1.x-Distill: Breaking the Diversity, Quality, and Efficiency Barrier in Distribution Matching Distillation

Haoyu Li[1⋆], Tingyan Wen[1⋆], Lin Qi[2⋆⋆], Zhe Wu[2], Yihuang Chen[2], Xing Zhou[2], Lifei Zhu[2], XueQian Wang[1], and Kai Zhang[1⋆⋆]

[1] Tsinghua University
[2] Central Media Technology Institute, Huawei

https://thu-accdiff.github.io/1.x-distill-page/

Abstract. Diffusion models produce high-quality text-to-image results, but their iterative denoising is computationally expensive. Distribution Matching Distillation (DMD) emerges as a promising path to few-step distillation, but suffers from diversity collapse and fidelity degradation when reduced to two steps or fewer. We present ***1.x-Distill***, the first fractional-step distillation framework that breaks the integer-step constraint of prior few-step methods and establishes 1.x-step generation as a practical regime for distilled diffusion models. Specifically, we first analyze the overlooked role of teacher CFG in DMD and introduce a simple yet effective modification to suppress mode collapse. Then, to improve performance under extreme steps, we introduce *Stagewise Focused Distillation*, a two-stage strategy that learns coarse structure through diversity-preserving distribution matching and refines details with inference-consistent adversarial distillation. Furthermore, we design a lightweight compensation module for *Distill–Cache co-Training*, which naturally incorporates block-level caching into our distillation pipeline. Experiments on SD3-Medium and SD3.5-Large show that ***1.x-Distill*** surpasses prior few-step methods, achieving better quality and diversity at 1.67 and 1.74 effective NFEs, respectively, with up to **33**× speedup over original 28×2 NFE sampling.

Keywords: Diffusion models· Text-to-image generation· Distribution matching distillation

1 Introduction

Diffusion models [4, 5, 11, 32, 34] have become the dominant paradigm at high-resolution image generation, but their iterative sampling steps leads to high computational cost. To mitigate this issue, recent research has actively explored step distillation [7, 33, 36–38, 41, 49, 50], which distill a multi-step pretrained

⋆ Equal Contribution
⋆⋆ Corresponding Author

Fig. 1: Visual results. ***1.x-Distill*** mitigates the mode collapse and quality degradation of vanilla DMD under extreme step reduction, delivering superior few-step results.

diffusion model into a few-step generator. Among them, Distribution matching distillation (DMD) [49, 50] reduces the student's sampling to a few steps by matching the output distribution of the teacher, and have demonstrated strong effectiveness on large-scale models.

However, as shown in the left panel of Fig. 1, existing distribution matching methods [6, 29, 49] face two major bottlenecks when pushed to two-step or fewer sampling. (1) Compared to trajectory-based distillation [7, 27, 52], DMD series suffer from severe diversity degradation. (2) Extreme step reduction forces each denoising step to carry more semantic and visual responsibility, which leads to pronounced quality degradation in the generated images.

While *mode collapse* in DMD is often attributed to the reverse KL formulation [26, 52], we provide a complementary perspective by analyzing the role of Classifier-Free Guidance (CFG) [12] during training. We observe that the strong CFG used in the real score prediction at high-noise timesteps can prematurely bias the student toward dominant modes. Rather than previous methods [14, 52] introducing additional training efforts to explicitly encourage mode-covering, we control the teacher guidance in a timestep-aware manner within the DMD framework. This simple yet effective modification improves the student diversity without extra modules or supervision.

To further overcome the quality bottleneck, we propose **Stagewise Focused Distillation (SFD)**. Student optimization is inherently stage-dependent, shifting from global structure formation to fine-detail refinement. Prior methods [6, 49] typically use uniform objectives throughout distillation, overlooking this training dynamics and leading to poor-quality generation. We therefore argue that a strong student should learn stage specific skills, and design SFD to align training objectives. In the early stage, we apply non-uniform importance sam-

pling and control the guidance in distribution matching to build structural stability and diversity. In the later stage, we switch to pixel-space adversarial distillation to enhance fine details. Distinct from prior approaches [1, 6, 37], our adversarial distillation is formulated in a training-inference consistent manner to refine generation without disrupting the structure. As a result, SFD makes two step sampling both structurally reliable and detail rich.

Even with high-quality 2-step sampling, further acceleration is still limited by heavy block-level computation. Since adjacent denoising steps are often similar, recomputing all blocks at every step is largely redundant, making cross-step reuse a natural complementary direction. However, existing cache methods [3, 23, 24] are mostly tailored to standard multi-step diffusion, and directly applying them to few-step distilled models causes visual degradation due to large reuse error.

To address this, we propose **Distill–Cache co-Training (DCT)**, the first approach to integrate block-level caching into few-step distillation through joint reuse and error correction. Notably, the second stage of SFD naturally provides recovery training for cache accelerated inference on the final step, making fractional step sampling feasible without extra complexity.

In summary, our contributions are as follows:

- We revisit the overlooked role of teacher CFG in DMD and introduce a simple yet effective modification to preserve sampling diversity.
- We propose ***1.x-Distill***, the first distillation framework that breaks the conventional integer-step constraint and achieves diverse, high-quality 1.x-step image generation.
- We introduce two key techniques. **SFD** aligns training objectives with stage-dependent learning dynamics to improve extreme few-step quality, while **DCT** integrates block-level caching with reuse-error correction to eliminate redundant computation.
- We achieve SOTA few-step performance on *SD3-Medium* and *SD3.5-Large*, attaining strong image quality with improved diversity at 1.67 and 1.74 effective NFE respectively, and up to 33× speedup over 28×2 NFE sampling.

2 Related Work

2.1 Few-Step Diffusion Distillation

Existing Few-step distillation methods can be broadly categorized into trajectory-based and distribution-based approaches. **Trajectory-based** methods aim to train a student to reproduce the PF-ODE trajectory of a teacher model. Early works such as *Progressive Distillation* [18, 36] reduce the number of sampling steps in a staged manner but suffer from high training cost and accumulated error. Another representative line, *Consistency Distillation* [7, 27, 41] enforces self-consistency along the trajectory. These methods require careful formulations and non-trivial implementation on large-scale models. **Distribution-based** methods aim to train a few-step student by aligning its output distribution with the target

distribution. *Adversarial Distillation* [18,37,38] can be viewed as a distribution-based approach, which introduces GAN-based [8] objectives to diffusion distillation. Another promising direction explores *score distillation* [28,45,50]. Representative method DMD [50] aligns the student distribution with the teacher via a reverse-KL objective and has become a strong baseline for large-scale few-step generation. Recent works such as DMD2 [49], DMDX [26], TDM [29], SenseFlow [6] and Decoupled-DMD [22] further improve DMD performance by enhancing training within original framework or combining additional objectives. Nevertheless, these methods still suffer from noticeable quality degradation under extreme step budgets for high-resolution generation.

2.2 Cache Accelerator for Diffusion Models

Cache-based acceleration has emerged as an important direction for diffusion efficiency by exploiting cross-timestep feature similarity in a lightweight, plug-and-play manner. Early U-Net-based [35] methods, such as DeepCache [30] and Faster Diffusion [17], pioneered cross-timestep feature reuse, which was later extended to Diffusion Transformers (DiTs) [31] by FORA [40] and Δ-DiT [3]. More recent training-free methods, such as TeaCache [23], EasyCache [53] and TaylorSeer [24] have shown strong effectiveness in conventional multi-step diffusion, typically in the 30–50 step regime. A closely related work, FastCache [21], uses a lightweight learnable linear layer to mitigate reuse error during multi-step inference. However, existing cache methods are largely tailored to standard multi-step sampling, where adjacent steps remain similar. This assumption breaks down in distilled few-step models, making naive feature reuse unreliable. How to effectively introduce caching into this regime without additional complex designs or training procedures remains largely unexplored.

3 Method

3.1 Preliminary: Distribution Matching Distillation

Our *1.x-Distill* framework is built to overcome the limitations of distribution matching distillation. Therefore, we briefly introduce it as follows.

DMD [49,50] trains a few-step student generator G_θ to emulate the output distribution of a pre-trained diffusion model. This goal is formulated as minimizing the reverse Kullback–Leibler divergence between the student distribution p_{fake} and the teacher-induced target distribution p_{real}:

$$\mathcal{L}_{\text{DMD}}(\theta) \;=\; \mathbb{E}_{x\sim p_{\text{fake}}}\Big[\mathrm{KL}(p_{\text{fake}}(x)\,\|\,p_{\text{real}}(x))\Big]. \tag{1}$$

To train G_θ with this objective, the gradient of Eq. (1) with respect to θ is calculated as:

$$\nabla_\theta \mathcal{L}_{\text{DMD}} \;=\; \mathbb{E}_{t\sim\mathcal{U},z}\Big[-\big(s_{\text{real}}(x_t) - \; s_{\text{fake}}(x_t)\big)\frac{\partial \hat{x}_0}{\partial \theta}\Big]. \tag{2}$$

where $\hat{x}_0$ is the denoising prediction of student generator G_θ and $x_t \sim q(x_t \mid \hat{x}_0, t)$ is the sample noised by perturbing the $\hat{x}_0$ according to the diffusion process at level $t \sim \mathcal{U}(0,1)$. The score functions [42] $s_{\text{real}}(x_t) \triangleq \nabla_{x_t} \log p_{\text{real}}(x_t)$ and $s_{\text{fake}}(x_t) \triangleq \nabla_{x_t} \log p_{\text{fake}}(x_t)$ are vector fields that point toward higher-density regions of the corresponding distributions at noise level t. While the real score is estimated by the pretrained model itself, the fake score is estimated by a multi-step proxy that is dynamically updated to describe p_{fake} with diffusion loss. In Eq. (2), the difference $s_{\text{real}}(x_t) - s_{\text{fake}}(x_t)$ drives the student update by pushing its samples toward the teacher-induced target distribution.

