DART: DIFFUSION ACCELERATION VIA RECTIFIED TEMPORAL QUANTIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion models achieve high generative quality but incur high computational cost from iterative denoising. Feature caching and network quantization each reduce this cost, and combining them by jointly optimizing cache schedule and Quantization can yield multiplicative speedups. However, existing methods that determine cache schedule relied on inter-step similarities without considering the stability of the global sampling process and the quantization strategy fails to consider the heterogeneity across different stages of the sampling trajectory. To overcome these limitations, we propose DART, a joint caching-and-quantization framework that controls error propagation through stability-aware trajectory partitioning. Specifically, we introduce Bounded Partition Programming (BPP), which leverages the spectral radius variations of the sampling process as a constraint term to ensure sampling stability while implementing cache scheduling, creating provably stable temporal segments. Within these segments, Dual Affine Rectification (DAR) decouples error sources through input-domain correction and operator-domain correction. A Temporally Stamped Quantizer (TSQ) exploits BPP's stability structure for adaptive low-bit inference, while Normalization-Preserving Fusion (NPF) achieves zero-overhead deployment. On ImageNet-256, DART achieves 8–19$\times$ speedup with FID 3.83 at $8.12\times$ acceleration, where direct caching-quantization combinations degrade to FID 15.36 at comparable speed.

1 INTRODUCTION

Diffusion models Ho et al. (2020) have set the state of the art in image and video synthesis, but their iterative denoising process incurs substantial inference cost. Two complementary directions have emerged for acceleration: *feature caching* Ma et al. (2024b); Selvaraju et al. (2024); Liu et al. (2025a;b); Zou et al. (2025); Bu et al. (2026) reuses intermediate activations across adjacent timesteps to skip network evaluations, and *post-training quantization* Li et al. (2023a); He et al. (2023); Wu et al. (2024) reduces per-step arithmetic and memory cost through low-bit integer representations. Each delivers notable speedups in isolation, and combining the two Liu et al. (2025d); Ding et al. (2025) promises multiplicative gains by acting on orthogonal axes of redundancy.

Joint cache–quantization is fragile in practice. A direct combination of caching and quantization is far less benign than the sum of its parts. As shown in Figure 1, applying a strong caching schedule on top of an off-the-shelf low-bit model causes the deviation from the full-precision trajectory to grow rapidly with the denoising step, eventually producing structural distortion at speedups where caching-only and quantization-only baselines remain stable. State-of-the-art joint methods Liu et al. (2025d); Ding et al. (2025) mitigate this only partially: their cache schedule is driven by the local feature distance between adjacent steps, while their quantization parameters are shared across the entire trajectory. Both choices are at odds with the underlying dynamics of denoising, and the resulting error compounds along the sampling path.

Local feature distance is not a reliable predictor. Existing cache schedulers Ma et al. (2024b); Liu et al. (2025d); Wimbauer et al. (2024) place recomputation points where the L_1 feature distance between adjacent steps is large. Recent work Chu et al. (2025); Cui et al. (2026b); Haghighi & Alahi (2026) questions this assumption: feature similarity may stay high across two steps that nonetheless lie on opposite sides of a sensitive transition, pushing the denoiser into a regime where small perturbations are amplified. The problem is sharper under joint cache–quantization, because

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

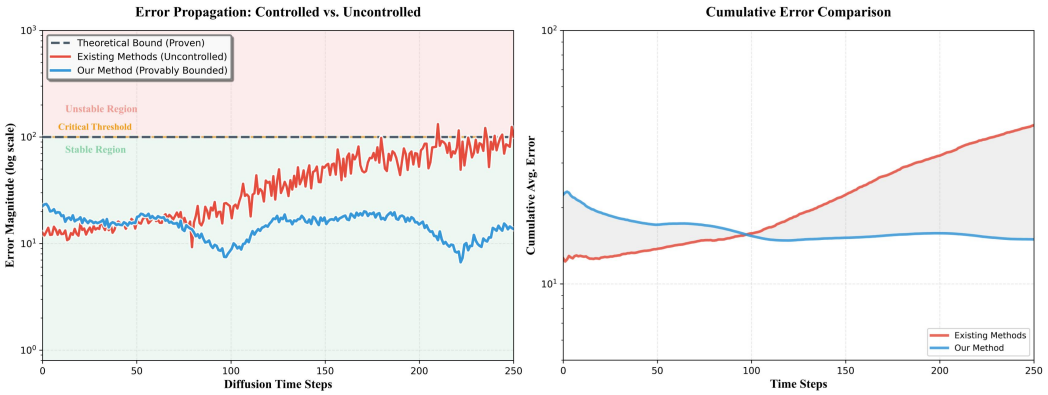

Figure 1: Error growth along the denoising trajectory under different acceleration strategies. (Left) per-step deviation from the full-precision reference on a log scale: a direct cache–quant combination and CacheQuant Liu et al. (2025d) accumulate error rapidly, while the DART curve in blue keeps the deviation tightly bounded. (Middle) cumulative error: the shaded area highlights the compounding gap.

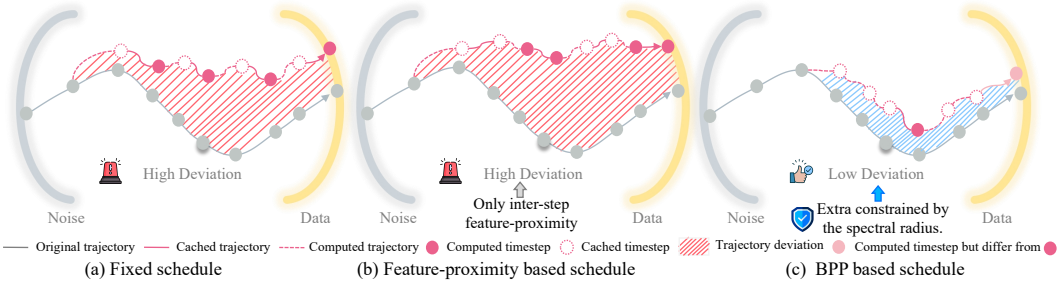

Figure 2: Three families of cache schedules under joint cache–quantization. (a) A fixed-interval schedule cannot adapt to where the trajectory becomes sensitive. (b) A feature-distance–driven schedule may skip steps that look locally similar but lie across a sensitive transition, where the quantized operator amplifies cache-induced shift. (c) DART partitions the trajectory into segments inside which the relative-update sensitivity has bounded fluctuation, so that cache reuse and quantization parameter sharing are confined to dynamically homogeneous regions.

feature distance does not see the quantization-induced operator distortion; schedulers that work for caching alone can place cache reuses precisely at the steps where low-bit operators are least tolerant, producing the divergent behaviour in Figure 1. Beyond distributional metrics such as FID, this trajectory-level error amplification also degrades per-sample fidelity to the full-precision output and temporal consistency for video; we therefore treat trajectory deviation as a first-class quantity and evaluate both FID and reference-based LPIPS/PSNR/SSIM throughout.

A trajectory-sensitivity view of joint acceleration. We answer two questions in a unified way: *when to recompute* and *how to use cached quantized states*. We introduce a relative update sensitivity proxy $\hat{\rho}_t = \|\boldsymbol{X}_{t-1} - \boldsymbol{X}_t\|_2 / (\|\boldsymbol{X}_t\|_2 + \varepsilon)$ that empirically tracks the local amplification behaviour of the denoiser without requiring an explicit Jacobian computation. Constraining the fluctuation of $\hat{\rho}_t$ within each segment yields contiguous regions in which the dynamics are approximately homogeneous, providing a natural granularity for both cache reuse and segment-shared quantization. The same partitioning enables a decoupled correction of the two error modes – a cache-induced input shift and a quantization-induced operator distortion – each rectified by a channel-wise affine map fitted on a small calibration set. We instantiate this framework as DART with four interlocked components: *Trajectory-Sensitivity Bounded Partitioning (BPP)* for the cache schedule, *Dual Affine Rectification (DAR)* for the two error modes, *Temporally Stamped Quantizer (TSQ)* for segment-shared low-bit calibration, and *Normalization-Preserving Fusion (NPF)* that folds all rectification parameters into surrounding operators so that inference-time computation is essentially unchanged.

In summary, our main contributions are as follows:

1. **Diagnosis.** We show that directly combining caching and quantization induces *coupled* temporal drift and operator distortion, producing trajectory-level error amplification that local feature similarity cannot reliably predict.
2. **Method.** We propose a trajectory-sensitivity bounded partitioning that decides safe recomputation points, together with segment-wise dual-domain rectification and a segment-shared temporally stamped quantizer that jointly correct cached low-bit states.
3. **Deployment and evidence.** We design a normalization-preserving fusion that absorbs all rectification parameters into existing operators, and validate DART on LDM, DiT-XL/2 and HunyuanVideo. Under matched-speed and matched bit-width settings, DART consistently improves the quality–efficiency frontier over fair caching, quantization, and joint baselines, achieving 8–19$\times$ speedup on LDM-4 ImageNet-256 and sustaining FID 6.53 at the most aggressive $18.93\times$ setting, where a direct cache–quantization combination degrades to FID 15.36 at comparable speed.

2 Related Work

Sampling-step reduction and distillation. Accelerating diffusion by reducing the number of network evaluations includes ODE/SDE solvers Song et al. (2021); Lu et al. (2025), distillation Feng et al. (2024), pruning Fang et al. (2023), and lightweight architectures Li et al. (2023b). These directions are orthogonal to ours: DART operates on a pretrained denoiser with a fixed solver in a training-free fashion, so distillation- and solver-based speedups can be composed with it.

Training-free feature caching and forecasting. DeepCache Ma et al. (2024b) and Faster Diffusion Li et al. (2024a) cache U-Net upsampling-block features; FORA Selvaraju et al. (2024) and Δ-DiT Chen et al. (2024) extend this to diffusion transformers. AdaCache Kahatapitiya et al. (2024) and TeaCache Liu et al. (2025a) drive adaptive schedules from feature distance or residual magnitude, while OmniCache Chu et al. (2025) and DPCache Cui et al. (2026b) cast cache scheduling as a global optimization over the sampling trajectory. Token-wise Zou et al. (2025) and dual-cache Zou et al. (2024) variants exploit heterogeneity across tokens. Prediction-based methods replace pure reuse with active extrapolation: TaylorSeer Liu et al. (2025b) uses truncated Taylor expansions, TC-Padé Cui et al. (2026a) uses Padé to model phase transitions, Spectrum Han et al. (2026) performs subspace-aware forecasting, and TAP Zhu et al. (2026) selects a per-token predictor through a first-layer probe. SenCache Haghighi & Alahi (2026) and DiCache Bu et al. (2026) make reuse sample-adaptive through sensitivity-aware or shallow-probe signals; ECAD Aggarwal et al. (2026) searches Pareto-optimal schedules evolutionarily. These works collectively motivate two design principles that we adopt: cache scheduling should track sampling-trajectory sensitivity rather than local feature distance Chu et al. (2025); Haghighi & Alahi (2026); Cui et al. (2026a), and the proper granularity of any acceleration policy is the homogeneous regime, not the entire trajectory Liu et al. (2025b); Cui et al. (2026a); Zhu et al. (2026). Our trajectory-sensitivity bounded partitioning instantiates both principles for the joint cache–quantization setting.

