ϵ-VAE: DENOISING AS VISUAL DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

In generative modeling, tokenization simplifies complex data into compact, structured representations, creating a more efficient, learnable space. For high-dimensional visual data, it reduces redundancy and emphasizes key features for high-quality generation. Current visual tokenization methods rely on a traditional autoencoder framework, where the encoder compresses data into latent representations, and the decoder reconstructs the original input. In this work, we offer a new perspective by proposing *denoising as decoding*, shifting from single-step reconstruction to iterative refinement. Specifically, we replace the decoder with a diffusion process that iteratively refines noise to recover the original image, guided by the latents provided by the encoder. We evaluate our approach by assessing both reconstruction (rFID) and generation quality (FID), comparing it to state-of-the-art autoencoding approach. We hope this work offers new insights into integrating iterative generation and autoencoding for improved compression and generation.

1 INTRODUCTION

Generative modeling aims to capture the underlying distribution of training data, enabling realistic sample generation during inference. A key preprocessing step is tokenization, which converts raw data into discrete tokens or continuous latent representations. In vision tasks, continuous latents are typically produced by an encoder, whereas discrete tokens are commonly derived from embeddings in language tasks. These compact representations allow models to efficiently learn complex patterns, enhancing the quality of generated outputs.

Two dominant paradigms in modern generative modeling are autoregression (Radford et al., 2018) and diffusion (Ho et al., 2020). Tokenization is an essential in both: discrete tokens direct step-by-step conditional generation in autoregressive models, while continuous latents streamline the denoising process in diffusion models. Empirical results across language (Achiam et al., 2023; Anil et al., 2023; Dubey et al., 2024) and vision (Baldridge et al., 2024; Esser et al., 2024; Brooks et al., 2024) tasks show that tokenization—whether discrete or continuous—improves generative performance. We focus on tokenization for latent diffusion models, which excel at generating high-dimensional visual data.

Given its central role in both paradigms, understanding how tokenization works is essential. In language processing, tokenization is relatively straightforward, involving segmenting text into discrete units such as words, subwords, or characters (Sennrich et al., 2015; Kudo & Richardson, 2018; Kudo, 2018). However, tokenization in visual domains poses greater challenges due to the continuous, high-dimensional, and redundant nature. Instead of direct segmentation, compact representations are typically learned using an autoencoding (Hinton & Salakhutdinov, 2006). Despite rapid advancements in visual generation techniques, the design of tokenizers has received relatively little attention. This is evident in the minimal evolution of tokenizers used in state-of-the-art models, which have remained largely unchanged since their initial introduction (Van Den Oord et al., 2017).

In this paper, we address this gap by revisiting the widely adopted visual autoencoding formulation (Esser et al., 2021), aiming to achieve higher compression rates and improved reconstruction quality, thereby enhancing generation quality of downstream generative models. Our key idea is to rethink the traditional autoencoding pipeline, which typically involves an encoder that compresses the input into a latent representation, followed by a decoder that reconstructs the original data in a single step. In our approach, we replace the deterministic decoder with a diffusion process. Here, the encoder still compresses the input into a latent representation, but instead of a one-step reconstruction, the diffusion model iteratively denoises the data to recover the original. This reframing turns the reconstruction phase into a step-by-step refinement, where the diffusion model, guided by the latent representation, progressively restores the original data.

To implement our approach effectively, several key design factors must be carefully considered. First, the architectural design must ensure effective conditioning of the diffusion decoder on the latent representations provided by the encoder. Second, the objectives for training the diffusion decoder should also explore potential synergies with traditional autoencoding losses, such as LPIPS (Zhang et al., 2018) and GAN (Esser et al., 2021). Finally, diffusion-specific design choices are crucial, including: (1) the model parameterization, which defines the prediction target for the diffusion decoder; (2) the noise schedule, which dictates the optimization trajectory; and (3) the distribution of time steps during training and testing, which balances noise levels during learning and generation. Our study systematically explores all these components under controlled experiments.

In summary, our contributions are as follows: (1) introducing a novel approach that fully leverages the capabilities of diffusion decoders for more practical diffusion-based autoencoding, achieving strong rFID, high sampling efficiency (within 1 to 3 steps), and robust resolution generalization; (2) presenting key design choices to optimize performance; and (3) conducting extensive controlled experiments that demonstrate our method achieves high-quality reconstruction and generation results, outperforming leading visual auto-encoding paradigms.

2 Background

We start by briefly reviewing the basic concepts required to understand the proposed method. A more detailed summary of related work is deferred to Appendix A.

Visual tokenization. To achieve efficient and scalable high-resolution image synthesis, common generative models, including autoregressive models (Razavi et al., 2019; Esser et al., 2021; Chang et al., 2022) and diffusion models (Rombach et al., 2022), are typically trained in a low-resolution latent space by first downsampling the input image using a tokenizer. The tokenizer is generally implemented as a convolutional autoencoder consisting of an encoder, $\mathcal{E}$, and a decoder, $\mathcal{G}$. Specifically, the encoder, $\mathcal{E}$, compresses an input image $\boldsymbol{x} \in \mathbb{R}^{H\times W\times 3}$ into a set of latent codes (*i.e.*, tokens), $\mathcal{E}(\boldsymbol{x}) = \boldsymbol{z} \in \mathbb{R}^{H/f\times W/f\times n_z}$, where f is the downsampling factor and n_z is the latent channel dimensions. The decoder, $\mathcal{G}$, then reconstructs the input from $\boldsymbol{z}$, such that $\mathcal{G}(\boldsymbol{z}) = \boldsymbol{x}$.

Training an autoencoder primarily involves several losses: reconstruction loss $\mathcal{L}_{\text{rec}}$, perceptual loss (LPIPS) $\mathcal{L}_{\text{LPIPS}}$, and adversarial loss $\mathcal{L}_{\text{adv}}$. The reconstruction loss minimizes pixel differences (*i.e.*, typically measured by the ℓ_1 or ℓ_2 distance) between $\boldsymbol{x}$ and $\mathcal{G}(\boldsymbol{z})$. The LPIPS loss (Zhang et al., 2018) enforces high-level structural similarities between inputs and reconstructions by minimizing differences in their intermediate features extracted from a pre-trained VGG network (Simonyan & Zisserman, 2015). The adversarial loss (Esser et al., 2021) introduces a discriminator, $\mathcal{D}$, which encourages more photorealistic outputs by distinguishing between real images, $\mathcal{D}(\boldsymbol{x})$, and reconstructions, $\mathcal{D}(\mathcal{G}(\boldsymbol{z}))$. The final training objective is a weighted combination of these losses:

$$\mathcal{L}_{\text{VAE}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{LPIPS}} \cdot \mathcal{L}_{\text{LPIPS}} + \lambda_{\text{adv}} \cdot \mathcal{L}_{\text{adv}}, \tag{1}$$

where the λ values are weighting coefficients. In this paper, we consider the autoencoder optimized by Eq. 1 as our main competing baseline (Esser et al., 2021), as it has become a standard tokenizer training scheme widely adopted in state-of-the-art image and video generative models (Chang et al., 2022; Rombach et al., 2022; Yu et al., 2022; 2023; Kondratyuk et al., 2024; Esser et al., 2024).

Diffusion. Given a data distribution $p_{\boldsymbol{x}}$ and a noise distribution $p_{\boldsymbol{\epsilon}}$, a diffusion process progressively corrupts clean data $\boldsymbol{x}_0 \sim p_{\boldsymbol{x}}$ by adding noise $\boldsymbol{\epsilon} \sim p_{\boldsymbol{\epsilon}}$ and then reverses this corruption to recover the original data (Song & Ermon, 2019; Ho et al., 2020), represented as:

$$\boldsymbol{x}_t = \alpha_t \cdot \boldsymbol{x}_0 + \sigma_t \cdot \boldsymbol{\epsilon}, \tag{2}$$

where $t \in [0, \mathrm{T}]$ and $\boldsymbol{\epsilon}$ is drawn from a standard Gaussian distribution, $p_{\boldsymbol{\epsilon}} = \mathcal{N}(0, I)$. The functions α_t and σ_t govern the trajectory between clean data and noise, affecting both training and sampling. The basic parameterization in Ho et al. (2020) defines $\sigma_t = \sqrt{1-\alpha_t^2}$ with $\alpha_t = \left(\prod_{s=0}^{t}(1-\beta_s)\right)^{\frac{1}{2}}$ for discrete timesteps. The diffusion coefficients β_t are linearly interpolated values between β_0 and β_{T-1} as $\beta_t = \beta_0 + \frac{t}{T-1}(\beta_{T-1} - \beta_0)$, with start and end values are set empirically.

The forward and reverse diffusion processes are described by the following factorizations:

$$q(\boldsymbol{x}_{\Delta t:\mathrm{T}}|\boldsymbol{x}_0) = \prod_{i=1}^{\mathrm{T}} q(\boldsymbol{x}_{i\cdot\Delta t}|\boldsymbol{x}_{(i-1)\cdot\Delta t}) \;\text{ and }\; p(\boldsymbol{x}_{0:\mathrm{T}}) = p(\boldsymbol{x}_\mathrm{T})\prod_{i=1}^{\mathrm{T}} p(\boldsymbol{x}_{(i-1)\cdot\Delta t}|\boldsymbol{x}_{i\cdot\Delta t}), \tag{3}$$

where the forward process $q(\boldsymbol{x}_{\Delta t:\mathrm{T}}|\boldsymbol{x}_0)$ transitions clean data $\boldsymbol{x}_0$ to noise $\boldsymbol{x}_\mathrm{T} = \boldsymbol{\epsilon}$, while the reverse process $p(\boldsymbol{x}_{0:\mathrm{T}})$ recovers clean data from noise. Δt denotes the time step interval or step size.

During training, the model learns the score function $\nabla \log p_t(\boldsymbol{x}) \propto -\frac{\epsilon}{\sigma_t}$, which represents gradient pointing toward the data distribution along the noise-to-data trajectory. In practice, the model $s_\Theta(\boldsymbol{x}_t, t)$ is optimized by minimizing the score-matching objective:

$$\mathcal{L}_{\text{score}} = \min_\Theta \mathbb{E}_{t\sim\pi(t),\epsilon\sim\mathcal{N}(0,I)} \left[w_t \|\sigma_t s_\Theta(\boldsymbol{x}_t, t) + \boldsymbol{\epsilon}\|^2 \right], \tag{4}$$

where $\pi(t)$ defines the time-step sampling distribution and w_t is a time-dependent weight. These elements together influence which time steps or noise levels are prioritized during training.