3.2 Controlling Guidance in Distribution Matching

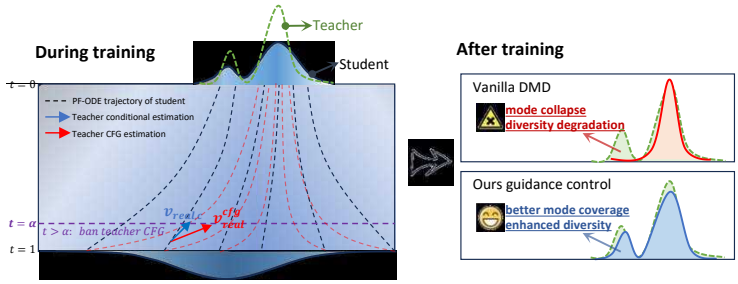


Fig. 2: An illustration of the effect of the teacher's CFG in distillation. At the high-noise timestep t, teacher estimation with strong guidance $v^{\text{cfg}}_{\text{real}} = v_{\text{real,c}} + (w-1)(v_{\text{real,c}} - v_{\text{real},\emptyset})$ tends to drives the student to collapse prematurely toward dominant modes. We propose to disable teacher CFG at $t \in (0, \alpha]$ during distribution matching, encouraging the student to cover more modes during early denoising trajectory.

Classifier-Free Guidance (CFG) [12] is a pervasive component in diffusion inference, yet its role in distribution matching distillation has been largely under-discussed. We notice that in previous open-source DMD-like methods [6, 29, 49], the real score in Eq. (2) is practically calculated with CFG under a strong guidance scale w:

$$\begin{aligned} s_{\text{real}}(x_t) &= s_{\text{real},\emptyset}(x_t) + w\big(s_{\text{real},c}(x_t) - s_{\text{real},\emptyset}(x_t)\big) \\ &= s_{\text{real},c}(x_t) + (w-1)\big(s_{\text{real},c}(x_t) - s_{\text{real},\emptyset}(x_t)\big). \end{aligned} \tag{3}$$

where $s_{\text{real},\emptyset}$ and $s_{\text{real},c}$ are the unconditional and conditional score estimation of the teacher model, respectively. This has also been noted in the a recent study [22], but we offer a different perspective in that overly strong guidance in the real score is an important driver of the *mode collapse* commonly observed in DMD-like methods.

Along the denoising trajectory of a multi-step diffusion model, CFG critically affects the diversity and fidelity trade-off. A higher guidance scale w improves

prompt adherence and fine details, while weaker guidance increases sample diversity. This mechanism also appears in DMD training. As shown in Fig. 2, matching a strongly guided real score yields overly biased supervision. In high-noise regimes, the biased target forces the student to match a mode-seeking score field rather than the full data distribution. As a result, the student is encouraged to collapse toward a few dominant modes early in the denoising trajectory, leading to severe diversity degradation.

A naïve remedy is to globally reduce the teacher guidance scale during distillation, but this substantially degrades quality by weakening the visual constraints for detail synthesis. We find that applying CFG at early timesteps more directly harms diversity, a phenomenon that has also been observed in multi-step diffusion sampling [16]. Therefore, we control the teacher guidance in a timestep-aware manner when constructing the real-score target:

$$s_{\text{real}}(x_t) = \begin{cases} s_{\text{real},\emptyset}(x_t) \;+\; w\big(s_{\text{real},c}(x_t) - s_{\text{real},\emptyset}(x_t)\big)\,, & t \in (0, \alpha] \\ s_{\text{real},c}(x_t), & t \in (\alpha, 1] \end{cases} \tag{4}$$

Following Eq. (4), we disable CFG in real score estimation for early timesteps $t \in (\alpha, 1]$ and use the fully conditional score $s_{\text{real},c}(x_t)$ instead, encouraging the student to learn richer coarse structures and cover more modes during early denoising trajectory. For mid-to-low noise level at $t \in (0, \alpha]$, it is necessary to maintain strong guidance to preserve prompt alignment and fine details. This simple modification retains the DMD framework, yet significantly improves diversity without sacrificing fidelity.

3.3 Stagewise Focused Distillation

Extreme 2-step distillation forces each step to handle both global structure and fine details, making a single uniform objective misaligned with learning dynamics. We propose **Stagewise Focused Distillation**, a two-stage framework with *Structure-focused Distribution Matching* for robust structure and *Detail-focused Adversarial Refinement* for fine details.

Stage I: Structure-focused Distribution Matching Conventional distribution matching is suboptimal in the extreme few-step regime, where stable optimization becomes much more difficult. As shown in Fig. 4, excessive updates from low-noise timesteps ($t \in (0, 0.5)$) are dominated by local texture perturbations, leading to over-sharpened images and abnormal color artifacts. This indicates that uniform timestep sampling misallocates training effort in Stage I. To address this, we design a *importance timestep sampling* strategy for the structure-focused stage. Under teacher scheduler setting (shift=3.0), the sampling probability peaks around $t = 0.75$ and decays rapidly when

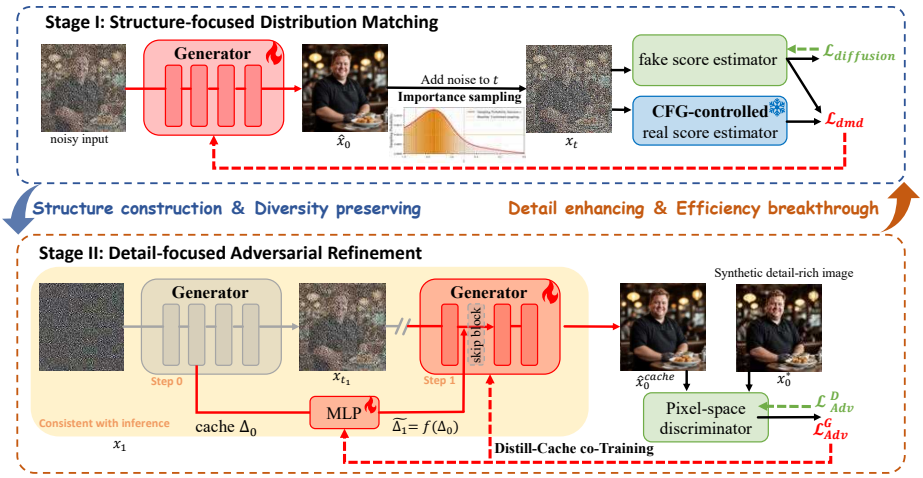


Fig. 3: Overview of *1.x-Distill*. Our guidance control (Sec. 3.2) and cache design (Sec. 3.4) are both constructed in the two-stage framework (Sec. 3.3). **Stage I**: Train the generator with DMD loss. Within DMD framework, we apply *importance sampling* on diffusion timestep t, and *control the guidance* according to sampled t when compute the real score. **Stage II**: Train the generator with pixel-space adversarial loss. Our GAN framework produces $\hat{x}_0$ along generator inference path, which naturally incorporates block-cache design. The generator and MLP module are jointly optimized.

Stage II: Detail-focused Adversarial Refinement After Stage I, the student already produces structurally plausible two-step samples with stable semantics. We therefore introduce a pixel-space GAN [8] objective in Stage II to further refine fine-grained details:

$$\begin{aligned}\mathcal{L}_{\text{Adv}}^{G} &= \mathbb{E}_{\hat{x}_0}\left[-\log D\left(V(\hat{x}_0)\right)\right], \\ \mathcal{L}_{\text{Adv}}^{D} &= \mathbb{E}_{x_0^\star}\left[-\log D\left(V(x_0^\star)\right)\right] + \mathbb{E}_{\hat{x}_0}\left[\log D\left(V(\hat{x}_0)\right)\right],\end{aligned} \tag{5}$$

where V denotes the VAE decoder and D is the pixel-space discriminator. Prior methods [6, 49] jointly optimize the generator with the DMD loss Eq. (2) and the GAN loss, generating samples $\hat{x}_0$ via single-step prediction from randomly sampled noise levels ($t \in (0, 1]$). Such training introduces large variation in generator outputs, making discriminator optimization unstable. In contrast, our GAN framework maintains training-inference consistency. We generate $\hat{x}_0$ along the few-step inference path and forward propagate the generator in the last step to focus on refinement without disrupting the structure learned in Stage I. We further simplify the construction of real samples. The "real" images $x_0^\star$ in our adversarial training are generated from the same noise by a multi-step model. Multi-step synthetic $x_0^\star$ typically exhibit richer details while remaining more structurally consistent with the distilled distribution. Consequently, our formulation removes the reliance on high-quality image datasets and reducing the domain gap between real and generated images that can otherwise bias detail

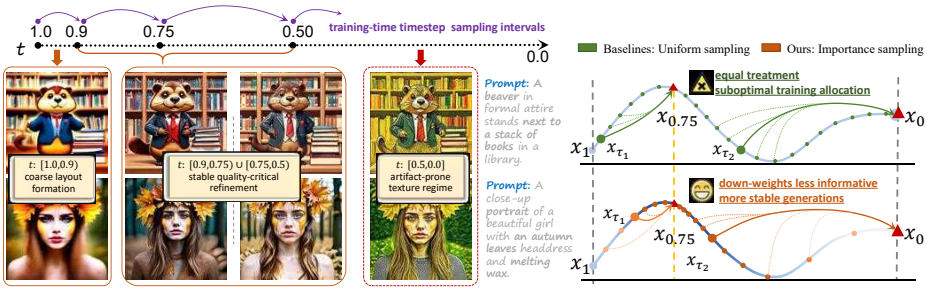
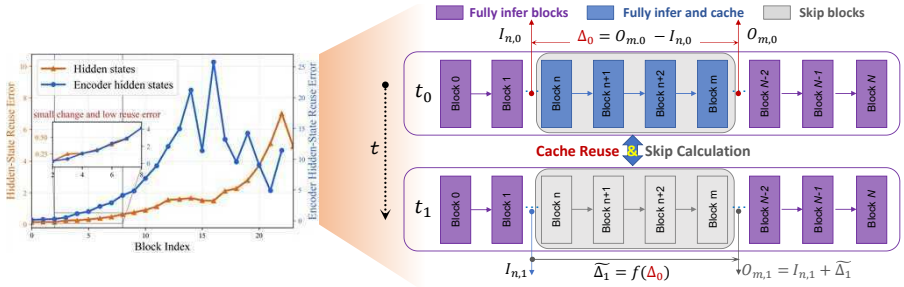


Fig. 4: Importance sampling in Stage I. *Left*: Under teacher scheduler (shift=3.0), we split timesteps from 1.0 to 0.0 into four windows to probe their effects. *Right*: Uniform sampling treats all timesteps equally, while our importance sampling down-weights less informative ones and concentrates training on the more reliable region.

refinement. For the discriminator, we follow the architecture in [20]. A frozen ConvNeXt [25] backbone is used to extract fine-grained features, followed by a trainable classification head, which empirically performs well for detail-oriented refinement tasks.

