MixFlow 2: One-Stage MixFlow Training for Flow Matching

Anonymous authors
Paper under double-blind review

Abstract

MixFlow reduces the training–testing discrepancy by training the velocity prediction network at a training timestep t using noisy teacher as training noisy data that correspond to training timestep t and slowed timesteps (with higher noise). It is different from standard training: training noisy data is only from the teacher noisy data at the training timestep t. The original MixFlow approach adopts a two-stage training strategy: standard training first learns accurate instantaneous velocity prediction, and MixFlow post-training then uses noisy teacher forcing to handle the Slow Flow problem in multi-step discrete sampling.

Direct application of original implementation to one-stage training, i.e., training from scratch, is not optimal. The reason is that training noisy data sampling is more from noisy teachers, and more capable of handling the Slow Flow problem, and less capable of learning instantaneous velocity prediction which is addressed by standard training with teacher forcing. We present a one-stage training approach, MixFlow 2, with the introduction of a novel sampling mechanism: the slowed timestep for forming the training noisy data is sampled from a symmetric triangular distribution, where intermediate slow timesteps are sampled more frequently. In addition, we observe that the prediction at high-noise training timesteps is less reliable, as high-noise timestep t is less-frequently sampled. This influences guided sampling: guidance at high-noise timesteps is harmful. So we adopt guidance interval to remove the guidance at high-noise timesteps, maintaining the generation quality achieved without guidance. Empirical results show that MixFlow 2 improves the class-conditional and text-to-image generation performance and outperforms original MixFlow.

1 Introduction

Diffusion and flow matching models have become the dominant approach to visual generation (Ho et al., 2020; Lipman et al., 2022; Ma et al., 2024). A fundamental challenge in training diffusion and flow matching models is the training–testing discrepancy: the network learns on clean ground-truth data, but at sampling it depends on its own generated noisy data. Existing methods perturb inputs during training (Ning et al., 2023) or redesign noise schedules (Ning et al., 2024; Li et al., 2024a).

The recently developed training approach, MixFlow (Li et al., 2026), makes a simple modification to standard training. In standard flow matching training, the input to the velocity prediction network includes the training noisy data $\mathbf{x}$, and the training timestep t. Standard training adopts teacher forcing: form the training noisy data $\mathbf{x}$ using the ground-truth noisy data $\mathbf{x}_t$ at the training timestep t. This results in efficient instantaneous velocity prediction, but is not effective for multi-step discrete sampling due to the training–testing discrepancy. MixFlow adopts *noisy teacher forcing*: form the training noisy data not just from the training timestep, but additionally takes the noisy data $\mathbf{x}_{m_t}$ from slowed timesteps m_t, i.e., higher noise timesteps. This simulates the sampling process, where the generated noisy data is usually slower (Li et al., 2026). Figure 1 illustrates the difference between standard training and MixFlow training.

The initial MixFlow implementation adopts a two-stage strategy: the first stage is standard training for learning accurate instantaneous velocity at each training timestep; the second stage is training with MixFlow setting for handling the Slow Flow problem. The sampling mechanism for training noisy data and timesteps, is designed mainly for handling the Slow Flow problem with the assumption that instantaneous velocity

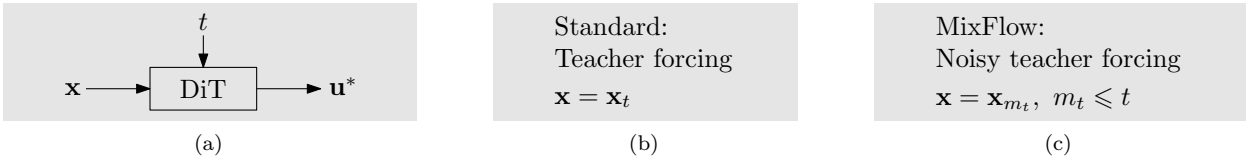


Figure 1: (a) The inputs to Diffusion Transformer (DiT) include: the training noisy data $\mathbf{x}$ and the training timestep t. (b) Standard training uses teacher forcing: the training noisy data is from the training timestep t. (c) MixFlow Training adopts noisy teacher forcing: the training noisy data is from the higher-noise timestep $m_t \leqslant t$.

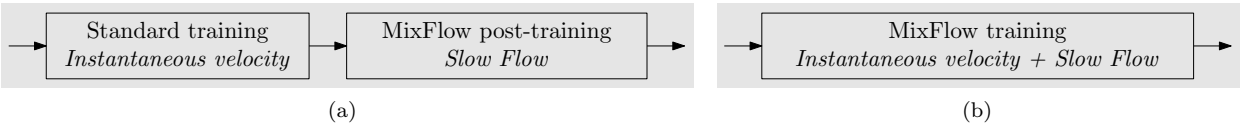


Figure 2: (a) Original MixFlow (Li et al., 2026). Two-stage training: first learn accurate instantaneous velocity with teacher forcing and then handle the Slow Flow issue with noisy teacher forcing. (b) MixFlow 2. One-stage training: simultaneously learn accurate instantaneous velocity and handle the Slow Flow issue with noisy teacher forcing.

is already learned. Consequently, it is not optimal to directly apply the sampling mechanism to train the diffusion model from scratch.

We present a one-stage training approach, named MixFlow 2, with a modified sampling mechanism: the slowed timestep for forming the training noisy data follows a symmetric triangular distribution. Intuitively, the noisy data at intermediate timesteps are more frequently sampled. In contrast, the slowed timestep in MixFlow (Li et al., 2026) follows a noise-skewed triangular distribution, with the mode close to the noise data, meaning that the higher noisy data are sampled more often for training the velocity network. It is empirically shown that symmetric triangular sampling outperforms noise-skewed triangular sampling for one-stage training, implying that such sampling scheme is capable of simultaneously learning instantaneous velocities as well as handling the Slow Flow issue. Figure 2 summarizes two-stage training in Li et al. (2026) and one-stage training in the proposed approach MixFlow 2.

On the other hand, the training timestep in MixFlow follows a clean-skewed triangular distribution (i.e., a $\mathrm{Beta}(2, 1)$ distribution) with the mode close to the clean data. Intuitively, the training timesteps at higher noise are sampled less often. It implies that velocity prediction network at training timesteps with higher noise is insufficiently trained and accordingly the velocity prediction at higher noise timesteps is not reliable. This unreliability is harmful for guided sampling as the unreliable prediction at higher noise timestep is scaled. Thus, we adopt guidance interval (Kynkäänniemi et al., 2024) to avoid scaling the prediction at higher-noise timesteps, mitigating the influence of unreliable predictions.

Empirical results show that MixFlow 2 improves the class-conditional and text-to-image generation performance. In particular, MixFlow 2 combined with RAE (Zheng et al., 2025) reaches the gFID score of 1.06 with guidance interval for class-conditional image generation. The results also show that MixFlow 2, one-stage training is superior over initial MixFlow, two-stage training. The gain from MixFlow 2 is consistently significant for text-to-image generation.

Our contributions are summarized as follows:

- We propose MixFlow 2, a one-stage training approach. It adjusts the training noisy data sampling scheme so that MixFlow 2 is capable of learning both instantaneous velocity, and noisy teacher forcing that aims to handle the Slow Flow issue.

