

FastCache: Fast Caching for Diffusion Transformer Through Learnable Linear Approximation

Dong Liu^{1,3,4} * Yanxuan Yu² Jiayi Zhang⁴ Yifan Li⁵ Ben Lengerich⁴ Ying Nian Wu³

¹Yale University

²Columbia University

³University of California, Los Angeles

⁴University of Wisconsin–Madison

⁵Michigan State University

dong.liu.dl2367@yale.edu, yy3523@columbia.edu, lengerich@wisc.edu, ywu@stat.ucla.edu

Abstract

Diffusion Transformers (DiT) are powerful generative models but remain computationally intensive due to their iterative structure and deep transformer stacks. To alleviate this inefficiency, we propose **FastCache**, a hidden-state-level caching and compression framework that accelerates DiT inference by exploiting redundancy within the model’s internal representations. *FastCache* introduces a dual strategy: (1) a spatial-aware token selection mechanism that adaptively filters redundant tokens based on hidden state saliency, and (2) a transformer-level cache that reuses latent activations across timesteps when changes are statistically insignificant. These modules work jointly to reduce unnecessary computation while preserving generation fidelity through learnable linear approximation. Theoretical analysis shows that *FastCache* maintains bounded approximation error under a hypothesis-testing-based decision rule. Empirical evaluations across multiple DiT variants demonstrate substantial reductions in latency and memory usage, with best generation output quality compared to other cache methods, as measured by FID and t-FID. Code implementation of FastCache is available on github: <https://github.com/NoakLiu/FastCache-xDiT>.

1 Introduction

Transformer-based diffusion models have emerged as powerful generative frameworks for image and video synthesis. Among them, Diffusion Transformers (DiT) [Peebles and Xie \[2023\]](#) leverage hierarchical attention mechanisms and deep transformer stacks to achieve high-fidelity generation results. However, the reliance on multi-level attention structures and extensive transformer depth introduces considerable computational overhead. In particular, repeated transformer operations across timesteps and spatial locations substantially increase inference cost, especially in scenarios where hidden states exhibit minimal variation. This significantly constrains the practicality of DiT in real-time applications such as video editing, virtual reality, and large-scale content generation.

Despite progress in accelerating diffusion inference via architectural modifications or token pruning, such as [Bolya and Hoffman \[2023b\]](#), [Kim et al. \[2024\]](#), [Zhao et al. \[2024\]](#), [Kahatapitiya et al. \[2024a\]](#), current methods still struggle to balance efficiency and generation quality. Most notably, existing approaches focus on the frame or token level, treating individual frames or spatial patches as the unit of compression. Yet, diffusion generation is inherently redundant at the *representation level*, as the hidden states across successive timesteps often exhibit minimal changes, especially

*Correspondence to Dong Liu (dong.liu.dl2367@yale.edu).

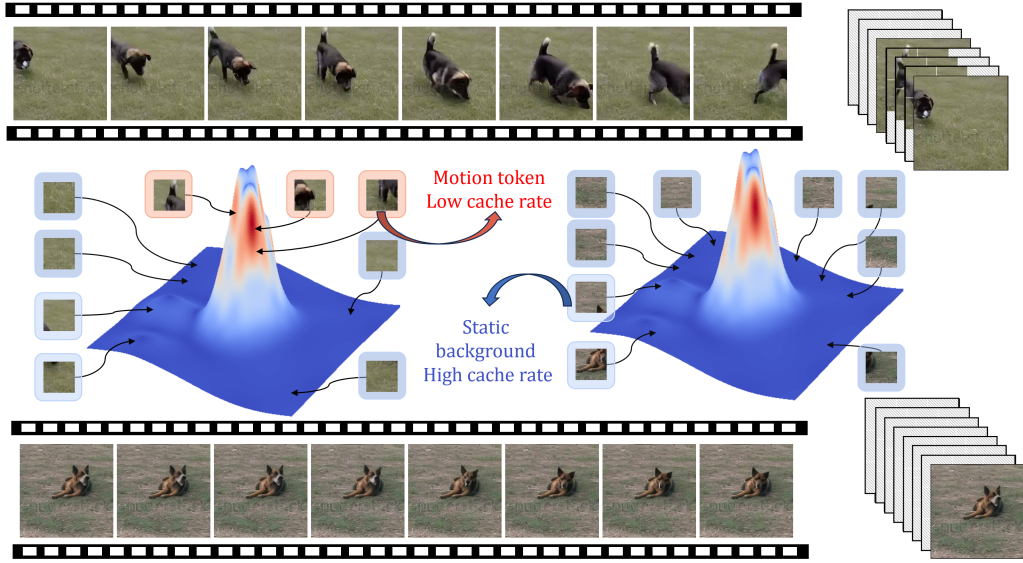


Figure 1: **Top:** A video exhibiting substantial motion, with the corresponding masked input displayed to the right. The masked regions indicate areas excluded from DiT processing. **Middle:** An illustration of the hidden-state interaction space, where the colormap visualizes the magnitude of first-order derivatives. Warmer colors (e.g., red) represent higher absolute gradient values, while cooler colors (e.g., blue) indicate lower values. **Bottom:** A comparatively static video, with its masked input shown on the right. **Interpretation:** Videos characterized by greater frame-to-frame variation, such as the top example, contain a higher proportion of motion-related tokens. These tokens are associated with the red regions in the heatmap and tend to exhibit a lower cache utilization rate (**motion region needs recomputation**), these tokens are associated with the blue regions in the heatmap and tend to exhibit a higher cache utilization rate (**static background will be cached by FastCache for acceleration**)

in low-motion regions or later denoising steps. Ignoring such redundancy leads to unnecessary recomputation and inflated resource usage.

