BREAKING THE LIKELIHOOD–QUALITY TRADE-OFF IN DIFFUSION MODELS BY MERGING PRETRAINED EXPERTS

Yasin Esfandiari[1]* **Stefan Bauer**[2,3] **Sebastian U. Stich**[4] **Andrea Dittadi**[2,3,5]
[1]Saarland University [2]Helmholtz AI [3]Technical University of Munich
[4]CISPA Helmholtz Center for Information Security [5]MPI for Intelligent Systems, Tübingen

ABSTRACT

Diffusion models have recently emerged as powerful generative models capable of producing highly realistic images. Despite their success, a persistent challenge remains: models that generate high-quality samples often assign poor likelihoods to data, and vice versa. This trade-off arises because perceptual quality depends more on modeling high-noise regions, while likelihood is dominated by sensitivity to low-level image statistics. In this work, we propose a simple yet effective method to overcome this trade-off by merging two pretrained diffusion experts, one focused on perceptual quality and the other on likelihood, within a Mixture-of-Experts framework. Our approach applies the image-quality expert during high noise levels and uses the likelihood expert in low noise levels. Empirically, our merged model consistently improves over both experts: on CIFAR-10, it achieves better likelihood and sample quality than either baseline. On ImageNet32, it matches the likelihood of the likelihood expert while surpassing the image-quality expert in FID, effectively breaking the likelihood–quality trade-off in diffusion models.

1 INTRODUCTION

Diffusion models (DMs) are a class of probabilistic generative models that learn to approximate a data distribution by reversing a forward noising process through a learned denoising procedure (Sohl-Dickstein et al., 2015; Ho et al., 2020; Nichol & Dhariwal, 2021). They have recently achieved state-of-the-art results, e.g., in image generation (Dhariwal & Nichol, 2021; Tang et al., 2024; Kim et al., 2024), density estimation (Kingma et al., 2021), and in text-to-image and text-to-video generation tasks (Esser et al., 2024; Polyak et al., 2024).

For image data, likelihood-based metrics and perceptual image quality often exhibit a disconnect in practice (Theis et al., 2015); that is, strong performance on one does not necessarily imply good performance on the other. In particular, Kim et al. (2021) highlights an inverse correlation between likelihood (typically measured via Negative Log-Likelihood, or NLL) and sample quality (commonly measured via Frechet Inception Distance, or FID). As a result, models optimized for NLL, such as Kingma et al. (2021), employ likelihood-weighted training objectives, whereas models targeting low FID scores (Nichol & Dhariwal, 2021; Kingma & Gao, 2024) adopt alternative weighting schemes. Consequently, a trade-off emerges: models with excellent sample quality often perform poorly on likelihood, and vice versa. Since NLL and FID reflect complementary aspects of model performance, addressing both is essential for building robust diffusion models that capture the data distribution and produce high-quality samples.

In this paper, we aim to overcome the NLL–FID trade-off by designing a model that can generate images with both high perceptual quality and strong likelihood. To do this, we start from two key empirical observations reported in the literature: (1) Higher noise levels are associated with perceptual image quality. DDPM (Ho et al., 2020) used a simplified objective that down-weights the loss at lower noise levels, allowing the model to focus on more challenging denoising steps at higher

*Work done while at Helmholtz AI. Correspondence to yaes00001@stud.uni-saarland.de.

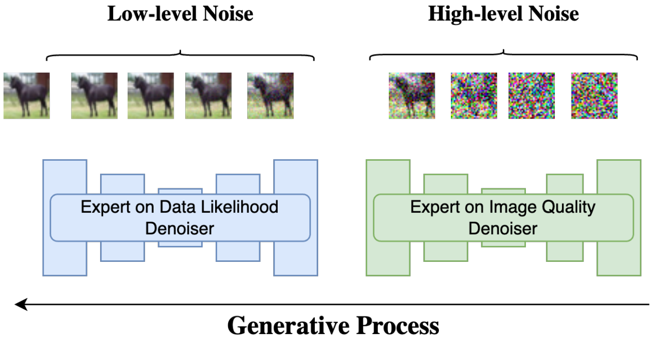
Low-level Noise
High-level Noise
Expert on Data Likelihood Denoiser
Expert on Image Quality Denoiser
Generative Process

Figure 1: Diagram of our proposed merged model where at time $\eta = 0.5$ we switch between denoisers. Note that the likelihood model is only used for almost imperceptible noise levels. This significantly improves the likelihood, which is sensitive to low-level color statistics, while leaving the FID unaffected.

noise levels. Similarly, Kim et al. (2021) showed that accurate score prediction at high noise levels is crucial for generating realistic samples, and that overly small truncation harms sample fidelity. (2) Likelihood is highly sensitive to low-level statistics, such as exact pixel values (Zheng et al., 2023b; Kim et al., 2021), while we are typically more interested in the overall structure of the image rather than exact pixel-level details. Supporting this, Kingma & Dhariwal (2018); Kingma & Gao (2024) showed that training on 5-bit images, which effectively discards fine details, can lead to better visual quality.

Motivated by these insights, our approach is simple: we merge two pretrained diffusion experts—one specialized in image quality, and the other in likelihood. For high noise levels, we use an expert on good image-quality model (EDM, Karras et al. 2022). For the low-noise steps, we switch to a likelihood expert trained for accurate density modeling (VDM, Kingma et al. 2021). An overview of the merged model is shown in Fig. 1. Starting from pure noise, we first denoise using the image-quality expert to obtain a clean, high-fidelity sample. Then, at a chosen intermediate step, we switch to the likelihood expert to refine the sample further, aiming to improve likelihood while maintaining perceptual quality. By carefully choosing the switching point, we achieve strong performance on both FID and NLL, effectively breaking the trade-off between likelihood and quality.

In Section 2, we provide the necessary background and discuss the underlying causes of the likelihood–quality trade-off. Section 3 reviews related work that has attempted to address this issue. In Section 4, we introduce our proposed method in detail, including modifications to the sampling procedure and likelihood evaluation. Section 5 presents our experimental setup and results. Finally, in Sections 6 and 7, we discuss the limitations of our approach and conclude the paper.

2 Background

2.1 Diffusion Models

Diffusion models (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020) are a type of generative models that learn to reverse a diffusion process that gradually adds noise to data. Following the variational diffusion models (VDM) framework (Kingma et al., 2021), let $\mathbf{x} \in \mathbb{R}^d$ denote a data point, and let $\{\mathbf{z}_{t(i)}\}_{i=0}^{i=T}$ be the latent variables in $\mathbb{R}^d$ over which the noising process is defined. This stochastic process is defined as a forward-time process from $t = 0$ to $t = 1$ such that the transition kernel $q(\mathbf{z}_{t(i)}|\mathbf{z}_{s(i)})$ is linear Gaussian, with $t(i) = \frac{i}{T}$ and $s(i) = \frac{i-1}{T}$.[1] The marginals of this process can be directly parameterized as $q\left(\mathbf{z}_t \mid \mathbf{x}\right) = \mathcal{N}\left(\mathbf{z}_t; \alpha_t \mathbf{x}, \sigma_t^2 \mathbf{I}\right)$, where $\alpha_t, \sigma_t \in \mathbb{R}_{>0}$ are smooth scalar-valued functions of t, named *noise schedule* parameters. We assume that the *signal-to-noise* ratio $\mathrm{SNR}(t) = \alpha_t^2/\sigma_t^2$ is strictly monotonically decreasing w.r.t. t, and we will consider the variance preserving (VP) process which entails $\alpha_t^2 + \sigma_t^2 = 1$ for all t.

[1] In the following, we will often omit the argument i to avoid clutter.

The forward process can be reversed when conditioning on the data, and the distribution $q(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x})$ is Gaussian and available in closed form. The *reverse process* (generative) is then defined as $p\left(\mathbf{z}_s \mid \mathbf{z}_t\right) = q\left(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x} = \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t, t)\right)$, i.e., as the ground-truth conditional reverse process where we replace the data $\mathbf{x}$—unavailable at inference time—with the output of a model that predicts $\mathbf{x}$ from its noisy version $\mathbf{z}_t = \alpha_t \mathbf{x} + \sigma_t \boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The model is learned by maximizing the Variational Lower Bound (VLB) of the marginal likelihood:

$$-\log p(\mathbf{x}) \leq -\mathrm{VLB}(\mathbf{x}) = \underbrace{D_{\mathrm{KL}}(q(\mathbf{z}_1 \mid \mathbf{x}) \| p(\mathbf{z}_1)}_{\text{Prior loss}} + \underbrace{\mathbb{E}_{q(\mathbf{z}_0 \mid \mathbf{x})}[-\log p(\mathbf{x} \mid \mathbf{z}_0)]}_{\text{Reconstruction loss}} + \underbrace{\mathcal{L}_T(\mathbf{x})}_{\text{Diffusion loss}}, \quad (1)$$

$$\mathcal{L}_T(\mathbf{x}) = \sum_{i=1}^{T} \mathbb{E}_{\mathbf{z}_{t(i)} \sim q(\mathbf{z}_{t(i)} \mid \mathbf{x})} D_{\mathrm{KL}}\left[q(\mathbf{z}_{s(i)} \mid \mathbf{z}_{t(i)}, \mathbf{x}) \| p_{\boldsymbol{\theta}}(\mathbf{z}_{s(i)} \mid \mathbf{z}_{t(i)})\right] \quad (2)$$