- We observe that the Beta$(2, 1)$ training timestep distribution leaves the predictions at high-noise timesteps less trained, making classifier-free guidance harmful at high-noise timesteps. We adopt guidance interval to retain the performance achieved in sampling without guidance.
- MixFlow 2 consistently improves generation quality across class-conditional and text-to-image generation. MixFlow 2, combined with RAE reaches a gFID of 1.06 on ImageNet 256×256.

2 Related Work

Diffusion and flow matching. Denoising diffusion models (Ho et al., 2020; Song et al., 2021a;b; Karras et al., 2022) reverse a Gaussian noising SDE by learning the score of the perturbed data distribution. Flow matching (Lipman et al., 2022; Liu et al., 2023; Dao et al., 2023) is a simulation-free approach for training continuous normalizing flows (Papamakarios et al., 2021), based on regressing vector fields of fixed conditional probability paths between noise and data. Diffusion Transformers, including DiT (Peebles & Xie, 2023), SiT (Ma et al., 2024), and DiffiT (Hatamizadeh et al., 2024), have become the standard architectures for class-conditional and text-to-image generation. Stable Diffusion 3 (SD3) (Esser et al., 2024), FLUX (Labs, 2024; Labs et al., 2025), and Z-Image (Z-Image Team et al., 2025) further scale flow matching to large text-to-image models. Our work follows the flow matching formulation and uses improved SiT (Ma et al., 2024; Yao et al., 2025) as the primary backbone.

Training timestep sampling and noise schedules. The training timestep distribution strongly affects the quality of diffusion training (Karras et al., 2022; Esser et al., 2024; Hoogeboom et al., 2023; Kingma et al., 2021). EDM (Karras et al., 2022) analyzes how the noise level distribution shapes the loss landscape and proposes a log-normal noise schedule. SD3 allocates more training to intermediate timesteps through a logit-normal training-timestep distribution, which yields better generation quality than uniform sampling. SD3 and Simple Diffusion (Hoogeboom et al., 2023) further shift the training timestep distribution according to image resolution. FasterDiT (Yao et al., 2024) selects the training timestep distribution to align signal-to-noise ratios across resolutions. Our work also focuses on the slowed timestep m_t that decouples the timestep used to form the noisy data and the training timestep used as the network input.

Training testing discrepancy in diffusion training. Standard diffusion training feeds the network with clean ground-truth noisy data, while at sampling time the network sees its own noisy outputs from previous steps. This training–testing discrepancy is a well-known issue in autoregressive sequence models (Ranzato et al., 2015; Bengio et al., 2015; Schmidt, 2019) and has been studied in diffusion under several formulations. Input perturbation (Ning et al., 2023) inject extra perturbation into the input noisy data to mimic the discrepancy at training time. Ning et al. (2024); Li et al. (2024a) study the source of this gap and propose noise schedules or weighting schemes to alleviate it. MixFlow (Li et al., 2026) differs by feeding the network the noisy data at a slowed timestep m_t together with training timestep t.

3 Method

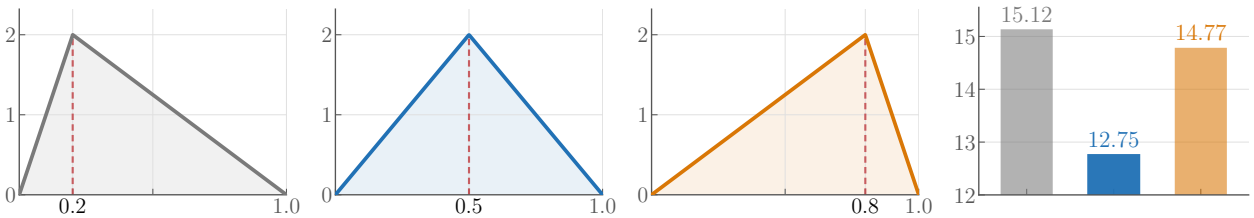
3.1 Preliminaries

Flow matching. Flow matching models (Lipman et al., 2022; Liu et al., 2023) learn a velocity field that transports the noise to the data. Usually, the noise distribution is a Gaussian distribution, $\mathbf{x}_0 \sim \mathcal{N}(0, 1)$, and the data distribution, $\mathbf{x}_1 \sim q$, is represented by data samples. The interpolation, ground-truth noisy data along the path, is a combination of noise and data:

$$\mathbf{x}_t = t\mathbf{x}_1 + (1 - t)\mathbf{x}_0. \tag{1}$$

The velocity prediction network is denoted by $\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}, t)$, where $\boldsymbol{\theta}$ is the network parameters. The network is optimized by minimizing the following loss function,

$$\mathbb{E}[\|\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}, t) - \mathbf{u}^*\|_2^2]. \tag{2}$$



(a) Noise-skewed triangular (b) Symmetric triangular (c) Clean-skewed triangular (d) gFID score

Figure 3: **Illustrating the marginal distributions for sampling the training noisy data and the impact on the generation performance.** (a) (b) (c) Marginal distributions for sampling the slowed timestep m_t: noise-skewed, symmetric, and clean-skewed triangular distributions. (d) gFID scores for the same three distributions, in the order of (a)–(c). The symmetric triangular sampling performs the best.

Here, the training noisy data $\mathbf{x}$ is from the training timestep t:

$$\mathbf{x} = \mathbf{x}_t. \tag{3}$$

$\mathbf{u}^*$ is the ground truth velocity: $\mathbf{u}^* = \mathbf{x}_1 - \mathbf{x}_0$.

The training timestep t is sampled from a distribution $p(t)$ over $[0, 1]$, e.g., the uniform distribution or the logit-normal distribution (Esser et al., 2024). The expectation is conducted over $\{t, \mathbf{x}_0, \mathbf{x}_1\}$ in Equation 2.

MixFlow. MixFlow (Li et al., 2026) learns the velocity prediction network $\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}, t)$ at training timestep t with the training noisy data at the slowed timestep m_t, including both the teacher noisy data at t and noisy teacher noisy data at the higher-noise timesteps. The initial approach is a two-stage approach: first train in the standard manner for accurate instantaneous velocity, and then train in the MixFlow manner by sampling more training noisy data from noisy teachers, to handle the Slow Flow issue. The Slow Flow issue is: the ground-truth interpolation that is the nearest to the generated noisy data at a given sampling timestep is observed to correspond to a higher-noise timestep (slowed timestep). MixFlow is intuitively a simple simulation of noisy data generated in multi-step discrete sampling.

Different from standard training, the training noisy data in Equation 2 is from a slowed timestep m_t:

$$\mathbf{x} = \mathbf{x}_{m_t}. \tag{4}$$

The training timestep and the slowed timestep are sampled as follows. The training timestep t follows a Beta distribution (positively skewed triangular distribution, which we also call it as clean-skewed triangular distribution),

$$t \sim \text{Beta}(2, 1), \text{ i.e., } p(t) = 2t, t \in [0, 1]. \tag{5}$$

Slowed timestep m_t, conditioned on the training timestep t, follows a uniform distribution over a range of slowed timesteps $[(1-\gamma)t, t]$:

$$m_t|t \sim \mathcal{U}[(1-\gamma)t, t], \text{ i.e., } p(m_t|t) = \frac{1}{\gamma t}, m_t \in [(1-\gamma)t, t], \tag{6}$$

where γ is the slowness strength parameter.

