LATENT DENOISING MAKES GOOD TOKENIZERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite their fundamental role, it remains unclear what properties could make tokenizers more effective for generative modeling. We observe that modern generative models share a conceptually similar training objective—reconstructing clean signals from corrupted inputs, such as signals degraded by Gaussian noise or masking—a process we term *denoising*. Motivated by this insight, we propose aligning tokenizer embeddings directly with the downstream denoising objective, encouraging latent embeddings that remain reconstructable even under significant corruption. To achieve this, we introduce the Latent Denoising Tokenizer (l-DeTok), a simple yet highly effective tokenizer trained to reconstruct clean images from latent embeddings corrupted via interpolative noise or random masking. Extensive experiments on class-conditioned (ImageNet 256×256 and 512×512) and text-conditioned (MSCOCO) image generation benchmarks demonstrate that our l-DeTok consistently improves generation quality across *six* representative generative models compared to prior tokenizers. Our findings highlight denoising as a fundamental design principle for tokenizer development, and we hope it could motivate new perspectives for future tokenizer design. Our code is included in the supplement and will be open source to facilitate reproducibility. Anonymous project page at: link.

1 INTRODUCTION

Modern visual generative models commonly operate on compact *latent embeddings*, produced by tokenizers, to circumvent the prohibitive complexity of pixel-level modeling (Rombach et al., 2022; Peebles & Xie, 2023b; Chang et al., 2022; Li et al., 2024a). Current tokenizers are typically trained as standard variational autoencoders (Kingma & Welling, 2014), primarily optimizing for pixel-level reconstruction. Despite their critical influence on downstream generative quality, it remains unclear what properties enable more effective tokenizers for generation. As a result, tokenizer development has lagged behind recent rapid advances in generative model architectures.

In this work, we ask: *What properties can make visual tokenizers more effective for generative modeling?* We observe that modern generative models, despite methodological differences, share a conceptually similar training objective—reconstructing original signals from corrupted ones. For instance, diffusion models remove diffusion-induced noise to recover clean signals (Ho et al., 2020; Peebles & Xie, 2023b), while autoregressive models reconstruct complete sequences from partially observed contexts (Li et al., 2024a; Chang et al., 2022), analogous to removing "masking noise" (He et al., 2022; Chen et al., 2025c; Devlin et al., 2019). We collectively refer to these reconstruction-from-deconstruction[1] processes as *denoising*.

This unified *denoising* perspective of modern generative models suggests that effective visual tokenizers for these models should produce latent embeddings that are reconstructable even under significant corruption. Such embeddings naturally align with the denoising objectives of downstream generative models, facilitating their training and subsequently enhancing their generation quality.

Motivated by this insight, we propose to train tokenizers as latent denoising autoencoders, termed as l-DeTok. Specifically, we corrupt latent embeddings via *interpolative noise*, generated by interpolating original embeddings with Gaussian noise. The tokenizer decoder is then trained to reconstruct clean images from these heavily noised latent embeddings. Additionally, we explore random masking, akin to masked autoencoders (MAE) (He et al., 2022), as an alternative and optional form of deconstruction and find it similarly effective.

[1] We use the terms "deconstruction" and "corruption" interchangeably throughout the paper.

Conceptually, these deconstruction-reconstruction strategies encourage latent embeddings to be robust, stable, and easily reconstructable under strong corruption, aligning with the downstream denoising tasks central to generative models. Indeed, our experiments show that stronger noise used in our l-DeTok training typically yields better downstream generative performance.

We demonstrate the effectiveness and generalizability of l-DeTok across *six representative generative models*, including non-autoregressive (DiT (Peebles & Xie, 2023b), SiT (Ma et al., 2024), LightningDiT (Yao & Wang, 2025)) and autoregressive models (MAR (Li et al., 2024a), RasterAR, RandomAR (Li et al., 2024a; Pang et al., 2025; Yu et al., 2024)) on the ImageNet generation benchmark. By adopting our tokenizer—without modifying model architectures—we push the limits for MAR models (Li et al., 2024a), improving FID from 2.31 to 1.55 for MAR-B, matching the performance of the original huge-sized MAR (1.55). For MAR-L, FID improves from 1.78 to 1.35. Importantly, these gains come *without* semantics distillation, thus avoiding reliance on visual encoders pretrained at a far larger scale (Oquab et al., 2024; Radford et al., 2021).

In summary, our work demonstrates a simple yet crucial insight: explicitly incorporating denoising objectives into tokenizer training significantly enhances their effectiveness for generative modeling since it is downstream task-aligned. We hope this perspective will stimulate new research directions in tokenizer design and accelerate future advances in generative modeling.

2 Related Work

Representation learning in visual recognition. Representation learning has been a decades-long pursuit in visual recognition, aiming to obtain transferable embeddings that generalize across diverse downstream tasks (Bengio et al., 2013). At the core of these approaches lies a foundational principle: pre-training should encourage representations to encode the information most relevant to downstream tasks. This principle has inspired the design of diverse and effective pretext tasks, such as instance discrimination (He et al., 2020; Chen et al., 2020b), self-distillation (Grill et al., 2020; Caron et al., 2021; Oquab et al., 2024), and masked-image reconstruction (He et al., 2022; Oquab et al., 2024; Zhou et al., 2022), each aligning representations with downstream utility. These insights motivate us to explore tokenizer embeddings that align with downstream generative tasks.

Visual tokenizers for generative modeling. Modern generative models typically rely on tokenizers to encode images into compact latent embeddings, significantly reducing computational complexity compared to pixel-level modeling. While conventional tokenizers optimize pixel reconstruction with KL-regularization (Kullback & Leibler, 1951), recent approaches (Chen et al., 2025a; Yao & Wang, 2025; Chen et al., 2025b; Li et al., 2025) have advocated semantics distillation from powerful pretrained vision models (Oquab et al., 2024; Radford et al., 2021). Such semantics-distilled tokenizers rely inherently on a two-stage pipeline: first training a vision encoder at a significantly larger scale in terms of compute and data, then distilling its features into latent embeddings. Yet, in many domains (*e.g.*, video, audio, 3D/4D), pre-trained models suitable for distillation may not exist. In contrast, our l-DeTok does not rely on these dependencies to achieve promising results.

Generative modeling frameworks. Generative modeling frameworks broadly include autoregressive (AR) (Chen et al., 2020a; Esser et al., 2021; Chang et al., 2022; Li et al., 2024a; Tian et al., 2025; Pang et al., 2025; Yu et al., 2024) and diffusion-based non-autoregressive (non-AR) methods (Ho et al., 2020; Rombach et al., 2022; Peebles & Xie, 2023b; Ma et al., 2024; Yao & Wang, 2025). AR models predict latent tokens sequentially conditioned on partial contexts, while non-AR models jointly generate tokens via iterative refinement, typically using diffusion (Ho et al., 2020; Sohl-Dickstein et al., 2015) or flow-based processes (Lipman et al., 2023; Esser et al., 2024). Despite methodological differences, both paradigms critically rely on high-quality latent embeddings from upstream tokenizers (Hansen-Estruch et al., 2025). Thus, improving tokenizer embeddings is fundamental for enhancing generative performance across diverse frameworks.

3 Method

Our goal is to design visual tokenizers that are more effective for generative modeling compared to standard autoencoders trained for pixel reconstruction. This section first revisits the core training objective shared by all modern generative models, *i.e.*, *denoising*, to motivate our design, and then details our latent denoising tokenizers (l-DeTok).

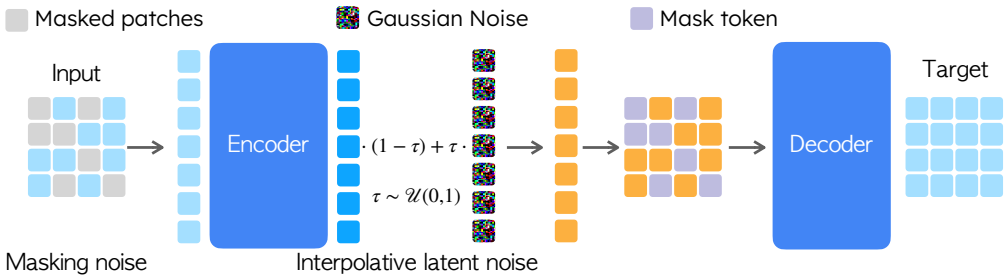


Figure 1: **Our latent denoising tokenizers (l-DeTok) framework.** During tokenizer training, we randomly mask input patches (*masking noise*) and interpolate encoder-produced latent embeddings with Gaussian noise (*interpolative latent noise*). The decoder processes these deconstructed latents and mask tokens to reconstruct the original images in pixels. We refer to this process as *denoising*. When serving as a tokenizer for downstream generative models, both noises are disabled.

3.1 Preliminaries of Generative Modeling

Modern generative frameworks can be primarily divided into non-autoregressive (non-AR) and autoregressive (AR) paradigms. Despite their methodological differences, both paradigms aim to gradually reconstruct the original representations from deconstructed ones.

