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

Recent advances in diffusion models achieved the highest perceptual quality. However, previous studies have shown that there is an inverse correlation between the perceptual quality and data-likelihood, as the data-likelihood is more influenced by low-level noise and the perceptual quality is more reliant on the high-level noise in diffusion models. Consequently, there exists a trade-off between the sample-quality and data-likelihood, and usually, models are trained to enhance one of those. In this paper, we present a simple-yet-novel method to alleviate this trade-off by fusing two different pre-trained diffusion models as a Mixture-of-Experts approach. For high noise levels, we use an expert model on perceptual quality, and for the low noise level, we leverage an expert model on data-likelihood. In experiments, our merged model achieves the best of both base models: comparable or better likelihood value compared to its expert likelihood component while almost reaching the perceptual quality of its expert on image quality.

1 INTRODUCTION

Generative models are a class of probabilistic Machine learning where they learn to approximate the underlying probability distribution of a dataset, allowing them to create new images that resemble the original data (Bartosh et al., 2023). Diffusion models (DMs) are a subclass of these models that match a data distribution by learning a reverse process to undo a noising forward process (Sohl-Dickstein et al., 2015; Ho et al., 2020; Nichol & Dhariwal, 2021). They have recently become state-of-the-art in Image-generation tasks (Dhariwal & Nichol, 2021; Tang et al., 2024; Kim et al., 2024), in density estimation (Kingma et al., 2021), and exhibit superior performance on text-to-image and text-to-video generation (Esser et al., 2024; Polyak et al., 2024).

For images, density estimation (likelihood) and image quality seem to often be quite independent in practice (Theis et al., 2015), i.e, good performance with respect to one criterion therefore need not imply good performance with respect to the other criteria. More specifically, Kim et al. (2021) stated empirical results from previous studies show an *inverse correlation* between likelihood and sample generation performances; so a model with a focus on Negative Log-Likelihood (NLL), such as (Kingma et al., 2021), uses likelihood weighting for training, while a model with a focus on sample generation quality, Frechet Inception Distance (FID), such as (Kingma & Gao, 2024; Nichol & Dhariwal, 2021) use a different weighting for training. Nevertheless, such models have a trade-off between NLL and FID; models with the emphasis on FID perform poorly on NLL and vice versa.

In this paper, we are overcoming this NLL-FID trade-off, i.e., by designing a model that can both generate good image quality while having good data-likelihood. Outlining the core concept of our idea, we proposed to *merge* two diffusion models; one is an expert on good image-quality generation, and the other is an expert on good data-likelihood. For the high-level noise, use an expert on good image-quality model (EDM, Karras et al. 2022), while for the low-level noise, use an expert on data-likelihood (VDM, Kingma et al. 2021); see Fig. 1. If we start from the noise using a good

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

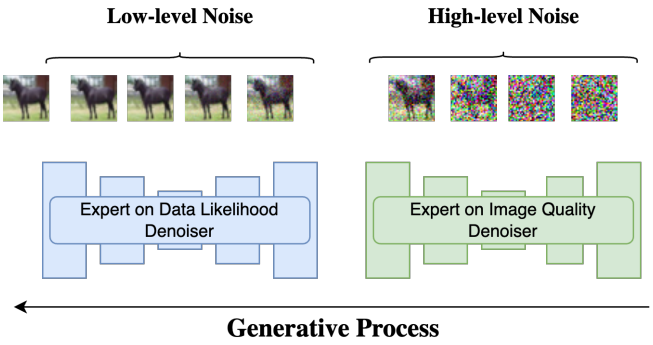

Figure 1: Diagram of our proposed merged model where at time $\eta = 0.5$ we switch between denoisers. Note that the likelihood model is only used for almost imperceptible noise levels. This significantly improves the likelihood, which is sensitive to low-level color statistics, while leaving the FID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

image-quality model, after some steps, we will have a good quality image, then we switch to a good likelihood model and while keeping a good-quality image, we are trying to get a good-likelihood in the rest of the steps. Intuitively, likelihood is, in practice, very sensitive to low-level details, i.e., when we are in the low-level noise time steps (Zheng et al., 2023b; Kim et al., 2021). By choosing the exact step to switch, we could almost get the best value on both metrics.

In Section 2, we discuss the necessary background, viewing why this trade-off happens, and in Section 3, we show the related works that addressed it. In Section 4, we propose our method in detail, including how the sampling and likelihood evaluation change. Section 5 contains the experiments and results we obtained. In Sections 6 and 7, we mention the limitations of our method and conclude.