The MixFlow loss function is written as follows,

$$\mathbb{E}_{t,\mathbf{x}_0,\mathbf{x}_1,m_t}[\|\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_{m_t}, t) - \mathbf{u}^*\|_2^2]. \tag{7}$$

3.2 MixFlow 2

Training noisy data sampling adjusted for efficient training. The original MixFlow implementation post-trains a instantaneous velocity prediction network pretrained by standard training. The task is mainly

on solving the Slow Flow problem, i.e., handling the case the generated noisy data is slower than the current sampling time in the pretrained model (Li et al., 2026). It is not optimal to directly apply the original MixFlow implementation to one-stage training which needs to handle instantaneous velocity prediction. This is because the noisy data (i.e., slowed timestep) is sampled from a noise-skewed triangular marginal distribution (derived below), so the intermediate timesteps are less exploited.

We analyze that the distribution of the training noisy data, i.e., the marginal distribution of the slowed timesteps in the original MixFlow training is a noise-skewed triangular distribution. The joint distribution $p(t, m_t)$ can be derived from the marginal distribution $p(t)$ in Equation 5 and the conditional distribution $p(m_t|t)$ in Equation 6:

$$p(t, m_t) = \frac{2}{\gamma}, t \in [0, 1], m_t \in [(1-\gamma)t, t]. \tag{8}$$

Here, the range of m_t depends on t. Integrating out t in the joint distribution $p(t, m_t)$ leads to the following marginal distribution:

$$p(m_t) = \begin{cases} \frac{2}{1-\gamma}\, m_t, & m_t \leqslant 1-\gamma, \\ \frac{2}{\gamma}\,(1-m_t), & m_t \geqslant 1-\gamma. \end{cases} \tag{9}$$

The distribution $p(m_t)$ is a triangular distribution over $[0, 1]$ with the mode $1-\gamma$: $\texttt{Triangular}(0, 1-\gamma, 1)$. In the original MixFlow implementation, $\gamma = 0.8$ and the marginal distribution is a noise-skewed triangular distribution. The distribution is illustrated in Figure 3(a). This noise-skewed triangular distribution has higher probability on higher-noise data, and is more capable for handling the Slow Flow issue.

In one stage training, we also need to handle instantaneous velocity prediction. Fortunately, it is easy to adjust the marginal distribution $p(m_t)$ so that the intermediate timesteps are more frequently sampled. We therefore set $\gamma = 0.5$, which makes $p(m_t)$ a symmetric triangular distribution, as illustrated in Figure 3(b),

$$p(m_t) = \texttt{Triangular}(0, 0.5, 1). \tag{10}$$

Figure 3(d) shows that symmetric triangular sampling performs the best, implying it is helpful for one-stage MixFlow training with noisy teacher forcing that also needs to more frequently sample intermediate noisy data that is more informative for velocity prediction.

On the other hand, $\gamma = 0.5$ means that for a training timestep t, a half of higher-noise data, from $\mathbf{x}_{t/2}$ to $\mathbf{x}_t$ is used for training. The other half, from $\mathbf{x}_0$ to $\mathbf{x}_{t/2}$, is not included for training, and the resulting model might be less capable of handing over-slowed prediction which is the input of the subsequent prediction during sampling, but is more capable of predicting the instantaneous velocity. Such a choice leads to a good balance between instantaneous velocity prediction and Slow Flow elimination, which is critical in one-stage MixFlow training.

Training timestep distribution leading to guided sampling with interval. The marginal distribution for training timestep is a clean-skewed triangular distribution, illustrated in Figure 3(c). This distribution indicates that higher-noise training timesteps are less sampled during training and thus the prediction at higher-noise timesteps is not reliable due to insufficient training.

This influences the sampling process, in particular when classifier-free guidance (CFG) is applied. CFG works by combining conditioned and unconditioned predictions at each sampling step:

$$\hat{\mathbf{u}}(\mathbf{x}_t, t) = \mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_t, t, \emptyset) + w(\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_t, t, c) - \mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_t, t, \emptyset)) \tag{11}$$

$$= w\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_t, t, c) + (1-w)\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_t, t, \emptyset). \tag{12}$$

Here, $\mathbf{x}_t$ means the prediction at sampling timestep t, ($\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_t, t, c)$ is conditioned prediction, $\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_t, t, \emptyset)$ is unconditioned prediction. w is the guidance scale, typically $w > 1$, which amplifies the unreliability of the prediction. This issue becomes more pronounced as the model size increases. To avoid such amplification, we adopt the guidance interval technique (Kynkäänniemi et al., 2024): only apply the guidance in an interval corresponding to lower noise, e.g., $t \in [t_h, 1]$ in our implementation,

$$\hat{\mathbf{u}}(\mathbf{x}_t, t) = w(t)\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_t, t, c) + (1-w(t))\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_t, t, \emptyset), \tag{13}$$

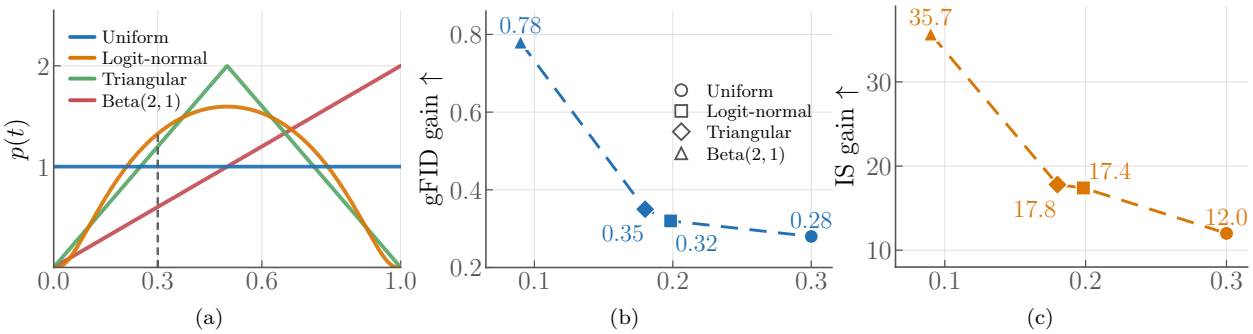


Figure 4: **Sampling high-noise timesteps less frequently benefits more from guidance interval.** The horizontal axis corresponds to the probability of sampling the interval $[0, 0.3]$, and the vertical axis corresponds to the gFID gain (b) and the IS gain (c). (a) depicts four probabilities for sampling training timesteps.

Algorithm 1 MixFlow 2: One-stage training

Input: dataset $\mathcal{D}$, initial model parameter $\boldsymbol{\theta}$, learning rate η
repeat
  Sample $\mathbf{x}_1 \sim \mathcal{D}$
  Sample $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  Sample $t \sim \text{Beta}(2, 1)$
  Sample $m_t \mid t \sim \mathcal{U}[0.5t, t]$
  $\mathbf{x}_{m_t} \leftarrow m_t \mathbf{x}_1 + (1 - m_t)\mathbf{x}_0$
  $\mathbf{u}^* \leftarrow \mathbf{x}_1 - \mathbf{x}_0$
  $\mathcal{L} \leftarrow \|\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_{m_t}, t) - \mathbf{u}^*\|_2^2$
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}$
until convergence

where $w(t) = 1$ for high-noise timesteps t, $t \in [0, t_h]$, and $w(t) = w$ for other sampling timesteps.