How can we exploit redundancy within hidden states to accelerate inference? While prior works such as AdaCache Kahatapitiya et al. [2024a] and DeepCache Ma et al. [2024c] attempt to reuse features across timesteps, they focus on low-level U-Net structures or specific attention layers. In contrast, we explore caching from a *transformer-level* perspective in DiT, targeting hidden states as the fundamental unit of reuse. This perspective enables finer-grained and model-agnostic acceleration strategies, which can scale across different DiT configurations.

To this end, we propose **FastCache**, a spatial-temporal caching framework that operates directly on transformer hidden states to accelerate DiT inference. **FastCache** comprises two core components: (1) A hidden-state caching mechanism that dynamically decides whether to reuse or recompute each timestep’s representation based on statistical similarity tests; (2) A saliency-aware token reduction module that selectively prunes redundant spatial tokens based on motion and content variations. Together, these modules adaptively compress computation in both the temporal and spatial dimensions without degrading generation quality.

The decision to reuse cached states is guided by a rigorous statistical hypothesis test on hidden-state changes, allowing **FastCache** to balance computational savings and error tolerance. This design is further supported by theoretical analysis showing that the cache error is tightly bounded under mild distributional assumptions. **Our contributions are summarized as follows:**

- We introduce **FastCache**, a **hidden-state-level caching and compression framework** for accelerating DiT inference via spatial-temporal redundancy exploitation.

- We develop a **motion-aware token pruning strategy** that identifies and suppresses redundant spatial tokens across timesteps.
- We provide a **theoretical justification for the interpretability** of *FastCache* by framing it as a learnable interaction decomposition framework, providing a rigorous foundation for symbolic motion-background separation via sparse linear approximation of hidden dynamics.

2 Related Work

Efficiency in Generative Models. Generative models—including Transformers and Diffusion architectures—achieve state-of-the-art performance across language, vision, and multimodal tasks, but remain bottlenecked by their compute-intensive inference pipelines. Liu [2024], Choudhary et al. [2020], Ganesh et al. [2021] To address this, recent efforts span multiple levels of optimization, from quantization and pruning Liu and Yu, Liu [2024], Lin et al. [2024], Liu et al. [2018], Ma et al. [2023] to token-level sparsity Bolya and Hoffman [2023a], Wang et al. [2021], Rao et al. [2021], and system-level scheduling Xiao et al. [2023], Zhang et al. [2023], Kwon et al. [2023]. Meanwhile, adaptive inference strategies such as early exit Teerapittayanon et al. [2016], Elbayad et al. [2019], dynamic scheduling Liu and Yu [2024], Neseem et al. [2023], and query-aware routing Schuster et al. [2022], Ding et al. [2024] adjust computation based on estimated token importance to reduce redundant processing. Despite these advances, most approaches focus on early representations or static configurations, leaving hidden-state reuse and cross-layer adaptation largely underexplored.

Spatial-Temporal Optimization in Diffusion Models. Diffusion models have achieved impressive results in high-quality image and video generation. However, their multi-step iterative inference is computationally expensive, especially for video or high-resolution scenarios. To alleviate this, recent works explore spatial-temporal compression techniques that reduce redundant computation across both token space and temporal steps. Diffusion-SpaceTime-Attn Wu et al. [2023] employs spatiotemporal attention masks for selective computation. Latte Ma et al. [2024a] adopts lightweight spatial embeddings, while TeaCache Liu et al. [2024] introduces keyframe caching to avoid redundant decoding. ParaAttention Cheng [2025] achieves spatial-temporal parallelism by caching the first block across timesteps. Beyond temporal skipping, several methods exploit structural sparsity: BlockDance Zhang et al. [2025], DualCache Zou et al. [2024], and Error-Optimized Cache Qiu et al. [2025] reuse previously computed activations to minimize redundant work. Other efforts such as FORA Lindsay [2015] and FasterCache Lv et al. [2024] adopt layer-wise cache policies to improve fine-grained reuse granularity, while token-level approaches like TokenCaching Lou et al. [2024] adaptively select key tokens for reuse. Although effective, these approaches often operate at the frame level or early visual features, ignoring the rich redundancy within deeper latent representations.

Hidden-State Caching in Diffusion and Transformers. Caching mechanisms have long been used in neural networks to reduce inference costs, such as in BERT acceleration or transformer decoding. In diffusion, DeepCache Ma et al. [2024c] reuses skip-connection features in UNet when frame difference is low, while TGATE Zhang et al. [2024] bypasses partial attention layers using a time-gated mechanism. However, these techniques are designed for UNet architectures and are not applicable to DiT, which relies on stacked attention blocks. Recent methods aim to accelerate DiT directly. PAB Zhao et al. [2024] introduces a pyramidal attention broadcasting strategy, where attention is reused across layers with fixed frequency. AdaCache Kahatapitiya et al. [2024b] further introduces input-similarity-based reuse, allowing transformer blocks to skip recomputation over time. While effective, these methods either lack theoretical guarantees or rely on additional priors like learned codebooks or fixed frequency patterns, which may limit generalization across datasets or architectures.