3.4 Caching for Distilled Model



(a) Temporal contribution variation. (b) Block-level caching with MLP compensation.

Fig. 5: Caching for a distilled 2-step student. (a) We measure block-wise reuse error as the contribution change across adjacent steps on *SD3-M*, $e_n = \|\Delta_{n,t+1} - \Delta_{n,t}\|_1$, where $\Delta_{n,t} = O_{n,t} - I_{n,t}$. Early blocks exhibit consistently small e_n, indicating strong temporal redundancy and low reuse error. **(b)** Leveraging this property, we cache the contribution of a block segment $[n, m]$ at step t_0, $\Delta_0 = O_{m,0} - I_{n,0}$, skip the segment at t_1, and recover the output via $\hat{O}_{m,1} = I_{n,1} + f(\Delta_0)$.

Our SFD achieves high-quality 2-step sampling, while direct 1-step distillation still degrades quality. To eliminate redundant computation in full per-iteration computation, we introduce *block-level caching* into the 2-step DiT-based student, pushing efficiency further and achieving 1.x-NFE inference.

Cache Design for 2-step Student We implement cache-accelerated inference through block-level feature reuse across consecutive denoising steps. Suppose the model is fully evaluated at step t, and a block segment $[n, m]$ is skipped at step $t+1$. Let $I_{n,t}$ and $O_{m,t}$ denote the input of block n and the output of block m, respectively. We cache the block contribution

$$\Delta_t = O_{m,t} - I_{n,t},$$

and directly reuse it to bypass the skipped computation at the next step. To reduce the resulting reuse error, we introduce a **learnable error-compensation module** $f(\cdot)$, implemented as a lightweight residual MLP, and predict the reused contribution as

$$\tilde{\Delta}_{t+1} = f(\Delta_t), \qquad \hat{O}_{m,t+1} = I_{n,t+1} + \tilde{\Delta}_{t+1}.$$

Since $f(\cdot)$ is negligible compared with the skipped DiT blocks, this design adds little overhead while substantially reducing reuse error. In our 2-step setting, the first step is fully computed and the second step reuses the predicted block contribution.

Distill–Cache co-Training Under our SFD framework, Stage II naturally supports cache recovery training, as the adversarial refinement strictly aligns with the inference pipeline. With caching enabled and the correction module f inserted before Stage II, the adversarial objective directly supervises cache-accelerated inference and helps recover from cache-induced distortions. Denote by $\hat{x}_0^{\text{cache}}$ the image decoded from the student output produced by the cache-accelerated second step. We optimize the adversarial objective:

$$\min_{\theta,\,\phi} \mathcal{L}_{\text{Adv}}^{G} = \min_{\theta,\,\phi} \mathbb{E}_{\hat{x}_0^{\text{cache}}} \left[-\log D\left(V(\hat{x}_0^{\text{cache}})\right) \right], \tag{6}$$

where θ denotes the parameters of the student backbone G_θ and ϕ denotes the parameters of the correction module f. In practice, we first freeze θ and warm up ϕ for a few iterations. We then jointly optimize detail enhancement and cache recovery under the same adversarial supervision. Notably, we require no feature-level alignment loss, as pixel-level adversarial supervision alone compensates reuse error, restores image quality, and enables stable fractional-step inference.

4 Experiment

4.1 Experimental Setup

Settings We apply ***1.x-Distill*** to two representative DiT-based text-to-image models, *SD3-Medium* (2B) and *SD3.5-Large* (8B) [5]. To provide a clear comparison of acceleration, we define the *effective NFE* (number of function evaluations) as the ratio of fully computed DiT blocks during student sampling to the total number of blocks in the original model. For *SD3-Medium* with 24 DiT blocks,

skipping layers 3–8 in the second denoising step yields an effective NFE of 1.75, while skipping layers 3–10 further reduces it to 1.67. For *SD3.5-Large* with 38 DiT blocks, skipping layers 3–12 in the second step yields an effective NFE of 1.74. We also report 4-step results of SFD without caching for direct comparison with prior 4-step methods. Since our adversarial training does not rely on external image datasets, we use only prompt data from JourneyDB [43] throughout training. More implementation and training details of our method are provided in the supplementary material.

Baselines We compare our method against all publicly available few-step checkpoints of *SD3-Medium* and *SD3.5-Large*, including trajectory- and distribution-based methods like Hyper-SD [33], PCM [44], Flash [2], LADD(Turbo) [37] and TDM [29]. Since most open-source models do not directly support 2-step inference, for fair comparison we also try our best to implement the 2-step results of representative distribution matching methods, including DMD2 [49] and TDM.

Table 1: Quantitative comparison on COCO-10K. * indicates results reproduced by us due to missing official checkpoints. FID is computed between teacher samples and student samples. Img-free indicates the method does not use external real-image datasets during training.

Method	Step	#NFE	FID [10]↓	CLIP [9]↑	AS [39]↑	PS [15]↑	IR [48]↑	HPSv2 [47]↑	Img-free
Stable Diffusion 3 Medium 1024×1024									
Base Model	28	28×2	–	0.3176	5.6348	22.5554	1.0429	30.7197	✗
Hyper-SD [33]	4	4×2	15.5475	0.3127	4.9582	21.6407	0.7543	28.7578	✗
PCM [44]	4	4	17.5605	0.3102	5.7743	22.0690	0.6715	29.0864	✗
Flash [2]	4	4	15.6443	**0.3166**	5.5485	22.3879	0.8938	29.4835	✗
DMD2* [49]	4	4	14.7125	0.3122	5.4632	22.4120	0.9981	31.0152	✗
TDM [29]	4	4	14.6424	0.3128	5.5494	22.4681	1.0021	31.7512	✓
Ours-SFD	4	4	**14.1349**	0.3149	**5.9197**	**22.8155**	**1.1196**	**32.5337**	✓
Δ (vs best baseline)	–	–	-0.5075	–	+0.1454	+0.2601	+0.0767	+0.7825	–
PCM	2	2	41.6561	0.3077	5.1325	20.9493	0.2011	24.8431	✗
TDM*	2	2	19.3005	0.3186	5.1441	22.4756	1.1101	31.4725	✓
1.x-Distill-slow	2	**1.75**	**15.7863**	**0.3208**	**5.1844**	**22.5161**	**1.1312**	**32.2550**	✓
Δ	–	–	-3.5142	+0.0022	+0.0403	+0.0405	+0.0211	+0.7825	–
1.x-Distill-fast	2	**1.67**	16.7179	0.3204	5.1206	22.3342	1.0673	31.6850	✓
Stable Diffusion 3.5 Large 1024×1024									
Base Model	28	28×2	-	0.3196	5.9178	22.5994	1.0641	31.1081	✗
Turbo [37]	4	4	**15.3123**	0.3161	**6.1308**	22.7418	0.9288	30.4127	✗
Ours-SFD	4	4	17.3588	**0.3187**	5.9939	**22.9046**	**1.2011**	**32.9020**	✓
Δ	–	–	–	+0.0026	–	+0.1628	+0.1370	+1.7939	–
TDM*	2	2	26.8084	**0.3224**	5.3110	22.1424	0.9307	28.4919	✓
1.x-Distill	2	**1.74**	**22.0545**	0.3191	**5.7976**	**22.7963**	**1.1463**	**32.0059**	✓
Δ	–	–	-4.7539	–	+0.4866	+0.6539	+0.2156	+3.5140	–

4.2 Main Results

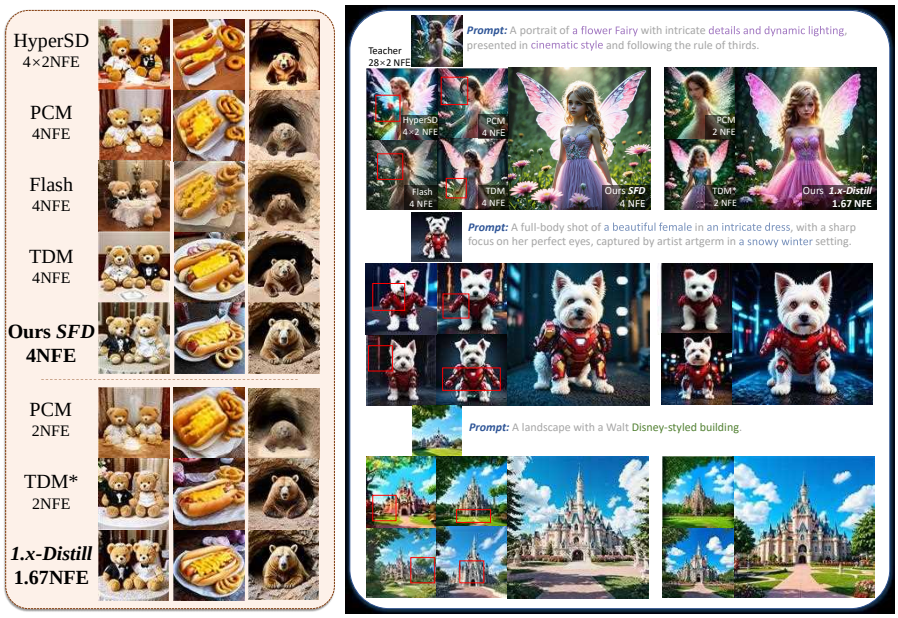


Fig. 6: Qualitative comparison on *SD3-Medium*. Since most open-source baselines only provide 4-step checkpoints, we first compare all methods at 4 steps for fairness. Our **SFD** already produces clearer and more appealing images, while other methods often show generation failures, color shifts, blur, or degraded aesthetics (red boxes). When pushed to 2 steps, ***1.x-Distill*** still clearly surpasses all baselines.