In the continuous-time limit ($T \rightarrow \infty$), and when rewriting $\mathcal{L}_T$ in terms of a noise-prediction model (as opposed to a data-prediction one), Kingma et al. (2021) showed that $\mathcal{L}_T$ simplifies as:

$$\mathcal{L}_\infty(\mathbf{x}) = \frac{1}{2} \mathbb{E}_{\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), t \sim \mathcal{U}(0,1)} \left[\frac{d\gamma_t}{dt} \cdot \left\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}\left(\mathbf{z}_t; t\right)\right\|_2^2\right], \quad \gamma_t = -\log(\mathrm{SNR}_t) \quad (3)$$

where $\gamma_t = -\log(\mathrm{SNR}_t)$ and $\mathbf{z}_t = \alpha_t \mathbf{x} + \sigma_t \boldsymbol{\epsilon}$. The continuous-time case can be equivalently defined directly in continuous time starting from a linear stochastic differential equation (SDE) (Song et al., 2020b; 2021):

$$\mathrm{d}\mathbf{z}_t = f(t)\mathbf{z}_t \,\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}_t, \quad \mathbf{z}_0 \sim q\left(\mathbf{z}_0 \mid \mathbf{x}\right), \quad (4)$$

where $\mathbf{w}_t \in \mathbb{R}^d$ is the standard Wiener process, and $f(t)$ and $g^2(t)$ are defined as:

$$f(t) = \frac{\mathrm{d}\log \alpha_t}{\mathrm{d}t}, \quad g^2(t) = \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log \alpha_t}{\mathrm{d}t}\sigma_t^2. \quad (5)$$

Sampling can be achieved through the *Backward Process* by solving the *Diffusion SDE* from time $t = 1$ to $t = 0$ in terms of *noise-prediction* model:

$$\mathrm{d}\mathbf{z}_t = \left[f(t)\mathbf{z}_t + \frac{g^2(t)}{\sigma_t}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}\left(\mathbf{z}_t, t\right)\right]\mathrm{d}t + g(t)\mathrm{d}\overline{\mathbf{w}}_t, \quad \mathbf{z}_1 \sim \mathcal{N}\left(\mathbf{0}, \tilde{\sigma}^2\mathbf{I}\right) \quad (6)$$

Song et al. (2020b) proved that for all diffusion processes, there exists a corresponding *deterministic process* whose trajectories share the same marginal probability densities $\{p_t\}_{t=0}^{t=1}$ named as *probability flow ODE*, which can be used for sampling similar to *Diffusion SDE*. When parameterized by a *noise-prediction* model, *Diffusion ODE* satisfies:

$$\frac{\mathrm{d}\mathbf{z}_t}{\mathrm{d}t} = \mathbf{h}_{\boldsymbol{\theta}}\left(\mathbf{z}_t, t\right) := f(t)\mathbf{z}_t + \frac{g^2(t)}{2\sigma_t}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}\left(\mathbf{z}_t, t\right), \quad \mathbf{z}_1 \sim \mathcal{N}\left(\mathbf{0}, \tilde{\sigma}^2\mathbf{I}\right) \quad (7)$$

The above formula allows us to compute the exact likelihood on any input data via the instantaneous change of variables formula as proposed in (Chen et al., 2018). Following Sahoo et al. (2023); Song et al. (2020b); Zheng et al. (2023b), the log-likelihood of $p_{\boldsymbol{\theta}}(\mathbf{z}_0)$ can be computed using Eq. (8), where we are integrating the divergence of the drift function:

$$\log p_{\boldsymbol{\theta}}\left(\mathbf{z}_0\right) = \log p_{\boldsymbol{\theta}}\left(\mathbf{z}_1\right) - \int_{t=0}^{t=1} \mathrm{tr}\left(\nabla_{\mathbf{z}_t}\mathbf{h}_{\boldsymbol{\theta}}\left(\mathbf{z}_t, t\right)\right)\mathrm{d}t \quad (8)$$

2.2 How do different weightings of loss affect the FID-NLL?

In this section, we briefly include a previous study that shows how the different weighting of the loss function can influence the likelihood and image quality. VDM++ (Kingma & Gao, 2024) proved how various diffusion model objectives in the literature can be understood as a special case of a *weighted loss* (Kingma et al., 2021) in Eq. (9), with different choices of weighting. Using the uniform weighting, $w(\gamma_t) = 1$, that corresponds to ELBO objective Eq. (3) and results in a good data-likelihood model, while setting the weighting term $w(\gamma_t) = \mathrm{d}t/\mathrm{d}\gamma_t$, produces good sample-quality outputs like L_{simple} in IDDPM (Nichol & Dhariwal, 2021).

$$\mathcal{L}_w(\mathbf{x}) = \frac{1}{2}\mathbb{E}_{t \sim \mathcal{U}(0,1), \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})}\left[w\left(\gamma_t\right) \cdot \frac{\mathrm{d}\gamma_t}{\mathrm{d}t} \cdot \left\|\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}\left(\mathbf{z}_t; \gamma_t\right) - \boldsymbol{\epsilon}\right\|_2^2\right], \quad \gamma_t = -\log(SNR_t) \quad (9)$$

3 Related Work

Likelihood experts. Several methods focus on improving likelihood. VDM (Kingma et al., 2021) and ScoreFlow (Song et al., 2021) directly optimize (a bound on) the data log-likelihood. i-DODE (Zheng et al., 2023b) introduces *velocity-prediction* and proposes an improved likelihood estimation technique. Other works (Sahoo et al., 2023; Nielsen et al., 2023; Bartosh et al., 2024) explore learnable forward processes, whereas our study focuses on standard diffusion models with fixed linear forward noise schedules.

Sample quality experts. Many studies improve the perceptual quality of generated samples by introducing better or more efficient samplers (Song et al., 2020a;b; Lu et al., 2022; Zheng et al., 2023a; Zhao et al., 2024; Karras et al., 2022; Zhou et al., 2024), addressing exposure bias (Ning et al., 2023), or applying alternative loss weighting strategies (Kingma & Gao, 2024; Ho et al., 2020). GMEM (Tang et al., 2024) enhances both quality and efficiency by incorporating an external memory bank into a transformer-based model, achieving state-of-the-art FID on CIFAR-10. PaGoDA (Kim et al., 2024), a distillation-based approach, achieves the best-known FID on ImageNet32. In this work, we focus on UNet-based diffusion models trained with simpler objectives such as *noise prediction*, and exclude distillation-based methods from our scope.

Experts on both metrics. Soft Truncation (Kim et al., 2021) proposes a training strategy that softens fixed truncation into a random variable, adjusting loss weighting across diffusion times to address the likelihood–quality trade-off. While aligned in motivation with our work, their approach requires training from scratch. In contrast, our method directly leverages existing pretrained models. CTM (Kim et al., 2023) uses a combination of loss terms, including an additional GAN loss, along with data augmentation to improve both metrics. In contrast, we address the trade-off from a different perspective, by merging experts trained with the standard denoising objective.

Mixture-of-Experts. Mixture-of-Experts (MoE) frameworks have been applied to diffusion models in contexts such as zero-shot text-to-image generation (Balaji et al., 2022; Feng et al., 2023) and controllable image synthesis (Bar-Tal et al., 2023). More recently, MDM (Kang et al., 2024) proposed a MoE strategy where each expert is trained on a specific time interval. While effective, their method employs identical architectures across experts and primarily targets training efficiency and sample quality. To the best of our knowledge, we are the first to address this trade-off by merging pretrained experts specialized separately in likelihood and sample quality.

4 Merging Experts

In this section, we present our method, which is based on the negative log Signal-to-Noise, defined as $\gamma_t = -\log(\mathrm{SNR}_t)$ (Kingma et al., 2021). As previously mentioned, the *Signal-to-Noise Ratio* decreases as we move from the data distribution toward pure noise. Since γ_t is a monotonically increasing function of time, values close to the data distribution correspond to smaller γ_t, while values near the noise correspond to larger γ_t.

We proposed to *merge* two pretrained diffusion models, each specialized in one of the two key generative modeling aspects: perceptual image quality or data likelihood. For the high γ_t region (corresponding to high noise levels), we use an expert model focused on image quality; for the low γ_t region (low noise), we use an expert model focused on likelihood (see Fig. 1). This design is supported by previous findings (Zheng et al., 2023b; Kim et al., 2021), which show that likelihood benefits from focusing on small time steps, while perceptual quality is improved by modeling large time steps effectively.

In our implementation, we use EDM (Karras et al., 2022) as the expert on perceptual quality in the high-noise region, and VDM (Kingma et al., 2021) as the expert on likelihood in the low-noise region. Let τ_1 and τ_2 represent the time step intervals corresponding to low and high noise, respectively. Given a threshold time step η over the full γ range of $\tau_1 \cup \tau_2$, our merged model $f_{\boldsymbol{\theta}(t)}(\mathbf{z}_t, \gamma_t)$, which serves as a denoising autoencoder, is defined as:

$$\boldsymbol{\theta}(t) = \begin{cases} \boldsymbol{\theta}_{\text{VDM}}, & t \leq \eta, \\ \boldsymbol{\theta}_{\text{EDM}}, & t > \eta. \end{cases} \tag{10}$$

4.1 Sampling

Given a sample from pure noise, we perform denoising in two stages. For the high-noise region (i.e., large γ_t), we apply the expert model trained for perceptual image quality (EDM). Once we reach the threshold time step $\eta \in (0, 1)$, we switch to the expert model trained for likelihood (VDM) and continue denoising in the low-noise region (i.e., small γ_t) until we reach $\gamma_{\min}$. The values of γ_t follow a fixed linear noise schedule, ranging from $\gamma_{\max}$ (corresponding to pure noise) to $\gamma_{\min}$ (corresponding to clean data), and η determines the time at which we switch from one expert to the other. Algorithms 1 – 2 detail the full sampling procedure. These are modified versions of samplers from Zheng et al. (2023b) and Kingma et al. (2021), respectively. The final step in both algorithms includes the reconstruction term (Kingma et al., 2021), which maps the latent sample back to the data space.