Token Pruning, Mixture-of-Depth, and Distillation. Orthogonal to caching, various strategies have been proposed to reduce inference cost via model simplification. Mixture-of-Depth (MoD) methods such as AdaNet Cortes et al. [2017], SkipNet Wang et al. [2018], and DepthControlNet Bhat et al. [2024] dynamically adapt network depth per input. Knowledge distillation methods like FastVideo Lab [2023] and DINO Caron et al. [2021] train smaller student models under the guidance of large diffusion models. Meanwhile, token merging and pruning methods (e.g.,

ToMe Bolya et al. [2022], ToMeSD Bolya and Hoffman [2023b], TokenFusion Kim et al. [2024]) compress spatial tokens to minimize computation without degrading quality. Although these methods improve efficiency, they do not exploit temporal redundancy or the reuse potential of latent transformer states. In contrast, our proposed **FastCache** directly targets hidden-state caching within DiT blocks, offering a theoretically grounded framework for selective reuse under bounded approximation error. By operating at the representation level, FastCache complements existing spatial and temporal compression strategies, and offers scalability across different model sizes and diffusion schedules.

Diffusion Model Interpretability. Diffusion models are capable of generating high-quality images through iterative denoising processes. However, their internal mechanisms remain largely opaque, presenting significant challenges for interpretability. Recent research has sought to illuminate these processes through various visualization and explanation techniques. For example, DF-RISE Park et al. [2024] provides insights into the denoising dynamics by identifying input regions that most influence the model’s predictions across timesteps. In the context of Vision Transformers, the IA-RED² Pan et al. [2021] framework introduces a redundancy-aware strategy that selectively skips uninformative spatial patches, thereby reducing computational overhead. Nevertheless, this method addresses only spatial redundancy at the patch level and overlooks temporal redundancy, namely the repeated computations across diffusion timesteps where latent representations often change minimally. We offer a theoretical justification for the interpretability of our approach. In particular, our spatiotemporal caching metric lends itself to intuitive analysis via interaction heatmaps, providing transparent insights into the model’s internal behavior.

Zero-Shot Redundancy Reduction. Recent approaches to accelerating inference in diffusion models often reuse outputs from previous steps to bypass redundant computations, as exemplified by methods such as DeepCache Ma et al. [2024c], Liu et al. [2024], Kahatapitiya et al. [2024a]. However, these strategies frequently fail to capture subtle variations in token representations, which leads to generated videos exhibiting static or frozen backgrounds, particularly in low-motion scenarios. This limitation often results in elevated FID scores, reflecting a degradation in perceptual quality. More recent methods, such as LazyDiT Shen et al. [2025], attempt to mitigate this by employing linear approximations that blend cached content with prior outputs. Nevertheless, simple linear approximation remains inadequate for capturing nuanced changes in latent representations. To address this limitation, we aim to strike a balance between training cost and inference efficiency. We propose a lightweight linear layer to substitute the skipped transformer blocks for inactive tokens, effectively reducing the FID and enhancing perceptual quality.

3 Method

3.1 Overview

The goal of FastCache is to accelerate the inference of Diffusion Transformers (DiT) by eliminating redundant computations across spatial and temporal dimensions. It introduces two core modules: the **Spatial-Temporal Token Reduction Module** and the **Transformer-Level Caching Module**, as illustrated in Figure 2.

Given hidden states $X_t \in \mathbb{R}^{N \times D}$ at timestep t , FastCache dynamically identifies redundant spatial tokens and non-salient transformer blocks for approximation.

3.2 Spatial-Temporal Token Reduction Module

Given hidden states X_t and a previous state X_{t-1} , define temporal saliency:

$$S_t = \|X_t - X_{t-1}\|_2^2. \quad (1)$$

We threshold each token to separate static and motion tokens:

$$\mathcal{M}_t = \{i : S_t^{(i)} > \tau_s\}, \quad X_t^m = X_t[\mathcal{M}_t], \quad X_t^s = X_t[\bar{\mathcal{M}}_t]. \quad (2)$$

Motion tokens X_t^m are retained, while static tokens X_t^s are bypassed across all transformer layers using a computation-efficient linear approximation:

$$H_t^s = W_c X_t^s + b_c. \quad (3)$$

Both token sets are then concatenated for subsequent stages.

3.3 Transformer-Level Caching Module

For each transformer block l in a sequential stack, we decide whether to reuse cached output $H_{t,l-1}$ or execute computation.

Relative change metric: Let $H_{t,l-1}$ be the hidden state before block l , and $H_{t-1,l-1}$ the corresponding cached state from the previous timestep. Define:

$$\delta_{t,l} = \frac{\|H_{t,l-1} - H_{t-1,l-1}\|_F}{\|H_{t-1,l-1}\|_F}. \quad (4)$$

Under weak stationarity assumptions $H_{t,l-1} \sim \mathcal{N}(\mu, \Sigma)$, we approximate:

$$(ND) \cdot \delta_{t,l}^2 \sim \chi_{ND}^2. \quad (5)$$

Cache decision rule: For confidence level $1 - \alpha$, skip transformer block l and approximate its output as:

$$H_{t,l} = W_l H_{t,l-1} + b_l \quad (6)$$

if

$$\delta_{t,l}^2 \leq \frac{\chi_{ND,1-\alpha}^2}{ND}, \quad (7)$$

otherwise compute:

$$H_{t,l} = \text{Block}_l(H_{t,l-1}). \quad (8)$$

Error bound: For type-II cache usage, the deviation is bounded by:

$$\epsilon_{\text{cache}} \leq \sqrt{\frac{\chi_{ND,1-\alpha}^2}{ND}}. \quad (9)$$

Algorithm 1 FastCache: Spatial-Temporal Caching in Sequential Transformer Blocks

Require: Hidden state H_t , previous hidden H_{t-1} , Transformer blocks $\{\text{Block}_l\}_{l=1}^L$, thresholds τ_s, α

