arXiv:2604.11521v2 [cs.LG] 25 Aug 2026

Continuous Adversarial Flow Models

Shanchuan Lin, Ceyuan Yang, Zhijie Lin, Hao Chen, and Haoqi Fan

ByteDance Seed

Abstract. We propose continuous adversarial flow models, a type of continuous-time flow model trained with an adversarial objective. Unlike flow matching, which uses a fixed mean-squared-error criterion, our approach introduces a learned discriminator to guide training. This change in objective induces a different generalized distribution, which empirically produces samples that are better aligned with the target data distribution. Our method is primarily proposed for post-training existing flow-matching models, although it can also train models from scratch. On the ImageNet 256px generation task, our post-training substantially improves the guidance-free FID of latent-space SiT from 8.26 to 3.63 and of pixel-space JiT from 7.17 to 3.57. It also improves guided generation, reducing FID from 2.06 to 1.53 for SiT and from 1.86 to 1.80 for JiT. We further evaluate our approach on text-to-image generation, where it achieves improved results on both the GenEval and DPG benchmarks.

Keywords: Generative models · Adversarial training · Flow models



Top: Flow Matching

Bottom: Continuous Adversarial Flow Models (Ours)

Fig. 1: Generation without guidance. Our method yields better generalization.

1 Introduction

Flow matching [46] has achieved significant success in recent years, yet a critical problem remains. The issue is particularly evident in the generation of visual

[1] Correspondence to: Shanchuan Lin <peterlin@bytedance.com>

[2] Code: https://github.com/ByteDance-Seed/Adversarial-Flow-Models

modalities, such as image [6, 67] and video [15, 65, 66] synthesis, where models often produce out-of-distribution samples unless guidance is applied [14, 21, 33]. While guidance improves sample quality, it alters the sampling distribution. How to more faithfully model the underlying distribution of the original data remains an open problem.

One reason flow matching generates out-of-distribution samples is that it uses a Euclidean distance criterion rather than a manifold-aware one. Concretely, flow matching (FM) learns the velocity field of a probability flow [68] between noise and data distributions. Training minimizes the squared L_2 loss between predicted and target velocities. In theory, this objective converges to the ground-truth flow under an infinite-capacity model, which would overfit and reproduce the training samples exactly. In practice, models with finite capacity must generalize, and therefore results in the generation of new data samples. However, the squared L_2 objective measures Euclidean distance rather than the manifold-aware distance, inducing incorrect generalization relative to the underlying data distribution.

Recent work has attempted to tackle the issue from different angles. Representational autoencoders [83] convert the data space on which flow matching operates and have empirically reported improvements in generation quality, but this requires operating in a latent space instead of the original data space. Riemannian flow matching [7] extends flow matching to non-Euclidean geometries, but this requires manual definition of the data manifold, which is often unknown for general datasets. Other work [44] replaces Euclidean loss with perceptual distances derived from frozen feature networks, motivated by the empirical finding that deep networks can serve as better perceptual metrics [81]. However, a fixed criterion network can be exploited by the generator [18], leading to artifacts in the generated samples. A way to mitigate generator hacking is to jointly train the criterion network with the generator, which yields a dynamic reminiscent of generative adversarial networks.

Generative adversarial networks (GANs) [17] are a standalone class of generative methods. They achieve strong performance on ImageNet benchmarks [26, 28, 43, 64] and are widely used in flow-model distillation for sharp image synthesis [40–42, 45, 57, 62, 63, 77, 78]. We hypothesize that this advantage arises because the discriminator networks are more sensitive to perceptual details, *e.g.* texture, sharpness, contour, *etc.*, than pointwise Euclidean losses, because they may have learned to better capture the manifold structure. Recent work, adversarial flow models (AFMs) [43], combines adversarial and flow modeling, improving training stability and extending adversarial objectives to multi-step flow training. However, AFMs are formulated in discrete time, leaving open the question of how to incorporate adversarial training into continuous-time flow modeling.

In this paper, we introduce continuous adversarial flow models (CAFMs), which extend AFMs to continuous time. CAFMs are a type of continuous normalizing flow (CNF) [8] that generates samples by integrating an ordinary differential equation (ODE) from noise to data. Like flow-matching models (FMs), CAFMs also learn the velocity field of a predefined probability flow with a simulation-free objective. Although FMs and CAFMs target the same ground-

truth flow, they differ in finite-capacity generalization because CAFMs use a learned discriminator rather than a fixed Euclidean criterion. Empirically, our experiments find that CAFMs produce more in-distribution samples, both perceptually and by various metrics. To the best of our knowledge, our work is the first to apply adversarial training in continuous-time flow modeling.

Since FMs and CAFMs learn the same ground-truth flow and differ mainly in model generalization, our method is primarily designed to post-train existing FMs for efficiency and practicality, although the objective can also be used for training from scratch. In class-conditional ImageNet [60] 256px generation, CAFM post-training improves the guidance-free FID for latent-space SiT [50] from 8.26 to 3.63, and for pixel-space JiT [37] from 7.17 to 3.57, using only 10 epochs of finetuning. CAFMs also achieve better guided generation, improving the FID from 2.06 to 1.53 for SiT and from 1.86 to 1.80 for JiT. In text-to-image experiments, CAFMs increase the GenEval [16] score from 0.81 to 0.85 and the DPG [25] score from 83.7 to 85.2. These results suggest promising prospects for integrating adversarial training into continuous-time flow modeling—not for few-step generation, but for improving sample fidelity and distribution matching.

2 Background

2.1 Flow Matching

Flow matching (FM) [46] formulates the generation problem as transporting samples from a prior distribution $z \sim \mathcal{Z} \in \mathbb{R}^n$, often Gaussian $\mathcal{N}(0, \mathbf{I})$, to the data distribution $x \sim \mathcal{X} \in \mathbb{R}^n$ over a probability flow, defined by an interpolation function:

$$x_t = A(t)x + B(t)z, \tag{1}$$

where $t \in [0, 1]$. Linear interpolation [46, 47] is commonly used, where $A(t) = 1 - t$, $B(t) = t$, and $x_t = (1 - t)x + tz$.

The time derivative at position x_t conditioned on x and z, called the conditional velocity $\bar{v}_t$, can be derived as:

$$\bar{v}_t = \frac{dA(t)}{dt}x + \frac{dB(t)}{dt}z, \tag{2}$$

which for linear interpolation, $\frac{dA(t)}{dt} = -1$, $\frac{dB(t)}{dt} = 1$, and $\bar{v}_t = -x + z$.

Flow matching trains a generator $G(x_t, t) : \mathbb{R}^n \times [0, 1] \to \mathbb{R}^n$ to match the conditional velocity $\bar{v}_t$:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{x,z,t}\left[d(G(x_t, t), \bar{v}_t)\right], \tag{3}$$

and finds that this conditional flow matching objective in expectation over independent coupling of x, z learns the marginal velocity $v_t = \mathbb{E}[\bar{v}_t \mid x_t]$ of the probability flow when the criterion $d(a, b)$ satisfies:

$$\arg\min_a \mathbb{E}_b[d(a, b)] = \mathbb{E}[b]. \tag{4}$$

The squared L_2 criterion is adopted. The mean squared error (MSE) variant, with an additional factor of $\frac{1}{n}$, is most commonly used:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{x,z,t}\left[\frac{1}{n}\|G(x_t,t) - \bar{v}_t\|_2^2\right]. \tag{5}$$

The resulting generator $G(x_t, t)$ defines a continuous-time flow model that predicts the marginal velocity field v_t at each state x_t along the probability flow. Samples are transported from the noise distribution to the data distribution by integrating the ODE:

$$x_0 = x_1 + \int_0^1 G(x_t,t)\ dt, \quad x_1 \sim \mathcal{Z}, \tag{6}$$

where the integration runs backward from $t = 1$ to $t = 0$.

The limitation of flow matching. Using any criterion $d(a, b)$ satisfying Eq. (4) in theory ensures the model converges to $v_t = \mathbb{E}[\bar{v}_t \mid x_t]$, but this overfits to generating only the training samples. In practice, models parameterized by neural networks have finite capacity and learn a generalized distribution. In this case, the loss objective affects the way of generalization. Consider:

$$d(a,b) = (a-b)^\top M(a-b), \tag{7}$$

where the squared L_2 criterion corresponds to the special case $M = I$. In general, M can be any strictly positive definite matrices (Sec. C). These objectives converge to the same ground-truth flow, but can induce different generalizations. Flow matching minimizes the isotropic Euclidean distance without awareness of the data manifold, leading to incorrect generalization and out-of-distribution generation.

Using a manifold criterion. A natural idea is to replace the squared L_2 criterion with one that measures distance on the data manifold. However, the underlying data manifold is not known in advance and must itself be inferred and generalized from the limited training data. Our work explores adversarial training, where a criterion network is learned simultaneously along with the generator. This is encouraged by previous empirical findings that deep networks can better capture the data manifold, as evidenced by their ability to serve as a better perceptual distance than the Euclidean metric [44, 81].

2.2 Adversarial Flow Models

Adversarial flow models (AFMs) [43] are a type of discrete-time flow model trained with an adversarial objective. The training involves a generator $G(x_s, s, t) : \mathbb{R}^n \times [0,1] \times [0,1] \to \mathbb{R}^n$ that transports samples from source x_s to target x_t on the probability flow, and a discriminator $D(x_t, t) : \mathbb{R}^n \times [0,1] \to \mathbb{R}$ that differentiates the real and generated x_t samples.

Adversarial training involves a minimax optimization game where D aims to maximize discrimination while G aims to minimize discrimination by D. The adversarial objective is defined as:

$$\mathcal{L}^D_{\text{adv}} = \mathbb{E}_{x,z,s,t}\left[f\big(D(x_t,t), D(G(x_s,s,t),t)\big)\right], \tag{8}$$

$$\mathcal{L}^G_{\text{adv}} = \mathbb{E}_{x,z,s,t}\left[f\big(D(G(x_s,s,t),t), D(x_t,t)\big)\right], \tag{9}$$

where $f(a,b) = -\log(\text{sigmoid}(a-b))$ is one of many viable contrastive functions used by recent work [26,27,29,43]. Training updates G and D in alternation and reaches equilibrium when $G(x_s,s,t)$ produces the same distribution of x_t.

AFMs additionally introduce an optimal transport objective on G:

$$\mathcal{L}^G_{\text{ot}} = \mathbb{E}_{x,z,s,t}\left[\frac{1}{n}\cdot\frac{1}{|t-s|}\cdot\|G(x_s,s,t)-x_s\|_2^2\right]. \tag{10}$$

It minimizes the distance to x_s, not $\bar{v}_s$, unlike flow matching. Over the expectation, this encourages G to predict targets x_t that are closest to the sources x_s, allowing G to learn a unique optimal transport for stable training.

Additionally, D is regulated by gradient penalties R_1 and R_2 [59] to mitigate the problem of vanishing gradient [2] and a centering penalty [31] to prevent logit drifting:

$$\mathcal{L}^D_{\text{r1}} = \mathbb{E}_{x,z,s,t}\left[\|\nabla_{x_t} D(x_t,t)\|_2^2\right], \tag{11}$$

$$\mathcal{L}^D_{\text{r2}} = \mathbb{E}_{x,z,s,t}\left[\|\nabla_{G(x_s,s,t)} D(G(x_s,s,t),t)\|_2^2\right], \tag{12}$$

$$\mathcal{L}^D_{\text{cp}} = \mathbb{E}_{x,z,s,t}\left[(D(x_t,t)+D(G(x_s,s,t),t))^2\right]. \tag{13}$$

The final training objectives of AFMs are:

$$\mathcal{L}^D_{\text{AFM}} = \mathcal{L}^D_{\text{adv}} + \lambda_{\text{gp}}\mathcal{L}^D_{\text{r1}} + \lambda_{\text{gp}}\mathcal{L}^D_{\text{r2}} + \lambda_{\text{cp}}\mathcal{L}^D_{\text{cp}}, \tag{14}$$

$$\mathcal{L}^G_{\text{AFM}} = \mathcal{L}^G_{\text{adv}} + \lambda_{\text{ot}}\mathcal{L}^G_{\text{ot}}. \tag{15}$$

To generate, AFMs transport samples from the noise distribution to the data distribution by solving the difference equation:

$$x_0 = x_1 + \sum_{i=1}^{S}\big(G(x_{\tau_i},\tau_i,\tau_{i-1}) - x_{\tau_i}\big), \quad x_1 \sim \mathcal{Z}, \tag{16}$$

where the summation runs backward from $i = S$ to $i = 1$ with a total of S sampling steps, and τ is a list of discrete timesteps satisfying $\tau_0 = 0,\ \tau_S = 1$.