Algorithm 1 Sampling with PF-ODE

```
procedure ODE SAMPLER WITH AN ADAPTIVE STEP SIZE
    Input: Threshold $\eta \in (0, 1)$; Smallest time step $\gamma_{min}$; Largest time step $\gamma_{max}$;
    $\mathbf{z}_T \sim \mathcal{N}\left(\mathbf{0}, \sigma_T^2 \mathbf{I}\right)$
    Compute intermediate time step using the *fixed linear noise schedule*
        $\gamma_\eta = \gamma_{min} + \eta \cdot (\gamma_{max} - \gamma_{min})$
    $\mathbf{z}_\eta \leftarrow \text{PF-ODE}_{\text{EDM}}\left(\gamma_{max}, \gamma_\eta, \mathbf{z}_T\right)$
    $\mathbf{z}_0 \leftarrow \text{PF-ODE}_{\text{VDM}}\left(\gamma_\eta, \gamma_{min}, \mathbf{z}_\eta\right)$
    $\mathbf{x} \sim p(\mathbf{x} \mid \mathbf{z}_0)$
end procedure
```

Algorithm 2 Ancestral Sampling

```
procedure VDM ANCESTRAL
    Input: Threshold $\eta \in (0, 1)$; $T$;
        Initial $\mathbf{z}_T \sim \mathcal{N}\left(\mathbf{0}, \sigma_T^2 \mathbf{I}\right)$
    for $t = T \dots 1$ do
        if $t/T \leq \eta$ then
            $\mathbf{z}_{t-1} \leftarrow \boldsymbol{\theta}_{\text{VDM}}\left(\mathbf{z}_t, \gamma_t\right)$
        else if $t/T > \eta$ then
            $\mathbf{z}_{t-1} \leftarrow \boldsymbol{\theta}_{\text{EDM}}\left(\mathbf{z}_t, \gamma_t\right)$
        end if
    end for
    $\mathbf{x} \sim p(\mathbf{x} \mid \mathbf{z}_0)$
end procedure
```

4.2 Likelihood Evaluation

We consider two approaches for evaluating the likelihood of our model: (1) the variational lower bound (VLB) (Kingma et al., 2021), and (2) exact likelihood computation using the probability flow ODE (Zheng et al., 2023b; Song et al., 2020b; 2021; Sahoo et al., 2023).

Variational Lower Bound (VLB). The VLB consists of three terms (Kingma et al., 2021), as shown in Eq. (2). The prior and reconstruction losses are model-independent and computed from γ values, while the diffusion loss depends on which expert is used. For each time step t, we apply $\boldsymbol{\theta}_{\text{VDM}}$ if $t \leq \eta$, and $\boldsymbol{\theta}_{\text{EDM}}$ otherwise, following Eq. (3) and the switching rule in Eq. (10).

Exact Likelihood via Probability Flow ODE. As an alternative, we compute the exact likelihood using the probability flow ODE defined in Eq. (11). In our merged setup Eq. (10), this results in two sequential ODE integrations—one for each expert. Starting from an almost clean sample $\mathbf{z}_0$, we integrate from $\gamma_{\min}$ to γ_η using $\text{PF-ODE}_{\text{VDM}}$, and then from γ_η to $\gamma_{\max}$ using $\text{PF-ODE}_{\text{EDM}}$.

5 Experiments

In this section, we compare our model against state-of-the-art baselines in terms of both sample quality and data likelihood.

5.1 Experimental Setup

Datasets. We evaluate our model on the test sets of CIFAR-10 (Krizhevsky & Hinton, 2009) and ImageNet32 (Deng et al., 2009). As two versions of ImageNet32 exist in the literature, we use the older version (denoted with an asterisk * in comparisons) to remain consistent with prior work. For CIFAR-10, we reproduce results using available training details. For ImageNet32, due to time constraints, we did not tune hyperparameters extensively, resulting in sub-optimal base models. Our primary focus is on CIFAR-10, with ImageNet32 results included to test whether similar trends hold.

Table 1: Likelihood (ODE) in bits/dimension (BPD) on the test set of CIFAR-10 and ImageNet32

		Threshold											
		EDM	$\boldsymbol{\eta}_{min}$	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	$\boldsymbol{\eta}_{max}$	**VDM**
CIFAR10	**NLL**	3.21	3.09	2.83	2.69	2.63	**2.62**	2.62	2.63	2.63	2.63	2.64	2.64
	NFE	204	232	234	236	253	259	254	251	257	273	274	248
ImageNet32	**NLL**	4.04	3.96	3.80	3.76	3.74	3.72	3.72	**3.72**	3.72	3.72	3.72	3.72
	NFE	195	185	192	186	180	180	196	210	220	232	236	205

Table 2: Image Quality in FID@50k on CIFAR-10 and ImageNet32 datasets using VDM Ancestral and ODE samplers. We wrote them in abbreviation to save some space.

		Threshold											
		EDM	$\boldsymbol{\eta}_{min}$	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	$\boldsymbol{\eta}_{max}$	**VDM**
CIFAR10	**VDM**	**3.37**	3.45	3.46	3.42	3.39	3.53	4.51	7.07	9.26	9.87	9.91	9.32
	ODE	2.02	2.04	2.05	2.03	**2.01**	2.14	2.82	4.75	6.86	7.67	7.73	9.37
	NFE	125	145	147	159	169	173	193	221	239	226	238	206
ImageNet32	**VDM**	8.50	8.60	8.60	8.50	8.31	7.94	**7.65**	8.61	9.84	9.99	10.01	9.89
	ODE	7.38	7.43	7.44	7.39	7.26	6.98	**6.58**	6.72	7.15	7.15	7.11	9.85
	NFE	120	140	144	150	166	169	180	204	207	189	189	158

Baselines. We compare our *merged* model with its constituent components—VDM and EDM—as baselines, each evaluated over their full native noise ranges. We also include results from related methods listed in Table 3; additional comparisons can be found in Table 6 in the Appendix.

Metrics and evaluation setup. We evaluate sample quality using Fréchet Inception Distance (FID) (Heusel et al., 2017), comparing 50k generated samples with reference statistics for each dataset using the evaluation code from Karras et al. (2022). For data likelihood, we use bits per dimension (BPD). As exact likelihood computation generally yields better results than the variational lower bound (VLB), we report exact likelihoods in our main experiments. For VLB results, refer to Table 5 in Appendix C.

Our experiments use the Variance-Preserving (VP) setting, where $\sigma_t^2 = \text{sigmoid}(\gamma_t)$ and $\alpha_t^2 = 1 - \sigma_t^2$, operating directly on pixel space. We exclude distillation-based methods and latent-space models. For likelihood evaluation, we re-implemented the PyTorch version of Truncated-Normal dequantization and the ODE sampler from the i-DODE repository[2] (Zheng et al., 2023b), which follows the γ-based formulation defined in Eq. (11). See Appendix A.1.1 for our derivation.

The time step range for VDM is $\gamma_{\text{VDM}} \in [-13.3, 5]$, while for EDM it is $\gamma_{\text{EDM}} \in [-12.43, 8.764]$. We define the merged model range as $\gamma_{\text{Merged}} \in [-13.3, 8.764]$, and select threshold values η at {3.94%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 82.94%} of this range. The smallest threshold, $\eta = 3.94\%$, corresponds to $\gamma = -12.43$ ($\boldsymbol{\eta}_{\text{min}}$), and the largest, $\eta = 82.94\%$, corresponds to $\gamma = 5$ ($\boldsymbol{\eta}_{\text{max}}$). Note that $\eta = 0.0$ and $\eta = 1.0$ correspond to pure EDM and VDM baselines, respectively, without any expert switching. Further implementation details are provided in Appendix B.

5.2 Results

5.2.1 Effect of Varying the Time Threshold η

We evaluate the impact of different threshold values η in our merged model. Table 1 reports NLL (in BPD) using Truncated-Normal dequantization without importance weighting ($K = 1$). Table 2 shows corresponding FID scores and the number of function evaluations (NFE) for unconditional generation. For VDM Ancestral sampling, we use 256 steps. ImageNet32 results are provided in Appendix C.

Figure 2 summarizes performance across different thresholds η. As shown, a clear trade-off exists between likelihood and sample quality. On CIFAR-10, our method achieves the best balance at

[2] https://github.com/thu-ml/i-DODE

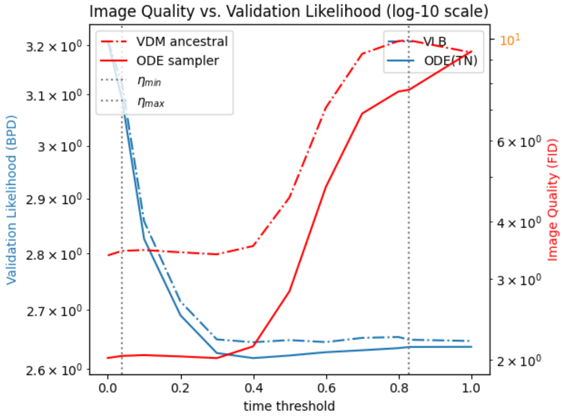

Figure 2: Our merged model's performance using different time step thresholds η on CIFAR-10. The EDM and VDM baselines are the $\eta \in \{0.0, 1.0\}$.

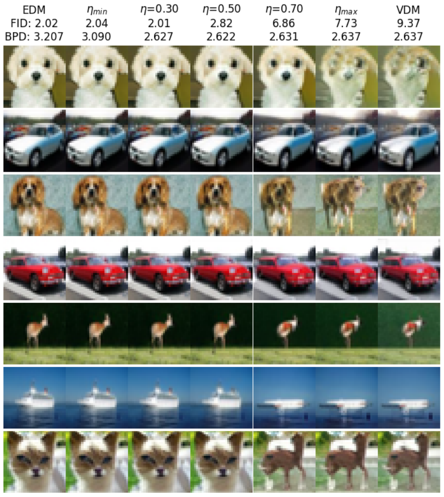

Figure 3: Qualitative results of our merged model performance using different time step thresholds η.

$\eta = 0.3$, while $\eta = 0.4$ yields even better likelihood than the VDM baseline, with only a minor FID drop (2.14 vs. 2.02). For full VLB values and a larger version of the plot, see Appendix C. On ImageNet32, $\eta = 0.5$ matches the VDM baseline in likelihood while surpassing EDM in FID. All baseline evaluations use their respective native γ ranges. **Importantly, we are able to identify a single threshold η that outperforms both base models across metrics, demonstrating that our method effectively breaks the likelihood–quality trade-off.**

Figure 3 shows qualitative results on CIFAR-10 using the ODE sampler. When using only the EDM model (left column), we observe high-quality samples but poor likelihood. As we begin to switch to the VDM model (e.g., η_{min} or $\eta = 0.3$), the likelihood improves while the sample quality remains nearly unchanged. **This exactly showcases the intuition behind our proposed method.** At higher thresholds, likelihood continues to improve, but visual quality starts to degrade. Remarkably, despite differences in architecture and training, both base models produce nearly identical outputs from the same noise input, both individually and within our merged model, highlighting a strong generalization effect (Kadkhodaie et al., 2023).

