Accelerating Diffusion Models via Equal-Risk Caching

Can Zhang[1] and Liangshun Zou[2⋆]

[1] Fudan University, Shanghai, China
[2] Tongji University, Shanghai, China
{yunxiao4495,xszz_2026}@163.com

Abstract. Diffusion models achieve strong image and video generation quality but remain slow at inference due to multi-step denoising. Caching accelerates inference by reusing intermediate features across timesteps, yet its effectiveness depends critically on refresh scheduling (when to perform full recomputation). Existing schedules fall into three paradigms: fixed-interval, online heuristic, and global optimization. The first two may miss temporal sensitivity structure or accumulated segment-level degradation, while the last incurs substantial offline cost. We propose Equal-Risk Caching (ERC), which builds a one-dimensional temporal risk profile from a single fixed-length probing cache curve and uses cumulative risk mass as a global surrogate for segment sensitivity. ERC places refresh boundaries by equal-mass partitioning on the cumulative risk axis, automatically allocating denser recomputation to high-risk regions and longer reuse to low-risk regions. This reduces the offline profiling cost of schedule construction from $\mathcal{O}(T \cdot L)$ to $\mathcal{O}(T)$. Experiments on multiple diffusion models demonstrate an overall competitive quality–speed trade-off.

Keywords: diffusion cache · image and video generation · diffusion model acceleration

1 Introduction

In recent years, diffusion models [10,13,36,38,39] have emerged as the paramount generative paradigm, fundamentally transforming the landscape of visual content synthesis. In the realm of image generation, these models have decisively surpassed previous state-of-the-art frameworks such as GANs [10]. By operating within compressed latent spaces [31, 32] and integrating robust deep language understanding [33], modern text-to-image diffusion models demonstrate unprecedented capabilities in producing photorealistic, highly diverse, and semantically intricate static visuals. Building upon this phenomenal success in image synthesis [31, 32], researchers have rapidly extended diffusion architectures to the temporal dimension for video generation [2, 6, 12]. Recent breakthroughs have successfully tackled complex spatio-temporal dependencies, empowering video dif-

⋆ Corresponding author.

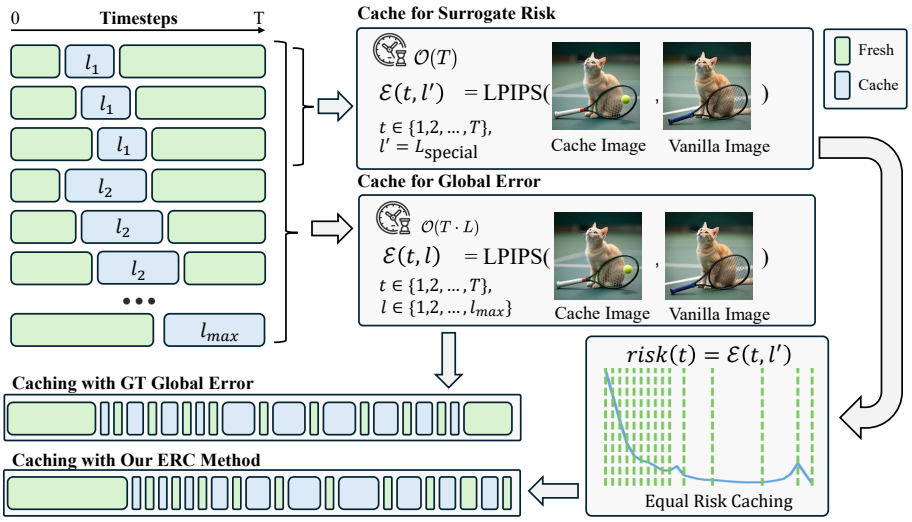

Fig. 1: Main pipeline of Equal-Risk Caching (ERC). We estimate a timestep-wise surrogate risk from a single probing cache length l' by measuring $\mathcal{E}(t, l')$ between cache and vanilla outputs, which scales as $\mathcal{O}(T)$ and avoids exhaustive global-error profiling over all lengths ($\mathcal{O}(T \cdot L)$). The obtained risk curve is then accumulated and partitioned into equal-risk areas to place refresh boundaries, producing an adaptive schedule that allocates more fresh computation to high-risk regions and longer cache reuse to low-risk regions (green: fresh, blue: cache).

fusion models to generate high-definition, temporally consistent dynamic scenes with astonishingly realistic motion and physical plausibility [3, 4].

However, despite these breathtaking generative capabilities, their practical deployment in real-world scenarios remains severely hindered by the exorbitant inference costs and slow generation speeds inherent to their multi-step, computationally massive iterative sampling paradigm. To accelerate diffusion models, a wide range of methods [7, 18, 20, 22, 23, 27, 34, 41] have been proposed, including sampler optimization, model distillation, token pruning, quantization, and caching. Among them, caching-based methods [5, 8, 26, 28, 35, 47] are particularly attractive because they require no additional training. By reusing similar features across timesteps, caching reduces redundant full forward passes during sampling, offering a practical inference-time acceleration strategy.

A key challenge in caching-based acceleration lies in determining *when* to perform a full forward recomputation (i.e., refresh). Existing approaches typically follow three paradigms. The first paradigm [8,21,35] adopts a naive fixed-interval strategy, refreshing at predefined timesteps while reusing cached features in between. The second paradigm [5, 20, 29, 48] dynamically triggers refreshes based on online local signals, such as the change in model outputs between adjacent timesteps and predefined thresholds. These methods assume that small varia-

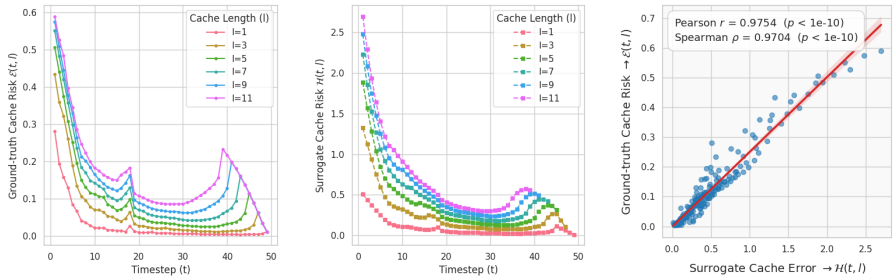

Fig. 2: Comparison and correlation between ground-truth and surrogate cache risks on FLUX.1 [dev]. (a) Ground-Truth Cache Risk $\mathcal{E}(t, l)$ evaluated across different timesteps and cache lengths. **(b)** Surrogate Cache Risk $\mathcal{H}(t, l)$ approximated using a fixed probe length $l' = 5$. **(c)** Correlation analysis between $\mathcal{H}(t, l)$ and $\mathcal{E}(t, l)$. The results exhibit a strong positive correlation, validating the effectiveness of using the cumulative risk as an efficient proxy.

tions between neighboring outputs imply limited final deviation. However, such locally greedy criteria ignore the accumulation of small errors over time, which may lead to noticeable degradation in generation quality and content consistency.

