UltraDiff: Transferring High-Fidelity Priors to Compressed Latent Spaces for High-Resolution Image Generation

Jingjing Ren[1,*] **Haitian Zheng**[2,*] **Haoyu Chen**[1] **Connelly Barnes**[2]
Zhifei Zhang[2] **Yuqian Zhou**[2] **Zhe Lin**[2] **Lei Zhu**[1,†]

[1]The Hong Kong University of Science and Technology (Guangzhou)
[2]Adobe Research

[*]Equal contribution [†]Corresponding author

Abstract

High-resolution synthesis with Latent Diffusion Models (LDMs) faces a fundamental trade-off between computational cost and generative fidelity. This trade-off is largely governed by the latent compression factor f (e.g. f8: $f{=}8$ and f32: $f{=}32$). While aggressive compression (f32) significantly improves efficiency, it suffers from severe information loss and ambiguity compared to standard f8 models. We propose UltraDiff, a framework that reformulates this challenge as joint-latent diffusion learning followed by marginal score distillation. We introduce MixTeacher, a dual-branch DiT that models the joint f8-f32 distribution, utilizing shared attention to resolve the ambiguity of the compressed space with high-fidelity f8 priors. Crucially, we propose vector field distillation in the native f32 space. Theoretically, this compels the student to learn the conditional expectation of the teacher's trajectory, effectively filtering out compression artifacts. Across the evaluated 1K–4K settings, UltraDiff approaches f8-level quality while preserving f32 efficiency. With 20 sampling steps on a single A100 GPU, it generates a 2K image in 6.2 seconds—over $40\times$ faster than our f8 baseline.

1 Introduction

The demand for ultra-high-definition (UHD) image and video synthesis has surged across streaming, gaming, and VR applications. While recent Diffusion Transformers (DiTs) [Esser et al., 2024, Chen et al., 2023, Podell et al., 2023, Black Forest Labs, 2024] excel at high-fidelity generation, self-attention dominates their compute and scales quadratically with the token count ($O(N^2)$), making UHD synthesis prohibitively expensive. To mitigate this, recent works [Team, 2025, Xie et al., 2024] explore VAEs with more aggressive spatial downsampling, i.e., increasing the latent downsampling factor from f8 ($f{=}8$) to f32 ($f{=}32$), reducing the 2D latent token count by $16\times$ compared to a standard f8 VAE, offering substantial efficiency gains.

However, the efficiency of aggressive downsampling comes at the cost of a substantially harder diffusion problem [Skorokhodov et al., 2025]. With f32 compression, the diffusion model must synthesize complex scenes from far fewer spatial tokens, yielding an ambiguous target distribution for the diffusion model to learn. This makes it difficult for the model to capture fine-grained details and precise spatial layout, leading to degraded perceptual fidelity and weaker prompt adherence compared to f8-based models [Black Forest Labs, 2024, Li et al., 2024, Kong et al., 2024].

Given the superior fidelity and detail generation capacity of f8 models, a natural question arises: **Can we transfer the rich generative knowledge of a pre-trained f8 model to guide the training of an efficient f32 model?** To study this, we conduct a controlled comparison between an f32 DiT trained

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



Figure 1: Our UltraDiff generates high-quality images across a variety of scenes, showcasing rich details and diverse content. Please zoom in to view the details. Additional images are provided in the supplementary materials.

from scratch and one initialized from a pre-trained f8 model. As shown in Fig. 2, f8 initialization consistently improves performance, indicating that generative knowledge indeed transfers across diffusion model with two latent spaces, despite the mismatch in latent representation, dimensionality and resolution. However, from Fig. 2, the benefit of initialization quickly saturates, leaving a substantial performance gap between the two models. This suggests that static parameter initialization alone is insufficient; and effective training requires dynamic, state-dependent guidance. Instead, a more natural solution is generative knowledge distillation where most existing methods [Leng et al., 2025, Ren et al., 2025a] rely on feature alignment, forcing a student to match intermediate representations of a trained teacher model. However, we argue that *static feature alignment* are suboptimal when distilling across mismatched latent resolutions, as aligning an f32 student to an f8 teacher requires interpolating feature grids, leading to quantization artifacts and checkerboard patterns. Moreover, such implicit feature-level regularization does not provide explicit distillation

supervision, which limits their ability to transfer generative knowledge from a pre-trained diffusion teacher.

To address these challenges for efficient UHD synthesis, we propose UltraDiff, a new framework that learns diffusion in a joint f8–f32 latent space and then distills the resulting score to the f32 latent space, yielding the final diffusion model in the f32 marginal for efficient sampling. Specifically, instead of treating the diffusion modeling in f8 and f32 spaces as two separate tasks, we introduce **MixTeacher**, a dual-branch DiT that models the joint distribution $p(\boldsymbol{x}^{\text{f8}}, \boldsymbol{x}^{\text{f32}})$ where $\boldsymbol{x}^{\text{f8}}$ and $\boldsymbol{x}^{\text{f32}}$ are extracted from the same image. Architecturally, MixTeacher comprises two DiT branches with separate parameters to accommodate the distinct feature distributions in the f8 and f32 latent spaces, while sharing self-attention layers within each block to enable dense cross-space interaction. With f8 initialization and layer-wise attention sharing, MixTeacher effectively learns a joint denoising model over paired f8 and f32 latents, effectively transferring generative knowledge from the pre-trained f8 branch to the f32 branch. Crucially, unlike other implicit feature distillation methods [Leng et al., 2025, Ren et al., 2025a], it enables direct distilling supervision in the f32 space. Specifically, our teacher projects its high-fidelity guidance into the student's f32 space. Theoretically, this forces the student to learn the marginal vector field, effectively approximating the conditional expectation of the high-fidelity trajectory. This process "integrates out" the ambiguity inherent to compressed generation, guiding the student toward structurally coherent manifolds while operating strictly within the efficient f32 latent space.

Our contributions are summarized as follows:

- We introduce MixTeacher, a unified architecture that learns the joint score function of f8 and f32 latents. Through a stochastic routing strategy and shared attention mechanism, it captures cross-space correlations while maintaining the marginal generation capabilities of each branch.
- We develop a marginal score distillation strategy that aligns the vector field directly in the native f32 space. By learning the conditional expectation of the teacher's trajectory, the student model effectively resolves the inherent ambiguity and difficulty of modeling complex dependencies in the compressed f32 space.
- Our method, UltraDiff, achieves a strong trade-off between quality and efficiency. It produces detail-rich 4K images with strong semantic alignment shown in Figure 1, approaches the fidelity of the evaluated f8 systems, and generates a 2K image in 6.2 seconds using 20 sampling steps on a single NVIDIA A100 GPU—$40.3\times$ faster than our f8 anchor. UltraDiff also scales to 4K generation, where the f8 anchor runs out of memory.

2 Related Work

2.1 Diffusion Transformer

Diffusion Transformers (DiTs) have become a widely used backbone for image and video synthesis [Chen et al., 2023, 2024, 2025, Black Forest Labs, 2025, Team, 2025, Li et al., 2024, 2026, Xiao et al., 2026, Ren et al., 2025b] because of their scaling behavior and strong generative performance [Peebles and Xie, 2023]. A typical system pairs a DiT with a variational autoencoder (VAE), which maps an image of size $3 \times H \times W$ to a latent tensor of size $c \times H/f \times W/f$. Self-attention then operates on the latent tokens, with computational and memory costs that grow quadratically with the sequence length [Chen et al., 2023, Ren et al., 2025a, Esser et al., 2024, Team, 2025, Ren et al., 2025c, 2026, Tian et al., 2024].

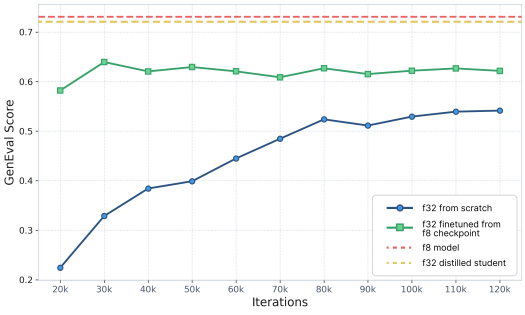

Figure 2: **Effectiveness of Cross-Space Knowledge Transfer.** GenEval scores during training. UltraDiff bridges the gap to the f8 teacher.