5.2.2 Comparing to other methods

Table 3 reports our results using Truncated-Normal dequantization and the adaptive-step ODE sampler, compared with existing methods from the literature. We focus on exact likelihood and ODE-based sampling, as they offer more consistent and favorable evaluations in our setting.

Among related approaches, we outperform Soft Truncation, which also aims to balance both likelihood and perceptual quality. i-DODE achieves strong likelihood by combining velocity parameterization with an error-bounded high-order Flow Matching objective. CTM improves both metrics using a mix of loss functions, including GAN-based losses, along with data augmentation. In contrast, our approach uses only standard denoising objectives, without any data augmentation for VDM and using default settings for EDM.

NFDM, MuLAN, and DiffEnc rely on learnable forward processes. Despite using a fixed linear schedule, our method achieves results competitive with DiffEnc and NFDM-OT. GMEM, PaGoDA, and SiD prioritize sample quality, often using distillation or transformer-based architectures. By contrast, we use a standard UNet architecture and focus on combining pretrained models.

Table 3: The comparison table for comparing our results with different models. By default, the evaluation of NLL is by Truncated Normal Dequantization, otherwise marked with other signs; Uniform Deq.†, Variational Deq.‡, VLB$^\vee$, Data Augmentation$^\uplus$., ImageNet32(old version)*.

Model	CIFAR10			ImageNet32		
	NLL(↓)	FID(↓)	NFE	NLL(↓)	FID(↓)	NFE
Main Baselines						
VDM (Kingma et al., 2021)	2.65$^\vee$	7.41	-	3.72$^{*\vee}$	-	-
EDM (w/ Heun Sampler) (Karras et al., 2022)	-	1.97	35	-	-	-
Focused on both FID-NLL						
Soft Truncation (Kim et al., 2021)	3.01†	3.96	-	3.90*†	8.42*	-
CTM ($\uplus$ - randomflip) (Kim et al., 2023)	2.43†	1.87	2	-	-	-
Focused on FID						
GMEM (Transformer-based) (Tang et al., 2024)	-	1.22	50	-	-	-
PaGoDA (distillation-based) (Kim et al., 2024)	-	-	-	-	0.79	1
SiD (distillation-based) (Zhou et al., 2024)	-	1.923	1	-	-	-
Focused on NLL						
i-DODE (VP) (Zheng et al., 2023b)	2.57	10.74	126	3.43/3.70*	9.09	152
i-DODE (VP, $\uplus$) (Zheng et al., 2023b)	2.42	3.76	215	-	-	-
Flow Matching (Lipman et al., 2022)	2.99†	6.35	142	3.53†	5.02	122
DiffEnc (Nielsen et al., 2023)	2.62$^\vee$	11.1	-	3.46$^\vee$	-	-
NFDM (Gaussian q, $\uplus$ - horizontalflip) (Bartosh et al., 2024)	2.49†	21.88	12	3.36	24.74	12
NFDM-OT($\uplus$ - horizontalflip) (Bartosh et al., 2024)	2.62†	5.20	12	3.45	4.11	12
MuLAN (w/o importance sampling k=1) (Sahoo et al., 2023)	2.59	-	-	3.71	-	-
Ours						
VDM (our evaluation, $\gamma \in$[-13.3, 5]) (Kingma et al., 2021)	2.64/2.66$^\vee$	9.37	206	3.72*/3.72$^{*\vee}$	9.85*	158
EDM (our evaluation, $\gamma \in$[-12.43, 8.764]) (Karras et al., 2022)	3.21	2.02	125	4.04*	7.38*	120
Ours NLL ($\eta = 0.4$, CIFAR10)	**2.62**	2.14	173	-	-	-
Ours ($\eta = 0.3$, CIFAR10)	<u>2.63</u>	**2.01**	169	-	-	-
Ours ($\eta = 0.5$, ImageNet32)	-	-	-	**3.72***	**6.58***	180

We do not claim state-of-the-art across all benchmarks, but we demonstrate that merging two pretrained diffusion models, one specialized in image quality and the other in likelihood, consistently improves both metrics over using either model individually.

6 Limitations and Future Work

Our method depends on the specific models being merged. We focus exclusively on pixel-space diffusion models with fixed linear noise schedules and standard denoising objectives, excluding distillation-based and latent-space methods. We also note that FID scores could likely improve further by integrating more advanced samplers such as Heun (Karras et al., 2022) or DPM-Solver-v3 (Zheng et al., 2023a). Additionally, selecting the optimal switching threshold η currently requires a search procedure. Exploring automated threshold selection, alternative architectures, and more advanced samplers are promising directions for future work.

7 Conclusion

We proposed a simple yet effective method to address the trade-off between sample quality and data likelihood in diffusion models. By merging two pretrained experts, one focused on image quality and the other on likelihood, we show that it is possible to improve both metrics compared to using each model individually.

On CIFAR-10, our merged model achieves better likelihood and sample quality than both baselines. On ImageNet32, it matches the likelihood of the likelihood expert while surpassing the image-quality expert in FID, effectively breaking the trade-off between the two objectives.

Our approach requires no retraining, works with existing pretrained models, and can be easily extended. While selecting the switching threshold currently requires a search, future work may explore automated selection, improved samplers, and alternative architectures.

Acknowledgments

The computations are done on the HPC Cluster at Helmholtz Munich and CISPA GPU Cluster at Saarbrücken. We will release the code in a Github repository.[3]

References

Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022.

Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Qinsheng Zhang, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, et al. ediff-i: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022.

Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. Multidiffusion: Fusing diffusion paths for controlled image generation. *arXiv preprint arXiv:2302.08113*, 2023.

Grigory Bartosh, Dmitry Vetrov, and Christian A Naesseth. Neural diffusion models. *arXiv preprint arXiv:2310.08337*, 2023.

Grigory Bartosh, Dmitry Vetrov, and Christian A Naesseth. Neural flow diffusion models: Learnable forward process for improved diffusion modelling. *arXiv preprint arXiv:2404.12940*, 2024.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

J.R. Dormand and P.J. Prince. A family of embedded runge-kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1):19–26, 1980. ISSN 0377-0427. doi: https://doi.org/10.1016/0771-050X(80)90013-3. URL `https://www.sciencedirect.com/science/article/pii/0771050X80900133`.

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 *Forty-first International Conference on Machine Learning*, 2024.

Zhida Feng, Zhenyu Zhang, Xintong Yu, Yewei Fang, Lanxin Li, Xuyi Chen, Yuxiang Lu, Jiaxiang Liu, Weichong Yin, Shikun Feng, et al. Ernie-vilg 2.0: Improving text-to-image diffusion model with knowledge-enhanced mixture-of-denoising-experts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10135–10145, 2023.

Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

Matej Grcić, Ivan Grubišić, and Siniša Šegvić. Densely connected normalizing flows. *Advances in Neural Information Processing Systems*, 34:23968–23982, 2021.

Louay Hazami, Rayhane Mama, and Ragavan Thurairatnam. Efficientvdvae: Less is more. *arXiv preprint arXiv:2203.13751*, 2022.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[3] `https://github.com/yasin-esfandiari/Breaking-Likelihood-Quality-Tradeoff-in-Diffusion-Models`

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Emiel Hoogeboom, Alexey A Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. *arXiv preprint arXiv:2110.02037*, 2021.

Michael F. Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.

Zahra Kadkhodaie, Florentin Guth, Eero P Simoncelli, and Stéphane Mallat. Generalization in diffusion models arises from geometry-adaptive harmonic representation. *arXiv preprint arXiv:2310.02557*, 2023.

Seoungyoon Kang, Yunji Jung, and Hyunjung Shim. Local expert diffusion models for efficient training in denoising diffusion probabilistic models. In *2nd Workshop on Sustainable AI*, 2024. URL https://openreview.net/forum?id=xWO2kx0x9O.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

Dongjun Kim, Seungjae Shin, Kyungwoo Song, Wanmo Kang, and Il-Chul Moon. Soft truncation: A universal training technique of score-based diffusion model for high precision score estimation. *arXiv preprint arXiv:2106.05527*, 2021.

Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. *arXiv preprint arXiv:2310.02279*, 2023.

Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Yuhta Takida, Naoki Murata, Toshimitsu Uesaka, Yuki Mitsufuji, and Stefano Ermon. Pagoda: Progressive growing of a one-step generator from a low-resolution diffusion teacher. *arXiv preprint arXiv:2405.14822*, 2024.

Diederik Kingma and Ruiqi Gao. Understanding diffusion objectives as the elbo with simple data augmentation. *Advances in Neural Information Processing Systems*, 36, 2024.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

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

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, Toronto, ON, Canada, 2009.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022.

Aaron Lou and Stefano Ermon. Reflected diffusion models. In *International Conference on Machine Learning*, pp. 22675–22701. PMLR, 2023.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pp. 8162–8171. PMLR, 2021.

Beatrix MG Nielsen, Anders Christensen, Andrea Dittadi, and Ole Winther. Diffenc: Variational diffusion with a learned encoder. *arXiv preprint arXiv:2310.19789*, 2023.

Mang Ning, Mingxiao Li, Jianlin Su, Albert Ali Salah, and Itir Onal Ertugrul. Elucidating the exposure bias in diffusion models. *arXiv preprint arXiv:2308.15321*, 2023.

Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, et al. Movie gen: A cast of media foundation models. *arXiv preprint arXiv:2410.13720*, 2024.

Subham Sekhar Sahoo, Aaron Gokaslan, Chris De Sa, and Volodymyr Kuleshov. Diffusion models with learned adaptive noise. *arXiv preprint arXiv:2312.13236*, 2023.