Non-autoregressive generative models. Non-autoregressive models, exemplified by diffusion (Ho et al., 2020; Peebles & Xie, 2023b) and flow-matching methods (Lipman et al., 2023), learn to iteratively refine latent representations deconstructed by controlled noise. Given a latent representation of input $\mathbf{X}_0$, the forward noising process progressively corrupts these latents into $\mathbf{X}_t$:

$$\mathbf{X}_t = a(t)\,\mathbf{X}_0 + b(t)\,\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{1}$$

where $a(t)$ and $b(t)$ are noise schedules. Generative models are trained to revert this deconstruction:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{X},\boldsymbol{\epsilon},t}\Big[\|\epsilon_{\boldsymbol{\theta}}(\mathbf{X}_t, t) - \boldsymbol{\epsilon}_t\|^2\Big], \tag{2}$$

where $\epsilon_{\boldsymbol{\theta}}$ is a learnable noise estimator parameterized by $\boldsymbol{\theta}$. Essentially, non-AR diffusion models learn to *reconstruct original latents from intermediate latents deconstructed by noise*.

Autoregressive generative models. Autoregressive approaches factorize image generation into a sequential prediction problem. Given an ordered sequence of latent tokens $\{\mathbf{x}^1, \ldots, \mathbf{x}^N\}$, AR methods factorize the joint distribution as:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \prod_{i=1}^{N} p_{\boldsymbol{\theta}}(\mathbf{x}^i | \mathbf{x}^1, \ldots, \mathbf{x}^{i-1}), \tag{3}$$

where $\mathbf{x}^i$ denotes the latent tokens generated at step i. Recent generalized AR variants extend this framework to arbitrary generation orders (Li et al., 2024a; Yu et al., 2024; Pang et al., 2025) or set-wise prediction strategies (Tian et al., 2025; Ren et al., 2025). Nonetheless, the fundamental training objective remains consistent: reconstructing full sequences from partially observed—or equivalently, partially *masked*—contexts. In other words, AR models learn to *reconstruct original latents from intermediate latents deconstructed by masking*.

3.2 Latent Denoising Tokenizers

Motivated by the discussions above, we propose latent denoising tokenizer (l-DeTok), a simple tokenizer trained by reconstructing original images from deconstructed latent representations. This deconstruction-reconstruction design aligns with the denoising tasks employed by modern generative models. Figure 1 shows an overview of our method. We detail each component next.

Overview. Our tokenizer follows an encoder-decoder architecture based on Vision Transformers (ViT) (Dosovitskiy et al., 2021). Input images are divided into non-overlapping patches, linearly projected into embedding vectors, and added with positional embeddings. During training, we deconstruct these embeddings using two complementary strategies: (i) injecting noise in latent embeddings and (ii) randomly masking image patches. The decoder reconstructs original images from these deconstructed embeddings. This strategy encourages easy-to-reconstruct latent embeddings under heavy corruption, aiming to simplify downstream denoising tasks in generative models.

Noising as deconstruction. Our core idea is to deconstruct latent embeddings by noise. Specifically, given latent embeddings $\mathbf{x}$ from the encoder, we *interpolate* them with Gaussian noise as follows:

$$\mathbf{x}' = (1-\tau)\mathbf{x} + \tau\boldsymbol{\varepsilon}(\gamma), \quad \text{where} \quad \boldsymbol{\varepsilon}(\gamma) \sim \gamma \cdot \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad \tau \sim \mathcal{U}(0,1). \tag{4}$$

Here, the scalar γ controls noise standard deviation, and the factor τ specifies noise level. Critically, this *interpolative* strategy differs from the conventional additive noise, *i.e.*, $\mathbf{x}' = \mathbf{x} + \tau\boldsymbol{\varepsilon}$, employed by standard VAEs (Pu et al., 2016), as it ensures latents can be effectively and heavily corrupted when the noise level τ is high. Moreover, random sampling of τ encourages latents to remain robust across diverse corruption levels. At inference time, latent noising is disabled ($\tau = 0$).

Masking as deconstruction. We further generalize our denoising perspective by interpreting masking as another form of latent deconstruction (Chen et al., 2025c). Inspired by masked autoencoders (MAE) (He et al., 2022), we randomly mask a subset of image patches. Different from MAE, we use a random masking ratio. Concretely, given an input image partitioned into patches, we mask a random subset, where the masking ratio m is sampled from a slightly biased uniform distribution:

$$m = \max(0, \mathcal{U}(-0.1, M)), \tag{5}$$

where $\mathcal{U}(-0.1, M)$ denotes a uniform distribution on $[-0.1, M]$. The slight bias towards zero reduces the distribution gap between training and inference (no masking). The encoder processes only the visible patches, and masked positions are represented by shared learnable `[MASK]` tokens at the decoder input. At inference time, all patches are visible ($m = 0$).

Training objectives. Our tokenizer is trained to reconstruct the original images from corrupted latent embeddings. The training objective follows established practice (Rombach et al., 2022; Esser et al., 2021; Yu et al., 2025a), combining pixel-wise mean-squared-error (MSE), latent-space KL-regularization (Pu et al., 2016), perceptual losses (VGG- and ConvNeXt-based as in Yu et al. (2025a)), and an adversarial GAN objective (Goodfellow et al., 2014):

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{MSE}} + \lambda_{\text{KL}}\mathcal{L}_{\text{KL}} + \lambda_{\text{percep}}\mathcal{L}_{\text{percep}} + \lambda_{\text{GAN}}\mathcal{L}_{\text{GAN}}, \tag{6}$$

where each λ controls the contribution of the corresponding loss component.

4 Implementation

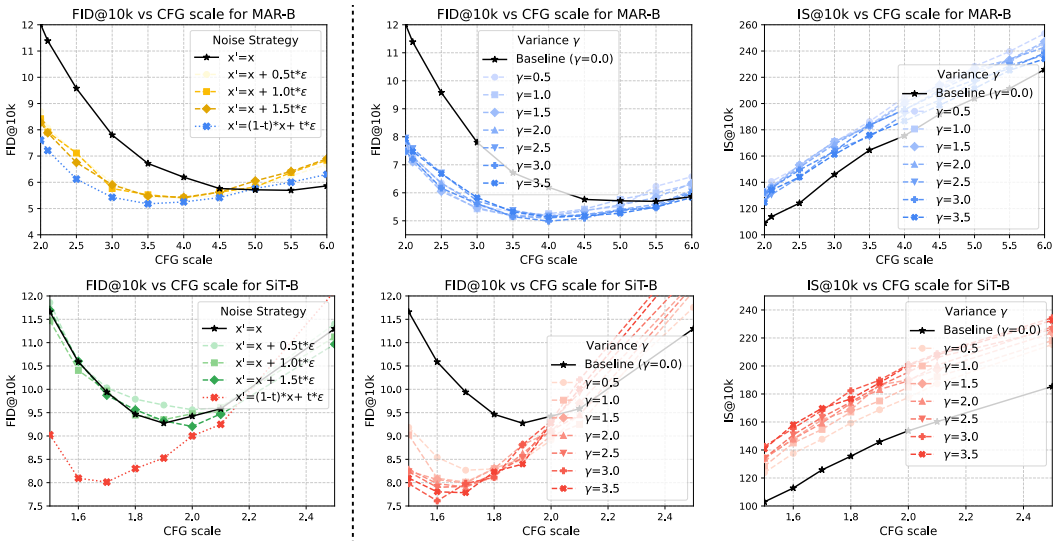
We briefly summarize implementation details here, including datasets, evaluation metrics, and model training procedures. Due to limited space, we defer comprehensive details to Sections A.1 and A.2.

Dataset and metrics. We primarily experiment with class-conditioned image generation on the ImageNet dataset (Russakovsky et al., 2015) at 256×256 and 512×512 resolutions, and further evaluate text-to-image generation on the MS-COCO dataset (Lin et al., 2014). During training, images are center-cropped, randomly horizontally flipped, and normalized to $[-1, 1]$. Following Dhariwal & Nichol (2021), generative models are evaluated using Fréchet Inception Distance (FID), Inception Score (IS), precision, and recall. For reconstruction quality, we report rFID and rPSNR.

Tokenizer baselines. We benchmark our tokenizer against a diverse set of tokenizers: (i) MAR-VAE from MAR (Li et al., 2024a), trained on ImageNet using the implementation from Esser et al. (2021); (ii) VA-VAE (Yao & Wang, 2025), aligning latent embeddings with DINOv2 features; (iii) MAETok (Chen et al., 2025a), distilling HOG, DINOv2 (Oquab et al., 2024), and CLIP (Radford et al., 2021) features through auxiliary decoders; (iv) SD-VAE from Stable-Diffusion (Rombach et al., 2022), trained on significantly larger datasets. Additionally, for controlled comparisons, we also train our own baseline tokenizer without the proposed denoising approaches as a baseline.

Our tokenizer implementation. We implement our tokenizer using ViTs for both encoder and decoder (Vaswani et al., 2017; Dosovitskiy et al., 2021). We adopt recent architectural advances from LLaMA (Touvron et al., 2023), including RoPE (Su et al., 2024) (together with learned positional embeddings following Fang et al. (2023)), RMSNorm (Zhang & Sennrich, 2019), and SwiGLU-FFN (Shazeer, 2020). We use a patch size of 16, and the latent dimension is set to 16.

Tokenizer training. We set the default weights in Eq. 6 to $\lambda_{\text{KL}} = 10^{-6}$, $\lambda_{\text{percep}} = 1.0$, and $\lambda_{\text{GAN}} = 0.1$. In ablation studies, we use ViT-S for the encoder and ViT-B for the decoder, disable the GAN loss, and train for 50 epochs. For final experiments, we use ViT-B for both encoder and decoder, train for 200 epochs, and activate the GAN loss starting from epoch 100.