Conceptually, the diffusion model learns the tangent of the trajectory at each point along the path. During sampling, it progressively recovers clean data from noise based on its predictions.

Rectified flow. Rectified flow provides a specific parametrization of α_t and σ_t such that the trajectory between data and noise follows a "straight" path (Liu et al., 2023; Albergo & Vanden-Eijnden, 2023; Lipman et al., 2022). This trajectory is represented as:

$$\boldsymbol{x}_t = (1-t)\cdot\boldsymbol{x}_0 + t\cdot\boldsymbol{\epsilon}, \tag{5}$$

where $t \in [0, 1]$. In this formulation, the gradient along the trajectory, $\boldsymbol{\epsilon} - \boldsymbol{x}_0$, is deterministic, often referred to as the velocity. The model $v_\Theta(\boldsymbol{x}_t, t)$ is parameterized to predict velocity by minimizing:

$$\min_\Theta \mathbb{E}_{t\sim\pi(t),\boldsymbol{\epsilon}\sim\mathcal{N}(0,I)} \left[\|v_\Theta(\boldsymbol{x}_t, t) - (\boldsymbol{\epsilon} - \boldsymbol{x})\|^2 \right]. \tag{6}$$

We note that this objective is equivalent to a score matching form (Eq. 4), with the weight $w_t = (\frac{1}{1-t})^2$. This equivalence highlights that alternative model parameterizations reduce to a standard denoising objective, where the primary difference lies in the time-dependent weighting functions and the corresponding optimization trajectory (Kingma & Gao, 2024).

During sampling, the model follows a simple probability flow ODE:

$$\mathrm{d}\boldsymbol{x}_t = v_\Theta(\boldsymbol{x}_t, t)\cdot \mathrm{d}t. \tag{7}$$

Although a perfect straight path could theoretically be solved in a single step, the independent coupling between data and noise often results in curved trajectories, necessitating multiple steps to generate high-quality samples (Liu et al., 2023; Lee et al., 2024). In practice, we iteratively apply the standard Euler solver (Euler, 1845) to sample data from noise.

3 Method

We introduce $\boldsymbol{\epsilon}$-VAE, with an overview provided in Figure 1. The core idea is to replace single-step, deterministic decoding with an iterative, stochastic denoising process. By reframing autoencoding as a conditional denoising problem, we anticipate two key improvements: (1) more effective generation of latent representations, allowing the downstream latent diffusion model to learn more efficiently, and (2) enhanced decoding quality due to the iterative and stochastic nature of the diffusion process.

We systematically explore the design space of model architecture, objectives, and diffusion training configurations, including noise and time scheduling. While this work primarily focuses on generating continuous latents for latent diffusion models, the concept of iterative decoding could also be extended to discrete tokens, which we leave for future exploration.

3.1 Modeling

$\boldsymbol{\epsilon}$-VAE retains the encoder $\mathcal{E}$ while enhancing the decoder $\mathcal{G}$ by incorporating a diffusion model, transforming the standard decoding process into an iterative denoising task.

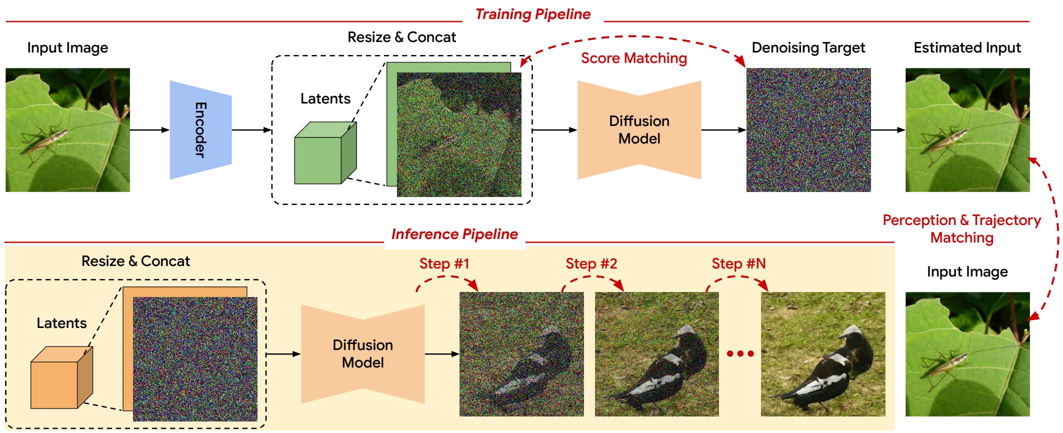

Figure 1: **An overview of ϵ-VAE.** We frame visual decoding as an iterative denoising problem by replacing the autoencoder decoder with a diffusion model, optimized using a combination of score, perception, and trajectory matching losses. During inference, images are reconstructed (or generated) from encoded (or sampled) latents through an iterative denoising process. The number of sampling steps N can be flexibly adjusted within small NFE regimes (from 1 to 3). We empirically confirm that ϵ-VAE significantly outperforms the standard VAE schema, even with just a few steps.

Conditional denoising. Specifically, the input $\boldsymbol{x} \sim p_{\boldsymbol{x}}$ is encoded by the encoder as $\boldsymbol{z} = \mathcal{E}(\boldsymbol{x})$, and this encoding serves as a condition to guide the subsequent denoising process. This reformulates the reverse process in Eq. 3 into a conditional form (Nichol & Dhariwal, 2021; Saharia et al., 2022b):

$$p(\boldsymbol{x}_{0:\mathrm{T}}|\boldsymbol{z}) = p(\boldsymbol{x}_{\mathrm{T}}) \prod_{i=1}^{\mathrm{T}} p(\boldsymbol{x}_{(i-1)\cdot\Delta t}|\boldsymbol{x}_{i\cdot\Delta t}, \boldsymbol{z}), \tag{8}$$

where the denoising process from the noise $\boldsymbol{x}_{\mathrm{T}} = \boldsymbol{\epsilon}$ to the input $\boldsymbol{x}_0 = \boldsymbol{x}$, is additionally conditioned on $\boldsymbol{z}$ over time. Here, the decoder is no longer deterministic, as the process starts from random noise. For a more detailed discussion on this autoencoding formulation, we refer readers to Sec. 5.

Architecture and conditioning. We adopt the standard U-Net architecture from Dhariwal & Nichol (2021) for our diffusion decoder $\mathcal{G}$, while also exploring Transformer-based models (Peebles & Xie, 2023). For conditional denoising, we concatenate the conditioning signal with the input channel-wise, following the approach of diffusion-based super-resolution models (Ho et al., 2022; Saharia et al., 2022b). Specifically, low-resolution latents are upsampled using nearest-neighbor interpolation to match the resolution of $\boldsymbol{x}_t$, then concatenated along the channel dimension. In Appendix C.1, although we experimented with conditioning via AdaGN (Nichol & Dhariwal, 2021), it did not yield significant improvement and introduced additional overhead, so we adopt channel concatenation.

3.2 Objectives

We adopt the standard autoencoding objective from Eq. 1 to train ϵ-VAE, with a key modification: replacing the reconstruction loss $\mathcal{L}_{\text{rec}}$ used for the standard decoder with the score-matching loss $\mathcal{L}_{\text{score}}$ for training the diffusion decoder. Additionally, we introduce a strategy to adjust the perceptual $\mathcal{L}_{\text{LPIPS}}$ and adversarial $\mathcal{L}_{\text{adv}}$ losses to better align with the diffusion decoder training.

Velocity prediction. We adopt the rectified flow parameterization, utilizing a linear optimization trajectory between data and noise, combined with velocity-matching objective (Eq. 6):

$$\mathbb{E}_{t\sim\pi(t),\boldsymbol{\epsilon}\sim\mathcal{N}(0,I)} \left[\|\mathcal{G}(\boldsymbol{x}_t, t, \boldsymbol{z}) - (\boldsymbol{\epsilon} - \boldsymbol{x})\|^2 \right]. \tag{9}$$

Perceptual matching. The LPIPS loss (Zhang et al., 2018) minimizes the perceptual distance between the reconstructions and real images using pre-trained models, typically VGG network (Esser et al., 2021; Yu et al., 2023; 2022). We apply this feature-matching objective to train ϵ-VAE. However, unlike traditional autoencoders, ϵ-VAE predicts velocity instead of directly reconstructing the image during training, making it infeasible to compute the LPIPS loss directly between the prediction and the target image. To address this, we leverage the simple reversing step from Eq. 6 to estimate $\boldsymbol{x}_0$ from the prediction and $\boldsymbol{x}_t$ as follows:

$$\hat{\boldsymbol{x}}_0^t = \boldsymbol{x}_t - t \cdot \mathcal{G}(\boldsymbol{x}_t, t, \boldsymbol{z}), \tag{10}$$

where $\hat{\boldsymbol{x}}_0^t$ represents the reconstructed image estimated by the model at time t. We then compute the LPIPS loss between $\hat{\boldsymbol{x}}_0^t$ and the target real image $\boldsymbol{x}$.

Denoising trajectory matching. The adversarial loss encourages photorealistic outputs by comparing the reconstructions to real images. We modify this to better align with a diffusion decoder. Specifically, our approach adapts the standard adversarial loss to enforce trajectory consistency rather than solely on realism. In practice, we achieve this by minimizing the following divergence, $\mathcal{D}_{\text{adv}}$:

$$\min_{\Theta} \mathbb{E}_{t \sim p_t} \left[\mathcal{D}_{\text{adv}} \left(q(\boldsymbol{x}_0 | \boldsymbol{x}_t) || p_\Theta(\hat{\boldsymbol{x}}_0^t | \boldsymbol{x}_t) \right) \right], \tag{11}$$

where $\mathcal{D}_{\text{adv}}$ is a probability distance metric (Goodfellow et al., 2014; Arjovsky et al., 2017), and we adopt the basic non-saturating GAN (Goodfellow et al., 2014).