Quantitative Comparison Following prior work [26,50], we conduct our evaluation on 10K prompts from COCO-2014 [19]. We report FID [10] for distribution fidelity, CLIP Score [9] for prompt alignment and commonly used human-preference metrics including Pick Score [15], Aesthetic Score [39], HPSv2 [47], and ImageReward [48]. As shown in Tab. 1, *1.x-Distill* achieves a strong quality–efficiency trade-off on both *SD3-M* and *SD3.5-L*. Before caching, our **SFD** already performs strongly at 4 step, achieving the best preference scores on *SD3-M*. After enabling block caching, the advantage becomes clearer in the extreme few-step regime. On *SD3-M*, ***1.x-Distill***–slow surpasses the strongest reproduced baseline TDM at a lower effective NFE (1.75 vs. 2), and even outperforms all existing 4-NFE methods on most quality metrics. Increasing the cache ratio, ***1.x-Distill***–fast further reduces the effective NFE to 1.67 with only a modest performance drop. We further evaluate on DPG-Bench [13], a popular text-to-image benchmark, to comprehensively assess our models under complex prompts. As shown in Tab. 2, our distilled models outperform the original multi-step teachers in overall score under aggressive step compression.

In addition, we evaluate sampling diversity using LPIPS [51]. For each prompt, we generate four samples with different seeds and compute pairwise LPIPS distances, averaged over 1K COCO-2014 prompts. Results in Tab. 3 show that our

Table 2: Quantitative evaluation on DPG-Bench of our *1.x-Distill* model against its multi-step teacher.

Model	#NFE	Overall↑	Global	Entity	Attribute	Relation	Other
SD3-M	28×2	85.46	92.01	89.07	89.88	90.46	91.67
Ours	**1.75**	**86.13**	89.97	**92.26**	89.41	**90.77**	**92.01**
SD3.5-L	28×2	84.74	90.02	89.67	90.97	89.70	89.53
Ours	**1.74**	**85.11**	89.78	**90.79**	89.49	**92.21**	**89.66**

Table 3: Quantitative evaluation of diversity on COCO-1K using LPIPS.

Method	#NFE	LPIPS↑
SD3-M	28×2	0.6594
Flash	4	0.6161
TDM	4	0.6297
Ours	**1.75**	**0.6432**

method achieves substantially higher diversity than prior distribution-matching baselines (Flash and TDM).

Qualitative Comparison In addition to quantitative comparisons, we present qualitative results in Fig. 6. Across a wide range of prompts, *1.x-Distill* consistently produces visually superior images compared to prior methods. Remarkably, even under the extremely low compute budget (1.67 NFE), our distilled model preserves coherent global structure while generating rich and realistic fine details, even surpassing the teacher model. Besides, see the visual comparison of the diversity in the supplementary material.

User Study We conduct a user study to assess perceptual quality and prompt alignment. 20 human raters compare images generated by our method against strong few-step baselines on 3,200 prompts of 4 styles from HPSv2. The results in Fig. 7 show a clear human preference for *1.x-Distill*.

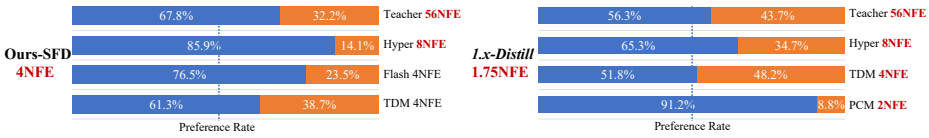


Fig. 7: User study: Comparing images generated by *1.x-Distill* with other models.

4.3 Ablation Studies

We perform comprehensive ablation studies to validate the effectiveness of each component and identify optimal design choices. Unless otherwise noted, all studies are performed in *SD3-Medium* at 1024×1024.

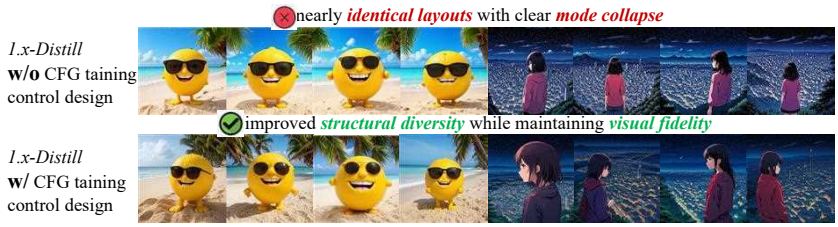


Fig. 8: Effect of guidance control strategy on sample diversity.

Effect of Guidance Control To validate our guidance control strategy, we compare the sampling results of *1.x-Distill* with and without it under a unified guidance scale $w = 7.0$. As shown in Fig. 8, enabling guidance control produces more diverse structural layouts while preserving comparable image quality. As discussed in Sec. 3.2, completely disabling the teacher CFG in the mid-to-low noise regime may harm distillation results. We further vary the threshold α to identify the optimal control boundary. As shown in Fig. 8, when $\alpha < 0.92$, the distilled model exhibits clear degradation in quality metrics. This is because the student increasingly relies on mid-to-low noise timesteps to learn perceptual quality rather than structural diversity. We therefore set $a = 0.94$ in our *1.x-Distill*.

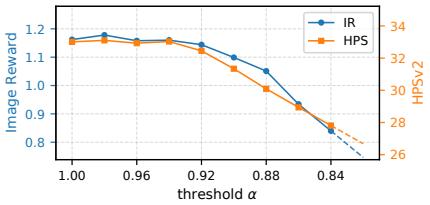


Fig. 9: Effect of control threshold α.

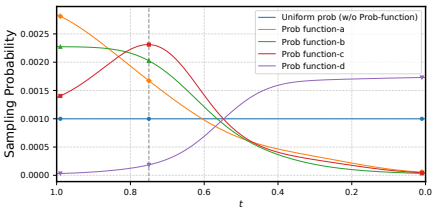


Fig. 10: Ablation of sampling probability.

Table 4: Ablations of Stagewise Focused Distillation (SFD). *Top*: Stage I timestep sampling ablations where our strategy (c in Fig. 10) outperforms uniform sampling. *Bottom*: Stage II further improves detail fidelity over Stage I. All settings use the same training iterations with controlled guidance.

Stage I	Stage II	CLIP↑	AS↑	PS↑	IR↑	HPSv2↑
✓: Uniform sampling	✗	0.3154	5.1432	22.4546	1.1081	31.4725
✓: a	✗	0.3161	5.6044	22.4337	1.1170	31.8419
✓: b	✗	0.3155	5.5442	22.3769	1.1020	31.8944
✓: c	✗	0.3159	5.6210	22.4694	1.1226	32.0208
✓: d	✗	0.3127	4.6196	22.4185	1.1419	30.2768
✗	✓	0.2144	3.3287	13.1621	0.1352	19.8334
✓: c	✓	**0.3184**	**5.7485**	**22.6995**	**1.1601**	**33.0293**

Effect of Stagewise Focused Distillation In structure-focused Stage I, we bias distribution matching away from the low-noise regime using non-uniform timestep sampling. Four schemes (Fig. 10) are evaluated when training our 2-step model and we report the results in the top of Tab. 4. Compared with uniform sampling and the low-noise–biased curve d, curves a–c that downweight low-noise timesteps consistently perform better. Curve c performs best, as it also moderately reduces sampling near the pure-noise end, enabling more effective distillation.

However, only structure-focused training in stage I is not good enough as details generation ability remains suboptimal. See the bottom part of Tab. 4, by further enabling the proposed Detail-focused Adversarial Refinement (Stage II), the student model obtains consistent gains across all quality metrics, indicating that Stage II effectively complements Stage I by enhancing fine details.

Effect of our Cache Design We conduct extensive experiments on proposed caching design for extremely few-step distilled models, addressing two questions: Compared with training-free caching applied after distillation, can DCT (Sec. 3.4) recover the quality degradation? Whether the lightweight MLP is necessary for reuse-error compensation?

Table 5: Ablation results on cache settings and training variants.

