang, Muyang Li, Yao Lu, and Song Han. Deep Compression Autoencoder for Efficient High-Resolution Diffusion Models. *arXiv preprint arXiv:2410.10733*, 2024. 3

[8] Yinbo Chen, Rohit Girdhar, Xiaolong Wang, Sai Saketh Rambhatla, and Ishan Misra. Diffusion Autoencoders are Scalable Image Tokenizers. *arXiv preprint arxiv:2501.18593*, 2025. 3, 6

[9] Kevin Clark, Paul Vicol, Kevin Swersky, and David J Fleet. Directly Fine-Tuning Diffusion Models on Differentiable Rewards. In *International Conference on Learning Representations (ICLR)*, 2024. 3, 4

[10] Katherine Crowson, Stefan Andreas Baumann, Alex Birch, Tanishq Mathew Abraham, Daniel Z Kaplan, and Enrico Shippole. Scalable High-Resolution Pixel-Space Image Synthesis with Hourglass Diffusion Transformers. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024. 2

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (ICCV)*, 2009. 2, 5

[12] 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 *International Conference on Learning Representations (ICLR)*, 2021. 4, 7

[13] 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 (CVPR)*, 2021. 1, 3, 5

[14] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling Rectified Flow Transformers for High-Resolution Image Synthesis. *arXiv preprint arXiv:2403.03206*, 2024. 1, 2, 3, 4, 7, 8

[15] NVIDIA et. al. Cosmos World Foundation Model Platform for Physical AI. *arXiv preprint arXiv:2501.03575*, 2025. 1

[16] Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. DPOK: Reinforcement Learning for Fine-tuning Text-to-Image Diffusion Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 3

[17] Christopher Fifty, Ronald G. Junkins, Dennis Duan, Aniketh Iger, Jerry W. Liu, Ehsan Amid, Sebastian Thrun, and Christopher Ré. Restructuring Vector Quantization with the Rotation Trick. *arXiv preprint arXiv:2410.06424*, 2024. 8

[18] Jiatao Gu, Shuangfei Zhai, Yizhe Zhang, Josh Susskind, and Navdeep Jaitly. Matryoshka Diffusion Models. In *International Conference on Learning Representations (ICLR)*, 2024. 1, 2

[19] 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 (NeurIPS)*, 2018. 5, 6

[20] Jonathan Ho and Tim Salimans. Classifier-Free Diffusion Guidance. *arXiv preprint arXiv:2207.12598*, 2022. 8

[21] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, 2020. 1

[22] Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. Simple diffusion: End-to-end diffusion for high resolution images. *arXiv preprint arXiv:2301.11093*, 2023. 1, 2

[23] Emiel Hoogeboom, Eirikur Agustsson, Fabian Mentzer, Luca Versari, George Toderici, and Lucas Theis. High-Fidelity Image Compression with Score-based Generative Models. *arXiv preprint arXiv:2305.18231*, 2024. 2, 3, 5, 8

[24] Emiel Hoogeboom, Thomas Mensink, Jonathan Heek, Kay Lamerigts, Ruiqi Gao, and Tim Salimans. Simpler Diffusion (SiD2): 1.5 FID on ImageNet512 with pixel-space diffusion. *arXiv preprint arXiv:2410.19324*, 2024. 2, 7

[25] Drew A. Hudson, Daniel Zoran, Mateusz Malinowski, Andrew K. Lampinen, Andrew Jaegle, James L. McClelland, Loic Matthey, Felix Hill, and Alexander Lerchner. SODA: Bottleneck Diffusion Models for Representation Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3

[26] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. *European Conference on Computer Vision (ECCV)*, 2016. 4

[27] Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and Improving the Training Dynamics of Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 7

[28] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 7

[29] Dan Kondratyuk, Lijun Yu, Xiuye Gu, José Lezama, Jonathan Huang, Grant Schindler, Rachel Hornung, Vighnesh Birodkar, Jimmy Yan, Ming-Chang Chiu, Krishna Somandepalli, Hassan Akbari, Yair Alon, Yong Cheng, Josh Dillon, Agrim Gupta, Meera Hahn, Anja Hauth, David Hendon, Alonso Martinez, David Minnen, Mikhail Sirotenko, Kihyuk Sohn, Xuan Yang, Hartwig Adam, Ming-Hsuan Yang, Irfan Essa, Huisheng Wang, David A. Ross, Bryan Seybold, and Lu Jiang. VideoPoet: A Large Language Model for Zero-Shot Video Generation. *arXiv preprint arXiv:2312.14125*, 2024. 1