2 Background

2.1 Diffusion Models

Diffusion models (Kingma et al., 2021; Ho et al., 2020; Sohl-Dickstein et al., 2015) are a type of generative models that consist of two processes. Starting with a D-dimensional random variable $\mathbf{x}$(images), we define the *Forward Process* with noisy latent variables $\mathbf{z}_t$, where t runs from $t = 0$(least noisy) to $t = 1$(most noisy), satisfies the transition kernel $q(\mathbf{z}_t \mid \mathbf{x}) = \mathcal{N}\left(\mathbf{z}_t; \alpha_t \mathbf{x}, \sigma_t^2 \mathbf{I}\right)$, where $\alpha_t, \sigma_t \in \mathbb{R}_{>0}$ are smooth scalar-valued functions of t, named *noise schedule* parameters. We assume that *signal-to-noise* ratio $SNR(t) = \alpha_t^2/\sigma_t^2$ is strictly monotonically decreasing w.r.t. t. The transition can be easily applied by $\mathbf{z}_t = \alpha_t \mathbf{x} + \sigma_t \boldsymbol{\epsilon}, \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, where $\mathbf{z}_1$ is approximately a pure Gaussian. By discretizing of time range (0, 1) into $T > 0$ timesteps of width $\frac{1}{T}$, we define $t(i) = \frac{i}{T}$ and $s(i) = \frac{i-1}{T}$ and we use $\mathbf{z}_{0:1}$ to denote the subset of variables associated with these timesteps. The *reverse process* (generative) is defined as $p(\mathbf{z}_s \mid \mathbf{z}_t) = q(\mathbf{z}_s \mid \mathbf{z}_t, \mathbf{x} = \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t, t))$, which is parameterized by a denoising Neural Network that predicts x from its noisy version $\mathbf{z}_t$. In practice, we parameterize the Network as a *noise-prediction* model, in which we optimize the parameters towards the Variational Lower Bound (VLB) of the marginal likelihood, defined as:

$$
\begin{aligned}
-\log p(\mathbf{x}) \leq -\mathrm{VLB}(\mathbf{x}) &= \underbrace{D_{\mathrm{KL}}(q(\mathbf{z}_1 \mid \mathbf{x}) \| p(\mathbf{z}_1))}_{\text{Prior loss}} + \underbrace{\mathbb{E}_{q(\mathbf{z}_0 \mid \mathbf{x})}[-\log p(\mathbf{x} \mid \mathbf{z}_0)]}_{\text{Reconstruction loss}} + \underbrace{\mathcal{L}_T(\mathbf{x})}_{\text{Diffusion loss}}, \\
\mathcal{L}_T(\mathbf{x}) &= \sum_{i=1}^{T} \mathbb{E}_{\mathbf{z}_{t(i)} \sim q(\mathbf{z}_{t(i)} \mid \mathbf{x})} D_{\mathrm{KL}}\left[q(\mathbf{z}_{s(i)} \mid \mathbf{z}_{t(i)}, \mathbf{x}) \| p_{\boldsymbol{\theta}}(\mathbf{z}_{s(i)} \mid \mathbf{z}_{t(i)})\right]
\end{aligned}
\tag{1}
$$

Kingma et al. (2021) proved that in the continuous case ($T \to \infty$), the $\mathcal{L}_T(\mathbf{x})$ simplifies further to a *noise-prediction* loss Eq. (2), where in the *variance-preserving* diffusion we have $\alpha_t = \sqrt{\mathrm{sigmoid}(-\gamma_t)}, \sigma_t = \sqrt{\mathrm{sigmoid}(\gamma_t)}$, and $\mathbf{z}_t = \alpha_t \mathbf{x} + \sigma_t \boldsymbol{\epsilon}$.

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

The continuous case can be equivalently defined as a linear stochastic differential equation (*SDE*) (Song et al., 2020b; Lu et al., 2022), with the following *Forward Process*:

$$\mathrm{d}\mathbf{z}_t = f(t)\mathbf{z}_t \,\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}_t, \quad \mathbf{z}_0 \sim q\left(\mathbf{z}_0 \mid \mathbf{x}\right), \tag{3}$$

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

$$f(t) = \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}, \quad g^2(t) = \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\sigma_t^2. \tag{4}$$

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:

$$\mathrm{d}\mathbf{z}_t = \left[f(t)\mathbf{z}_t + \frac{g^2(t)}{\sigma_t}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}\left(\mathbf{z}_t, t\right)\right]\mathrm{d}t + g(t)\mathrm{d}\overline{\mathbf{w}}_t, \quad \mathbf{z}_1 \sim \mathcal{N}\left(\mathbf{0}, \tilde{\sigma}^2\mathbf{I}\right) \tag{5}$$

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:

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

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. (7), where we are integrating the divergence of the drift function:

$$\log p_{\boldsymbol{\theta}}\left(\mathbf{z}_0\right) = \log p_{\boldsymbol{\theta}}\left(\mathbf{z}_1\right) - \int_{t=0}^{t=1} \mathrm{tr}\left(\nabla_{\mathbf{z}_t}\mathbf{h}_{\boldsymbol{\theta}}\left(\mathbf{z}_t, t\right)\right)\mathrm{d}t \tag{7}$$

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. (8), with different choices of weighting. Using the uniform weighting, $w(\gamma_t) = 1$, that corresponds to ELBO objective Eq. (2) 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).

$$\mathcal{L}_w(\mathbf{x}) = \frac{1}{2}\mathbb{E}_{t\sim\mathcal{U}(0,1),\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[w\left(\gamma_t\right)\cdot\frac{\mathrm{d}\gamma_t}{\mathrm{d}t}\cdot\|\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}\left(\mathbf{z}_t;\gamma_t\right) - \boldsymbol{\epsilon}\|_2^2\right], \quad \gamma_t = -\log(SNR_t) \tag{8}$$

3 Related Works

In this section, we discuss how the previous works in the literature can be relevant to ours. To do so, we try to categorize them based on which one of the objectives (sample-quality or data-likelihood), they are mainly optimizing in their methods. They could also optimize both. Lastly, we mention some of the available methods that could be related due to leveraging Mixture-of-Experts.