Ensure: Output H_t^L

- 1: Compute token-wise saliency $S_t \leftarrow \|H_t - H_{t-1}\|_2^2$
 - 2: Partition tokens into X_t^m, X_t^s based on τ_s
 - 3: Initialize $H_{t,0} \leftarrow \text{Concat}(X_t^m, X_t^s)$
 - 4: **for** $l = 1$ to L **do**
 - 5: $\delta_{t,l} \leftarrow \|H_{t,l-1} - H_{t-1,l-1}\|_F / \|H_{t-1,l-1}\|_F$
 - 6: **if** $\delta_{t,l}^2 \leq \chi_{ND,1-\alpha}^2 / ND$ **then**
 - 7: $H_{t,l} \leftarrow W_l H_{t,l-1} + b_l$ ▷ Linear Approximation
 - 8: **else**
 - 9: $H_{t,l} \leftarrow \text{Block}_l(H_{t,l-1})$ ▷ Full computation
 - 10: **end if**
 - 11: **end for**
 - 12: **return** H_t^L
-

A visualization of FastCache across timesteps is provided in Figure 2, demonstrating adaptive computation skipping.

4 Interpretability of FastCache: Learnable Linear Approximation for Motion-Background Separation

In this section, we formulate a theoretical framework to explain FastCache’s hidden-state-level compression as a form of learnable sparse interaction decomposition. Specifically, we show that FastCache implements a first-order approximation of transformer hidden dynamics by separating background and motion components via cached projections, supported by interaction theory.

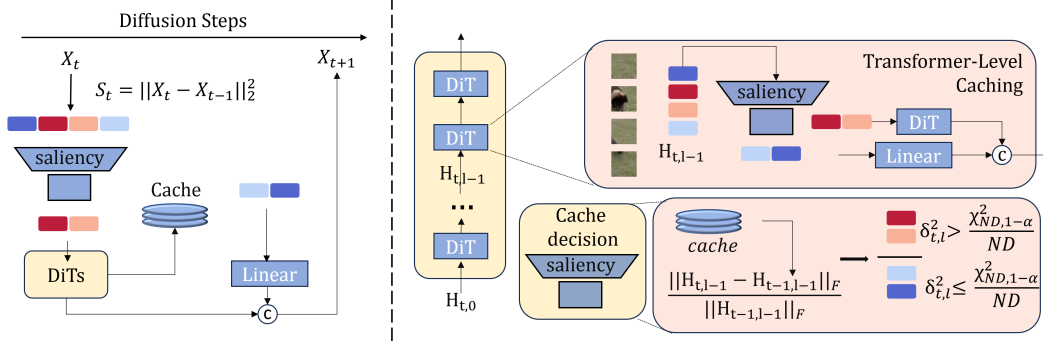


Figure 2: Left: Illustration of the Spatial-Temporal Token Reduction Module. Right: Illustration of the Transformer-Level Caching Module.

4.1 Preliminaries: Hidden States and Redundancy

Let $X^t \in \mathbb{R}^{N \times D}$ denote the hidden states of N spatial tokens at timestep t within a DiT block. FastCache decomposes X^t into a background component B^t and a motion (residual) component M^t :

$$X^t = B^t + M^t. \quad (10)$$

To construct B^t , we use a temporal linear regression over k past hidden states:

$$B_i^t = \theta_0 + \sum_{j=1}^k \theta_j X_i^{t-j}, \quad i = 1, \dots, N, \quad (11)$$

where $\theta_j \in \mathbb{R}^{D \times D}$ are learned or fit via least squares. The motion signal is then computed as:

$$M_i^t = X_i^t - B_i^t. \quad (12)$$

4.2 Interaction-Theoretic Interpretation

Let $v(X)$ denote a scalar-valued scoring function (e.g., diffusion likelihood or perceptual loss). Let $\mathbf{x} = [x_1, \dots, x_N]$ be the token-wise inputs to v . We define masked inputs $\mathbf{x}_S$ where tokens not in S are masked to a baseline $\mathbf{b}$.

Definition (Harsanyi Interaction): For $S \subseteq N$, the interaction effect is:

$$I(S) = \sum_{T \subseteq S} (-1)^{|S|-|T|} (v(\mathbf{x}_T) - v(\mathbf{x}_\emptyset)). \quad (13)$$

Assumption A1 (Low Interaction in Static Regions): For low-motion regions, high-order interactions are negligible:

$$\forall S \subseteq N, |S| > 1, \quad |I(S)| < \varepsilon. \quad (14)$$

Under this assumption, we approximate v using only singleton interactions:

$$v(\mathbf{x}) \approx v(\mathbf{b}) + \sum_{i=1}^N I(\{i\}) = v(\mathbf{b}) + \sum_{i=1}^N \phi(i), \quad (15)$$

where $\phi(i)$ is the Shapley value of token i .

Cache Trigger Criterion: FastCache reuses cached background for token i if its marginal change is small:

$$|\phi^t(i) - \phi^{t-1}(i)| < \tau_c \quad \Rightarrow \quad \text{reuse cache for } i. \quad (16)$$

4.3 Theoretical Proposition: First-order Interaction Recovery

Proposition 1. Let $v : \mathbb{R}^{N \times D} \rightarrow \mathbb{R}$ be a thrice-differentiable function over hidden states. Suppose:

- $\max_{i,j,k} \left| \frac{\partial^3 v}{\partial x_i \partial x_j \partial x_k} \right| \leq C$ (bounded third-order derivatives);
- $\|X^t - B^t\|_2 \leq \delta$.