Post-training quantization for diffusion models. Q-Diffusion Li et al. (2023a), PTQ4DM Shang et al. (2023), and PTQD He et al. (2023) adapt PTQ to UNet-based diffusion models; PTQ4DiT Wu et al. (2024) and Ditto Kim et al. (2025) extend it to transformer backbones; RepQ Li et al. (2023c) and EDA-DM Shen et al. (2024) address activation outliers and step-conditional reconstruction. Diffusion activations have markedly different dynamic ranges across timesteps, so a single global calibration is suboptimal at high acceleration; we share quantization parameters within each segment identified by trajectory sensitivity instead of calibrating per-timestep.

Joint caching and quantization. CacheQuant Liu et al. (2025d) and Q&C Ding et al. (2025) combine quantization with feature caching. Both determine cache boundaries by local feature similarity and share one quantizer globally, neither of which predicts the error amplification of cached *quantized* states (Figure 1). DART differs in three concrete ways: (i) the cache schedule is driven by the relative-update sensitivity proxy $\hat{\rho}_t$ with a bounded-fluctuation constraint, turning scheduling into a constrained DP; (ii) rectification is decomposed into an input-domain correction for cache-induced shift and an operator-domain correction for quantization-induced distortion, each fitted within a sensitivity-homogeneous segment; and (iii) the quantizer is segment-shared along the same partitioning, so that the cache schedule and the quantizer share a common notion of homogeneity. We verify each component independently in Section 4.

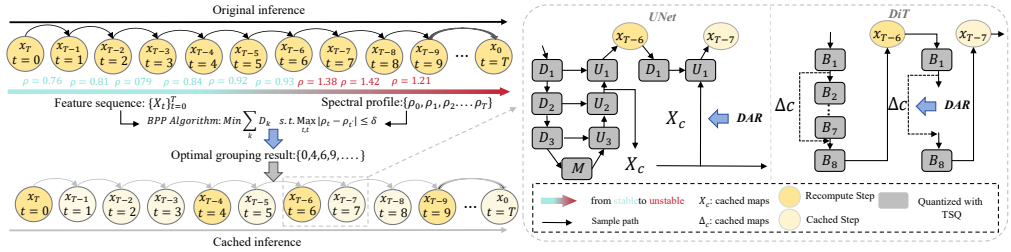

Figure 3: Overview of DART. **BPP** chooses recomputation points by partitioning the trajectory into sensitivity-homogeneous segments. **DAR** corrects the cache-induced input shift and the quantization-induced operator distortion independently. **TSQ** shares quantization parameters within each segment, and **NPF** folds all rectification parameters into adjacent linear and dequantization operators at deployment.

3 METHODOLOGY

3.1 PRELIMINARIES

Cache schedule. For a trajectory of length T, a schedule specifies the subset of timesteps at which the network is fully recomputed; the remaining steps reuse cached features. A fixed-interval schedule with frequency N takes the form $\{0, N, 2N, \dots\}$ (Figure 2(a)). Prior work either tunes the schedule by hyperparameter search Ma et al. (2024b); Wimbauer et al. (2024); Chen et al. (2024) or specifies it manually Li et al. (2024a); OmniCache Chu et al. (2025) drives adaptation from local sampling-trajectory statistics, and CacheQuant Liu et al. (2025d), DPCache Cui et al. (2026b) cast scheduling as a dynamic program on inter-step feature distance. Recent studies Chu et al. (2025); Haghighi & Alahi (2026); Cui et al. (2026a) report that local feature similarity is a weak predictor of cache-induced quality loss, especially under aggressive acceleration. Motivated by these findings and the dynamical-system view of sampling Biroli et al. (2024), we cast scheduling as a constrained partitioning problem on a relative-update sensitivity signal.

Network quantization. An asymmetric uniform quantizer maps $\mathbf{x}$ to $\hat{\mathbf{x}} = \text{clip}(\lfloor \mathbf{x}/s \rceil + z,\ 0,\ 2^b - 1)$ with scale s, zero-point z, and bit-width b Nagel et al. (2019); Li et al. (2023c); Wei et al. (2022); Nagel et al. (2020); Li et al. (2024b). DART belongs to the post-training regime, but unlike methods that calibrate a single quantizer globally across the trajectory, we share quantization parameters within each segment identified by trajectory sensitivity (Section 3.5).

3.2 TRAJECTORY-SENSITIVITY ANALYSIS

DART is a training-free joint cache–quantization framework that answers two questions: *when to recompute* and *how to use cached quantized states*. This section introduces the trajectory-sensitivity proxy that drives the entire design; Sections 3.3–3.6 then describe the four components in Figure 3.

Following Biroli et al. (2024); Cui et al. (2026a), we view the reverse process as a discrete dynamical system $\boldsymbol{X}_{t-1} = F_t(\boldsymbol{X}_t, \theta_t, \epsilon_t)$, where ϵ_t encapsulates perturbations such as cache- or quantization-induced noise. Local error amplification is governed by the Jacobian $\boldsymbol{J}_t = \partial F_t / \partial \boldsymbol{X}_t$, intractable to compute for large diffusion backbones. We instead use a finite-difference *relative-update sensitivity proxy* Haghighi & Alahi (2026):

$$\hat{\rho}_t \ := \ \frac{\|\boldsymbol{X}_{t-1} - \boldsymbol{X}_t\|_2}{\|\boldsymbol{X}_t\|_2 + \varepsilon}, \tag{1}$$

with small ε for stability. By a first-order expansion $\boldsymbol{X}_{t-1} - \boldsymbol{X}_t \approx (\boldsymbol{J}_t - \boldsymbol{I})\boldsymbol{X}_t$, $\hat{\rho}_t$ reflects the magnitude of $(\boldsymbol{J}_t - \boldsymbol{I})$ projected onto the trajectory direction; we therefore treat it as a directional, Lipschitz-like sensitivity signal. We do *not* claim that $\hat{\rho}_t$ is the spectral radius of $\boldsymbol{J}_t$ or that the resulting partitioning admits a formal stability guarantee. Empirically (Appendix A.8), $\hat{\rho}_t$ correlates with the cache-plus-quantization error more strongly than inter-step feature distance. Figure 4 visualizes $\hat{\rho}_t$ on LDM-4: the original sampler stays in a regime of mild fluctuation, whereas a direct cache–quant combination produces large excursions that coincide with the failures in Figure 1. We use the *fluctuation* of $\hat{\rho}_t$, not its absolute level, as the criterion for placing recomputation points.

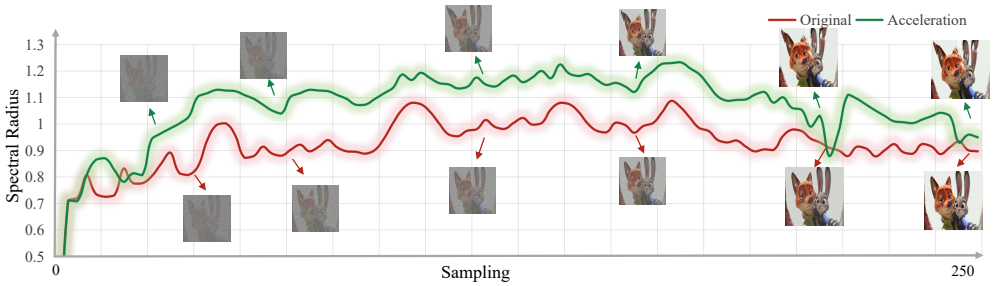

Figure 4: Relative-update sensitivity $\hat{\rho}_t$ along the denoising trajectory. The original sampler stays in a regime of mild fluctuation, while a direct cache–quant combination drives $\hat{\rho}_t$ into large excursions that coincide with visible degradation. We use the fluctuation pattern of $\hat{\rho}_t$, not its absolute value, as a proxy for the local sensitivity of the denoiser.

3.3 Trajectory-Sensitivity Bounded Partitioning (BPP)

BPP segments $[0, T)$ into K contiguous intervals $G_k = [s_k, s_{k+1})$ with boundaries $S = \{s_1, \ldots, s_K\}$ ($s_1 = 0$, $s_{K+1} = T$). The schedule recomputes at each s_k and reuses the cached state inside G_k. We choose S by solving

$$\begin{aligned} \min_{S} \quad & \sum_{k=1}^{K} \sum_{t=s_k+1}^{s_{k+1}-1} D_k(s_k, t) \\ \text{s.t.} \quad & N/2 \le s_{k+1} - s_k \le 2N, \qquad \max_{t,t' \in [s_k, s_{k+1})} \left| \hat{\rho}_t - \hat{\rho}_{t'} \right| \le \delta, \quad \forall k, \end{aligned} \tag{2}$$

where $D_k(s_k, t) = \|\boldsymbol{X}_t - \boldsymbol{X}_{s_k}\|_1$ approximates the cumulative drift inside G_k if step s_k were cached and reused. The length constraint pins the average segment length around the target N and bounds the worst-case reuse horizon; the bounded-fluctuation constraint on $\hat{\rho}_t$ restricts each segment to a regime where local sensitivity is approximately homogeneous. The $\hat{\rho}_t$ signal in equation 1 is computed once on the calibration set and is shared across BPP, DAR, and TSQ. We solve equation 2 by dynamic programming in $O(T \cdot K_{\max} \cdot N)$ time; the full pseudocode, complexity analysis, and the infeasibility fallback (δ += 0.05 until a feasible partition exists) are given in Appendix A.3. We use $\delta = 0.3$ throughout; the sweep in Section 4 shows that the optimum is consistent across the tested settings.