More recently, a third line of work [11] measures the final risk induced by different caching intervals offline and employs path search or dynamic programming to identify an optimal schedule. By explicitly modeling the global effect of caching segments, these methods effectively prevent instability caused by local heuristics and significantly improve content consistency under acceleration. Nevertheless, constructing a complete risk map over all possible intervals requires extensive offline evaluation. The computational cost grows rapidly with the number of timesteps and candidate segment lengths, making low-cost and flexible scheduling difficult.

At their core, these approaches face the same fundamental challenge: caching schedules aim to decide refresh boundaries without a structural characterization of how caching-induced risk distributes over time. Local methods approximate risk using instantaneous variations, while global optimization methods treat interval risk as a black-box quantity that must be exhaustively enumerated. These represent two extremes: low overhead but prone to cumulative-error-induced quality drift, and stronger global quality control but high offline computational cost. Both lack a continuous surrogate that describes the temporal structure of risk accumulation.

As shown in Fig. 2, we observe that caching-induced risk exhibits a stable temporal structure throughout the diffusion process. Certain time regions are more sensitive to feature reuse, while others tolerate longer caching intervals. Furthermore, for caching segments with different starting timesteps and lengths, the final risk shows a strong monotonic correlation with a cumulative quan-

tity defined along the time axis. Importantly, this cumulative measure can be constructed from a single fixed-length caching curve, without enumerating the complete interval risk map.

Based on this observation, we reformulate the refresh scheduling problem from selecting timesteps to partitioning risk mass. Specifically, we treat full forward computations as segment boundaries and perform equal partitioning along a cumulative risk axis to directly determine the refresh set. We term this strategy **Equal-Risk Caching**. Unlike heuristic methods that rely on local signals, our approach determines the schedule using a global cumulative surrogate. Unlike error-map-based optimization methods, it avoids the costly construction of a full interval risk graph, requiring only a single proxy curve to generate stable schedules.

Because refresh density is automatically concentrated in high-risk regions and extended in low-risk regions, Equal-Risk Caching stabilizes the final risk under the same computational budget while significantly reducing offline modeling cost. Experiments across multiple diffusion models demonstrate that our method achieves a superior quality–speed trade-off compared to heuristic methods and approaches the performance of global optimization strategies under identical computation budgets.

Our contributions are summarized as follows:

- **Correlation-Based Efficient Risk Surrogate.** We identify a strong correlation between the ground-truth caching risk and a lightweight cumulative temporal signal. This insight enables a linear-time $\mathcal{O}(T)$ surrogate metric, bypassing the expensive $\mathcal{O}(T \cdot L)$ evaluation of the full risk map.
- **Equal-Risk Caching Strategy.** We propose **Equal-Risk Caching**, a simple yet effective algorithm that determines optimal refresh boundaries by equal-mass partitioning on the cumulative risk axis. This avoids complex search heuristics and directly derives schedules from the 1D surrogate.
- **Broad Applicability and Performance.** We validate our method on diverse Text-to-Image and Text-to-Video diffusion models. Our method achieves stable acceleration with competitive generation quality while requiring substantially lower offline computation cost than existing baselines.

2 Related Work

2.1 Diffusion Acceleration

Accelerating diffusion sampling has become an important research direction due to the high computational cost of iterative denoising. Existing approaches improve efficiency from several perspectives, including sampler design, attention sparsification, token compression, model quantization, and feature reuse.

A direct strategy is to reduce the number of sampling steps. DDIM [37] generalizes DDPM [13] to non-Markovian diffusion processes and enables deterministic sampling with significantly fewer steps. DPM-Solver++ [24] formulates diffusion sampling as solving an ODE under a data-prediction parameterization

and derives high-order solvers tailored to diffusion dynamics. UniPC [46] unifies predictor and corrector steps into a single framework, achieving strong sample quality with very few function evaluations.

Beyond sampler optimization, recent works reduce per-step computation. Sparse VideoGen (SVG) [44] observes structured sparsity patterns in attention heads and applies dynamic sparse computation during inference. Token-level redundancy is addressed by methods such as Fourier Token Merging [22], which clusters tokens in the frequency domain before merging them, and ToMA [25], which formulates token merging as a submodular optimization problem with reversible attention-like transformations. Model quantization further reduces memory and arithmetic cost. SVDQuant [18] mitigates activation outliers by absorbing them into low-rank components, enabling stable 4-bit diffusion models with competitive performance.

Orthogonal to these techniques, caching accelerates inference by reusing intermediate features across timesteps. Since diffusion trajectories exhibit temporal smoothness, feature reuse can substantially reduce redundant computation without retraining.

2.2 Caching in Diffusion Models

DeepCache [28] is an early attempt to exploit temporal redundancy in diffusion models. It studies both uniform and manually designed non-uniform caching intervals within U-Net architectures, demonstrating that intermediate features, especially in later denoising stages, change smoothly across steps.

TaylorSeer [21] improves upon uniform caching by approximating feature evolution with Taylor series expansion, using higher-order derivatives to predict intermediate representations. TeaCache [20] introduces a calibrated polynomial estimator to model output variation conditioned on time embeddings. These methods rely on explicit approximation of feature dynamics.

More recent approaches aim to reduce calibration overhead and better control accumulated error. MagCache [29] discovers a "magnitude decay law" between consecutive residuals to reduce the reliance on multi-prompt offline calibration. EasyCache [48] leverages the relative stability of transformation rates across denoising stages to estimate output deviation without polynomial fitting. DiCache [5] introduces a lightweight online probe, exploiting the correlation between shallow-layer risk and final output error to guide adaptive caching. LeMiCa [11] argues that local heuristic risk may underestimate global degradation. It measures the final output error induced by caching each segment, formulates caching scheduling as a directed graph, and constrains worst-case path error to bound accumulated deviation.