Figure 4 (b)(c) show that the generation performance benefits more from guidance interval when higher-noise timesteps are less frequently sampled for training. We study four training timestep sampling schemes in standard training: uniform, logit-normal, symmetric triangular, and Beta$(2, 1)$ (clean-skewed triangular), shown in Figure 4 (a). The sampling probabilities in high-noise timestep interval $[0, 0.3]$ are 0.30, 0.19, 0.18, 0.09, respectively. We report the gFID gain between sampling with limited guidance interval and with normal CFG. The results in Figure 4 clearly show that the benefit from guidance interval is larger for lower sampling probabilities in the high-noise timestep interval. More results are given in Section 4.2.

4 Experiments

4.1 Setup

We report results for class-conditional generation on ImageNet. We use SD-VAE (Rombach et al., 2022) to encode the image into the latent representation and decode the representation into the image. We use SiT (Ma et al., 2024) with improved settings from Lightning-DiT (Yao et al., 2025): RMSNorm (Zhang & Sennrich, 2019), RoPE (Su et al., 2024), `torch.compile` (Ansel et al., 2024), a batch size of 1024, and AdamW (Loshchilov & Hutter, 2017) with $\beta_2 = 0.95$. We train the models for 800 epochs and conduct the sampling process with the Heun solver (Karras et al., 2022) using 250 steps. We also validate our approach by applying MixFlow 2 to RAE (Zheng et al., 2025). We evaluate the performance in terms of gFID (Heusel et al., 2017) and Inception Score (Salimans et al., 2016).

Table 1: **Symmetric triangular distribution performs better for training noisy data sampling than clean-skewed and noise-skewed triangular distributions for MixFlow 2.** The scores are from no guidance.

Method	SiT-S/2		SiT-B/2		SiT-L/2		SiT-XL/2	
	gFID ↓	IS ↑	gFID ↓	IS ↑	gFID ↓	IS ↑	gFID ↓	IS ↑
Clean-skewed	32.12	40.1	14.77	87.1	8.10	129.9	7.03	136.1
Symmetric	**28.43**	**54.7**	**12.75**	**101.0**	**7.29**	**144.8**	**6.49**	**158.5**
Noise-skewed	34.24	43.2	15.12	94.7	8.23	130.4	7.12	133.3

4.2 Ablation Studies

Training noisy data from symmetric triangular sampling is beneficial. Table 1 reports the class-conditional generation performance for three triangular sampling schemes: noise-skewed, symmetric, and clean-skewed. The noise-skewed triangular sampling is advantageous in the original implementation of MixFlow post-training as instantaneous velocity prediction is already learned in pretraining and the post-training stage only needs to handle the Slow Flow issue. The results validate that symmetric triangular sampling performs the best. This suggests that MixFlow 2 benefits from more frequently sampling intermediate noisy data that provide more valuable information for velocity prediction. Such sampling is also helpful for standard training (Esser et al., 2024), and the results with symmetric sampling for standard training are provided in Appendix B

MixFlow 2 handles instantaneous velocity prediction as well as the Slow Flow issue. Figure 5 shows how the gFID score and the IS change with training steps for MixFlow 2 and standard training, both with symmetric triangular sampling. One can see that at the early training steps, standard training performs better, and at the later training steps, MixFlow 2 performs better. The comparison at the early training steps is mainly because (1) standard training only learns instantaneous velocity prediction, which is easier than the multi-task learning problem in MixFlow 2, where the model must additionally handle the Slow Flow issue with noisy teacher forcing; and (2) the benefit from the additional noisy teacher forcing task for discretized sampling is weak when velocity prediction is not very accurate. The comparison at the later training steps shows that, when velocity prediction becomes more accurate and the model better handles the Slow Flow issue, noisy teacher forcing provides more benefit for multi-step discretized sampling than instantaneous velocity alone.

Separation of training noisy data and training timestep sampling is beneficial. We compare MixFlow 2 to the cases that the noisy data is from the training timestep, i.e., standard training, with two training timestep sampling distributions: symmetric triangular and $\text{Beta}(2, 1)$. Table 2 shows the results. One can see that (1) standard training without separated sampling performs worse than MixFlow 2; (2) symmetric triangular sampling for standard training works well but still is inferior to MixFlow 2; and (3) $\text{Beta}(2, 1)$ for standard training performs the worst. In other words, applying symmetric triangular sampling or $\text{Beta}(2, 1)$ sampling, for that the training noisy data is from the training timestep, does not lead to satisfactory performance. The comparison implies that separated sampling for training noisy data and training timestep is beneficial.

Guidance interval is effective for handling insufficient training at high-noise training timesteps. We report the results for standard training with uniform training timestep sampling, logit-normal training timestep sampling, and MixFlow 2, to show that guidance interval is effective for sampling. The results are given in Table 3. There are several key observations. Without guidance, MixFlow 2 consistently performs much better than standard training. With CFG, MixFlow 2 still performs better, but the superiority becomes minor for larger models. With guidance interval, all the models get improved performance, and MixFlow 2 gets more improvement. The results are consistent to the analysis in Section 3.2: MixFlow 2 samples high-noise timesteps for training less frequently, the prediction at high-noise timesteps is less reliable, and thus benefits more from guidance interval.

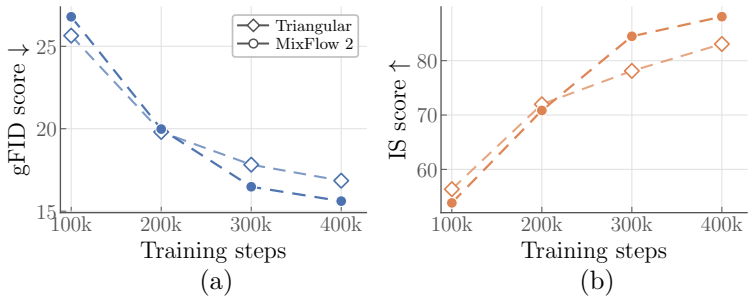


Figure 5: **MixFlow 2 performs worse than standard training with triangular sampling in early training, and better at the later training steps.** In early training, standard training learns only the instantaneous velocity prediction task, while MixFlow 2 must also learn noisy teacher forcing for slowed timesteps. At later training steps, the velocity predictor becomes more accurate, and the additional noisy teacher forcing signal improves multi-step sampling, leading to lower gFID (a) and higher IS (b).

Method	gFID ↓	IS ↑
No separation		
Uniform	16.37	82.6
Symmetric triangular	14.54	93.1
Beta(2, 1)	18.48	72.9
Separation		
MixFlow 2	**12.75**	**101.0**

Table 2: **Separating training noisy data sampling from training timestep sampling is beneficial.** MixFlow 2: separation, different sampling for training noisy data and training timestep. Standard training: no separation, training noisy data from the training timestep.

Table 3: **Guidance interval is more helpful for MixFlow 2.** MixFlow 2 samples high-noise timesteps for training less frequently than standard training with logit-normal sampling and uniform sampling.