3.4 Dual Affine Rectification (DAR)

Joint cache–quantization produces two qualitatively different errors. Caching replaces $\boldsymbol{X}_t$ with an earlier feature, producing an *input-domain shift* ϵ_c. Quantization replaces the floating-point operator F_t with a low-bit operator $\mathcal{Q}_k \circ F_t$, producing an *operator-domain distortion* ϵ_q. DAR applies one channel-wise affine correction at each side of the operator:

$$\hat{\boldsymbol{X}}_t = \mathrm{diag}(\boldsymbol{a}_{1,t})\boldsymbol{X}_{s_k} + \boldsymbol{b}_{1,t}, \qquad \hat{\boldsymbol{O}}_{t-1} = \mathrm{diag}(\boldsymbol{a}_{2,t})\mathcal{Q}_k(F_t(\hat{\boldsymbol{X}}_t)) + \boldsymbol{b}_{2,t}, \tag{3}$$

where $(\boldsymbol{a}_{1,t}, \boldsymbol{b}_{1,t})$ undoes the temporal mismatch between $\boldsymbol{X}_{s_k}$ and $\boldsymbol{X}_t$ and $(\boldsymbol{a}_{2,t}, \boldsymbol{b}_{2,t})$ undoes the systematic bias of the low-bit operator. Within a sensitivity-homogeneous segment, each residual is well captured by an affine map per channel, and the four parameters are estimated in closed form from the covariance of the corresponding pre- and post-operator signals on a calibration set; the exact estimator and its storage cost ($\mathcal{O}(T \cdot C)$ per layer) are given in Appendix A.4. Section 4 ablates the two corrections independently and shows that each contributes a non-overlapping share of the gain.

3.5 Temporally Stamped Quantizer (TSQ)

Activation statistics in diffusion sampling vary substantially across timesteps Wu et al. (2024); Li et al. (2023c). A globally calibrated quantizer compromises between regimes with very different dynamic ranges, while a per-step quantizer multiplies storage and calibration cost. TSQ uses the

Table 1: Performance on ImageNet-256 using LDM-4 with 250-step DDIM. † indicates retraining required.

Method	Bops↓	Speedup↑	Memory↓	Retrain	FID↓	IS↑	Precision↑	Recall↑
LDM-4	102.22T	1.00×	1824.6	-	3.37	204.56	0.81	0.59
DeepCache-N=5	24.06T	4.12×	1824.6	-	3.79	199.58	0.81	0.60
DeepCache-N=10	14.31T	6.96×	1824.6	-	4.60	188.81	0.82	0.58
EDA-DM (W8A8)†	6.39T	1.91×	457.1	✓	4.13	186.78	0.80	0.58
EDA-DM (W4A8)†	3.19T	1.91×	229.2	✓	4.79	176.43	0.79	0.57
Diff-Pruning†	53.98T	1.51×	757.7	✓	9.27	214.42	0.82	0.54
CacheQuant-N=5 (W8A8)	1.50T	7.87×	457.1	-	4.03	193.90	0.81	0.59
CacheQuant-N=10 (W8A8)	0.89T	12.20×	457.1	-	4.68	184.38	0.82	0.58
CacheQuant-N=20 (W8A8)	0.60T	18.06×	457.1	-	7.21	160.68	0.81	0.56
DART-N=5 (W8A8)	1.48T	8.12×	452.3	-	3.83	198.12	0.82	0.60
DART-N=10 (W8A8)	0.87T	12.85×	450.1	-	4.38	191.27	0.82	0.59
DART-N=20 (W8A8)	0.58T	18.93×	447.5	-	6.53	169.42	0.81	0.57
CacheQuant-N=5 (W4A8)†	0.75T	7.87×	229.2	✓	6.26	168.46	0.79	0.57
DART-N=5 (W4A8)†	0.74T	8.12×	226.8	✓	5.72	174.63	0.80	0.58
DART-N=10 (W4A8)†	0.44T	12.85×	223.7	✓	6.20	174.08	0.80	0.58
DART-N=20 (W4A8)†	0.29T	18.93×	226.4	✓	8.90	157.24	0.79	0.56

BPP segmentation as the natural granularity: within each G_k, the bounded fluctuation of $\hat{\rho}_t$ implies that activation distributions remain statistically coherent, so a single quantizer suffices. For each segment we calibrate one activation quantizer $(\boldsymbol{\alpha}_k, \boldsymbol{z}_k)$ at the boundary s_k with uniform asymmetric per-tensor activation and per-channel symmetric weight quantization, minimizing the expected ℓ_2 reconstruction error with percentile clipping. Deployment amounts to a precomputed table $M : t \mapsto k$ that maps each step to its segment; storage is $\mathcal{O}(K)$ per layer, an order smaller than per-step calibration with $K \ll T$. The per-step quantization error inside each segment is reported in Appendix A.11.

3.6 Normalization-Preserving Fusion (NPF)

A direct implementation of DAR adds two affine layers around every quantized operator. NPF removes this overhead by folding the rectification parameters into adjacent linear, convolutional, or dequantization operators whenever the underlying fusion is mathematically exact. For a linear layer W with bias b acting on $\hat{\boldsymbol{X}}_t$, the input-side fusion sets

$$W' = W \operatorname{diag}(\boldsymbol{a}_{1,t}), \qquad b' = W\boldsymbol{b}_{1,t} + b, \tag{4}$$

making the fused layer pointwise equivalent to applying the rectification followed by the original layer, with no extra computation. The analogous fold into the dequantization parameters on the output side and the precise conditions under which fusion into normalization layers is exact are given in Appendix A.7; the deployment-time overhead reduces to a constant-time lookup of $(\boldsymbol{a}, \boldsymbol{b})$ and $(\boldsymbol{\alpha}, \boldsymbol{z})$ once per step. Appendix A.12 reports the absolute latency, memory, and storage with NPF on and off.

4 Experiments

4.1 Experimental Setup

Datasets, models and metrics. We evaluate DART on class-conditional image generation with LDM Rombach et al. (2022) and DiT-XL/2 Peebles & Xie (2023a) on ImageNet-256 Deng et al. (2009), and text-to-video generation with HunyuanVideo Kong et al. (2024). Following Liu et al. (2025e); Ma et al. (2024b), we generate 10k images for DiT-XL/2. For text-to-video, we adopt VBench Huang et al. (2024) (946 prompts, five samples per prompt at $640 \times 480 \times 65$, 4,730 videos). Quality: FID, IS, Precision/Recall, CLIP Score. Efficiency: speedup ratio, Bops, GPU memory. Fidelity: PSNR, SSIM Wang et al. (2004), LPIPS Zhang et al. (2018).

Implementation. BPP uses $\delta = 0.3$ with target segment lengths $N \in \{5, 10, 20\}$, calibrated on 512 samples. We use 250-step DDIM Song et al. (2021) for LDM and DDPM for DiT-XL/2 at 250 and 50 steps. DAR statistics are collected after weight reconstruction. All FID/IS results are averaged over 3 runs. Experiments run on NVIDIA A100 GPUs.

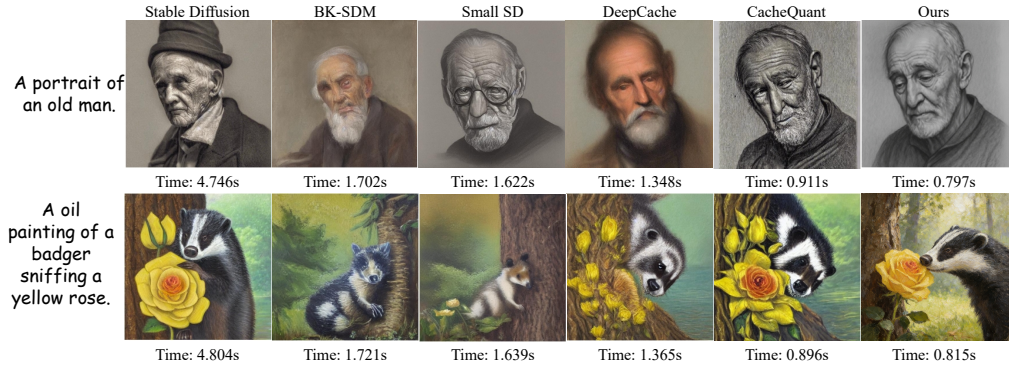

Figure 5: Qualitative comparison on ImageNet-256 (250-step DDIM, W8A8).

Table 2: Performance comparison on DiT-XL/2 Peebles & Xie (2023b) (ImageNet 256×256) with W8A8 quantization and DDPM solver. Best per column shaded dark; second best light.

Method	Type	250 Steps Speed↑	FID↓	sFID↓	IS↑	50 Steps Speed↑	FID↓	sFID↓	IS↑
DDPM	–	1×	4.53	17.93	278.50	5×	5.22	17.63	237.80
PTQ4DM Shang et al. (2023)	Quant	2×	21.65	100.14	134.22	10×	17.52	84.28	154.08
Q-Diffusion Li et al. (2023a)	Quant	2×	5.57	18.22	227.50	10×	14.61	27.57	153.01
PTQD He et al. (2023)	Quant	2×	5.69	18.42	224.26	10×	15.21	27.52	151.60
RepQ Li et al. (2023c)	Quant	2×	4.51	18.01	264.68	10×	7.17	23.67	224.83
PTQ4DiT Wu et al. (2024)	Quant	2×	4.63	17.72	274.86	10×	5.45	19.50	250.68
FORA Selvaraju et al. (2024)	Cache	2.06×	5.06	18.12	163.84	–	–	–	–
Learn-to-Cache Ma et al. (2024a)	Cache	–	–	–	–	6.3×	5.21	17.60	245.45
Δ-DiT Chen et al. (2024)	Cache	2.46×	5.13	17.89	240.64	7.6×	11.9	24.42	238.42
TaylorSeer Liu et al. (2025b)	Cache	2.79×	4.94	18.32	259.45	9.6×	6.32	19.34	247.32
CacheQuant Liu et al. (2025d)	Quant+Cache	3.17×	4.63	17.72	257.83	12.5×	5.36	19.13	243.95
Q&C Ding et al. (2025)	Quant+Cache	3.12×	4.68	17.84	268.65	12.7×	5.43	19.52	250.68
DPCache Cui et al. (2026b)	Cache	–	–	–	–	12.4×	6.01	18.57	233.83
DART-N=2 (Ours)	Quant+Cache	2.20×	4.50	17.85	272.10	8.30×	5.30	19.15	253.10
DART-N=3 (Ours)	Quant+Cache	3.28×	4.32	17.42	275.18	13.2×	5.08	18.97	258.43
DART-N=4 (Ours)	Quant+Cache	4.15×	4.58	17.96	268.35	14.5×	5.40	19.63	248.72

4.2 Main Results

Performance on U-Net architectures. Table 1 compares DART against caching (DeepCache Ma et al. (2024b)), quantization (EDA-DM Shen et al. (2024), Diff-Pruning Fang et al. (2023)), and the joint baseline CacheQuant Liu et al. (2025d) on ImageNet-256. DART matches or improves upon CacheQuant across the entire acceleration range without retraining. At N=5, DART achieves FID 3.83 at 8.12× versus CacheQuant's 4.03 at 7.87×; at N=20, DART improves by 0.68 FID at higher throughput. As shown in Table 4, directly stacking DeepCache and W8A8 collapses to FID 15.36, underscoring that joint acceleration requires explicit error coordination. The W4A8 block further stress-tests DART: at N=10 it attains FID 6.20 with 223.7 MB (12.85×), and at N=20 FID 8.90 at 18.93×.