Cache	+Train	+MLP	#NFE	CLIP↑	PS↑	IR↑	HPS↑
SFD w/o cache			2.00	0.3184	22.6995	1.1601	33.0293
6 blocks	✗	✗	1.75	0.3170	22.2042	1.0152	30.9216
6 blocks	✓	✗	1.75	0.3205	22.3113	1.0500	31.0544
6 blocks	✓	✓	1.75	**0.3208**	**22.5161**	**1.1312**	**32.2550**
8 blocks	✗	✗	1.67	0.3183	21.7768	0.8944	29.6062
8 blocks	✓	✗	1.67	0.3198	21.9289	0.9238	30.0994
8 blocks	✓	✓	1.67	**0.3204**	**22.3342**	**1.0673**	**31.6850**

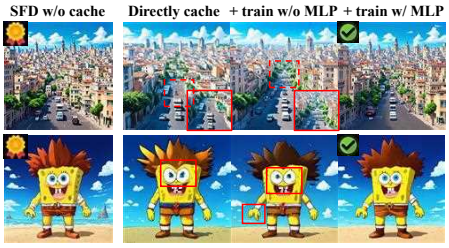


Fig. 11: Qualitative ablation of block-level caching for our DCT.

As shown in Tab. 5 and Fig. 11, directly applying in block caching after distillation causes severe quality degradation in both quantitative metrics and visual fidelity, showing that cache acceleration is not an effective plug-and-play component in distilled few-step models. Instead, introducing caching during distillation and optimizing it with DCT largely restores image quality. We further find that if remove the MLP, DCT only partially compensates reuse errors and yields limited recovery. In contrast, by explicitly predicting the reused block contribution, the lightweight MLP significantly improves fidelity, bringing the cached model much closer to the full-computation baseline. These results validate the effectiveness of DCT and the necessity of explicit reuse-error compensation.

Note. Additional experimental analyses are provided in the supplementary material.

5 Conclusion

In this work, we present ***1.x-Distill***, the first framework that pushes distribution matching distillation beyond the conventional integer-step regime. To address diversity degradation problem in DMD, we revisit the overlooked role of teacher CFG and introduce a guidance control strategy. We then propose SFD, which decouples structure and detail learning to improve generation quality under extreme step compression. Furthermore, we combine learnable block-level caching into our distillation via DCT. On SD3-Medium and SD3.5-Large, *1.x-Distill* achieves remarkable performance in both sampling diversity and image quality at 1.67 and 1.74 effective NFE, respectively.

Limitations & future work. While our method demonstrates promising results, its effectiveness on recent larger-scale generative models, such as Qwen-Image(20B) [46], remains to be further explored. In addition, extending *1.x-Distill* to video generation is also an important direction for future work.

References

1. Bandyopadhyay, H., Entezari, R., Scott, J., Adithyan, R., Song, Y.Z., Jampani, V.: Sd3. 5-flash: Distribution-guided distillation of generative flows. arXiv preprint arXiv:2509.21318 (2025) 3
2. Chadebec, C., Tasar, O., Benaroche, E., Aubin, B.: Flash diffusion: Accelerating any conditional diffusion model for few steps image generation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 15686–15695 (2025) 10, 23
3. Chen, P., Shen, M., Ye, P., Cao, J., Tu, C., Bouganis, C.S., Zhao, Y., Chen, T.: δ-dit: A training-free acceleration method tailored for diffusion transformers. arXiv preprint arXiv:2406.01125 (2024) 3, 4
4. Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. Advances in neural information processing systems **34**, 8780–8794 (2021) 1
5. Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., et al.: Scaling rectified flow transformers for high-resolution image synthesis. In: Forty-first international conference on machine learning (2024) 1, 9
6. Ge, X., Zhang, X., Xu, T., Zhang, Y., Zhang, X., Wang, Y., Zhang, J.: Seneflow: Scaling distribution matching for flow-based text-to-image distillation. arXiv preprint arXiv:2506.00523 (2025) 2, 3, 4, 5, 7
7. Geng, Z., Deng, M., Bai, X., Kolter, J.Z., He, K.: Mean flows for one-step generative modeling. arXiv preprint arXiv:2505.13447 (2025) 1, 2, 3
8. Goodfellow, I., 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) 4, 7
9. Hessel, J., Holtzman, A., Forbes, M., Bras, R.L., Choi, Y.: Clipscore: A reference-free evaluation metric for image captioning. arXiv preprint arXiv:2104.08718 (2021) 10, 11, 23
10. 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) 10, 11, 23

11. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems **33**, 6840–6851 (2020) 1
12. Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598 (2022) 2, 5
13. 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) 11, 23
14. Jiang, D., Liu, D., Wang, Z., Wu, Q., Li, L., Li, H., Jin, X., Liu, D., Li, Z., Zhang, B., et al.: Distribution matching distillation meets reinforcement learning. arXiv preprint arXiv:2511.13649 (2025) 2
15. Kirstain, Y., Polyak, A., Singer, U., Matiana, S., Penna, J., Levy, O.: Pick-a-pic: An open dataset of user preferences for text-to-image generation. Advances in neural information processing systems **36**, 36652–36663 (2023) 10, 11, 23
16. Kynkäänniemi, T., Aittala, M., Karras, T., Laine, S., Aila, T., Lehtinen, J.: Applying guidance in a limited interval improves sample and distribution quality in diffusion models. Advances in Neural Information Processing Systems **37**, 122458–122483 (2024) 6
17. Li, S., Hu, T., van de Weijer, J., Khan, F.S., Liu, T., Li, L., Yang, S., Wang, Y., Cheng, M.M., Yang, J.: Faster diffusion: Rethinking the role of the encoder for diffusion model inference. Advances in neural information processing systems **37** (2024) 4
18. Lin, S., Wang, A., Yang, X.: Sdxl-lightning: Progressive adversarial diffusion distillation. arXiv preprint arXiv:2402.13929 (2024) 3, 4
19. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014) 11
20. Lin, X., Yu, F., Hu, J., You, Z., Shi, W., Ren, J.S., Gu, J., Dong, C.: Harnessing diffusion-yielded score priors for image restoration. ACM Transactions on Graphics (TOG) **44**(6), 1–21 (2025) 8, 21
21. Liu, D., Yu, Y., Zhang, J., Li, Y., Lengerich, B., Wu, Y.N.: Fastcache: Fast caching for diffusion transformer through learnable linear approximation. arXiv preprint arXiv:2505.20353 (2025) 4
22. Liu, D., Gao, P., Liu, D., Du, R., Li, Z., Wu, Q., Jin, X., Cao, S., Zhang, S., Li, H., et al.: Decoupled dmd: Cfg augmentation as the spear, distribution matching as the shield. arXiv preprint arXiv:2511.22677 (2025) 4, 5
23. Liu, F., Zhang, S., Wang, X., Wei, Y., Qiu, H., Zhao, Y., Zhang, Y., Ye, Q., Wan, F.: Timestep embedding tells: It's time to cache for video diffusion model. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 7353–7363 (June 2025) 3, 4
24. Liu, J., Zou, C., Lyu, Y., Chen, J., Zhang, L.: From reusing to forecasting: Accelerating diffusion models with taylorseers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15853–15863 (October 2025) 3, 4
25. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11976–11986 (2022) 8
26. Lu, Y., Ren, Y., Xia, X., Lin, S., Wang, X., Xiao, X., Ma, A.J., Xie, X., Lai, J.H.: Adversarial distribution matching for diffusion distillation towards efficient image and video synthesis. arXiv preprint arXiv:2507.18569 (2025) 2, 4, 11

27. Luo, S., Tan, Y., Huang, L., Li, J., Zhao, H.: Latent consistency models: Synthesizing high-resolution images with few-step inference. arXiv preprint arXiv:2310.04378 (2023) 2, 3
28. Luo, W., Hu, T., Zhang, S., Sun, J., Li, Z., Zhang, Z.: Diff-instruct: A universal approach for transferring knowledge from pre-trained diffusion models. Advances in Neural Information Processing Systems **36**, 76525–76546 (2023) 4
29. Luo, Y., Hu, T., Sun, J., Cai, Y., Tang, J.: Learning few-step diffusion models by trajectory distribution matching. arXiv preprint arXiv:2503.06674 (2025) 2, 4, 5, 10, 23
30. Ma, X., Fang, G., Wang, X.: Deepcache: Accelerating diffusion models for free. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15762–15772 (2024) 4
31. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4195–4205 (2023) 4
32. Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: Sdxl: Improving latent diffusion models for high-resolution image synthesis. arXiv preprint arXiv:2307.01952 (2023) 1
33. 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. arXiv preprint arXiv:2404.13686 (2024) 1, 10, 23
34. 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) 1
35. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18. pp. 234–241. Springer (2015) 4
36. Salimans, T., Ho, J.: Progressive distillation for fast sampling of diffusion models. arXiv preprint arXiv:2202.00512 (2022) 1, 3
37. 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) 1, 3, 4, 10, 23
38. Sauer, A., Lorenz, D., Blattmann, A., Rombach, R.: Adversarial diffusion distillation. In: European Conference on Computer Vision. pp. 87–103. Springer (2024) 1, 4
39. Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al.: Laion-5b: An open large-scale dataset for training next generation image-text models. Advances in neural information processing systems **35**, 25278–25294 (2022) 10, 11, 23
40. Selvaraju, P., Ding, T., Chen, T., Zharkov, I., Liang, L.: Fora: Fast-forward caching in diffusion transformer acceleration. arXiv preprint arXiv:2407.01425 (2024) 4
41. Song, Y., Dhariwal, P., Chen, M., Sutskever, I.: Consistency models. In: International Conference on Machine Learning. pp. 32211–32252. PMLR (2023) 1, 3
42. Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. arXiv preprint arXiv:2011.13456 (2020) 5
43. Sun, K., Pan, J., Ge, Y., Li, H., Duan, H., Wu, X., Zhang, R., Zhou, A., Qin, Z., Wang, Y., et al.: Journeydb: A benchmark for generative image understanding. Advances in neural information processing systems **36**, 49659–49678 (2023) 10