The limitation of adversarial flow models. AFMs are a form of discrete-time flow models. Although the timestep interval $|t-s|$ can be made arbitrarily small, the training becomes increasingly unstable, and the objective breaks down when $|t-s| = 0$. It is not clear how to extend adversarial training to continuous-time flow modeling. Furthermore, AFMs still have the gradient-vanishing problem [2]. They rely on gradient penalties [59], discriminator augmentation [32], and discriminator reset [43] to mitigate the issue.

3 Method

3.1 Continuous Adversarial Flow Models

We propose continuous adversarial flow models (CAFMs) to extend adversarial training to continuous-time flow modeling. Our method involves a generator $G(x_t, t) : \mathbb{R}^n \times [0, 1] \to \mathbb{R}^n$ of the same form as in flow matching, which predicts the velocity field v_t at x_t, and a discriminator $D(x_t, t) : \mathbb{R}^n \times [0, 1] \to \mathbb{R}$ of the same form as in AFMs. Unlike discrete-time adversarial training, we discriminate v_t in the derivative space of D, explicitly reflecting the physical property of velocity v_t as a derivative of position x_t.

Specifically, we denote the Jacobian-Vector Product (JVP) of D with primal (x_t, t) and tangent $(\dot{x}_t, \dot{t})$ as:

$$D_{\text{jvp}}(x_t, t, \dot{x}_t, \dot{t}) = \frac{\partial D(x_t, t)}{\partial x_t}\dot{x}_t + \frac{\partial D(x_t, t)}{\partial t}\dot{t}, \tag{17}$$

where:

$$\frac{\partial D(x_t, t)}{\partial x_t} \in \mathbb{R}^{1\times n} \quad \text{and} \quad \frac{\partial D(x_t, t)}{\partial t} \in \mathbb{R}^{1\times 1} \tag{18}$$

are the Jacobian matrices of the actual network $D(x_t, t)$ with respect to the primal variables x_t and t. The entire JVP function also outputs a scalar, which we use as the discrimination logit:

$$D_{\text{jvp}}(x_t, t, \dot{x}_t, \dot{t}) : (\mathbb{R}^n \times [0, 1] \times \mathbb{R}^n \times [0, 1]) \to \mathbb{R}. \tag{19}$$

During training, D_{jvp} is evaluated using (x_t, t) as primal and $(\bar{v}_t, T)$ as tangent, where $T = 1$ for networks trained with $t \in [0, 1]$. The continuous-time adversarial objectives are defined as:

$$\mathcal{L}^D_{\text{adv}'} = \mathbb{E}_{x,z,t}[f(D_{\text{jvp}}(x_t, t, \bar{v}_t, T), D_{\text{jvp}}(x_t, t, G(x_t, t), T))], \tag{20}$$

$$\mathcal{L}^G_{\text{adv}'} = \mathbb{E}_{x,z,t}[f(D_{\text{jvp}}(x_t, t, G(x_t, t), T), D_{\text{jvp}}(x_t, t, \bar{v}_t, T))], \tag{21}$$

where we adopt a bounded contrastive function, similar to prior work [51]:

$$f(a, b) = (a - 1)^2 + (b + 1)^2. \tag{22}$$

The gradients with respect to the model parameters are backpropagated through the JVP.

Fig. 2 visualizes the intuition and training dynamics of our method. Intuitively, our discriminator D learns a scalar potential whose directional derivative distinguishes real and fake flows. D learns to assign higher potential to more realistic directions, and G is optimized toward the direction that maximizes D's potential. Training reaches equilibrium when G learns the ground-truth flow and D outputs flat potentials everywhere.

Since the objectives in Eqs. (20) and (21) only penalize the derivative while the absolute value of D is free to drift, we also include a centering penalty to keep the absolute value of D centered around zero:

$$\mathcal{L}^D_{\text{cp}'} = \mathbb{E}_{x,z,t}\left[D(x_t, t)^2\right]. \tag{23}$$

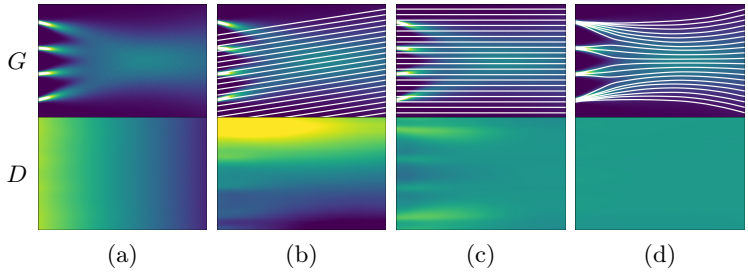


Fig. 2: Visualization of the training dynamic. Top: learned $G(x_t, t)$ trajectories over the probability flow. Bottom: corresponding $-D(x_t, t)$ values at all x_t. $-D$ is taken for more intuitive visualization as the generation process runs backward in time. (a) shows if only training D with $\bar{v}_t$ as positives without G as negatives, D degenerates to uniform gradient. (b,c) show how D reacts to G during training. (d) shows D converges to 0 everywhere as G converges to the ground-truth flow.

When training on high-dimensional flows where $n > 1$, the discriminator, which projects an n-dimensional input to a scalar value, creates ambiguity because multiple $v_t \in \mathbb{R}^n$ can yield the same value. G may learn to exploit the null space, and D is then updated to counter this behavior. However, this causes slow convergence. A regularizer can be added to encourage G to pick the minimum-norm solution, which is related to optimal transport regularization. The continuous-time optimal transport regularization on G is equivalent to its discrete counterpart in Eq. (10) in the limit of $|t - s| \to 0$:

$$\mathcal{L}^G_{\text{ot}'} = \mathbb{E}_{x,z,t}\left[\frac{1}{n}\|G(x_t, t)\|_2^2\right]. \tag{24}$$

Extending adversarial training to continuous time also mitigates the gradient vanishing problem (Sec. E). Empirically, we find that CAFMs can be trained without gradient penalties in our experiments. We also find it beneficial to train D toward optimality by updating it for N steps per update of G.

The final objectives for CAFMs resemble the discrete-time counterparts, except for the removal of the gradient penalties:

$$\mathcal{L}^D_{\text{CAFM}} = \mathcal{L}^D_{\text{adv}'} + \lambda_{\text{cp}}\mathcal{L}^D_{\text{cp}'}, \tag{25}$$

$$\mathcal{L}^G_{\text{CAFM}} = \mathcal{L}^G_{\text{adv}'} + \lambda_{\text{ot}}\mathcal{L}^G_{\text{ot}'}. \tag{26}$$

We follow AFMs to gradually reduce λ_{ot} when training from scratch. For post-training existing flow-matching models, we set $\lambda_{\text{ot}} = 0$ to completely eliminate the bias of the Euclidean norm. The centering penalty is set to $\lambda_{\text{cp}} = 0.001$. We provide additional proofs and discussions in Sec. D of the appendix.

3.2 Practical and Efficient Implementation

JVP can be efficiently computed with forward-mode automatic differentiation. It computes both $D(x_t, t)$ and $D_{\text{jvp}}(x_t, t, \dot{x}_t, \dot{t})$ in a single forward pass, allowing us to derive the adversarial loss $\mathcal{L}^D_{\text{adv}'}$ and the centering penalty $\mathcal{L}^D_{\text{cp}'}$ together efficiently. Additionally, we use vectorizing map (vmap) to efficiently compute multiple tangents at the same primal when updating D. A concise PyTorch implementation is provided in Algorithm 1. For larger-scale training, JVP and vmap are compatible with PyTorch's DDP [36], FSDP [82], and gradient checkpointing [10]. Implementation details are in Sec. F.

Algorithm 1 Continuous adversarial flow training

```python
from functools import partial
from torch import mean, ones_like, stack, unbind
from torch.func import jvp, vmap

def step(G, D, x, z, t, c, mode, cp_scale, ot_scale):
    D.requires_grad_(mode == "dis")
    G.requires_grad_(mode == "gen")

    D = partial(D, condition=c)
    G = partial(G, condition=c)

    x_t = (1 - t) * x + t * z
    v_t = -x + z
    u_t = G(x_t, t)
    T = ones_like(t)

    if mode == "dis":
        o, do = vmap(lambda *tangents: jvp(D, (x_t, t), tangents))(
            stack([v_t, u_t]),
            stack([T, T])
        )
        dv, du = unbind(do)
        return (
            mean((dv - 1) ** 2) +
            mean((du + 1) ** 2) +
            mean(o ** 2) * cp_scale
        )
    else:
        _, du = jvp(D, (x_t, t), (u_t, T))
        return (
            mean((du - 1) ** 2) +
            mean(u_t ** 2) * ot_scale
        )
```

In terms of the network architecture, there are no restrictions on G because it does not involve JVP computation and can use any architectures, same as in flow matching. For D, we find that switching LayerNorm [4] to RMSNorm [80] significantly improves training stability, consistent with the findings from previous research involving JVP computation [85]. Unlike prior work, we do not find additional normalization on modulation necessary [49,85]. Our experiments show that CAFMs work well with standard transformers [70] as both G and D.

3.3 Pre-training *vs*. Post-training

Although CAFMs can be trained from scratch, it is inherently less efficient than FMs due to the involvement of an extra discriminator network, the forward and backward computation of JVP, and the multiple steps of discriminator learning per generator update. Since both FMs and CAFMs learn the same probability flow and differ only in model generalization, it is much more efficient to pre-train models with FM objective and post-train with CAFM objective. Therefore, we primarily propose CAFMs for post-training, but still show that the objective can be used to train models from scratch, albeit less efficiently.

4 Experiment

4.1 ImageNet Generation Post-training

On the class-conditional ImageNet [60] 256px generation task, we conduct experiments to post-train both latent-space flow-matching model SiT [50] and pixel-space flow-matching model JiT [37] with the CAFM objective and obtain significant performance gains in both guidance-free and guided settings, as measured by the Fréchet Inception Distance (FID) [20] and Inception Score (IS) [61].

Table 1: SiT-XL/2 on ImageNet 256px. Full comparisons are in Tab. 7 of the appendix.

CFG	Method	Epoch	FID↓	IS↑
None	SiT	1400	8.26	131.65
	SiT+FM	1400+10	8.64	131.91
	SiT+CAFM	1400+10	**3.63**	**178.08**
1.1	SiT	1400	5.55	161.77
	SiT+CAFM	1400+10	**2.27**	**212.06**
1.2	SiT	1400	3.65	190.57
	SiT+CAFM	1400+10	**1.66**	**238.44**
1.3	SiT	1400	2.57	220.52
	SiT+CAFM	1400+10	**1.53**	**263.52**
1.4	SiT	1400	2.07	248.31
	SiT+CAFM	1400+10	**1.66**	**283.59**
1.5	SiT	1400	2.06	277.50
	SiT+CAFM	1400+10	**1.97**	**301.91**
1.6	SiT	1400	**2.25**	293.72
	SiT+CAFM	1400+10	2.37	**316.78**

Table 2: JiT-H/16 on ImageNet 256px. Full comparisons are in Tab. 13 of the appendix.