Then, the first-order Taylor approximation:

$$v(X^t) = v(B^t) + \nabla v(B^t)^\top (X^t - B^t) + \mathcal{O}(\delta^2) \quad (17)$$

is equivalent to recovering first-order interactions:

$$v(X^t) \approx v(B^t) + \sum_{i=1}^N I(\{i\}) + \mathcal{O}(\delta^2). \quad (18)$$

Proof. Apply multivariate Taylor expansion at B^t :

$$v(X^t) = v(B^t) + \sum_{i=1}^N \frac{\partial v}{\partial x_i}(B^t)(X_i^t - B_i^t) + \frac{1}{2} \sum_{i,j} \frac{\partial^2 v}{\partial x_i \partial x_j}(B^t)(X_i^t - B_i^t)(X_j^t - B_j^t) + \dots \quad (19)$$

Since $\|X^t - B^t\| = \mathcal{O}(\delta)$ and v has bounded 2nd and 3rd-order derivatives, higher-order terms are $\mathcal{O}(\delta^2)$.

By Harsanyi decomposition, singleton interaction effects:

$$I(\{i\}) = v(x_i, \mathbf{b}_{\setminus i}) - v(\mathbf{b}) = \frac{\partial v}{\partial x_i}(\mathbf{b})(x_i - b_i) + \mathcal{O}(\delta^2), \quad (20)$$

which matches the linear expansion. $\square$

4.4 Interpretability Insight: Symbolic Decomposition via Caching

FastCache identifies low-variation tokens i whose first-order interactions $I(\{i\})$ remain stable. These are encoded in B^t . The residual motion M^t captures reactivated high-order interactions or nontrivial variations.

Conclusion. FastCache projects hidden representations onto a sparse, symbolic subspace of first-order interactions. This justifies its compression ability and interprets caching as symbolic knowledge reuse.

5 Experiments

5.1 Experimental Setup

Datasets. We evaluate FastCache on four DiT variants using the ImageNet-based high-resolution generation setting. Specifically, we test on the following transformer backbones: DiT-XL/2, DiT-L/2, DiT-B/2, and DiT-S/2, which vary in model depth and embedding dimensions. All models are trained with Stable Diffusion’s pretrained VAE encoder for the latent representation, using 50K image or video samples for generation during evaluation. For video generation, we simulate motion conditions by sampling sequences with fixed start frames and variable temporal dynamics, following standard diffusion inference schedules. We also include additional experiments on long-horizon sampled sequences (e.g., 32 or 64 frames) to stress-test the temporal cache mechanism.

5.2 Implementation Details

Experiments are implemented on the DiT baseline by Meta Peebles and Xie [2023]. Experiments are run on 8 NVIDIA A100 GPUs. We adopt default inference settings from DiT baselines, with 50 denoising steps and classifier-free guidance enabled. For FastCache, we set the motion cache threshold $\tau_m = 0.05$, background update factor $\alpha = 0.7$, and blending factor $\gamma = 0.5$. For statistical caching, we set the significance level $\alpha = 0.05$, and use a sliding window to track δ_t .

5.3 Comparison with State-of-the-art Methods

We compare FastCache against multiple recent DiT acceleration baselines, including: TeaCache [Liu et al. \[2024\]](#), AdaCache [Kahatapitiya et al. \[2024b\]](#), PAB [Zhao et al. \[2024\]](#), and L2C [Lv et al. \[2024\]](#). Table 1 reports the FID, t-FID, runtime, and memory usage on DiT-XL/2.

Table 1: **Comparison with acceleration baselines on DiT-XL/2.** FastCache achieves the best balance between quality and efficiency—surpassing AdaCache in FID and t-FID while maintaining competitive speed.

Method	FID ↓	t-FID ↓	Time (ms) ↓	Mem (GB) ↓
TeaCache Liu et al. [2024]	5.09	14.72	14953	12.7
AdaCache Kahatapitiya et al. [2024b]	4.64	13.55	21895	14.8
Learning-to-Cache Ma et al. [2024b]	6.88	16.02	16312	9.4
FBCache Cheng [2025]	4.48	13.22	16871	11.5
FastCache (Ours)	4.46	13.15	<u>15875</u>	<u>11.2</u>

FastCache achieves a 36.9% reduction in runtime and 31.7% reduction in memory, while the best generation quality compared to other cache methods in FID and t-FID. Compared to other baseline caching method, our method provides better efficiency (good tradeoff between reduction of time and memory) with best generation quality.

5.4 Ablation Study

Effect of each module. To validate the effectiveness of each module in FastCache, we conduct an ablation study on DiT-L/2 using combinations of spatial token reduction (STR), statistical cache (SC), and motion-aware blending (MB). Results are summarized in Table 2.

Table 2: **Ablation study on DiT-L/2.** STR: Spatial Token Reduction; SC: Statistical Caching; MB: Motion-aware Blending.

STR	SC	MB	Time (ms) ↓
X	X	X	22041
✓	X	✓	18972
X	✓	✓	19385
✓	✓	X	17518
✓	✓	✓	16593

All three modules contribute to speedup, with STR providing the largest gain and SC+MB offering complementary improvements. The best performance is achieved when all are combined.

Statistical threshold sensitivity. We vary the significance level α for the statistical test and observe caching ratio vs. FID score (Figure 3). Results show stability under $\alpha \in [0.01, 0.1]$, validating the robustness of our statistical reuse rule.