(a) Additive *vs.* interpolative (b) Latent noise variance

Figure 2: **Ablation on latent noise design. (a) Additive *vs.* interpolative noise.** Interpolative noise clearly outperforms additive noise for both MAR (Li et al., 2024a) and SiT (Ma et al., 2024). Both interpolative and additive latent noise lead to improved performance for MAR. **(b) Latent noise standard deviation (γ).** Our l-DeTok remains robust across various noise standard deviations. Generally, increasing γ improves generation quality, with best results achieved around $\gamma = 3.0$.

Generative models. To evaluate the broad effectiveness of a tokenizer, we experiment with *six* representative generative models, including three non-autoregressive models: DiT (Peebles & Xie, 2023b), SiT (Ma et al., 2024), and LightningDiT (Yao & Wang, 2025); and three autoregressive models: MAR (Li et al., 2024a), causal RandomAR, and RasterAR based on RAR (Yu et al., 2024) and *diffloss* (Li et al., 2024a). We follow the officially released implementations to reimplement all methods within a unified codebase, standardizing training and evaluation across methods. Previous works on visual tokenizers often exclusively evaluate on non-AR models (*e.g.*, DiT, SiT), yet we find improvements from non-AR models *do not* necessarily generalize to AR models (more on this later). Our experiments aim to provide useful data points toward a more universal tokenizer.

Generative model training. We use a standardized training recipe for all generative models. Specifically, we follow the hyperparameters from Yao & Wang (2025), training generative models with a global batch size of 1024, using AdamW (Loshchilov & Hutter, 2019) with a constant learning rate of 2×10^{-4}, without warm-up, gradient clipping, or weight decay. Autoregressive models adopt the three-layer 1024-channel *diffloss* MLP from Li et al. (2024a). For ablation studies, we train generative models for 100 epochs. For larger-scale experiments on MAR models, we train for 800 epochs.

5 Experiments

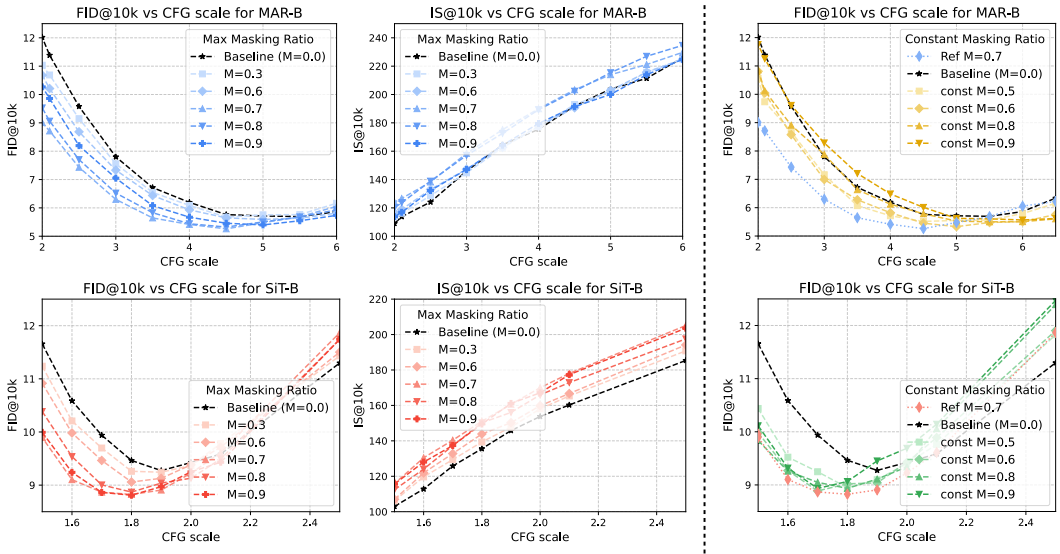
5.1 Main Properties

We use SiT-B (Ma et al., 2024) and MAR-B (Li et al., 2024a) as representative non-autoregressive (non-AR) and autoregressive (AR) generative models to study the generalizability of a tokenizer. Quantitative reconstruction results for tokenizers trained under different noises are in Sec. B.1.

5.1.1 Properties of Latent Noising

We first study latent noising as the sole form of deconstruction, without any masking.

Interpolative *vs.* additive noise. An important design choice in our l-DeTok is the use of interpolative latent noise instead of additive noise, which we ablate here. Specifically, we compare two latent noising variants: interpolative noise, *i.e.*, $\mathbf{x}' = (1-\tau)\mathbf{x} + \tau\boldsymbol{\varepsilon}$ (Eq. 4), and additive noise, *i.e.*, $\mathbf{x}' = \mathbf{x} + \tau\boldsymbol{\varepsilon}$. We set the noise standard deviation to $\gamma = 1.0$ here. Figure 2-(a) presents the results: Interpolative noise clearly outperforms additive noise for both SiT (Ma et al., 2024) and MAR (Li et al., 2024a). This aligns with our expectation: interpolative noise ensures latent embed-



(a) Random masking ratio (b) Constant masking ratio

Figure 3: **Ablation on masking ratio.** We show generative performance (FID↓) with varying masking ratios for tokenizers trained with **(a)** random and **(b)** constant masking ratio. Both MAR (Li et al., 2024b) (top) and SiT (Ma et al., 2024) (bottom) benefit from masking-based tokenizers, favoring heavy masking (70% to 90%). Randomized masking consistently outperforms constant masking.

dings are heavily corrupted at high noise levels. In contrast, additive noise can potentially create shortcuts by making original signals remain dominant, reducing the effectiveness of the additive noise. Nonetheless, we observe the additive latent noise still improves MAR but not SiT.

Noise standard deviation. Figure 2-(b) studies the effect of noise standard deviation (γ in Eq. 4). Both SiT and MAR consistently improve with interpolative latent noise across all tested levels. Performance peaks at moderately high standard deviation, indicating that stronger corruption generally yields more effective latents. This result confirms our key hypothesis: challenging *denoising* tasks naturally produce robust, downstream-aligned latents that benefit generative modeling.

5.1.2 Properties of Masking

We next investigate masking noise independently, without any latent noise.

Masking ratio. We examine how varying the maximal masking ratio M (Eq. 5) influences generation quality. As shown in Figure 3-(a), both SiT and MAR benefit from masking-based tokenizer training in generation quality. Masking ratios between 70% and 90% consistently yield stronger performance compared to low masking ratios (*e.g.*, 30%), favoring high degrees of masking. This behavior mirrors observations from latent denoising, *i.e.*, challenging denoising is more beneficial, indicating a common underlying principle under the *deconstruction-reconstruction* strategy.

Constant *vs.* randomized masking ratio. Figure 3-(b) compares randomized masking ratios against constant ones. For constant ratios, we further fine-tune the tokenizer decoder for 10 epochs using *fully-visible* latents to alleviate the distribution shift between training and inference since fully-visible inputs are absent in constant-ratio training. This adjustment improves rPSNR by 1–3 and rFID by 0.6 (details in Sec. B.1). Figure 3-(b) shows randomized masking consistently outperforms constant masking, as it encourages latent embeddings to be robust to varying corruption levels. This aligns naturally with downstream tasks that require denoising across diverse corruption levels.

5.1.3 Joint Denoising

Prior ablations indicate that both latent noising and masking can independently improve generation quality, with latent noising showing a stronger effect. Here, we investigate the effect of joint denoising. Based on prior results, we fix the noise standard deviation to $\gamma = 3.0$ and the masking ratio to $M = 0.7$. Figure 4 and Table 1 summarize the results. With joint denoising, our l-DeTok achieves FID scores of 5.50 (SiT-B) and 2.65 (MAR-B) with CFG (Table 1). In comparison, our baseline

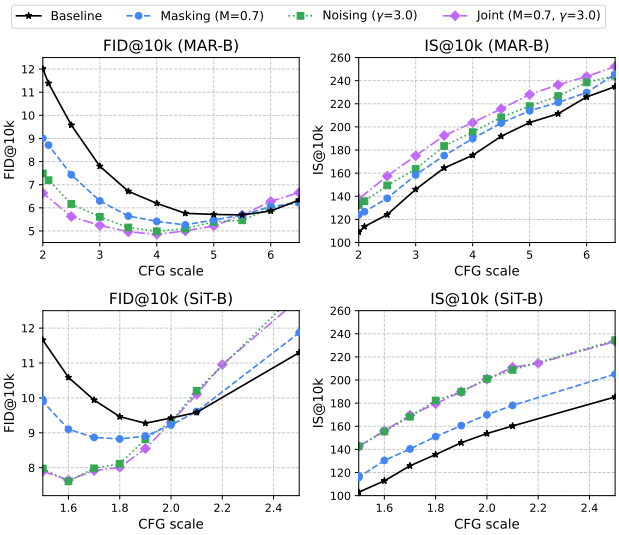


Figure 4: **Impact of tokenizer training strategies on generative performance.** We compare our baseline tokenizer with our l-DeTok variants: masking-only (M=0.7), latent noising-only (γ=3.0), and their combination (joint noising). Both masking and latent noising independently improve generation quality, with latent noising showing a stronger effect. Joint noising further improves performance for MAR, particularly in inception scores (IS), but provides limited additional benefit for SiT when latent noising is already applied. FID@50k scores are detailed in Tab. 1.

Table 1: **Effectiveness of denoising**. We report FID and IS evaluated on 50,000 images here. Compared to baselines, we see substantial gains in generative models when using our l-DeTok. Extended: larger encoder, longer training, GAN enabled midway through training.