CFG	Method	Epoch	FID↓	IS↑
None	JiT	600	7.17	151.54
	JiT+FM	600+10	9.30	139.00
	JiT+CAFM	600+10	**3.57**	**198.08**
1.4	JiT	600	3.24	219.52
	JiT+CAFM	600+10	**2.01**	**258.46**
1.6	JiT	600	2.49	244.61
	JiT+CAFM	600+10	**1.84**	**275.96**
1.8	JiT	600	2.12	265.43
	JiT+CAFM	600+10	**1.80**	**290.71**
2.0	JiT	600	1.96	281.38
	JiT+CAFM	600+10	**1.83**	**301.96**
2.2	JiT	600	**1.86**	303.40
	JiT+CAFM	600+10	1.88	**310.54**
2.4	JiT	600	2.19	310.20
	JiT+CAFM	600+10	**1.95**	**319.23**

SiT. We adopt the officially pre-trained SiT-XL/2 model as the starting point for G and keep the architecture completely unchanged. D adopts the same architecture and weight initialization, except changing all LayerNorm to RMSNorm. We additionally follow the same modifications in AFM to prepend a learnable

[CLS] token at input and add projection layers for the discriminator logit output. We use the same batch size of 256 as the original SiT. We set the learning rate to 1e-5 for both G and D. We use Adam [35] optimizer with $\beta = (0, 0.95)$. For the first 2 epochs, we freeze G and only update D for it to adapt to the new architecture. Then, we set $N = 16$ to update D 16 times per G's update. Epochs are measured as the combined number of images seen by both G and D throughout our experiments. We use an exponential moving average (EMA) with a short decay of 0.99 on G. We set $\lambda_{\text{ot}} = 0$ for post-training. We use the exact inference and evaluation code provided by SiT, and use the Euler-Maruyama SDE sampler with 250 integration steps to match SiT's best setting. Table 1 shows that CAFM post-training significantly improves the FID from 8.26 to 3.63 in the guidance-free setting, and also improves the best FID from 2.06 to 1.53 in the guided setting under just 10 epochs of finetuning. For classifier-free guidance (CFG) [21], the sweep finds that CAFMs achieve the best FID using CFG 1.3, which is lower than the original SiT at CFG 1.5. CAFMs improve generation at almost every swept CFG level. We also run a controlled trial by using the FM objective. It does not yield benefits compared to the SiT baseline, and the FID difference can be within the error margin of random evaluation sampling. This proves that the gains are a result of the CAFM objective. More ablation studies are provided in Sec. A.

JiT. We adopt the officially pre-trained JiT-H/16 model as the starting point for G and keep the architecture completely unchanged. D adopts the same architecture and weight initialization as G. Because JiT already uses RMSNorm and has in-context class tokens, we simply take the first class token and add projection layers for the discriminator output. We convert the x-prediction result by G to v before giving it to D. We use the same batch size of 1024 as the original JiT. Follow SiT, we also set the learning rate to 1e-5, $\beta = (0, 0.95)$, $N = 16$, and EMA decay to 0.99. D is warmed up for the first 4 epochs. We use the exact inference and evaluation code provided by JiT. We follow JiT to use the Heun ODE sampler with 50 steps. Table 2 shows that CAFMs also significantly improve the FID from 7.17 to 3.57 in the guidance-free setting, and improves the best FID from 1.86 to 1.80 in the guided setting. CAFMs improve performance in almost all swept CFG levels. We find that the FM control trial produces worse results than JiT's official checkpoint despite our best effort to reproduce. Regardless, it is sufficient to prove that the gains are originated from the CAFM objective.

Comparisons to the state of the arts. Tables 3 and 4 compare our results to other methods. Under SD-VAE [58] latent space and without using DINOv2 [55], our method achieves the best performance in both guided and guidance-free settings among the models compared. Since all works use the DiT architecture and similar training settings, it is easier to attribute the gain to our method. In pixel space, settings vary significantly, making it harder to pinpoint contributions by the method from architectural improvements. We suspect that SiD [22] achieves better FID in the guidance-free setting because its 2B-parameter model can over-

Table 3: SD-VAE latent-space continuous flow models on ImageNet 256px. Methods in gray use DINOv2 [55].

Guided	Method	Param	FID↓
No	DiT-XL/2 [56]	675M	9.62
	SiT-XL/2 [50]	675M	8.26
	SiT-XL/2+Disperse [73]	675M	7.43
	DDT-XL [75]	675M	6.27
	SiT-XL/2+REPA [79]	675M	5.90
	SiT-XL/2+CAFM	675M	**3.63**
Yes	DiT-XL/2 [56]	675M	2.27
	SiT-XL/2 [50]	675M	2.06
	SiT-XL/2+Disperse [73]	675M	1.97
	SiT-XL/2+CAFM	675M	**1.53**
	SiT-XL/2+REPA [79]	675M	1.42
	DDT-XL [75]	675M	1.26

Table 4: Pixel-space continuous flow models on ImageNet 256px. Please consider that architectures and settings vary.

Guided	Method	Param	FID↓
No	ADM [14]	554M	10.94
	JiT-H/16 [37]	956M	7.17
	JiT-H/16+CAFM	956M	**3.57**
	SiD [22]	2B	**2.77**
Yes	ADM-G [14]	554M	4.59
	SiD [22]	2B	2.44
	PixNerd-XL/16 [74]	700M	2.15
	PixelFlow-XL/4 [9]	677M	1.98
	JiT-H/16 [37]	956M	1.86
	JiT-G/16 [37]	2B	1.82
	JiT-H/16+CAFM	956M	**1.80**
	SiD2 [23]	653M	**1.38**

fit the training data better. Overall, our method also achieves very competitive performance in the pixel space.

4.2 Text-to-Image Generation Post-training

Setup. We experiment post-training with a text-to-image generation model, Z-Image [6], using our CAFM objective. We first train the model with FM on our data for 10K iterations, then switch to CAFM for 20K, while keeping the FM trial running to match the iterations. Following common finetuning practice, FM training uses a batch size of 1024, AdamW optimizer [48] with a learning rate of 5e-5, $\beta = (0.9, 0.95)$, weight decay of 0.01, and an EMA decay of 0.999. Then, we switch to the CAFM objective while matching most of the FM settings. We lower D's learning rate to 3e-5 to avoid loss spiking while keeping G at 5e-5. We set $\beta = (0, 0.95)$. We lower the EMA decay to 0.99 to account for $N = 16$ discriminator update steps.

Evaluation. Our models are evaluated by both GenEval [16] in Tab. 5 and by DPG-Bench [25] in Tab. 6. In GenEval, we use the prompt expansion (PE) provided by prior work [1,13]. In both benchmarks, CAFM post-training significantly improves the performance of guidance-free generation, while also improving the guided setting.

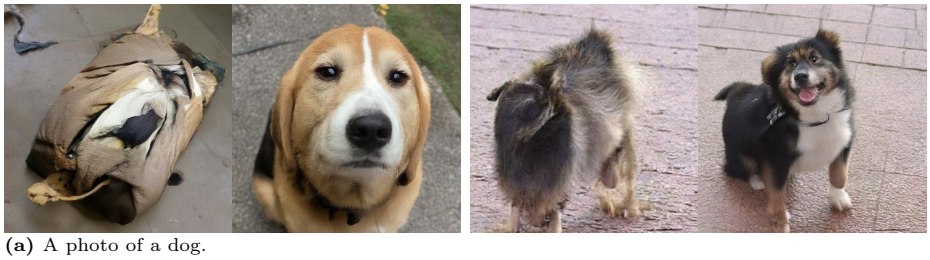
Limitation. Although CAFMs empirically achieve better performance in guidance-free generation, there is no guarantee that the models generalize to the true underlying data distribution, especially in the low-density regions containing outliers. Guidance can be used orthogonally to our method to improve benchmark scores as a low-temperature sampling technique.

Table 5: GenEval [16] on 512px text-to-image generation.

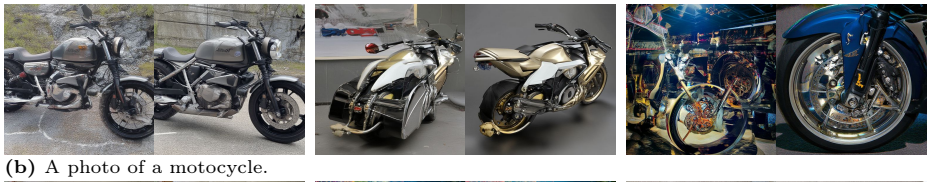
Method	PE	CFG	Single Obj.	Two Obj.	Color Attr.	Position	Counting	Colors.	Overall
FM	No	No	0.72	0.23	0.11	0.09	0.25	0.59	0.33
CAFM			0.85	0.42	0.17	0.16	0.41	0.61	0.44
FM	Yes	No	0.95	0.66	0.35	0.40	0.42	0.81	0.60
CAFM			0.99	0.83	0.50	0.52	0.57	0.86	0.71
FM	Yes	Yes	0.99	0.89	0.62	0.69	0.77	0.89	0.81
CAFM			0.99	0.92	0.71	0.71	0.81	0.94	0.85

Table 6: DPG-Bench [25] on 512px text-to-image generation.

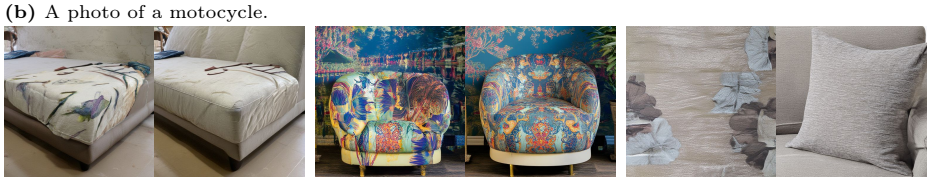
Method	CFG	Global	Entity	Attribute	Relation	Other	Overall
FM	No	81.34	82.96	81.71	83.17	85.07	72.25
CAFM		87.82	86.65	86.33	86.49	84.85	77.21
FM	Yes	90.34	90.56	88.98	88.17	90.71	83.67
CAFM		89.55	89.83	89.99	91.20	91.88	85.21



(a) A photo of a dog.



(b) A photo of a motocycle.



(c) A photo of a couch.



(d) A photo of a bus.

Fig. 3: Curated text-to-image samples on GenEval prompts.
Without PE and CFG to show the most diverse range of samples.
Left is FM. Right is CAFM. More visualizations are in Fig. 10 of the appendix.

4.3 ImageNet Generation Trained from Scratch

Although CAFM is proposed primarily as a post-training method, for completeness, we also experiment with training from scratch using the CAFM objective on ImageNet 256px. We use the SiT-B/2 architecture with a batch size of 256, an optimizer learning rate of 1e-4 with $\beta = (0, 0.95)$ for both G and D, and an EMA decay of 0.9999, matching the original pre-training settings of SiT. The hyperparameters of the discriminator updates per generator update N and the optimal transport loss weighting λ_{ot} are searched during training for the fastest convergence. In Fig. 4, we show that the CAFM objective can be used to train from scratch, but converges more slowly than FM under the same epochs, fitting our expectation in Sec. 3.3.

Ablation studies on the hyperparameters. Overall, we find that λ_{ot} should decrease over training and N should increase over training for the best performance. In Fig. 4a, we first fix $\lambda_{\text{ot}} = 1$ and compare the hyperparameter of N. We find $N = 4$ outperforms $N = 1$ after the first 50 epochs, so we use $N = 4$. Then in Fig. 4b, we search for λ_{ot} and find that $\lambda_{\text{ot}} = 4$ converges the fastest in the first 50 epochs. Therefore, we use $N = 4, \lambda_{\text{ot}} = 4$ as the initial settings. In Fig. 4c, we experiment with a lower λ_{ot} to 1 since 160 epochs and see further FID improvement, while $\lambda_{\text{ot}} = 4$ eventually plateaus. This shows the importance of decreasing λ_{ot} over training, concurring with the findings of AFM. In Fig. 4d, we further increase N to 8 at 700 epochs and see faster convergence at the later stage. Note that we have swept other changes during training, including decreasing the learning rate, further decreasing λ_{ot}, and further increasing N to match the settings of post-training, but they yield worse performance. We suspect these changes are too early within our 1000-epoch pre-training budget. We leave further explorations on pre-training to future work.