Most systems use an $8\times$ spatial compression factor and often reduce the token count further with 2×2 patch embedding [Podell et al., 2023, Chen et al., 2023, 2024, Li et al., 2024, Esser et al., 2024,

Black Forest Labs, 2024]. More aggressive compression, such as $f = 32$, improves efficiency for high-resolution and long-sequence generation [Xie et al., 2024, HaCohen et al., 2024, Team, 2025]. However, the smaller spatial grid makes structure and semantic alignment harder to learn [Xie et al., 2024, HaCohen et al., 2024, Pernias et al., 2023]. UltraDiff addresses this problem by transferring knowledge from a strong f8 model to an f32 model through initialization and cross-latent distillation, followed by mixed-resolution training.

Recent work has also explored making one autoencoder latent space explicitly compatible with another during VAE training [Yi et al., 2025, Zhao et al., 2024]. UltraDiff addresses a complementary setting: our f8 and f32 VAEs are trained independently and then kept fixed, while their relationship is learned at the diffusion vector-field level through shared attention in MixTeacher. Explicit compatible-VAE training may further simplify cross-space transfer and is an interesting direction for future work.

2.2 Diffusion Distillation

Distillation has become a general mechanism for adapting diffusion models to diverse objectives and deployment constraints. For acceleration, progressive and distribution–matching distillation compress long sampling trajectories into few- or even one-step generators while maintaining competitive fidelity [Salimans and Ho, 2022, Yin et al., 2024a,b, Kang et al., 2024]. Related consistency and rectified-flow formulations further enable fast sampling and have been applied in high-resolution restoration pipelines [Xu et al., 2025, Sun et al., 2025, Wang et al., 2024b]. Beyond speed, distillation can change the inference pattern or model architecture. For example, a bidirectional teacher can supervise a causal student for streaming video generation [Yin et al., 2024c, Huang et al., 2025, Novack et al., 2024, Shin et al., 2025, Kodaira et al., 2025]. Distillation also supports modality transfer and prior reuse; score-distillation techniques leverage strong 2D text-to-image priors to supervise 3D synthesis, with recent work addressing stability and mode collapse [Yu et al., 2023, Ma et al., 2024, Wang et al., 2024a, 2025]. Another class of methods aligns semantic features from external encoders, such as DINOv2 or other diffusion networks, with a model's internal representations to accelerate training and improve image fidelity [Yu et al., 2024, Leng et al., 2025, Ren et al., 2025a]. We adopt the spirit of distillation to bridge a latent-scale mismatch: a strong f8 teacher guides an f32 DiT via initialization and cross-latent supervision, yielding a high-compression student that retains fidelity while benefiting from the computational savings of far fewer tokens.

3 Method

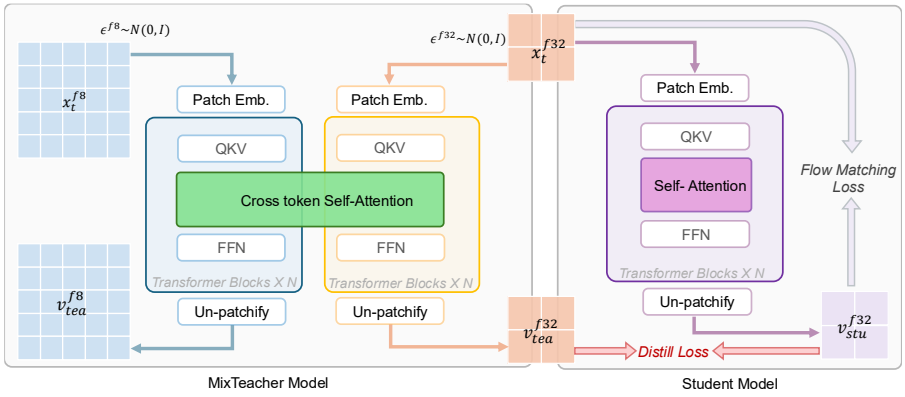

Figure 3: **Overview of UltraDiff.** (Left) The **MixTeacher** models the joint distribution of f8 and f32 latents via a dual-branch DiT with shared attention. It acts as an omniscient supervisor that sees both high-fidelity (f8) and high-efficiency (f32) views of the data. (Right) The **Student** is a standard f32 DiT trained via *Marginal Score Distillation*. It learns to approximate the teacher's optimal denoising direction projected onto the f32 subspace with a flow-matching objective.

We formulate the problem of high-efficiency, high-fidelity synthesis as joint-latent diffusion learning followed by marginal score distillation. Our goal is to train an efficient student model that operates

in the high-compression space but inherits the generative vector field of the high-fidelity joint distribution.

Motivation: Feasibility of Cross-Space Transfer. Before detailing our framework, we first empirically validate the hypothesis that the generative representations of f8 (high-fidelity) and f32 (high-efficiency) latent spaces share transferable structural patterns, despite their representation and dimensional mismatch. We conducted a pilot study comparing a baseline f32 DiT model trained from scratch against one initialized with parameters from a pre-trained f8 model (excluding latent-dependent layers). As shown in Fig. 2, the f8-initialized model consistently outperforms the randomly initialized baseline. This confirms that priors learned in the f8 space are transferable. However, a persistent performance gap remains. This saturation suggests that static parameter initialization is insufficient. This motivates our proposal of MixTeacher, which enables dynamic, interaction-based knowledge transfer via joint latent modeling.

3.1 Joint-Latent Modeling via MixTeacher

To capture the correlation between high-frequency details (f8) and compact semantics (f32), we introduce MixTeacher, a unified diffusion model parameterized by Φ. We consider the data as tuples in a product space $\mathcal{X}_{\text{joint}} = \mathcal{X}_{\text{f8}} \times \mathcal{X}_{\text{f32}}$. The MixTeacher approximates the joint score function $\nabla \log p(\mathbf{x}^{\text{f8}}, \mathbf{x}^{\text{f32}})$. To provide reliable cross-space guidance, we design MixTeacher as a mixed-token dual-branch DiT. As illustrated in Fig. 3, the f8 and f32 branches maintain separate parameters—including QKV projections, FFNs, normalization layers, patch embeddings, and output heads—but share the self-attention mechanism across tokens in each layer. This design enables cross-latent space interaction while respecting the distinct distributions of each space.

Formally, given an input image $\mathbf{I} \in \mathbb{R}^{3\times H\times W}$, we extract its f8 and f32 latents via respective VAE encoders: $\mathbf{x}_0^{\text{f8}} \in \mathbb{R}^{C_8\times \frac{H}{8}\times \frac{W}{8}}$ and $\mathbf{x}_0^{\text{f32}} \in \mathbb{R}^{C_{32}\times \frac{H}{32}\times \frac{W}{32}}$. During training, the noisy latents $\mathbf{x}_t^{\text{f8}}$ and $\mathbf{x}_t^{\text{f32}}$ are fed to the MixTeacher. Inside the model, each branch processes its own tokens using its specific parameters. Crucially, the resulting token sets are concatenated and passed through the shared self-attention at every layer. This enables dense interaction between the two modalities, allowing the f32 tokens to incorporate fine-grained structural details in the f8 tokens. The model output is formulated as the predicted velocity field $\mathbf{v}$, which is supervised by the target vector field $\mathbf{u}$ derived from the forward flow matching process.

Training Objective and Stochastic Routing. We train the MixTeacher using a flow-matching objective [Lipman et al., 2022]. To capture the joint distribution while maintaining generative capabilities of individual branches over the f8 or f32 marginals, we employ a stochastic routing strategy. Formally, we optimize a weighted objective across three topological configurations defined by a branch indicator mask $(m_{\texttt{f8}}, m_{\texttt{f32}}) \in \{(1,1),(1,0),(0,1)\}$, indicating the activation status of the f8 and f32 branches, respectively. The unified loss function is:

$$\mathcal{L} = \mathbb{E}_{t,\mathbf{x},(m_{\texttt{f8}},m_{\texttt{f32}})}\left[\sum_{k\in\{\texttt{f8},\texttt{f32}\}} m_k \cdot \|\mathbf{v}_\Phi^k(\mathbf{x}_t^k) - \mathbf{u}_t^k\|^2\right], \tag{1}$$

where $m_k \in \{0,1\}$ controls whether loss is applied on the specified latent space, $\mathbf{v}_\Phi^k$ is the vector field predictor parametrized by Φ and $\mathbf{u}_t^k$ is the target vector field.