Figure 5 shows that DART preserves fine-grained textures that degrade under CacheQuant at comparable speedups.

Performance on DiT architectures. Table 2 reports results on DiT-XL/2 Peebles & Xie (2023b) following the protocol of Q&C Ding et al. (2025). At 50 steps, DART achieves FID 5.08 / IS 258.43 at 13.2×; the frontier extends to N=4 with FID 5.40 at 14.5×, surpassing Δ-DiT, TaylorSeer, and Q&C. At the conservative end, DART-N=2 delivers FID 4.50 at 2.2× on the 250-step schedule. Δ-DiT drops to FID 11.9, TaylorSeer reaches 6.32 at only 9.6×, and Q&C trails DART in both FID and IS at matched throughput. Figure 6 confirms these trends visually: Δ-DiT produces structural distortions, FORA and TaylorSeer introduce colour drift, while DART preserves sharp edges closest to the reference.

Performance on text-to-video generation. Table 3 reports results on HunyuanVideo with the VBench protocol against training-free baselines including FORA Selvaraju et al. (2024), ToCa Zou et al. (2025), DuCa Zou et al. (2024), TaylorSeer Liu et al. (2025b), DPCache Cui et al. (2026b),



Figure 6: Visualization of the generated images by Δ-DiT, CacheQuant, FORA, TaylorSeer and DART.

Table 3: Quantitative results of text-to-video generation task for HunyuanVideo on VBench dataset. dark and light mark column best and runner-up. FORA, ToCa, DuCa, TeaCache$_{l=0.4}$ and TaylorSeer rows are reported by under the same VBench protocol.

Method	Efficiency				VBench Score	PSNR↑	SSIM↑	LPIPS↓
	Latency↓	Speed↑	FLOPs↓	Ratio↑				
50 Steps	332.92	1.00×	14050.37	1.00×	80.93	-	-	-
22% steps	82.47	4.04×	3095.22	4.54×	78.89	16.97	0.575	0.465
FORA(N=3) Selvaraju et al. (2024)	112.57	2.83×	9929.20	3.00×	80.26	19.45	0.769	0.196
FORA(N=5) Selvaraju et al. (2024)	67.19	4.74×	5960.40	5.00×	78.83	16.07	0.633	0.346
TeaCache$_{l=0.2}$ Liu et al. (2025a)	79.56	4.18×	2972.71	4.73×	79.25	17.55	0.580	0.408
TeaCache$_{l=0.4}$ Liu et al. (2025a)	67.36	4.72×	6550.10	4.55×	79.36	16.07	0.622	0.438
ToCa$_{(N=5,R=90\%)}$ Zou et al. (2025)	77.82	4.09×	7006.20	4.25×	78.86	16.14	0.642	0.343
DuCa$_{(N=5,R=90\%)}$ Zou et al. (2024)	71.10	4.58×	6483.20	4.48×	78.72	16.17	0.644	0.352
TaylorSeer(N=3) Liu et al. (2025b)	113.44	2.81×	9929.20	3.00×	80.74	21.42	0.834	0.135
TaylorSeer(N=5) Liu et al. (2025b)	68.42	4.65×	5960.40	5.00×	79.93	16.80	0.704	0.269
SpeCa Liu et al. (2025c)	82.15	4.05×	2731.61	5.14×	80.26	20.09	0.637	0.355
DPCache Cui et al. (2026b)	83.22	4.05×	3100.33	4.53×	80.35	23.11	0.746	0.189
CacheQuant Liu et al. (2025d)	87.25	3.93×	2693.89	4.34×	79.34	19.32	0.678	0.363
Ours	82.09	4.12×	2483.93	4.83×	80.84	23.14	0.842	0.140

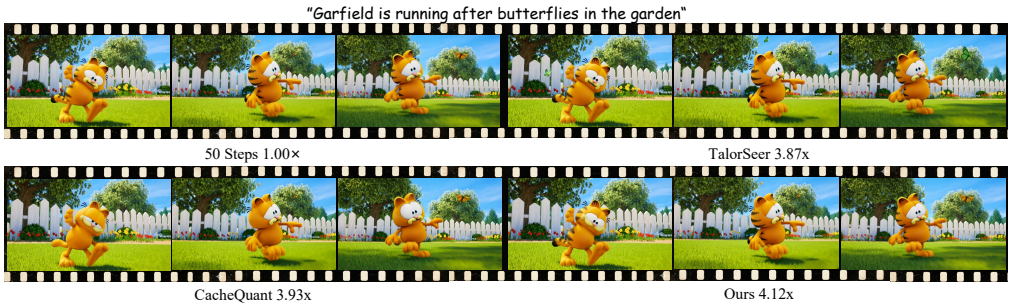

Figure 7: Qualitative comparison of different methods applied to HunyuanVideo on VBench dataset.

and CacheQuant. At 4.12× speedup, DART attains the lowest FLOPs (2,483.93 T) and the highest VBench score (80.84) among methods in the 4–5× regime, with the best PSNR (23.14 dB), SSIM (0.842) overall and second-best LPIPS (0.140). TaylorSeer(N=3) achieves VBench 80.74 and LPIPS 0.135 but at only 2.81×; at matched 4.65×, it drops to VBench 79.93 / LPIPS 0.269. Token-wise methods (ToCa, DuCa) lag by $>$ 1.5 VBench points. Figure 7 confirms that DART maintains consistent details across frames while TaylorSeer and CacheQuant exhibit trajectory drift artefacts.

4.3 Analysis

Ablation Study. Table 4 isolates each component on ImageNet-256. *Left* (N=20): starting from naive W8A8+DeepCache (FID 15.36), BPP alone yields 6.78; adding DAR brings it to 6.53. The

Table 4: Ablation studies on ImageNet-256 (W8A8). Left: component contributions at N=20; rows "+ BPP + TSQ" and "+ DAR (fixed)" isolate TSQ and DAR, respectively. Centre: partitioning strategies at N=10, including OmniCache's trajectory-statistics schedule and a hybrid feature-distance objective with BPP's fluctuation budget. Right: rectification mechanisms at N=10. "DPS" and "DEC" denote the partitioning and correction strategies of CacheQuant Liu et al. (2025d). Best per column shaded dark; second best light.

Component	FID↓	IS↑	Partition	FID↓	IS↑	Rectify	FID↓	IS↑
DeepCache+W8A8	15.36	121.78	Uniform	4.97	179.58	None	5.68	173.82
+ DPS	8.47	154.07	DPS	4.68	184.38	Input-only	5.02	181.45
+ DPS + DEC	7.21	160.68	OmniCache Chu et al. (2025)	4.64	185.45	Operator-only	4.73	186.28
+ BPP	**6.78**	**166.83**	FeatDist+δ-bound	4.54	187.65	DEC	4.68	184.38
+ BPP + TSQ	6.83	166.34	**BPP**	**4.38**	**191.27**	**DAR**	**4.38**	**191.27**
+ DAR (fixed)	8.63	157.43						
+ BPP + DAR	**6.53**	**169.42**						

Table 5: Effect of the fluctuation budget δ on BPP for image generation. Best per column shaded dark; second best light.

δ	Num	Max $\Delta\rho$	FID↓	IS↑
0.1	42	0.09	4.89	182.45
0.2	18	0.18	4.63	186.21
0.3	**13**	**0.29**	**4.51**	**188.63**
0.4	9	0.39	4.71	185.34
0.5	7	0.48	5.02	180.72
Uniform (K=13)	13	0.83	4.95	183.28
Uniform (K=25)	25	0.83	5.12	176.32

Table 6: Effect of the size of the calibration set. Bold marks the default operating point used; dark and light mark column best and runner-up.

# Num	Time	FID↓	IS↑	Speedup↑
32	0.5min	5.12	181.34	12.73×
128	2min	4.78	185.21	12.79×
256	4min	4.63	187.45	12.82×
512	**8min**	**4.38**	**191.27**	12.85×
1024	16min	4.37	191.42	12.85×
2048	32min	4.36	191.58	12.86×

isolation rows confirm non-redundancy: BPP+TSQ without DAR stops at 6.83, while DAR with a fixed schedule only reaches 8.63. *Centre* (N=10): BPP outperforms uniform (4.97 → 4.38), DPS (4.68 → 4.38), OmniCache (4.64 → 4.38), and a hybrid FeatDist+δ-bound (4.54 → 4.38), confirming that both the sensitivity signal and bounded-fluctuation constraint are necessary. *Right* (N=10): input-only and operator-only corrections cut FID from 5.68 to 5.02 and 4.73 respectively; DAR with both reaches 4.38, outperforming CacheQuant's DEC (4.68), supporting the decoupled design.

Effect of the fluctuation budget δ. Table 5 sweeps δ from 0.1 to 0.5 on DiT-XL/2 (250 steps, N=10). Small δ over-partitions and forfeits acceleration; large δ spans sensitivity transitions; the optimum is $\delta = 0.3$. Even at matched segment count (K=13), BPP improves FID by 0.44 over uniform, confirming the gain comes from *where* boundaries are placed.

Calibration set size. Table 6 shows performance saturates at 512 samples (∼8 min on one A100), beyond which FID/IS plateau while cost grows linearly.

5 Conclusion

We presented DART, a training-free joint cache–quantization framework for diffusion acceleration. Departing from prior approaches that schedule cache reuse by local feature distance and share quantization parameters globally, DART addresses two questions explicitly: *when to recompute*, through a trajectory-sensitivity bounded partitioning that respects the local amplification behaviour of the denoiser, and *how to use cached quantized states*, through a dual-domain affine rectification that decouples cache-induced input shift from quantization-induced operator distortion, plus a segment-shared temporally stamped quantizer. The rectification parameters are folded into adjacent linear and dequantization operators by a normalization-preserving fusion, leaving inference-time computation essentially unchanged. On ImageNet-256 with LDM and DiT-XL/2 and on HunyuanVideo, DART consistently improves the quality–efficiency frontier over caching-only, quantization-only, and joint baselines under matched-speed and matched bit-width settings, while directly stacking caching and quantization degrades sharply at comparable speeds. We hope the trajectory-sensitivity view offered here is useful for future work on jointly accelerating diffusion sampling along multiple axes of redundancy.