Likelihood experts. Some methods like VDM (Kingma et al., 2021) and ScoreFlow (Song et al., 2021) directly optimize (a bound to) the log-likelihood. i-DODE (Zheng et al., 2023b) proposed *velocity-prediction* and a better method for likelihood estimation. Sahoo et al. (2023); Nielsen et al. (2023); Bartosh et al. (2024) proposed a learnable forward diffusion model, while we focus on the fixed linear DMs in this study.

Sample quality experts. There more papers which focused on quality of image generation; some by introducing better or faster 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), by applying techniques, e.g. mitigating exposure bias (Ning et al., 2023), introducing a monotonic weighting function, or special weighting term (Kingma & Gao, 2024; Ho et al., 2020). GMEM (Tang et al., 2024) uses an external memory bank, which enhances quality and efficiency when used with a transformer that achieves SOTA FID on CIFAR-10, and PaGoDA (Kim et al., 2024), a distillation-based method, is SOTA FID on ImageNet32, to best of our knowledge. In our study, we focused on UNet-based DMs with simpler objective functions like *noise-prediction* and excluded distillation methods.

Expert on both metrics. Kim et al. (2021) propose a method for training DMs to have a good performance on both metrics (Soft-Truncation). While they have an aligned purpose with ours, the difference is that our method can leverage pre-trained models without any further training. CTM (Kim et al., 2023) uses multiple loss terms, including an additional GAN loss, along with data augmentation to achieve good results on both metrics, but we handle this trade-off from another perspective, by fusing models with simpler objective functions, i.e., *noise-prediction* $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{z}_t; \gamma_t)$, *image-prediction* $\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{z}_t; \gamma_t)$.

Mixture-of-Experts. These technique was incorporated into DMs for zero-shot text-to-image synthesis (Balaji et al., 2022; Feng et al., 2023) and controllable generation (Bar-Tal et al., 2023). Recently, MDM (Kang et al., 2024) proposed a Mixture-of-Expert approach in which each expert is trained on its designated time interval. However, each expert has the same architecture as others, and the focus is on boosting training efficiency and sample-quality. To the best of our knowledge, we are the first to propose using multiple experts to alleviate the sample quality-likelihood imbalance.

4 Model Fusion

In this section, we explain our method based on $\gamma_t = -\log(SNR_t)$ (Kingma et al., 2021). As previously mentioned, the *Signal-to-Noise* ratio decreases as we go from the data distribution towards the noise. However, the γ_t is a monotonic increasing function, so the values close to data distribution are smaller than the values closer to noise. We proposed to *merge* two different diffusion models, in which each of them is good at one of the image-quality, measured by FID (Heusel et al., 2017), or data-likelihood, measured by bits/dimension(BPD) as follows. For the high γ_t region (high-level noise), we use an expert model on good image-quality, and for the low γ_t region (low-level noise), we use the an expert model on good-likelihood(see Fig. 1). This is because the good-likelihood comes from emphasizing on small time steps, while good image-quality comes from emphasizing on large time steps (Zheng et al., 2023b; Kim et al., 2021).

In this paper, we leverage EDM (Karras et al., 2022) model as an expert on good sample quality for the high noise (time step interval τ_2), and VDM (Kingma et al., 2021) model as an expert on good data-likelihood for the low noise region (time step interval τ_1). Given a threshold time step η on the overlapping γ range of $\tau_1 \cap \tau_2$, our merged model $f_{\boldsymbol{\theta}(t)}(\mathbf{z}_t, \gamma_t)$ shown in Fig. 1, as a denoising auto-encoder, can be denoted as follow:

$$\boldsymbol{\theta}(t) = \begin{cases} \boldsymbol{\theta}_{VDM}, & t \leq \eta, \\ \boldsymbol{\theta}_{EDM}, & t > \eta. \end{cases} \tag{9}$$

4.1 Sampling

Starting from a noise sample, we denoise the image using the EDM model until we reach the time step threshold η, and for the rest of the steps, we use VDM model until we get clean images. The Algorithms 1–2 demonstrate the steps, and are modified version of samplers from Zheng et al. (2023b) and Kingma et al. (2021), respectively. The last step in both algorithms is the reconstruction term (Kingma et al., 2021) in that we get 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 γ_{min}; Largest time step γ_{max};
  $\mathbf{z}_T \sim \mathcal{N}\left(\mathbf{0}, \sigma_T^2 \mathbf{I}_D\right)$
  Compute intermediate time step using the *fixed linear noise schedule*
    $\gamma_\eta = \gamma_{min} + \eta * (\gamma_{max} - \gamma_{min})$
  $\mathbf{z}_\eta \leftarrow \text{PF-ODE}_{EDM}\left(\gamma_{max}, \gamma_\eta, \mathbf{z}_T\right)$
  $\mathbf{z}_0 \leftarrow \text{PF-ODE}_{VDM}\left(\gamma_\eta, \gamma_{min}, \mathbf{z}_\eta\right)$
  $\mathbf{x} \sim p(\mathbf{x} \mid \mathbf{z}_0)$
end procedure

Algorithm 2 Ancestral Sampling

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

4.2 LIKELIHOOD EVALUATION

We considered two methods for evaluating the likelihood of the model on the datasets. One method is the variational lower bound of likelihood (VLB) (Kingma et al., 2021), and the other is computing the exact likelihood value (Zheng et al., 2023b; Song et al., 2020b; 2021; Sahoo et al., 2023).