More specifically, we sample among: (i) **Joint Mode** ($m_{\texttt{f8}} = 1, m_{\texttt{f32}} = 1$) while self-attention between two branches is enabled to learn the score jointly; (ii) **f8-Solo** ($m_{\texttt{f8}} = 1, m_{\texttt{f32}} = 0$), preserving valid generation in the original f8 space; and (iii) **f32-Solo** ($m_{\texttt{f8}} = 0, m_{\texttt{f32}} = 1$), which is crucial for ensuring the generation capability of the f32 branch without conditioning on f8 tokens. The final optimization objective is the expectation over these routing modes, governed by the sampling probabilities $p_{\text{joint}}, p_{\text{f8}}, p_{\text{f32}}$. This strategy prevents the "co-adaptation" pitfall where the f32 branch might become overly reliant on f8 features, ensuring that the structural knowledge transfer is robust and generalizable.

3.2 Distilling the Marginal Score

Once trained, MixTeacher provides a strong teacher signal for supervising the f32 student. Specifically, we aim to distill the knowledge of MixTeacher into a student model θ that operates *only* in the f32

marginal space, while the student aims to mimic the vector field predicted by the teacher's f32 branch. However, since the student lacks access to the high-fidelity f8 latent $\mathbf{x}_t^{\text{f8}}$ utilized by the teacher, the distillation target is inherently stochastic from the student's perspective. We formulate the objective as minimizing the expected discrepancy over the joint distribution of the f8 and f32 latents:

$$\mathcal{L}_{\text{distill}}(\theta) = \mathbb{E}_{t,(\mathbf{x}_t^{\text{f32}},\mathbf{x}_t^{\text{f8}})\sim q_t}\left[\left\|\mathbf{v}_\theta(\mathbf{x}_t^{\text{f32}},t) - \mathbf{v}_\Phi^{\text{f32}}(\mathbf{x}_t^{\text{f32}},\mathbf{x}_t^{\text{f8}},t)\right\|^2\right]. \tag{2}$$

Minimizing this L_2 objective inherently forces the student to approximate the conditional expectation of the teacher's output given the $f32$ observation, i.e., the optimal estimator $\mathbf{v}_\theta^*$ is thus given by:

$$\mathbf{v}_\theta^*(\mathbf{x}_t^{\text{f32}},t) = \mathbb{E}_{q(\mathbf{x}^{\text{f8}}|\mathbf{x}^{\text{f32}})}\left[\mathbf{v}_\Phi^{\text{f32}}(\mathbf{x}_t^{\text{f32}},\mathbf{x}_t^{\text{f8}},t)\right]. \tag{3}$$

The student learns the marginal vector field projected from the joint space. By aggregating supervision over the coherent f8 priors provided by the teacher, this projection acts as a structural filter that effectively "integrates out" the ambiguity and stochastic aliasing artifacts inherent to the f32 space. The student is thus guided toward a structurally consistent manifold rather than fitting the noisy, underspecified f32 distribution directly.

Crucially, this projection does not result in a blurry average. As visualized in Fig. 4, in the mixed routing mode, the output of the teacher's f32 branch is visually sharp and indistinguishable from the f8 branch. This indicates that the teacher's attention sharing mechanism successfully resolves ambiguity, collapsing the conditional distribution into a highly peaked, determinate signal. Consequently, the distillation target $\mathbf{v}_\Phi^{\text{f32}}$ serves as a structurally determinate vector field, providing a clean supervision signal that enables the student to recover fine details despite the high compression ratio. We train the student model using a composite objective of distillation loss and flow matching loss:

$$\mathcal{L}_{\text{total}} = \lambda\mathcal{L}_{\text{distill}} + \mathbb{E}_{t,\mathbf{x}}\left[\|\mathbf{v}_\theta(\mathbf{x}_t^{\text{f32}}) - \mathbf{u}_t^{\text{f32}}\|^2\right]. \tag{4}$$

This comparison separates the effect of the proposed training procedure from the strength of the pretrained f8 anchor. Directly reusing the f8 parameters improves the f32 model from 0.54 to 0.61 GenEval, but leaves a substantial gap to the 0.73 f8 anchor. With the same f8 initialization, f32 VAE, and DiT backbone, UltraDiff reaches 0.72. The gain from 0.61 to 0.72 therefore cannot be explained by initialization or VAE quality alone and instead measures the contribution of joint-latent modeling and marginal score distillation.

Table 1: **Controlled comparison under the same framework.** All f32 variants use the same f32 VAE and DiT backbone. Initialization alone transfers part of the f8 prior, whereas marginal score distillation recovers most of the left gap.

Method	CLIP↑	IR↑	GenEval↑	Token
f32 from scratch	27.21	0.70	0.54	$\frac{H}{32}\times\frac{W}{32}$
f32 after f8 initialization	28.91	0.91	0.61	$\frac{H}{32}\times\frac{W}{32}$
UltraDiff f32 student	29.27	0.98	0.72	$\frac{H}{32}\times\frac{W}{32}$
f8 anchor	29.25	1.08	0.73	$\frac{H}{8}\times\frac{W}{8}$

4 Experiment

4.1 Implementation Details

We employ a Diffusion Transformer (DiT) architecture that models text and image tokens as a unified sequence. Specifically, text embeddings (via T5-XXL [Raffel et al., 2020]) and image latents are concatenated along the sequence dimension and processed through a shared bidirectional transformer backbone. We use a patch size of 1 for image embeddings. For latent encoding, we utilize high-capacity VAEs with latent channel dimensions of $C_{\text{f8}} = 16$ and $C_{\text{f32}} = 64$, corresponding to $8\times$ and $32\times$ downsampling factors, respectively. We apply channel-wise unit-variance normalization to both f8 and f32 latents before they enter the DiT backbone. We first train a DiT in the f32 latent space under two settings: (i) from scratch, and (ii) initialized from the f8 model's weights, except for the shape-dependent patch embedding and output heads, which are randomly initialized. For MixTeacher, the f8 branch is initialized from a strong pretrained f8 T2I model, and the f32 branch is initialized from the previously trained f32 model with f8 initialization. We train the teacher at 512 and 1024 resolutions with varying aspect ratios. All UltraDiff stages use 64 NVIDIA A100 GPUs with FSDP. After filtering and deduplication, the available corpus contains approximately 0.37B text–image pairs. Table 2 reports the optimization steps and aggregate compute for each stage. The three stages consume 463.0 GPU-days in total, about 26% of the 1792 GPU-days required to train our lower-quality f32 baseline directly. These costs treat the fixed f8/f32 VAEs and pretrained f8 anchor as available components; further optimization settings are provided in the appendix.

Table 2: **Stage-wise training resources.** Compute is aggregated over 64 NVIDIA A100 GPUs.

Stage	Steps	Compute	Output
f32 transfer	120K	66.7 GPU-days	f32 initialization
MixTeacher training	5K	29.6 GPU-days	Joint f8–f32 teacher
Student distillation and scaling	50K + 10K	366.7 GPU-days	Standalone f32 student
UltraDiff adaptation total	185K	463.0 GPU-days	Final UltraDiff model



(a) f8 (Left) & f32 (right) – Mix (b) f8 – Solo (c) f32 – Solo

Figure 4: Visual results of different inference modes of the MixTeacher.

Table 3: MJHQ-3K and GenEval results for different MixTeacher modes. The f32 branch in mixed mode matches the performance of the base f8 model.