Setup	MAR-B			SiT-B		
	w/ CFG		w/o CFG	w/ CFG		w/o CFG
	FID↓	IS↑	FID↓	FID↓	IS↑	FID↓
Baseline	3.31	247.63	16.70	6.97	181.61	26.82
Masking only	2.90	243.00	12.57	6.43	189.24	22.50
Latent noise only	2.77	249.02	8.82	5.56	193.52	16.11
Joint noise	2.65	263.03	7.48	5.50	195.07	16.03
+Extended	**2.43**	**266.50**	**6.72**	**5.13**	**207.35**	**14.81**

tokenizer—trained with identical settings but without any noise—obtains significantly worse results: 6.97 (SiT-B) and 3.31 (MAR-B). We observe that joint denoising is more effective for MAR, but provides limited additional benefit for SiT when latent noising is already applied. This indicates that latent denoising is essential, while the masking-based denoising is *optional*.

Lastly, with joint denoising, we increase the encoder to base size, train for 200 epochs, and enable the GAN loss starting from epoch 100 (+Extended in Table 1). Under this setting, FID improves to 5.13 (SiT) and 2.43 (MAR). We adopt this improved tokenizer for all subsequent evaluations.

5.2 Generalization Experiments

To comprehensively evaluate tokenizer generalizability, we compare performance across six representative generative models: three non-autoregressive (DiT (Peebles & Xie, 2023b), SiT (Ma et al., 2024), LightningDiT (Yao & Wang, 2025)) and three autoregressive (MAR (Li et al., 2024a), RandomAR, RasterAR). RandomAR and RasterAR are adapted from RAR (Yu et al., 2024). We modify them to support decoding continuous tokens via *diffloss* MLPs (Li et al., 2024a). RandomAR generates tokens in a random order while RasterAR uses a raster-scan order. We use base-sized models and train them for 100 epochs for experiments here.

Comparisons with standard convolutional tokenizers. Our l-DeTok tokenizer consistently outperforms conventional tokenizers (Rombach et al., 2022; Esser et al., 2021; Peebles & Xie, 2023a), by large margins across both AR and non-AR models. Table 2 presents the results. Compared to the best existing tokenizer (MAR-VAE), our method significantly improves FID (with CFG) from 3.71 to 2.43 (~34%) for MAR, from 11.78 to 5.22 (~56%) for RandomAR, and from 7.99 to 4.46 (~44%) for RasterAR. Improvements for non-autoregressive models are also consistent.

Comparisons with semantics-distilled tokenizers. Our l-DeTok generalizes significantly better than prior semantics-distilled tokenizers. Table 2 compares our method with recent approaches such as VA-VAE (Yao & Wang, 2025) and MAETok (Chen et al., 2025a), which distill semantics from pretrained encoders. *Surprisingly*, we empirically find these tokenizers, despite promising performance in non-AR models, do not generalize well to AR models. Previous studies implicitly assume that tokenizer improvements from one generative paradigm naturally transfer to others. However, our experiments *challenge* this assumption, revealing a previously unrecognized gap: tokenizer effectiveness in one paradigm *does not* necessarily transfer to others.

Table 2: **Generalizability comparison of tokenizers across different generative models.** We compare various tokenizers on representative generative models. Our l-DeTok tokenizer outperforms other tokenizers for AR models, and also surpasses standard tokenizers trained without semantics distillation for non-AR models. All results are obtained with optimal CFG scales.

Tokenizer	rFID↓	Autoregressive Models						Non-autoregressive Models					
		MAR		RandomAR		RasterAR		SiT		DiT		Light.DiT	
		FID↓	IS↑	FID↓	IS↑	FID↓	IS↑	FID↓	IS↑	FID↓	IS↑	FID↓	IS↑
Tokenizers trained with semantics distillation from external pretrained models													
VA-VAE (Yao & Wang, 2025)	**0.28**	16.66	144.5	38.13	68.3	15.88	160.5	4.33	221.1	4.91	213.9	2.86	**275.1**
MAETok (Chen et al., 2025a)	0.48	6.99	201.8	24.83	97.6	15.92	127.2	4.77	**243.2**	5.24	**224.7**	3.92	273.3
Our l-DeTok + Distillation	0.85	2.52	254.1	5.57	180.3	11.99	158.8	**3.40**	232.6	**3.91**	221.7	**2.18**	243.0
Tokenizers trained without semantics distillation													
SD-VAE (Rombach et al., 2022)	0.61	4.64	259.8	13.11	141.8	8.26	179.3	7.66	187.5	8.33	179.8	4.24	223.7
MAR-VAE (Li et al., 2024a)	0.53	3.71	265.3	11.78	147.9	7.99	189.7	6.26	177.5	8.20	171.8	3.98	218.7
Our l-DeTok	0.68	**2.43**	**266.5**	**5.22**	**248.9**	**4.46**	**257.7**	5.13	207.3	6.58	173.9	3.63	225.4

In sharp contrast, our method generalizes robustly across both non-AR and AR models. Critically, our l-DeTok achieves this *without* any semantics distillation from pretrained encoders. To further investigate *whether* our l-DeTok can benefit from semantics distillation, we incorporate an auxiliary semantics-distillation loss (details in Sec. A.4). Remarkably, this *privileged* version of our tokenizer achieves the best FID scores for non-AR models, surpassing *all* previous semantics-distilled tokenizers. Although AR models benefit relatively less from distillation and may even slightly degrade, their performance still remains substantially superior to previous methods.

l-DeTok with convolutional tokenizers. Our l-DeTok generalizes across Transformer- and convolution-based tokenizers, as the idea of *latent denoising* is architecture-agnostic. To demonstrate this, we train CNN-based tokenizers for 50 epochs with and without our denoising objective (details in Sec. A.5). With the default CNN baseline, MAR-B achieves 3.32 FID and SiT-B 7.11 FID at best CFGs. With our denoising-based CNN tokenizers, MAR-B improves to 2.82 FID and SiT-B to 5.62 FID, demonstrating that l-DeTok provides consistent gains also in convolutional networks.

5.3 Benchmarking with Previous Systems

We compare against leading generative systems in Table 3. For this experiment, we train MAR-B and MAR-L for 800 epochs. Simply adopting our tokenizer substantially improves the generative performance: MAR-B achieves an FID of 1.55 (from 2.31), and MAR-L further improves to 1.35 (from 1.78). Notably, our MAR-B and MAR-L both match or surpass the previously best-performing huge-size MAR model (1.35 *vs.* 1.55). Qualitative results are provided in Figure 5. We also report

Table 3: **System-level comparison** on ImageNet 256×256 class-conditioned generation. Our approach enables MAR models (Li et al., 2024a) to achieve leading results without relying on semantics distillation. †: With additional decoder fine-tuning (see Sec. A.3 for details).

	#params	w/o CFG				w/ CFG			
		FID↓	IS↑	Pre.↑	Rec.↑	FID↓	IS↑	Pre.↑	Rec.↑
With semantics distillation from external pretrained models									
SiT-XL + REPA (Yu et al., 2025b)	675M	5.90	157.8	0.70	0.69	1.42	305.7	0.80	0.64
SiT-XL + MAETok (Chen et al., 2025a)	675M	2.31	**216.5**	-	-	1.67	**311.2**	-	-
LightningDiT + MAETok (Chen et al., 2025a)	675M	2.21	208.3	-	-	1.73	308.4	-	-
LightningDiT + VAVAE (Yao & Wang, 2025)	675M	**2.17**	205.6	0.77	0.65	1.35	295.3	0.79	0.65
DDT-XL (Wang et al., 2025)	675M	6.27	154.7	0.68	0.69	**1.26**	310.6	0.79	0.65
Without semantics distillation from external pretrained models									
DiT-XL/2 (Peebles & Xie, 2023a)	675M	9.62	121.5	0.67	0.67	2.27	278.2	0.83	0.57
SiT-XL/2 (Ma et al., 2024)	675M	8.30	-	-	-	2.06	270.3	0.82	0.59
VAR-d30 (Tian et al., 2025)	2.0B	-	-	-	-	1.92	**323.1**	0.82	0.59
LlamaGen-3B (Sun et al., 2024)	3.1B	-	-	-	-	2.18	263.3	0.81	0.58
RandAR-XXL (Pang et al., 2025)	1.4B	-	-	-	-	2.15	322.0	0.79	0.62
CausalFusion (Deng et al., 2024)	676M	3.61	180.9	0.75	0.66	1.77	282.3	0.82	0.61
MAR-B + MAR-VAE (Li et al., 2024a)	208M	3.48	192.4	0.78	0.58	2.31	281.7	0.82	0.57
MAR-L + MAR-VAE (Li et al., 2024a)	479M	2.60	221.4	0.79	0.60	1.78	296.0	0.81	0.60
MAR-H + MAR-VAE (Li et al., 2024a)	943M	2.35	227.8	0.79	0.62	1.55	303.7	0.81	0.62
MAR-B (Li et al., 2024a) + our l-DeTok	208M	2.79	195.9	0.80	0.60	1.61	289.7	0.81	0.62
MAR-B (Li et al., 2024a) + our l-DeTok†	208M	2.94	195.5	0.80	0.59	1.55	291.0	0.81	0.62
MAR-L (Li et al., 2024a) + our l-DeTok	479M	**1.84**	238.4	0.82	0.60	1.43	303.5	0.82	0.61
MAR-L (Li et al., 2024a) + our l-DeTok†	479M	1.86	**238.6**	0.82	0.61	**1.35**	304.1	0.81	0.62

Figure 5: **Qualitative Results.** We show selected examples of class-conditional generation on ImageNet 256×256 using MAR-L (Li et al., 2024a) trained with our tokenizer.

results on ImageNet 512 × 512 in Table 4. Our MAR-B variant already matches MAR-L, a 2.3× larger model, when trained from scratch. Our MAR-L achieves remarkable 1.61 FID and 315.7 IS.