[30] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved Precision and Recall Metric for Assessing Generative Models. In *Advances in Neural Information Processing Systems*, 2019. 6

[31] 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. *arXiv preprint arXiv:2404.07724*, 2024. 8

[32] Black Forest Labs. Flux. `https://github.com/black-forest-labs/flux`, 2024. 4

[33] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *International Conference on Learning Representations (ICLR)*, 2023. 1, 2, 3, 5

[34] 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. 1, 2, 3, 4

[35] Zhuoyan Luo, Fengyuan Shi, Yixiao Ge, Yujiu Yang, Limin Wang, and Ying Shan. Open-MAGVIT2: An Open-Source Project Toward Democratizing Auto-regressive Visual Generation. *arXiv preprint arXiv:2409.04410*, 2025. 1, 3, 5, 6

[36] Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite Scalar Quantization: VQ-VAE Made Simple. *arXiv preprint arXiv:2309.15505*, 2023. 4, 8

[37] Charlie Nash, Jacob Menick, Sander Dieleman, and Peter W. Battaglia. Generating Images with Sparse Representations. *arXiv preprint arXiv:2103.03841*, 2021. 6

[38] OpenAI. DALL-E 3. OpenAI Labs, 2023. A text-to-image AI model developed by OpenAI, capable of generating high-quality images from detailed prompts. https://openai.com/dall-e-3. 3, 8

[39] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *arXiv preprint arXiv:1912.01703*, 2019. 7

[40] William Peebles and Saining Xie. Scalable Diffusion Models with Transformers. *International Conference on Computer Vision (ICCV)*, 2023. 4

[41] Mihir Prabhudesai, Anirudh Goyal, Deepak Pathak, and Katerina Fragkiadaki. Aligning Text-to-Image Diffusion Models with Reward Backpropagation. *arXiv preprint arXiv:2310.03739*, 2023. 3, 4

[42] Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion Autoencoders: Toward a Meaningful and Decodable Representation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3, 4

[43] 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 (CVPR)*, 2022. 3

[44] 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. *arXiv preprint arXiv:2205.11487*, 2022. 1

[45] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved Techniques for Training GANs. In *Advances in Neural Information Processing Systems*, 2016. 6

[46] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 2015. 1

[47] Yang Song and Stefano Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 1

[48] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive Model Beats Diffusion: Llama for Scalable Image Generation. *arXiv preprint arXiv:2406.06525*, 2024. 3, 5, 6

[49] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*, 2023. 3

[50] Hung-Yu Tseng, Lu Jiang, Ce Liu, Ming-Hsuan Yang, and Weilong Yang. Regularing generative adversarial networks under limited data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 3

[51] Mark Weber, Lijun Yu, Qihang Yu, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. MaskBit: Embedding-free Image Generation via Bit Tokens. *arXiv prepprint arXiv:2409.16211*, 2024. 1, 3, 5, 6

[52] Yilun Xu, Gabriele Corso, Tommi Jaakkola, Arash Vahdat, and Karsten Kreis. Disco-diff: Enhancing continuous diffusion models with discrete latents. In *International Conference on Machine Learning*, 2024. 3

[53] Greg Yang, Edward J. Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor Programs V: Tuning Large Neural Networks via Zero-Shot Hyperparameter Transfer. *arXiv preprint arXiv:2203.03466*, 2022. 3, 4, 6

[54] Ruihan Yang and Stephan Mandt. Lossy Image Compression with Conditional Diffusion Models. In *Advances in Neural Information Processing Systems (NeurIPS)*. Curran Associates, Inc., 2023. 2, 3, 4, 8

[55] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized Image Modeling with Improved VQGAN. In *International Conference on Learning Representations (ICLR)*, 2022. 2, 6

[56] Lijun Yu, José Lezama, Nitesh B. Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Vighnesh Birodkar, 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 *International Conference on Learning Representations (ICLR)*, 2024. 1, 3, 4, 5, 8

[57] Qihang Yu, Mark Weber, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. An Image is Worth 32 Tokens for Reconstruction and Generation. *Neural Information Processing Systems (NeurIPS)*, 2024. 1, 2, 3, 5, 6

[58] 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 (CVPR)*, 2018. 5

[59] Long Zhao, Sanghyun Woo, Ziyu Wan, Yandong Li, Han Zhang, Boqing Gong, Hartwig Adam, Xuhui Jia, and Ting Liu. Epsilon-vae: Denoising as visual decoding, 2025. 3

[60] Yue Zhao, Yuanjun Xiong, and Philipp Krähenbühl. Image and Video Tokenization with Binary Spherical Quantization. *arXiv preprint arXiv:2406.07548*, 2024. 8