Method	CLIP↑	IR↑	GenEval↑	Token
f8 & f32 Mixed (f8 Branch)	29.27	1.05	0.74	$\frac{H}{8} \times \frac{W}{8}$
f8 & f32 Mixed (f32 Branch)	29.24	1.02	0.74	$\frac{H}{32} \times \frac{W}{32}$
f8-Solo	29.26	1.06	0.73	$\frac{H}{8} \times \frac{W}{8}$
f32-Solo	28.88	0.85	0.62	$\frac{H}{32} \times \frac{W}{32}$

4.2 Ablation Study

Evaluation criteria. We evaluate models using GenEval [Ghosh et al., 2023], CLIP similarity [Radford et al., 2021], and ImageReward (IR) [Xu et al., 2023] at a resolution of 512×512 for fast iteration. CLIP and IR are computed on a 3k subset of MJHQ-30k [Li et al., 2024] to further reduce evaluation cost.

Train in f32 space. We train the T2I model in the f32 latent space for 120k iterations until convergence. The results show that the model with f8 initialization exhibits much better performance than the model trained from scratch. As shown in Figure 2, the model initialized from f8 (green curve) consistently outperforms the scratch-trained model (blue curve) according to GenEval [Ghosh et al., 2023] throughout training. Table 1 reports similar gains on CLIP and IR metrics, showing the benefit of transferring parameters from the f8 model. However, simple initialization still leaves a large gap compared with the f8 model, as shown in Figure 2 and Table 1. This suggests that, in the highly compressed f32 space, good initialization helps but is not sufficient. Even with f8 initialization, optimization remains difficult, and the model cannot fully match the f8 teacher under a purely data-driven objective. Leveraging the strong f8 model, we distill its behavior to enhance f32 performance.

Exploration of MixTeacher Routing Strategy. To identify the optimal strategy for training the MixTeacher, we evaluate three routing configurations in Table 5. First, exclusive joint training ($p_{joint} = 1.0$) performs poorly; the lack of single-branch constraints causes the f32 branch to exploit f8 shortcuts, leading to representation collapse. Second, mixing only Joint and f8-Solo modes (e.g., $p_{joint} = 0.5, p_{f8} = 0.5$) also fails, as the f32 branch remains under-trained, creating a bottleneck that drags down the joint performance. These results underscore a critical insight: maintaining the independent generative capability of each branch is a prerequisite for learning a robust joint distribution. Consequently, we adopt a balanced configuration ($p_{joint} = 0.4, p_{f32} = 0.4, p_{f8} = 0.2$). This strategy explicitly enforces the marginal capability of the f32 branch, ensuring it provides a high-quality supervision target. We assign a lower weight to f8-Solo (0.2) since the strong pre-trained f8 branch requires less tuning, allowing us to allocate more budget to the f32 and joint objectives.

Analysis of MixTeacher Inference. Having determined the suitable routing configuration, we further analyze the inference behavior of the MixTeacher (Table 3). By comparing the inference modes, we observe that the Mixed Mode effectively bridges the gap between the coarse f32 latent space and the high-fidelity target. While the f32-Solo baseline (0.62 GenEval) yields suboptimal performance due to the aggressive compression, the Mixed Mode (0.74 GenEval) demonstrates that the teacher successfully utilizes the f8 context to resolve structural ambiguities. Crucially, as visualized in Figure 4, the images generated by the f32 and f8 branches in Mixed Mode are visually indistinguishable, matching in both high-frequency details and semantic alignment. This indicates that although the two branches reside in distinct latent spaces, the MixTeacher successfully projects

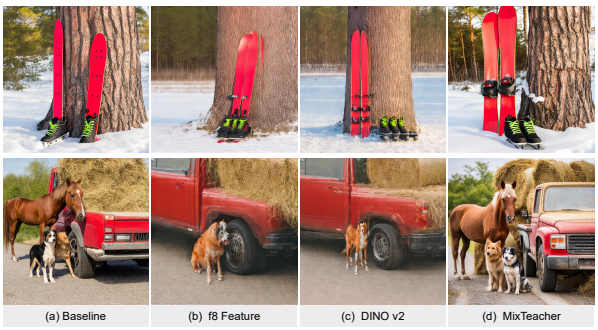

Figure 5: Visual comparison of (a) the f8 base model, (b) f32 model distilled from an f8 teacher [Ren et al., 2025a], (c) f32 model distilled from DINOv2 features via REPA [Yu et al., 2024], and (d) f32 model distilled from our MixTeacher.

Table 4: Results on MJHQ-3K and GenEval for different distillation methods.

Teacher	CLIP↑	IR↑	GenEval↑
Baseline	28.91	0.91	0.61
DINO v2 [Leng et al., 2025]	27.53	0.06	0.33
f8 model [Ren et al., 2025a]	27.87	0.05	0.31
MixTeacher (Ours)	29.27	0.98	0.72

Table 5: Impact of routing strategy on GenEval across inference modes.

Routing Strategy			Inference Mode (GenEval ↑)			
p_{mix}	p_{f32}	p_{f8}	f8-Solo	f32-Solo	Mix(f8)	Mix(f32)
1.0	0.0	0.0	0.56	0.28	0.68	0.68
0.5	0.0	0.5	0.65	0.15	0.65	0.64
0.4	0.4	0.2	0.73	0.62	0.74	0.74

Table 6: **Controlled attribution of the high-resolution gains.** Both models use the same f8-initialized f32 backbone and the same mixed-resolution scaling stage. DPG, GenEval, and HPSv3 are evaluated at 1K; FID, aesthetic score, and CLIP similarity are evaluated at 4K.

Model	DPG↑	GenEval↑	HPSv3↑	FID↓	Aesthetic↑	CLIP↑
No distillation + high-res	74.02	0.602	7.68	149.59	6.16	29.31
UltraDiff	**85.40**	**0.680**	**9.39**	**146.60**	**6.34**	**30.14**

the high-fidelity priors into the compressed f32 representation. Consequently, this f32 output serves as a high-quality, structurally determinate distillation signal directly within the student's native space.

Distillation strategy We explore four strategies in Table 4 and Fig. 5: (1) Baseline — an f32 model initialized from the f8 model (no distillation). (2) DINOv2 teacher — align the DiT's internal features to DINOv2 following REPA [Leng et al., 2025]. (3) f8 feature teacher — align f32 features to those from the f8 model following Ren et al. [2025a]. (4) MixTeacher (ours) — process f8 and f32 tokens together and use the f32 branch as the distillation signal. External semantic guidance from DINOv2 brings little gain. Our baseline has already been pre-trained in f8 and then fine-tuned in f32, so semantic priors are strong; additional supervision from a generic image encoder is redundant and not matched to the diffusion objective. Moreover, feature alignment to DINOv2 requires grid upsampling from f32, which induces checkerboard/ringing even with learned convolutional mapping. The problem is amplified under variable aspect ratios (tall/wide samples), where DINOv2 offers limited spatial guidance for the diffusion trajectories. Directly aligning to the f8 feature grid performs similarly or worse for the same reason: the f8 lattice is $4\times$ finer per dimension than f32, so supervision again depends on learned upsampling across mismatched token layouts, introducing aliasing and bias. In contrast, ours supervises in the f32 space using the f32 branch's v-prediction. This avoids cross-grid interpolation, matches the denoising objective, and yields sharper details and better text–image consistency, as seen qualitatively in Figure 5 and quantitatively in Table 4.

Distillation versus high-resolution training. To isolate the effect of marginal score distillation, we train a same-backbone control from the same f8-initialized f32 checkpoint, using identical high-resolution scaling but without distillation. During scaling, 1K samples use both distillation and flow matching, while 2K–4K samples use flow matching only. As shown in Table 6, UltraDiff improves DPG, GenEval, and HPSv3 by 11.38, 0.078, and 1.71 at 1K, respectively, while also achieving better FID, aesthetic score, and CLIP similarity at 4K. These results highlight the complementary roles of distillation in transferring the f8 prior and high-resolution flow matching in adapting to larger spatial scales.