John Skilling. The eigenvalues of mega-dimensional matrices. *Maximum Entropy and Bayesian Methods: Cambridge, England, 1988*, pp. 455–466, 1989.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020a.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b.

Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. *Advances in neural information processing systems*, 34:1415–1428, 2021.

Yi Tang, Peng Sun, Zhenglin Cheng, and Tao Lin. Generative modeling with explicit memory. *arXiv preprint arXiv:2412.08781*, 2024.

Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*, 2015.

Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. *Advances in neural information processing systems*, 34:11287–11302, 2021.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024.

Kaiwen Zheng, Cheng Lu, Jianfei Chen, and Jun Zhu. Dpm-solver-v3: Improved diffusion ode solver with empirical model statistics. *Advances in Neural Information Processing Systems*, 36: 55502–55542, 2023a.

Kaiwen Zheng, Cheng Lu, Jianfei Chen, and Jun Zhu. Improved techniques for maximum likelihood estimation for diffusion odes. In *International Conference on Machine Learning*, pp. 42363–42389. PMLR, 2023b.

Mingyuan Zhou, Huangjie Zheng, Zhendong Wang, Mingzhang Yin, and Hai Huang. Score identity distillation: Exponentially fast distillation of pretrained diffusion models for one-step generation. In *International Conference on Machine Learning*, 2024.

A Appendix

A.1 Proofs and Derivations

A.1.1 Probability-Flow ODE

Here, we put the derivation of the PF-ODE in terms of $\gamma_t = -\log(\mathrm{SNR}_t)$:

$$\frac{\mathrm{d}\mathbf{z}_t}{\mathrm{d}\gamma_t} = -\frac{1}{2} \cdot \text{Sigmoid}(\gamma_t) \cdot \mathbf{z}_t + \frac{1}{2} \cdot \sigma_t \cdot \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t) \tag{11}$$

Let us simplify $g^2(t)$ in terms of $\lambda_t := \log\left(\alpha_t/\sigma_t\right) = -\frac{1}{2}\gamma_t$ from Eq. (5):

$$f(t) = \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}, \quad g^2(t) = \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\sigma_t^2$$

$$g^2(t) = \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\sigma_t^2 = 2\sigma_t^2\left(\frac{\mathrm{d}\log\sigma_t}{\mathrm{d}t} - \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\right) = -2\sigma_t^2\frac{\mathrm{d}\lambda_t}{\mathrm{d}t} \tag{12}$$

Please note that the above λ_t is half-log(SNR) (Lu et al., 2022), and is half of $\lambda_{\text{VDM++}}$ (Kingma & Gao, 2024). Moreover, our model input is based on γ_t, and since there is a bijection between t and γ_t, we can use $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t)$ instead of $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, t)$. Here are the steps to go from PF-ODE Eq. (7) in terms of time step variable t to time step variable γ:

$$\frac{\mathrm{d}\mathbf{z}_t}{\mathrm{d}t} = h_{\boldsymbol{\theta}}\left(\mathbf{z}_t, \gamma_t\right) := f(t)\mathbf{z}_t + \frac{g^2(t)}{2\sigma_t}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}\left(\mathbf{z}_t, \gamma_t\right), \quad \mathbf{z}_T \sim \mathcal{N}\left(\mathbf{0}, \tilde{\sigma}^2\mathbf{I}\right)$$

Substituting equations $f(t)$ and $g^2(t)$ the above equation yields:

$$\begin{aligned}\frac{\mathrm{d}\mathbf{z}_t}{\mathrm{d}t} &= f(t)\mathbf{z}_t + \frac{g^2(t)}{2\sigma_t}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t) \\ &= \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\mathbf{z}_t - \sigma_t\frac{\mathrm{d}\lambda_t}{\mathrm{d}t}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t)\end{aligned}$$

Using the *Chain Rule* $\frac{\mathrm{d}\mathbf{z}}{\mathrm{d}\gamma} = \frac{\mathrm{d}\mathbf{z}}{\mathrm{d}t} \cdot \frac{\mathrm{d}t}{\mathrm{d}\gamma}$ and VP-formula $\alpha_t = \text{Sigmoid}(-\gamma_t)^{1/2}$, we can re-write the equation above as:

$$\begin{aligned}\frac{\mathrm{d}\mathbf{z}}{\mathrm{d}\gamma} &= \left[f(t)\mathbf{z}_t + \frac{g^2(t)}{2\sigma_t}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t)\right] \cdot \frac{\mathrm{d}t}{\mathrm{d}\gamma} \\ &= \left[\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\mathbf{z}_t - \sigma_t\frac{\mathrm{d}\lambda_t}{\mathrm{d}t}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t)\right] \cdot \frac{\mathrm{d}t}{\mathrm{d}\gamma} \\ &= \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}\gamma}\mathbf{z}_t - \sigma_t\frac{\mathrm{d}\lambda_t}{\mathrm{d}\gamma}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t) \\ &= \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}\gamma}\mathbf{z}_t + \frac{1}{2}\sigma_t \cdot \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t) \\ &= \frac{\mathrm{d}}{\mathrm{d}\gamma} \cdot \log(\text{Sigmoid}(-\gamma_t))^{1/2} \cdot \mathbf{z}_t + \frac{1}{2}\sigma_t \cdot \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t) \\ &= \frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}\gamma} \cdot \log(\text{Sigmoid}(-\gamma_t)) \cdot \mathbf{z}_t + \frac{1}{2}\sigma_t \cdot \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t) \\ &= \frac{1}{2} \cdot \frac{1}{\text{Sigmoid}(-\gamma_t)}\frac{\mathrm{d}}{\mathrm{d}\gamma} \cdot \text{Sigmoid}(-\gamma_t) \cdot \mathbf{z}_t + \frac{1}{2}\sigma_t \cdot \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t) \\ &= \frac{1}{2} \cdot \frac{-1}{\text{Sigmoid}(-\gamma_t)} \cdot \text{Sigmoid}(-\gamma_t) \cdot (1 - \text{Sigmoid}(-\gamma_t)) \cdot \mathbf{z}_t + \frac{1}{2}\sigma_t \cdot \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t) \\ &= -\frac{1}{2} \cdot (1 - \text{Sigmoid}(-\gamma_t)) \cdot \mathbf{z}_t + \frac{1}{2}\sigma_t \cdot \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t) \\ &= -\frac{1}{2} \cdot \text{Sigmoid}(\gamma_t) \cdot \mathbf{z}_t + \frac{1}{2}\sigma_t \cdot \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t) = h_{\boldsymbol{\theta}}\left(\mathbf{z}_t, \gamma_t\right)\end{aligned}$$

The integration bounds in Eq. (8) would be $[\gamma_0, \gamma_1]$ with the above drift function.

$$\log p_{\boldsymbol{\theta}}\left(\mathbf{z}_0\right) = \log p_{\boldsymbol{\theta}}\left(\mathbf{z}_1\right) - \int_{\gamma_0}^{\gamma_1} \operatorname{tr}\left(\nabla_{\mathbf{z}_t}\mathbf{h}_{\boldsymbol{\theta}}\left(\mathbf{z}_t, \gamma_t\right)\right)\mathrm{d}\gamma \tag{13}$$

A.1.2 DEQUANTIZATION

Real-world image datasets typically consist of discrete data X with 8-bit integer values from 0 to 255. These are commonly normalized to the continuous range $[-1, 1]$, denoted by $\mathbf{x}$ (Zheng et al., 2023b; Sahoo et al., 2023). Dequantization methods assume that we train a continuous model distribution $p_{\boldsymbol{\theta}}$ over $\mathbf{x}$, and define the discrete model distribution as:

$$P_{\boldsymbol{\theta}}\left(\mathbf{x}\right) = \int_{\mathbf{u}\in\left[-\frac{1}{256},\frac{1}{256}\right]^d} p_{\boldsymbol{\epsilon}}(\mathbf{x}+\mathbf{u})\mathrm{d}\mathbf{u} \tag{14}$$

where $p_{\boldsymbol{\epsilon}}$ is Diffusion ODE defined at $\boldsymbol{\epsilon}$. To train $P_{\boldsymbol{\theta}}(\mathbf{x})$ by maximum likelihood estimation, variational dequantization (Ho et al., 2020; Zheng et al., 2023b) introduces a dequantization distribution $q(\mathbf{u}|\mathbf{x})$ and jointly trains $p_{\boldsymbol{\epsilon}}$ and $q(\mathbf{u}|\mathbf{x})$ by maximizing the variational lower bound:

$$\log P_{\boldsymbol{\theta}}\left(\mathbf{x}\right) \geq \mathbb{E}_{q(\mathbf{u}|\mathbf{x})}\left[\log p_{\boldsymbol{\epsilon}}\left(\mathbf{x}+\mathbf{u}\right) - \log q\left(\mathbf{u} \mid \mathbf{x}\right)\right] \tag{15}$$

The term $\log p_{\boldsymbol{\epsilon}}(\mathbf{x}+\mathbf{u})$ can be evaluated using the instantaneous change-of-variables formula (Chen et al., 2018), as shown in Eq. (13).