44. Wang, F.Y., Huang, Z., Bergman, A., 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) 10, 23
45. Wang, Z., Lu, C., Wang, Y., Bao, F., Li, C., Su, H., Zhu, J.: Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. Advances in neural information processing systems **36**, 8406–8441 (2023) 4
46. Wu, C., Li, J., Zhou, J., Lin, J., Gao, K., Yan, K., Yin, S.m., Bai, S., Xu, X., Chen, Y., et al.: Qwen-image technical report. arXiv preprint arXiv:2508.02324 (2025) 15
47. Wu, X., Hao, Y., Sun, K., Chen, Y., Zhu, F., Zhao, R., Li, H.: Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. arXiv preprint arXiv:2306.09341 (2023) 10, 11, 23
48. Xu, J., Liu, X., Wu, Y., Tong, Y., Li, Q., Ding, M., Tang, J., Dong, Y.: Imagereward: Learning and evaluating human preferences for text-to-image generation. Advances in Neural Information Processing Systems **36**, 15903–15935 (2023) 10, 11, 23
49. 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) 1, 2, 4, 5, 7, 10
50. Yin, T., Gharbi, M., Zhang, R., Shechtman, E., Durand, F., Freeman, W.T., Park, T.: One-step diffusion with distribution matching distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6613–6623 (2024) 1, 2, 4, 11
51. 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) 11
52. 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. arXiv preprint arXiv:2510.08431 (2025) 2
53. Zhou, X., Liang, D., Chen, K., Feng, T., Chen, X., Lin, H., Ding, Y., Tan, F., Zhao, H., Bai, X.: Less is enough: Training-free video diffusion acceleration via runtime-adaptive caching. arXiv preprint arXiv:2507.02860 (2025) 4

Appendix

Table of Contents

A Algorithm

Algorithm 1 presents the training pseudocode of our *1.x-Distill*: Stage I performs structure-focused distribution matching with CFG-controlled teacher guidance, while Stage II refines fine-grained details via pixel-space adversarial supervision under the cached inference path. Algorithm 2 presents the inference procedure of *1.x-Distill*.

Algorithm 1 1.x-Distill Training Procedure

Require: Pretrained teacher model μ_{real}, 2-step generator schedule $S = \{t_0, t_1\}$ (*e.g.*$\{1.0, 0.75\}$), pixel-space discriminator D, VAE decoder V
Ensure: Optimized student generator G_θ with attached MLP module f_ϕ

$G_\theta \leftarrow \text{CopyWeights}(\mu_{\text{real}})$ ▷ Initialize generator
$\mu_{\text{fake}} \leftarrow \text{CopyWeights}(\mu_{\text{real}})$ ▷ Initialize fake score estimator
# -- Stage I: Structure-focused Distribution Matching --
for iter $= 1$ **to** max_iters_stage1 **do**
  $x_{t_0} \sim \mathcal{N}(0, I)$
  Sample t_i from S
  $x_{t_i} \leftarrow \texttt{BackwardSimulation}(G_\theta, x_{t_0}, t_0 \to t_i)$ ▷ Get noisy input as DMD2
  $\hat{x}_0 \leftarrow G_\theta(x_{t_i})$
  if iter **mod** $TTUR_1 == 0$ **then**
  # Update generator G_θ
    $t \sim \texttt{ImportanceSampling}(0, 1)$ ▷ Importance Sampling Sec. 3.3
    $x_t \leftarrow \texttt{AddNoise}(\hat{x}_0, t)$
    $\nabla_\theta \mathcal{L}_{\texttt{DMD}} \leftarrow \texttt{GradDMD}(\texttt{CFG controlled } \mu_{\texttt{real}}, \mu_{\texttt{fake}}, x_t)$ ▷ Eq. (2), Eq. (4)
    $G_\theta \leftarrow \texttt{update}(G_\theta, \nabla_\theta \mathcal{L}_{\texttt{DMD}})$
  end if
  $t \sim \mathcal{U}(0, 1)$
  $x_t \leftarrow \texttt{AddNoise}(\texttt{detach}(\hat{x}_0), t)$
  $\mathcal{L}_{\texttt{diffusion}} \leftarrow \texttt{DiffusionLoss}(\mu_{\texttt{fake}}(x_t, t), \texttt{detach}(\hat{x}_0))$
  $\mu_{\texttt{fake}} \leftarrow \texttt{update}(\mu_{\texttt{fake}}, \mathcal{L}_{\texttt{diffusion}})$
end for
# -- Stage II: Detail-focused Adversarial Refinement --
for iter $= 1$ **to** max_iters_stage2 **do**
  $x_{t_0} \sim \mathcal{N}(0, I)$
  $x_{t_1}, \Delta_0 \leftarrow G_\theta(x_{t_0}, t_0)$
  $\hat{x}_0^{\text{cache}} \leftarrow G_\theta(\texttt{detach}(x_{t_1}), t_1, f_\phi(\Delta_0))$ ▷ Get $\hat{x}_0^{\text{cache}}$ along inference path
  if iter **mod** $TTUR_2 == 0$ **then**
  # Update generator G_θ, MLP f_ϕ
    $\mathcal{L}_{\text{Adv}}^G \leftarrow -\log D\left(V(\hat{x}_0^{\text{cache}})\right)$
    $G_\theta \leftarrow \texttt{update}(G_\theta, \mathcal{L}_{\text{Adv}}^G)$
    $f_\phi \leftarrow \texttt{update}(f_\phi, \mathcal{L}_{\text{Adv}}^G)$
  end if
  $\mathcal{L}_{\text{Adv}}^D \leftarrow -\log D\left(V(x_0^\star)\right) + \log D\left(V(\hat{x}_0^{\text{cache}})\right)$
  $D \leftarrow \texttt{update}(D, \mathcal{L}_{\text{Adv}}^G)$
end for

Algorithm 2 1.x-Distill Inference Procedure

Require: Distilled 2-step generator G_θ with schedule $S = \{t_0, t_1\}$, trained lightweight module f_ϕ, VAE decoder V
Ensure: Clean image sample x_0
1: $x_{t_0} \sim \mathcal{N}(0, I)$
2: $x_{t_1}, \Delta_0 \leftarrow G_\theta(x_{t_0}, t_0)$
3: $x_0 \leftarrow G_\theta(x_{t_1}, t_1, f_\phi(\Delta_0))$
4: $x_0 \leftarrow V(x_0)$ ▷ Decode latents to pixel-space image

B Implementation Details

B.1 Discriminator Design

Our discriminator architecture follows the design in [20], consisting of a frozen ConvNeXt backbone and a lightweight trainable head. Specifically, we use the ConvNeXt-XXL visual encoder from a pretrained OpenCLIP model[3] as the feature extractor, which outputs multi-level feature maps with channel dimensions $\{384, 768, 1536\}$ together with a pooled global feature of dimension 1024. On top of these representations, we attach a multi-level discriminator head. Each intermediate feature map is processed by spectrally normalized convolution layers with LeakyReLU activations and BlurPool downsampling to produce realism predictions at different spatial scales. The pooled feature is further passed through a linear classifier to obtain a global realism score. Predictions from all levels are first averaged within each level and then summed across levels to produce the final adversarial signal as Eq. (5).

B.2 Training Details

We implement our *1.x-Distill* framework in PyTorch and train on 8 NVIDIA A100 GPUs. We adopt Fully Sharded Data Parallel (FSDP) to scale training across GPUs and enable mixed-precision training in `torch.bfloat16` for both efficiency and stability.

Follow the shift(3.0) from the scheduler of teacher model, we set the generator timestep schedule to $S = \{1.0,\ 0.9,\ 0.75,\ 0.5\}$ for 4-step SFD, and $S = \{1.0,\ 0.75\}$ for 2-step *1.x-Distill*. For optimization, we employ the `AdamW` optimizer across all trainable components, including the student generator G_θ, the fake score estimator μ_{fake}, the pixel-space discriminator D, and the MLP module f_ϕ. We set the weight decay to $1e-4$ and the exponential moving average coefficients $(\beta_1, \beta_2) = (0, 0.999)$ in Stage I, $(0.9, 0.95)$ in Stage II. The learning rate and other configurations are listed in Tab. 6. Notably, the total training cost of *1.x-Distill* is about 71 GPU-hours on SD3-Medium (2B) and 104 GPU-hours on SD3.5-Large (8B). By contrast, DMD2 trains for 64×60 GPU hours on SDXL (3.5B), indicating that *1.x-Distill* is significantly more training-efficient.

[3] `laion/CLIP-convnext_xxlarge-laion2B-s34B-b82K-augreg-soup`

Table 6: Training configurations for SD3-Medium and SD3.5-Large.

Config	SD3-Medium	SD3.5-Large
Resolution	1024×1024	
Total Batch Size	32	
Precision	`bfloat16`(t in `float32`)	
Stage I		
Generator Learning Rate	5e-6	2e-6
Fake Learning Rate	4e-5	2e-5
Guidance Scale w	7.0	7.0
Control Threshold α	0.94	0.94
Generator Update Frequency	1	1
Max Iteration	1000	600
Training Cost	27 GPU hours	40 GPU hours
Stage II		
Cache config (skip blocks)	[3~8]/[3~10]	[3~12]
Generator Learning Rate	1e-6	5e-7
MLP Learning Rate	1e-3	1e-3
Discriminator Learning Rate	1e-5	1e-5
Generator Update Frequency	3	3
MLP Warmup Iteration	500	500
Max Iteration	2000	2000
Training Cost	44 GPU hours	64 GPU hours

For adversarial training in Stage II, the reference images $x^\star$ are generated using an 8-step model distilled for less than 30 GPU hours by our distribution matching method. Compared to directly using the teacher model, it reduces the cost of generating $x^\star$ during training to only about 14% of the original computation. Moreover, the generated images exhibit richer details than those produced by the teacher model, which further improves the effectiveness of adversarial training.