4.3 Comparative Study

Evaluation Criteria We evaluate our UltraDiff and other methods using DPG Bench [Hu et al., 2024], GenEval [Ghosh et al., 2023] and human preference score (HPSV3) [Ma et al., 2025]. For quantitative evaluation, we adopt the protocols from Diffusion-4K [Zhang et al., 2025]. FID is computed on the entire Aesthetic-4K [Zhang et al., 2025] evaluation set, with images resized to

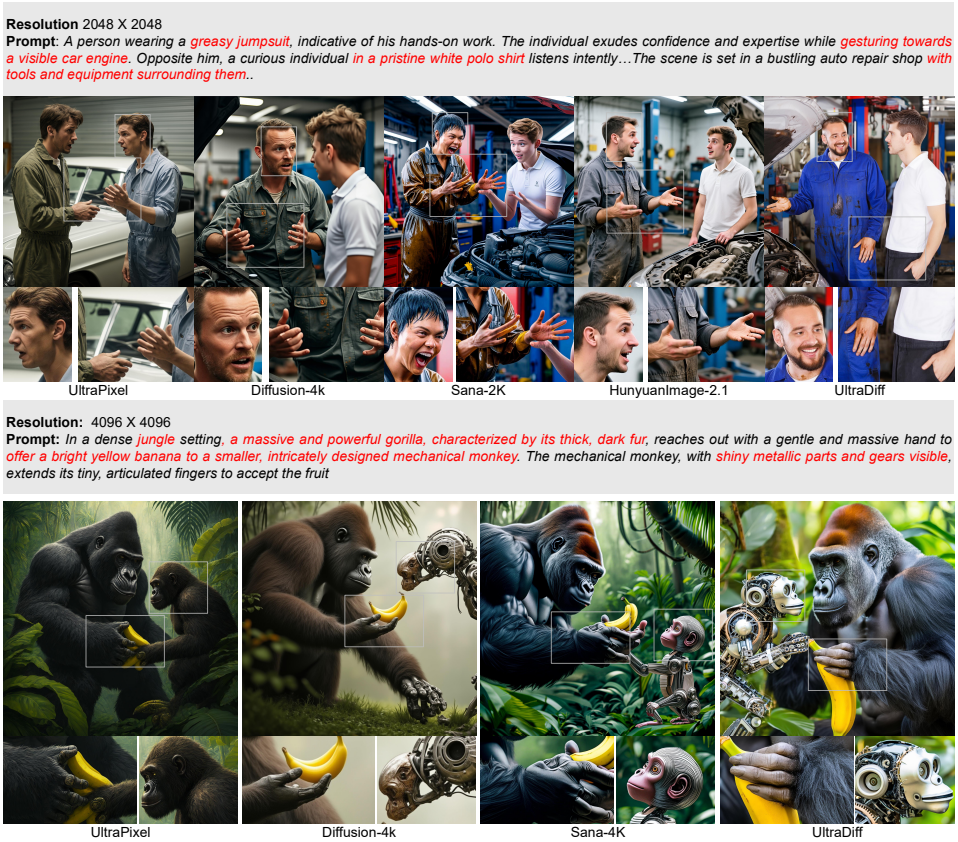

Figure 6: High-resolution image synthesis comparison at 2K and 4K. UltraDiff produces sharper details, cleaner textures, and stronger text–image alignment than competing high-resolution models.

Table 7: Comparison with other T2I models on higher resolution of 2048×2048 and 4096×4096.

Resolution	Method	Params(B)	Latent($C \times H \times W$)	Latency(s) / Step	FID↓	pFID↓	Aesthetic↑	CLIP↑
2048^2	Ultrapixel [Ren et al., 2024]	5.1	$16 \times \frac{H}{32} \times \frac{W}{32}$	12 / 20	46.60	43.03	6.24	30.21
	Diffusion-4k [Zhang et al., 2025]	12	$16 \times \frac{H}{42} \times \frac{W}{42}$	23 / 50	**37.79**	*42.43*	6.28	29.21
	Sana-2K [Xie et al., 2024]	1.6	$32 \times \frac{H}{32} \times \frac{W}{32}$	**5.8 / 20**	45.99	42.99	6.18	30.13
	Hunyuan-2.1 [Team, 2025]	17	$64 \times \frac{H}{32} \times \frac{W}{32}$	82 / 50	43.45	48.72	**6.44**	*30.17*
	Ours	2.0	$64 \times \frac{H}{32} \times \frac{W}{32}$	*6.2 / 20*	*39.52*	**42.17**	*6.40*	**30.53**
4096^2	Ultrapixel [Ren et al., 2024]	5.1	$16 \times \frac{H}{42} \times \frac{W}{42}$	73 / 20	155.17	57.14	5.91	27.39
	Diffusion-4k [Zhang et al., 2025]	12.0	$16 \times \frac{H}{32} \times \frac{W}{32}$	122 / 50	162.94	75.70	5.68	27.20
	Sana-4K [Xie et al., 2024]	1.6	$32 \times \frac{H}{32} \times \frac{W}{32}$	**21.3 / 20**	*149.20*	*56.25*	*6.12*	*29.97*
	Ours	2.0	$64 \times \frac{H}{32} \times \frac{W}{32}$	*28.3 / 20*	**146.60**	**54.27**	**6.34**	**30.14**

299×299. To assess high-frequency details, we report pFID, calculated by extracting 4 random non-overlapping patches from each ground truth and generated image.

Results at 1024^2. We compare UltraDiff with complete T2I systems based on f8 VAEs [Black Forest Labs, 2024, Podell et al., 2023, Chen et al., 2024, Li et al., 2024] and with the f32-based Sana system [Xie et al., 2024]. Table 8 shows that UltraDiff offers a strong practical quality–efficiency trade-off, ranking at or near the top among models of similar size on DPG and GenEval while remaining competitive on HPSv3. These cross-system results provide deployment-oriented context rather than a controlled causal comparison: the systems differ in backbone architecture, parameter count, training data, text encoders, autoencoders, optimization recipes, and inference infrastructure. We therefore establish the contribution of UltraDiff using the within-framework controls in Tables 1 and 6, where the relevant f32 models share the same VAE, DiT backbone, initialization, and high-resolution training protocol.

Results of 2048^2 and 4096^2. UltraDiff uses a high VAE compression ratio ($\frac{H}{32} \times \frac{W}{32}$), which enables 2K and 4K synthesis within one checkpoint that supports multiple resolutions natively. We compare 2K and 4K results against methods capable of high-resolution output, including approaches that scale pretrained models through architectural or inference-time adjustments, such as UltraPixel and Diffusion-4K, and systems that natively support such resolutions, such as Sana-2K, Sana-4K,

Table 8: Comparison with other T2I models on 1024^2.

Method	Token	Params(B)	DPG↑	GenEval↑	HPSv3↑
SDXL [Podell et al., 2023]	$\frac{H}{8} \times \frac{W}{8}$	2.6	74.6	0.55	7.87
PlayGroundv2.5 [Li et al., 2024]	$\frac{H}{8} \times \frac{W}{8}$	2.6	75.5	0.56	*9.90*
PixArtΣ [Chen et al., 2024]	$\frac{H}{16} \times \frac{W}{16}$	0.6	80.5	0.54	9.20
SD3-medium [Esser et al., 2024]	$\frac{H}{16} \times \frac{W}{16}$	2.0	84.1	0.62	9.08
FLUX-dev [Black Forest Labs, 2024]	$\frac{H}{16} \times \frac{W}{16}$	12	83.8	0.67	**10.15**
FLUX-schnell [Black Forest Labs, 2024]	$\frac{H}{16} \times \frac{W}{16}$	12	84.8	**0.71**	9.37
Sana [Xie et al., 2024]	$\frac{H}{32} \times \frac{W}{32}$	1.6	*84.5*	0.66	8.92
UltraDiff (Ours)	$\frac{H}{32} \times \frac{W}{32}$	2.0	**85.4**	*0.68*	9.39

and HunyuanImage-2.1, as shown in Figure 6 and Table 7. At 2K the post adjusted methods are competitive and UltraDiff maintains strong metrics with fast inference, while at 4K these adjusted methods degrade in quality, for example Ultrapixel struggles to preserve semantic alignment and Diffusion-4K begins to produce blurred results. Hunyuan-2.1, which employs a larger parameter budget with the same f32 compression, also performs well, suggesting that high compression scales effectively when paired with sufficient capacity. Both Sana and UltraDiff are efficient at high resolution. Sana is faster because it employs linear attention, whereas UltraDiff uses self-attention at a similar parameter scale and compression ratio. Under the reported system-level evaluation, UltraDiff obtains stronger overall image-quality and semantic-alignment metrics. Because the two systems differ in more than the attention mechanism, we do not interpret this comparison as isolating the effect of any single component.