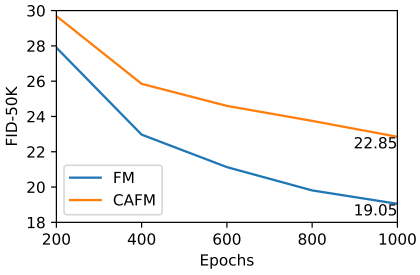
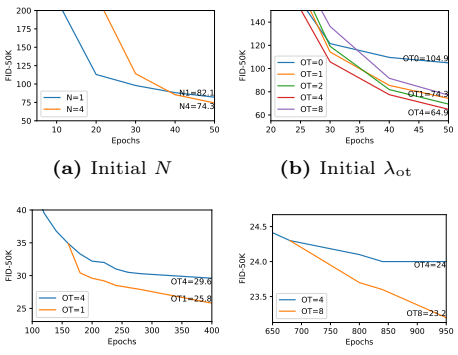


Fig. 4: SiT-B/2 pre-training on ImageNet 256px generation. Although CAFM can be used for pre-training, it converges slower than FM under the same epochs, fitting our expectation that CAFM is more suitable for post-training.



(a) Initial N **(b)** Initial λ_{ot}

(c) Reduce λ_{ot} at 160ep **(d)** Increase N at 700ep

Fig. 5: Ablation studies on the effect of different hyperparameters.

5 Related Work

Unifying Adversarial and Flow Modeling. Adversarial training originates from generative adversarial networks (GANs) [17]. Recent work, adversarial flow models (AFMs), combines adversarial and discrete-time flow modeling. Our work on CAFMs is an extension of AFMs into continuous time.

Adversarial Post-Training. Adversarial post-training of existing flow models has largely been researched as distillation methods to achieve few-step generation [11, 30, 40–42, 45, 57, 62, 63, 71, 72, 77, 78]. Our work applies adversarial post-training on continuous-time flow models for inducing different model generalization instead.

Generalization Behavior. Prior work [7, 12, 52] has explored lifting the flow models to custom manifolds, where the trajectories lie on the defined manifolds and hence also alter the model generalization. Our method still flows through the Euclidean space with the same ground-truth trajectories as standard flow matching but only induces different generalization through the loss objectives, related to prior perceptual loss research [44]. Recent work has explored training on different latent spaces [5, 58, 69, 83], which alters the space in which flow matching operates and implicitly changes generalization behavior. Our method is effective in both the generic latent space and the pixel space.

Guidance. Guidance steers the sampling process of generative models toward a modified distribution. It can be derived from the gradient of an external classifier network [14, 34] or from an implicit classifier obtained from a pair of flow models through Bayes' rule [21, 24, 33]. Guidance has effects similar to low-temperature sampling [76], but we suspect that the improvement in sample quality also stems from the use of the explicit or implicit classifier that better captures the data manifold. Unlike guidance which can produce canonical and out-of-distribution samples [39], our method converges to the ground-truth flow and remains faithful to the original distribution. Guidance can be applied orthogonally, and our experiments show that improving the base model also improves guided results.

Divergence Measures. Flow matching, through its connection to score matching [68], minimizes forward KL divergence. GANs can minimize different divergences [54], which also influences generalization. We compare different objectives in Tab. 12 of the appendix and leave further investigation to future work.

6 Conclusion

We have introduced continuous adversarial flow models (CAFMs), a type of continuous-time flow model trained with the adversarial objective. We have empirically demonstrated that our objective can be efficiently used as a post-training method on flow-matching models and provides performance improvement on ImageNet generation and on text-conditional image generation. Our work offers exciting prospects for future research.

Acknowledgment

We thank Kunchang Li and Yuwei Guo for their valuable discussions and assistance.

References

1. Ai, Y., Han, J., Zhuang, S., Mao, W., Hu, X., Yang, Z., Yang, Z., Huang, H., Yue, X., Chen, H.: Bitdance: Scaling autoregressive generative models with binary tokens. arXiv preprint arXiv:2602.14041 (2026) 11
2. Arjovsky, M., Bottou, L.: Towards principled methods for training generative adversarial networks. In: International Conference on Learning Representations (2017) 5
3. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: International conference on machine learning. pp. 214–223. Pmlr (2017) 37
4. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016) 8, 39
5. Black Forest Labs: FLUX.2: Analyzing and enhancing the latent space of FLUX – representation comparison (2025), https://bfl.ai/research/representation-comparison 14
6. Cai, H., Cao, S., Du, R., Gao, P., Hoi, S., Hou, Z., Huang, S., Jiang, D., Jin, X., Li, L., et al.: Z-image: An efficient image generation foundation model with single-stream diffusion transformer. arXiv preprint arXiv:2511.22699 (2025) 2, 11, 27
7. Chen, R.T.Q., Lipman, Y.: Flow matching on general geometries. In: The Twelfth International Conference on Learning Representations (2024) 2, 14
8. Chen, R.T., Rubanova, Y., Bettencourt, J., Duvenaud, D.K.: Neural ordinary differential equations. Advances in neural information processing systems **31** (2018) 2
9. Chen, S., Ge, C., Zhang, S., Sun, P., Luo, P.: Pixelflow: Pixel-space generative models with flow. arXiv preprint arXiv:2504.07963 (2025) 11
10. Chen, T., Xu, B., Zhang, C., Guestrin, C.: Training deep nets with sublinear memory cost. arXiv preprint arXiv:1604.06174 (2016) 8
11. Choudhury, R., Lin, S., Wang, J., Chen, H., Zhao, Q., Cheng, F., Jiang, L., Kitani, K., Jeni, L.A.: Skipsr: Faster super resolution with token skipping. arXiv preprint arXiv:2510.08799 (2025) 14
12. De Bortoli, V., Mathieu, E., Hutchinson, M., Thornton, J., Teh, Y.W., Doucet, A.: Riemannian score-based generative modelling. Advances in neural information processing systems **35**, 2406–2422 (2022) 14
13. Deng, C., Zhu, D., Li, K., Gou, C., Li, F., Wang, Z., Zhong, S., Yu, W., Nie, X., Song, Z., et al.: Emerging properties in unified multimodal pretraining. arXiv preprint arXiv:2505.14683 (2025) 11
14. Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. Advances in neural information processing systems **34**, 8780–8794 (2021) 2, 11, 14, 20
15. Gao, Y., Guo, H., Hoang, T., Huang, W., Jiang, L., Kong, F., Li, H., Li, J., Li, L., Li, X., et al.: Seedance 1.0: Exploring the boundaries of video generation models. arXiv preprint arXiv:2506.09113 (2025) 2
16. Ghosh, D., Hajishirzi, H., Schmidt, L.: Geneval: An object-focused framework for evaluating text-to-image alignment. Advances in Neural Information Processing Systems **36**, 52132–52152 (2023) 3, 11, 12, 28

17. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014) 2, 14, 20, 21
18. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014) 2
19. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. Advances in neural information processing systems **30** (2017) 37
20. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017) 9
21. Ho, J., Salimans, T.: Classifier-free diffusion guidance. In: NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications (2021) 2, 10, 14
22. Hoogeboom, E., Heek, J., Salimans, T.: simple diffusion: End-to-end diffusion for high resolution images. In: International Conference on Machine Learning. pp. 13213–13232. PMLR (2023) 10, 11
23. Hoogeboom, E., Mensink, T., Heek, J., Lamerigts, K., Gao, R., Salimans, T.: Simpler diffusion: 1.5 fid on imagenet512 with pixel-space diffusion. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 18062–18071 (2025) 11
24. Hu, V.T., Chen, Y., Caron, M., Asano, Y.M., Snoek, C.G., Ommer, B.: Guided diffusion from self-supervised diffusion features. arXiv preprint arXiv:2312.08825 (2023) 14
25. Hu, X., Wang, R., Fang, Y., Fu, B., Cheng, P., Yu, G.: Ella: Equip diffusion models with llm for enhanced semantic alignment. arXiv preprint arXiv:2403.05135 (2024) 3, 11, 12, 28
26. Huang, N., Gokaslan, A., Kuleshov, V., Tompkin, J.: The gan is dead; long live the gan! a modern gan baseline. Advances in Neural Information Processing Systems **37**, 44177–44215 (2024) 2, 5
27. Hudson, D.A., Zitnick, L.: Generative adversarial transformers. In: International conference on machine learning. pp. 4487–4499. PMLR (2021) 5
28. Hyun, S., Lee, M., Heo, J.P.: Scalable gans with transformers. arXiv preprint arXiv:2509.24935 (2025) 2
29. Jolicoeur-Martineau, A.: The relativistic discriminator: a key element missing from standard GAN. In: International Conference on Learning Representations (2019) 5
30. Kang, M., Zhang, R., Barnes, C., Paris, S., Kwak, S., Park, J., Shechtman, E., Zhu, J.Y., Park, T.: Distilling diffusion models into conditional gans. In: European Conference on Computer Vision. pp. 428–447. Springer (2024) 14
31. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. In: International Conference on Learning Representations (2018) 5
32. Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., Aila, T.: Training generative adversarial networks with limited data. Advances in neural information processing systems **33**, 12104–12114 (2020) 5, 38
33. Karras, T., Aittala, M., Kynkäänniemi, T., Lehtinen, J., Aila, T., Laine, S.: Guiding a diffusion model with a bad version of itself. Advances in Neural Information Processing Systems **37**, 52996–53021 (2024) 2, 14
34. Kim, D., Kim, Y., Kwon, S.J., Kang, W., Moon, I.C.: Refining generative process with discriminator guidance in score-based diffusion models. In: International Conference on Machine Learning. pp. 16567–16598. PMLR (2023) 14

35. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR) (2015) 10
36. Li, S., Zhao, Y., Varma, R., Salpekar, O., Noordhuis, P., Li, T., Paszke, A., Smith, J., Vaughan, B., Damania, P., et al.: Pytorch distributed: Experiences on accelerating data parallel training. arXiv preprint arXiv:2006.15704 (2020) 8
37. Li, T., He, K.: Back to basics: Let denoising generative models denoise. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 36115–36125 (2026) 3, 9, 11
38. Lim, J.H., Ye, J.C.: Geometric gan. arXiv preprint arXiv:1705.02894 (2017) 21
39. Lin, S., Liu, B., Li, J., Yang, X.: Common diffusion noise schedules and sample steps are flawed. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. pp. 5404–5411 (2024) 14
40. Lin, S., Wang, A., Yang, X.: Sdxl-lightning: Progressive adversarial diffusion distillation. arXiv preprint arXiv:2402.13929 (2024) 2, 14
41. Lin, S., Xia, X., Ren, Y., Yang, C., Xiao, X., Jiang, L.: Diffusion adversarial post-training for one-step video generation. In: Forty-second International Conference on Machine Learning (2025) 2, 14, 27
42. Lin, S., Yang, C., He, H., Jiang, J., Ren, Y., Xia, X., Zhao, Y., Xiao, X., Jiang, L.: Autoregressive adversarial post-training for real-time interactive video generation. In: The Thirty-ninth Annual Conference on Neural Information Processing Systems (2026) 2, 14
43. Lin, S., Yang, C., Lin, Z., Chen, H., Fan, H.: Adversarial flow models. In: Forty-third International Conference on Machine Learning (2026) 2, 4, 5, 37
44. Lin, S., Yang, X.: Diffusion model with perceptual loss. arXiv preprint arXiv:2401.00110 (2023) 2, 4, 14
45. Lin, S., Yang, X.: Animatediff-lightning: Cross-model diffusion distillation. arXiv preprint arXiv:2403.12706 (2024) 2, 14
46. Lipman, Y., Chen, R.T.Q., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. In: The Eleventh International Conference on Learning Representations (2023) 1, 3
47. Liu, X., Gong, C., Liu, Q.: Flow straight and fast: Learning to generate and transfer data with rectified flow. In: The Eleventh International Conference on Learning Representations (ICLR) (2023) 3
48. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2019) 11
49. Lu, C., Song, Y.: Simplifying, stabilizing and scaling continuous-time consistency models. In: The Thirteenth International Conference on Learning Representations (2025) 8
50. Ma, N., Goldstein, M., Albergo, M.S., Boffi, N.M., Vanden-Eijnden, E., Xie, S.: Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In: European Conference on Computer Vision. pp. 23–40. Springer (2024) 3, 9, 11
51. Mao, X., Li, Q., Xie, H., Lau, R.Y., Wang, Z., Paul Smolley, S.: Least squares generative adversarial networks. In: Proceedings of the IEEE international conference on computer vision. pp. 2794–2802 (2017) 6, 20, 21, 35
52. Mathieu, E., Nickel, M.: Riemannian continuous normalizing flows. Advances in neural information processing systems **33**, 2503–2515 (2020) 14
53. Mescheder, L., Geiger, A., Nowozin, S.: Which training methods for gans do actually converge? In: International conference on machine learning. pp. 3481–3490. PMLR (2018) 37