Model	#params	FID↓	IS↑
ADM (Dhariwal & Nichol, 2021)	554M	7.72	172.7
DiT-XL/2 (Peebles & Xie, 2023b)	675M	3.04	240.8
SiT-XL/2 (Ma et al., 2024)	675M	2.62	252.2
SiT-XL/2 + REPA (Yu et al., 2025b)	675M	2.08	274.6
MAR-L + MAR-VAE (Li et al., 2024a)	479M	1.73	279.9
MAR-B + Ours (scratch, 400ep)	208M	1.83	279.6
MAR-L + Ours (fine-tune, 200ep)	479M	**1.61**	**315.7**

Table 4: **System-level comparison** on ImageNet 512×512 class-conditioned generation. Our tokenizer is fine-tuned from ImageNet 256×256 checkpoint for 25 epochs. Scratch *vs.* fine-tune: Generative models trained from scratch or initialized from ImageNet 256×256 checkpoints (see Sec. A.3 for details).

5.4 Text-to-Image Generation

We further validate l-DeTok on text-to-image (T2I) generation. Following the protocol in Bao et al. (2022); Yu et al. (2025b), we train models from scratch on the MS-COCO train split (Lin et al., 2014) and evaluate on the validation split using FID-30k. We compare T2I variants of MAR-B and SiT-B under identical conditions, varying only the tokenizer (details in Sec. C). Table 5 summarizes the results, demonstrating that l-DeTok substantially improves not only sample quality and diversity (FID), but also text–image conditioning alignment (CLIP score). Qualitatively (see Figures. C.1, C.2,C.3, and C.4), other tokenizers frequently produce the "spot artifacts" reported in previous works (Fan et al., 2025; Team et al., 2025) under the text-to-image setting, whereas such artifacts are notably absent with l-DeTok.

Tokenizer	T2I MAR-B		T2I SiT-B	
	FID↓	CLIP↑	FID↓	CLIP↑
VA-VAE (Yao & Wang, 2025)	34.64	21.98	5.83	**25.07**
SD-VAE Rombach et al. (2022)	19.75	22.86	6.63	24.00
MAR-VAE (Li et al., 2024a)	12.49	22.07	5.74	23.50
Ours (l-DeTok)	**4.97**	**24.82**	**4.31**	24.61

Table 5: **MS-COCO text-to-image generation.** Comparison of tokenizers on T2I-variants MAR-B and SiT-B, reported with best CFG scales. Our l-DeTok achieves both lower FID (better diversity) and higher CLIP scores (better alignment), outperforming all others.

6 Conclusion

Limitations. On the general side, our study primarily investigates continuous-valued tokenizers; the effectiveness of our denoising strategy for discrete, vector-quantized (VQ) tokenizers remains an open question. Exploring denoising-based objectives in the context of VQ tokenizers is a valuable future direction. On the technical side, we observe a training/inference discrepancy in our tokenizer: the decoder is primarily trained on noise-injected latent embeddings, whereas it operates on almost noise-free embeddings during inference. Fine-tuning the decoder on clean latent embeddings partially addresses this discrepancy. Further investigation into mitigating this discrepancy could yield additional improvements. Nonetheless, the core motivation and insights of our work remain robust.

Conclusion. The strong generalization and effectiveness of our l-DeTok across different generative models highlights a fundamental yet underexplored opportunity: tokenizer representations alone can substantially advance different generative approaches without architectural changes. Our findings suggest that explicitly aligning tokenizer training with downstream denoising tasks is surprisingly beneficial for generative modeling, complementing traditional focuses on pixel-level accuracy or semantic alignment. We hope this perspective will inspire further research into tokenizer designs. Lastly, the core idea of this work is general and extends beyond image tokenization. Exploring its potential in broader generative modeling, such as video generation, action generation, protein design, and other domains, will be an exciting direction for future research.

REFERENCES

Fan Bao, Chongxuan Li, Yue Cao, and Jun Zhu. All are worth words: a vit backbone for score-based diffusion models. In *NeurIPS 2022 Workshop on Score-Based Methods*, 2022.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. MaskGIT: Masked generative image Transformer. In *CVPR*, 2022.

Hao Chen, Yujin Han, Fangyi Chen, Xiang Li, Yidong Wang, Jindong Wang, Ze Wang, Zicheng Liu, Difan Zou, and Bhiksha Raj. Masked autoencoders are effective tokenizers for diffusion models. *arXiv preprint arXiv:2502.03444*, 2025a.

Hao Chen, Ze Wang, Xiang Li, Ximeng Sun, Fangyi Chen, Jiang Liu, Jindong Wang, Bhiksha Raj, Zicheng Liu, and Emad Barsoum. Softvq-vae: Efficient 1-dimensional continuous tokenizer. In *CVPR*, 2025b.

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *ICML*, 2020a.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020b.

Xinlei Chen, Zhuang Liu, Saining Xie, and Kaiming He. Deconstructing denoising diffusion models for self-supervised learning. In *ICLR*, 2025c.

Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. In *ICML*, 2023.

Chaorui Deng, Deyao Zhu, Kunchang Li, Shi Guang, and Haoqi Fan. Causal diffusion transformers for generative modeling. *arXiv preprint arXiv:2412.12095*, 2024.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *NeurIPS*, 2021.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *ICML*, 2024.

Lijie Fan, Tianhong Li, Siyang Qin, Yuanzhen Li, Chen Sun, Michael Rubinstein, Deqing Sun, Kaiming He, and Yonglong Tian. Fluid: Scaling autoregressive text-to-image generative models with continuous tokens. In *ICLR*, 2025.

Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva: Exploring the limits of masked visual representation learning at scale. In *CVPR*, 2023.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch SGD: Training ImageNet in 1 hour. *arXiv:1706.02677*, 2017.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *NeurIPS*, 2020.

Philippe Hansen-Estruch, David Yan, Ching-Yao Chung, Orr Zohar, Jialiang Wang, Tingbo Hou, Tao Xu, Sriram Vishwanath, Peter Vajda, and Xinlei Chen. Learnings from scaling visual tokenizers for reconstruction and generation. *arXiv preprint arXiv:2501.09755*, 2025.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. In *NeurIPS*, 2024a.

Xiang Li, Kai Qiu, Hao Chen, Jason Kuen, Jiuxiang Gu, Bhiksha Raj, and Zhe Lin. Imagefolder: Autoregressive image generation with folded tokens. In *ICLR*, 2025.

Yazhe Li, Jorg Bornschein, and Ting Chen. Denoising autoregressive representation learning. In *ICML*, 2024b.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *ICLR*, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *ECCV*, 2024.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research Journal*, pp. 1–31, 2024.

Ziqi Pang, Tianyuan Zhang, Fujun Luan, Yunze Man, Hao Tan, Kai Zhang, William T Freeman, and Yu-Xiong Wang. Randar: Decoder-only autoregressive visual generation in random orders. In *CVPR*, 2025.

William Peebles and Saining Xie. Scalable diffusion models with Transformers. In *ICCV*, 2023a.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023b.

Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. Variational autoencoder for deep learning of images, labels and captions. In *NeurIPS*, 2016.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Sucheng Ren, Qihang Yu, Ju He, Xiaohui Shen, Alan Yuille, and Liang-Chieh Chen. Flowar: Scale-wise autoregressive image generation meets flow matching. In *ICML*, 2025.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

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.

NextStep Team, Chunrui Han, Guopeng Li, Jingwei Wu, Quan Sun, Yan Cai, Yuang Peng, Zheng Ge, Deyu Zhou, Haomiao Tang, et al. Nextstep-1: Toward autoregressive image generation with continuous tokens at scale. *arXiv preprint arXiv:2508.10711*, 2025.

Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. In *NeurIPS*, 2025.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Shuai Wang, Zhi Tian, Weilin Huang, and Limin Wang. Ddt: Decoupled diffusion transformer. *arXiv preprint arXiv:2504.05741*, 2025.

Jingfeng Yao and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *CVPR*, 2025.

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. *arXiv preprint arXiv:2110.04627*, 2021.

Qihang Yu, Ju He, Xueqing Deng, Xiaohui Shen, and Liang-Chieh Chen. Randomized autoregressive visual generation. *arXiv preprint arXiv:2411.00776*, 2024.

Qihang Yu, Mark Weber, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. An image is worth 32 tokens for reconstruction and generation. In *NeurIPS*, 2025a.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *ICLR*, 2025b.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *NeurIPS*, 2019.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. In *ICLR*, 2022.

APPENDIX

A TRAINING AND INFERENCE DETAILS

A.1 TOKENIZER

Model. We implement our tokenizer using ViTs for both encoder and decoder (Vaswani et al., 2017; Dosovitskiy et al., 2021). We provide detailed model parameters in Table A.1. We adopt recent architectural advances from LLaMA (Touvron et al., 2023), including RoPE (Su et al., 2024) (together with learnable positional embeddings following Fang et al. (2023)), RMSNorm (Zhang & Sennrich, 2019), and SwiGLU-FFN (Shazeer, 2020). The encoder operates at a patch size of 16, yielding 256 latent tokens for each 256×256 image, while the decoder uses patch size 1 as there is no resolution change. The latent dimension is set to 16. We omit the `[CLS]` token for simplicity.