Variational Lower Bound (VLB) The VLB consists of three terms as shown in Eq. (1). For computing the likelihood using this method in our case, we do the following. The Prior loss and Reconstruction loss can be computed regardless of models based on the γ value, and only the Diffusion loss is different. Given a batch of images, we calculated the Diffusion loss Eq. (2), using $\boldsymbol{\theta}_{VDM}$ if the time step of noisy image $\mathbf{z}_t$, denoted as t, is less than equal the threshold η, and $\boldsymbol{\theta}_{EDM}$ if the time step is greater the threshold η. Please refer to equation 9.

Exact likelihood computation with Probability-flow ODE For an alternative method of evaluation, we use the PF-ODE based on γ Eq. (10). In our merged model Eq. (9), we have two separate PF-ODEs, one for each of the models. Starting from the almost clean data $\mathbf{z}_0$, we integrate from the initial time step γ_{min} until the threshold time step γ_η using PF-ODE_{VDM}, and for the rest from time step γ_η until time step γ_{max} we use PF-ODE_{EDM}.

5 EXPERIMENTS

In this section, we compare our results with state-of-the-art models on both generated image-quality and data-likelihood.

5.1 EXPERIMENTAL SETUP

Datasets. We evaluate our proposed model on the test set of CIFAR-10 Krizhevsky & Hinton (2009) and ImageNet32 (Deng et al., 2009) datasets. While there are two version of ImageNet32 in the literature, we used the old version, for all of our experiments (marked with asterisk * when we compare to other methods), to be consistent with the previous methods. For CIFAR-10, we had the training details to reproduce the results. However, for ImageNet32, due to time limits, we did not optimize the hyperparameters, and we have sub-optimal base models. The focus of our study is CIFAR-10, and the ImageNet32 results are to check if there exist similar trends.

Metrics and evaluation setup. For the sample-quality we used Fréchet Inception Distance (FID) (Heusel et al., 2017) metric and compared 50k generated images with reference statistics for the datasets following scripts from (Karras et al., 2022), while for the data-likelihood under the models, we used bits/dimension (BPD). As the exact likelihood computation shows us better results than VLB, we consider that for our main experiments. For the VLB evaluation, please refer to Table 5 in Appendix C.

Baselines. We used the base models that we merged (VDM and EDM) as baselines since we wanted to show our *merged* model works better than each one of the base models separately. They

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

		Threshold										
		0.0475	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**VDM**
CIFAR10	**NLL**	3.09	2.86	2.73	2.65	**2.62**	2.62	2.62	2.63	2.63	2.63	2.64
	NFE	210	210	213	217	233	238	235	234	234	248	244
ImageNet32	**NLL**	3.96	3.82	3.78	3.75	3.73	3.72	**3.72**	3.72	3.72	3.72	3.72
	NFE	174	177	179	174	169	169	186	196	207	214	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										
		0.0475	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**VDM**
CIFAR10	**VDM**	**2.88**	2.92	2.91	2.89	2.89	2.98	3.66	5.6	7.62	8.93	9.19
	ODE	2.85	2.84	**2.84**	2.84	2.86	2.96	3.45	4.92	7.02	8.68	9.37
	NFE	127	131	137	143	149	159	177	192	211	223	206
ImageNet32	**VDM**	8.30	8.30	8.25	8.15	7.95	7.59	**7.41**	8.07	9.36	9.81	9.89
	ODE	8.80	8.80	8.79	8.74	8.62	8.39	**8.06**	8.16	8.95	9.63	9.85
	NFE	124	129	130	136	141	147	153	171	175	180	158

are evaluated on their whole defined noise ranges. We also compared our method with some of the related methods mentioned in Table 3. Please refer to Table 6 in the Appendix for other related methods.

In our experiments, we focused on the Variance-Preserving(VP) diffusion model settings, where $\sigma_t^2 = \text{sigmoid}(\gamma_t), \alpha_t^2 = 1 - \sigma_t^2$, which operates on the pixel space, so the distillation methods and latent space models are excluded. For our experiments, we re-implemented the PyTorch version of Truncated-Normal dequantization and ODE sampler in the code repository [1] of the i-DODE paper (Zheng et al., 2023b), which is based on time step variable γ in Eq. (10). Please refer to Appendix A.1.1 for our derivation.

The VDM model with the time step interval τ_1 is defined on $\gamma_{VDM} \in [-13.3, 5]$, while the EDM model time step interval τ_2 is defined on $\gamma_{EDM} \in [-12.43, 8.764]$. We considered the $\gamma_{Merged} \in [-13.3, 5]$ and selected the time step threshold η on the $\{4.75\%, 10\%, 20\%, 30\%, 40\%, 50\%, 60\%, 70\%, 80\%, 90\%\}$ of this mentioned range. The first threshold value corresponds to the $\gamma = -12.43$, which is the smallest time step on the overlapping γ ranges. Additionally, $\eta = 1.0$ is the same as VDM baseline. Please refer to Appendix B.1 for implementation details.