Controlled inference efficiency. We compare the final f32 student with the f8 anchor using 20 sampling steps on a single NVIDIA A100 GPU. As shown in Table 9, f32 compression reduces the time per image from 27.3s to 3.2s at 1024^2 ($8.5\times$) and from 249.8s to 6.2s at 2048^2 ($40.3\times$). At 4096^2, the f8 model runs out of memory, whereas UltraDiff completes generation in 28.3s. Because both models use the same sampling setup, this comparison isolates the efficiency gained from the smaller f32 latent grid.

Table 9: **Controlled inference efficiency.** Time per image with 20 sampling steps on one NVIDIA A100 GPU.

Resolution	f8 anchor	UltraDiff	Speedup
1024^2	27.3s	3.2s	$8.5\times$
2048^2	249.8s	6.2s	$40.3\times$
4096^2	OOM	28.3s	–

5 Conclusion

In this work, we presented UltraDiff, a novel distillation framework designed to enable high-fidelity image synthesis within highly compressed latent spaces (e.g., f32). To overcome the inherent difficulty of training in such compact representations, we introduced MixTeacher, a dual-branch architecture that establishes a direct, grid-aligned supervision signal for the student model. By distilling the marginal vector field from a high-quality anchor directly into the target latent space, our method effectively transfers complex structural details. Extensive experiments demonstrate that UltraDiff substantially narrows the quality gap between f8 and f32 generation in the evaluated DiT pipeline while retaining the inference efficiency of f32 architectures. With 20 sampling steps on a single A100 GPU, UltraDiff generates a 2K image in 6.2 seconds, a $40.3\times$ speedup over the f8 anchor, and remains practical at 4K where the anchor runs out of memory. We hope this streamlined distillation paradigm paves the way for more scalable and efficient generative models in the future.

Acknowledgment. This work is supported by the National Natural Science Foundation of China (Project No.82572383), Adobe Research Gift Grant, and Guangdong Basic and Applied Basic Research Foundation (No.2025A1515140021).

Impact Statement. UltraDiff enables efficient, high-quality text-to-image generation in highly compressed latent spaces, potentially reducing the computational and energy costs of 2K–4K image synthesis and broadening access to high-resolution generative models. However, this efficiency may also facilitate the creation of misleading or harmful content. We therefore encourage responsible deployment with appropriate safeguards, including safety evaluation, watermarking, and provenance tracking.

References

Black Forest Labs. Flux.1 [schnell]. `https://github.com/black-forest-labs/flux`, 2024. Accessed: 2024-08-01.

Black Forest Labs. Flux.2. `https://github.com/black-forest-labs/flux2`, 2025.

Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Yue Wu, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, et al. PixArt-α: Fast training of diffusion transformer for photorealistic text-to-image synthesis. *arXiv preprint arXiv:2310.00426*, 2023.

Junsong Chen, Chongjian Ge, Enze Xie, Yue Wu, Lewei Yao, Xiaozhe Ren, Zhongdao Wang, Ping Luo, Huchuan Lu, and Zhenguo Li. PixArt-σ: Weak-to-strong training of diffusion transformer for 4k text-to-image generation. In *European Conference on Computer Vision*, pages 74–91. Springer, 2024.

Junsong Chen, Yuyang Zhao, Jincheng Yu, Ruihang Chu, Junyu Chen, Shuai Yang, Xianbang Wang, Yicheng Pan, Daquan Zhou, Huan Ling, et al. Sana-video: Efficient video generation with block linear diffusion transformer. *arXiv preprint arXiv:2509.24695*, 2025.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36: 52132–52152, 2023.

Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, et al. Ltx-video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024.

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

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *arXiv preprint arXiv:2506.08009*, 2025.

Minguk Kang, Richard Zhang, Connelly Barnes, Sylvain Paris, Suha Kwak, Jaesik Park, Eli Shechtman, Jun-Yan Zhu, and Taesung Park. Distilling diffusion models into conditional gans. In *European Conference on Computer Vision*, pages 428–447. Springer, 2024.

Akio Kodaira, Tingbo Hou, Ji Hou, Masayoshi Tomizuka, and Yue Zhao. Streamdit: Real-time streaming text-to-video generation. *arXiv preprint arXiv:2507.03745*, 2025.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

Xingjian Leng, Jaskirat Singh, Yunzhong Hou, Zhenchang Xing, Saining Xie, and Liang Zheng. Repa-e: Unlocking vae for end-to-end tuning with latent diffusion transformers. *arXiv preprint arXiv:2504.10483*, 2025.

Daiqing Li, Aleks Kamko, Ehsan Akhgari, Ali Sabet, Linmiao Xu, and Suhail Doshi. Playground v2. 5: Three insights towards enhancing aesthetic quality in text-to-image generation. *arXiv preprint arXiv:2402.17245*, 2024.

Fan Li, Chonghuinan Wang, Lina Lei, Yuping Qiu, Jiaqi Xu, Jiaxiu Jiang, Xinran Qin, Zhikai Chen, Fenglong Song, Zhixin Wang, et al. Hp-edit: A human-preference post-training framework for image editing. *arXiv preprint arXiv:2604.19406*, 2026.

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

Yuhang Ma, Xiaoshi Wu, Keqiang Sun, and Hongsheng Li. Hpsv3: Towards wide-spectrum human preference score, 2025. URL https://arxiv.org/abs/2508.03789.

Zhiyuan Ma, Yuxiang Wei, Yabin Zhang, Xiangyu Zhu, Zhen Lei, and Lei Zhang. Scaledreamer: Scalable text-to-3d synthesis with asynchronous score distillation. In *European Conference on Computer Vision*, pages 1–19. Springer, 2024.

Zachary Novack, Ge Zhu, Jonah Casebeer, Julian McAuley, Taylor Berg-Kirkpatrick, and Nicholas J Bryan. Presto! distilling steps and layers for accelerating music generation. *arXiv preprint arXiv:2410.05167*, 2024.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

Pablo Pernias, Dominic Rampas, Mats L Richter, Christopher J Pal, and Marc Aubreville. Würstchen: An efficient architecture for large-scale text-to-image diffusion models. *arXiv preprint arXiv:2306.00637*, 2023.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Jingjing Ren, Wenbo Li, Haoyu Chen, Renjing Pei, Bin Shao, Yong Guo, Long Peng, Fenglong Song, and Lei Zhu. Ultrapixel: Advancing ultra high-resolution image synthesis to new peaks. *Advances in Neural Information Processing Systems*, 37:111131–111171, 2024.

Jingjing Ren, Wenbo Li, Zhongdao Wang, Haoze Sun, Bangzhen Liu, Haoyu Chen, Jiaqi Xu, Aoxue Li, Shifeng Zhang, Bin Shao, et al. Turbo2k: Towards ultra-efficient and high-quality 2k video synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18155–18165, 2025a.

Sucheng Ren, Qihang Yu, Ju He, Xiaohui Shen, Alan Yuille, and Liang-Chieh Chen. Flowar: Scale-wise autoregressive image generation meets flow matching. In *ICML*, 2025b.

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

Sucheng Ren, Yaodong Yu, Nataniel Ruiz, Feng Wang, and Cihang Xie. Decoupled scale-wise autoregressive modeling for visual generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4506–4515, 2026.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

Joonghyuk Shin, Zhengqi Li, Richard Zhang, Jun-Yan Zhu, Jaesik Park, Eli Schechtman, and Xun Huang. Motionstream: Real-time video generation with interactive motion controls. *arXiv preprint arXiv:2511.01266*, 2025.

Ivan Skorokhodov, Sharath Girish, Benran Hu, Willi Menapace, Yanyu Li, Rameen Abdal, Sergey Tulyakov, and Aliaksandr Siarohin. Improving the diffusability of autoencoders. *arXiv preprint arXiv:2502.14831*, 2025.

Haoze Sun, Linfeng Jiang, Fan Li, Renjing Pei, Zhixin Wang, Yong Guo, Jiaqi Xu, Haoyu Chen, Jin Han, Fenglong Song, et al. Pocketsr: The super-resolution expert in your pocket mobiles. *arXiv preprint arXiv:2510.03012*, 2025.