B.3 Evaluation Details

In this section, we provide additional details on the evaluation metrics and baseline methods to further demonstrate the comprehensiveness and fairness of our comparison.

Metrics We employ a diverse set of evaluation metrics covering distribution fidelity, prompt alignment, perceptual quality, and human preference:

- **FID** [10] measures the distribution distance between 2 set of images in the Inception feature space. We compute FID between teacher samples and student samples to evaluate generative fidelity after distillation.
- **CLIP Score** [9]. We compute CLIP Score using the CLIP ViT-B/32 model to measure the semantic similarity between generated images and their corresponding text prompts.
- **PickScore** [15]. A learned preference model trained on large-scale human pairwise comparisons, designed to approximate human judgments of overall image quality and prompt consistency.
- **Aesthetic Score** [39]. An aesthetic predictor trained on LAION aesthetic annotations, focusing on visual appeal and photographic quality.
- **HPSv2** [47] uses a reward model to capture general human preferences for text-to-image generation.
- **ImageReward** [48] uses a reward model trained with RLHF to jointly evaluates image quality and prompt alignment.

Together with DPG-Bench [13], LPIPS-based diversity, and user study, we provide a comprehensive evaluation of generation performance from multiple perspectives.

Baselines We compare *1.x-Distill* against a broad set of publicly available few-step diffusion models based on *SD3-Medium* and *SD3.5-Large*. The evaluated baselines include trajectory-based, distribution-based and combined distillation approaches:

- **Hyper-SD** [33] is a trajectory distillation method that combines consistency trajectory distillation with human feedback learning. The released checkpoint of Hyper-SD3-Medium is a LoRA weight that preserves the CFG mechanism. In our evaluation, we set the LoRA scale and guidance scale to the default values of 0.125 and 5.0, respectively.
- **PCM** [44] is a consistency distillation method. In our evaluation, we use the official 4-step and 2-step deterministic checkpoints with t `shift` $= 1$.
- **Flash** [2] is a distillation method that combines distribution matching and adversarial training.
- **LADD (Turbo)** [37] is a latent-space adversarial distillation method applied to SD3.5-Large.
- **TDM** [29]is a representative distribution matching distillation method that outperforms DMD2 in quality and efficiency. So we choose it as our main baseline. Since TDM only releases the 4-step distilled model on SD3-Medium, we follow its official code and try our best to distill the 2-step model on SD3-Medium and SD3-Large.

C Extended Experiments

C.1 Compensation Module

We further study the learnable error-compensation module $f(\cdot)$, which is the key component for stabilizing block reuse in our cached few-step inference.

Setup Since *SD3-Medium* contains 24 DiT blocks, we fix the same cache setting as *1.x-Distill*-slow, where blocks 3–8 in the second denoising step are skipped and approximated. This corresponds to an effective NFE of 1.75. We study the compensation module from two aspects: different training settings for block-level caching, and different implementations of the compensation module $f(\cdot)$.

Training settings. We first compare three training settings around block-level caching.

- Full-computation baseline. This variant uses Stage I and Stage II with pixel-space adversarial refinement, but without caching. It serves as the reference without 1.x acceleration.
- Direct cache after distillation. This variant applies block reuse after distillation, without cache-aware adversarial refinement in Stage II. It evaluates whether caching can be directly applied to the distilled model in a nearly plug-and-play manner.
- Distill–Cache co-Training (DCT). This is our full training setting, where cache is explicitly incorporated into Stage II and optimized jointly with the generator along the cached inference path.

Compensation module designs. Based on the cache-aware training setting above, we further compare several implementations of $f(\cdot)$.

- No compensation. The cached contribution is directly reused without learnable correction.
- Simple residual MLP (segment-level). Our default design, using a lightweight residual MLP with LayerNorm and a two-layer GELU MLP. The hidden dimension is $2\times$ the input dimension, and the output layer is zero-initialized.
- Simple residual MLP (per-block). Separate simple MLP predictors for individual block deltas.
- Deeper residual MLP. A stronger predictor formed by stacking residual MLP blocks (expansion ratio 2, depth 2, dropout 0).
- Transformer proxy. One native DiT transformer block for residual delta prediction.

Analysis The quantitative results are reported in Table 7. We next analyze the effect of cache-aware training and the design choice of the compensation module.

Effect of training strategy. Table 7 shows that block-level caching is not directly transferable to extremely few-step distilled models. Directly applying cache after distillation reduces NFE and latency, but causes clear degradation on all preference-oriented metrics. This suggests that feature reuse is substantially more difficult in distilled two-step models, where adjacent steps exhibit larger feature drift and direct reuse introduces significant error. In contrast, incorporating cache into Stage II and optimizing it through DCT largely restores image quality, showing that cache acceleration in this regime must be learned jointly with the generator.

Effect of compensation design. The comparison among different compensation modules further shows that explicit learnable correction is necessary for stable cross-step reuse. Even under cache-aware training, directly reusing the cached contribution without $f(\cdot)$ still leaves a noticeable performance gap, indicating that joint optimization alone is insufficient.

Table 7: Ablation of cache-aware training and compensation module designs. *Top*: different training settings for introducing block-level caching. *Bottom*: different implementations of the compensation module $f(\cdot)$ under the full DCT setting.

Stage I	Stage II		$f(\cdot)$	#NFE	Latency(s)↓	CLIP↑	PS↑	IR↑	HPS↑
	Cache	PixGAN							
✓	✗	✓	–	2.00	0.7413	0.3184	22.6995	1.1601	33.0293
✓	✓	✗	–	1.75	0.6538	0.3170	22.2042	1.0152	30.9216
✓	✓	✓	None	1.75	0.6561	0.3205	22.3113	1.0500	31.0544
✓	✓	✓	**Simple MLP (seg.)**	1.75	**0.6617**	**0.3208**	**22.5161**	**1.1312**	**32.2550**
✓	✓	✓	Simple MLP (per-block)	1.75	0.6649	0.3196	22.4728	1.1184	32.0713
✓	✓	✓	Deep ResMLP	1.75	0.6685	0.3207	22.4986	1.1249	32.1462
✓	✓	✓	1 DiT block	1.75	0.6768	0.3214	22.4415	1.1197	32.3826

Among all variants, the **simple residual MLP (segment-level)** provides the best overall trade-off. It restores most of the lost quality while remaining lightweight and stable to optimize. In comparison, the **per-block MLP** offers no clear advantage over segment-level prediction, the **deeper residual MLP** brings only marginal improvement, and the **Transformer proxy** fails to yield consistent gains despite its higher complexity. These results suggest that the correction needed for cross-step block reuse is relatively simple, and increasing the capacity of $f(\cdot)$ offers limited practical benefit.

C.2 Block Selection

We further study how to choose cached blocks, since block selection is critical to the quality–efficiency trade-off in our 1.x inference regime.

Setup All experiments in this section are conducted on *SD3-Medium* distilled to 2-step sampling. We use the simple residual MLP as in the previous subsection Sec.C.1 for error compensation and compare five cache settings, including three contiguous ranges, i.e., blocks 3–8, 10–15, and 16–21, and two mixed settings with the same number of cached blocks, i.e., blocks 3–6 with 10–13, and blocks 10–13 with 16–19. To study block sensitivity, we first directly apply cache after Stage I at inference time, as shown in Fig.12, and then verify whether the same trend remains after training.

Fig. 12: Effect of caching different block ranges. Early-block caching causes relatively mild degradation, while middle and late blocks lead to severe artifacts, consistent with their larger reuse error. With learnable compensation $f(\cdot)$, caching early blocks largely preserves image quality.

Table 8: Ablation of block selection on *SD3-Medium*. The first row reports the full two-stage SFD model without cache acceleration. The remaining rows compare different cached block ranges using the same simple residual MLP for error compensation.

Stage	Cached blocks	#NFE	Latency(s)↓	CLIP↑	PS↑	IR↑	HPS↑
SFD w/o cache	–	2.00	0.7413	0.3184	22.6995	1.1601	33.0293
DCT	**3–8**	1.75	**0.6617**	**0.3208**	**22.5161**	**1.1312**	**32.2550**
DCT	10–15	1.75	0.6620	0.3196	22.1887	1.0218	30.9846
DCT	16–21	1.75	0.6621	0.3179	21.7315	0.8734	29.2148
DCT	3–6 + 10–13	1.75	0.6627	0.3201	22.3419	1.0726	31.5327
DCT	10–13 + 16–19	1.75	0.6628	0.3188	21.9642	0.9481	30.1029

Analysis As shown in 5(a), we measure the block-wise reuse error on *SD3-Medium* as the contribution change across adjacent denoising steps

$$e_n = \|\Delta_{n,t+1} - \Delta_{n,t}\|_1, \qquad \Delta_{n,t} = O_{n,t} - I_{n,t}.$$

Early blocks consistently exhibit smaller reuse error, indicating stronger temporal redundancy, whereas later blocks show much larger reuse error.

The direct-cache results closely follow the reuse-error curve. As shown in Table 8, caching blocks 3–8 causes the smallest degradation, while caching blocks 10–15 and especially 16–21 lead to much larger quality drop, showing that early blocks are more suitable for reuse than later ones. The mixed settings show the

same trend. Although blocks 3–6 + 10–13 and blocks 10–13 + 16–19 cache the same number of blocks, their performance still differs noticeably, again following the reuse-error curve rather than the cache ratio alone. This suggests that uncached blocks cannot reliably absorb the distortion introduced by high-error cached ranges.

Based on these observations, we choose blocks 3–8 for *1.x-Distill*-slow, and further extend the cached range to blocks 3–10 for the more aggressive *1.x-Distill*-fast setting.

C.3 Training Objectives for DCT

We further investigate the effect of incorporating additional knowledge distillation (KD) objectives in Distill–Cache co-Training (DCT). Inspired by previous diffusion pruning works, we consider two commonly used KD formulations: a feature-level KD objective and an output-level KD objective.