5.2 Results

5.2.1 Effect of using different time threshold η

We evaluated our proposed technique using different time step thresholds η, and reported NLL in terms of BPD with Truncated-Normal dequantization without importance weighted estimator ($K = 1$) in Table 1. Furthermore, using the same settings, we reported FID for unconditional generation, along with the number of function evaluations (NFE) for image generation in Table 2. We used 256 steps for generating images using VDM Ancestral. For the ImageNet plots, refer to Appendix C

Fig. 2 illustrates the performance of our developed framework using different thresholds on all metrics. Examining the figure shows a clear trade-off between the Likelihood and Image-quality. Our method gets the best value for Likelihood at time step threshold $\eta = 0.4$, and the best value for Image-quality at $\eta = 0.2$ for CIFAR-10 dataset. Interestingly, the likelihood at $\eta = 0.4$ is even better than the VDM baseline, with a small 0.1 degradation in FID from EDM(2.86 instead of 2.85). Please refer to Appendix C for the exact values of the likelihood evaluation using Uniform dequan-

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

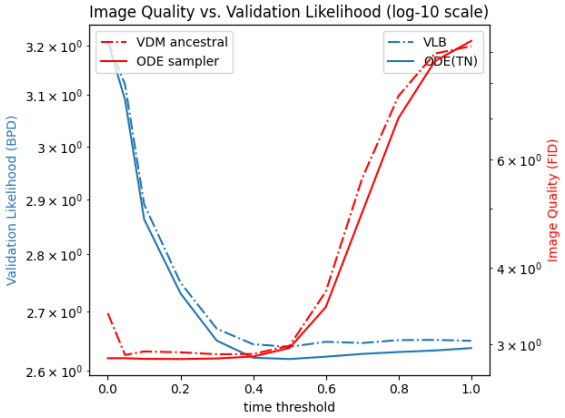

Figure 2: Visualization of our merged model performance using different time step thresholds η. on CIFAR-10

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

tization and VLB. For the ImageNet32 dataset, $\eta = 0.6$ gives us the same likelihood as VDM but with FID improvement. We reported the evaluations on both EDM and VDM on their defined γ range as other baselines. While these are the best, we would like to emphasize that it is also possible to pick a single η that is almost the best on both metrics.

Fig. 3 shows the qualitative results of our proposed method on CIFAR-10 using ODE sampler. When we use only the EDM model(left column) we have good sample quality but the likelihood is bad. As we start to switch to VDM model, we keep the sample quality the same, but the likelihood improves (see $\eta = 0.1$ or $\eta = 0.3$), and this exactly showcases the intuition behind our proposed method. For larger thresholds, the sample quality drops, but the likelihood improves. Notably, even though we used different base models for CIFAR-10, they generate the same samples given a fixed noise, both individually and in our proposed method (see Kadkhodaie et al. (2023) regarding this generalization).

5.2.2 Comparing to other methods

In Table 3, we presented our results using Truncated-Normal dequantization and adaptive-step ODE sampler, comparing them with other methods from the literature. We only compared the evaluation of our method using exact likelihood computation and ODE sampler, as we believe they are better for evaluation and give us better results.

Among the related methods, we surpassed Soft Truncation with the same focus on both metrics. i-DODE used velocity-parameterization besides an error-bounded high-order Flow Matching objective to get better NLL values. CTM uses data augmentation along with multiple loss terms to achieve their results, but we are handling this trade-off between metrics from another perspective by using models with simple losses (i.e., *noise-prediction*, *image-prediction*, *score-matching*). Moreover, our method does not use data augmentation for the VDM training, and we used the default settings for EDM. NFDM, MuLAN, and DiffEnc use a more complicated learnable forwards process. While we have fixed-linear forward process, we could get comparable results with DiffEnc and NFDM-OT. GMEM, PaGoDa, and SiD are focused on FID and use either distillation method or transformer architecture, while we focused on UNet architecture. We would like to point out that we are not claiming state-of-the-art, though claiming that if we merge two DMs, one expert on image quality and the other expert on data-likelihood, we will get better values for both metrics than if you use each of them individually.

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$^{\vee}$	9.38	206	3.72*/3.72$^{*\vee}$	9.85*	158
EDM (our evaluation, $\gamma \in$[-12.43, 8.764]) (Karras et al., 2022)	3.21	2.85	127	4.04*	7.38*	120
Ours NLL ($\eta = 0.4$, CIFAR10)	**2.62**	2.86	149	-	-	-
Ours FID ($\eta = 0.2$, CIFAR10)	2.73	**2.84**	137	-	-	-
Ours FID+NLL ($\eta = 0.6$, ImageNet32)	-	-	-	**3.72***	8.06*	153

6 Limitations and Future Works

Our method is heavily reliant on the models we choose to merge. We only considered the fixed-linear scheduled diffusion models with simple *noise-prediction* objective. Moreover, we did not consider the models that use distillation or operate the latent space, as we focus on pixel-space only. Additionally, we argue that the performance of our *Merged Model* for FID metrics would get much better if better samplers, like Heun (Karras et al., 2022) or DPM-Solver-v3 (Zheng et al., 2023a), could be incorporated. Additionally, to find a proper threshold to have both good image-quality and good data-likelihood we need to do a search. We leave using different architectures and samplers for future works.

7 Conclusion

In this work, we proposed a simple-yet-novel approach to addressing the trade-off between sample quality and data likelihood in diffusion models. By leveraging a Mixture-of-Experts framework, we merged two specialized models—one optimized for high-quality image generation and the other for strong likelihood estimation. Our method allows for a seamless transition between these two domains, ensuring that the generated samples maintain both high perceptual quality and favorable likelihood values.

Through experiments on CIFAR-10 and ImageNet32, we demonstrated that our merged model effectively balances these two objectives. Our results show that by carefully selecting the time step threshold for switching between models, we can achieve better likelihood values than the likelihood-focused baseline while maintaining competitive image quality. Additionally, our approach offers flexibility, as one could use two pre-trained models as base models.