Zheng et al. (2023b) propose *Truncated-Normal Dequantization* for better likelihood estimation by testing $p_{\boldsymbol{\epsilon}}$ on $\hat{\mathbf{x}}_{\boldsymbol{\epsilon}} = \alpha_{\boldsymbol{\epsilon}}\mathbf{x} + \sigma_{\boldsymbol{\epsilon}}\hat{\boldsymbol{\epsilon}}$, where $\hat{\boldsymbol{\epsilon}}$ follows the Truncated-normal distribution (a normal distribution with mean $\mathbf{0}$, covariance $\mathbf{I}$, and bounds $[-\frac{1}{256}, \frac{1}{256}]$ along each dimension):

$$\hat{\boldsymbol{\epsilon}} \sim \mathcal{TN}\left(\mathbf{0}, \mathbf{I}, -\frac{1}{256}, \frac{1}{256}\right) \tag{16}$$

In this setting, Eq. (15) simplifies to the expression below (see Zheng et al. (2023b), Appendix), where $u := \frac{\sigma_{\boldsymbol{\epsilon}}}{\alpha_{\boldsymbol{\epsilon}}}\hat{\boldsymbol{\epsilon}} \in \left[-\frac{1}{256}, \frac{1}{256}\right]$, and $Z := \mathrm{erf}(\tau/\sqrt{2})$:

$$\begin{aligned} \log P_{\boldsymbol{\theta}}(\mathbf{x}) \geq\ & \mathbb{E}_{\hat{\boldsymbol{\epsilon}}\sim\mathcal{TN}(\mathbf{0},\mathbf{I},-\tau,\tau)}\left[\log p_{\boldsymbol{\epsilon}}\left(\hat{\mathbf{x}}_{\boldsymbol{\epsilon}}\right)\right] \\ & + \frac{d}{2}\left(1 + \log\left(2\pi\sigma_{\boldsymbol{\epsilon}}^2\right)\right) + d\log Z - d\frac{\tau}{\sqrt{2\pi}Z}\exp\left(-\frac{1}{2}\tau^2\right) \end{aligned} \tag{17}$$

Using $\gamma_{\boldsymbol{\epsilon}} = 13.3$ leads to $\tau \approx 3$, so the truncated normal distribution $\mathcal{TN}(\mathbf{0}, \mathbf{I}, -\tau, \tau)$ becomes nearly identical to the standard normal $\mathcal{N}(\mathbf{0}, \mathbf{I})$ due to the 3-σ principle, resulting in a negligible train-test gap. Similarly, the term $\log p_{\boldsymbol{\epsilon}}(\hat{\mathbf{x}}_{\boldsymbol{\epsilon}})$ is equivalent to $\log p_{\boldsymbol{\epsilon}}(\mathbf{z}_0)$, and can be evaluated using Eq. (13).

A.1.3 LIKELIHOOD COMPUTATION

Computing the trace of the Jacobian of the drift function, $\mathrm{tr}(\nabla_{\mathbf{z}_t}\mathbf{h}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t))$, as required in Eq. (13), is computationally expensive. In practice, it is commonly estimated using the Skilling–Hutchinson trace estimator (Skilling, 1989; Hutchinson, 1989). Following prior works (Zheng et al., 2023b; Chen et al., 2018; Sahoo et al., 2023), we approximate this quantity as:

$$\mathrm{tr}\left(\nabla_{\mathbf{z}_t}\mathbf{h}_{\boldsymbol{\theta}}\left(\mathbf{z}_t, t\right)\right) = \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[\boldsymbol{\epsilon}^\top\nabla_{\mathbf{z}_t}\mathbf{h}_{\boldsymbol{\theta}}\left(\mathbf{z}_t, t\right)\boldsymbol{\epsilon}\right]$$

where the random variable $\boldsymbol{\epsilon}$ satisfying $\mathbb{E}_{p(\boldsymbol{\epsilon})}[\boldsymbol{\epsilon}] = \mathbf{0}$ and $\mathrm{Cov}_{p(\boldsymbol{\epsilon})}[\boldsymbol{\epsilon}] = \mathbf{I}$. Common choices for $p(\boldsymbol{\epsilon})$ include the Rademacher or Gaussian distribution. Importantly, the term $\mathrm{tr}(\nabla_{\mathbf{z}_t}\mathbf{h}_{\boldsymbol{\theta}}(\mathbf{z}_t, t))\boldsymbol{\epsilon}$ can be efficiently computed using Jacobian-vector products supported by deep learning frameworks.

In our implementation, we use the Rademacher distribution for $p(\boldsymbol{\epsilon})$ and adopt the same solver settings as prior work (Sahoo et al., 2023; Song et al., 2020b; Zheng et al., 2023b). Specifically, we use RK45 ODE solver (Dormand & Prince, 1980) with `atol=1e-5` and `rtol=1e-5` to compute the integral in Eq. (13) using `scipy.integrate.solve_ivp`.

B IMPLEMENTATION DETAILS

B.1 TRAINING BASE MODELS

CIFAR-10. We used the publicly available EDM checkpoint for CIFAR-10 (Krizhevsky & Hinton, 2009)[4]. For VDM, we trained the PyTorch re-implementation[5] based on the architecture described

[4] `https://nvlabs-fi-cdn.nvidia.com/edm/pretrained/edm-cifar10-32x32-uncond-vp.pkl`

[5] `https://github.com/addtt/variational-diffusion-models`

in Kingma et al. (2021). The model was trained for 10 million steps on 8×A100 GPUs (40GB), with no data augmentation, a fixed linear γ schedule, and a batch size of 128. Our trained VDM model achieved 2.64 BPD on the test set (exact likelihood) and 2.66 BPD under VLB evaluation. For comparison, the original paper reported 2.65 BPD (VLB).

ImageNet32. As multiple versions of ImageNet32 (Deng et al., 2009) exist in the literature, we followed the i-DODE setup (Zheng et al., 2023b), converting their TensorFlow records to PNG format with separate `train` and `val` folders. The VDM model was trained similarly to CIFAR-10, but with 256 channels and a total batch size of 512, on 8×A100 GPUs (80GB), following Kingma et al. (2021). Training was performed for 2 million steps.

Since no pretrained EDM model exists for ImageNet32, we trained one ourselves using the official EDM repository, with parameters `--cond 0 --arch ddpmpp --duration 1000`. The model was trained for 1000M images on 4×A100 GPUs (40GB) with a total batch size of 1024. No hyperparameter tuning was performed, so the resulting model is considered sub-optimal.

B.2 Evaluation Settings

For all evaluations, we used the Exponential Moving Average (EMA) version of each model. FID scores were computed using the procedure from Karras et al. (2022), with reference statistics calculated from the training sets of CIFAR-10 and ImageNet32 and stored in `.npz` format.

B.3 Model Integration and Compatibility

Noise formulation in VDM and EDM. VDM (Kingma et al., 2021) is a *noise-prediction* model where the latent variable is defined as $\mathbf{z}_t = \alpha_t \cdot \mathbf{x} + \sigma_t \cdot \boldsymbol{\epsilon}$, while EDM (Karras et al., 2022) uses an *image-denoising* formulation $\mathbf{z}_t = \mathbf{x} + \sigma_{\text{edm}} \cdot \boldsymbol{\epsilon}$, with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ in both cases.

Rescaling between models. To enable compatibility between the two models in our Merged Model, we must rescale $\mathbf{z}_t$ to match the input format expected by each model. Specifically, to map from VDM to EDM format:

$$\begin{aligned} \mathbf{z}_t &= \alpha_t \cdot \mathbf{x} + \sigma_t \cdot \boldsymbol{\epsilon} && \text{(Divide by } \alpha_t\text{)} \\ \Rightarrow \frac{\mathbf{z}_t}{\alpha_t} &= \mathbf{x} + \frac{\sigma_t}{\alpha_t} \cdot \boldsymbol{\epsilon} && \text{(Input format to EDM,} \quad \sigma_{edm} = \frac{\sigma_t}{\alpha_t}) \end{aligned} \tag{18}$$

Sampling implementation. Given this mapping, the following PyTorch code implements the conditional sampling logic $p(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x})$ for VDM ancestral sampling, incorporating model switching at threshold η (the number in the comment indicates the equation number in (Kingma et al., 2021)):

```python
def sample_p_s_t(model, z, t, s, threshold_eta):
    gamma_t = model.gamma(t)
    gamma_s = model.gamma(s)
    c = -expm1(gamma_s - gamma_t)                              # eq 34
    alpha_t = torch.sqrt(torch.sigmoid(-gamma_t))              # eq 4
    alpha_s = torch.sqrt(torch.sigmoid(-gamma_s))
    sigma_t = torch.sqrt(torch.sigmoid(gamma_t))               # eq 3
    sigma_s = torch.sqrt(torch.sigmoid(gamma_s))

    # use VDM model
    if t <= threshold_eta:
        pred_noise = model.model1(z, gamma_t)
    # use EDM model
    else:
        class_labels = None                                # unconditional
        pred_img = model.model2(z/alpha_t, sigma_t/alpha_t, class_labels)
        pred_noise = (z - alpha_t * pred_img) / sigma_t

    mean = alpha_s / alpha_t * (z - c * sigma_t * pred_noise)    # eq 32

    scale = sigma_s * torch.sqrt(c)                            # eq 33
    return mean + scale * torch.randn_like(z)                  # eq 34
```

EDM noise range in γ space. When using ODE solvers based on $\gamma_t = -\log \mathrm{SNR}_t = -\log\left(\frac{\alpha_t^2}{\sigma_t^2}\right)$, we must identify the γ ranges corresponding to EDM time steps. According to Karras et al. (2022), EDM operates over the time interval $t \in [0.002, 80.0]$, which corresponds to the γ range shown in Table 4:

	α_{edm}	σ_{edm}	γ
t_{min}	1	0.002	-12.43
t_{max}	1	80	8.764

Table 4: Defined γ_t values for EDM model

Possible switching thresholds. In our experiments, we used the combined range $\gamma_{\text{Merged}} \in [-13.3, 8.764]$. The normalized formulation of the threshold is:

$$\gamma = 22.064 \cdot \eta - 13.3 \quad \text{or} \quad \eta = \frac{\gamma + 13.3}{22.064}.$$

Substituting the limits $\gamma = -12.43$ and $\gamma = 5$ yields $\eta_{\min} = 0.0394307$ and $\eta_{\max} = 0.829405$.

C Additional results and Visualizations

C.1 Extra evaluations of our method

VLB evaluation. Table 5 reports likelihood evaluations of our *Merged Model* using the Variational Lower Bound (VLB). We ran each experiment 10 times with a batch size of 512 and report the mean and standard deviation.