Compared with LeMiCa, our method does not require high-coverage offline interval evaluation. Instead, we model the distribution density of risk along the time axis and allocate refresh steps accordingly, enabling lightweight and risk-aware scheduling.

Algorithm 1 Equal-Risk Caching

Input: Diffusion model, timesteps $\{0, \ldots, T-1\}$, fixed surrogate length l', refresh budget $\mathcal{B}$
Output: Refresh boundary timesteps $\{t_i\}_{i=0}^{\mathcal{B}-1}$
// Step 1: Estimate per-timestep surrogate risk
for $t = 1$ to $T-1$ **do**
    Compute $r(t) = \mathcal{E}(t, l')$ using Eq. (4)
end for
// Step 2: Construct $R(T)$ as the prefix sum of risks before timestep t
// must refresh when t=0, so r(0) is excluded, then we define R(1)=0
$R(1) \leftarrow 0$
for $t = 2$ to T **do**
    $R(t) \leftarrow R(t-1) + r(t-1)$
end for
// Step 3: Equal-risk partition
// t_i: the i-th refresh boundary timestep
$t_0 \leftarrow 0$
$\Delta R \leftarrow R(T)/(\mathcal{B}-1)$
for $i = 1$ to $\mathcal{B}-1$ **do**
    $t_i \leftarrow \min\{t \mid R(t) \geq i \cdot \Delta R\}$
end for
$t_{\mathcal{B}-1} \leftarrow T-1$
return $\{t_0, t_1, \ldots, t_{\mathcal{B}-1}\}$

3 Method

We propose a diffusion caching framework built upon two core ideas: (1) a low-cost surrogate for cache risk estimation, and (2) an equal-risk scheduling strategy that distributes refresh boundary steps according to the cumulative risk density over time. The complete procedure is summarized in Fig. 1 and Alg. 1.

3.1 Formulation

Diffusion Model. Diffusion models generate data by progressively reversing a forward noise process. Starting from pure noise $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, the inference process iteratively applies a denoising network for T steps. Modern diffusion architectures, such as DiT [30], typically adopt the Flow Matching framework [19]. The generative process is modeled as an Ordinary Differential Equation (ODE) that transforms noise to data over time $t \in [0, 1]$. The model $v_\theta(\mathbf{x}_t, t, \mathbf{c})$ is trained to predict the velocity field of the flow. During inference, this ODE is solved numerically to step backward from t to $t - \Delta t$:

$$\mathbf{x}_{t-\Delta t} = \mathbf{x}_t - \Delta t \cdot v_\theta(\mathbf{x}_t, t, \mathbf{c}), \tag{1}$$

where Δt represents the step size and $\mathbf{c}$ denotes the conditioning signal (e.g., text prompts). Since the heavy velocity prediction network v_θ must be fully evaluated at every integration step to determine the update direction, the sequential generation incurs significant computational costs.

Cache Reuse Strategies. At cache steps, we reuse cached features instead of performing the full computation. Different methods employ different reuse formulations; here we summarize two representative strategies commonly used in diffusion caching works. The first is direct reuse from the latest full-computation timestep:

$$\mathbf{F}_{\text{pred},t} = \mathbf{F}_{t_\alpha}, \tag{2}$$

The second is first-order extrapolation using the two most recent full-computation timesteps:

$$\mathbf{F}_{\text{pred},t} = \alpha \mathbf{F}_{t_\alpha} + \beta \mathbf{F}_{t_\beta}. \tag{3}$$

Here t_α and t_β denote cached reference timesteps corresponding to the two most recent steps where full model computation is performed. The coefficients α and β are scalar weights determined by the specific reuse formulation (e.g., the dynamic cache trajectory alignment in DiCache [5] or the discrete Taylor expansion in TaylorSeer [21]). Intuitively, first-order extrapolation captures local temporal trends by linearly combining features from the two latest fully computed steps. Our method is orthogonal to the reuse strategy: ERC focuses on *when* to refresh, while the reuse rule determines *how* cached features are utilized between refreshes.

3.2 Cache Risk Surrogate

Consider a diffusion process with T timesteps. Let x_0 denote the output generated by full inference without caching. For a given starting timestep t and cache length l, we denote by $x_{t,l}$ the final output obtained when caching is continuously applied for l consecutive steps starting from timestep t. We define the ground-truth cache risk as

$$\mathcal{E}(t, l) = \text{LPIPS}(x_{t,l}, x_0), \tag{4}$$

which measures the perceptual deviation between the cached output and the full inference result. The variation trend of this risk is illustrated in Fig. 2 a). To determine appropriate caching intervals, one would need to evaluate $\mathcal{E}(t, l)$ for all combinations of (t, l). If $l \in \mathcal{L}$ denotes the set of candidate cache lengths, the total computational cost becomes $\mathcal{O}(T \cdot L)$. This quadratic dependence on T makes exhaustive evaluation impractical for large diffusion steps.

To reduce the complexity, we introduce a surrogate cache error. Instead of evaluating every (t, l) pair, we fix a surrogate cache length l' in advance and define a per-timestep risk signal as

$$r(t) = \mathcal{E}(t, l'). \tag{5}$$

Here l' is a predefined probe length shared across all timesteps. It serves as a consistent surrogate to measure the cache sensitivity at each timestep. In practice, the choice of l' is determined empirically, and we analyze its impact in the ablation study.

Based on this per-step risk, we approximate the segment-level cache error using cumulative risk:

$$\mathcal{H}(t,l) = \sum_{\hat{t}=t}^{t+l-1} r(\hat{t}). \tag{6}$$

The surrogate risk $\mathcal{H}(t,l)$ can be computed in $\mathcal{O}(T)$ time, since $r(t)$ is evaluated only once per timestep. The variation trend is displayed in Fig. 2 b.

Empirically, as illustrated in Fig. 2, we observe a strong positive correlation between the ground-truth cache risk $\mathcal{E}(t,l)$ (Fig. 2 a) and the surrogate cumulative risk $\mathcal{H}(t,l)$ (Fig. 2 b). The scatter plot in Fig. 2 c further validates this relation with significantly high correlation coefficients.