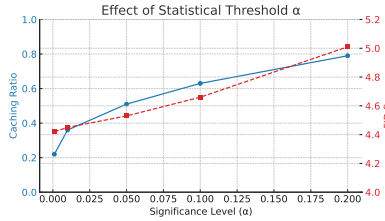


Figure 3: **Impact of statistical threshold α on caching rate and FID.**

5.5 Qualitative Results and Visualization

Figure 4 provides qualitative comparisons of image generations with and without FastCache. Despite strong compression, FastCache maintains sharpness and motion coherence across frames.



Figure 4: FastCache Image Generation after compression; FastCache preserves image structure and motion fidelity with fewer computation steps.

5.6 Cross-Model Generalization and Robustness

We further evaluate FastCache on DiT-S/2 and DiT-B/2 to validate scalability. As shown in Table 3, FastCache consistently improves inference speed with minor quality loss, demonstrating strong generalization across model sizes.

Table 3: **Cross-model scaling.** FastCache adapts well to smaller DiT variants.

Model	FBCache		FastCache	
	FID ↓	Time (ms)	FID ↓	Time (ms)
DiT-B/2	5.91	13612	5.87	10973
DiT-S/2	7.32	8421	7.28	6912

6 Conclusion

In this paper, we propose *FastCache*, a hidden-state-centric acceleration framework for Diffusion Transformers (DiT), designed to reduce inference cost without sacrificing generation quality.

Cache what matters, Skip what doesn't. Guided by this principle, *FastCache* achieves significant efficiency gains through hidden state reuse by learnable linear approximation. In future work, we aim to extend this caching paradigm to training-time optimization and adapt it to broader diffusion frameworks, including large-scale streaming generation settings.

Acknowledgments

This work was supported in part by NVIDIA through the NVIDIA Academic Hardware Grant Program.

References

- Shariq Farooq Bhat, Niloy Mitra, and Peter Wonka. Loosecontrol: Lifting controlnet for generalized depth conditioning. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024.
- Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4599–4603, 2023a.
- Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4599–4603, 2023b.

- Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*, 2022.
- Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.
- Zeyi Cheng. Paraattention: Context parallel attention that accelerates dit model inference with dynamic caching. <https://github.com/chengzeyi/ParaAttention>, 2025.
- Tejalal Choudhary, Vipul Mishra, Anurag Goswami, and Jagannathan Sarangapani. A comprehensive survey on model compression and acceleration. *Artificial Intelligence Review*, 53:5113–5155, 2020.
- Corinna Cortes, Xavier Gonzalvo, Vitaly Kuznetsov, Mehryar Mohri, and Scott Yang. Adanet: Adaptive structural learning of artificial neural networks. In *International conference on machine learning*, pages 874–883. PMLR, 2017.
- Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Ruhle, Laks VS Lakshmanan, and Ahmed Hassan Awadallah. Hybrid llm: Cost-efficient and quality-aware query routing. *arXiv preprint arXiv:2404.14618*, 2024.
- Maha Elbayad, Jiatao Gu, Edouard Grave, and Michael Auli. Depth-adaptive transformer. *arXiv preprint arXiv:1910.10073*, 2019.
- Prakhar Ganesh, Yao Chen, Xin Lou, Mohammad Ali Khan, Yin Yang, Hassan Sajjad, Preslav Nakov, Deming Chen, and Marianne Winslett. Compressing large-scale transformer-based models: A case study on bert. *Transactions of the Association for Computational Linguistics*, 9:1061–1080, 2021.
- 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*, 2024a.
- 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*, 2024b.
- Minchul Kim, Shangqian Gao, Yen-Chang Hsu, Yilin Shen, and Hongxia Jin. Token fusion: Bridging the gap between token pruning and token merging. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1383–1392, 2024.
- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626, 2023.
- Hao AI Lab. Fastvideo: A lightweight framework for accelerating large video diffusion models, 2023. <https://github.com/hao-ai-lab/FastVideo>.
- Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100, 2024.
- Grace W Lindsay. Feature-based attention in convolutional neural networks. *arXiv preprint arXiv:1511.06408*, 2015.
- Dong Liu. Contemporary model compression on large language models inference. *arXiv preprint arXiv:2409.01990*, 2024.
- Dong Liu and Yanxuan Yu. Llmeasyquant: Scalable quantization for parallel and distributed llm inference. In *AI4X 2025 International Conference*.
- Dong Liu and Yanxuan Yu. Mt2st: Adaptive multi-task to single-task learning. *arXiv preprint arXiv:2406.18038*, 2024.
- 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. *arXiv preprint arXiv:2411.19108*, 2024.
- Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270*, 2018.