For adversarial training, we design a time-dependent discriminator that takes time as input using AdaGN approach (Dhariwal & Nichol, 2021). To simulate the trajectory, we concatenate $\boldsymbol{x}_0$ and $\boldsymbol{x}_t$ along the channel dimension. The generator parameterized by Θ, and the discriminator, parameterized by Φ, are then optimized through a minimax game as:

$$\min_{\Theta} \max_{\Phi} \mathcal{L}_{\text{adv}} = \mathbb{E}_{q(\boldsymbol{x}_0|\boldsymbol{x}_t)} \left[\log \mathcal{D}_\Phi(\boldsymbol{x}_0, \boldsymbol{x}_t, t) \right] + \mathbb{E}_{p_\Theta(\hat{\boldsymbol{x}}_0^t|\boldsymbol{x}_t)} \left[\log \left(1 - \mathcal{D}_\Phi(\hat{\boldsymbol{x}}_0^t, \boldsymbol{x}_t, t) \right) \right], \tag{12}$$

where fake trajectories $p_\Theta(\hat{\boldsymbol{x}}_0^t|\boldsymbol{x}_t)$ are contrasted with real trajectories $q(\boldsymbol{x}_0|\boldsymbol{x}_t)$. To further stabilize training, we apply the R_1 gradient penalty to the discriminator parameters (Mescheder et al., 2018). In Appendix C.1, we explore alternative matching approaches, including the standard adversarial method of comparing individual reconstructions $\hat{\boldsymbol{x}}_0^t$ with real images $\boldsymbol{x}_0$, matching the trajectory steps $\boldsymbol{x}_t \to \boldsymbol{x}_{t-\Delta t}$ (Xiao et al., 2022; Wang et al., 2024a), and our start-to-end trajectory matching $\boldsymbol{x}_t \to \boldsymbol{x}_0$, with the latter showing the best performance.

Final training objective combines $\mathcal{L}_{\text{score}}$, $\mathcal{L}_{\text{LPIPS}}$, and $\mathcal{L}_{\text{adv}}$, with empirically adjusted weights.

3.3 Noise and time scheduling

Noise scheduling. In diffusion models, noise scheduling involves progressively adding noise to the data over time by defining specific functions for α_t and σ_t in Eq. 2. This process is crucial as it determines the signal-to-noise ratio, $\lambda_t = \frac{\alpha_t^2}{\sigma_t^2}$, which directly influences training dynamics. Noise scheduling can also be adjusted by scaling the intermediate states $\boldsymbol{x}_t$ with a constant factor $\gamma \in (0, 1]$, which shifts the signal-to-noise ratio downward. This makes training more challenging over time while preserving the shape of the trajectory (Chen, 2023).

In this work, we define α_t and σ_t according to rectified flow formulation, while also scaling $\boldsymbol{x}_t$ by γ, with the value chosen empirically. However, when $\gamma \neq 1$, the variance of $\boldsymbol{x}_t$ changes, which can degrade performance (Karras et al., 2022). To address this, we normalize the denoising input $\boldsymbol{x}_t$ by its variance after scaling, ensuring it preserves unit variance over time (Chen, 2023).

Time scheduling. Another important aspect in diffusion models is time scheduling for both training and sampling, controlled by $\pi(t)$ during training and Δt during sampling, as outlined in Eq. 3 and Eq. 4. A common choice for $\pi(t)$ is the uniform distribution $\mathcal{U}(0, T)$, which applies equal weight to each time step during training. Similarly, uniform time steps $\Delta t = \frac{1}{T}$ are typically used for sampling. However, to improve model performance on more challenging time steps and focus on noisy regimes during sampling, the time scheduling strategy should be adjusted accordingly.

In this work, we sample t from a logit-normal distribution (Atchison & Shen, 1980), which emphasizes intermediate timesteps (Esser et al., 2024). During sampling, we apply a reversed logarithm mapping function ρ_{log}, defined as:

$$\rho_{\text{log}}(t; m, n) = \frac{\log(m) - \log\left(t \cdot (m - n) + n\right)}{\log(m) - \log(n)}, \tag{13}$$

where we set $m = 1$ and $n = 100$, resulting in denser sampling steps early in the inference process.

4 Experiments

We evaluate the effectiveness of ϵ-VAE on image reconstruction and generation tasks using the ImageNet (Deng et al., 2009). The VAE formulation by Esser et al. (2021) serves as a strong baseline due to its widespread use in modern image generative models (Rombach et al., 2022; Peebles & Xie, 2023; Esser et al., 2024). We perform controlled experiments to compare reconstruction and generation quality by varying model scale, latent dimension, downsampling rates, and input resolution.

Model configurations. We use the encoder and discriminator architectures from VQGAN (Esser et al., 2021) and keep consistent across all models. The decoder design follows BigGAN (Brock et al., 2019) for VAE and from ADM (Dhariwal & Nichol, 2021) for ϵ-VAE. Additionally, we experiment with the DiT architecture (Peebles & Xie, 2023) for ϵ-VAE. To evaluate model scaling, we test five decoder variants: base (B), medium (M), large (L), extra-large (XL), and huge (H), by adjusting width and depth accordingly. Further model specifications are provided in Appendix B.1.

We experiment with two encoder configurations: (1) a light-weight version with 6M parameters, a downsampling rate of 16, and 8 latent channels; (2) a standard version based on Stable Diffusion with 34M parameters, a downsampling rate of 8, and 4 latent channels. Configuration (1) is intentionally designed as a more challenging setup and serves as the primary focus of analysis in the paper. For this configuration, we further explore downsampling rates of 4, 8, and 32, as well as latent dimensions of 4, 16, and 32 channels. Both VAE and ϵ-VAE are trained to reconstruct 128×128 images under these controlled conditions. Additionally, we validate our method in the standard setup of Configuration (2) (detailed in Appendix C.2), where we compare it against state-of-the-art VAEs.

Evaluation. We evaluate the autoencoder on both reconstruction and generation quality using Fréchet Inception Distance (FID) (Heusel et al., 2017) as the primary metric, computed on 10,000 validation images. For reconstruction quality (rFID), FID is computed at both training and higher resolutions to assess generalization across resolutions. For generation quality (FID), we generate latents from the trained autoencoders and use them to train the DiT-XL/2 latent generative model (Peebles & Xie, 2023). This latent model remains fixed across all generation experiments, meaning improved autoencoder latents directly enhance generation quality. We also report Inception Score (IS) (Salimans et al., 2016) and Precision/Recall (Kynkäänniemi et al., 2019) as secondary metrics.

4.1 Reconstruction quality

Decoder architecture. We explore two major architectural designs: the UNet-based architecture from ADM (Dhariwal & Nichol, 2021) and the Transformer-based DiT (Peebles & Xie, 2023). We compare various model sizes–ADM:$\{$B, M, L, XL, H$\}$ and DiT:$\{$S, B, L, XL$\}$ with patch sizes of $\{$4, 8$\}$. The results are summarized in Figure 2 (left). ADM consistently outperforms DiT across the board. While we observe rFID improvements in DiT when increasing the number of tokens by reducing patch size, this comes with significant computational overhead. The overall result aligns with the original design intentions: ADM for pixel-level generation and DiT for latent-level generation. For the following experiments, we use the ADM architecture for our diffusion decoder.

Compression rate. Compression can be achieved by adjusting either the channel dimensions of the latents or the downsampling factor of the encoder. In Figure 2 (middle and right), we compare VAE and ϵ-VAE across these two aspects. The results show that ϵ-VAE consistently outperforms VAE in terms of rFID, particularly as the compression ratio increases. Specifically, as shown on the middle graph, ϵ-VAE achieves lower rFIDs than VAE across all channel dimensions, with a notable gap at lower dimensions (4 and 8). On the right graph, ϵ-VAE maintains lower rFIDs than VAE even as the downsampling factor increases, with the gap widening significantly at larger factors (16 and 32). Furthermore, ϵ-VAE delivers comparable or superior rFIDs even when the compression ratio is doubled, demonstrating its robustness and effectiveness in high-compression scenarios.

Model scaling. We investigate the impact of model scaling by comparing VAE and ϵ-VAE across five model variants, all trained and evaluated at a resolution of 128×128, as summarized in Table 1. The results demonstrate that ϵ-VAE consistently achieves significantly better rFID scores than VAE, with an average relative improvement of over 40%, and even the smallest ϵ-VAE model outperforms VAE at largest scale. While the U-Net-based decoder of ϵ-VAE has about twice as many parameters as standard decoder of VAE, grouping models by similar sizes, highlighted in blue, red, and green, shows that performance gains are not simply due to increased model parameters.

Table 1: **Model scaling and resolution generalization analysis.** Five model variants are trained and evaluated. Δ_{rFID} represents the absolute differences (or relative ratio) in rFID between the corresponding model size variants of VAE and ϵ-VAE. † denotes resolution generalization experiments. To fairly evaluate the impact of ϵ-VAE under controlled model parameters, we highlight three groups of model variants with comparable parameters, using blue, red, and green.

Models	$\mathcal{G}$ params (M)	ImageNet $\times$128		ImageNet $\times$256 †		ImageNet $\times$512 †	
		rFID ↓	Δ_{rFID}	rFID ↓	Δ_{rFID}	rFID ↓	Δ_{rFID}
VAE (B)	10.14	11.15	-	5.74	-	3.69	-
VAE (M)	22.79	9.26	-	4.63	-	2.69	-
VAE (L)	40.48	8.49	-	4.78	-	2.78	-
VAE (XL)	65.27	7.58	-	4.42	-	2.41	-
VAE (H)	161.81	7.12	-	4.29	-	2.37	-
ϵ-VAE (B)	20.63	6.24	4.91 (44.0%)	3.90	1.84 (32.0%)	2.06	1.63 (44.2%)
ϵ-VAE (M)	49.33	5.42	3.84 (41.5%)	2.79	1.84 (39.7%)	2.02	0.67 (24.9%)
ϵ-VAE (L)	88.98	4.71	3.78 (44.5%)	2.60	2.03 (43.8%)	1.92	0.86 (30.9%)
ϵ-VAE (XL)	140.63	4.18	3.40 (44.9%)	2.38	2.04 (46.2%)	1.82	0.59 (24.5%)
ϵ-VAE (H)	355.62	4.04	3.08 (43.3%)	2.31	1.98 (46.2%)	1.78	0.59 (24.9%)

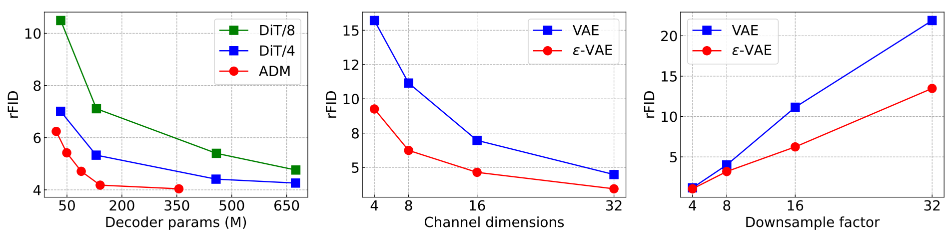

Figure 2: **Architecture and compression analysis.** The ϵ-VAE decoder uses either a UNet-based ADM or Transformer-based DiT (left). ϵ-VAE and VAE under different compression rates by varying latent channel dimensions (middle) or encoder downsampling factors (right).