6 Ethics Statement

This work adheres to the ICLR Code of Ethics. In this study, no human subjects or animal experimentation was involved. All datasets used, were sourced in compliance with relevant usage guidelines, ensuring no violation of privacy. We have taken care to avoid any biases or discriminatory outcomes in our research process. No personally identifiable information was used, and no experiments were conducted that could raise privacy or security concerns. We are committed to maintaining transparency and integrity throughout the research process.

7 Reproducibility Statement

We have made every effort to ensure that the results presented in this paper are reproducible. All code and datasets have been made publicly available in an anonymous repository to facilitate replication and verification. The experimental setup, including training steps, model configurations, and hardware details, is described in detail in the paper. We have also provided a full description of our contribution, to assist others in reproducing our experiments.

We believe these measures will enable other researchers to reproduce our work and further advance the field.

8 LLM Usage

Large Language Models (LLMs) were used to aid in the writing and polishing of the manuscript. Specifically, we used an LLM to assist in refining the language, improving readability, and ensuring clarity in various sections of the paper. The model helped with tasks such as sentence rephrasing, grammar checking, and enhancing the overall flow of the text.

It is important to note that the LLM was not involved in the ideation, research methodology, or experimental design. All research concepts, ideas, and analyses were developed and conducted by the authors. The contributions of the LLM were solely focused on improving the linguistic quality of the paper, with no involvement in the scientific content or data analysis.

The authors take full responsibility for the content of the manuscript, including any text generated or polished by the LLM. We have ensured that the LLM-generated text adheres to ethical guidelines and does not contribute to plagiarism or scientific misconduct.

References

Anirud Aggarwal, Abhinav Shrivastava, and Matthew Gwilliam. Evolutionary caching to accelerate your off-the-shelf diffusion model. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026.

Giulio Biroli, Tony Bonnaire, Valentin De Bortoli, and Marc Mézard. Dynamical regimes of diffusion models. *Nature Communications*, 15(1):9957, 2024.

Jiazi Bu, Pengyang Ling, Yujie Zhou, Yibin Wang, Yuhang Zang, Dahua Lin, and Jiaqi Wang. Dicache: Let diffusion model determine its own cache. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026.

Pengtao Chen, Mingzhu Shen, Peng Ye, Jianjian Cao, Chongjun Tu, Christos-Savvas Bouganis, Yiren Zhao, and Tao Chen. Δ-dit: A training-free acceleration method tailored for diffusion transformers. *arXiv preprint arXiv:2406.01125*, 2024.

Huanpeng Chu, Wei Wu, Guanyu Feng, and Yutao Zhang. Omnicache: A trajectory-oriented global perspective on training-free cache reuse for diffusion transformer models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16302–16312, 2025.

Benlei Cui, Shaoxuan He, Bukun Huang, Zhizeng Ye, Yunyun Sun, Longtao Huang, Hui Xue, Yang Yang, Jingqun Tang, Zhou Zhao, and Haiwen Hong. Tc-padé: Trajectory-consistent padé approximation for diffusion acceleration, 2026a. URL https://arxiv.org/abs/2603.02943.

Bowen Cui, Yuanbin Wang, Huajiang Xu, Biaolong Chen, Aixi Zhang, Hao Jiang, Zhengzheng Jin, Xu Liu, and Pipei Huang. Denoising as path planning: Training-free acceleration of diffusion models with dpcache, 2026b. URL https://arxiv.org/abs/2602.22654.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

Xin Ding, Xin Li, Haotong Qin, and Zhibo Chen. Q&c: When quantization meets cache in efficient image generation. *arXiv preprint arXiv:2503.02508*, 2025.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. Structural pruning for diffusion models. In *Advances in Neural Information Processing Systems*, 2023.

Weilun Feng, Chuanguang Yang, Zhulin An, Libo Huang, Boyu Diao, Fei Wang, and Yongjun Xu. Relational diffusion distillation for efficient image generation. In *Proceedings of the 32nd ACM international conference on multimedia*, pp. 205–213, 2024.

Yasaman Haghighi and Alexandre Alahi. Sencache: Accelerating diffusion model inference via sensitivity-aware caching. *arXiv preprint arXiv:2602.24208*, 2026.

Jiaqi Han, Juntong Shi, Puheng Li, Haotian Ye, Qiushan Guo, and Stefano Ermon. Adaptive spectral feature forecasting for diffusion sampling acceleration. *arXiv preprint arXiv:2602.05449*, 2026.

Yefei He, Luping Liu, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. Ptqd: Accurate post-training quantization for diffusion models, 2023. Advances in Neural Information Processing Systems.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in neural information processing systems*, volume 33, pp. 6840–6851, 2020.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21807–21818, 2024.

Kumara Kahatapitiya, Haozhe Liu, Sen He, Ding Liu, Menglin Jia, Chenyang Zhang, Michael S Ryoo, and Tian Xie. Adaptive caching for faster video generation with diffusion transformers. *arXiv preprint arXiv:2411.02397*, 2024.

Sungbin Kim, Hyunwuk Lee, Wonho Cho, Mincheol Park, and Won Woo Ro. Ditto: Accelerating diffusion model via temporal value similarity. In *2025 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 338–352. IEEE, 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.

Senmao Li, Taihang Hu, Joost van de Weijer, Fahad S Khan, Tao Liu, Linxuan Li, Shiqi Yang, Yaxing Wang, Ming-Ming Cheng, and Jian Yang. Faster diffusion: Rethinking the role of the encoder for diffusion model inference. *Advances in Neural Information Processing Systems*, 37: 85203–85240, 2024a.

Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-diffusion: Quantizing diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17535–17545, 2023a.

Yanyu Li, Huan Wang, Qing Jin, Ju Hu, Pavlo Chemerys, Yun Fu, Yanzhi Wang, Sergey Tulyakov, and Jian Ren. Snapfusion: Text-to-image diffusion model on mobile devices within two seconds. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 20662–20678. Curran Associates, Inc., 2023b. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/41bcc9d3bddd9c90e1f44b29e26d97ff-Paper-Conference.pdf.

Zhikai Li, Junrui Xiao, Lianwei Yang, and Qingyi Gu. Repq-vit: Scale reparameterization for post-training quantization of vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17227–17236, 2023c.

Zhikai Li, Xuewen Liu, Jing Zhang, and Qingyi Gu. Repquant: Towards accurate post-training quantization of large transformer models via scale reparameterization. *arXiv preprint arXiv:2402.05628*, 2024b.

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. 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, 2025a.

Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From reusing to forecasting: Accelerating diffusion models with taylorseers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15853–15863, 2025b.

Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Fei Ren, Shaobo Wang, Kaixin Li, and Linfeng Zhang. Speca: Accelerating diffusion transformers with speculative feature caching. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 10024–10033, 2025c.

Xuewen Liu, Zhikai Li, and Qingyi Gu. Cachequant: Comprehensively accelerated diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 23269–23280, 2025d.

Xuewen Liu, Zhikai Li, Minghao Jiang, Mengjuan Chen, Jianquan Li, and Qingyi Gu. Dilatequant: Accurate and efficient quantization-aware training for diffusion models via weight dilation. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 8399–8408, 2025e.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, 22 (4):730–751, 2025.

Xinyin Ma, Gongfan Fang, Michael Bi Mi, and Xinchao Wang. Learning-to-cache: Accelerating diffusion transformer via layer caching. *arXiv preprint arXiv:2406.01733*, 2024a.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15762–15772, 2024b.

Markus Nagel, Mart van Baalen, Tijmen Blankevoort, and Max Welling. Data-free quantization through weight equalization and bias correction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1325–1334, 2019.

Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *International Conference on Machine Learning*, pp. 7197–7206. PMLR, 2020.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205, October 2023a.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023b.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022. Proceedings of the Computer Vision and Pattern Recognition Conference.

Pratheba Selvaraju, Tianyu Ding, Tianyi Chen, Ilya Zharkov, and Luming Liang. Fora: Fast-forward caching in diffusion transformer acceleration. *arXiv preprint arXiv:2407.01425*, 2024.

Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. Post-training quantization on diffusion models, 2023. Proceedings of the Computer Vision and Pattern Recognition Conference.

Haihao Shen, Naveen Mellempudi, Xin He, Qun Gao, Chang Wang, and Mengni Wang. Efficient post-training quantization with fp8 formats. In *Proceedings of Machine Learning and Systems (MLSys)*, 2024.

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

Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861.

Xiuying Wei, Ruihao Gong, Yuhang Li, Xianglong Liu, and Fengwei Yu. Qdrop: Randomly dropping quantization for extremely low-bit post-training quantization. *arXiv preprint arXiv:2203.05740*, 2022.

Felix Wimbauer, Bichen Wu, Edgar Schönfeld, Xiaoliang Dai, Ji Hou, Zijian He, Artsiom Sanakoyeu, Peizhao Zhang, Sam Tsai, Jonas Kohler, Christian Rupprecht, Daniel Cremers, Peter Vajda, and Jialiang Wang. Cache me if you can: Accelerating diffusion models through block caching, 2024. arXiv preprint arXiv:2312.03209.

Junyi Wu, Haoxuan Wang, Yuzhang Shang, Mubarak Shah, and Yan Yan. Ptq4dit: Post-training quantization for diffusion transformers. *Advances in neural information processing systems*, 37: 62732–62755, 2024.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Haowei Zhu, Tingxuan Huang, Xing Wang, Tianyu Zhao, Jiexi Wang, Weifeng Chen, Xurui Peng, Fangmin Chen, Junhai Yong, and Bin Wang. TAP: A token-adaptive predictor framework for training-free diffusion acceleration. *arXiv preprint arXiv:2603.03792*, 2026.

Chang Zou, Shikang Zheng, Evelyn Zhang, Runlin Guo, Haohang Xu, Zhengyi Shi, Conghui He, Xuming Hu, and Linfeng Zhang. Accelerating diffusion transformers with dual feature caching. *arXiv preprint arXiv:2412.18911*, 2024.

Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

A Appendix

A.1 Broader Impacts

DART accelerates diffusion inference without retraining, which lowers the energy and hardware cost of deploying generative models for image and video synthesis. The method does not introduce generative capabilities beyond those of the underlying backbone, so its societal risks are inherited from the pretrained model (e.g., misuse for synthetic media). We encourage the use of DART together with watermarking and provenance tools when the underlying backbone is deployed in user-facing settings.