54. Nowozin, S., Cseke, B., Tomioka, R.: f-gan: Training generative neural samplers using variational divergence minimization. Advances in neural information processing systems **29** (2016) 14
55. Oquab, M., Darcet, T., Moutakanni, T., Vo, H.V., Szafraniec, M., Khalidov, V., Fernandez, P., HAZIZA, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.Y., Li, S.W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: DINOv2: Learning robust visual features without supervision. Transactions on Machine Learning Research (2024), featured Certification 10, 11
56. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4195–4205 (2023) 11
57. Ren, Y., Xia, X., Lu, Y., Zhang, J., Wu, J., Xie, P., Wang, X., Xiao, X.: Hyper-sd: Trajectory segmented consistency model for efficient image synthesis. Advances in neural information processing systems **37**, 117340–117362 (2024) 2, 14
58. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022) 10, 14
59. Roth, K., Lucchi, A., Nowozin, S., Hofmann, T.: Stabilizing training of generative adversarial networks through regularization. Advances in neural information processing systems **30** (2017) 5, 37
60. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115**(3), 211–252 (2015) 3, 9
61. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. Advances in neural information processing systems **29** (2016) 9
62. Sauer, A., Boesel, F., Dockhorn, T., Blattmann, A., Esser, P., Rombach, R.: Fast high-resolution image synthesis with latent adversarial diffusion distillation. In: SIGGRAPH Asia 2024 Conference Papers. pp. 1–11 (2024) 2, 14
63. Sauer, A., Lorenz, D., Blattmann, A., Rombach, R.: Adversarial diffusion distillation. In: European Conference on Computer Vision. pp. 87–103. Springer (2024) 2, 14
64. Sauer, A., Schwarz, K., Geiger, A.: Stylegan-xl: Scaling stylegan to large diverse datasets. In: ACM SIGGRAPH 2022 conference proceedings. pp. 1–10 (2022) 2
65. Seawead, T., Yang, C., Lin, Z., Zhao, Y., Lin, S., Ma, Z., Guo, H., Chen, H., Qi, L., Wang, S., et al.: Seaweed-7b: Cost-effective training of video generation foundation model. arXiv preprint arXiv:2504.08685 (2025) 2
66. Seedance, T., Chen, H., Chen, S., Chen, X., Chen, Y., Chen, Y., Chen, Z., Cheng, F., Cheng, T., Cheng, X., et al.: Seedance 1.5 pro: A native audio-visual joint generation foundation model. arXiv preprint arXiv:2512.13507 (2025) 2
67. Seedream, T., Chen, Y., Gao, Y., Gong, L., Guo, M., Guo, Q., Guo, Z., Hou, X., Huang, W., Huang, Y., et al.: Seedream 4.0: Toward next-generation multimodal image generation. arXiv preprint arXiv:2509.20427 (2025) 2
68. Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. In: International Conference on Learning Representations (2021) 2, 14

69. Tong, S., Zheng, B., Wang, Z., Tang, B., Ma, N., Brown, E., Yang, J., Fergus, R., LeCun, Y., Xie, S.: Scaling text-to-image diffusion transformers with representation autoencoders. arXiv preprint arXiv:2601.16208 (2026) 14
70. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017) 8
71. Wang, F.Y., Huang, Z., Bergman, A.W., Shen, D., Gao, P., Lingelbach, M., Sun, K., Bian, W., Song, G., Liu, Y., et al.: Phased consistency models. Advances in neural information processing systems **37**, 83951–84009 (2024) 14
72. Wang, J., Lin, S., Lin, Z., Ren, Y., Wei, M., Yue, Z., Zhou, S., Chen, H., Zhao, Y., Yang, C., Xiao, X., Loy, C.C., Jiang, L.: SeedVR2: One-step video restoration via diffusion adversarial post-training. In: The Fourteenth International Conference on Learning Representations (2026) 14
73. Wang, R., He, K.: Diffuse and disperse: Image generation with representation regularization. arXiv preprint arXiv:2506.09027 (2025) 11
74. Wang, S., Gao, Z., Zhu, C., Huang, W., Wang, L.: Pixnerd: Pixel neural field diffusion. In: The Fourteenth International Conference on Learning Representations (2026) 11
75. Wang, S., Tian, Z., Huang, W., Wang, L.: Ddt: Decoupled diffusion transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 40633–40642 (2026) 11
76. Xu, Y., Wu, Y., Park, S., Zhou, Z., Tulsiani, S.: Temporal score rescaling for temperature sampling in diffusion and flow models. In: Forty-third International Conference on Machine Learning (2026) 14
77. Xu, Y., Zhao, Y., Xiao, Z., Hou, T.: Ufogen: You forward once large scale text-to-image generation via diffusion gans. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8196–8206 (2024) 2, 14
78. Yin, T., Gharbi, M., Park, T., Zhang, R., Shechtman, E., Durand, F., Freeman, B.: Improved distribution matching distillation for fast image synthesis. Advances in neural information processing systems **37**, 47455–47487 (2024) 2, 14
79. Yu, S., Kwak, S., Jang, H., Jeong, J., Huang, J., Shin, J., Xie, S.: Representation alignment for generation: Training diffusion transformers is easier than you think. In: The Thirteenth International Conference on Learning Representations (2025) 11
80. Zhang, B., Sennrich, R.: Root mean square layer normalization. Advances in neural information processing systems **32** (2019) 8, 39
81. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 586–595 (2018) 2, 4
82. Zhao, Y., Gu, A., Varma, R., Luo, L., Huang, C.C., Xu, M., Wright, L., Shojanazeri, H., Ott, M., Shleifer, S., et al.: Pytorch fsdp: experiences on scaling fully sharded data parallel. arXiv preprint arXiv:2304.11277 (2023) 8
83. Zheng, B., Ma, N., Tong, S., Xie, S.: Diffusion transformers with representation autoencoders. In: The Fourteenth International Conference on Learning Representations (2026) 2, 14
84. Zheng, K., Wang, Y., Ma, Q., Chen, H., Zhang, J., Balaji, Y., Chen, J., Liu, M.Y., Zhu, J., Zhang, Q.: Large scale diffusion distillation via score-regularized continuous-time consistency. In: The Fourteenth International Conference on Learning Representations (2026) 39
85. Zhou, L., Parger, M., Haque, A., Song, J.: Terminal velocity matching. In: The Fourteenth International Conference on Learning Representations (2026) 8, 39

A Additional Results on ImageNet Post-training

Table 7 provides the full evaluation metrics of SiT-XL/2 following ADM [14].

Table 7: SiT-XL/2 full metrics on ImageNet 256px.

CFG	Method	Epoch	FID↓	IS↑	sFID ↓	Prec.↑	Rec↑
None	SiT	1400	8.26	131.65	6.32	0.68	0.67
	SiT+FM	1400+10	8.64	131.91	6.36	0.68	0.67
	SiT+CAFM	1400+10	**3.63**	**178.08**	**4.72**	**0.71**	**0.69**
1.1	SiT	1400	5.55	161.77	5.67	0.72	0.66
	SiT+FM	1400+10	5.53	161.72	5.68	0.72	0.65
	SiT+CAFM	1400+10	**2.27**	**212.06**	**4.55**	**0.74**	**0.67**
1.2	SiT	1400	3.65	190.57	5.19	0.75	0.64
	SiT+FM	1400+10	3.62	191.56	5.19	0.75	0.64
	SiT+CAFM	1400+10	**1.66**	**238.44**	**4.49**	**0.77**	**0.66**
1.3	SiT	1400	2.57	220.52	4.83	**0.78**	0.63
	SiT+FM	1400+10	2.55	219.89	4.85	0.78	0.62
	SiT+CAFM	1400+10	**1.53**	**263.52**	**4.59**	**0.78**	**0.64**
1.4	SiT	1400	2.07	248.31	**4.61**	**0.80**	0.61
	SiT+FM	1400+10	2.09	247.98	4.63	**0.80**	0.61
	SiT+CAFM	1400+10	**1.66**	**283.59**	4.75	0.79	**0.63**
1.5	SiT	1400	2.06	277.50	**4.49**	**0.83**	0.59
	SiT+FM	1400+10	2.02	270.94	4.53	0.82	0.59
	SiT+CAFM	1400+10	**1.97**	**301.91**	5.03	0.81	**0.62**
1.6	SiT	1400	**2.25**	293.72	**4.51**	**0.84**	0.58
	SiT+FM	1400+10	2.26	292.53	4.53	**0.84**	0.57
	SiT+CAFM	1400+10	2.37	**316.78**	5.35	0.81	0.60

Table 8 shows that $N = 8$ leads to divergence, while $N = 32$ leads to slower learning, so we pick $N = 16$. Table 9 shows that $\lambda_{\text{ot}} = 0$ yields the best result for post-training. Table 10 shows that increasing the learning rate for both G, D causes grad norm spikes and divergence. Table 11 shows that FID stays the same after training longer. Table 12 shows that the least square loss [51] produces stronger results than non-saturating loss [17].

Table 8: SiT-XL/2 CAFM post-train ablation study on N.

N	8	16	32
FID↓	294.91	**3.63**	3.68

Table 9: SiT-XL/2 CAFM post-train ablation study on λ_{ot}.

λ_{ot}	0	0.01
FID↓	**3.63**	4.50

Table 10: SiT-XL/2 CAFM post-train ablation study on learning rate.

LR	1e-5	5e-5
FID↓	**3.63**	283.96

Table 11: SiT-XL/2 CAFM post-train ablation study on epochs.

Epoch	10	20
FID↓	**3.63**	3.64

Table 12: SiT-XL/2 CAFM post-train ablation study on $f(a, b)$.
Non-saturating [17]: $f(a, b) = -\log(\sigma(a)) - \log(1 - \sigma(b))$.
Hinge [38]: $f^D(a, b) = \max(0, 1 - a) + \max(0, 1 + b)$, $f^G(a, b) = -a + b$.
Least squares [51]: $f(a, b) = (a - 1)^2 + (b + 1)^2$.
CFG is swept separately and the best result is reported for each.

CFG	$f(a, b)$	Epoch	FID↓	IS↑	sFID ↓	Prec.↑	Rec↑
None	Non-saturating	1400+10	**3.54**	167.45	5.20	**0.72**	0.68
	Hinge	1400+10	4.00	175.21	5.36	0.71	0.68
	Least squares	1400+10	3.63	**178.08**	**4.72**	0.71	**0.69**
1.3	Non-saturating	1400+10	1.58	245.10	4.84	**0.79**	**0.64**
	Hinge	1400+10	1.57	258.46	4.86	**0.79**	**0.64**
	Least squares	1400+10	**1.53**	**263.52**	**4.59**	0.78	**0.64**

Table 13 shows the full evaluation metrics of JiT-H/16. The FID and IS metrics are computed using the code provided by JiT, and the other metrics are computed using the code provided by ADM. The JiT settings follow those of SiT, and we do not conduct separate ablation studies on JiT.