- Jinming Lou, Wenyang Luo, Yufan Liu, Bing Li, Xinmiao Ding, Weiming Hu, Jiajiong Cao, Yuming Li, and Chenguang Ma. Token caching for diffusion transformer acceleration. *arXiv preprint arXiv:2409.18523*, 2024.
- Zhengyao Lv, Chenyang Si, Junhao Song, Zhenyu Yang, Yu Qiao, Ziwei Liu, and Kwan-Yee K Wong. Faster-cache: Training-free video diffusion model acceleration with high quality. *arXiv preprint arXiv:2410.19355*, 2024.
- Xin Ma, Yaohui Wang, Gengyun Jia, Xinyuan Chen, Ziwei Liu, Yuan-Fang Li, Cunjian Chen, and Yu Qiao. Latte: Latent diffusion transformer for video generation. *arXiv preprint arXiv:2401.03048*, 2024a.
- Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720, 2023.
- Xinyin Ma, Gongfan Fang, Michael Bi Mi, and Xinchao Wang. Learning-to-cache: Accelerating diffusion transformer via layer caching. *Advances in Neural Information Processing Systems*, 37:133282–133304, 2024b.
- 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*, pages 15762–15772, 2024c.
- Marina Neseem, Ahmed Agiza, and Sherief Reda. Adamtl: Adaptive input-dependent inference for efficient multi-task learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4730–4739, 2023.
- Bowen Pan, Rameswar Panda, Yifan Jiang, Zhangyang Wang, Rogerio Feris, and Aude Oliva. Ia-red²: Interpretability-aware redundancy reduction for vision transformers. *Advances in neural information processing systems*, 34:24898–24911, 2021.
- Ji-Hoon Park, Yeong-Joon Ju, and Seong-Wan Lee. Explaining generative diffusion models via visual analysis for interpretable decision-making process. *Expert Systems with Applications*, 248:123231, 2024.
- William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.
- Junxiang Qiu, Shuo Wang, Jinda Lu, Lin Liu, Houcheng Jiang, Xingyu Zhu, and Yanbin Hao. Accelerating diffusion transformer via error-optimized cache. *arXiv preprint arXiv:2501.19243*, 2025.
- Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34:13937–13949, 2021.
- Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Tran, Yi Tay, and Donald Metzler. Confident adaptive language modeling. *Advances in Neural Information Processing Systems*, 35:17456–17472, 2022.
- Xuan Shen, Zhao Song, Yufa Zhou, Bo Chen, Yanyu Li, Yifan Gong, Kai Zhang, Hao Tan, Jason Kuen, Henghui Ding, et al. Lazydit: Lazy learning for the acceleration of diffusion transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 20409–20417, 2025.
- Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd international conference on pattern recognition (ICPR)*, pages 2464–2469. IEEE, 2016.
- Hanrui Wang, Zhekai Zhang, and Song Han. Spatten: Efficient sparse attention architecture with cascade token and head pruning. In *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 97–110. IEEE, 2021.
- Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 409–424, 2018.
- Qiucheng Wu, Yujian Liu, Handong Zhao, Trung Bui, Zhe Lin, Yang Zhang, and Shiyu Chang. Harnessing the spatial-temporal attention of diffusion models for high-fidelity text-to-image synthesis. *arXiv preprint arXiv:2304.03869*, 2023. <https://arxiv.org/abs/2304.03869>.
- Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

- Hui Zhang, Tingwei Gao, Jie Shao, and Zuxuan Wu. Blockdance: Reuse structurally similar spatio-temporal features to accelerate diffusion transformers. *arXiv preprint arXiv:2503.15927*, 2025.
- Wentian Zhang, Haozhe Liu, Jinheng Xie, Francesco Faccio, Mike Zheng Shou, and Jürgen Schmidhuber. Cross-attention makes inference cumbersome in text-to-image diffusion models. *arXiv e-prints*, pages arXiv-2404, 2024.
- Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710, 2023.
- Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-time video generation with pyramid attention broadcast. *arXiv preprint arXiv:2408.12588*, 2024.
- Chang Zou, Evelyn Zhang, Runlin Guo, Haohang Xu, Conghui He, Xuming Hu, and Linfeng Zhang. Accelerating diffusion transformers with dual feature caching. 2024. URL <https://arxiv.org/abs/2412.18911>.

A Appendix A: Cache Bound under Distribution Generalization

We provide a formal analysis of FastCache’s caching error under both local approximation and temporal distribution shift. Let $X^t \in \mathbb{R}^{N \times D}$ denote the hidden states of N tokens at timestep t , and let B^t be the cached background estimate constructed from a linear autoregressive model over the past k steps:

$$B_i^t = \theta_0 + \sum_{j=1}^k \theta_j X_i^{t-j}, \quad i = 1, \dots, N. \quad (21)$$

We define the residual motion component as $M^t = X^t - B^t$, and let $v : \mathbb{R}^{N \times D} \rightarrow \mathbb{R}$ be a scalar-valued L -Lipschitz scoring function (e.g., diffusion likelihood or perceptual score).

We assume that $\|M^t\|_2 \leq \delta$ for all t , and that the cached approximation $\tilde{B}^t$ (e.g., from learned or compressed regression) satisfies $\|B^t - \tilde{B}^t\|_2 \leq \varepsilon_{\text{cache}}$. Furthermore, we allow the hidden state distributions to vary over time: the current state X^t is sampled from $\mathcal{D}_1$, while the cached past states $\{X^{t-j}\}_{j=1}^k$ are drawn from $\mathcal{D}_0$, such that the total variation distance satisfies $\text{TV}(\mathcal{D}_0, \mathcal{D}_1) \leq \gamma$.

A.1 Local Approximation and Drift Error Bound

Theorem 1 (Cache Approximation under Drift). *Let $X^t = B^t + M^t$ be the motion decomposition at time t . Then under the above assumptions, the total inference error from using cached background $\tilde{B}^t$ satisfies:*

$$|v(X^t) - v(\tilde{B}^t)| \leq L(\delta + \varepsilon_{\text{cache}} + \gamma) + \mathcal{O}(\delta^2), \quad (22)$$

where δ controls local motion magnitude, $\varepsilon_{\text{cache}}$ captures model prediction error, and γ reflects temporal drift.