Table A.1: **Model Size.** We report the model configurations and parameter counts of tokenizer encoders and decoders. The total tokenizer size is the sum of encoder and decoder parameters. For example, a tokenizer combining a ViT-S encoder and ViT-B decoder (S-B) has 111.6M parameters, while a tokenizer with both ViT-B encoder and decoder (B-B) has 171.7M parameters.

Size	Hidden Size	Blocks	Heads	Parameters
Small (S)	512	8	8	25.75M
Base (B)	768	12	12	85.85M

Training. Our tokenizer is trained using the weighted loss defined in Eq. 6, with default weights $\lambda_{\text{KL}} = 10^{-6}$, $\lambda_{\text{percep}} = 1.0$, and $\lambda_{\text{GAN}} = 0.1$. For ablation studies, we use a ViT-S encoder and a ViT-B decoder, disable GAN loss, and train for 50 epochs. We observe that including a GAN loss (Yu et al., 2021; Goodfellow et al., 2014; Esser et al., 2021) sharpens reconstructions but roughly doubles training time, without altering result trends. For our final experiments, we adopt ViT-B for both encoder and decoder, enable GAN loss from epoch 100, and train for 200 epochs. In both settings, we use a global batch size of 1024, and the peak learning rate is set to 4.0×10^{-4} (scaled linearly from 1.0×10^{-4} at a batch size of 256 (Goyal et al., 2017)). We apply linear warm-up for 25% of the total epochs (12 epochs for 50-epoch training, and 50 epochs for 200-epoch training), followed by cosine learning rate decay. We use the AdamW optimizer (Loshchilov & Hutter, 2019) with β parameters (0.9, 0.95) and a weight decay of 1.0×10^{-4}. The only data augmentation employed is horizontal flipping. Our reconstruction loss closely follows the implementation in Yu et al. (2025a).

Training budget. Training an S-B tokenizer (see Table A.1) without GAN loss for 50 epochs takes roughly 160 NVIDIA A100 GPU hours (enabling GAN loss from epoch 20 extends training to about 288 A100 GPU hours). Training a B-B tokenizer with GAN loss enabled from epoch 100 for a total of 200 epochs takes approximately 1,150 A100 GPU hours.

Pseudo-code of latent denoising. See Algorithm 1.

A.2 GENERATIVE MODELS

Model. To evaluate the broad effectiveness of a tokenizer, we experiment with *six* representative generative models, including three non-autoregressive models: DiT (Peebles & Xie, 2023b), SiT (Ma et al., 2024), and LightningDiT (Yao & Wang, 2025); and three autoregressive models: MAR (Li et al., 2024a), causal RandomAR, and RasterAR based on RAR (Yu et al., 2024) and *diffloss* (Li et al., 2024a). We follow their officially released code to re-implement all generative models within our codebase to standardize training and evaluation, ensuring fair comparisons.

We introduce some modifications: for DiT (Peebles & Xie, 2023b), SiT (Ma et al., 2024), and LightningDiT (Yao & Wang, 2025), we apply classifier-free guidance (CFG) to all latent channels, rather than only the first three channels used in their original implementations.[2] For training on 1D tokens (*e.g.*, MAETok (Chen et al., 2025a)), we use simple 1D learnable positional embeddings and disable RoPE used in LightningDiT. We adopt the default samplers from the original implementations, using 250 denoising steps during inference. Due to the frequently observed training instability in SiT

[2]See original implementations in DiT, SiT, and LightningDiT.

Algorithm 1 Latent Denoising: PyTorch-like Pseudo-code

```
def denoise(x, encoder, decoder, max_mask_ratio=0.7, gamma=3.0):
    # encode input image to latent embeddings under (optional) masking
    z, ids_restore = encoder(x, max_mask_ratio=max_mask_ratio)

    # variational latent embeddings
    posteriors = diagonal_gaussian_dist(z)
    z_sampled = posteriors.sample()

    # sample interpolation factor uniformly from [0, 1]
    bsz, n_tokens, chans = z_sampled.shape
    device = z_sampled.device
    noise_level = torch.rand(bsz, 1, 1, device=device).expand(-1, n_tokens,
        chans)

    # generate Gaussian noise
    noise = gamma * torch.randn(bsz, n_tokens, chans, device=device)

    # interpolate latent embeddings with noise
    z_noised = (1 - noise_level) * z_quantized + noise_level * noise

    # reconstruct the inputs
    recon = decoder(z_noised, ids_restore)
    return recon
```

and LightningDiT in our early experiments, we apply QK-Norm (Dehghani et al., 2023) to stabilize training. These modifications are consistent across all tokenizers for fairness.

For autoregressive (AR) methods, we use the default MAR model and implement RandomAR and RasterAR following RAR (Yu et al., 2024). To produce continuous tokens, we employ the *diffloss* (Li et al., 2024a) with a 3-layer, 1024-channel MLP head. We disable dropout in all MLP layers within Transformer blocks for autoregressive models. For inference, we use 64 autoregressive steps and 100 denoising steps for all experiments, except those in system-level comparison (Tables 3 and 4), where we use 256 autoregressive steps or 512 autoregressive steps. Sampling is performed with the default MAR sampler across all AR models. Following MAR (Li et al., 2024a), we set the sampling temperature to 1.0 when using CFG, and sweep temperatures when CFG is disabled (*i.e.*, CFG=1.0).

Training. We use a standardized training recipe for all generative models. Specifically, we follow the hyperparameters from Yao & Wang (2025), training generative models with a global batch size of 1024, using AdamW optimizer (Loshchilov & Hutter, 2019) with a constant learning rate of 2×10^{-4}, without warm-up, gradient clipping, or weight decay. We do not tune these hyperparameters. For ablation studies, we train generative models for 100 epochs. For larger-scale experiments on MAR models, we train them for 800 epochs. All models utilize exponential moving average (EMA) with a decay rate of 0.9999.

Standardization. We always standardize tokenizer outputs by subtracting the channel-wise mean and dividing by the channel-wise standard deviation, both computed from the ImageNet training set. For publicly available tokenizers, we use their official standardization steps.

Metrics. Unless noted otherwise, we report FID@50k scores with classifier-free guidance (CFG), with optimal CFG scales searched from FID@10k results. We abbreviate FID@50k as FID.

Training budget. Training DiT-B, SiT-B, and LightningDiT-B for 100 epochs takes approximately 128 to 200 A100 hours using locally cached tokens. Training MAR-B, RandomAR-B, and RasterAR-B for 100 epochs takes approximately 220 to 250 A100 hours. For the final MAR-B model used in the system-level comparison (Table 3), training for 800 epochs takes roughly 2,450 A100 hours; training MAR-L for the same duration takes about 3,850 A100 hours (online evaluation time included).

A.3 Tokenizer tuning

Decoder fine-tuning. We observe an interesting discrepancy between training and inference in our *l*-DeTok. Our decoder is trained predominantly with noise-corrupted latent embeddings but encounters nearly clean embeddings during inference. To address this gap, we fine-tune the decoder

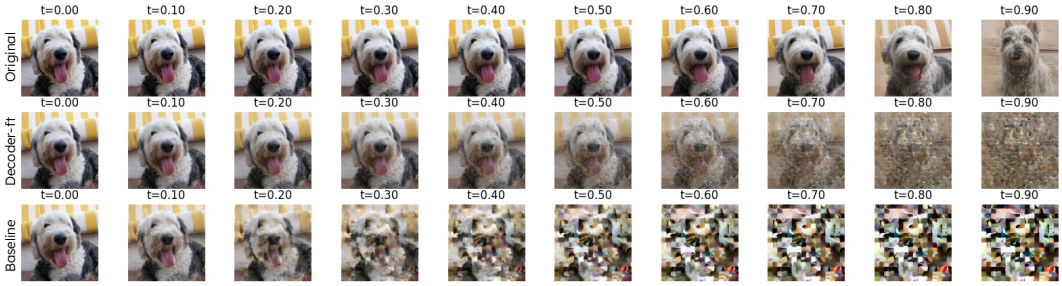


Figure A.1: **Visualization of latent denoising.** Images generated from latent embeddings corrupted with varying noise levels (t) by the original decoder (top), fine-tuned decoder (middle), and baseline decoder (bottom). The fine-tuned decoder shows a reduced ability to recover images from noisy embeddings compared to the original decoder. The baseline tokenizer trained without the denoising objective fails to reconstruct original images from noisy latents.

Figure A.2: **Visualization of "mask" denoising.** Images generated from masked inputs with varying masking ratios (m) by the original decoder (top) and fine-tuned decoder (bottom). The fine-tuned decoder exhibits diminished capability in reconstructing masked regions compared to the original decoder.

on clean latent embeddings, *i.e.*, masking and latent noising are disabled, for an additional 100 epochs. This adjustment partially alleviates the discrepancy, improving the FID of the 800-epoch MAR-L model from 1.43 to 1.35 and MAR-B model from 1.61 to 1.55 (Table 3). Figure A.1 and Figure A.2 compare the denoising capability of different tokenizers. Although fine-tuning decoder reduces the decoder's denoising strength, it enhances the image quality (FID and IS) when decoding from generated latent embeddings. Crucially, this demonstrates that the quality of latent representations produced by the tokenizer *encoder*, rather than the *decoder*'s error-tolerance, is the primary driver of generative model improvements. Therefore, our l-DeTok does more than improving the the decoder's robustness to sampling errors.