Resolution generalization. A notable feature of conventional autocencoders is their capacity to generalize and reconstruct images at higher resolutions during inference (Rombach et al., 2022). To assess this, we conduct inference on images with resolutions of 256 $\times$ 256 and 512 $\times$ 512, using ϵ-VAE and VAE models trained at 128 $\times$ 128. As shown in Table 1, ϵ-VAE effectively generalizes to higher resolutions, consistently preserving its performance advantage over VAE.

Runtime efficiency. On a Tesla V100 GPU, VAE (M) achieves 114.13 images/sec throughput, while the throughput of ϵ-VAE (B) is 20.68 images/sec when the sampling step is three and increased to 62.94 images/sec if we sample by one step. ϵ-VAE requires more compute costs than VAE due to its U-Net design. We discuss potential directions to improve our runtime efficiency in Sec. 5.

4.2 Generation Quality

Given the trained VAE and ϵ-VAE models, we now evaluate their autoencoding performance. In practice, we first generate latents using the trained autoencoders, then train a new latent generative model based on these representations. The compact, learnable latent space produced by the encoder enhances the learning efficiency of latent generative model, while effective decoding of the sampled latents ensures high-quality outputs. Thus, both the encoding and decoding capabilities of autoencoder contribute to the overall generative performance. For this evaluation, we perform standard unconditional image generation tasks using the DiT-XL/2 model as our latent generative model (Peebles & Xie, 2023). Further details on the training setup are provided in Appendix B.3.

Table 2 presents the image generation results of VAE and ϵ-VAE at resolutions of 128 $\times$ 128 and 256 $\times$ 256. The results show that ϵ-VAE consistently outperforms VAE across all model scales. Notably, ϵ-VAE (B) surpasses VAE (H), consistent with our earlier findings in Sec. 4.1. These results confirm that the performance gains from the reconstruction task successfully transfer to the generation task, further validating the effectiveness of ϵ-VAE.

It is important to note that the primary focus of this experiment is not to achieve state-of-the-art generation results, but to provide a fair comparison of ϵ-VAE's autoencoding capabilities under

Table 2: **Image generation quality.** The DiT-XL/2 is trained on latents provided by the trained autoencoders, VAE and ϵ-VAE, with varying model sizes using ImageNet. We evaluate the generation quality at resolutions of 128×128 and 256×256 using four standard metrics. Additionally, we report rFID to determine if the improvement trend observed in reconstruction task extends to the generation task. We highlight three groups of model variants with comparable parameters.

Models	ImageNet $\times 128$					ImageNet $\times 256$				
	rFID ↓	FID ↓	IS ↑	Prec. ↑	Rec. ↑	rFID ↓	FID ↓	IS ↑	Prec. ↑	Rec. ↑
VAE (B)	11.15	36.8	17.9	0.48	0.53	5.74	46.6	23.4	0.45	0.56
VAE (M)	9.26	34.6	18.2	0.49	0.55	4.63	44.7	23.8	0.47	0.58
VAE (L)	8.49	33.9	18.4	0.50	0.56	4.78	44.3	24.7	0.47	0.59
VAE (XL)	7.58	31.7	19.3	0.51	0.57	4.42	43.1	24.9	0.47	0.59
VAE (H)	7.12	30.9	19.8	0.52	0.57	4.29	41.6	25.9	0.48	0.59
ϵ-VAE (B)	6.24	29.5	20.7	0.53	0.59	3.90	39.5	25.2	0.46	0.61
ϵ-VAE (M)	5.42	27.6	21.2	0.55	0.59	2.79	35.4	26.2	0.51	0.62
ϵ-VAE (L)	4.71	27.3	22.1	0.55	0.59	2.60	34.8	26.5	0.51	0.63
ϵ-VAE (XL)	4.18	25.3	22.7	0.55	0.59	2.38	34.0	27.4	0.53	0.63
ϵ-VAE (H)	4.04	24.9	23.0	0.56	0.60	2.31	33.2	27.5	0.54	0.64

a controlled experimental setup. We demonstrate that our approach consistently outperforms the leading autoencoding method (Esser et al., 2021) across varying model scales and input resolutions.

4.3 Ablation Studies

We conduct a component-wise analysis to validate our key design choices. We evaluate the reconstruction quality (rFID) and sampling efficiency (NFE). The results are summarized in Table 3.

Baseline. Our evaluation begins with a baseline model: an autoencoder with a diffusion decoder, trained solely using the score matching objective. This baseline follows the vanilla diffusion setup from Ho et al. (2020), including their UNet architecture, parameterization, and training configurations, while extending to a conditional form as described in Eq. 8. Building on this baseline, we progressively introduce updates and evaluate the impact of our proposed method.

Impact of proposals. In **(a)**, transitioning from standard diffusion to rectified flow (Liu et al., 2023) straightens the optimization path, resulting in significant gains in rFID scores and NFE. In **(b)**, adopting a logit-normal time step distribution optimizes rectified flow training (Esser et al., 2024), further improving both rFID scores and NFE. In **(c)**, updates to the UNet architecture (Nichol & Dhariwal, 2021) contribute to enhanced rFID scores. In **(d)**, LPIPS loss is applied to match reconstructions $\hat{\boldsymbol{x}}_0^t$ with real images $\boldsymbol{x}_0$. In **(e)**, adversarial trajectory matching loss aligns $(\hat{\boldsymbol{x}}_0^t, \boldsymbol{x}_t)$ with $(\boldsymbol{x}_0, \boldsymbol{x}_t)$, the target transition in rectified flow. Both objectives improve model understanding of the underlying optimization trajectory, significantly enhancing rFID scores and NFE.

Up to this point, with the full implementation of Eq. 1, we can compare our proposal with the VAE (B) model, which achieves an rFID score of 11.15. Our model, with a score of 8.24, already surpasses this baseline. We further improve performance by optimizing noise and time scheduling within our framework, as described next.

In **(f)**, scaling $\boldsymbol{x}_t$ reduces the signal-to-noise ratio (Chen, 2023), presenting challenges for more effective learning during training. Figure 3 (middle) demonstrates that a scaling factor of 0.6 produces the best results. Finally, in **(g)**, reversed logarithmic time step spacing during inference allows for denser evaluations in noisier regions. Figure 3 (right) demonstrates that this method provides more stable sampling in the lower NFE regime compared to the original uniform spacing.

5 Discussion

Distribution-aware compression. Traditional image compression methods optimize the rate-distortion trade-off (Shannon et al., 1959), prioritizing compactness over input fidelity. Building on this, we also aim to capture the broader input distribution during compression, generating compact representations suitable for latent generative models. This approach introduces an additional dimension to the trade-off, perception or distribution fidelity (Blau & Michaeli, 2018), which aligns more closely with the rate-distortion-perception framework (Blau & Michaeli, 2019).

Table 3: **Ablation study on key design choices for the ϵ-VAE diffusion decoder.** A systematic evaluation of the architecture ($\star$), objectives (†), and noise & time scheduling (§). Each row progressively modifies or builds upon the baseline decoder, showing improvements in performance.

Ablation	**NFE**	**rFID**
Baseline: DDPM-based diffusion decoder	1,000	28.22
† *(a)* Diffusion $\rightarrow$ Rectified flow parameterization	100	24.11
§ *(b)* Uniform $\rightarrow$ Logit-normal time step sampling during training	50	23.44
$\star$ *(c)* DDPM UNet $\rightarrow$ ADM UNet	50	22.04
† *(d)* Perceptual matching on $\hat{\boldsymbol{x}}_0^t$ and $\boldsymbol{x}_0$	10	11.76
† *(e)* Adversarial denoising trajectory matching on $(\hat{\boldsymbol{x}}_0^t, \boldsymbol{x}_t)$ and $(\boldsymbol{x}_0, \boldsymbol{x}_t)$	5	8.24
§ *(f)* Scale $\boldsymbol{x}_t$ by $\gamma = 0.6$	5	7.08
§ *(g)* Uniform $\rightarrow$ Reversed logarithm time spacing during inference	3	6.24

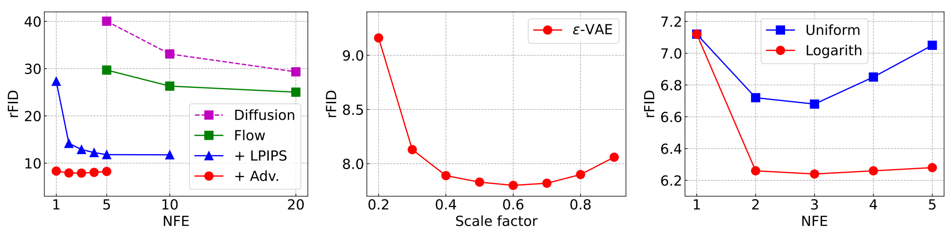

Figure 3: **Impact of our major diffusion decoder designs.** Improved training objectives, particularly perceptual matching loss and adversarial denoising trajectory matching loss, significantly contribute to better rFID scores and NFE (left). Effective noise scheduling by modulating the scaling factor γ further enhances rFID, with an optimum value of 0.6 in our experiments (middle). Lastly, adjusting time step spacing during inference ensures stable sampling in low NFE regimes (right).