This indicates that the cumulative per-step risk captures the dominant structure of cache-induced degradation. Therefore, in practice, we replace $\mathcal{E}(t,l)$ with $\mathcal{H}(t,l)$ for cache schedule design, achieving substantial computational savings without constructing the full two-dimensional risk map.

3.3 Equal-Risk Caching

Building upon the observed correlation, we propose **Equal-Risk Caching**, a strategy that determines refresh boundaries by analyzing the surrogate risk distribution rather than relying on expensive ground-truth evaluations. The core motivation behind this strategy is to allocate computational resources adaptively according to the model's temporal sensitivity. In diffusion processes, the Ground-Truth Cache Risk $\mathcal{E}(t,l)$ varies significantly across timesteps. A uniform caching schedule (i.e., fixed interval size) is suboptimal because it treats all timesteps equally: high-risk regions accumulate error rapidly, leading to artifacts, while low-risk regions waste computations on unnecessary refreshes. To address this, we treat the caching schedule as a *risk allocation problem*. By distributing the total allowable risk budget equally across segments, we naturally enforce denser refreshes in high-sensitivity regions and allow longer cache reuse in stable regions.

Formally, given the per-step risk signal $r(t)$, we define the cumulative risk function $R(t)$ as the prefix sum of risks before timestep t:

$$R(t) = \begin{cases} 0, & t = 1, \\ R(t-1) + r(t-1), & 2 \le t \le T. \end{cases} \tag{7}$$

We set $R(1) = 0$ as the origin of the cumulative risk because the initial timestep is always computed from scratch, and no cache-induced risk is defined at $t = 0$. The total accumulated risk over the cacheable diffusion timesteps $t \ge 1$ is $R(T)$. Suppose we are given a refresh budget of $\mathcal{B}$ segments, where each segment is identified by its right-end refresh boundary. The initial timestep accounts for one mandatory refresh excluded from the risk partition. Instead of uniformly partitioning the time axis, we partition the cumulative risk axis into equal portions. The target risk mass for each segment is calculated with remaining refresh budget as:

$$\Delta R = \frac{R(T)}{\mathcal{B} - 1}. \tag{8}$$

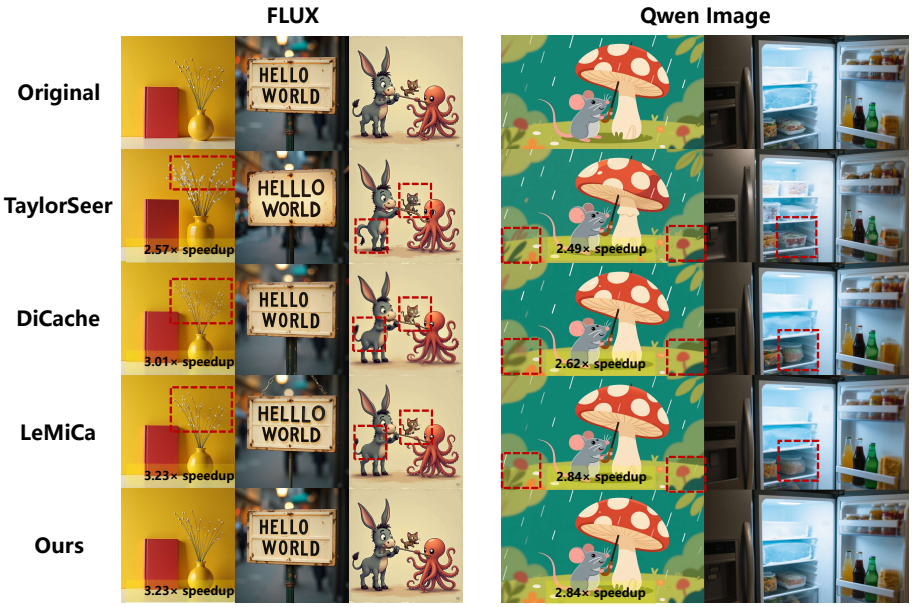

Fig. 3: Qualitative comparison in text-to-image generation. Best viewed zoomed in.

Let $\{t_i\}_{i=0}^{\mathcal{B}-1}$ denote the refresh boundaries, with $t_0 = 0$ and $t_{\mathcal{B}-1} = T-1$. By Eq. (6) and Eq. (7), the surrogate segment risk is exactly the cumulative-risk increment:

$$\mathcal{H}(t_i, t_{i+1} - t_i) = R(t_{i+1}) - R(t_i). \tag{9}$$

Under equal-risk partitioning, we then choose boundaries so that

$$R(t_{i+1}) - R(t_i) \approx \Delta R, \tag{10}$$

for $i = 1, \ldots, \mathcal{B} - 2$. Here t_{i+1} denotes the next segment boundary (i.e., refresh boundary) in the sequence $\{t_j\}_{j=0}^{\mathcal{B}-1}$ rather than a one-step increment in denoising time. The approximation arises because timesteps are discrete, so the exact target risk ΔR cannot always be matched. This formulation yields an equal-risk caching schedule. Regions with high $r(t)$ will cause $R(t)$ to grow sharply, consuming the quota ΔR quickly and triggering an early refresh. Conversely, regions with low $r(t)$ result in slow growth of $R(t)$, allowing for extended cache intervals.

4 Experiments

4.1 Experimental Setup

Metrics. Following prior work, we use LPIPS [45], PSNR [15], and SSIM [42] as visual consistency metrics relative to the vanilla result. For video models,

Table 1: Quantitative comparison in text-to-image generation on FLUX.1 [dev] and Qwen-Image. Best and second-best results within the same recomputation budget are highlighted in bold and underline.