Table 5: VLB evaluation in terms of bpd on CIFAR-10 and ImageNet32 datasets

Threshold	**CIFAR10**		**ImageNet32**	
	Mean(↓)	Std	Mean(↓)	Std
0.0	3.21	0.06	4.01	0.004
tmin	3.12	0.005	3.98	0.004
0.1	2.86	0.009	3.82	0.004
0.2	2.71	0.008	3.78	0.004
0.3	2.65	0.007	3.75	0.005
0.4	2.64	0.008	3.73	0.005
0.5	2.65	0.009	3.73	0.004
0.6	2.65	0.007	3.73	0.005
0.7	2.65	0.008	3.73	0.007
0.8	2.65	0.009	3.73	0.005
tmax	2.65	0.008	3.73	0.002
1.0	2.65	0.005	3.73	0.004

Comparison across all metrics. Fig. 4 – Fig. 5 show our model's performance across all metrics on CIFAR-10 and ImageNet32 using both samplers (ODE and ancestral) along with VLB and ODE-based likelihood (with Truncated-Normal dequantization). Metrics are shown on a $\log_{10}$ scale. The EDM and VDM baselines correspond to $\eta = 0.0$ and $\eta = 1.0$, respectively, and do not involve expert switching.

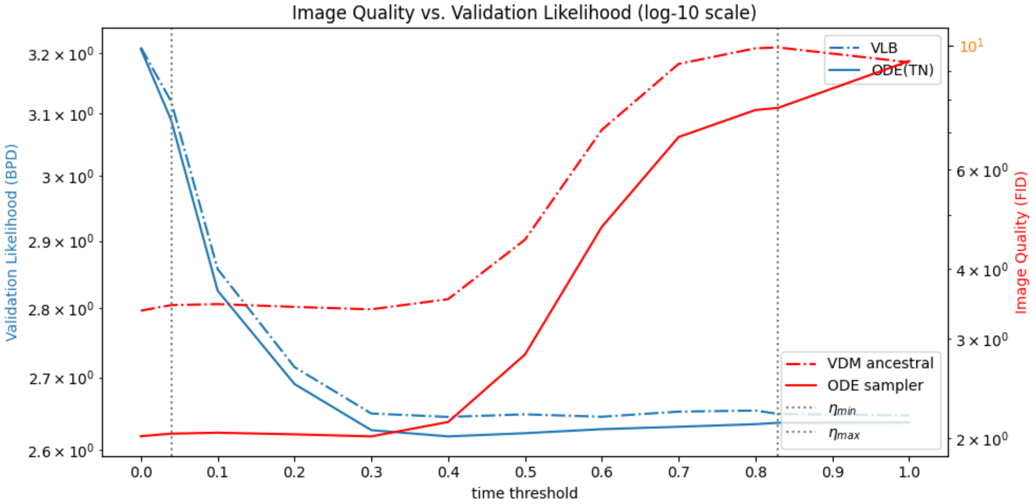

Figure 4: Performance on CIFAR-10 across all metrics. The EDM and VDM baselines are the $\eta = 0.0$ and $\eta = 1.0$, respectively, and they do not mean a switching threshold.

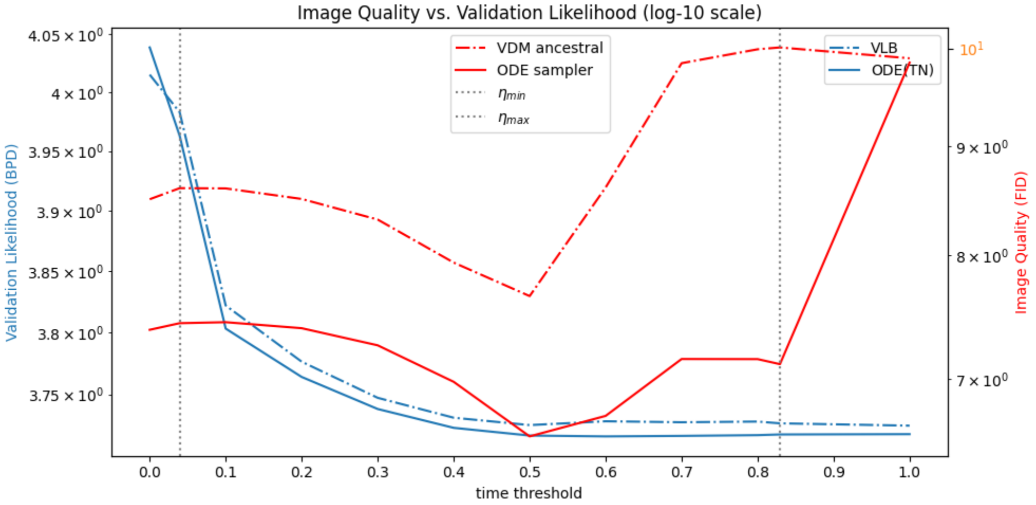

Figure 5: Performance on ImageNet32 across all metrics. The EDM and VDM baselines are the $\eta = 0.0$ and $\eta = 1.0$, respectively, and they do not mean a switching threshold

C.2 Datasets and Visualizations

This section presents qualitative visualizations of generated samples from our *Merged Model* on the ImageNet32 dataset. Using a fixed random seed, we generate samples across different switching thresholds η, as shown in Fig. 6.

In the leftmost column of the figure (corresponding to EDM with $\eta = 0.0$), the samples exhibit strong perceptual quality but poor likelihood. As η increases from its minimum possible value $\eta_{\min}$, likelihood improves while image quality remains largely unchanged. At $\eta = 0.5$, the model achieves the same likelihood as the VDM expert while surpassing the EDM baseline in FID. This is visually reflected in several rows; for instance, in the fourth row (flower category), the object becomes progressively less sharp as η increases, eventually appearing blurred when only the VDM expert is used.

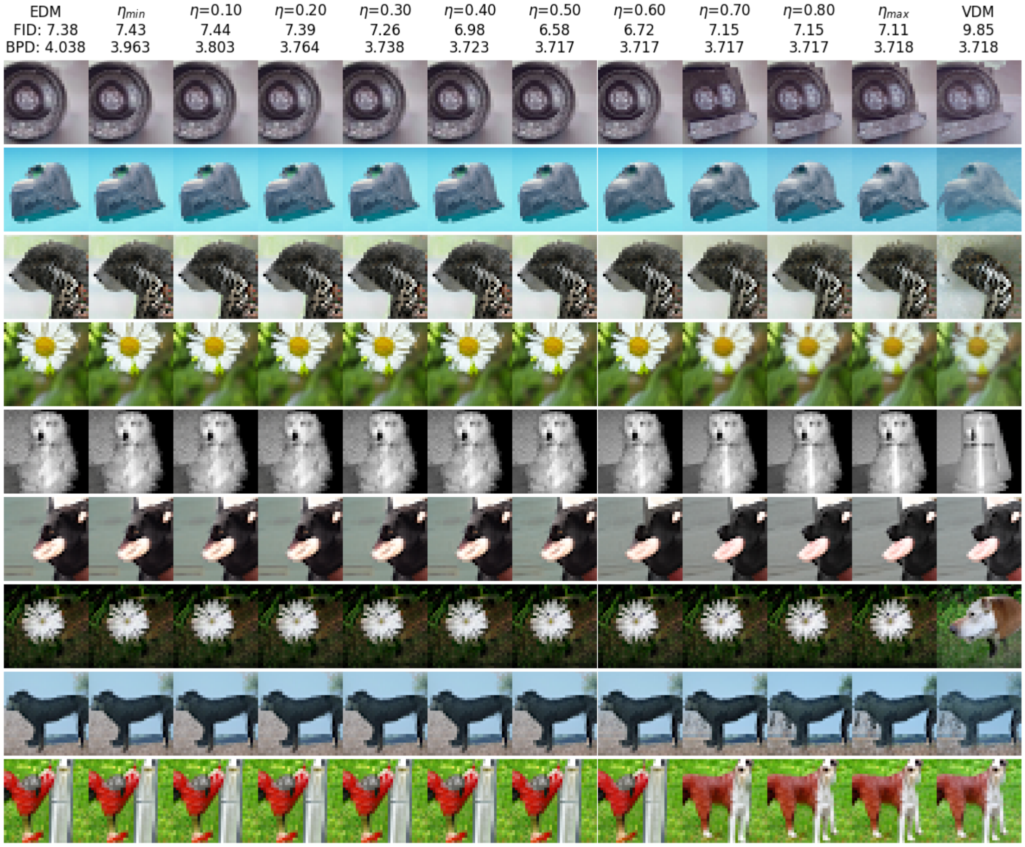

Figure 6: Visualization of generated images using different thresholds η on ImageNet32 dataset

We also present randomly generated samples (without fixed seeds) from our proposed method and baseline models, shown in Fig. 7 through Fig. 10. These visualizations include generations on CIFAR-10 using both the ODE and VDM ancestral samplers across a range of η values.

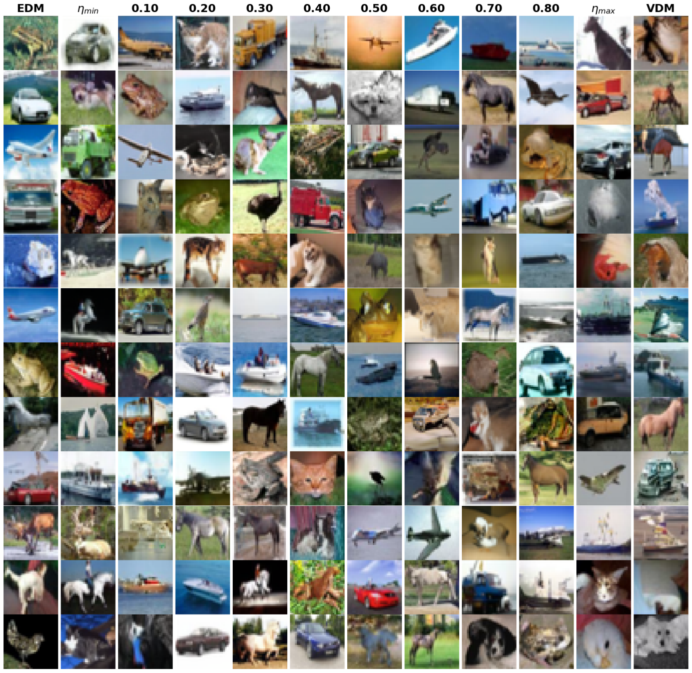

Figure 7: Random samples on CIFAR-10 using ODE sampler (with different η).