Model	Method	W/o guidance		Standard CFG		Guidance interval		Gain	
		gFID ↓	IS ↑	gFID ↓	IS ↑	gFID ↓	IS ↑	ΔgFID ↓	ΔIS ↑
SiT-S/2	Standard FM (uniform)	35.78	40.1	14.70	97.3	11.72	118.4	-2.98	+21.1
	Standard FM (logit-normal)	31.92	49.0	11.90	109.1	8.75	136.9	-3.15	+27.8
	MixFlow 2	**28.43**	**54.7**	**10.43**	**120.5**	**6.87**	**153.1**	**-3.56**	**+32.6**
SiT-B/2	Standard FM (uniform)	16.37	82.6	4.16	188.4	3.88	200.4	-0.28	+12.0
	Standard FM (logit-normal)	14.79	90.5	3.45	202.7	3.13	216.6	-0.32	+13.9
	MixFlow 2	**12.75**	**101.0**	**3.21**	**218.9**	**2.62**	**234.2**	**-0.59**	**+15.3**
SiT-L/2	Standard FM (uniform)	9.73	130.8	2.20	267.9	1.98	273.5	-0.22	+5.6
	Standard FM (logit-normal)	8.25	135.1	**2.03**	272.3	1.77	280.3	-0.26	+8.0
	MixFlow 2	**7.29**	**144.8**	2.10	**277.9**	**1.60**	**287.2**	**-0.50**	**+9.3**
SiT-XL/2	Standard FM (uniform)	8.76	138.3	2.04	277.7	1.82	279.1	-0.22	+1.4
	Standard FM (logit-normal)	7.72	144.7	**1.97**	281.4	1.69	286.5	-0.28	+5.1
	MixFlow 2	**6.49**	**158.5**	2.03	**289.1**	**1.58**	**295.1**	**-0.45**	**+6.0**

4.3 Comparison to Related Solutions

Table 4: **MixFlow 2 outperforms input perturbation.** The results are from no guidance.

Method	gFID ↓	IS ↑
Input perturbation (uniform)	15.95	84.45
Input perturbation (logit-normal)	14.73	92.6
MixFlow 2	**12.75**	**101.0**

Comparison with input perturbation. Input perturbation (Ning et al., 2023) trains the diffusion model by adding a Gaussian perturbation to the ground-truth noisy data to simulate inference-time prediction errors. The perturbed data may include ground-truth noisy data at lower-noise timesteps, which is shown to harm the training (Li et al., 2026). The two input-perturbation baselines use uniform and logit-normal timestep sampling, respectively. The results in Table 4 show that MixFlow 2 is superior to input perturbation.

Table 5: **MixFlow 2, one-stage training, with fewer training epochs, outperforms original MixFlow, post-training a pretrained model.** The results are from no guidance.

Model	Method	Pretraining epochs	Post-training epochs	Total epochs	gFID ↓	IS ↑
SiT-S/2	Standard FM (uniform)+ MixFlow	800	200	1000	29.65	50.9
	Standard FM (logit-normal) + MixFlow	800	200	1000	28.89	52.3
	MixFlow 2	800	–	800	**28.43**	**54.7**
SiT-B/2	Standard FM (uniform) + MixFlow	800	200	1000	12.94	97.7
	Standard FM (logit-normal) + MixFlow	800	200	1000	12.83	99.9
	MixFlow 2	800	–	800	**12.75**	**101.0**
SiT-L/2	Standard FM (uniform) + MixFlow	800	200	1000	7.40	139.9
	Standard FM (logit-normal) + MixFlow	800	200	1000	7.35	142.2
	MixFlow 2	800	–	800	**7.29**	**144.8**
SiT-XL/2	Standard FM (uniform) + MixFlow	800	200	1000	6.58	154.1
	Standard FM (logit-normal) + MixFlow	800	200	1000	6.54	156.8
	MixFlow 2	800	–	800	**6.49**	**158.5**

Comparison with the original MixFlow: pretraining vs posttraining. We compare MixFlow 2, one-stage training, to the original MixFlow, post-training a pretrained model (Li et al., 2026). For post-training, we first train a standard flow matching model for 800 epochs and then train the model with the original MixFlow for extra 200 epochs. We include two post-training baselines: one starts from standard flow matching with uniform timestep sampling, and the other starts from standard flow matching with logit-normal timestep sampling. The results in Table 5 show that MixFlow 2 with fewer training epochs performs slightly better than original MixFlow.

4.4 State-of-the-Art Class-Conditional Generation

We compare MixFlow 2 with state-of-the-art models on ImageNet 256×256 and 512×512. Following RAE (Zheng et al., 2025), we use autoguidance and class-balanced sampling for evaluation. The guidance scale is 1.6, and the guidance interval is $[0.05, 1.0]$ for both 256×256 and 512×512.

Table 6 and Table 7 report the results. MixFlow 2 + RAE achieves strong results among prior diffusion models in terms of gFID, achieving gFID scores of 1.06 with guidance at 256×256, and 1.08 with guidance at 512×512. The strong gFID scores are supported by qualitative samples that show fine-grained details and high visual quality. Visualization samples are provided in Figure 6. We provide additional visualization samples in Appendix C.

4.5 Text-to-Image Generation

We adopt the architecture of Z-Image (Z-Image Team et al., 2025): a Qwen-7B text encoder and a 5.5B-parameter DiT backbone. We use the data provided. The pretraining dataset is a combination of CC12M (Changpinyo et al., 2021), SA-1B (Kirillov et al., 2023), and JourneyDB (Sun et al., 2023): approximately 39.3M real images and 24.7M synthetic images generated by FLUX.1-schnell (Labs, 2024).

We follow ScaleRAE (Tong et al., 2026) to pretrain and finetune the model. The pretraining setting is as follows. The batch size is 2048, the number of training steps is 60k, and the learning rate is 2×10^{-5}. There is a 800 steps warmup with ratio 0.0134. We further finetune the model on the BLIP-3o 60k dataset (Chen et al., 2025) using the setting same as ScaleRAE.

We perform the evaluation on the GenEval (Ghosh et al., 2023) and DPG-Bench (Hu et al., 2024). We also report standard flow matching results with uniform and logit-normal training-timestep sampling.

Table 6: **Class-conditional generation on ImageNet** 256×256**.** MixFlow 2 + RAE reaches a gFID of **1.46** without guidance and **1.06** with guidance. The rows in gray are results with class-balanced sampling. The top-3 results per metric are in bold.