Method	LPIPS ↓	SSIM ↑	PSNR ↑	Latency(s) ↓	Speed ↑
FLUX.1 [dev] 1024×1024					
Original	–	–	–	15.60	1.00×
LeMiCa ($\mathcal{B} = 15$)	**0.1857**	**0.8637**	<u>24.9852</u>	4.83	3.23×
Ours ($\mathcal{B} = 15$)	<u>0.1954</u>	<u>0.8628</u>	**25.4227**	4.83	3.23×
TeaCache ($l = 0.6$)	0.3548	0.7191	17.8193	4.97	3.14×
DiCache ($\delta = 0.4$)	0.2235	0.8396	24.2674	5.19	3.01×
TaylorSeer ($\mathcal{N} = 3, O = 2$)	0.2088	0.8329	21.7558	6.08	2.57×
TeaCache ($l = 0.4$)	0.3117	0.7496	18.7167	6.45	2.42×
LeMiCa ($\mathcal{B} = 26$)	<u>0.0886</u>	<u>0.9354</u>	<u>30.1025</u>	8.21	1.90×
Ours ($\mathcal{B} = 26$)	**0.0816**	**0.9442**	**31.5304**	8.21	1.90×
Qwen-Image 1664×928					
Original	–	–	–	45.20	1.00×
LeMiCa ($\mathcal{B} = 17$)	0.1506	0.9234	27.9708	15.94	2.84×
LeMiCa($\mathcal{B} = 17$)+DCTA	0.1194	0.9344	<u>28.5831</u>	15.94	2.84×
Ours ($\mathcal{B} = 17$)	0.1268	0.9316	27.6859	15.94	2.84×
Ours ($\mathcal{B} = 17$)+DCTA	**0.0842**	**0.9477**	**28.9992**	15.94	2.84×
DiCache ($\delta = 0.25$)	<u>0.0978</u>	<u>0.9420</u>	28.3096	17.26	2.62×
TaylorSeer ($\mathcal{N} = 3, O = 2$)	0.2055	0.8416	20.0409	18.14	2.49×
LeMiCa ($\mathcal{B} = 25$)	<u>0.1192</u>	<u>0.9490</u>	<u>31.5807</u>	23.03	1.96×
Ours ($\mathcal{B} = 25$)	**0.0653**	**0.9711**	**33.5913**	23.03	1.96×

we additionally use VBench [16] to evaluate human preference. We measure inference efficiency using speedup ratio and inference latency.

Models and baselines. We evaluate on three representative diffusion models: FLUX.1 [dev] [1], Wan2.1 [40], and Qwen Image [43]. Baselines include TaylorSeer [21], TeaCache [20], DiCache [5], and LeMiCa [11]. Our method primarily focuses on *when* to refresh cached features; for *how* to reuse cache at selected timesteps, multiple design choices are possible. By default, we follow LeMiCa [11] and reuse the residual cache from the most recent full computation. On Wan2.1-T2V-1.3B and Qwen Image, we additionally combine our method and LeMiCa [11] with DCTA (dynamic cache trajectory alignment, proposed in DiCache [5]) to provide a fairer comparison with higher-order cache reuse methods such as DiCache and TaylorSeer.

Implementation details. For text-to-image generation, we use 200 prompts from DrawBench [33]. For text-to-video generation, we randomly sample 200 prompts from VBench and conduct evaluations at a resolution of 480×832 with 81 frames. Inference experiments for FLUX.1 [dev] and Qwen Image are con-

Table 2: Quantitative comparison in text-to-video generation on Wan2.1-T2V-1.3B. Best and second-best results within the same recomputation budget are highlighted in bold and underline.

Method	LPIPS ↓	SSIM ↑	PSNR ↑	VBench(%) ↑	Latency(s) ↓	Speed ↑
Original	–	–	–	74.80	276.58	1.00×
LeMiCa ($\mathcal{B} = 17$)	0.1622	0.8912	29.5242	73.98	101.73	2.72×
Ours ($\mathcal{B} = 17$)	0.1642	0.8890	29.3016	74.08	101.73	2.72×
Ours ($\mathcal{B} = 17$)+DCTA	**0.1242**	**0.9179**	**30.6336**	**74.80**	101.73	2.72×
DiCache ($\delta = 0.2$)	<u>0.1594</u>	<u>0.8879</u>	<u>28.2667</u>	74.24	106.72	2.59×
TaylorSeer ($\mathcal{N} = 3, O = 2$)	0.3047	0.7412	20.2040	<u>74.39</u>	113.02	2.45×
LeMiCa ($\mathcal{B} = 25$)	<u>0.1146</u>	<u>0.9345</u>	<u>33.5376</u>	<u>74.53</u>	144.10	1.92×
Ours ($\mathcal{B} = 25$)	**0.1133**	**0.9357**	**33.5746**	**74.64**	144.10	1.92×
TeaCache ($l = 0.07$)	0.1291	0.9151	30.4393	74.61	176.44	1.57×

ducted on NVIDIA RTX PRO 6000 96GB GPUs, while Wan2.1-T2V-1.3B is evaluated on NVIDIA RTX 4090D 24GB GPUs. Unless otherwise specified, the original (non-cached) inference uses 50 sampling steps. FlashAttention-2 [9] is enabled by default in all experiments. For FLUX.1 [dev], we disable the classifier-free guidance (CFG) [14] unconditional branch by default. For Wan2.1-T2V-1.3B and Qwen Image, the unconditional branch is included at all timesteps, with a single whitespace character used as the unified negative prompt.

4.2 Comparison with Existing Methods

Quantitative analysis.

Tab. 1 reports text-to-image results on FLUX.1 [dev] and Qwen-Image. Under the same recomputation budget and latency, ERC consistently delivers competitive quality against existing schedulers. On FLUX.1 [dev] at $\mathcal{B} = 15$ (3.23×), ERC achieves the best PSNR (25.4227). LPIPS/SSIM remain close to LeMiCa (0.1954/0.8628 vs. 0.1857/0.8637), and ERC clearly outperforms online baselines such as DiCache (0.2235, 0.8396, 24.2674) and TeaCache (0.3548, 0.7191, 17.8193). At a milder acceleration ($\mathcal{B} = 26$, 1.90×), ERC obtains the best results on all three metrics (LPIPS 0.0816, SSIM 0.9442, PSNR 31.5304), showing that equal-risk partitioning remains effective when quality requirements are stricter.

On Qwen-Image, ERC also shows strong robustness across budgets. At $\mathcal{B} = 17$ (2.84×), ERC improves over LeMiCa on LPIPS/SSIM (0.1268/0.9316 vs. 0.1506/0.9234), while LeMiCa is slightly higher on PSNR (27.9708 vs. 27.6859). ERC+DCTA further achieves the best fidelity (0.0842, 0.9477, 28.9992), surpassing DiCache and TaylorSeer by a clear margin. At $\mathcal{B} = 25$ (1.96×), ERC again ranks first on all metrics (0.0653, 0.9711, 33.5913), confirming that our schedule construction generalizes well from aggressive to moderate acceleration regimes.

Tab. 2 shows similar trends on Wan2.1-T2V-1.3B. At $\mathcal{B} = 25$ (1.92×), ERC improves over LeMiCa on LPIPS (0.1133 vs. 0.1146), SSIM (0.9357 vs.