To further verify that performance gains primarily stem from improved latent representations rather than enhanced decoder-side error tolerance, we conduct an additional experiment by fine-tuning only the MAR-VAE decoder on the denoising task while keeping the encoder frozen. In this way, we can reuse pre-trained models for comparison since their modeling space, *encoder latent space*, remains fixed. Contrary to expectations, this setup—improving decoder robustness without altering latent embeddings—actually leads to slightly worse FIDs (around 0.1 to 0.3 degradation). This indicates that merely boosting decoder denoising capability exacerbates the training-inference discrepancy, as generative models are already proficient at denoising. These results reinforce that the primary advantage of our approach lies explicitly in the improved quality of latent embeddings, not in decoder-side denoising enhancements.

Adopting tokenizers to 512×512 resolution. To save computation, we fine-tune tokenizer checkpoints from ImageNet-256×256 for the ImageNet-512×512 generation task. Specifically, we halve the learning rate to 2.0×10^{-4} and fine-tune tokenizers initialized from the 200-epoch 256×256 checkpoints for an additional 25 epochs, enabling GAN loss from the start. We compare full fine-tuning to decoder-only fine-tuning: decoder-only fine-tuning achieves better PSNR (26.3 *vs.* 24.9) and comparable rFID (0.92 *vs.* 0.86), yet fully fine-tuned tokenizers consistently provide superior downstream generative performance. For instance, using fully fine-tuned tokenizers, MAR-B achieves 2.66 FID and SiT achieves 5.98 FID, compared to decoder-only fine-tuned tokenizers at

3.63 and 6.32 FID, respectively. All generative models are trained from scratch for 100 epochs, and we report FID@50K at optimal CFG scales. Consequently, we adopt fully fine-tuned tokenizers.

These results reinforce our core motivation: encoder-produced tokenizer embeddings primarily drive generative quality, while decoder fine-tuning provides incremental improvements. Therefore, improving encoder is more fundamental than improving the decoder.

Adapting generative models to 512×512 resolution. Similar to tokenizer fine-tuning, we fine-tune a MAR-L checkpoint from ImageNet-256×256 for the ImageNet-512×512 generation task to reduce compute. Additionally, we train a MAR-B model from scratch to verify the compatibility of our tokenizer for end-to-end training. Note that we do not aim to hill climb the performance metrics in Table 4, since our tokenizer is only fine-tuned for 25 epochs rather than trained from scratch for many epochs. Instead, our intention is for these results to demonstrate the broad applicability of our approach.

A.4 l-DeTok with Semantics Distillation

Our l-DeTok *already* holds great promise for generative models across both autoregressive (AR) and non-autoregressive (non-AR) methods without semantics distillation. It is natural to question whether incorporating semantics distillation could further improve its performance.

To investigate this, we introduce an auxiliary semantics-distillation loss into our tokenizer training pipeline. Specifically, we implement the latent representation projector as a three-layer MLP, similar to REPA (Yu et al., 2025b). The noised latent embeddings (`z_noised` at line 18 in Algorithm 1) are projected through the following layers: `Linear(16, 2048)` $\rightarrow$ `SiLU()` $\rightarrow$ `Linear(2048, 2048)` $\rightarrow$ `SiLU()` $\rightarrow$ `Linear(2048, 768)`, where 16 represents the latent embedding dimension and 2048 the intermediate projection dimension. This projector is trained to maximize the cosine similarity between the projected embeddings and semantic features extracted from the pretrained DINOv2-Base model (Oquab et al., 2024).

We have also experimented with a decoder-based semantics-distillation approach, *i.e.*, using an auxiliary Transformer-based decoder to predict pretrained features. After training both implementations for 50 epochs, we found their results were very similar, though the decoder-based implementation required more computation because of the decoder overhead. Thus, we adopted the simpler MLP-based approach for efficiency.

The loss coefficient for minimizing the cosine distance is set to 1.0 for simplicity, which we do not sweep or tune. We train this tokenizer using the identical settings as the "+Extended" protocol described in Table 1: 200 epochs in total, activating GAN loss from epoch 100.

The results from this experiment (summarized in Table 2) demonstrate substantial improvements, particularly for diffusion-based non-AR models (e.g., SiT). With semantics distillation, our l-DeTok achieves state-of-the-art FID scores for non-AR models, clearly surpassing previous semantics-distilled tokenizers such as MAETok (Chen et al., 2025a) and VA-VAE (Yao & Wang, 2025). While AR models benefit relatively less from distillation—and performance may even slightly degrade compared to the non-distilled l-DeTok—their generative quality remains notably better than previous approaches.

Nonetheless, we emphasize the practical significance of our original l-DeTok formulation without semantics distillation, especially for domains like video, audio, proteins, poses, or trajectories, where strong pretrained semantic models comparable to DINOv2 (Oquab et al., 2024) may be unavailable or significantly weaker. Thus, our proposed tokenizer remains broadly valuable both with and without semantic distillation.

A.5 Convolutional Tokenizer

Model. We reuse MAR-VAE and SD-VAE architectures with a downsampling ratio of 16 and latent dimension 16, producing latent representations of shape $(16, 16, 16)$ for (C, H, W). We add only a *few lines* of latent-denoising code (similar to Algorithm 1) to CNN-based tokenizers. Tokenizers are trained for 50 epochs, both with and without latent-denoising, using hyperparameters identical

to our Transformer-based tokenizers, potentially disadvantaging CNN models.[3] We enable GAN loss from epoch 10 for CNN tokenizers. We do not sweep the noise standard deviation and only run with $\gamma = 3.0$ and $\gamma = 0.0$ (baseline). Further hyperparameters and noise strength tuning and longer training may yield additional improvements for CNN-based tokenizers, which we leave for future work.

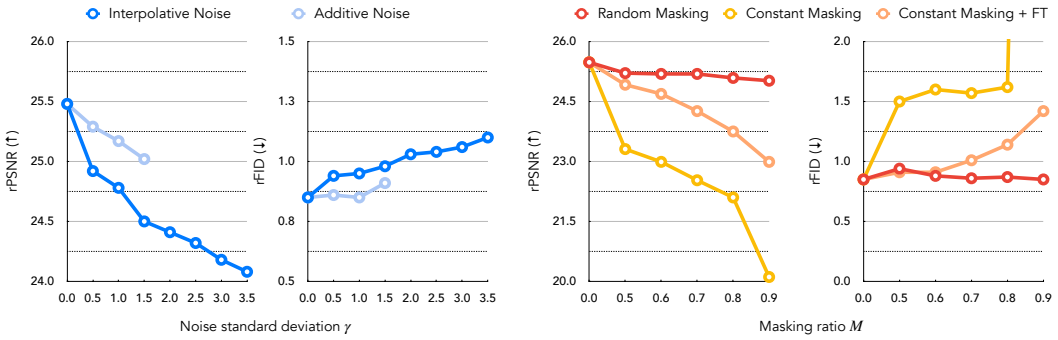
Results. We report FID@50k under best CFG scales. With the default CNN baseline, MAR-B achieves 3.32 FID and SiT-B 7.11 FID. With our denoising-based CNN tokenizers, MAR-B improves to 2.82 FID and SiT-B to 5.62 FID, demonstrating that l-DeTok provides consistent gains also in convolutional networks. Our results clearly demonstrate the general, architecture-agnostic nature of our latent-denoising idea.

B Additional Results

B.1 Effect of Noising on Reconstruction Performance

Figure B.1 reports tokenizer reconstruction quality (measured by rPSNR and rFID) under various noising strategies discussed in Section 5.1. All tokenizers are trained for 50 epochs without GAN loss. As expected, reconstruction performance degrades with increased noise strength (noise standard deviation or masking ratio). Nevertheless, even under substantial noise levels, our tokenizers achieve reasonably strong reconstruction quality, consistently exceeding 24.0 rPSNR and remaining below 1.2 rFID.

Figure B.1-(b) highlights the importance of the additional 10-epoch decoder fine-tuning step for constant-masking tokenizers, which effectively mitigates the visual degradation and photorealism loss stemming from the training-inference distribution mismatch. This fine-tuning significantly improves decoder reconstruction, particularly at higher masking ratios. Training with randomized masking consistently outperforms constant masking, indicating that variable masking ratios lead to more robust reconstruction and superior generative outcomes.



(a) Reconstruction performance vs. Latent Noise

(b) Reconstruction performance vs. Masking

Figure B.1: **Tokenizer reconstruction under various noising strategies.** We report rPSNR and rFID for tokenizers studied in Section 5.1, trained for 50 epochs without GAN loss. With masking ratio $M = 0.9$, constant masking unexpectedly collapses performance, yielding an unexpected rFID of 13.95 (out of plot range).

B.2 FID *vs.* CFG curves.

Figures B.2 and B.3 compare how classifier-free guidance (CFG) scales influence generative performance (FID and IS) across different tokenizers and generative models. We use warm colors to denote semantics-distilled tokenizers (marked by "*"), and cool colors for tokenizers without distillation. Our l-DeTok consistently achieves stronger performance across varying CFG scales, improving notably over standard tokenizers and matching or even surpassing semantics-distilled ones. Final FID scores computed over 50,000 generated images at optimal CFG scales are summarized in Table 2.

[3] In our experience, Transformer- and CNN-based tokenizers typically require distinct hyperparameter sets.