While our method successfully mitigates the trade-off, certain limitations remain. The choice of expert models and the threshold for switching between them can influence performance. Additionally, exploring more advanced sampling techniques and alternative architectures could further enhance results. Ultimately, our work provides a promising direction for improving diffusion models by balancing competing objectives without sacrificing performance in either domain.

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 [2].

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.

[2] 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.

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, 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 $\gamma_t = -\log(SNR_t)$:

$$\frac{\mathrm{d}\mathbf{z}_t}{\mathrm{d}\gamma_t} = -\frac{1}{2} \cdot \text{Sigmoid}(\gamma_t) \cdot \mathbf{z}_t + \frac{1}{2} \cdot \sigma_t \cdot \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t) \tag{10}$$

Let us simplify $g^2(t)$ in terms of $\lambda_t := \log(\alpha_t/\sigma_t) = -\frac{1}{2}\gamma_t$ from Eq. (4):

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

$$g^2(t) = \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\sigma_t^2 = 2\sigma_t^2\left(\frac{\mathrm{d}\log\sigma_t}{\mathrm{d}t} - \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\right) = -2\sigma_t^2\frac{\mathrm{d}\lambda_t}{\mathrm{d}t} \tag{11}$$

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. (6) 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. (7) would be $[\gamma_0, \gamma_1]$ with the above drift function.

$$\log p_{\boldsymbol{\theta}}\left(\mathbf{z}_0\right) = \log p_{\boldsymbol{\theta}}\left(\mathbf{z}_1\right) - \int_{\gamma_0}^{\gamma_1} \mathrm{tr}\left(\nabla_{\mathbf{z}_t}\mathbf{h}_{\boldsymbol{\theta}}\left(\mathbf{z}_t, \gamma_t\right)\right)\mathrm{d}\gamma \tag{12}$$

A.1.2 DEQUANTIZATION

Real-world image datasets are usually based on discrete data X as 8-bit integers $\{0, 1, 2, ..., 255\}$, and it is common to normalize data into [-1, 1], represented by $\mathbf{x}$ (Zheng et al., 2023b; Sahoo et al., 2023). Dequantization methods assume that we have trained a continuous model distribution $p_{\boldsymbol{\theta}}$ for $\mathbf{x}$, and define the discrete model distribution as:

$$P_{\boldsymbol{\theta}}\left(\mathbf{x}\right) = \int_{\mathbf{u} \in \left[-\frac{1}{256}, \frac{1}{256}\right]^d} p_{\boldsymbol{\epsilon}}(\mathbf{x} + \mathbf{u})\mathrm{d}\mathbf{u} \tag{13}$$

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 train $p_{\boldsymbol{\epsilon}}$ and $q(\mathbf{u}|\mathbf{x})$ by a variational lower bound:

$$\log P_{\boldsymbol{\theta}}(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{u}|\mathbf{x})}\left[\log p_{\boldsymbol{\epsilon}}(\mathbf{x}+\mathbf{u}) - \log q(\mathbf{u} \mid \mathbf{x})\right] \tag{14}$$

The term $\log p_{\boldsymbol{\epsilon}}(\mathbf{x}+\mathbf{u})$ can be solved by instantanous change of variable (Chen et al., 2018) (see Eq. (12)).

Zheng et al. (2023b) proposed *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):

$$\hat{\boldsymbol{\epsilon}} \sim \mathcal{TN}\left(\mathbf{0}, \mathbf{I}, -\frac{1}{256}, \frac{1}{256}\right) \tag{15}$$

The Eq. (14) further simplifies to the equation below (detailed explanation in appendix (Zheng et al., 2023b)), with $u := \frac{\sigma_{\boldsymbol{\epsilon}}}{\alpha_{\boldsymbol{\epsilon}}}\hat{\boldsymbol{\epsilon}} \in \left[-\frac{1}{256}, \frac{1}{256}\right]$ and error function $Z := \operatorname{erf}\left(\frac{\tau}{\sqrt{2}}\right)$:

$$\log P_{\boldsymbol{\theta}}(\mathbf{x}) \geq \mathbb{E}_{\hat{\boldsymbol{\epsilon}}\sim\mathcal{TN}(\mathbf{0},\mathbf{I},-\tau,\tau)}\left[\log p_{\boldsymbol{\epsilon}}(\hat{\mathbf{x}}_{\boldsymbol{\epsilon}})\right] + \frac{d}{2}\left(1 + \log\left(2\pi\sigma_{\boldsymbol{\epsilon}}^2\right)\right) + d\log Z - d\frac{\tau}{\sqrt{2\pi}Z}\exp\left(-\frac{1}{2}\tau^2\right) \tag{16}$$

Using $\gamma_{\boldsymbol{\epsilon}} = 13.3$ leads to $\tau \approx 3$ and the truncated-normal distribution $\mathcal{TN}(\mathbf{0}, \mathbf{I}, -\tau, \tau)$ is almost the same as the standard normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$ due to the 3-σ principle, resulting in a negligible training-evaluation gap. Similarly, for the first term $\log p_{\boldsymbol{\epsilon}}(\hat{\mathbf{x}}_{\boldsymbol{\epsilon}})$, which basically same as $\log p_{\boldsymbol{\epsilon}}(\mathbf{z}_0)$, we can use the Eq. (12).