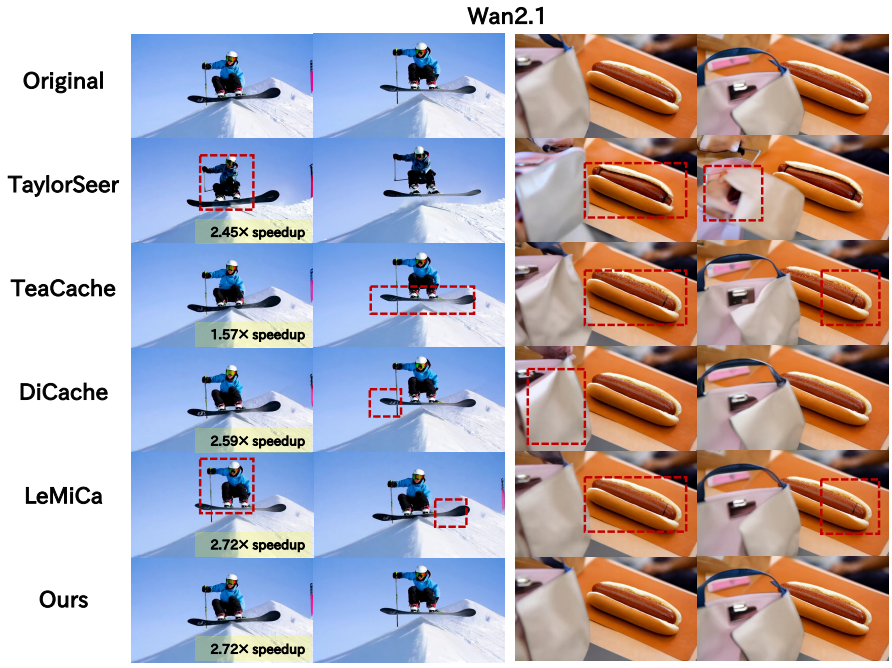

Fig. 4: Qualitative comparison in text-to-video generation. Best viewed zoomed in.

0.9345), PSNR (33.5746 vs. 33.5376), and VBench (74.64 vs. 74.53). At the more aggressive setting $\mathcal{B} = 17$ (2.72×), plain ERC is close to LeMiCa and achieves slightly higher VBench (74.08 vs. 73.98), while ERC+DCTA provides the strongest quality (0.1242, 0.9179, 30.6336) and VBench (74.80). It also substantially outperforms DiCache (0.1594, 0.8879, 28.2667) and TaylorSeer (0.3047, 0.7412, 20.2040). These results indicate that ERC offers a favorable and stable quality–speed trade-off across both text-to-image and text-to-video generation.

Qualitative analysis. To demonstrate the visual superiority of our approach, we provide qualitative comparisons against existing acceleration baselines. Fig. 3 illustrates the generated results on state-of-the-art text-to-image models, including FLUX and Qwen-Image. While achieving significant inference acceleration, our method maintains exceptional visual fidelity, producing images that are nearly indistinguishable from the original full inference baseline. In contrast, as highlighted by the red bounding boxes, alternative caching methods suffer from severe error accumulation, leading to structural distortions, missing details, and inconsistencies compared to the original outputs.

Furthermore, we extend our qualitative analysis to the challenging video generation task using Wan2.1, as shown in Fig. 4. Since video generation is highly sensitive to temporal feature deviations, baseline methods exhibit noticeable structural inconsistency with the original video frames (indicated by the red boxes). Conversely, our approach robustly preserves both spatial details and

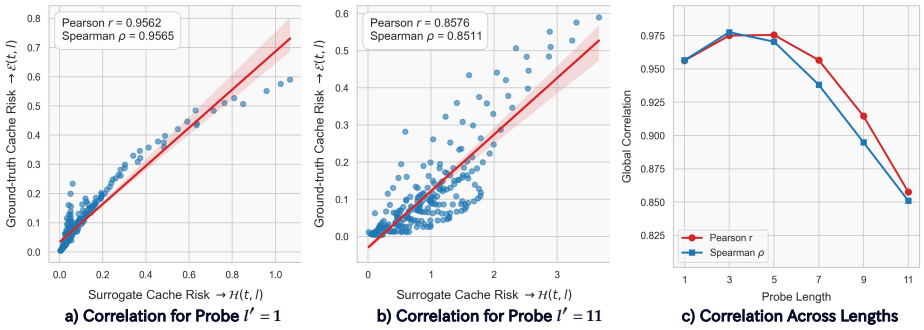

Fig. 5: Correlation for different probe lengths l' on FLUX.1 [dev].

temporal consistency, strictly aligning with the unaccelerated original model while providing substantial speedups.

4.3 Ablation Studies

Choice of the probe length l'. We instantiate the proxy as $r(t) = \mathcal{E}(t, l')$ and investigate how the probe length l' affects the quality of the resulting ERC schedule. Specifically, we vary $l' \in \{1, 3, 5, 7, 9, 11\}$, construct ERC schedules using the same procedure, and evaluate them under two recomputation budgets on FLUX.1 [dev].

Tab. 3 shows an overall trend: a moderate probe length l' works best, while overly short or long probes degrade performance. At both $\mathcal{B} = 15$ and $\mathcal{B} = 26$, $l' = 5$ achieves the best LPIPS/SSIM/PSNR simultaneously.

Furthermore, Fig. 5 provides the scatter plots for $l' = 1$ and $l' = 11$, together with correlations across probe lengths. The accumulated surrogate remains positively correlated with the 2D ground-truth, but the correlation decreases for overly short or long probes, with $l' = 11$ being the weakest. This observation is consistent with the results in the Tab. 3: a moderate probe length, e.g., $l' = 5$, gives the best PSNR, while $l' = 1$ still outperforms $l' = 11$.

A possible explanation is that very short probes may underestimate nonlinear accumulated errors, while very long probes mix different sensitive regions and obscure timestep-specific risk. Thus, a moderate probe length better balances local risk estimation and long-range error accumulation, leading to more effective recomputation schedules.

Number of offline prompts. We further analyze how many offline prompts are needed to build a reliable risk proxy $r(t)$. We fix the probing length to l' and vary the prompt count as $P \in \{1, 5, 10, 20, 50\}$. For each P, we construct the proxy with the same pipeline and evaluate the resulting schedule under two recomputation budgets.