Iterative and stochastic decoding. A key question within the rate-distortion-perception trade-off is whether the iterative, stochastic nature of diffusion decoding offers advantages over traditional single-step, deterministic methods (Kingma, 2013). The strengths of diffusion (Ho et al., 2020) lie in its iterative process, which progressively refines the latent space for more accurate reconstructions, while stochasticity allows for capturing complex variations within the distribution. Although iterative methods may appear less efficient, our formulation is optimized to achieve optimal results in just three steps and also supports single-step decoding, ensuring decoding efficiency remains practical (see Figure 3 (left)). While stochasticity might suggest the risk of "hallucination" in reconstructions, the outputs remain faithful to the underlying distribution by design, producing perceptually plausible results. This advantage is particularly evident under extreme compression scenarios (see Figure 4), with the degree of stochasticity adapting based on compression levels (see Figure 5).

Scalability. As discussed in Section 4.1, our diffusion-based decoding method maintains the resolution generalizability typically found in standard autoencoders. This feature is highly practical: the autoencoder is trained on lower-resolution images, while the subsequent latent generative model is trained on latents derived from higher-resolution inputs. However, we acknowledge that memory overhead and throughput become concerns with our UNet-based diffusion decoder, especially for high-resolution inputs. This challenge becomes more pronounced as models, datasets, or resolutions scale up. A promising future direction is patch-based diffusion (Ding et al., 2024; Wang et al., 2024b), which partitions the input into smaller, independently processed patches. This approach has the potential to reduce memory usage and enable faster parallel decoding.

6 Conclusion

We present ϵ-VAE, an effective visual tokenization framework that introduces a diffusion decoder into standard autoencoders, turning single-step decoding into a multi-step probabilistic process. By exploring key design choices in modeling, objectives, and diffusion training, we demonstrate significant performance improvements. Our approach outperforms traditional visual autoencoders in both reconstruction and generation quality, particularly in high-compression scenarios. We hope our concept of iterative generation during decoding inspires further advancements in visual autoencoding.

Figure 4: **Reconstruction results with varying downsampling ratios.** ϵ-VAE maintains both high fidelity and perceptual quality, even under extreme downsampling conditions, whereas VAE fails to preserve semantic integrity. *Best viewed when zoomed-in and in color.*

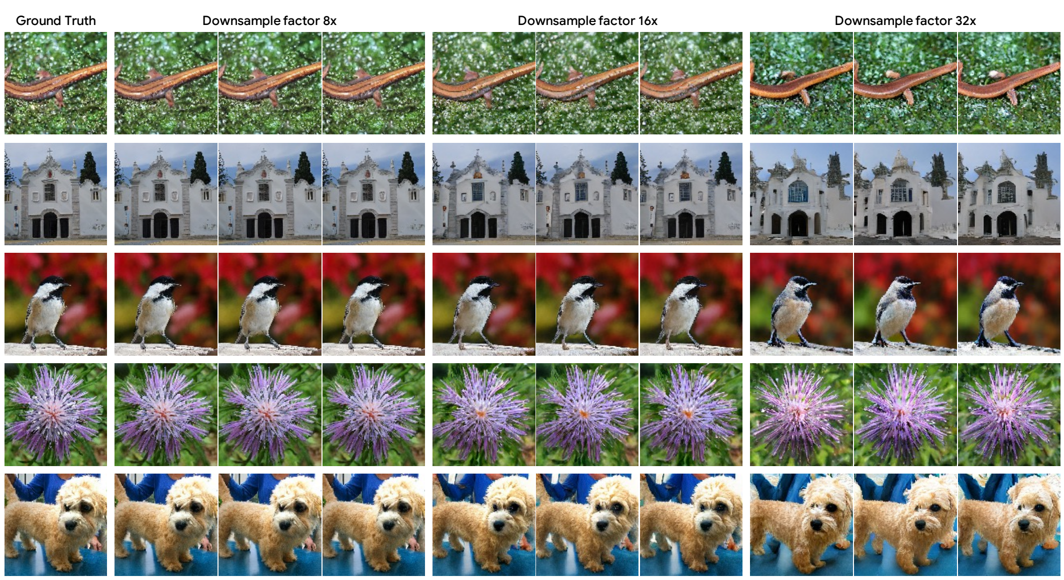

Figure 5: **ϵ-VAE reconstruction results with varying random seeds and downsampling ratios.** At lower compression levels, the reconstruction behaves more deterministically, whereas higher compression introduces stochasticity, enabling more flexible reconstruction of plausible inputs. *Best viewed when zoomed-in and in color.*

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *ICLR*, 2023.

Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, pp. 214–223, 2017.

Jhon Atchison and Sheng M Shen. Logistic-normal distributions: Some properties and uses. *Biometrika*, 67(2):261–272, 1980.

Jason Baldridge, Jakob Bauer, Mukul Bhutani, Nicole Brichtova, Andrew Bunner, Kelvin Chan, Yichang Chen, Sander Dieleman, Yuqing Du, Zach Eaton-Rosen, et al. Imagen 3. *arXiv preprint arXiv:2408.07009*, 2024.

James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *OpenAI Blog*, 2023. URL `https://cdn.openai.com/papers/dall-e-3.pdf`.

Vighnesh Birodkar, Gabriel Barcik, James Lyon, Sergey Ioffe, David Minnen, and Joshua V Dillon. Sample what you cant compress. *arXiv preprint arXiv:2409.02529*, 2024.

Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *CVPR*, pp. 6228–6237, 2018.

Yochai Blau and Tomer Michaeli. Rethinking lossy compression: The rate-distortion-perception tradeoff. In *ICML*, pp. 675–685, 2019.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/jax-ml/jax`.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019.

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. *OpenAI Blog*, 2024. URL `https://openai.com/research/video-generation-models-as-world-simulators`.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. MaskGIT: Masked generative image transformer. In *CVPR*, pp. 11315–11325, 2022.

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *ICML*, pp. 1691–1703, 2020.

Ting Chen. On the importance of noise scheduling for diffusion models. *arXiv preprint arXiv:2301.10972*, 2023.

Ting Chen, Lala Li, Saurabh Saxena, Geoffrey Hinton, and David J Fleet. A generalist framework for panoptic segmentation of images and videos. In *ICCV*, pp. 909–919, 2023.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, pp. 248–255, 2009.

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

Zheng Ding, Mengqi Zhang, Jiajun Wu, and Zhuowen Tu. Patched denoising diffusion models for high-resolution image synthesis. In *ICLR*, 2024.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*, 2024.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, pp. 12873–12883, 2021.

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. In *ICML*, 2024.

Leonhard Euler. *Institutionum calculi integralis*, volume 4. impensis Academiae imperialis scientiarum, 1845.

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

Agrim Gupta, Lijun Yu, Kihyuk Sohn, Xiuye Gu, Meera Hahn, Li Fei-Fei, Irfan Essa, Lu Jiang, and José Lezama. Photorealistic video generation with diffusion models. *arXiv preprint arXiv:2312.06662*, 2023.

Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2024. URL `http://github.com/google/flax`.

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. In *NeurIPS*, 2017.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

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

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

Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *Journal of Machine Learning Research*, 23(47):1–33, 2022.

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*, 2023a.

Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *ICML*, pp. 13213–13232, 2023b.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, pp. 4401–4410, 2019.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *NeurIPS*, 2022.

Diederik Kingma and Ruiqi Gao. Understanding diffusion objectives as the elbo with simple data augmentation. In *NeurIPS*, 2024.

Diederik P Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Dan Kondratyuk, Lijun Yu, Xiuye Gu, José Lezama, Jonathan Huang, Rachel Hornung, Hartwig Adam, Hassan Akbari, Yair Alon, Vighnesh Birodkar, et al. VideoPoet: A large language model for zero-shot video generation. In *ICML*, 2024.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint arXiv:1804.10959*, 2018.

Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. In *NeurIPS*, 2019.

Sangyun Lee, Zinan Lin, and Giulia Fanti. Improving the training of rectified flows. *arXiv preprint arXiv:2405.20320*, 2024.

Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *arXiv preprint arXiv:2406.11838*, 2024.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, pp. 740–755, 2014.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *ICLR*, 2022.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023.

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

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. *arXiv preprint arXiv:2401.08740*, 2024.

Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for GANs do actually converge? In *ICML*, pp. 3481–3490, 2018.

Thuan Hoang Nguyen and Anh Tran. SwiftBrush: One-step text-to-image diffusion model with variational score distillation. In *CVPR*, pp. 7807–7816, 2024.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *ICML*, pp. 8162–8171, 2021.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, pp. 4195–4205, 2023.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *AAAI*, 2018.

Pablo Pernias, Dominic Rampas, Mats L Richter, Christopher J Pal, and Marc Aubreville. Würstchen: An efficient architecture for large-scale text-to-image diffusion models. In *ICLR*, 2024.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *ICLR*, 2024.

Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *CVPR*, pp. 10619–10629, 2022.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *OpenAI Blog*, 2018.

Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with VQ-VAE-2. In *NeurIPS*, 2019.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pp. 10684–10695, 2022.

Seyedmorteza Sadat, Jakob Buhmann, Derek Bradley, Otmar Hilliges, and Romann M Weber. LiteVAE: Lightweight and efficient variational autoencoders for latent diffusion models. *arXiv preprint arXiv:2405.14477*, 2024.

Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH*, pp. 1–10, 2022a.

Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE TPAMI*, 45(4):4713–4726, 2022b.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *NeurIPS*, 2016.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *ECCV*, pp. 87–103, 2024.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

Claude E Shannon et al. Coding theorems for a discrete source with a fidelity criterion. *IRE Nat. Conv. Rec*, 4(142-163):1, 1959.

Jie Shi, Chenfei Wu, Jian Liang, Xiang Liu, and Nan Duan. DiVAE: Photorealistic images synthesis with denoising diffusion decoder. *arXiv preprint arXiv:2206.00386*, 2022.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *NeurIPS*, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021.

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.

Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *NeurIPS*, 2016.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. In *NeurIPS*, 2017.

Fu-Yun Wang, Zhaoyang Huang, Alexander William Bergman, Dazhong Shen, Peng Gao, Michael Lingelbach, Keqiang Sun, Weikang Bian, Guanglu Song, Yu Liu, et al. Phased consistency model. *arXiv preprint arXiv:2405.18407*, 2024a.

Zhendong Wang, Yifan Jiang, Huangjie Zheng, Peihao Wang, Pengcheng He, Zhangyang "Atlas" Wang, Weizhu Chen, and Mingyuan Zhou. Patch diffusion: Faster and more data-efficient training of diffusion models. In *NeurIPS*, 2024b.

Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, pp. 3–19, 2018.

Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion GANs. In *ICLR*, 2022.

Ruihan Yang and Stephan Mandt. Lossy image compression with conditional diffusion models. In *NeurIPS*, 2024.

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 VQ-GAN. In *ICLR*, 2022.

Lijun Yu, Yong Cheng, Kihyuk Sohn, José Lezama, Han Zhang, Huiwen Chang, Alexander G Hauptmann, Ming-Hsuan Yang, Yuan Hao, Irfan Essa, et al. MAGVIT: Masked generative video transformer. In *CVPR*, pp. 10459–10469, 2023.

Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Agrim Gupta, Xiuye Gu, Alexander G Hauptmann, et al. Language model beats diffusion–tokenizer is key to visual generation. In *ICLR*, 2024a.

Qihang Yu, Mark Weber, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. An image is worth 32 tokens for reconstruction and generation. *arXiv preprint arXiv:2406.07550*, 2024b.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pp. 586–595, 2018.

Yue Zhao, Yuanjun Xiong, and Philipp Krähenbühl. Image and video tokenization with binary spherical quantization. *arXiv preprint arXiv:2406.07548*, 2024.

A RELATED WORK

Image tokenization. Image tokenization is crucial for effective generative modeling, transforming images into compact, structured representations. A common approach employs an autoencoder framework (Hinton & Salakhutdinov, 2006), where the encoder compresses images into low-dimensional latent representations, and the decoder reconstructs the original input. These latent representations can be either discrete commonly used in autoregressive models (Van den Oord et al., 2016; Van Den Oord et al., 2017; Chen et al., 2020; Chang et al., 2022; Yu et al., 2023; Kondratyuk et al., 2024), or continuous, as found in diffusion models (Ho et al., 2020; Dhariwal & Nichol, 2021; Rombach et al., 2022; Peebles & Xie, 2023; Gupta et al., 2023; Brooks et al., 2024). The foundational form of visual autoencoding today originates from Van Den Oord et al. (2017). While advancements have been made in modeling (Yu et al., 2022; 2024b), objectives (Zhang et al., 2018; Karras et al., 2019; Esser et al., 2021), and quantization methods (Yu et al., 2024a; Zhao et al., 2024), the core encoding-and-decoding scheme remains largely the same.

In this work, we propose a new perspective by replacing the traditional decoder with a diffusion process. Specifically, our new formulation retains the encoder but introduces a conditional diffusion decoder. Within this framework, we systematically study various design choices, resulting in a significantly enhanced autoencoding setup.

Additionally, we refer to the recent work MAR (Li et al., 2024), which leverages diffusion to model per-token distribution in autoregressive frameworks. In contrast, our approach models the overall input distribution in autoencoders using diffusion. This difference leads to distinct applications of diffusion during generation. For instance, MAR generates samples autoregressively, decoding each token iteratively using diffusion, token by token. In our method, we first sample all tokens from the downstream generative model and then decode them iteratively using diffusion as a whole.

Image compression. Our work shares similarities with recent image compression approaches that leverage diffusion models. For example, Hoogeboom et al. (2023a); Birodkar et al. (2024) use diffusion to refine autoencoder residuals, enhancing high-frequency details. Yang & Mandt (2024) employs a diffusion decoder conditioned on quantized discrete codes and omits the GAN loss. However, these methods primarily focus on the traditional rate-distortion tradeoff, balancing rate (compactness) and distortion (input fidelity) (Shannon et al., 1959), with the goal of storing and transmitting data efficiently without significant loss of information.

In this work, we emphasize perception (distribution fidelity) alongside the rate-distortion tradeoff, ensuring that reconstructions more closely align with the overall data distribution (Heusel et al., 2017; Zhang et al., 2018; Blau & Michaeli, 2019), thereby enhancing the decoded results from the sampled latents of downstream generative models. We achieve this by directly integrating the diffusion process into the decoder, unlike Hoogeboom et al. (2023a); Birodkar et al. (2024). Moreover, unlike Yang & Mandt (2024), we do not impose strict rate-distortion regularization in the latent space and allow the GAN loss to synergize with our approach.

Diffusion decoder. Several studies (Preechakul et al., 2022; Shi et al., 2022; Pernias et al., 2024; Nguyen & Tran, 2024; Sauer et al., 2024; Luo et al., 2023) have explored diffusion decoders conditioned on compressed latents of the input, which are relevant to our work. We outline the key differences between these works and ϵ-VAE: First, prior works have not fully leveraged the synergy between diffusion decoders and standard VAE training objectives. In this work, we enhance state-of-the-art VAE objectives by replacing the reconstruction loss with a score matching loss and adapting LPIPS and GAN losses to ensure compatibility with diffusion decoders. These changes yield significant improvements in autoencoding performance, as evidenced by lower rFID scores and faster inference. Second, we are the first to investigate various parameterizations (*e.g.*, epsilon and velocity) and demonstrate that modern velocity parameterization, coupled with optimized train and test-time noise scheduling, provides substantial benefits. These enhancements improve both reconstruction performance and sampling efficiency. Third, previous diffusion-based decoders (Preechakul et al., 2022; Shi et al., 2022; Pernias et al., 2024), which often rely on ad-hoc techniques like distillation or consistency regularization to speed up inference (Nguyen & Tran, 2024; Sauer et al., 2024; Luo et al., 2023), our approach achieves fast decoding (1 to 3 steps) without such techniques. This is made possible by integrating our proposed objectives and parameterizations. Last but not least, ϵ-VAE exhibits strong resolution generalization capabilities, a key property of standard VAEs. In contrast, models like DiffusionAE (Preechakul et al., 2022) and DiVAE (Shi et al., 2022) either lack this ability or are inherently limited. For example, DiVAE's bottleneck add/concat design restricts its capacity to generalize across resolutions.

Another closely related work, SWYCC (Birodkar et al., 2024), also explores joint learning of continuous encoders and decoders using a diffusion model. However, SWYCC differs fundamentally from our approach: it replaces the GAN loss with a diffusion-based loss, while we focus on identifying optimal synergies between traditional autoencoding losses (including GAN loss) and diffusion-based decoding. Our goal is to identify an optimal strategy for combining these elements, rather than simply substituting one for another.

Image generation. Recent advances in image generation span a wide range of approaches, including VAEs (Kingma, 2013), GANs (Goodfellow et al., 2014), autoregressive models (Chen et al., 2020) and diffusion models (Song et al., 2021; Ho et al., 2020). Among these, diffusion models have emerged as the leading approach for generating high-dimensional data such as images (Saharia et al., 2022a; Baldridge et al., 2024; Esser et al., 2024) and videos (Brooks et al., 2024; Gupta et al., 2023), where the gradual refinement of global structure is crucial. The current focus in diffusion-based generative models lies in advancing architectures (Rombach et al., 2022; Peebles & Xie, 2023; Hoogeboom et al., 2023b), parameterizations (Karras et al., 2022; Kingma & Gao, 2024; Ma et al., 2024; Esser et al., 2024), or better training dynamics (Nichol & Dhariwal, 2021; Chen, 2023; Chen et al., 2023). However, tokenization, an essential component in modern diffusion models, often receives less attention.

In this work, we focus on providing compact continuous latents without applying quantization during autoencoder training (Rombach et al., 2022), as they have been shown to be effective in state-of-the-art latent diffusion models (Rombach et al., 2022; Saharia et al., 2022a; Peebles & Xie, 2023; Esser et al., 2024; Baldridge et al., 2024). We compare our autoencoding performance against the baseline approach (Esser et al., 2021) using the DiT framework (Peebles & Xie, 2023) as the downstream generative model.

B Experiment Setups

In this section, we provide additional details on our experiment configurations for reproducibility.

B.1 Model Specifications

Table 4 summarizes the primary architecture details for each decoder variant. The channel dimension is the number of channels of the first U-Net layer, while the depth multipliers are the multipliers for subsequent resolutions. The number of residual blocks denotes the number of residual stacks contained in each resolution.

Table 4: **Hyper-parameters for decoder variants.**

Models	Channel dim.	Depth multipliers	# Residual blocks
Base (B)	64	$\{1, 1, 2, 2, 4\}$	2
Medium (M)	96	$\{1, 1, 2, 2, 4\}$	2
Large (L)	128	$\{1, 1, 2, 2, 4\}$	2
Extra-large (XL)	128	$\{1, 1, 2, 2, 4\}$	4
Huge (H)	256	$\{1, 1, 2, 2, 4\}$	2

B.2 Additional Implementation Details

During the training of discriminators, Esser et al. (2021) introduced an adaptive weighting strategy for λ_{adv}. However, we notice that this adaptive weighting does not introduce any benefit which is consistent with the observation made by Sadat et al. (2024). Thus, we set $\lambda_{\text{adv}} = 0.5$ in the experiments for more stable model training across different configurations.

Training. The autoencoder loss follows Eq. 1, with weights set to $\lambda_{\text{LPIPS}} = 0.5$ and $\lambda_{\text{adv}} = 0.5$. We use the Adam optimizer (Kingma & Ba, 2015) with $\beta_1 = 0$ and $\beta_2 = 0.999$, applying a linear

Table 5: **Additional image reconstruction results on ImageNet 128×128.**

Configurations	NFE	rFID
Baseline (c) in Table 3:		
Inject conditioning by channel-wise concatenation	50	22.04
Inject conditioning by AdaGN	50	22.01
Baseline (e) in Table 3:		
Matching the distribution of $\hat{\boldsymbol{x}}_0^t$ and $\boldsymbol{x}_0$	-	N/A
Matching the trajectory of $\boldsymbol{x}_t \to \boldsymbol{x}_0$	5	8.24
Matching the trajectory of $\boldsymbol{x}_t \to \boldsymbol{x}_{t-\Delta t}$	5	10.53

learning rate warmup over the first 5,000 steps, followed by a constant rate of 0.0001 for a total of one million steps. The batch size is 256, with data augmentations including random cropping and horizontal flipping. An exponential moving average of model weights is maintained with a decay rate of 0.999. All models are implemented in JAX/Flax (Bradbury et al., 2018; Heek et al., 2024) and trained on TPU-v5lite pods.