A.1.3 Likelihood computation

Computing the trace of Jacobian of the drift function, $\operatorname{tr}(\nabla_{\mathbf{z}_t}\mathbf{h}_{\boldsymbol{\theta}}(\mathbf{z}_t, \gamma_t))$ in Eq. (12), is expensive, and in practice, it can be estimated by Skilling-Hutchinson trace estimator (Skilling, 1989; Hutchinson, 1989). We follow the previous studies (Zheng et al., 2023b; Chen et al., 2018; Sahoo et al., 2023) to compute this term.

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

where the random variable $\boldsymbol{\epsilon}$ satisfies $\mathbb{E}_{p(\boldsymbol{\epsilon})}[\boldsymbol{\epsilon}] = \mathbf{0}$ and $\operatorname{Cov}_{p(\boldsymbol{\epsilon})}[\boldsymbol{\epsilon}] = \mathbf{I}$. Common choices for $p(\boldsymbol{\epsilon})$ include Rademacher or Gaussian distributions. Notably, the term $\operatorname{tr}(\nabla_{\mathbf{z}_t}\mathbf{h}_{\boldsymbol{\theta}}(\mathbf{z}_t, t))\boldsymbol{\epsilon}$ can be computed efficiently using "Jacobian-vector-product" computation in Deep Learning frameworks. We use Rademacher distribution for computing $p(\boldsymbol{\epsilon})$, and following the default setting in prior works Sahoo et al. (2023); Song et al. (2020b); Zheng et al. (2023b), we use RK45 ODE solver (Dormand & Prince, 1980) with `atol=1e-5` and `rtol=1e-5` to compute the integral in Eq. (12) using `scipy.integrate.solve_ivp`.

B Implementation Details

B.1 Training details

CIFAR-10 Dataset We used the available checkpoint of EDM model for CIFAR-10 (Krizhevsky & Hinton, 2009) dataset [3] in our experiments. We trained the PyTorch re-implementation of VDM [4] following the same architecture in (Kingma et al., 2021), for 10 million updates with no data-augmentation, fixed-linear γ, and batch size of 128. Our trained VDM model reached 2.64 BPD on test-set using exact likelihood and 2.66 based on VLB evaluation (their reported number was 2.65 based on VLB). We used EMA(Exponential Moving Average) version of the model for our experiments.

ImageNet32 Dataset Studies use different versions of ImageNet32 dataset (Deng et al., 2009). We pre-processed the tensorflow-records provided from i-DODE (Zheng et al., 2023b), and save

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

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

the dataset as PNG files in two *train* and *val* folders. We trained the VDM model similar to the Cifar-10 dataset, except with double the number of channels to 256 with a total batch size of 512, following (Kingma et al., 2021). We trained it for 2 million steps (similar to them). Since there were no checkpoints for EDM model on ImageNet32, we followed the same setting of CIFAR-10 on EDM repo with parameters `--cond 0 --arch ddpmpp` with the duration of 1000M images `--duration 1000`. We did not tune hyperparameters, and the obtained model is sub-optimal.

For measuring the FID we followed EDM (Karras et al., 2022) and computed the reference statistics of *ImageNet32* as `.npz` file, and used it for the FID calculation of our experiments. Similar to CIFAR-10 experiments, we used EMA(Exponential Moving Average) version of the model for our experiments.

B.2 Model details

The VDM (Kingma et al., 2021) is a *noise-prediction* model where $\mathbf{z}_t = \alpha_t \cdot \mathbf{x} + \sigma_t \cdot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, while EDM (Karras et al., 2022) is an *image-denoising* model where $\mathbf{z}_t = \mathbf{x} + \sigma_{edm} \cdot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. In order to make our *Merged Model* work, we have to apply correct mapping before going from one model to another. Based on the above formula, the conversion of an image $\mathbf{z}_t$ in latent space of VDM needs the rescaling as defined here:

$$\begin{aligned} \mathbf{z}_t &= \alpha_t \cdot \mathbf{x} + \sigma_t \cdot \boldsymbol{\epsilon} \qquad \text{(Divide by } \alpha_t\text{)} \\ \Rightarrow \frac{\mathbf{z}_t}{\alpha_t} &= \mathbf{x} + \frac{\sigma_t}{\alpha_t} \cdot \boldsymbol{\epsilon} \quad \text{(Input format to EDM,} \quad \sigma_{edm} = \frac{\sigma_t}{\alpha_t}\text{)} \end{aligned} \tag{17}$$

The python code we used for computing $p(\mathbf{z}_s|\mathbf{z}_t, \mathbf{x})$ in VDM ancestral sampling, which illustrates this conversion (the number in the comment indicates the equation number in (Kingma et al., 2021) paper):

```
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
```

In EDM (Karras et al., 2022), we have $t_{min} = 0.002$, and end time $t_{max} = 80.0$. In terms of $\gamma_t = -\log SNR_t = -\log \frac{\alpha_t^2}{\sigma_t^2}$, the EDM is trained on $\gamma_{EDM} \in [-12.43, 8.764]$ based on 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

C ADDITIONAL RESULTS AND VISUALIZATIONS

C.1 EXTRA EVALUATIONS OF OUR METHOD

Table 5 are the reported values for the likelihood evaluation of our *Merged Model* using the Variational Lower Bound (VLB). We used a batch size of 512 and ran each experiment 10 times, then reported the mean and standard deviation of the results.