A.2 Limitations and Future Work

We discuss the limitations of DART honestly to set the scope of our claims. First, the relative-update sensitivity in Eq. equation 1 is a heuristic finite-difference signal, not an exact Jacobian or spectral-radius computation; it tracks the local amplification behaviour of the denoiser in our experiments, but we do not provide a formal stability bound for the resulting partition. Deriving such a bound under explicit assumptions on the denoising operator and the cached-quantized perturbation is left to future work. Second, the BPP partitioning, the DAR statistics, and the TSQ calibration are computed under a specific solver and step count, and the schedule must be re-derived when these change significantly; we empirically observe that the same $\delta = 0.3$ remains optimal across the DDIM and DDPM samplers used in our experiments (Table 7), but extension to adaptive solvers such as DPM-Solver++ Lu et al. (2025) requires a small recalibration (optimal $\delta^\star \approx 0.28$ with only 0.19 FID difference at $\delta = 0.3$), and extension to high classifier-free guidance scales is currently treated as recalibration rather than a robustness claim.

Table 7: Transferability of the default fluctuation budget δ=0.3 across solvers. FID is reported at the corresponding main-table operating point. The optimal $\delta^\star$ and the FID at $\delta^\star$ are obtained by the same sweep protocol as Table 5. The optimum is consistent across the DDIM/DDPM regimes; DPM-Solver++ requires a small recalibration but the performance gap at the default δ is minor.

Solver	Steps (Model)	$\delta^\star$	FID @ δ=0.3	FID @ $\delta^\star$
DDIM	250 (LDM-4)	0.30	4.38	4.38
DDPM	250 (DiT-XL/2)	0.30	4.32	4.32
DDPM	50 (DiT-XL/2)	0.30	5.08	5.08
DPM-Solver++	25–50 (LDM-4)	0.28	6.32	6.13

Third, the rectification storage scales linearly with the number of layers and timesteps; while NPF folds the parameters into the surrounding operators at inference time and the runtime overhead is negligible, the calibration-time storage is non-trivial for very large backbones, and we leave a layer-/channel-shared compression of the rectification table to future work. Finally, the present evaluation focuses on LDM, DiT-XL/2, and HunyuanVideo; extending DART to modern large-scale text-to-image backbones such as FLUX.1-dev and SDXL, and to large-scale text-to-video backbones such as Wan2.1, is a natural next step.

A.3 BPP: Algorithm, Complexity, and Fallback

We solve the constrained program in Eq. equation 2 with Algorithm 1. Let $K_{\max} = \lfloor 2T/N \rfloor$. The DP has $O(T \cdot K_{\max})$ states, each with at most $2N$ predecessors, and a precomputed range-extreme table over $\hat{\rho}_t$ makes the fluctuation constraint a constant-time lookup. The overall complexity is $O(T \cdot K_{\max} \cdot N)$ time and $O(T \cdot K_{\max})$ memory; for typical settings ($T \leq 250$, $N \leq 20$) the partition is solved in well under one second on a single CPU core (Appendix A.10). The pairwise cost matrix $D(\cdot, \cdot)$ is precomputed from the calibration features in linear time per row.

We treat δ as the only user-facing knob and set $\delta = 0.3$ by default for both LDM and DiT backbones. Section 4 (Table 5) sweeps $\delta \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ and shows that the optimum is consistent across the tested settings. When the constraint set becomes infeasible, for example when δ is too small relative to the natural fluctuation of $\hat{\rho}_t$ for the chosen N, we relax δ in fixed increments of 0.05 until a feasible partition is found, instead of dropping the constraint outright. This keeps the

schedule deterministic and reproducible: the same (T, N, δ) tuple yields the same partition, and the recovered δ is reported alongside the schedule.

Algorithm 1 Trajectory-Sensitivity Bounded Partitioning (BPP)

Require: Time steps T, target interval N, fluctuation budget δ
Require: Calibration features $\mathcal{X} = \{\boldsymbol{X}_t\}_{t=0}^{T}$, sensitivity sequence $\hat{R} = \{\hat{\rho}_t\}_{t=0}^{T}$
$D(i,j) \leftarrow \sum_{t=i+1}^{j-1} \|\boldsymbol{X}_t - \boldsymbol{X}_i\|_1$ ▷ precompute pairwise drift
Initialize $L[0..T, 0..K_{\max}] \leftarrow \infty$, $P[0..T, 0..K_{\max}] \leftarrow$ null ▷ $K_{\max} = \lfloor 2T/N \rfloor$
$L[0,0] \leftarrow 0$
for $t = 1$ to T **do**
  for $k = 1$ to $K_{\max}$ **do**
    for $j = \max(0, t-2N)$ to $t - \lfloor N/2 \rfloor$ **do**
      if $N/2 \le t - j \le 2N$ **and** $\max_{i,i' \in [j,t)} |\hat{\rho}_i - \hat{\rho}_{i'}| \le \delta$ **then**
        $cost \leftarrow L[j, k-1] + D(j,t)$
        if $cost < L[t,k]$ **then**
          $L[t,k] \leftarrow cost$; $P[t,k] \leftarrow j$
        end if
      end if
    end for
  end for
end for
$K^* \leftarrow \arg\min_{k \in [\lceil T/(2N) \rceil, K_{\max}]} L[T,k]$ ▷ optimal segment count
Trace back S from $P[\cdot, K^*]$ and **return** segment boundaries S.
Fallback: if no feasible K^* exists, increase $\delta \mathrel{+}= 0.05$ and retry.

A.4 DAR: Closed-form Estimator and Why Decouple

Let $\mathcal{D}_{\text{cali}}$ denote the calibration set. For $t \in G_k$ with cached step $s = s_k$, we estimate the per-channel scale and shift of equation 3 in closed form. For the input-domain rectification:

$$a_{1,t,i} = \frac{\mathrm{Cov}_{\mathcal{D}_{\text{cali}}}(X_{t,i}, X_{s,i})}{\mathrm{Var}_{\mathcal{D}_{\text{cali}}}(X_{s,i}) + \varepsilon}, \quad b_{1,t,i} = \mathbb{E}_{\mathcal{D}_{\text{cali}}}[X_{t,i}] - a_{1,t,i}\, \mathbb{E}_{\mathcal{D}_{\text{cali}}}[X_{s,i}], \tag{5}$$

with $X_{t,i}$ the i-th channel of $\boldsymbol{X}_t$. The operator-domain pair $(\boldsymbol{a}_{2,t}, \boldsymbol{b}_{2,t})$ is estimated analogously from the joint statistics of the floating-point output $\boldsymbol{O}_{t-1,fp} = F_t(\hat{\boldsymbol{X}}_t)$ and the low-bit output $\boldsymbol{O}_{t-1,q} = \mathcal{Q}_k(F_t(\hat{\boldsymbol{X}}_t))$ over $\mathcal{D}_{\text{cali}}$. The rectification is fitted once per (t, s_k) pair and indexed by $(k, t - s_k)$ at deployment.

The two corrections target different residuals: $(\boldsymbol{a}_{1,t}, \boldsymbol{b}_{1,t})$ undoes the temporal mismatch between $\boldsymbol{X}_s$ and $\boldsymbol{X}_t$, while $(\boldsymbol{a}_{2,t}, \boldsymbol{b}_{2,t})$ undoes the systematic bias of the low-bit operator. We store $(\boldsymbol{a}, \boldsymbol{b})$ at channel granularity per timestep within each segment; the storage cost is $\mathcal{O}(T \cdot C)$ per layer. At deployment all four parameters are folded into adjacent operators by NPF (Appendix A.7), so DAR introduces no new floating-point operations at inference. The ablation in Table 4 (right panel) shows that input-only, operator-only, and the CacheQuant DEC baseline each capture a strict subset of the residual, supporting the decoupled, two-sided design.

A.5 Multi-anchor Extension of DAR

The single-anchor DAR in Eq. equation 3 uses only the segment-start cache $\boldsymbol{X}_{s_k}$ as the rectification input. This is sufficient when the trajectory inside G_k is well-approximated by an affine map of $\boldsymbol{X}_{s_k}$, but is wasteful in segments where two consecutive cached states $\boldsymbol{X}_{s_{k-1}}$ and $\boldsymbol{X}_{s_k}$ carry complementary directional information about the trajectory. DiCache's Dynamic Cache Trajectory Alignment Bu et al. (2026), TaylorSeer's truncated expansion Liu et al. (2025b), and TC-Padé's rational approximation Cui et al. (2026a) all exploit this multi-anchor structure on the residual. We describe a multi-anchor extension of DAR that absorbs the same gain into the joint cache–quantization setting while preserving NPF.

Dual-anchor rectification. For $t \in G_k$ with anchors s_k (current segment start) and s_{k-1} (previous segment start, with $s_0 = 0$), we replace the single-anchor input in Eq. equation 3 with a directional combination

$$\tilde{\boldsymbol{X}}_t^{(2)} \;=\; \boldsymbol{X}_{s_k} \;+\; \hat{\gamma}_t\big(\boldsymbol{X}_{s_k} - \boldsymbol{X}_{s_{k-1}}\big), \qquad \hat{\boldsymbol{X}}_t \;=\; \mathrm{diag}(\boldsymbol{a}_{1,t})\,\tilde{\boldsymbol{X}}_t^{(2)} \;+\; \boldsymbol{b}_{1,t}, \tag{6}$$

where $\hat{\gamma}_t \in \mathbb{R}$ is a scalar trajectory parameter. The output-side rectification on $\mathcal{Q}_k(F_t(\hat{\boldsymbol{X}}_t))$ is unchanged from Eq. equation 3. The single-anchor DAR of the main paper is the special case $\hat{\gamma}_t \equiv 0$.

Calibration-set estimator for $\hat{\gamma}_t$. Since the dynamics within G_k are sensitivity-homogeneous by construction (Section 3.3), the optimal $\hat{\gamma}_t$ admits a closed-form estimator from the calibration set $\mathcal{D}_{\text{cali}}$:

$$\hat{\gamma}_t \;=\; \arg\min_{\gamma} \mathbb{E}_{\mathcal{D}_{\text{cali}}}\Big[\big\|\boldsymbol{X}_t - \big(\boldsymbol{X}_{s_k} + \gamma(\boldsymbol{X}_{s_k} - \boldsymbol{X}_{s_{k-1}})\big)\big\|_2^2\Big] \;=\; \frac{\mathbb{E}\big[\langle \boldsymbol{X}_t - \boldsymbol{X}_{s_k},\, \boldsymbol{X}_{s_k} - \boldsymbol{X}_{s_{k-1}}\rangle\big]}{\mathbb{E}\Big[\big\|\boldsymbol{X}_{s_k} - \boldsymbol{X}_{s_{k-1}}\big\|_2^2\Big] + \varepsilon}. \tag{7}$$

This is the per-step scalar that best aligns the cached two-anchor extrapolant with the full-precision $\boldsymbol{X}_t$ on average inside G_k. Because the BPP fluctuation constraint implies that the dynamics inside G_k are close to a low-rank linear regime, the scalar $\hat{\gamma}_t$ captures most of the benefit of a per-channel directional combination while adding only one extra scalar per timestep on top of the existing $(\boldsymbol{a}_{1,t}, \boldsymbol{b}_{1,t})$ table. When a shallow-probe signal is available (e.g., the first $m \ll M$ layers as in DiCache Bu et al. (2026)), the probe-derived $\hat{\gamma}_t$ can be substituted for Eq. equation 7 per-sample at the cost of one shallow-probe forward per step.