As shown in Tab. 4, performance improves substantially from very small sample sizes to a moderate size ($P = 10$). For $\mathcal{B} = 15$, increasing P to 20 or 50 yields slight additional gains, while for $\mathcal{B} = 26$ the results at $P = 10, 20, 50$

Table 3: Ablation on the probing cache length l' for constructing $r(t) = \mathcal{E}(t, l')$ on FLUX.1 [dev].

l'	$\mathcal{B} = 15$ (3.23×)			$\mathcal{B} = 26$ (1.90×)		
	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓	SSIM ↑	PSNR ↑
1	0.2333	0.8475	25.2465	0.1199	0.9279	30.8135
3	0.2117	0.8547	25.3649	0.0960	0.9385	31.2367
5	0.1954	0.8628	25.4227	0.0816	0.9442	31.5304
7	0.1990	0.8603	25.0275	0.0891	0.9400	31.1941
9	0.2082	0.8486	24.3353	0.0920	0.9361	30.6320
11	0.2186	0.8415	24.0448	0.0997	0.9284	29.5831

Table 4: Ablation on the number of offline prompts P for constructing $r(t) = \mathcal{E}(t, l')$ with $l' = 5$ on FLUX.1 [dev].

P	$\mathcal{B} = 15$ (3.23×)			$\mathcal{B} = 26$ (1.90×)		
	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓	SSIM ↑	PSNR ↑
1	0.2350	0.8407	25.2008	0.1080	0.9313	30.6737
5	0.2131	0.8552	25.4343	0.1022	0.9329	30.6291
10	0.1954	0.8628	25.4227	0.0816	0.9442	31.5304
20	0.1936	0.8644	25.5033	0.0820	0.9419	31.2272
50	0.1926	0.8648	25.4993	0.0819	0.9464	31.6257

remain close with only minor fluctuations. This demonstrates that a moderate number of offline prompts already provides a stable and competitive proxy.

OOD prompt generalization. To further test distribution robustness, we construct an out-of-distribution (OOD) prompt set by sampling 200 prompts from GenAI [17], disjoint from the DrawBench prompts used in the main experiments. We keep the same inference configuration as in the FLUX.1 [dev] text-to-image setting and compare ERC ($l' = 5$) with LeMiCa under the same recomputation budgets $\mathcal{B} \in \{15, 26\}$.

As shown in Tab. 5, at the more aggressive budget ($\mathcal{B} = 15$), the two methods are close: LeMiCa is slightly better on LPIPS (0.1960 vs. 0.1988), while ERC achieves higher SSIM/PSNR (0.8689/25.5790 vs. 0.8658/25.0213). At $\mathcal{B} = 26$, ERC outperforms LeMiCa on all three metrics (0.0822, 0.9491, 31.9375 vs. 0.0921, 0.9383, 29.9245), indicating stronger generalization to unseen prompt distributions.

Performance in few-step settings. We evaluate our method, MagCache, and DiCache using DPM-Solver++ 2M with $T = 30$ on Qwen Image, as shown in Tab. 6. All methods directly reuse the previous-step residual cache without DCTA. Since LeMiCa has not released its DAG construction code, we could not reliably reproduce it for this setting.

Table 5: OOD prompt evaluation on a 200-prompt GenAI set (disjoint from DrawBench) under the FLUX.1 [dev] setting. Best results are in bold.

Method	$\mathcal{B} = 15$ (3.23×)			$\mathcal{B} = 26$ (1.90×)		
	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓	SSIM ↑	PSNR ↑
LeMiCa	**0.1960**	0.8658	25.0213	0.0921	0.9383	29.9245
Ours	0.1988	**0.8689**	**25.5790**	**0.0822**	**0.9491**	**31.9375**

Table 6: Performance in few-step settings on Qwen Image.

Method	LPIPS↓	SSIM↑	PSNR↑	Speed↑
Ours ($\mathcal{B} = 15$)	**0.0694**	**0.9741**	**34.4771**	1.95×
MagCache	0.0708	0.9676	32.6547	1.95×
DiCache	0.0794	0.9655	32.4783	1.88×

5 Conclusion

In this paper, we present Equal-Risk Caching (ERC), a training-free caching method for accelerating diffusion models. Our method is motivated by an empirical correlation between the expensive ground-truth cache risk and a lightweight surrogate risk estimated from probing runs. Based on this observation, ERC constructs refresh schedules by equal-mass partitioning on the cumulative risk axis, allocating computation adaptively across timesteps. This design reduces schedule construction complexity from $\mathcal{O}(T \cdot L)$ to $\mathcal{O}(T)$ and maintains practical flexibility. Experiments on both text-to-image and text-to-video models show an overall competitive quality–speed trade-off across settings.

References

1. Black Forest Labs: Flux.1-dev. `https://huggingface.co/black-forest-labs/FLUX.1-dev` (2024), accessed: 2025-11-27
2. Blattmann, A., Dockhorn, T., Kulal, S., Mendelevitch, D., Kilian, M., Lorenz, D., Levi, Y., English, Z., Voleti, V., Letts, A., et al.: Stable video diffusion: Scaling latent video diffusion models to large datasets. arXiv preprint arXiv:2311.15127 (2023)
3. Blattmann, A., Rombach, R., Ling, H., Dockhorn, T., et al.: Align your latents: High-resolution video synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 22563–22575 (2023)
4. Brooks, T., Peebles, B., Holmes, C., DePue, W., Guo, Y., Jing, L., Schnurr, D., Taylor, J., Luhman, T., Luhman, E., Ng, C., Wang, R., Ramesh, A.: Video generation models as world simulators. OpenAI technical report (2024), `https://openai.com/index/video-generation-models-as-world-simulators/`, accessed: 2026-06-28