B.3 Latent diffusion model

We follow the setting in Peebles & Xie (2023) to train the latent diffusion models for unconditional image generation on the ImageNet dataset. The DiT-XL/2 architecture is used for all experiments. The diffusion hyperparameters from ADM (Dhariwal & Nichol, 2021) are kept. To be specific, we use a $t_{\text{max}} = 1000$ linear variance schedule ranging from 0.0001 to 0.02, and results are generated using 250 DDPM sampling steps. All models are trained with Adam (Kingma & Ba, 2015) with no weight decay. We use a constant learning rate of 0.0001 and a batch size of 256. Horizontal flipping and random cropping are used for data augmentation. We maintain an exponential moving average of DiT weights over training with a decay of 0.9999. We use identical training hyperparameters across all experiments and train models for one million steps in total. No classifier-free guidance (Ho & Salimans, 2022) is employed since we target unconditional generation.

C Additional experimental results

C.1 Results under encoder configuration (1)

Conditioning. In addition to injecting conditioning via channel-wise concatenation, we explore providing conditioning to the diffusion model by adaptive group normalization (AdaGN) (Nichol & Dhariwal, 2021; Dhariwal & Nichol, 2021). To achieve this, we resize the conditioning (*i.e.*, encoded latents) via bilinear sampling to the desired resolution of each stage in the U-Net model, and incorporates it into each residual block after a group normalization operation (Wu & He, 2018). This is similar to adaptive instance norm (Karras et al., 2019) and FiLM (Perez et al., 2018). We report the results in Table 5 (top), where we find that channel-wise concatenation and AdaGN obtain similar reconstruction quality in terms of rFID. Because of the additional computational cost required by AdaGN, we thus apply channel-wise concatenation in our model by default.

Trajectory matching. The proposed denoising trajectory matching objective matches the start-to-end trajectory $\boldsymbol{x}_t \to \boldsymbol{x}_0$ by default. One alternative choice is to directly matching the distribution of $\hat{\boldsymbol{x}}_0^t$ and $\boldsymbol{x}_0$ without coupling on $\boldsymbol{x}_t$. However, we find this formulation leads to unstable training and could not produce reasonable results. Here, we present the results when matching the trajectory of $\boldsymbol{x}_t \to \boldsymbol{x}_{t-\Delta t}$, which is commonly used in previous work (Xiao et al., 2022; Wang et al., 2024a). Specifically, for each timestep t during training, we randomly sample a step Δt from $(0, t)$. Then, we construct the real trajectory by computing $\boldsymbol{x}_{t-\Delta t}$ via Eq. 5 and concatenating it with $\boldsymbol{x}_t$, while the fake trajectory is obtained in a similar way but using Eq. 10 instead. Table 5 (bottom) shows the comparison. We observe that matching trajectory $\boldsymbol{x}_t \to \boldsymbol{x}_0$ yields better performance than matching trajectory $\boldsymbol{x}_t \to \boldsymbol{x}_{t-\Delta t}$, confirming the effectiveness of the proposed objective which is designed for the rectified flow formulation.

Table 6: **Comparisons with state-of-the-art image autoencoders.** The results are computed on 256×256 ImageNet 50K validation set and COCO-2017 5K validation set.

Models	$\mathcal{G}$ params (M)	Latent dim.	ImageNet (rFID)	COCO (rFID)
VQGAN (Esser et al., 2021)	49.49	4	1.44	6.58
ViT-VQGAN (Yu et al., 2022)	32	32	1.28	-
LlamaGen (Sun et al., 2024)	49.49	8	0.59	4.19
SD-VAE	49.49	4	0.74	4.45
SDXL-VAE (Podell et al., 2024)	49.49	4	0.68	4.07
OAI-VAE (Betker et al., 2023)	49.49	4	0.81	4.59
ϵ-VAE (B)	20.63	4	0.52	4.24
ϵ-VAE (M)	49.33	4	0.47	3.98
ϵ-VAE (L)	88.98	4	0.45	3.92
ϵ-VAE (XL)	140.63	4	0.43	3.80
ϵ-VAE (H)	355.62	4	**0.38**	**3.65**

Table 7: **Benchmarking class-conditional image generation on ImageNet 256×256.**

VAE used in LDM	VAE downsampling rate	LDM token length	ImageNet FID-50K
SD-VAE	8	32×32	9.42
ϵ-VAE (M)	8	32×32	9.39
ϵ-VAE (M)	16	16×16	10.68

Comparison with plain diffusion ADM. Under the same training setup of Table 3, we directly trained a plain diffusion model (ADM) for comparison, which resulted in rFID score of 38.26. Its conditional form is already provided as a baseline in Table 3, achieving 28.22. This demonstrates that our conditional form $p(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{z})$ offers a better approximation of the true posterior $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ compared to the standard form $p(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$. By further combining LPIPS and GAN loss, we achieve rFID of 8.24, outperforming its VAE counterpart, which achieves 11.15. With better training configurations, our final rFID improves to 6.24. This progression, from plain diffusion ADM to ϵ-VAE, underscores the significance of our proposals and their impact.

C.2 Results under encoder configuration (2)

We provide additional image reconstruction results under the same configuration as VAEs in Stable Diffusion (SD-VAE): a standard encoder with 34M parameters, a downsample rate of 8, and a channel dimension of 4 for 256×256 image reconstruction. We evaluate rFID on the full validation sets of ImageNet and COCO-2017 (Lin et al., 2014), with the results summarized in Table 6.

Our finds reveal that ϵ-VAE outperforms state-of-the-art VAEs when the decoder sizes are comparable (highlighted in red), and its performance can be further improved by scaling up the decoder. This demonstrates the strong model scalability of our framework.

Class-conditional image generation. In addition, we emphasize that when combined with Latent Diffusion Models (LDMs) for class-conditional image generation, ϵ-VAE achieves comparable generation quality while using only 25% of the token length typically required by SD-VAE. To demonstrate this, we train an additional ϵ-VAE (M) under the same configuration as SD-VAE but with double the downsampling rate. We then compare our model to SD-VAE by training DiT/2 in a class-conditional image generation setup (without classifier-free guidance) on ImageNet at 256×256. Following the experimental setup outlined in the DiT paper (Peebles & Xie, 2023), all DiTs are trained for one million steps. The results, presented in Table 7, show that this token length reduction significantly accelerates latent diffusion model generation, reducing overall inference time while maintaining competitive generation quality.

LPIPS, PSNR, and SSIM. We also report additional evaluation metrics: ϵ-VAE achieves 0.152 LPIPS, 25.11 PSNR, and 0.71 SSIM on ImageNet, performing comparably to the standard SD-VAE, which achieves 0.160 LPIPS, 25.83 PSNR, and 0.73 SSIM. As highlighted in Section 5 of the main paper, our approach prioritizes preserving the overall perceptual distribution of images rather than achieving pixel-perfect reconstruction. This aligns with our focus on perception-based compression under high compression rates. Consequently, ϵ-VAE excels in metrics such as rFID, which reflect differences in perceived image distributions, rather than in pixel-level metrics like PSNR and SSIM.

D ADDITIONAL VISUAL RESULTS

Qualitative reconstructions under encoder configuration (1). Figure 8 provides qualitative reconstruction results where we vary the decoder scales. We see that increasing the scale of the model yields significant improvements in visual fidelity, and ϵ-VAE outperforms VAE at corresponding decoder scales. Figure 9 and Figure 10 show additional qualitative results when we vary the downsampling ratios and random seeds.

Qualitative reconstructions under encoder configuration (2). We provide additional visual comparisons between ϵ-VAE and SD-VAE at resolutions of 512×512 (Figure 6) and 256×256 (Figure 7). Our observations indicate that ϵ-VAE delivers significantly better visual quality than SD-VAE, particularly when reconstructing local regions with complex textures or structures, such as human faces and small text.