Table 13: JiT-H/16 full metrics on ImageNet 256px.

CFG	Method	Epoch	FID↓	IS↑	sFID ↓	Prec.↑	Rec↑
None	JiT	600	7.17	151.54	5.51	0.68	**0.67**
	JiT+FM	600+10	9.30	139.00	6.16	0.67	0.66
	JiT+CAFM	600+10	**3.57**	**198.08**	**4.77**	**0.74**	0.65
1.2	JiT	600	4.60	188.88	5.34	0.71	**0.66**
	JiT+FM	600+10	5.97	176.99	5.88	0.71	0.65
	JiT+CAFM	600+10	**2.47**	**232.69**	**4.76**	**0.76**	0.64
1.4	JiT	600	3.24	219.52	5.25	0.74	**0.66**
	JiT+FM	600+10	4.07	209.78	5.66	0.74	0.64
	JiT+CAFM	600+10	**2.01**	**258.46**	**4.81**	**0.77**	0.64
1.6	JiT	600	2.49	244.61	5.22	0.76	**0.65**
	JiT+FM	600+10	3.01	237.46	5.51	0.75	0.63
	JiT+CAFM	600+10	**1.84**	**275.96**	**4.91**	**0.78**	0.63
1.8	JiT	600	2.12	265.43	5.20	0.77	**0.65**
	JiT+FM	600+10	2.43	261.26	5.41	0.77	0.63
	JiT+CAFM	600+10	**1.80**	**290.71**	**5.03**	**0.79**	0.63
2.0	JiT	600	1.96	281.38	5.22	0.78	**0.64**
	JiT+FM	600+10	2.12	280.24	5.34	0.78	0.62
	JiT+CAFM	600+10	**1.83**	**301.96**	**5.16**	**0.79**	0.63
2.2	JiT	600	**1.86**	303.40	**5.27**	0.78	**0.64**
	JiT+FM	600+10	1.98	296.68	5.31	0.79	0.62
	JiT+CAFM	600+10	1.88	**310.54**	**5.27**	**0.80**	0.62
2.4	JiT	600	2.19	310.20	5.32	0.79	**0.63**
	JiT+FM	600+10	**1.94**	310.06	**5.29**	**0.80**	0.61
	JiT+CAFM	600+10	1.95	**319.23**	5.40	0.79	0.62

Table 14 enumerates the hyperparameters used for our CAFM post-training.

Table 14: ImageNet CAFM post-training hyperparameters
*We formulate x_0 as image while JiT formulates x_0 as noise, so the timestep is reversed in writing.

	SiT	JiT
G, D learning rate	1e-5	1e-5
batch size	256	1024
total epoch	10	10
D warm-up epoch	2	4
timesteps	uniform(0, 1)	lognormal(0.8, 0.8)*
CFG interval	[0, 1]	[0, 0.9]*
Sampler	SDE 250 steps	Heun 50 steps
Adam β	(0.0, 0.95)	
weight decay	0	
N	16	
λ_{ot}	0	
EMA decay	0.99	
precision	TF32	

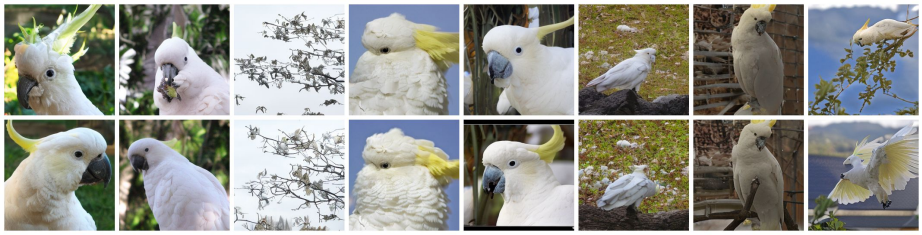
Qualitative comparisons are provided in Figs. 6 to 9 in the following pages.



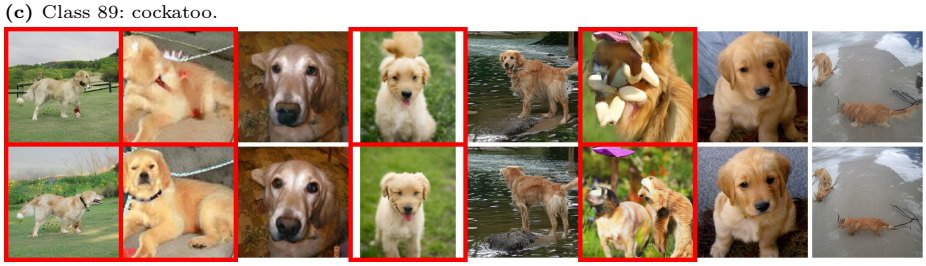
(a) Class 0: tench.



(b) Class 3: tiger shark.



(c) Class 89: cockatoo.



(d) Class 207: golden retriever.

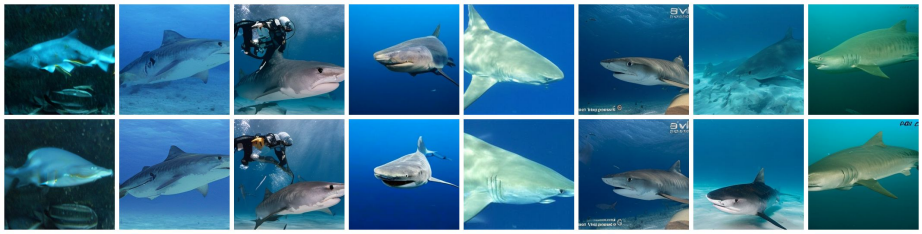


(e) Class 279: white fox.

Fig. 6: SiT-XL/2 guidance-free, latent-space ImageNet 256px generation.
Top is FM (FID 8.26). Bottom is CAFM (FID 3.63).
Uncurated. We highlight samples with visible improvements in red.



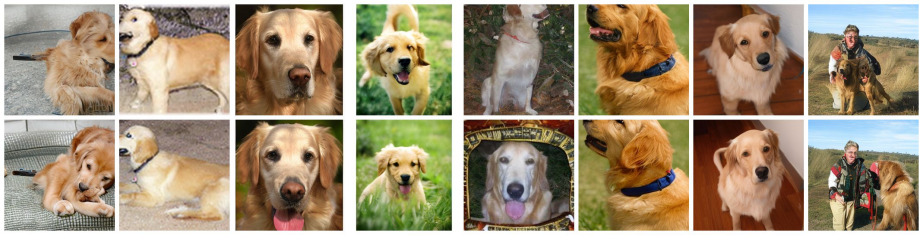
(a) Class 0: tench.



(b) Class 3: tiger shark.



(c) Class 89: cockatoo.



(d) Class 207: golden retriever.



(e) Class 279: white fox.

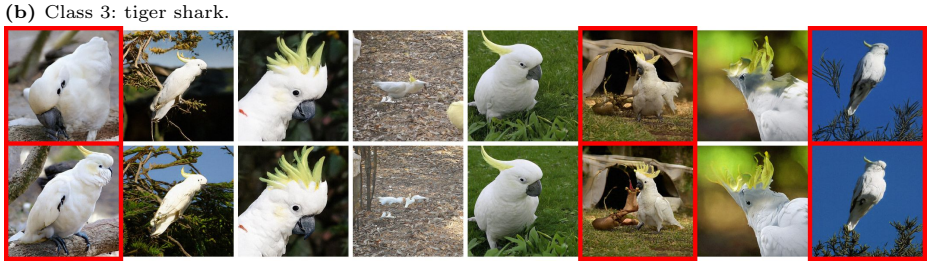
Fig. 7: SiT-XL/2 guided, latent-space ImageNet 256px generation.
Top is FM (CFG 1.5, FID 2.06). Bottom is CAFM (CFG 1.3, FID 1.53).
Uncurated.



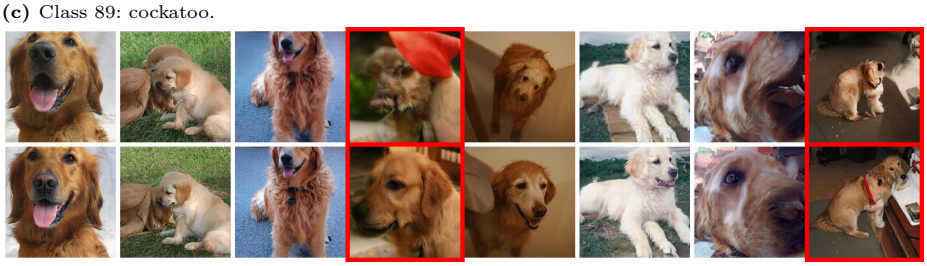
(a) Class 0: tench.



(b) Class 3: tiger shark.



(c) Class 89: cockatoo.



(d) Class 207: golden retriever.



(e) Class 279: white fox.

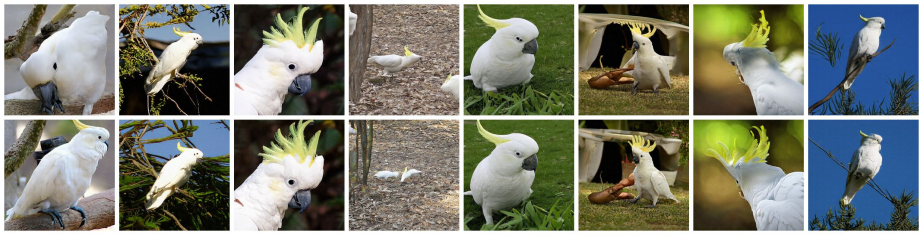
Fig. 8: JiT-H/16 guidance-free, pixel-space ImageNet 256px generation.
Top is FM (FID 7.17). Bottom is CAFM (FID 3.57).
Uncurated. We highlight samples with visible improvements in red.



(a) Class 0: tench.



(b) Class 3: tiger shark.



(c) Class 89: cockatoo.



(d) Class 207: golden retriever.



(e) Class 279: white fox.

Fig. 9: JiT-H/16 guided, pixel-space ImageNet 256px generation.
Top is FM (CFG2.2, FID 1.86). Bottom is CAFM (CFG1.8, FID 1.80).
Uncurated.

B Additional Results on Text-to-Image Post-training

Architecture. Our text-to-image generation experiments are conducted on Z-Image [6] model, an open-source, 6B-parameter, single-stream diffusion transformer. We use the pre-distillation checkpoint, which is suitable for our continuous flow experiments. For the generator, we adopt the exact architecture without changes. For the discriminator, we follow APT [41] to add a cross-attention layer on the visual features at the last layer to project the discriminator logit. Compared to inserting [CLS] at input, this design allows most parts of the transformer to stay intact. Because Z-Image already uses RMSNorm, no changes are made to the normalization layers.

Dataset and training. Z-Image is trained on proprietary supervised finetuning (SFT) data, which are inaccessible to us. Also, the SFT data likely contain high-quality images generated by prior text-to-image models using CFG, which is implicitly a form of CFG distillation. For our experiments, we use open-source image datasets that contain only natural images, filtered and recaptioned. To eliminate the dataset from being an influencing factor, we first finetune Z-Image on our data using the FM objective for 10k iterations, and we find that the model quickly adapts. Then we run CAFM finetuning, while keeping the FM trial running with equivalent iterations for comparison. The CAFM finetuning is run for a total of 20k iterations including both G, D updates. The hyperparameters are listed in Tab. 15.