Method	W/o guidance		W/ guidance	
	gFID↓	IS↑	gFID↓	IS↑
Autoregressive				
VAR (Tian et al., 2024)	–	–	1.73	**350.2**
MAR (Li et al., 2024b)	2.35	227.8	1.55	303.7
xAR (Sucheng et al., 2025)	–	–	1.24	301.6
Diffusion				
ADM (Dhariwal & Nichol, 2021)	10.94	101.0	3.94	215.8
RIN (Jabri et al., 2023)	3.42	182.0	–	–
PixNerd (Wang et al., 2025a)	–	–	1.87	298.0
SiD2 (Hoogeboom et al., 2024)	–	–	1.38	–
DiT (Peebles & Xie, 2023)	9.62	121.5	2.27	278.2
MaskDiT (Zheng et al., 2023)	5.69	177.9	2.28	276.6
SiT (Ma et al., 2024)	8.61	131.7	2.06	270.3
MDTv2 (Gao et al., 2023)	–	–	1.58	**314.7**
REG (Wu et al., 2025)	1.80	230.8	1.36	299.4
REPA (Yu et al., 2025)	5.84	158.7	1.28	305.7
VA-VAE (Yao et al., 2025)	2.05	207.7	1.25	295.3
REPA-E (Leng et al., 2025)	1.69	219.3	1.12	302.9
DDT (Wang et al., 2025b)	6.27	154.7	1.26	**310.6**
RAE (Zheng et al., 2025)	1.51	**242.9**	1.13	262.6
MixFlow (Li et al., 2026)	**1.43**	**239.8**	**1.10**	259.7
LightningDiT-XL + IG (Zhou et al., 2025)	**1.24**	229.6	**1.07**	274.1
MixFlow 2 + RAE	**1.46**	**251.7**	**1.06**	264.0

Table 8 presents the results for pretraining and finetuning. On GenEval, MixFlow 2 raises the overall score from 0.47 to 0.56. On DPG-Bench, the overall score raises from 71.43 to 77.56. The results are better than ScaleRAE that adopts an RAE-based text-to-image system with a SigLIP-2 So400M representation encoder: 0.55 on GenEval. Our approach achieves consistent improvement for finetuning. After finetuning, MixFlow 2 raises the GenEval score from 0.75 to 0.82 and the DPG-Bench score from 73.10 to 78.76, showing that the improvement remains effective after finetuning. Visual results in Figure 7 show better object counting and spatial-relation alignment while preserving high image quality.

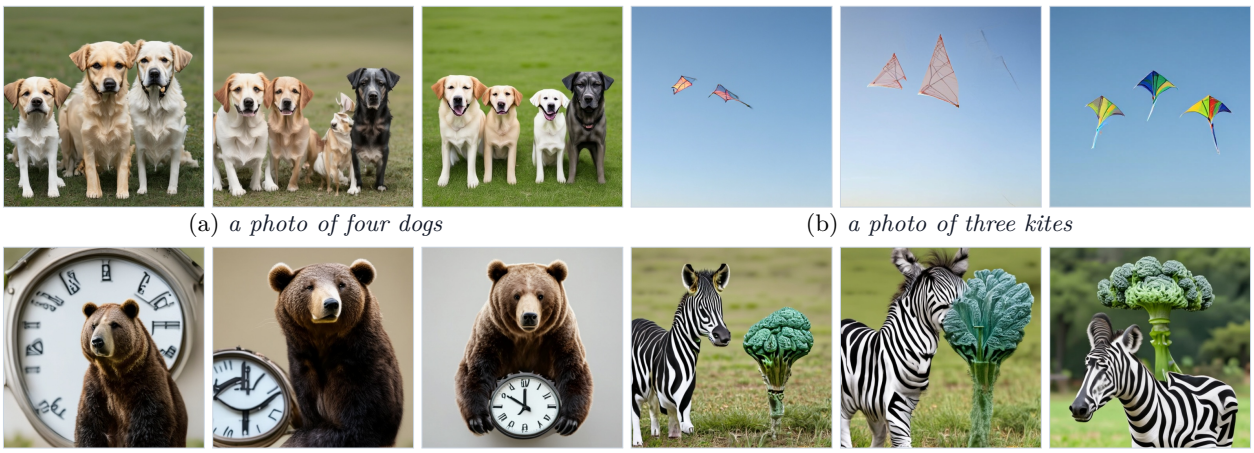
Figure 6: **Qualitative samples from MixFlow 2 + RAE trained at ImageNet** 256×256 **and** 512×512 **resolution with autoguidance.** The samples show fine-grained details, and high visual quality.

Table 7: **Class-conditional generation on ImageNet** 512×512**.** MixFlow 2 + RAE reaches a gFID of **1.55** without guidance and **1.08** with guidance. The rows in gray correspond to the results with class-balanced sampling. The top-3 results per metric are in bold.

Method	**W/o guidance**		**W/ guidance**	
	gFID↓	IS↑	gFID↓	IS↑
Autoregressive				
MAGViT-v2 (Yu et al., 2023)	3.07	213.1	1.91	**324.3**
VAR (Tian et al., 2024)	–	–	2.63	303.2
xAR (Sucheng et al., 2025)	–	–	1.70	281.5
Diffusion				
ADM (Dhariwal & Nichol, 2021)	–	–	3.85	221.7
SiD2 (Hoogeboom et al., 2024)	–	–	1.50	–
DiT (Peebles & Xie, 2023)	11.93	–	3.04	240.8
EDM2 (Karras et al., 2024)	1.91	–	1.25	–
SiT (Ma et al., 2024)	9.20	125.3	2.62	252.2
DiffiT (Hatamizadeh et al., 2024)	–	–	2.67	252.1
REPA (Yu et al., 2025)	–	–	2.08	274.6
REG (Wu et al., 2025)	–	–	1.68	**306.9**
DDT (Wang et al., 2025b)	–	–	1.28	**305.1**
RAE (Zheng et al., 2025)	**1.61**	**234.1**	**1.13**	259.6
MixFlow (Li et al., 2026)	**1.55**	**237.0**	**1.10**	256.9
MixFlow 2 + RAE	**1.55**	**243.4**	**1.08**	265.1

Table 8: **MixFlow 2 consistently improves the performance for text-to-image generation.**

Method	**Pretraining**		**Finetuning**	
	GenEval ↑	DPG-Bench ↑	GenEval ↑	DPG-Bench ↑
Standard FM (uniform)	0.47	71.43	0.75	73.10
Standard FM (logit-normal)	0.50	75.80	0.77	74.50
MixFlow 2	**0.56**	**77.56**	**0.82**	**78.76**



(a) *a photo of four dogs* (b) *a photo of three kites*

(c) *a photo of a bear above a clock* (d) *a photo of a zebra below a broccoli*

Figure 7: **Qualitative comparison examples.** Each group shows Standard FM (uniform), Standard FM (logit-normal), and MixFlow 2 from left to right. The examples are generated after finetuning, and MixFlow 2 shows better object counting and spatial-relation alignment while preserving high image quality.

5 Conclusion

We extend the original two-stage MixFlow training strategy, standard training and MixFlow post-training to one-stage training strategy, MixFlow 2. The key point lies in symmetric triangular sampling for training noisy data – noisy teachers, to trade-off two simultaneous tasks: instantaneous velocity learning and Slow Flow elimination. The other point is guided sampling with interval for handling insufficient training at high-noise timesteps. MixFlow 2 outperforms original two-stage MixFlow. The effectiveness is validated by class-conditional image generation and text-to-image generation.

References

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, et al. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, pp. 929–947, 2024.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Neural Information Processing Systems (NeurIPS)*, 28, 2015.

Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Jiuhai Chen, Zhiyang Xu, Xichen Pan, Yushi Hu, Can Qin, Tom Goldstein, Lifu Huang, Tianyi Zhou, Saining Xie, Silvio Savarese, et al. Blip3-o: A family of fully open unified multimodal models-architecture, training and dataset. *arXiv preprint arXiv:2505.09568*, 2025.