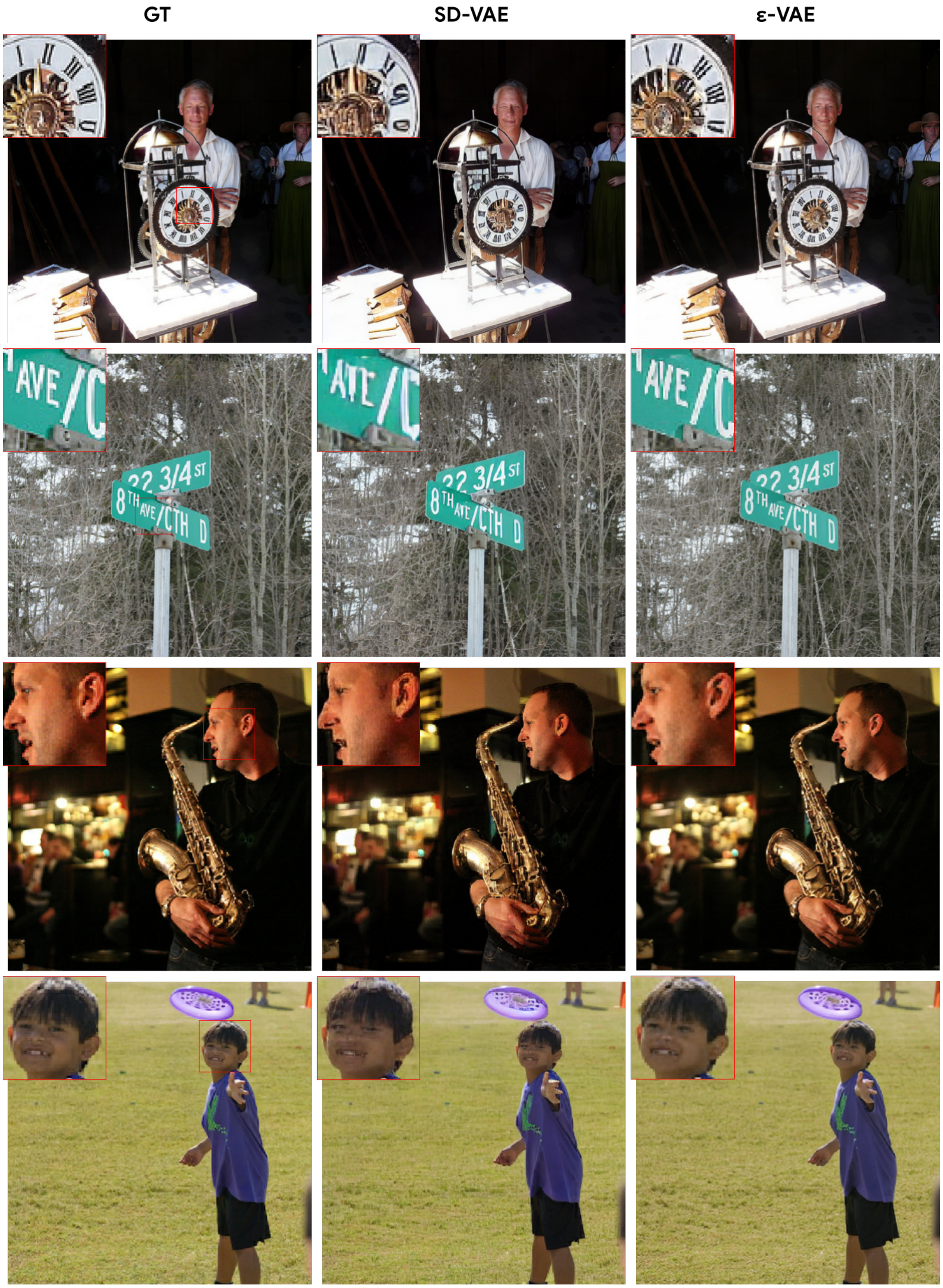

Figure 6: **Image reconstruction results under the SD-VAE (f8-c4) configuration at** 512×512 **resolution.** ϵ-VAE produces more accurate visual details than SD-VAE in the highlighted regions with text or human face. *Best viewed when zoomed-in and in color.*

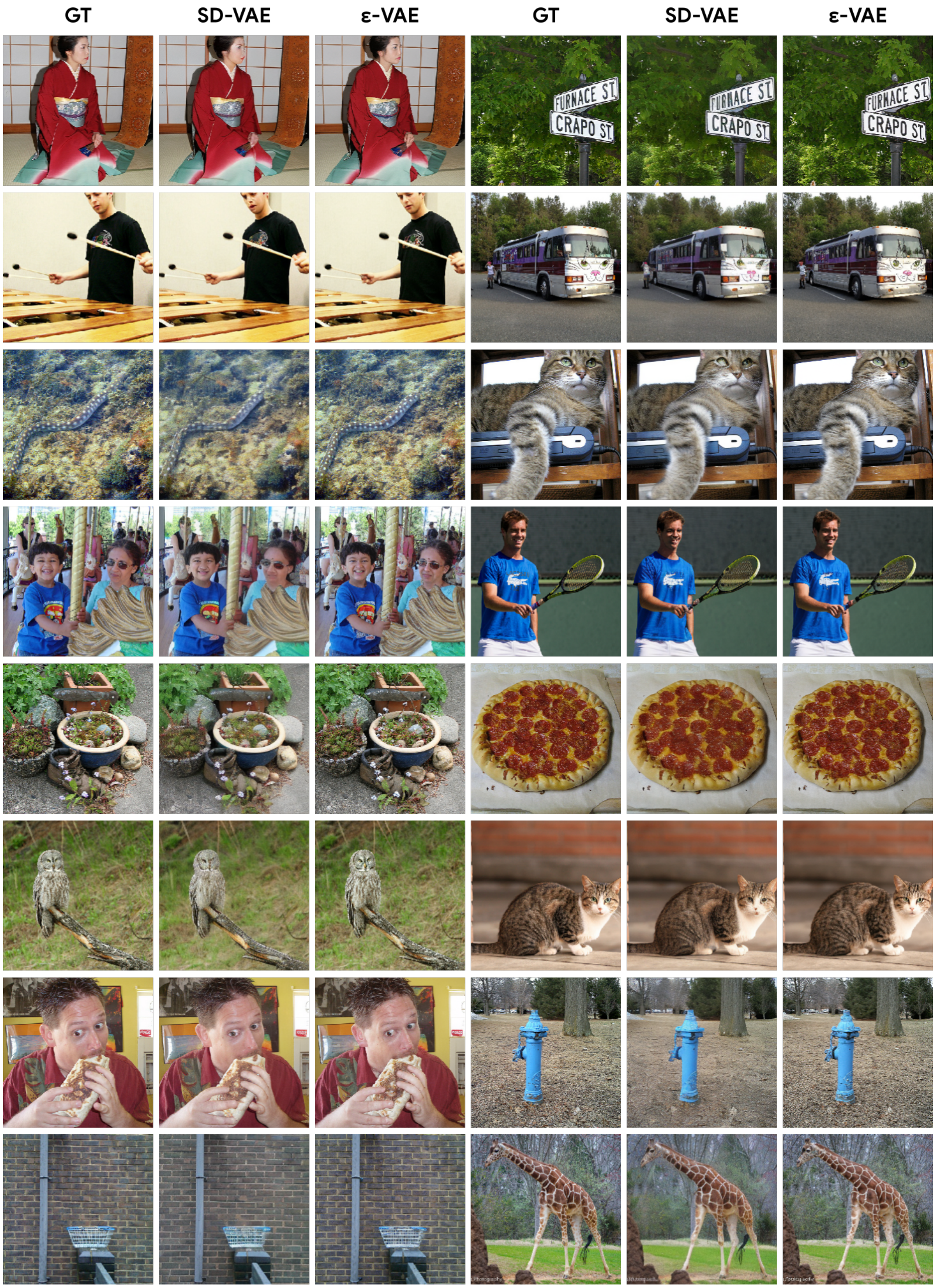

Figure 7: **Image reconstruction results under the SD-VAE (f8-c4) configuration at** 256×256 **resolution.** ϵ-VAE produces significantly better visual details than SD-VAE when reconstructing local regions with complex textures or structures, such as human faces and small texts. *Best viewed when zoomed-in and in color.*

Figure 8: **Reconstruction results with varying decoder size..** ϵ-VAE produces better perceptual quality than VAE at corresponding decoder scales, especially when input images contain complex textures or structure. *Best viewed when zoomed-in and in color.*

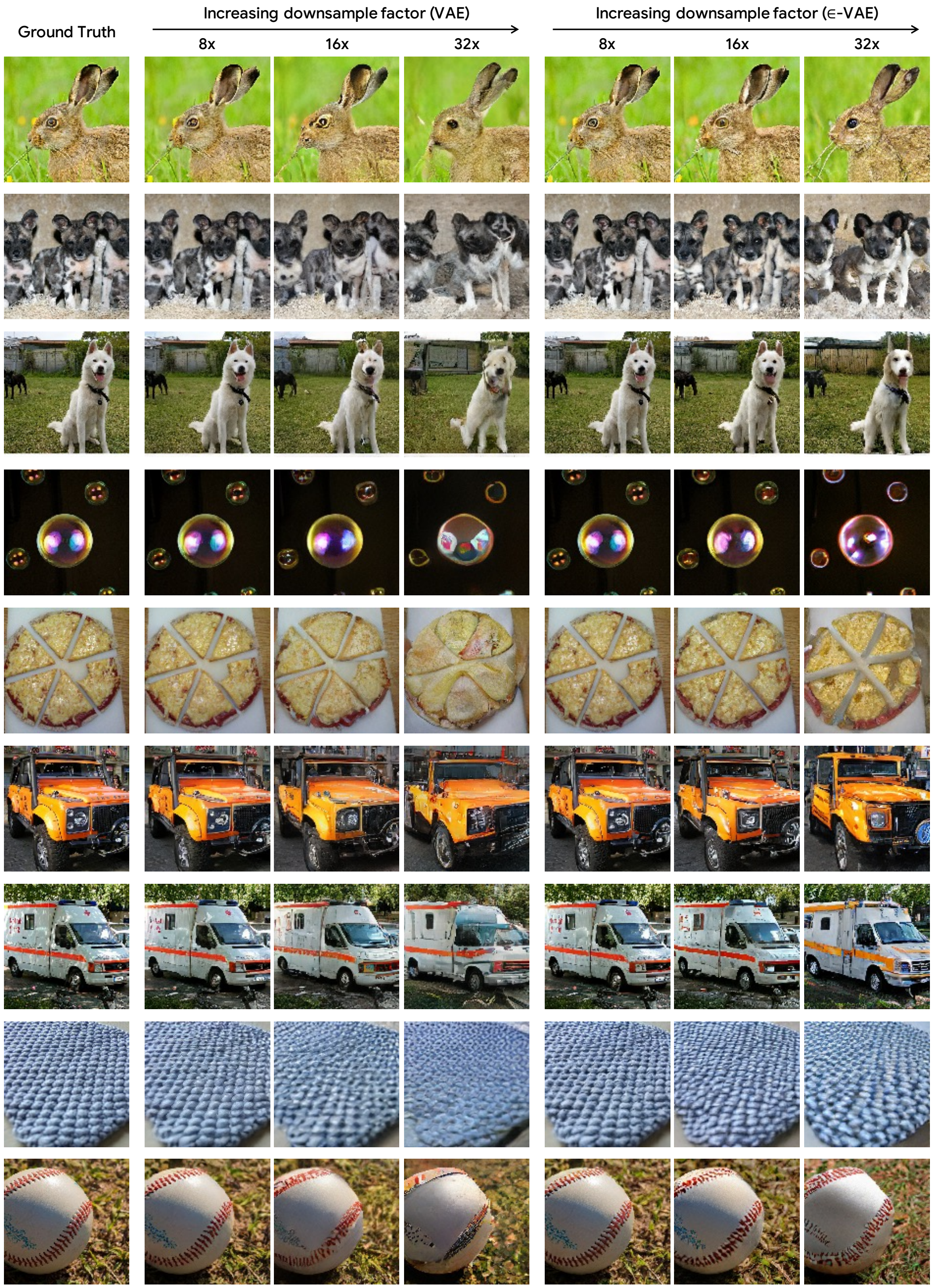

Figure 9: **Reconstruction results with varying downsampling ratios.** ϵ-VAE achieve higher fidelity and better perceptual quality than VAE, especially under extreme downsampling factors. *Best viewed when zoomed-in and in color.*

Figure 10: **ϵ-VAE reconstruction results with varying random seeds and downsampling ratios.** We can see greater diversity in the reconstruction results along with the increased downsampling factors. *Best viewed when zoomed-in and in color*.