Table 15: Text-to-image CAFM post-training hyperparameters

	FM	CAFM
G learning rate	5e-5	5e-5
D learning rate		3e-5
N		16
λ_{ot}		0
AdamW β	(0.9, 0.95)	(0, 0.95)
iterations	30k	FM10k+20k
resolution	512px	
batch size	1024	
weight decay	0.01	
timesteps	uniform(0,1), shift 3	
sampler	Euler 50 steps, shift 6	
CFG dropout	0.1	
CFG scale	4 when used	
precision	BF16	

Additional results. Tables 16 and 17 shows the metrics including the original Z-Image model for reference. In GenEval, our CAFM-finetuned model beats both the original and our FM-finetuned baseline. But in the DPG benchmark, our model performs worse than the original ZImage model, which we believe is due to the use of different datasets. These tables are provided only for reference. Only the FM-finetuned model is the fair comparison baseline, and for this reason, we removed the original Z-Image model from the tables in the main text. More qualitative comparisons are provided in Fig. 10.

Table 16: GenEval [16] on 512px T2I generation including original ZImage.

*Trained on different datasets. Not directly comparable.

Method	PE	CFG	Single Obj.	Two Obj.	Color Attr.	Position	Counting	Colors.	Overall
ZImage			0.78	0.37	0.16	0.11	0.32	0.59	0.39*
ZImage+FM	No	No	0.72	0.23	0.11	0.09	0.25	0.59	0.33
ZImage+CAFM			0.85	0.42	0.17	0.16	0.41	0.61	0.44
ZImage			0.96	0.83	0.43	0.47	0.51	0.85	0.68*
ZImage+FM	Yes	No	0.95	0.66	0.35	0.40	0.42	0.81	0.60
ZImage+CAFM			0.99	0.83	0.50	0.52	0.57	0.86	0.71
ZImage			0.98	0.95	0.70	0.67	0.67	0.95	0.82*
ZImage+FM	Yes	Yes	0.99	0.89	0.62	0.69	0.77	0.89	0.81
ZImage+CAFM			0.99	0.92	0.71	0.71	0.81	0.94	0.85

Table 17: DPG-Bench [25] on 512px T2I generation including original ZImage.

*Trained on different datasets. Not directly comparable.

Method	CFG	Global	Entity	Attribute	Relation	Other	Overall
ZImage		87.27	87.22	86.44	88.97	89.21	79.83*
ZImage+FM	No	81.34	82.96	81.71	83.17	85.07	72.25
ZImage+CAFM		87.82	86.65	86.33	86.49	84.85	77.21
ZImage		90.55	91.71	91.49	92.43	87.94	86.35*
ZImage+FM	Yes	90.34	90.56	88.98	88.17	90.71	83.67
ZImage+CAFM		89.55	89.83	89.99	91.20	91.88	85.21



(a) DPG prompt 0: Eight cabbages in a dewy morning field



(b) DPG prompt 1: A red pickup truck parked on a beach at dusk, palm trees in the back



(c) DPG prompt 2: A bathroom with a white rectangular bathtub full of bubbles



(d) DPG prompt 3: An antique mahogany desk with a detailed globe and vintage pens



(e) DPG prompt 5: A calculator on a wooden desk with scattered papers

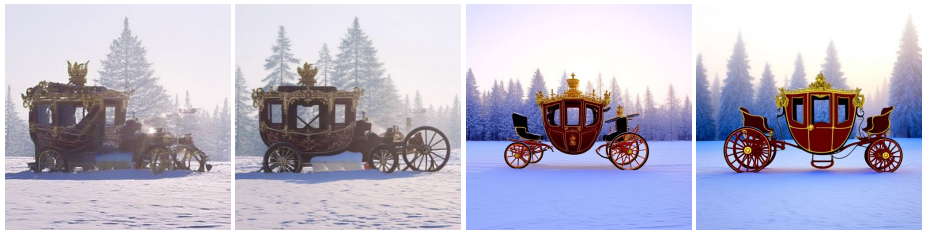
Fig. 10: Curated text-to-image comparisons on DPG benchmark prompts. Prompts are shortened for paper presentation.
(part 1 of 4)



(f) DPG prompt 8: A red hoverboard on a city street at sunset, surrounded by tall buildings



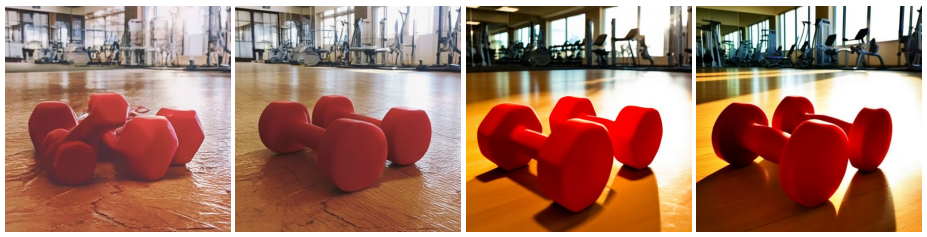
(g) DPG prompt 10: Three purple eggplants on a rustic wooden table with a napkin



(h) DPG prompt 20: A red and gold royal carriage in a snowy landscape with pine trees

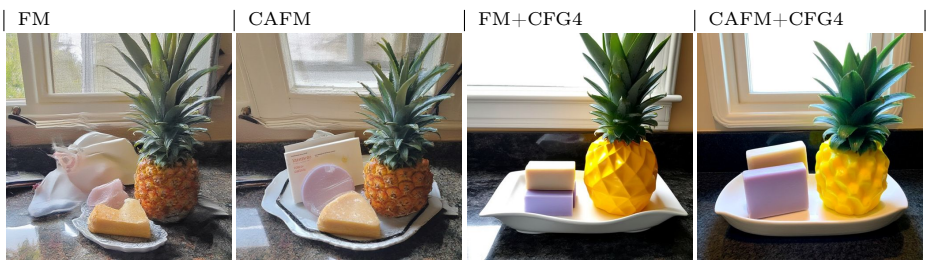


(i) DPG prompt 40: Two black motorcycle helmets hanging on a white wall with tools



(j) DPG prompt 50: Three red dumbbells on a wooden gym floor

Fig. 10: Curated text-to-image comparisons on DPG benchmark prompts. Prompts are shortened for paper presentation.
(part 2 of 4)



(k) DPG prompt 80: A lavender and an oatmeal soap beside a yellow pineapple on a white dish



(l) DPG prompt 100: A white desk with beauty products next to a handgun on the floor



(m) DPG prompt 120: A musician playing a recorder in a quiet room, lit by a glowing lantern



(n) DPG prompt 140: Three folded red towels on concrete beside a white scooter under sky



(o) DPG prompt 160: A silver sailboat at twilight with someone cooking noodles on deck

Fig. 10: Curated text-to-image comparisons on DPG benchmark prompts. Prompts are shortened for paper presentation.
(part 3 of 4)



(p) DPG prompt 170: Three green high heels in a storefront display. A mop leans on the window



(q) DPG prompt 200: A showerhead dripping onto a urinal in a restroom with grey dividers



(r) DPG prompt 220: Red, yellow, and blue billiard balls rolling on a table beside curling stones



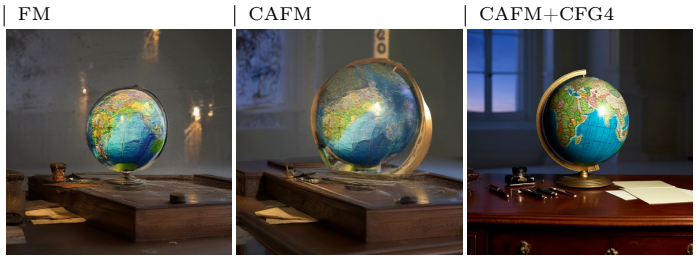
(s) DPG prompt 230: A pink lipstick and two sparkling necklaces on a dark wooden dresser



(t) DPG prompt 250: A busy airport terminal with windows showing airplanes, seating areas, flight displays, and black surveillance cameras overhead.

Fig. 10: Curated text-to-image comparisons on DPG benchmark prompts. Prompts are shortened for paper presentation.
(part 4 of 4)

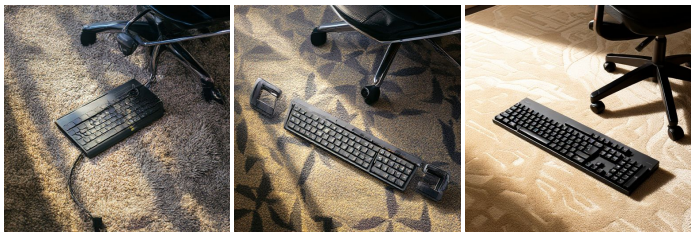
Limitation. Despite CAFM improving the generalization of the model, Figure 11 shows that guidance-free generation can still yield incorrect images sometimes, especially in the low-density regions containing outliers. Our research does not claim that our method can reach production quality in the guidance-free setting, but only to demonstrate that our method can improve the generalization of the model, which yields gains in both the guidance-free and guided settings.



(a) DPG prompt 3: An antique mahogany desk with a detailed globe and vintage pens



(b) DPG prompt 4: Four pens forming a rectangle on a beige desk, with five pencils arranged in a circle at the center



(c) DPG prompt 13: A black keyboard resting diagonally on a beige carpet in a sunlit home office, with a nearby office chair.



(d) DPG prompt 15: An ornate silver makeup mirror on a white marble vanity surrounded by cosmetics and perfume bottles in natural light.

Fig. 11: Failure cases for guidance-free text-to-image generation.

C On Flow Matching Objective

This section shows that criteria other than the squared L_2 metric can also be valid for flow matching.

Consider the criterion:

$$d(a, b) = (a - b)^\top M(a - b), \tag{27}$$

where the squared L_2 criterion corresponds to the special case $M = I$.

We show that any strictly positive definite (SPD) matrix $M \in \mathbb{R}^{n\times n}$, defined by:

$$d(a, b) = (a - b)^\top M(a - b) > 0, \quad \forall a \neq b, \tag{28}$$

also satisfies:

$$\arg\min_a \mathbb{E}_b[d(a, b)] = \mathbb{E}[b], \tag{29}$$

and hence converges to the marginal velocity $v_t = \mathbb{E}[\bar{v}_t \mid x_t]$ under the conditional flow matching objective:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{x,z,t}[d(G(x_t, t), \bar{v}_t)]. \tag{30}$$

First, we expand the expectation, where the expectation is taken over b:

$$\mathbb{E}[d(a, b)] = \mathbb{E}[(a - b)^\top M(a - b)] \tag{31}$$

$$= a^\top Ma - 2a^\top M\mathbb{E}[b] + \mathbb{E}[b^\top Mb]. \tag{32}$$

Then, we take the derivative with respect to a. The last term, $\mathbb{E}[b^\top Mb]$, is constant with respect to a and is therefore dropped:

$$\nabla_a\mathbb{E}[d(a, b)] = 2Ma - 2M\mathbb{E}[b]. \tag{33}$$

Finally, we set the gradient to zero:

$$2Ma - 2M\mathbb{E}[b] = 0, \tag{34}$$

$$M(a - \mathbb{E}[b]) = 0. \tag{35}$$

Since M is strictly positive definite, it has no zero eigenvalues, so the only solution is:

$$a - \mathbb{E}[b] = 0, \tag{36}$$

$$a = \mathbb{E}[b]. \tag{37}$$

This concludes that the minimizer in a is exactly $\mathbb{E}[b]$, just as in the standard L_2 case. In theory, the choice of strictly positive definite M is irrelevant, since all such choices converge to the ground-truth marginal velocity. In practice, however, different choices of M can affect model generalization under finite capacity.