Quan Dao, Hao Phung, Binh Nguyen, and Anh Tran. Flow matching in latent space. *arXiv preprint arXiv:2307.08698*, 2023.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Neural Information Processing Systems (NeurIPS)*, 2021.

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

Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. Mdtv2: Masked diffusion transformer is a strong image synthesizer. *arXiv preprint arXiv:2303.14389*, 2023.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Neural Information Processing Systems (NeurIPS)*, 2023.

Ali Hatamizadeh, Jiaming Song, Guilin Liu, Jan Kautz, and Arash Vahdat. Diffit: Diffusion vision transformers for image generation. In *European Conference on Computer Vision (ECCV)*, 2024.

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. *Neural Information Processing Systems (NeurIPS)*, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Neural Information Processing Systems (NeurIPS)*, 2020.

Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *International Conference on Machine Learning (ICML)*, 2023.

Emiel Hoogeboom, Thomas Mensink, Jonathan Heek, Kay Lamerigts, Ruiqi Gao, and Tim Salimans. Simpler diffusion (sid2): 1.5 fid on imagenet512 with pixel-space diffusion. *arXiv preprint arXiv:2410.19324*, 2024.

Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. Ella: Equip diffusion models with llm for enhanced semantic alignment. *arXiv preprint arXiv:2403.05135*, 2024.

Allan Jabri, David J. Fleet, and Ting Chen. Scalable adaptive computation for iterative generation. *International Conference on Machine Learning (ICML)*, 2023.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Neural Information Processing Systems (NeurIPS)*, 2022.

Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and improving the training dynamics of diffusion models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Neural Information Processing Systems (NeurIPS)*, 2021.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 3992–4003, 2023.

Tuomas Kynkäänniemi, Miika Aittala, Tero Karras, Samuli Laine, Timo Aila, and Jaakko Lehtinen. Applying guidance in a limited interval improves sample and distribution quality in diffusion models. In *Neural Information Processing Systems (NeurIPS)*, 2024.

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

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, et al. Flux. 1 kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025.

Xingjian Leng, Jaskirat Singh, Yunzhong Hou, Zhenchang Xing, Saining Xie, and Liang Zheng. Repa-e: Unlocking vae for end-to-end tuning of latent diffusion transformers. *IEEE International Conference on Computer Vision (ICCV)*, 2025.

Hui Li, Jiayue Lyu, Fu-Yun Wang, Kaihui Cheng, Siyu Zhu, and Jingdong Wang. Mixflow training: Alleviating exposure bias with slowed interpolation mixture. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Mingxiao Li, Tingyu Qu, Ruicong Yao, Wei Sun, and Marie-Francine Moens. Alleviating exposure bias in diffusion models through sampling with shifted time steps. In *International Conference on Learning Representations (ICLR)*, 2024a.

Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *Neural Information Processing Systems (NeurIPS)*, 2024b.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2022.

Xingchao Liu, Chengyue Gong, et al. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations (ICLR)*, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

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

Mang Ning, Enver Sangineto, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Input perturbation reduces exposure bias in diffusion models. In *International Conference on Machine Learning (ICML)*, 2023.

Mang Ning, Mingxiao Li, Jianlin Su, Albert Ali Salah, and Itir Onal Ertugrul. Elucidating the exposure bias in diffusion models. *International Conference on Learning Representations (ICLR)*, 2024.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 2021.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE International Conference on Computer Vision (ICCV)*, 2023.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Neural Information Processing Systems (NeurIPS)*, 2016.

Florian Schmidt. Generalization in generation: A closer look at exposure bias. *arXiv preprint arXiv:1910.00292*, 2019.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021b.

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

Ren Sucheng, Yu Qihang, He Ju, Shen Xiaohui, Yuille Alan, and Chen Liang-Chieh. Beyond next-token: Next-x prediction for autoregressive visual generation. *arXiv preprint arXiv:2502.20388*, 2025.

Keqiang Sun, Junting Pan, Yuying Ge, Hao Li, Haodong Duan, Xiaoshi Wu, Renrui Zhang, Aojun Zhou, Zipeng Qin, Yi Wang, et al. Journeydb: A benchmark for generative image understanding. *Neural Information Processing Systems (NeurIPS)*, 2023.

Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Neural Information Processing Systems (NeurIPS)*, 2024.

Shengbang Tong, Boyang Zheng, Ziteng Wang, Bingda Tang, Nanye Ma, Ellis Brown, Jihan Yang, Rob Fergus, Yann LeCun, and Saining Xie. Scaling text-to-image diffusion transformers with representation autoencoders. *arXiv preprint arXiv:2601.16208*, 2026.

Shuai Wang, Ziteng Gao, Chenhui Zhu, Weilin Huang, and Limin Wang. Pixnerd: Pixel neural field diffusion. *arXiv preprint arXiv:2507.23268*, 2025a.

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

Ge Wu, Shen Zhang, Ruijing Shi, Shanghua Gao, Zhenyuan Chen, Lei Wang, Zhaowei Chen, Hongcheng Gao, Yao Tang, Jian Yang, et al. Representation entanglement for generation: Training diffusion transformers is much easier than you think. *Neural Information Processing Systems (NeurIPS)*, 2025.

Jingfeng Yao, Cheng Wang, Wenyu Liu, and Xinggang Wang. Fasterdit: Towards faster diffusion transformers training without architecture modification. *Neural Information Processing Systems (NeurIPS)*, 2024.

Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Vighnesh Birodkar, Agrim Gupta, Xiuye Gu, et al. Language model beats diffusion–tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*, 2023.

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

Z-Image Team, Huanqia Cai, Sihan Cao, Ruoyi Du, Peng Gao, Steven Hoi, Zhaohui Hou, Shijie Huang, Dengyang Jiang, Xin Jin, Liangchen Li, Zhen Li, Zhong-Yu Li, David Liu, Dongyang Liu, Junhan Shi, Qilong Wu, Feng Yu, Chi Zhang, Shifeng Zhang, and Shilin Zhou. Z-image: An efficient image generation foundation model with single-stream diffusion transformer. *arXiv preprint arXiv:2511.22699*, 2025.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Neural Information Processing Systems (NeurIPS)*, 2019.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. In *arXiv preprint arXiv:2510.11690*, 2025.

Hongkai Zheng, Weili Nie, Arash Vahdat, and Anima Anandkumar. Fast training of diffusion models with masked transformers. *arXiv preprint arXiv:2306.09305*, 2023.

Xingyu Zhou, Qifan Li, Xiaobin Hu, Hai Chen, and Shuhang Gu. Guiding a diffusion transformer with the internal dynamics of itself. *arXiv preprint arXiv:2512.24176*, 2025.

A Implementation

Our ImageNet experiments are conducted with the SiT and RAE training codebases. All experiments are conducted on 8-GPU H200 nodes.

Class-conditional generation. Tables 9 and 10 summarize the optimization and architecture settings for class-conditional generation. The SiT configurations follow SiT (Ma et al., 2024) and Lightning-DiT (Yao et al., 2025), and the RAE configuration follows RAE (Zheng et al., 2025). The default RAE training recipe trains for 800 epochs on TPUs; in our GPU reproduction, the 800-epoch checkpoint does not reach the performance reported in the RAE paper, whereas extending training to 1100 epochs matches the reported result. We therefore use 1100 epochs for the RAE setting.