5. Bu, J., Ling, P., Zhou, Y., Wang, Y., Zang, Y., Lin, D., Wang, J.: Dicache: Let diffusion model determine its own cache. In: International Conference on Learning Representations (2026)
6. Chen, H., Zhang, Y., Cun, X., Xia, M., Wang, X., Weng, C., Shan, Y.: Videocrafter2: Overcoming data limitations for high-quality video diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7310–7320 (2024)
7. Chen, L., Meng, Y., Tang, C., Ma, X., Jiang, J., Wang, X., Wang, Z., Zhu, W.: Q-dit: Accurate post-training quantization for diffusion transformers. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 28306–28315 (2025)
8. 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)
9. Dao, T.: Flashattention-2: Faster attention with better parallelism and work partitioning. In: International Conference on Learning Representations. vol. 2024, pp. 35549–35562 (2024)
10. Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. Advances in Neural Information Processing Systems **34**, 8780–8794 (2021)
11. Gao, H., Chen, P., Shi, F., Tan, C., Liu, Z., Zhao, F., Wang, K., Lian, S.: Lemica: Lexicographic minimax path caching for efficient diffusion-based video generation. In: Annual Conference on Neural Information Processing Systems (2025)
12. Guo, Y., Yang, C., Rao, A., Liang, Z., Wang, Y., Qiao, Y., Agrawala, M., Lin, D., Dai, B.: Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. arXiv preprint arXiv:2307.04725 (2023)
13. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in Neural Information Processing Systems **33**, 6840–6851 (2020)
14. Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598 (2022)
15. Hore, A., Ziou, D.: Image quality metrics: Psnr vs. ssim. In: 2010 20th international conference on pattern recognition. pp. 2366–2369. IEEE (2010)
16. Huang, Z., He, Y., Yu, J., Zhang, F., Si, C., Jiang, Y., Zhang, Y., Wu, T., Jin, Q., Chanpaisit, N., Wang, Y., Chen, X., Wang, L., Lin, D., Qiao, Y., Liu, Z.: VBench: Comprehensive benchmark suite for video generative models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2024)
17. Li, B., Lin, Z., Pathak, D., Li, J., Fei, Y., Wu, K., Ling, T., Xia, X., Zhang, P., Neubig, G., et al.: Genai-bench: Evaluating and improving compositional text-to-visual generation. arXiv preprint arXiv:2406.13743 (2024)
18. Li, M., Lin, Y., Zhang, Z., Cai, T., Li, X., Guo, J., Xie, E., Meng, C., Zhu, J.Y., Han, S.: Svdquant: Absorbing outliers by low-rank component for 4-bit diffusion models. In: International Conference on Learning Representations. vol. 2025, pp. 97831–97858 (2025)
19. Lipman, Y., Chen, R.T., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. arXiv preprint arXiv:2210.02747 (2022)
20. 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 (2025)
21. Liu, J., Zou, C., Lyu, Y., Chen, J., Zhang, L.: From reusing to forecasting: Accelerating diffusion models with taylorseers. arXiv preprint arXiv:2503.06923 (2025)

22. Liu, J., Shen, X.: Fourier token merging: Understanding and capitalizing frequency domain for efficient image generation. In: Advances in Neural Information Processing Systems. vol. 38, pp. 23904–23925 (2025)
23. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. In: Advances in neural information processing systems. vol. 35, pp. 5775–5787 (2022)
24. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. Machine Intelligence Research **22**(4), 730–751 (2025)
25. Lu, W., Zheng, S., Xia, Y., Wang, S.: Toma: Token merge with attention for diffusion models. In: Proceedings of the 42nd International Conference on Machine Learning. vol. 267, pp. 40930–40951 (2025)
26. Lv, Z., Si, C., Song, J., Yang, Z., Qiao, Y., Liu, Z., Wong, K.Y.K.: Fastercache: Training-free video diffusion model acceleration with high quality. arXiv preprint arXiv:2410.19355 (2024)
27. Ma, X., Fang, G., Bi Mi, M., Wang, X.: Learning-to-cache: Accelerating diffusion transformer via layer caching. In: Advances in Neural Information Processing Systems. vol. 37, pp. 133282–133304 (2024)
28. 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 (June 2024)
29. Ma, Z., Wei, L., Wang, F., Zhang, S., Tian, Q.: Magcache: Fast video generation with magnitude-aware cache. In: Advances in Neural Information Processing Systems. vol. 38, pp. 34348–34380 (2025)
30. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4195–4205 (2023)
31. Podell, D., English, Z., Lacey, K., Blattmann, A., et al.: Sdxl: Improving latent diffusion models for high-resolution image synthesis. arXiv preprint arXiv:2307.01952 (2023)
32. 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)
33. Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E.L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al.: Photorealistic text-to-image diffusion models with deep language understanding. Advances in neural information processing systems **35**, 36479–36494 (2022)
34. Sauer, A., Lorenz, D., Blattmann, A., Rombach, R.: Adversarial diffusion distillation. In: European Conference on Computer Vision. pp. 87–103. Springer (2024)
35. Selvaraju, P., Ding, T., Chen, T., Zharkov, I., Liang, L.: Fora: Fast-forward caching in diffusion transformer acceleration. arXiv preprint arXiv:2407.01425 (2024)
36. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: International conference on machine learning. vol. 37, pp. 2256–2265. pmlr (2015)
37. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. In: International Conference on Learning Representations (2021)
38. Song, Y., Ermon, S.: Generative modeling by estimating gradients of the data distribution. In: Advances in neural information processing systems. vol. 32 (2019)
39. Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. In: International Conference on Learning Representations (2021)

40. Wan, T., Wang, A., Ai, B., Wen, B., Mao, C., Xie, C.W., Chen, D., Yu, F., Zhao, H., Yang, J., et al.: Wan: Open and advanced large-scale video generative models. arXiv preprint arXiv:2503.20314 (2025)
41. Wang, X., Zhang, S., Zhang, H., Liu, Y., Zhang, Y., Gao, C., Sang, N.: Videolcm: Video latent consistency model. arXiv preprint arXiv:2312.09109 (2023)
42. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing **13**(4), 600–612 (2004)
43. 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)
44. Xi, H., Yang, S., Zhao, Y., Xu, C., Li, M., Li, X., Lin, Y., Cai, H., Zhang, J., Li, D., et al.: Sparse videogen: Accelerating video diffusion transformers with spatial-temporal sparsity. arXiv preprint arXiv:2502.01776 (2025)
45. 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 (2018)
46. Zhao, W., Bai, L., Rao, Y., Zhou, J., Lu, J.: Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. In: Advances in Neural Information Processing Systems. vol. 36, pp. 49842–49869 (2023)
47. Zhao, X., Jin, X., Wang, K., You, Y.: Real-time video generation with pyramid attention broadcast. arXiv preprint arXiv:2408.12588 (2024)
48. 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)