Tencent Hunyuan Team. Hunyuanimage 2.1, 2025. URL https://github.com/Tencent-Hunyuan/HunyuanImage-2.1. Accessed 2025-10-27.

Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in neural information processing systems*, 37:84839–84865, 2024.

Hanyang Wang, Fangfu Liu, Jiawei Chi, and Yueqi Duan. Videoscene: Distilling video diffusion model to generate 3d scenes in one step. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16475–16485. IEEE, 2025.

Peihao Wang, Dejia Xu, Zhiwen Fan, Dilin Wang, Sreyas Mohan, Forrest Iandola, Rakesh Ranjan, Yilei Li, Qiang Liu, Zhangyang Wang, et al. Taming mode collapse in score distillation for text-to-3d generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9037–9047, 2024a.

Yufei Wang, Wenhan Yang, Xinyuan Chen, Yaohui Wang, Lanqing Guo, Lap-Pui Chau, Ziwei Liu, Yu Qiao, Alex C Kot, and Bihan Wen. Sinsr: diffusion-based image super-resolution in a single step. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 25796–25805, 2024b.

Yicheng Xiao, Wenxun Dai, Xinran Qin, Lin Song, Maoquan Zhang, Hang Xu, Yukang Chen, Yitong Li, Guohui Zhang, Yuan Zhang, et al. Joyai-video-edit: Real-time open-ended video editing with autoregressive diffusion. *arXiv preprint arXiv:2608.03974*, 2026.

Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, et al. Sana: Efficient high-resolution image synthesis with linear diffusion transformers. *arXiv preprint arXiv:2410.10629*, 2024.

Jiaqi Xu, Wenbo Li, Haoze Sun, Fan Li, Zhixin Wang, Long Peng, Jingjing Ren, Haoran Yang, Xiaowei Hu, Renjing Pei, et al. Fast image super-resolution via consistency rectified flow. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11755–11765, 2025.

Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:15903–15935, 2023.

Qiaosi Yi et al. Fine-structure preserved real-world image super-resolution via transfer vae training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and Bill Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in neural information processing systems*, 37:47455–47487, 2024a.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6613–6623, 2024b.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast causal video generators. *arXiv e-prints*, pages arXiv–2412, 2024c.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. *arXiv preprint arXiv:2410.06940*, 2024.

Xin Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Song-Hai Zhang, and Xiaojuan Qi. Text-to-3d with classifier score distillation. *arXiv preprint arXiv:2310.19415*, 2023.

Jinjin Zhang, Qiuyu Huang, Junjie Liu, Xiefan Guo, and Di Huang. Diffusion-4k: Ultra-high-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Sijie Zhao et al. Cv-vae: A compatible video vae for latent generative video models. *Advances in Neural Information Processing Systems*, 2024.

Supplementary Material

UltraDiff: Transferring High-Fidelity Priors to Compressed Latent Spaces for High-resolution Image Generation

This supplementary material provides comprehensive implementation details and additional results to support the main paper. We organize the content as follows:

- **Section A** discusses limitation and future work.
- **Section B** provides implementation and reproducibility details, including training protocols, data/model disclosure, and the open-source plan.
- **Section C** provides the specific architectural specifications and hyperparameters for the DiT backbone, VAEs, and training settings.
- **Additional Visual Results.** Due to the strict file size limit of the PDF submission, we provide extensive visual comparisons and a gallery of high-resolution results in the **separate attached supplementary files**.

A Limitations and Future Work

While UltraDiff demonstrates the effectiveness of joint-latent distillation for efficient high-fidelity synthesis, several limitations define the scope of the current evidence.

Architecture scope. We evaluate UltraDiff on one autoencoder–DiT pipeline. The method only adds latent-specific branches and shared attention during teacher training, and the distilled student remains a standard single-branch DiT. This makes the framework compatible in principle with other DiT-based pipelines, but this architectural compatibility should not be interpreted as empirical validation across backbones. Substantially different denoising architectures remain untested.

Dependence on pretrained components and multi-stage training. UltraDiff assumes access to a strong f8 text-to-image anchor and fixed f8/f32 VAEs. Its f32-transfer, MixTeacher, and distillation stages add training complexity, although their combined adaptation cost is lower than directly training our f32 baseline (Section B). This complexity is confined to training; inference uses only the final single-branch f32 student.

Independent latent spaces. The f8 and f32 VAEs are trained independently and are frozen during UltraDiff training. MixTeacher therefore learns their relationship at the diffusion vector-field level rather than enforcing explicit compatibility between their latent spaces. Combining UltraDiff with compatible-VAE training [Yi et al., 2025, Zhao et al., 2024] may further improve transfer.

Reproducibility and data access. The training data and f8 anchor are proprietary, which limits exact reproduction of the reported checkpoint. We provide the architecture, objectives, stage-wise optimization settings, data scale, and compute cost so that the method itself can be implemented and evaluated with alternative components. Validation with a fully open-source teacher–student pipeline remains important future work.

Model scaling and conditioning. We have not explored very large f32 backbones (e.g., $>$ 10B parameters), and the current model uses T5-XXL rather than a modern vision-language model for conditioning. Studying larger backbones, stronger multimodal conditioning, and additional DiT families is left for future work.

B Implementation Details

B.1 Training Strategy

To train our model, we employ a multi-stage training strategy designed to stabilize the learning process.

Phase 1: F32 Pre-training. We first train the DiT model in the f32 latent space for 120,000 iterations. This model is initialized from an f8 model, while the patch embedding and output prediction heads are randomly initialized. We refer to this checkpoint as *f32-phase1*.

Phase 2: Teacher Preparation. For the teacher model training, we utilize a dual-branch initialization: the f8 branch is initialized with the f8 model, and the f32 branch is initialized with the *f32-phase1* checkpoint. We train this teacher model for 5,000 iterations at resolutions of 512^2 and 1024^2 of varying aspect ratio.

Phase 3: Distillation and Scaling. Following teacher preparation, we freeze the teacher model and initialize a student model using *f32-phase1* to perform distillation. The student model is first trained at resolutions of 512 and 1024 for 50,000 iterations. Subsequently, we scale the training resolution to 1K, 2K, 3K, and 4K mixed resolution for 10,000 iterations.

High-resolution distillation protocol. Our MixTeacher is prepared at 512 and 1024 resolutions, where the teacher provides stable and reliable supervision. We therefore apply the distillation loss only during the 512–1K training stage. For higher resolutions, including 2K, 3K, and 4K, we disable the teacher-based distillation loss and continue training the student using the standard flow-matching objective. This design avoids relying on an unstable or prohibitively expensive high-resolution teacher. In practice, the 1K distillation stage transfers the high-fidelity f8 priors into the f32 student, while the subsequent high-resolution flow-matching stage adapts the same student to larger resolutions efficiently.

Loss Function: For 1K resolution, we employ a distillation loss and flow matching loss. For higher resolutions (2K, 3K, 4K), we exclusively calculate the flow matching loss to maximize computational efficiency.

Distributed Training. All training is conducted on 64 NVIDIA A100 GPUs. We utilize Fully Sharded Data Parallel (FSDP) with a hybrid sharding strategy to manage memory consumption effectively during distributed training.

B.2 Training Cost and Practical Setup

The f8/f32 VAEs are trained separately before UltraDiff and then kept fixed, while the f8 text-to-image anchor is treated as an available pretrained component. Table 2 in the main paper reports the stage-wise optimization steps and aggregate GPU-days required to adapt these fixed components into the final f32 student. The multi-stage process is used only during model development; inference uses the final standalone f32 student without the teacher or f8 branch.

B.3 Model, Data, and Open-source Plan

The strong f8 text-to-image model used in our pipeline is an internal proprietary model. The filtered corpus size is reported in the main paper, and stage-wise training resources are summarized in Table 2. All data were collected and curated under copyright-compliant data governance.