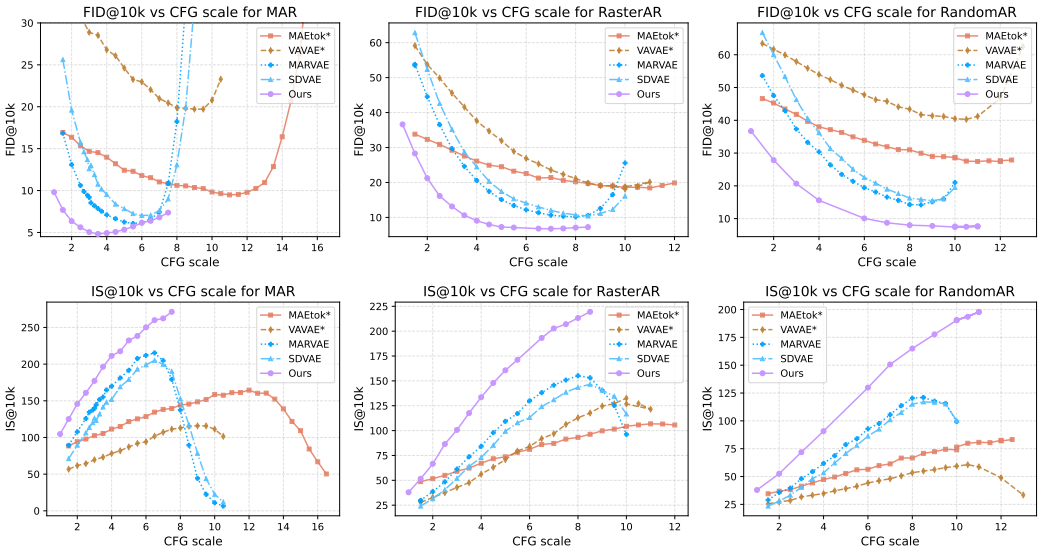


Figure B.2: **Tokenizer comparison for autoregressive models.** We report FID@10k and IS@10k scores at different classifier-free guidance (CFG) scales. Semantics-distilled tokenizers (denoted by "*") are shown in warm colors, while those without distillation are shown in cool colors. Our l-DeTok consistently achieves superior FID and IS metrics compared to other tokenizers. See Table 2 for results on 50,000 images evaluated at the optimal CFG scales.

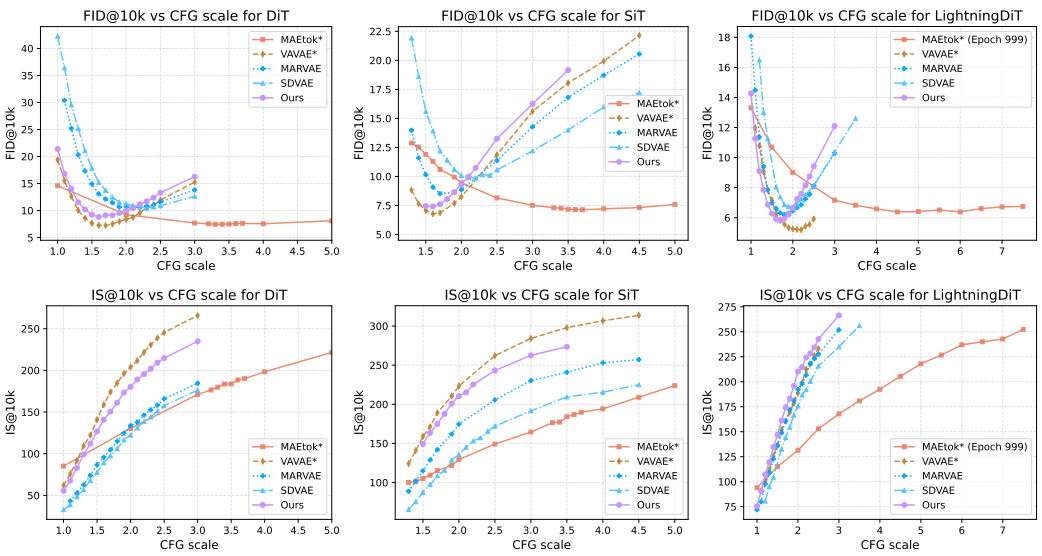


Figure B.3: **Tokenizer comparison for non-autoregressive models.** We report FID@10k and IS@10k scores at different classifier-free guidance (CFG) scales. Semantics-distilled tokenizers (denoted by "*") are shown in warm colors, while those without distillation are shown in cool colors. Our l-DeTok consistently outperforms standard (non-semantics-distilled) tokenizers, matching the performance of the best semantics-distilled tokenizer (VA-VAE (Yao & Wang, 2025)) and surpassing MAETok (Chen et al., 2025a) in IS. See Table 2 for results on 50,000 images evaluated at the optimal CFG scales.

C Text-to-Image Generation

We further validate l-DeTok on text-to-image (T2I) generation. Following the protocol in (Bao et al., 2022; Yu et al., 2025b), we train models from scratch on the MS-COCO train split (Lin

Figure C.1: **Text-to-image generation using MAR-VAE tokenizer.** Uncurated text-conditional generations on MS-COCO with T2I-MAR-B trained using MAR-VAE tokenizer (Li et al., 2024a). Red boxes highlight the "spot artifacts" reported in previous works (Fan et al., 2025; Team et al., 2025), which notably do not appear when using our tokenizer (see Figure C.4 for comparison).

et al., 2014) and evaluate on the validation split. To adapt SiT and MAR for T2I, we make minimal modifications: (1) For SiT, we concatenate text and image tokens, computing the loss only on image tokens. The AdaLN modulation input is simplified from {timestep embedding + class embedding} to only timestep embedding. (2) For MAR, we fill the buffer tokens, which originally stores class embeddings, with text embeddings. In both cases, text embeddings are extracted using CLIP-L as in Bao et al. (2022); Yu et al. (2025b).

We train T2I variants of MAR-B and SiT-B under identical settings with different tokenizers. Except for SDVAE, all tokenizers are pretrained on ImageNet 256×256 without any tuning on the MSCOCO dataset. MAR models are trained for 800 epochs while SiT models are trained for 1000 epochs.

Qualitative Text-to-Image Comparison. Figures C.1, C.2, C.3, and C.4 visualize uncurated text-to-image samples generated by T2I-MAR with various tokenizers. Other tokenizers frequently produce "spot artifacts" as previously reported (Fan et al., 2025; Team et al., 2025), whereas such artifacts are notably absent with our l-DeTok.

Text prompts for each column (from left to right) are: "A green train is coming down the tracks.", "A group of skiers are preparing to ski down a mountain.", "A small kitchen with a low ceiling.", "A group of elephants walking in muddy water.", "A living area with a television and a table.", "A road with traffic lights, street lights and cars.", "A bus driving in a city area with traffic signs.", "A bus pulls over to the curb close to an intersection.", "A group of people are walking and one is holding an umbrella.", "A baseball player taking a swing at an incoming ball.", "A city street lined with brick buildings and trees.", and "A close up of a plate of broccoli and sauce."

Figure C.2: **Text-to-image generation using SD-VAE tokenizer.** Uncurated text-conditional generations on MS-COCO with T2I-MAR-B trained using SD-VAE tokenizer (Rombach et al., 2022). Red boxes highlight the "spot artifacts" reported in previous works (Fan et al., 2025; Team et al., 2025), which notably do not appear when using our tokenizer (see Figure C.4 for comparison).

Figure C.3: **Text-to-image generation using VA-VAE tokenizer.** Uncurated text-conditional generations on MS-COCO with T2I-MAR-B trained using VA-VAE tokenizer (Yao & Wang, 2025). Red boxes highlight the "spot artifacts" reported in previous works (Fan et al., 2025; Team et al., 2025), which notably do not appear when using our tokenizer (see Figure C.4 for comparison).

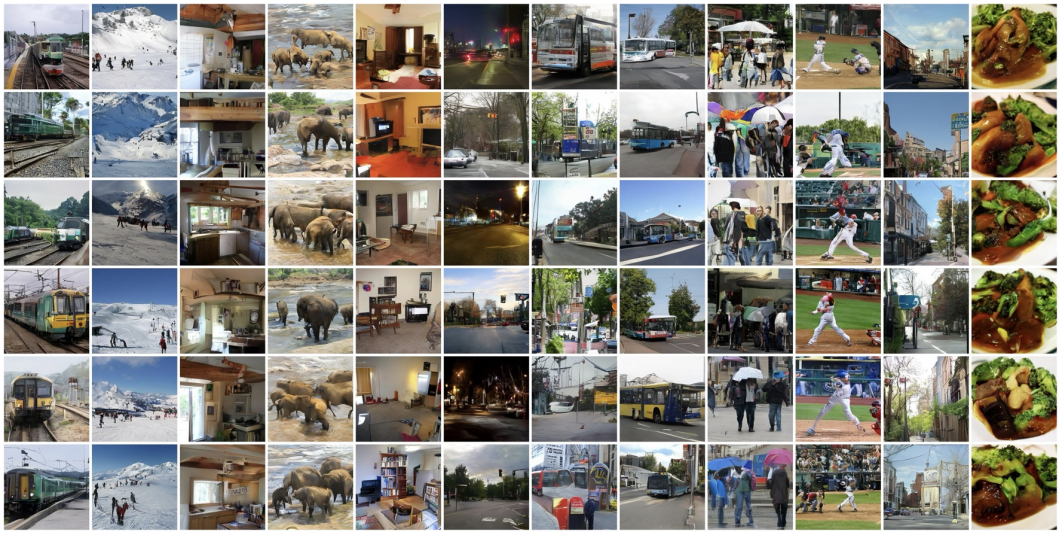


Figure C.4: **Text-to-image generation using our l-DeTok tokenizer.** Uncurated text-conditional generations on MS-COCO with T2I-MAR-B trained using our l-DeTok tokenizer (Yao & Wang, 2025).