D On Discriminator JVP Designs

One may ask why we formulate D using JVP instead of simply defining it as:

$$D(x_t, t, v_t) : (\mathbb{R}^n \times [0,1] \times \mathbb{R}^n) \to \mathbb{R}, \tag{38}$$

and define the adversarial training objectives as follows:

$$\mathcal{L}^D_{\text{adv}'} = \mathbb{E}_{x,z,t}\left[f\big(D(x_t,t,v_t), D(x_t,t,G(x_t,t))\big)\right], \tag{39}$$

$$\mathcal{L}^G_{\text{adv}'} = \mathbb{E}_{x,z,t}\left[f\big(D(x_t,t,G(x_t,t)), D(x_t,t,v_t)\big)\right]. \tag{40}$$

This naive formulation has at least two fundamental problems:

First, the ground-truth marginal velocity v_t used in Eqs. (39) and (40) is generally inaccessible during training. If we replace it with the conditional velocity $\bar{v}_t$, the objective no longer enforces learning of the true marginal velocity field. The key issue is that D is nonlinear, so in general

$$\mathbb{E}[D(\bar{v})] \neq D(\mathbb{E}[\bar{v}]). \tag{41}$$

Therefore, matching discriminator responses to conditional targets does not imply matching the marginal target. At equilibrium, G would need to represent the full conditional-velocity distribution at each x_t; however, for a fixed (x_t, t), the generator outputs only a single deterministic velocity. Consequently, G is repeatedly pulled toward incompatible conditional targets $\bar{v}_t$, leading to oscillatory updates.

Second, even in the idealized setting where the marginal velocity v_t were accessible, this direct formulation can suffer from vanishing or uninformative gradients. The real target at each x_t is effectively a point mass (Dirac-like) in velocity space. When the supports overlap poorly, the discriminator can separate real and generated samples too easily, saturate, and provide weak learning signals to G.

Our work instead formulates the discriminator as $D(x_t, t)$ and performs discrimination in JVP space:

$$D_{\text{jvp}}(x_t, t, \dot{x}_t, \dot{t}) = \frac{\partial D(x_t,t)}{\partial x_t}\dot{x}_t + \frac{\partial D(x_t,t)}{\partial t}\dot{t}, \tag{42}$$

with adversarial objectives:

$$\mathcal{L}^D_{\text{adv}'} = \mathbb{E}_{x,z,t}[f(D_{\text{jvp}}(x_t,t,\bar{v}_t,T), D_{\text{jvp}}(x_t,t,G(x_t,t),T))], \tag{43}$$

$$\mathcal{L}^G_{\text{adv}'} = \mathbb{E}_{x,z,t}[f(D_{\text{jvp}}(x_t,t,G(x_t,t),T), D_{\text{jvp}}(x_t,t,\bar{v}_t,T))], \tag{44}$$

where

$$f(a,b) = (a-1)^2 + (b+1)^2 \tag{45}$$

is an LSGAN-like [51] contrastive function.

This encourages:

$$D_{\text{jvp}}(x_t,t,v_t,1) = +1, \tag{46}$$

$$D_{\text{jvp}}(x_t,t,G(x_t,t),1) = -1. \tag{47}$$

By Eq. (62) and linearity of expectation,

$$\mathbb{E}[D_{\text{jvp}}(x_t, t, \bar{v}_t, 1) - D_{\text{jvp}}(x_t, t, G(x_t, t), 1)] \tag{48}$$

$$= \mathbb{E}\left[\frac{\partial D(x_t, t)}{\partial x_t}(\bar{v}_t - G(x_t, t))\right] \tag{49}$$

$$= \frac{\partial D(x_t, t)}{\partial x_t}(\mathbb{E}[\bar{v}_t] - G(x_t, t)) \tag{50}$$

$$= \frac{\partial D(x_t, t)}{\partial x_t}(v_t - G(x_t, t)). \tag{51}$$

When $G(x_t, t) \neq v_t$, D_{jvp} can be optimized toward Eqs. (46) and (47). Equilibrium is reached only when $G(x_t, t) = v_t$ for all x_t, in which case

$$\frac{\partial D(x_t, t)}{\partial x_t}(v_t - G(x_t, t)) = 0. \tag{52}$$

However, the above derivation relies only on the linearity of JVP. An alternative is to parameterize D using separate networks:

$$D(x_t, t, v_t) = A(x_t, t)^\top v_t + B(x_t, t), \tag{53}$$

where

$$A(x_t, t) : \mathbb{R}^n \times [0, 1] \to \mathbb{R}^n, \tag{54}$$

$$B(x_t, t) : \mathbb{R}^n \times [0, 1] \to \mathbb{R}^1. \tag{55}$$

This approach differs from JVP because JVP additionally enforces:

$$A(x_t, t) = \nabla_x D(x_t, t), \tag{56}$$

$$B(x_t, t) = \partial_t D(x_t, t). \tag{57}$$

Because of these constraints,

$$D(x_1, 1) - D(x_0, 0) = \int_0^1 D_{\text{jvp}}(x_t, t, v_t, 1)\ dt. \tag{58}$$

The discriminator is globally consistent along trajectories. Empirically, we find that these constraints improve optimization, and the separate formulation does not yield good results on high-dimensional data.

E On the Vanishing-Gradient Problem

The vanishing-gradient problem in adversarial training arises when the real data distribution and the generator distribution have non-overlapping support. Formally, the support of a distribution $p(x)$ is the set on which it assigns positive probability mass:

$$\operatorname{supp}(p) = \{x \mid p(x) > 0\}. \tag{59}$$

In high-dimensional spaces, data distributions (e.g., images) often concentrate near a tiny, low-dimensional manifold. When the support of the real data distribution S_{data} and that of the generator distribution S_G do not overlap,

$$S_{\text{data}} \cap S_G = \emptyset, \tag{60}$$

an optimal discriminator can separate the two distributions perfectly. As a result, it forms a steep decision boundary in regions between the supports while producing nearly flat gradients around generated samples, yielding little useful learning signal for the generator.

There are two approaches to theoretically mitigating the vanishing-gradient issue. WGAN [3] proposes using the Wasserstein-1 distance, but it requires the discriminator network to be 1-Lipschitz. Enforcing this condition is difficult in practice, so it is often replaced by gradient penalties [19, 59], which provide a softer constraint at the cost of weaker theoretical guarantees. Another approach is to add instance noise [53]:

$$\tilde{x} = x + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I}). \tag{61}$$

This is equivalent to convolving the data distribution with a Gaussian distribution, which has support everywhere. Therefore, the resulting distribution also has support everywhere.

Although $\sigma > 0$ can be arbitrarily small and still ensures that the convolved distribution has support everywhere in theory, the off-manifold region carries only tiny probability mass. In practice, the discriminator can still form sharp decision boundaries in these regions. Increasing σ further mitigates gradient vanishing, but it also causes the model to learn a noisy distribution rather than the intended data distribution. The optimal σ depends on the distance between the real and generator manifolds and on the capacity of the discriminator.

Adversarial flow models [43] allow piecewise learning on the probability flow $G(x_s, s, t) \to x_t$. The vanishing-gradient problem should be further mitigated as the discretization interval $|t - s|$ decreases. Continuous adversarial flow models extend this formulation to continuous time, where $|t - s| \to 0$. From this perspective, they should mitigate the vanishing-gradient problem.

From a mathematical perspective, we also show that, for CAFMs, the gradient with respect to G does not vanish under an optimal D because of the linearization effect of JVP. Specifically, let

$$D_{\text{jvp}}(x_t, t, G(x_t, t), 1) = J_x(x_t, t)G(x_t, t) + J_t(x_t, t), \tag{62}$$

where

$$J_x(x_t, t) = \frac{\partial D(x_t, t)}{\partial x_t}, \quad J_t(x_t, t) = \frac{\partial D(x_t, t)}{\partial t} \tag{63}$$

are the Jacobian matrices of the discriminator network. In backpropagation, the gradient with respect to $G(x_t, t)$ is simply

$$\frac{\partial \mathcal{L}}{\partial G(x_t, t)} = J_x(x_t, t)^\top g, \tag{64}$$

where

$$g := \frac{\partial \mathcal{L}}{\partial D_{\text{jvp}}} \tag{65}$$

is the gradient propagated from the loss.

Therefore, $G(x_t, t)$ receives a nonzero gradient whenever $g \neq 0$ and $J_x(x_t, t) \neq 0$. We previously showed in Sec. D that whenever $G(x_t, t) \neq v_t$, an optimal D learns a nonzero $J_x(x_t, t)$ for discrimination.

For the least squares contrastive function

$$f(a, b) = (a - 1)^2 + (b + 1)^2. \tag{66}$$

The derivative with respect to a is:

$$\frac{df(a, b)}{da} = 2(a - 1). \tag{67}$$

Therefore, it has a non-zero gradient with respect to input a for any $a \neq 1$. In the case of an optimal D, it outputs -1 for the v_t prediction by G.

In practice, we also find that our method can train properly without gradient penalties and discriminator augmentation [32].

F On Implementation of JVP

Our experiment is conducted in PyTorch, where we use `torch.func.jvp` and `torch.func.vmap` for forward-mode Jacobian-vector product (JVP) and vectorizing map (Vmap). Both functions are compatible with DDP, FSDP, and gradient checkpointing with a special arrangement.

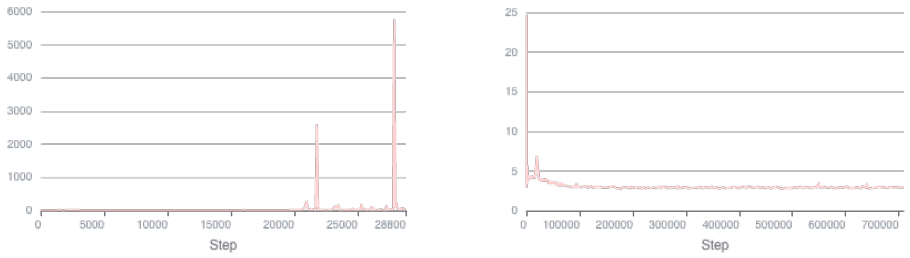
Specifically, it is important to implement it as `ddp(jvp(D))`, instead of `jvp(ddp(D))`, so that JVP (and Vmap) is only wrapped around the network as a regular PyTorch module, instead of applying JVP to DDP, which includes incompatible gradient synchronization logic. DDP is used for the ImageNet experiments.

For FSDP and gradient checkpointing, an example implementation can be found in prior work, rCM [84]. rCM wraps JVP on every `nn.Module`, which we find unnecessary and excessive. It is sufficient to only wrap JVP (and Vmap) to the top-level submodules for FSDP sharding and gradient checkpointing. FSDP and gradient checkpointing are used for the text-to-image generation experiments.

For attention, we use the `math` fused kernel of PyTorch's scaled dot product attention, which supports both JVP and Vmap natively and is sufficient for image generation training.

G On LayerNorm and RMSNorm

We empirically find that switching the discriminator LayerNorm [4] to RMSNorm [80] significantly improves training stability. Figure 12 shows the discriminator gradient norm when pre-training the SiT-B/2 model on ImageNet 256px. LayerNorm causes large spikes in discriminator gradient norm, whereas RMSNorm does not under equal settings. Prior work involving JVP has also found that RMSNorm provides better stability [85].



(a) LayerNorm **(b)** RMSNorm

Fig. 12: Discriminator gradient norm.

H On Computational Efficiency

Discriminator update count N. Our method performs N discriminator updates for each generator update, with the goal of keeping the discriminator close to its local optimum throughout training. Importantly, this schedule does *not* imply that the generator converges N times more slowly than in standard flow matching. In classical flow matching, the model estimates the marginal velocity field by taking an expectation over Monte Carlo samples of conditional velocities. Analogously, in our framework, the discriminator estimates flow potentials using Monte Carlo samples of conditional velocities. The key distinction is that the generator is then updated using these learned potentials, moving in the direction that increases the potential most strongly. This coupling enables stable and informative generator updates even when discriminator optimization is emphasized.

JVP computation efficiency. For ImageNet SiT-XL/2 post-training, CAFM requires approximately 4.8× more wall-clock time per epoch than FM. This overhead arises from introducing an additional discriminator network, along with its forward pass and backward JVP computations. We consider this additional computation acceptable for post-training applications.