For scale context, SD3-medium reports 1B pretraining images followed by 30M aesthetic fine-tuning images and 3M preference images [Esser et al., 2024]. PixArt-Σ reports a 33M-image high-resolution corpus and is initialized from PixArt-α, whose pretraining used 25M images [Chen et al., 2024, 2023]. SDXL, Playground v2.5, FLUX.1, and Sana do not disclose directly comparable unique-image counts in their primary reports. Because filtering criteria, repeated sampling, and training stages differ across systems, these figures provide scale context rather than a controlled data-efficiency comparison.

We plan to release the training and inference code for UltraDiff, including the MixTeacher training pipeline and marginal score distillation procedure. However, since the current implementation relies on internal teacher/student backbones and proprietary training data, the exact pretrained weights used in this work may not be releasable.

UltraDiff is not tied to a specific proprietary implementation, but the present experiments validate only one DiT family. The framework can in principle be instantiated with open-source f8 text-to-image models and compressed-latent f32-style DiT backbones without changing the student's inference architecture. Empirical validation on fully open-source and substantially different backbones remains future work.

C Detailed Model Architectures and Hyperparameters

In this section, we provide the specific architectural hyperparameters for the Diffusion Transformer (DiT), the Variational Autoencoders (VAEs), and the conditioning models used in UltraDiff.

C.1 Diffusion Transformer (DiT) Architecture

We employ a shared DiT backbone for both the teacher (dual-branch) and the student (single-branch). The model processes text embeddings (from T5-XXL) and image latents as a concatenated sequence. Crucially, since the f32 latent space is highly compressed, we utilize a patch size $p = 1$ for patch embedding layer.

Table 10 summarizes the configuration of our DiT backbone.

Table 10: **UltraDiff DiT Architecture Specifications.** The model follows a standard Transformer backbone with Rotary Positional Embeddings (RoPE) and QK-Norm.

Hyperparameter	**Value**
Model Depth (N)	28 layers
Hidden Dimension (D)	2048
Attention Heads	16
KV Heads	16
Head Dimension	$2048/16 = 128$
FFN Multiplier	1.3 (SwiGLU variant)
Input Processing	
Input Patch Size	1×1
Latent Channels (f8 branch)	16
Latent Channels (f32 branch)	64
Positional Embedding	3D Rotary (RoPE)
RoPE Base (θ)	10,000
Conditioning	
Text Encoder	T5-XXL
Text Embedding Dim	4096

C.2 VAE Architectures and Training Details

We use two KL-regularized continuous VAEs. Each encoder predicts the mean and variance of a diagonal Gaussian distribution. Table 12 compares their architectures.

VAE reconstruction quality. Both VAEs are trained from scratch and then frozen during diffusion-model training. Table 11 compares them with public autoencoders under the same held-out reconstruction protocol. The scale and channel count refer to each VAE's native latent output, before any model-specific patchification. FLUX.2 is therefore listed as f8/C32; its subsequent 2×2 patchification is part of the DiT input pipeline, not the VAE.

Our f32 VAE has lower reconstruction fidelity than our f8 VAE, consistent with its stronger information bottleneck. Results among public f32 VAEs are mixed: HunyuanImage-2.1 has higher PSNR and SSIM than our f32 VAE, but worse LPIPS, rFID, and PatchFID. Reconstruction quality is an important constraint, but it does not fully determine downstream generation quality; the diffusability of the latent space also matters [Skorokhodov et al., 2025].

All f32 variants in Table 1 use the same f32 VAE. The GenEval gain from 0.61 with f8 initialization to 0.72 with UltraDiff therefore reflects better modeling of the same compressed latent space, rather than a change in autoencoder reconstruction quality.

f8 VAE. For the f8 branch, we utilize a standard 2D KL-VAE architecture.

Table 11: **VAE reconstruction quality.** Scale and channels describe the native VAE latent output. All models are evaluated under the same reconstruction protocol.

VAE	Scale	Channels	PSNR↑	SSIM↑	LPIPS↓	rFID↓	PatchFID↓
Ours (f8)	f8	16	30.9588	0.8719	0.0343	2.1645	3.7048
Ours (f32)	f32	64	26.6258	0.7416	0.0681	6.7395	3.2330
SANA 1.5 DC-AE [Xie et al., 2024]	f32	32	25.3647	0.6888	0.0930	7.8557	4.2969
FLUX.2 autoencoder [Black Forest Labs, 2025]	f8	32	33.8557	0.9242	0.0144	1.1360	0.9926
HunyuanImage-2.1 VAE [Team, 2025]	f32	64	29.4606	0.8217	0.1111	9.7390	10.1938
Stable Diffusion 3.5 VAE [Esser et al., 2024]	f8	16	30.6493	0.8609	0.0298	2.4047	1.3692

- **Encoder & Decoder:** The encoder downsamples the input by a factor of 8 spatially. It utilizes a channel multiplier strategy of $[1, 2, 4, 6]$ with a base channel width of 128.
- **Continuous Latent Space:** The encoder outputs a distribution parameters map with $z = 16$ channels. We apply a slight KL-divergence penalty to regularize this continuous space towards a standard normal distribution.
- **Upsampling:** The decoder employs a `PixelShuffle` strategy for upsampling, preserving high-frequency details.

f32 VAE. The f32 VAE is a specialized deep 2D KL-VAE designed for extreme spatial compression ($32\times$).

- **Encoder & Decoder:** The encoder achieves a $32\times$ downsampling factor through a deep 6-stage network with channel multipliers $[1, 2, 4, 6, 8, 8]$. It uses standard 2D Convolutions and `GroupNorm`.
- **Latent Space:** To compensate for the significant loss of spatial resolution, the continuous latent dimension is increased to $z = 64$.
- **Upsampling:** The decoder uses a `Nearest+Conv` upsampling strategy, providing robust reconstruction for the highly compressed latent grid.

Table 12: **Detailed Hyperparameters for f8 and f32 VAEs.** Both models utilize 2D KL-regularized architectures with continuous latent spaces. The f32 model features a deeper network and larger latent channels to handle the $32\times$ compression rate.

Configuration	f8 VAE (Teacher Anchor)	f32 VAE (Target)
Type	2D KL-VAE	2D KL-VAE
Latent Space	Continuous (Gaussian)	Continuous (Gaussian)
Downsampling Factor	$8\times$	$32\times$
Latent Channels (C)	16	64
Encoder		
Channel Base	128	128
Channel Multipliers	$[1, 2, 4, 6]$	$[1, 2, 4, 6, 8, 8]$
ResBlocks per Level	2	2
Decoder		
Channel Multipliers	$[6, 4, 2, 1]$	$[8, 8, 6, 4, 2, 1]$
Upsampling Method	Residual PixelShuffle	Nearest + Conv
Loss weighting		
Perceptual Loss (LPIPS)	1.0	0.5
KL Weight	1×10^{-12}	1×10^{-12}
Loss Formulation	LPVAE	LPVAE

C.3 Latent Normalization

A critical component of our method is addressing the distribution shift between the two continuous latent spaces. We apply channel-wise normalization before feeding latents into the DiT.

C.4 Training Hyperparameters

We train the models using the AdamW optimizer with a linear warmup and cosine decay scheduler. The specific training settings are detailed in Table 13.

Table 13: **Training hyperparameters.** Common optimizer settings and stage-specific training lengths used in the UltraDiff pipeline.

Parameter	**Value**
Optimizer	AdamW
Adam β_1, β_2	0.9, 0.95
Learning Rate (LR)	1×10^{-4}
Weight Decay	0.01
LR Schedule	Linear Warmup + Cosine Decay
Warmup Steps	500
Min LR	1×10^{-5}
Gradient Clipping	1.0
EMA Decay	0.9999
EMA Start Step	2,000
Batch Size	2048 (Global)
Training Steps	Phase 1: 120K; Phase 2: 5K; Phase 3: 50K + 10K
Loss Weighting	
Prediction Target	Velocity (v-prediction)
Loss Type	MSE (L_2)
Noise Scheduler	Linear Interpolant Scheduler

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See abstract and introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Section A

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: [N/A]

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Section 4.2 and appendix

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [N/A]

   Justification: We will opensource upon acceptance of this manuscript.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: See experiment.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: See experiment.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See experiment

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: [N/A]

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: See Broader impacts

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [No]

    Justification: [No]

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: [No]

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [No]

    Justification: [No]

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: [N/A]

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.