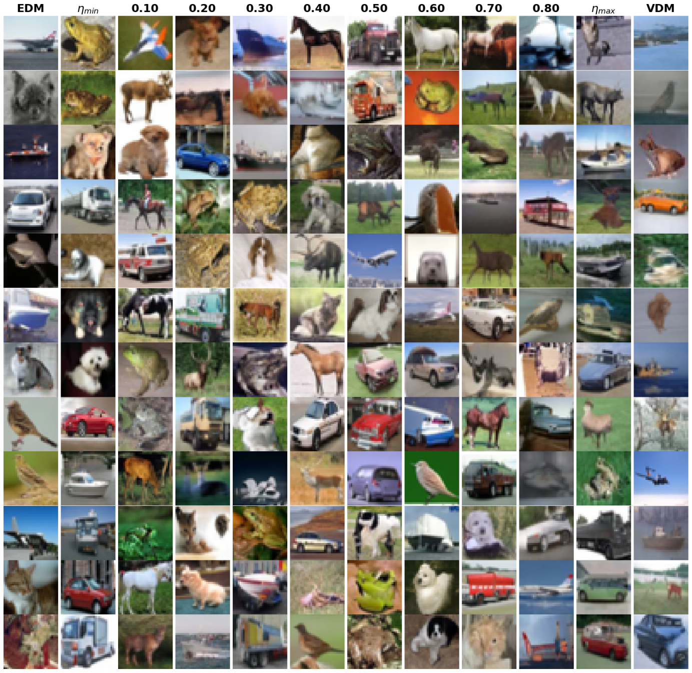

Figure 8: Random samples on CIFAR-10 using VDM ancestral sampler (with different η).

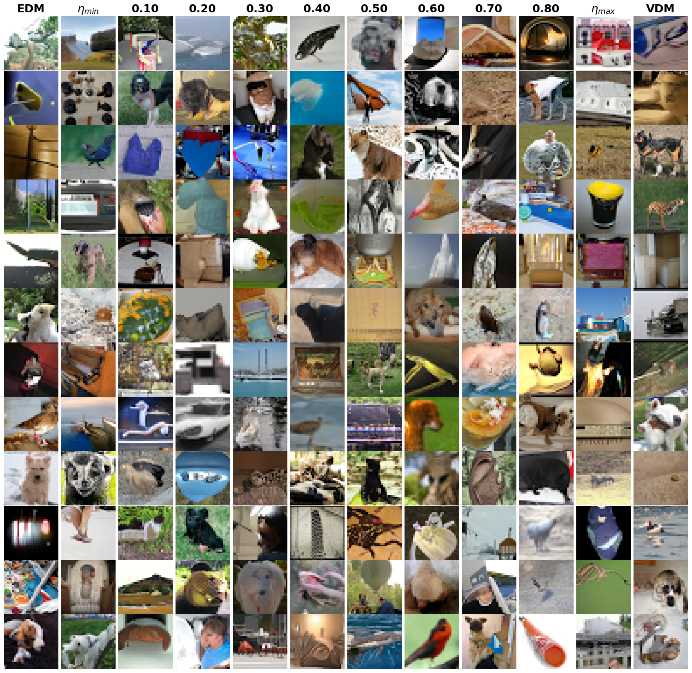

Figure 9: Random samples on ImageNet32 using ODE sampler (with different η).

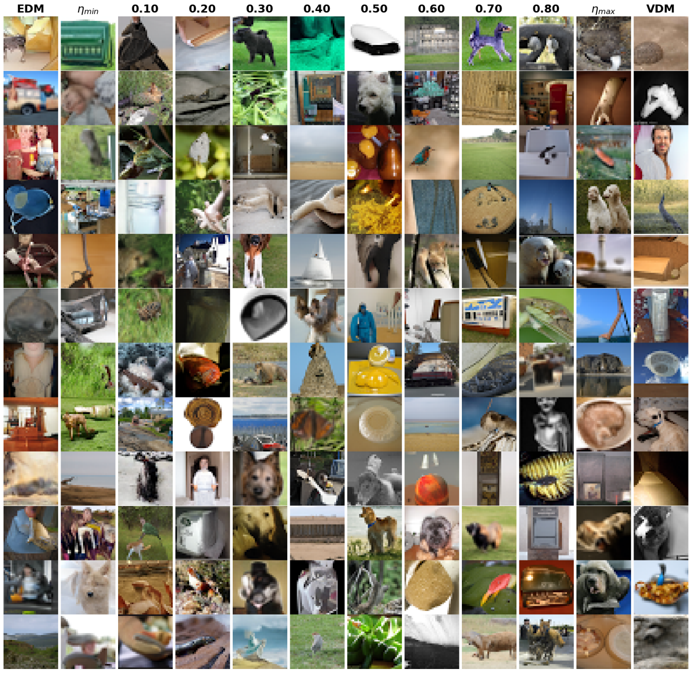

Figure 10: Random samples on ImageNet32 using VDM ancestral sampler (with different η).

C.3 Full Comparison Table

Table 6 provides an extended version of Table 3, including additional methods from the literature.

Table 6: The full comparison table for comparing our results with different models. By default, the evaluation of NLL is by Truncated Normal Dequantization, otherwise marked with other signs; Uniform Deq.†, Variational Deq.‡, VLB$^{\vee}$, Data Augmentation$^{\uplus}$., ImageNet32(old version)*.

Model	**CIFAR10**			**ImageNet32**		
	NLL(↓)	FID(↓)	NFE	NLL(↓)	FID(↓)	NFE
Main Baselines						
VDM (Kingma et al., 2021)	2.65$^{\vee}$	7.41	-	3.72$^{*\vee}$	-	-
EDM (w/ Heun Sampler) (Karras et al., 2022)	-	1.97	35	-	-	-
Focused on both FID-NLL						
Soft Truncation (Kim et al., 2021)	3.01†	3.96	-	3.90*†	8.42*	-
CTM ($\uplus$ - randomflip) (Kim et al., 2023)	2.43†	1.87	2	-	-	-
ScoreSDE ($\uplus$ - randomflip) (Song et al., 2020b)	2.99†	2.92	-	-	-	-
LSGM (FID) (Vahdat et al., 2021)	3.43	2.10	-	-	-	-
DDPM++ cont. (deep, sub-VP) (Song et al., 2020b)	2.99†	2.92	-	-	-	-
Reflected Diffusion Models (Lou & Ermon, 2023)	2.68	2.72	-	3.74	-	-
Focused on FID						
GMEM (Transformer-based) (Tang et al., 2024)	-	1.22	50	-	-	-
PaGoDA (distillation-based) (Kim et al., 2024)	-	-	-	-	0.79	1
SiD (distillation-based) (Zhou et al., 2024)	-	1.923	1	-	-	-
ScoreFlow (VP, FID) (Song et al., 2021)	3.04‡	3.98	-	3.84*‡	8.34*	-
PNDM (Liu et al., 2022)	-	3.26	-	-	-	-
Focused on NLL						
i-DODE (VP) (Zheng et al., 2023b)	2.57	10.74	126	3.43/3.70*	9.09	152
i-DODE (VP, $\uplus$) (Zheng et al., 2023b)	2.42	3.76	215	-	-	-
Flow Matching (Lipman et al., 2022)	2.99†	6.35	142	3.53†	5.02	122
DiffEnc (Nielsen et al., 2023)	2.62$^{\vee}$	11.1	-	3.46$^{\vee}$	-	-
NDM ($\uplus$ - horizontalflip) (Bartosh et al., 2023)	2.70†	-	-	3.55	-	-
NFDM (Gaussian q, $\uplus$ - horizontalflip) (Bartosh et al., 2024)	2.49†	21.88	12	3.36	24.74	12
NFDM (non-Gaussian q, $\uplus$ - horizontalflip) (Bartosh et al., 2024)	2.48†	-	-	3.34	-	-
NFDM-OT($\uplus$ - horizontalflip) (Bartosh et al., 2024)	2.62†	5.20	12	3.45	4.11	12
ScoreFlow (deep, sub-VP, NLL) (Song et al., 2021)	2.81‡	5.40	-	3.76*‡	10.18*	-
Stochastic Interp. (Albergo & Vanden-Eijnden, 2022)	2.99†	10.27	-	3.48†	8.49	-
MuLAN (w/o importance sampling k=1) (Sahoo et al., 2023)	2.59	-	-	3.71	-	-
MuLAN (w/ importance sampling k=20) (Sahoo et al., 2023)	2.55	-	-	3.67	-	-
Improved DDPM (L_{vlb}) (Nichol & Dhariwal, 2021)	2.94$^{\vee}$	11.47	-	-	-	-
FFJORD (Grathwohl et al., 2018)	3.4	-	-	-	-	-
Improved DDPM (L_{vlb}) (Nichol & Dhariwal, 2021)	2.94$^{\vee}$	11.47	-	-	-	-
ARDM-Upscale 4(autoregressive) (Hoogeboom et al., 2021)	2.64	-	-	-	-	-
Efficient-VDVAE (Hazami et al., 2022)	2.87$^{\vee}$	-	-	3.58	-	-
DenseFlow-74-10 (Grcić et al., 2021)	2.98‡	34.90	-	3.63	-	-
Ours						
VDM (our evaluation, $\gamma \in$[-13.3, 5]) (Kingma et al., 2021)	2.64/2.66$^{\vee}$	9.37	206	3.72*/3.72$^{*\vee}$	9.85*	158
EDM (our evaluation, $\gamma \in$[-12.43, 8.764]) (Karras et al., 2022)	3.21	2.02	125	4.04*	7.38*	120
Ours NLL ($\eta = 0.4$, CIFAR10)	**2.62**	2.14	173	-	-	-
Ours ($\eta = 0.3$, CIFAR10)	2.63	**2.01**	169	-	-	-
Ours ($\eta = 0.5$, ImageNet32)	-	-	-	**3.72***	**6.58***	180