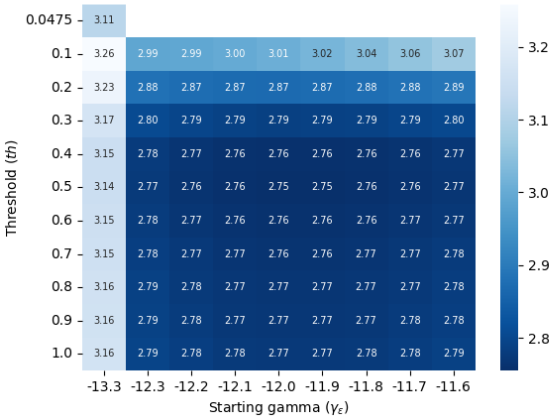

Figure 4: Evaluation with Uniform dequantization on CIFAR-10

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

Threshold	**CIFAR10**		**ImageNet32**	
	Mean(↓)	Std	Mean(↓)	Std
0.0475	3.12	0.006	3.98	0.004
0.1	2.89	0.005	3.84	0.002
0.2	2.75	0.008	3.79	0.003
0.3	2.67	0.007	3.76	0.004
0.4	**2.64**	0.005	3.74	0.004
0.5	**2.64**	0.008	3.73	0.003
0.6	2.65	0.007	3.73	0.004
0.7	2.65	0.008	3.73	0.005
0.8	2.65	0.004	3.73	0.004
0.9	2.65	0.008	3.73	0.001
1.0	2.65	0.007	**3.73**	0.005

The evaluation using the Uniform dequantization is depicted in Fig. 4. Following (Zheng et al., 2023b), we tuned the lower bound of integral γ_0 using different values $\{-13.3, -12.3, -12.2, -12.1, -12.0, -11.9, -11.8, -11.7, -11.6\}$ while keeping the upper bound of integral $\gamma_1 = 5$. Using the $g_0 \in \{-12.0, -11.9\}$ with the threshold time step $\eta = 0.5$ gives us the best results 2.75.

Additionally, the Varitaion Lower Bound (VLB) evaluation of our proposed method in the table Table 5 along with FID using VDM Ancestral Algorithm 2 is depicted in Figures 5–6 on a log-10 scale. The ODE evaluation with Truncated-Normal dequantization and ODE sampler on ImageNet32 is shown in Fig. 7.

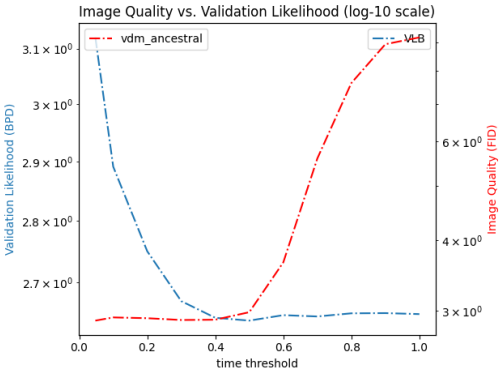

Figure 5: Model performance using VLB and VDM Ancestral on CIFAR-10.

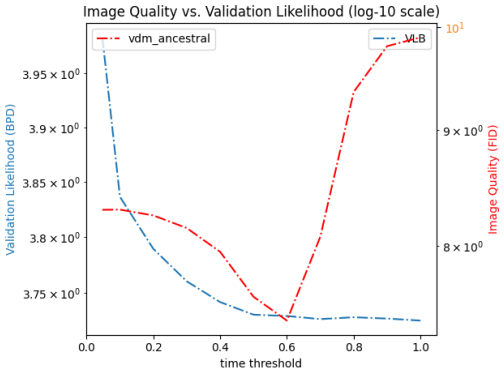

Figure 6: Model performance using VLB and VDM Ancestral on Imagenet32

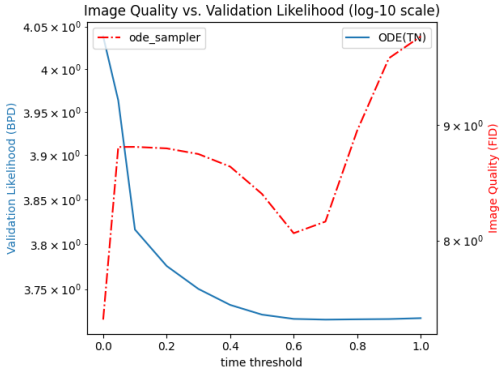

Figure 7: Model performance using ODE(TN) and ODE sampler on ImageNet32.

C.2 Datasets and Visualizations

In this section, we illustrated, given a fixed seed, how the images look using different thresholds. The samples with such setting for ImageNet32 dataset is given in Fig. 8

Figure 8: Visualization of generated images using different thresholds η on ImageNet32 dataset

C.3 Full Comparison Table

Table 6 is the extended version of Table 3, and it encompasses other methods in 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	-	-	-	-
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	-	-	-	-
Reflected Diffusion Models (Lou & Ermon, 2023)	2.68	2.72	-	3.74	-	-
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$^{\vee}$	9.38	206	3.72*/3.72$^{*\vee}$	9.85*	158
EDM (our evaluation, $\gamma \in$[-12.43, 8.764]) (Karras et al., 2022)	3.21	2.85	127	4.04*	7.38*	120
Ours NLL ($\eta = 0.4$, CIFAR10)	**2.62**	2.86	149	-	-	-
Ours FID ($\eta = 0.2$, CIFAR10)	2.73	**2.84**	137	-	-	-
Ours FID+NLL ($\eta = 0.6$, ImageNet32)	-	-	-	**3.72***	8.06*	153