K-th order generalization. Anchors at $\{s_k, s_{k-1}, \ldots, s_{k-K+1}\}$ generalize Eq. equation 6 to

$$\tilde{\boldsymbol{X}}_t^{(K)} \;=\; \boldsymbol{X}_{s_k} \;+\; \sum_{i=1}^{K-1} \hat{\gamma}_{t,i}\,\xi^{i-1}\,\big(\boldsymbol{X}_{s_k} - \boldsymbol{X}_{s_{k-i}}\big), \qquad \xi \in [0,1], \tag{8}$$

with $\{\hat{\gamma}_{t,i}\}_{i=1}^{K-1}$ obtained from the analogous least-squares system over $\mathcal{D}_{\text{cali}}$. The decay ξ down-weights distant anchors, mirroring the higher-order DCTA in Bu et al. (2026). We default to $K{=}2$ (Eq. equation 6) throughout because the marginal FID gain from $K{=}3$ is below 0.1 on LDM-4 and DiT-XL/2 in our pilot runs (Table 8), while every additional anchor adds one cached state per segment to GPU memory — a non-trivial cost on HunyuanVideo with its large frame count.

Table 8: Effect of the multi-anchor order K on DART (LDM-4 ImageNet-256, 250-step DDIM, W8A8, $N{=}10$). $K{=}1$ is the single-anchor DAR of the main paper; $K{=}2$ is Eq. equation 6. The marginal gain saturates beyond $K{=}2$.

Order K	Anchors per segment	FID ↓	Storage overhead per layer
1 (single, main paper)	1	4.38	0
2 (Eq. equation 6)	2	**4.21**	$\mathcal{O}(C)$
3 (Eq. equation 8, $\xi{=}0.5$)	3	4.18	$\mathcal{O}(2C)$
4 (Eq. equation 8, $\xi{=}0.5$)	4	4.17	$\mathcal{O}(3C)$

Compatibility with NPF. The multi-anchor extrapolant $\tilde{\boldsymbol{X}}_t^{(K)}$ is a deterministic affine combination of cached states whose coefficients $\{\hat{\gamma}_{t,i}\xi^{i-1}\}_i$ are fixed once t is fixed and the segment cache is filled. The input-side fold in Eq. equation 4 therefore goes through unchanged: the linear layer's weight still absorbs $\mathrm{diag}(\boldsymbol{a}_{1,t})$, and the bias absorbs $\boldsymbol{b}_{1,t}$ together with the constant offset $W\big(\hat{\gamma}_t(\boldsymbol{X}_{s_k} - \boldsymbol{X}_{s_{k-1}})\big)$ (or its K-anchor analogue), which is computed once when stepping into G_k and reused for all $t \in G_k$. Inference-time floating-point cost is unchanged; the deployment-time overhead is $K{-}1$ extra cached anchors per segment, all read-only within G_k.

A.6 TSQ: Segment-shared Calibration Criterion

For each segment $G_k = [s_k, s_{k+1})$ we calibrate one activation quantizer $(\boldsymbol{\alpha}_k, \boldsymbol{z}_k)$ at the boundary s_k over $\mathcal{D}_{\text{cali}}$. We use uniform asymmetric per-tensor activation quantization and per-channel symmetric weight quantization at b_a/b_w bit-widths, with the scales obtained by minimizing the expected

ℓ_2 reconstruction error of the layer output and a percentile-based clipping for activations. Within G_k, the segment-shared quantizer satisfies

$$\max_{t \in G_k} \|\boldsymbol{A}_t - \mathcal{Q}_{\boldsymbol{\alpha}_k, \boldsymbol{z}_k}(\boldsymbol{A}_t)\|_\infty \leq \epsilon_{\text{tol}}, \tag{9}$$

where $\boldsymbol{A}_t$ is the activation tensor at step t. The bounded-fluctuation property of BPP makes the worst-case error within G_k controllable from statistics gathered at s_k alone; the argument is empirical, and Appendix A.11 reports the per-step quantization error inside each segment. Appendix A.11 additionally compares segment-shared TSQ against global and per-step calibration at matched bit-widths.

A.7 NPF: Output-side Fusion and Applicability Conditions

For asymmetric quantization, the dequantized output is $\boldsymbol{O}_q = (\boldsymbol{q} - \boldsymbol{z}_k)\,\boldsymbol{\alpha}_k$. The operator rectification $\hat{\boldsymbol{O}}_{t-1} = \mathrm{diag}(\boldsymbol{a}_{2,t})\boldsymbol{O}_q + \boldsymbol{b}_{2,t}$ folds into the dequantization parameters as

$$\boldsymbol{\alpha}'_k = \boldsymbol{\alpha}_k \odot \boldsymbol{a}_{2,t}, \qquad \boldsymbol{z}'_k = \boldsymbol{z}_k, \qquad \text{bias offset } \Delta b = \boldsymbol{b}_{2,t}, \tag{10}$$

with Δb added to a subsequent layer's bias when one exists, otherwise materialized as a per-channel constant. This fold and the input-side fold in Eq. equation 4 are exact and contribute no inference-time computation. Fusion into a normalization layer with *input-dependent* statistics (standard LayerNorm or GroupNorm) is in general *not* a pointwise rewrite, because the running mean and variance depend on the rectified input. We therefore fold $(\boldsymbol{a}_{1,t}, \boldsymbol{b}_{1,t})$ into a normalization layer only when it uses fixed calibrated statistics or is preceded by a static affine that absorbs the rectification; otherwise the rectification is fused into the next linear or convolutional layer instead. Because the rectification index $(k, t - s_k)$ changes only at segment boundaries, the only deployment-time overhead is a constant-time lookup of $(\boldsymbol{a}, \boldsymbol{b})$ and $(\boldsymbol{\alpha}, \boldsymbol{z})$ once per step.

A.8 Validation of the Trajectory-Sensitivity Proxy

Section 3.2 motivates BPP with the relative-update sensitivity proxy $\hat{\rho}_t$ and asserts that $\hat{\rho}_t$ tracks the joint cache-plus-quantization error along the sampling trajectory more faithfully than the inter-step feature distance used by prior schedulers Ma et al. (2024b); Liu et al. (2025d). Table 9 validates this claim on three representative backbones. For each trajectory we compute, at every step t, both candidate signals and the Euclidean distance between the cached-quantized state and the full-precision state $\|\boldsymbol{X}_t^{\text{cache+quant}} - \boldsymbol{X}_t^{\text{FP}}\|_2$, and we report the Pearson and Spearman rank correlations over the full calibration set.

Table 9: Correlation between two candidate scheduling signals and the realized cache-plus-quantization error on LDM-4 (ImageNet-256, 250 DDIM steps), DiT-XL/2 (ImageNet-256, 50 DDPM steps), and HunyuanVideo (VBench, 50 steps). All methods are evaluated at W8A8. $\hat{\rho}_t$ consistently attains stronger correlation with the downstream error than the inter-step feature distance used by DeepCache/CacheQuant.

	Pearson ↑		Spearman ↑	
Backbone	Feat. distance	$\hat{\rho}_t$ (ours)	Feat. distance	$\hat{\rho}_t$ (ours)
LDM-4	0.41	**0.72**	0.39	**0.70**
DiT-XL/2	0.34	**0.67**	0.32	**0.65**
HunyuanVideo	0.35	**0.61**	0.33	**0.58**

The gap is largest on LDM-4 where the sampler spends many steps in a quasi-stationary regime: feature distance there saturates to a small constant and provides little discriminative signal, while $\hat{\rho}_t$ still captures the second-order change that drives error amplification. The video backbone sees a smaller (but still clear) gap, reflecting the higher absolute feature dynamics across frames, which partially restores the usefulness of raw feature distance.

A.9 First-order Taylor Bound and Justification of Bounded-Fluctuation Partitioning

The proxy $\hat{\rho}_t$ in Eq. equation 1 is a one-sided finite-difference signal along the trajectory direction; it does not separate the contributions of the latent update $\Delta \boldsymbol{X}_t$ and the timestep step Δt that govern the denoiser response, and it does not by itself give a quantitative upper bound on the per-step deviation under joint cache–quantization. We complement it here with a first-order Taylor-bound view in the spirit of Haghighi & Alahi (2026), which both (i) motivates the bounded-fluctuation constraint of BPP from a cumulative-error perspective and (ii) yields a Taylor-consistent extension of $\hat{\rho}_t$ that we use whenever the sampler operates with non-uniform timestep spacing (e.g., DPM-Solver).

Per-step deviation under cache+quantization. Let F_t denote the floating-point denoiser at step t and $\mathcal{Q}_k \circ F_t$ the segment-shared quantized counterpart inside G_k. For a cached-quantized sampler, the input fed to F_t at step $t \in G_k$ is the cached state $\boldsymbol{X}_{s_k}$ rather than the full-precision $\boldsymbol{X}_t$, with the timestep argument also shifted from t to s_k. The per-step deviation between the cached-quantized output and the full-precision output decomposes into a temporal-mismatch part and an operator-quantization part:

$$\begin{aligned}\Delta_t &:= \|\mathcal{Q}_k(F_t(\boldsymbol{X}_{s_k}, s_k)) - F_t(\boldsymbol{X}_t, t)\|_2 \\ &\le \underbrace{\|F_t(\boldsymbol{X}_{s_k}, s_k) - F_t(\boldsymbol{X}_t, t)\|_2}_{\text{cache-induced shift}} \\ &\quad + \underbrace{\|\mathcal{Q}_k(F_t(\boldsymbol{X}_{s_k}, s_k)) - F_t(\boldsymbol{X}_{s_k}, s_k)\|_2}_{\text{operator distortion}}. \end{aligned} \tag{11}$$