Table 9: **Optimization hyperparameters for class-conditional training.**

Method	**SiT**	**RAE**
optimizer	AdamW	AdamW
learning rate schedule	constant, 2×10^{-4}	linear decay, 2×10^{-4} to 2×10^{-5} after 40-epoch warmup
global batch size	1024	1024
training epochs	800	1100
optimizer betas	$(0.9, 0.95)$	$(0.9, 0.95)$
model	SiT-S/B/L/XL	DiT^{DH}-XL
latent space	SD-VAE	DINOv2-B RAE

Table 10: **Architectural specifications of class-conditional diffusion backbones.**

Model	**Hidden size**	**Heads**	**Depth**
SiT-S/2	384	6	12
SiT-B/2	768	12	12
SiT-L/2	1024	16	24
SiT-XL/2	1152	16	28
DiT^{DH}-XL	1152	16	28

Following SiT (Ma et al., 2024), we use 250 sampling steps with the Heun solver, a guidance scale of 1.9, and the guidance interval $[0.3, 1.0]$. Following RAE (Zheng et al., 2025), we use 50 sampling steps with the Euler solver, an autoguidance scale of 1.6, the guidance interval $[0.05, 1.0]$, and class-balanced sampling.

Text-to-image generation. For text-to-image pretraining, we follow the data recipe of ScaleRAE (Tong et al., 2026) and use the Z-Image (Z-Image Team et al., 2025) architecture with a Qwen-7B text encoder and a 5.5B DiT backbone. The DiT diffusion backbone is randomly initialized and trained from scratch, while the text encoder and VAE are kept fixed. The text-to-image training and architecture hyperparameters are summarized in Table 11. After pretraining, we finetune all models on the BLIP-3o 60k dataset (Chen et al., 2025). The detailed hyperparameters are also provided in Table 11. For text-to-image evaluation, we use 50 sampling steps with the Euler solver and the guidance interval $[0.1, 1.0]$. For pretraining evaluation, we use a guidance scale of 8. For finetuning evaluation, we use a guidance scale of 4.

B Additional Results

We provide additional results for class-conditional generation and text-to-image generation.

Comparison with standard training under uniform, logit-normal, and triangular training timestep sampling. We compares MixFlow 2 with standard flow matching under three training timestep

Table 11: **Text-to-image DiT training and architecture hyperparameters.**

	Pretraining	**Finetuning**
hidden size	3072	3072
heads	32	32
depth	32	32
optimizer	AdamW	AdamW
learning rate schedule	constant w/ warmup ratio 0.0134	constant w/ warmup ratio 0.0134
global batch size	2048	512
training epochs	2	64
max learning rate	2×10^{-5}	1×10^{-5}
CFG dropout	0.1	0.1
time-shift	–	2
resolution	256×256	512×512

distributions: uniform, logit-normal, and triangular sampling. Table 12 reports unguided ImageNet 256×256 results across SiT-S/2, SiT-B/2, SiT-L/2, and SiT-XL/2. MixFlow 2 achieves the best gFID and IS at every model size.

Table 13 reports the corresponding SiT-XL/2 results under guidance interval and autoguidance. MixFlow 2 remains the best method in both guided settings, reaching gFID 1.58 with guidance interval and 1.50 with autoguidance.

Table 12: **MixFlow 2 outperforms standard training with uniform, logit-normal, and triangular training timestep sampling across model sizes.** The results are from no guidance.

Method	**SiT-S/2**		**SiT-B/2**		**SiT-L/2**		**SiT-XL/2**	
	gFID ↓	IS ↑	gFID ↓	IS ↑	gFID ↓	IS ↑	gFID ↓	IS ↑
Standard FM (uniform)	35.78	40.1	16.37	82.6	9.73	130.8	8.76	138.3
Standard FM (logit-normal)	31.92	49.0	14.79	90.5	8.25	135.1	7.72	144.7
Standard FM (triangular)	31.32	50.4	14.54	93.1	8.12	135.3	7.67	143.4
MixFlow 2	**28.43**	**54.7**	**12.75**	**101.0**	**7.29**	**144.8**	**6.49**	**158.5**

Table 13: **MixFlow 2 outperforms standard training with uniform, logit-normal, and triangular training timestep sampling under guidance interval and autoguidance.** The results are on SiT-XL/2.

Guidance	**Method**	**Scale**	**Interval**	gFID ↓	IS ↑
Guidance interval	Standard FM (uniform)	1.9	[0.3, 1.0]	1.85	268.9
	Standard FM (logit-normal)	1.9	[0.3, 1.0]	1.69	286.5
	Standard FM (triangular)	1.9	[0.3, 1.0]	1.69	287.3
	MixFlow 2	1.9	[0.3, 1.0]	**1.58**	**295.1**
Autoguidance	Standard FM (uniform)	1.9	–	1.75	230.1
	Standard FM (logit-normal)	1.9	–	1.64	235.1
	Standard FM (triangular)	1.9	–	1.65	235.9
	MixFlow 2	1.9	–	**1.50**	**240.2**

Detailed text-to-image results. Tables 14 and 15 report the detailed text-to-image results. We follow the same evaluation protocol as the main text and report detailed scores after pretraining and finetuning. MixFlow 2 gives the best overall score, outperforms both standard flow matching results with uniform and logit-normal training timestep sampling on GenEval and DPG-Bench.

Table 14: **Detailed GenEval scores.** MixFlow 2 outperforms standard flow matching with uniform and logit-normal timestep sampling in both pretraining and finetuning.

Method	Single	Two	Counting	Colors	Position	Attribute binding	Overall ↑
Pretraining							
Standard FM (uniform)	0.90	0.40	0.34	0.71	0.12	0.33	0.47
Standard FM (logit-normal)	0.91	0.50	0.35	0.77	0.16	0.32	0.50
MixFlow 2	0.94	0.53	0.43	0.82	0.25	0.38	**0.56**
Finetuning							
Standard FM (uniform)	0.94	0.79	0.63	0.81	0.72	0.60	0.75
Standard FM (logit-normal)	0.95	0.85	0.66	0.83	0.72	0.63	0.77
MixFlow 2	0.98	0.90	0.71	0.88	0.78	0.68	**0.82**

Table 15: **Detailed DPG-Bench scores.** MixFlow 2 outperforms standard flow matching with uniform and logit-normal timestep sampling in both pretraining and finetuning.

Method	Global	Entity	Attribute	Relation	Other	Overall ↑
Pretraining						
Standard FM (uniform)	79.42	82.78	83.91	85.12	82.45	71.43
Standard FM (logit-normal)	83.67	85.43	84.60	87.98	83.87	75.80
MixFlow 2	87.27	85.62	85.13	86.57	86.04	**77.56**
Finetuning						
Standard FM (uniform)	81.56	83.58	84.40	85.20	83.67	73.10
Standard FM (logit-normal)	88.35	85.50	84.96	86.03	84.05	76.50
MixFlow 2	89.10	86.12	85.97	88.12	86.23	**78.76**

C Qualitative Results

We provide additional ImageNet 256×256 and 512×512 samples generated by MixFlow 2.

Figure 8: **Additional class-conditional** 256×256 **samples.**

Figure 9: **Additional class-conditional** 512×512 **samples.**