The feature-level KD objective encourages the predicted block contribution produced by the MLP to match the ground-truth contribution of the skipped blocks:

$$\mathcal{L}_{\text{feat}} = \mathbb{E}\,\|\Delta_1 - f(\Delta_0)\|_2^2\,. \tag{7}$$

The output-level KD objective directly constrains the prediction of the cached model to match the full-computation model. Let $v(x_t, t)$ denote the velocity prediction of the original model and $v^{\text{cache}}(x_t, t, f(\Delta_0))$ denote the prediction of the cached model using the reused block contribution predicted by $f(\Delta_0)$. The output-level KD loss is defined as:

$$\mathcal{L}_{\text{out}} = \mathbb{E}\left\|v(x_{t_1}, t_1) - v^{\text{cache}}(x_{t_1}, t_1, f(\Delta_0))\right\|_2^2. \tag{8}$$

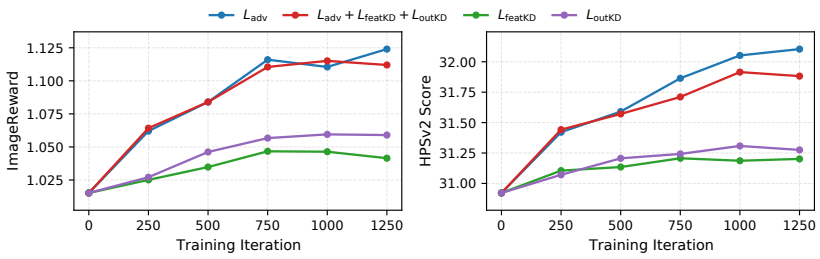


Fig. 13: Effect of different training objectives in DCT. Experiments are conducted on SD3-Medium with a cache block 3–8, using identical training configurations.

We compare different training objectives for DCT, including adversarial loss only ($\mathcal{L}_{\text{adv}}$), adversarial loss with both KD objectives ($\mathcal{L}_{\text{adv}} + \mathcal{L}_{\text{feat}} + \mathcal{L}_{\text{out}}$), and the KD objectives individually. As shown in Fig. 13, adding feature-level and output-level KD does not provide noticeable improvement, while using the KD objectives alone leads to significantly worse performance. These results indicate

that pixel-space adversarial supervision already provides an effective signal for correcting cache-induced errors, and additional KD constraints are unnecessary in our setting.

D Additional Visual Results

Figure 14 presents the visual comparison of diversity on SD3-Medium. Compared with DMD-like methods, our approach improves sample diversity while maintaining generation quality and prompt alignment. Further comparisons on SD3-Medium and SD3.5-Large are provided in Fig. 15 and Fig. 16, respectively. Even with only 1.x NFE sampling, *1.x-Distill* produces images with rich details and strong visual realism.

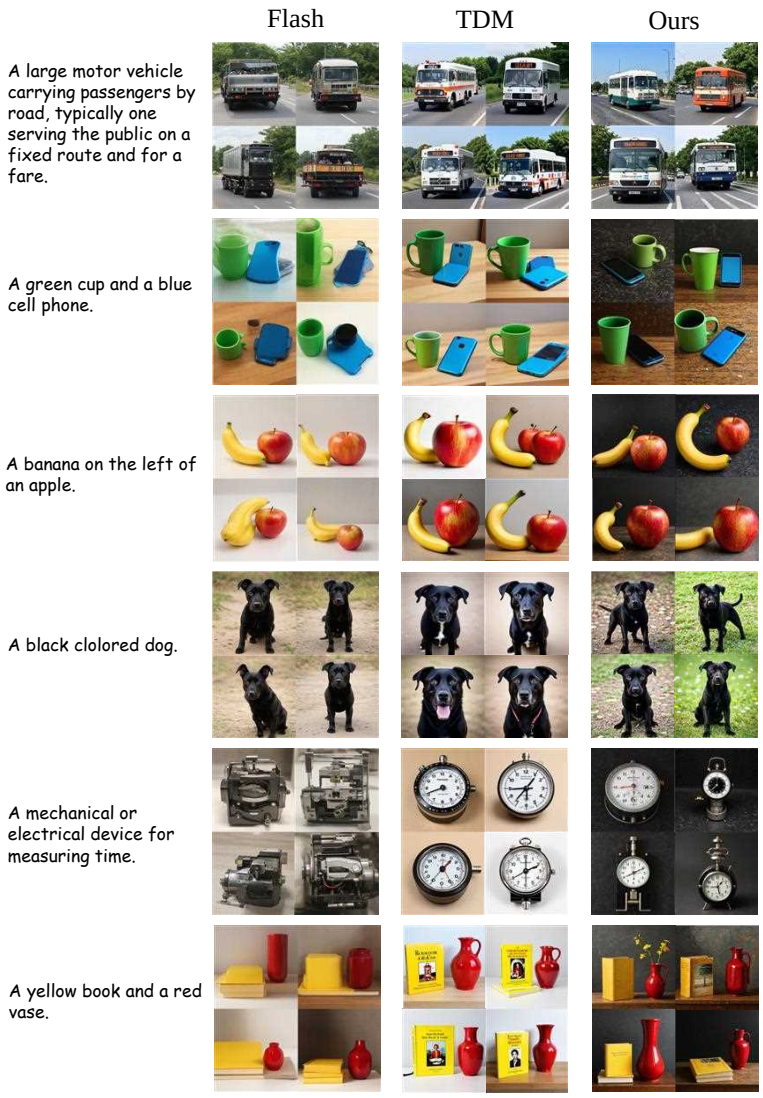


Fig. 14: Visual comparison of diversity under the 4-NFE setting distilled from SD3-Medium. Compared with two distribution-matching baselines, our approach produces more diverse samples while maintaining generation quality and prompt alignment.

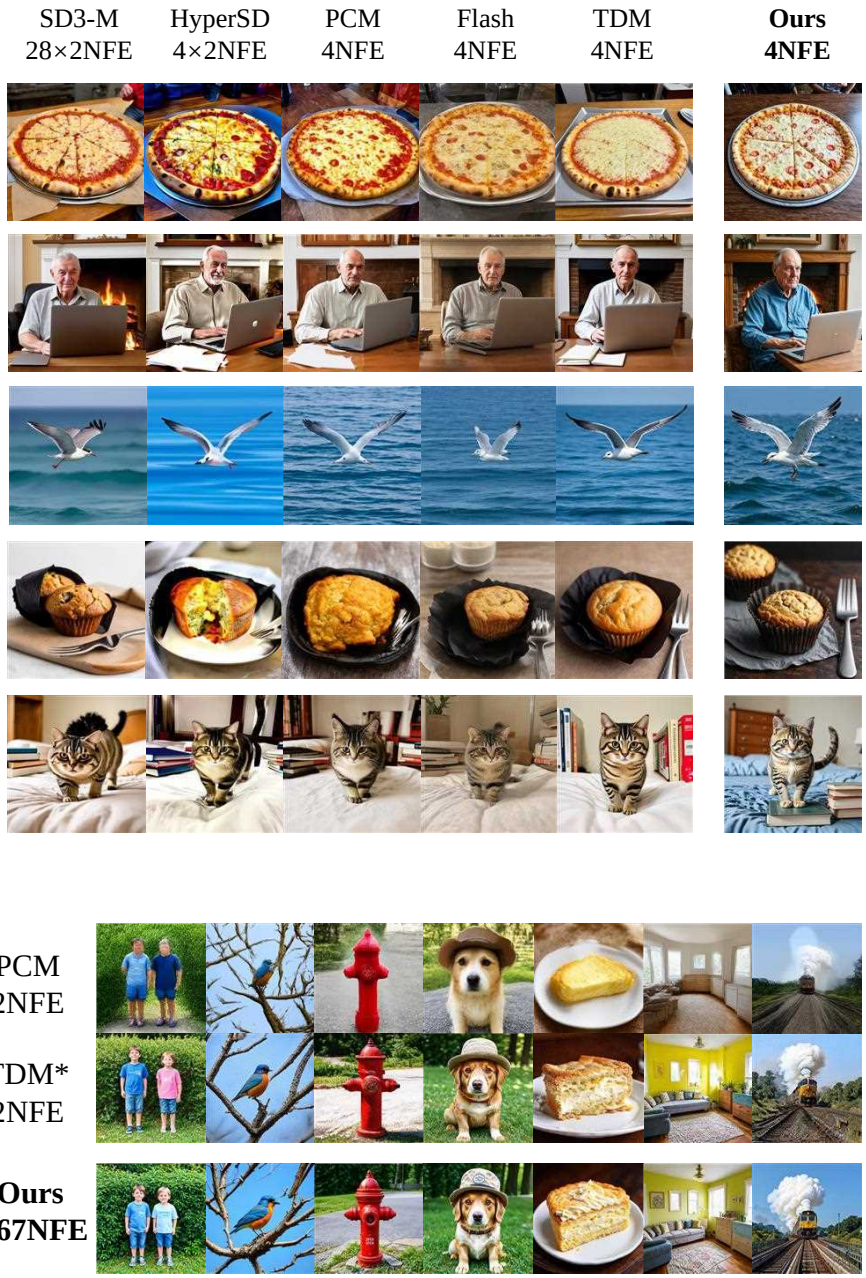


Fig. 15: Additional qualitative comparison on SD3-Medium. Even with only 1.x NFE sampling, *1.x-Distill* produces images with more realistic details than existing few-step baselines. Please zoom in for details.

Fig. 16: Additional qualitative comparison on SD3.5-Large. Even with only 1.x NFE sampling, *1.x-Distill* produces images with more realistic details than existing few-step baselines. Please zoom in for details.