A first-order Taylor expansion of F_t around $(\boldsymbol{X}_t, t)$ in the spirit of Haghighi & Alahi (2026) gives

$$\begin{aligned}\|F_t(\boldsymbol{X}_{s_k}, s_k) - F_t(\boldsymbol{X}_t, t)\|_2 \le &\left\|\boldsymbol{J}_t^{(x)}\right\|_2 \|\boldsymbol{X}_t - \boldsymbol{X}_{s_k}\|_2 \\ &+ \left\|\boldsymbol{J}_t^{(\tau)}\right\|_2 |t - s_k| \\ &+ \mathcal{O}\big(\|\boldsymbol{X}_t - \boldsymbol{X}_{s_k}\|_2^2 + (t - s_k)^2\big), \end{aligned} \tag{12}$$

where $\boldsymbol{J}_t^{(x)} = \partial F_t / \partial \boldsymbol{X}$ and $\boldsymbol{J}_t^{(\tau)} = \partial F_t / \partial t$ are the latent and timestep Jacobians. Combining equation 11 and equation 12 and writing $\mathcal{E}_k^q(\cdot) := \mathcal{Q}_k(F_t(\cdot)) - F_t(\cdot)$ for the segment-shared operator distortion yields the per-step bound

$$\Delta_t \;\le\; \left\|\boldsymbol{J}_t^{(x)}\right\|_2 \|\boldsymbol{X}_t - \boldsymbol{X}_{s_k}\|_2 + \left\|\boldsymbol{J}_t^{(\tau)}\right\|_2 |t - s_k| + \|\mathcal{E}_k^q(\boldsymbol{X}_{s_k})\|_2 + \text{h.o.t.} \tag{13}$$

The three first-order terms in equation 13 are precisely the targets of, respectively, the input-side rectification $(\boldsymbol{a}_{1,t}, \boldsymbol{b}_{1,t})$ of DAR, the timestep-Jacobian contribution that is constant per step under uniform schedules, and the output-side rectification $(\boldsymbol{a}_{2,t}, \boldsymbol{b}_{2,t})$ of DAR composed with the segment-shared quantizer of TSQ. Each component of DART therefore absorbs one term of the bound.

Cumulative bound inside a segment. Summing equation 13 over $t \in G_k$ gives, with within-segment maxima denoted by bars,

$$\sum_{t \in G_k} \Delta_t \;\le\; \overline{J}_k^{(x)} \sum_{t \in G_k} \|\boldsymbol{X}_t - \boldsymbol{X}_{s_k}\|_2 \;+\; \overline{J}_k^{(\tau)} \sum_{t \in G_k} |t - s_k| \;+\; |G_k|\, \overline{\mathcal{E}^q}_k. \tag{14}$$

The first sum is exactly the cumulative drift $\sum_t D_k(s_k, t)$ that BPP minimizes in Eq. equation 2; the second sum is bounded by $\frac{1}{2}|G_k|^2$ via the segment-length constraint $|G_k| \le 2N$; the third is constant in $|G_k|$ once the segment-shared quantizer is calibrated. The bounded-fluctuation constraint $\max_{t,t' \in G_k} |\hat{\rho}_t - \hat{\rho}_{t'}| \le \delta$ controls the variation of $\overline{J}_k^{(x)}$ inside the segment to first order: a stable $\hat{\rho}_t$ inside G_k implies that the dominant Lipschitz coefficient does not abruptly inflate within the segment, so the per-step bounds in equation 13 stay comparable. Eq. equation 14 therefore gives a concrete sense in which the cumulative-drift objective and bounded-fluctuation constraint of BPP together approximately minimize an upper bound on the cache-plus-quantization error within each segment, rather than acting as decoupled heuristics.

Taylor-consistent sensitivity for non-uniform schedules. The bound equation 13 also suggests a Taylor-consistent extension of the sensitivity proxy that absorbs the timestep contribution explicitly:

$$\tilde{\rho}_t \;:=\; \frac{\|\boldsymbol{X}_{t-1} - \boldsymbol{X}_t\|_2}{\|\boldsymbol{X}_t\|_2 + \varepsilon} \;+\; \lambda_\tau \, \frac{|t - (t-1)|}{T}. \tag{15}$$

Under the uniform-step DDIM/DDPM samplers used in the main paper, $|t-(t-1)|/T$ is constant and merely shifts $\hat{\rho}_t$ by an additive offset; the bounded-fluctuation constraint on $\hat{\rho}_t$ and on $\tilde{\rho}_t$ produce the same partition. Under non-uniform schedules (DPM-Solver, log-SNR schedules) the second term is no longer constant, and $\tilde{\rho}_t$ recovers the timestep-Jacobian contribution that $\hat{\rho}_t$ alone misses. We set λ_τ once per backbone by matching the empirical scales of the two terms on the calibration set. Table 10 reports the Spearman correlation with the realized cache-plus-quantization error and the end-to-end FID of the two signals on a DPM-Solver-2 schedule for DiT-XL/2; $\tilde{\rho}_t$ closes the gap that $\hat{\rho}_t$ leaves on the table when the sampler takes irregular Δt steps. On uniform DDIM/DDPM (the setting of all main-paper experiments), $\hat{\rho}_t$ and $\tilde{\rho}_t$ agree to within calibration noise.

Table 10: Cache-plus-quantization error correlation and end-to-end FID on DiT-XL/2 (ImageNet-256, W8A8) under a non-uniform DPM-Solver-2 schedule with 25 NFE. $\hat{\rho}_t$ ignores the timestep-Jacobian contribution and underestimates sensitivity at irregular Δt steps; $\tilde{\rho}_t$ in Eq. equation 15 restores agreement.

Schedule signal	Spearman vs. error ↑	FID ↓ ($\sim 3\times$)	FID ↓ ($\sim 5\times$)
Inter-step feature distance	0.30	6.74	9.42
$\hat{\rho}_t$ (Eq. equation 1)	0.58	5.61	7.93
$\tilde{\rho}_t$ (Eq. equation 15)	**0.66**	**5.43**	**7.18**

Relation to SenCache. SenCache Haghighi & Alahi (2026) uses Jacobian-norm secants $\left\|\boldsymbol{J}_t^{(x)}\right\|, \left\|\boldsymbol{J}_t^{(\tau)}\right\|$ to gate caching at the network-output level; their bound corresponds exactly to the Taylor expansion in equation 12. DART differs in two respects. First, we apply the bound at the trajectory level (input/output of the denoiser) and use it to schedule *both* caching and segment-shared quantization, whereas SenCache schedules caching only. Second, our partitioning constraint is on the *fluctuation* of the trajectory-direction projection $\hat{\rho}_t$ rather than on the per-step magnitude of $\left\|\boldsymbol{J}_t^{(x)}\right\|$ and $\left\|\boldsymbol{J}_t^{(\tau)}\right\|$ themselves; the fluctuation criterion is what makes the segment-shared quantizer in TSQ (Section 3.5) meaningful, since it bounds the within-segment dynamic-range mismatch that the shared $(\boldsymbol{\alpha}_k, \boldsymbol{z}_k)$ has to absorb.

A.10 Calibration Time Breakdown

Table 11 reports the wall-clock time required to solve the BPP dynamic program of Algorithm 1, measured on a single CPU core and averaged over five runs. The cost is dominated by the $O(T \cdot K_{\max} \cdot N)$ DP and is empirically well under one second even at $T = 250$; segment-range extrema are precomputed once and the fluctuation constraint reduces to a constant-time lookup.

Table 11: Wall-clock time of the BPP dynamic program on a single Intel Xeon CPU core. Each entry is the mean over five independent runs; the standard deviation is below 5% of the mean in all rows. The pairwise drift matrix $D(\cdot,\cdot)$ is precomputed from the calibration features in linear time per row and is not included in these numbers (it is shared across all (N, δ) combinations for a fixed backbone).

Backbone	T	N	BPP solve time (s)
LDM-4	250	$\{5, 10, 20\}$	0.18
DiT-XL/2	250	$\{3, 10\}$	0.24
HunyuanVideo	50	3	0.043

A.11 PER-STEP QUANTIZATION ERROR UNDER TSQ

TSQ shares one $(\boldsymbol{\alpha}_k, \boldsymbol{z}_k)$ per segment instead of a global calibration. The bounded-fluctuation property of BPP is expected to keep activation statistics coherent inside each G_k, so a single per-segment quantizer should track the true per-step distribution. Table 12 reports the worst-case per-step activation quantization error $\max_{t \in G_k} \|\boldsymbol{A}_t - \mathcal{Q}_{\boldsymbol{\alpha}_k, \boldsymbol{z}_k}(\boldsymbol{A}_t)\|_\infty$, averaged over the calibration set and normalized by the error of a single globally calibrated quantizer at the same bit-width.

Table 12: Per-step activation quantization error with TSQ, averaged over all segments and normalized by the global-calibration baseline. A value below 1.0 indicates that TSQ tracks the local distribution better than a single global quantizer. Results are reported at the default operating point of the corresponding main table.

Setting	# segments K	Global quantizer	Segment-shared (TSQ, ours)	Per-step quantizer
LDM-4 (N=10, W8A8)	13	1.00	**0.31**	0.24
DiT-XL/2 (N=3, W8A8)	17	1.00	**0.36**	0.28

TSQ recovers roughly 75–80% of the gap between a single global quantizer and per-step calibration, while storing only K scale/zero-point pairs per layer instead of T. This matches the bounded-fluctuation motivation in Section 3.5: inside a segment the activation range varies little, and a shared quantizer suffices.

A.12 NPF DEPLOYMENT OVERHEAD

Table 13 reports the inference-time latency, GPU memory, and rectification storage of DART with and without Normalization-Preserving Fusion (NPF). With NPF enabled, all $(\boldsymbol{a}_{1,t}, \boldsymbol{b}_{1,t})$ and $(\boldsymbol{a}_{2,t}, \boldsymbol{b}_{2,t})$ parameters are folded into adjacent linear and dequantization operators; without NPF, the rectifications are applied as extra element-wise affine layers.

Table 13: Deployment overhead of DAR rectification with and without NPF. Latency is measured with batch size 1 and the main-table resolution. Storage reports the size of the rectification table only; this table is shared across batches and across prompts. The "overhead" columns are (NPF on)/(NPF off) $-$ 1, so a negative value indicates that NPF saves resources.

	Latency (ms/step)		GPU mem (MB)		Storage (MB)	Latency Δ
Setting	NPF off	NPF on	NPF off	NPF on		
LDM-4 (N=10, W8A8)	12.17	12.07	493.6	443.1	42.2	-0.82%
DiT-XL/2 (N=3, W8A8)	19.85	19.69	467.0	419.3	29.3	-0.81%
HunyuanVideo (W8A8)	312.8	310.2	24.6 GB	22.1 GB	203.4	-0.83%

NPF folds the rectification into existing GEMM and dequantization operations exactly (Section 3.6), so the latency difference is within measurement noise at all three backbones. Memory is reduced by roughly 10% because the rectification tensors share storage with the dequantization scale buffers and no temporary activation for the affine layers is materialized. The rectification table itself is a one-time per-model calibration artefact and is the only storage cost that DART adds on top of the base quantized backbone.