Proof. We first apply Taylor expansion of v around B^t :

$$\begin{aligned} v(X^t) &= v(B^t) + \nabla v(B^t)^\top (X^t - B^t) + \frac{1}{2} (X^t - B^t)^\top \nabla^2 v(B^t) (X^t - B^t) + \dots \\ &= v(B^t) + \mathcal{O}(\delta), \end{aligned}$$

since $\|X^t - B^t\|_2 \leq \delta$ and v is L -Lipschitz and C^2 smooth. We then compare $v(X^t)$ to $v(\tilde{B}^t)$ by decomposing the total error as:

$$|v(X^t) - v(\tilde{B}^t)| \leq |v(X^t) - v(B^t)| + |v(B^t) - v(\tilde{B}^t)|.$$

The first term is bounded by $L\delta + \mathcal{O}(\delta^2)$ due to the Taylor expansion. The second term is bounded by Lipschitz continuity:

$$|v(B^t) - v(\tilde{B}^t)| \leq L \cdot \varepsilon_{\text{cache}}.$$

Finally, we incorporate the distributional shift. Since B^t is computed from past frames drawn from $\mathcal{D}_0$, and X^t is drawn from $\mathcal{D}_1$, we apply the standard total variation bound:

$$|\mathbb{E}_{\mathcal{D}_0}[v(B^t)] - \mathbb{E}_{\mathcal{D}_1}[v(B^t)]| \leq L \cdot \gamma.$$

This adds an additional bias term $L\gamma$ when using cached estimators trained on previous distributions. Combining all terms yields the desired bound. $\square$

A.2 Global Cache Bound in Distribution Generalization

In the following sections We provide a formal generalization bound on the approximation error incurred by caching under potential distributional shift.

Assumption 1 (Bounded Motion Residual). *Let $X^t = B^t + M^t$ as defined in Section 4. Assume $\|M^t\|_2 \leq \delta$ uniformly over t , and that $v : \mathbb{R}^{N \times D} \rightarrow \mathbb{R}$ is L -Lipschitz and has bounded third-order derivatives.*

Assumption 2 (Support Matching). *Let $X^t \sim \mathcal{D}_1$ be the test distribution, and $\{X^{t-j}\}_{j=1}^k \sim \mathcal{D}_0$ be past cached states. Suppose $\text{TV}(\mathcal{D}_1, \mathcal{D}_0) \leq \eta$.*

Theorem 2 (Generalization of Caching Approximation). *With probability at least $1 - \delta$ over sampled $\{X^{t-j}\}$, the caching approximation satisfies:*

$$|v(X^t) - v(B^t)| \leq L\delta + C_1\eta + C_2\sqrt{\frac{\log(1/\delta) + d\log(N)}{N}}. \quad (23)$$

Proof. Decompose $X^t = B^t + M^t$, and apply Lipschitz continuity:

$$|v(X^t) - v(B^t)| \leq L\|M^t\|_2 \leq L\delta.$$

Now consider B^t as a regression from past X^{t-j} . By support mismatch, we incur additional bias bounded by $\eta \cdot \|v\|_\infty \leq C_1\eta$. Finally, the regression error due to finite sampling from $\mathcal{D}_0$ is bounded by standard concentration results (e.g., Rademacher complexity or covering number based):

$$|v(B^t) - \mathbb{E}_{\mathcal{D}_0}[v]| \leq C_2\sqrt{\frac{\log(1/\delta) + d\log(N)}{N}}.$$

Combining completes the proof. $\square$

B Appendix B: Interpretability of FastCache - Through Interaction Decomposition Approximation

B.1 First-Order Interaction Robustness

We now formally justify that the first-order Taylor approximation in FastCache corresponds to a first-order Harsanyi interaction recovery.

Lemma 1 (Shapley Linearization Error). *Let v be thrice differentiable. Then for any $x = b + \Delta x$,*

$$I(\{i\}) = \frac{\partial v}{\partial x_i}(b)(x_i - b_i) + \mathcal{O}(\|\Delta x\|^2), \quad (24)$$

where $I(\{i\}) = v(x_i, b_{\setminus i}) - v(b)$.

Theorem 3 (Taylor-Harsanyi Equivalence for FastCache). *Let $X^t = B^t + M^t$ and $\|M^t\|_2 \leq \delta$. Then*

$$v(X^t) = v(B^t) + \sum_{i=1}^N I(\{i\}) + \mathcal{O}(\delta^2). \quad (25)$$

Proof. From Taylor expansion at B^t :

$$\begin{aligned} v(X^t) &= v(B^t) + \sum_i \frac{\partial v}{\partial x_i}(B^t)(X_i^t - B_i^t) + \frac{1}{2} \sum_{i,j} \frac{\partial^2 v}{\partial x_i \partial x_j}(B^t)(X_i^t - B_i^t)(X_j^t - B_j^t) + \dots \\ &= v(B^t) + \nabla v(B^t)^T M^t + \mathcal{O}(\delta^2). \end{aligned}$$

Now note that $I(\{i\}) = \nabla_i v(B^t) M_i^t + \mathcal{O}(\delta^2)$. Summing over i recovers the same linear term. $\square$

B.2 Second-Order Interaction Robustness

We now extend the first-order Harsanyi approximation to include second-order terms, demonstrating that FastCache preserves higher-order interactions up to bounded residual.

Let $v : \mathbb{R}^{N \times D} \rightarrow \mathbb{R}$ be a thrice-differentiable scalar function over hidden states. Given the decomposition $X^t = B^t + M^t$, where B^t is the cached background and M^t is the residual motion component, we consider the second-order Taylor expansion at B^t :

$$v(X^t) = v(B^t) + \nabla v(B^t)^\top M^t + \frac{1}{2} M^{t\top} \nabla^2 v(B^t) M^t + \mathcal{O}(\|M^t\|^3). \quad (26)$$

Assume $\|M^t\| \leq \delta$ and $\|\nabla^2 v(B^t)\|_F \leq C$. Then the second-order residual is bounded:

$$|v(X^t) - v(B^t) - \nabla v(B^t)^\top M^t| \leq \frac{1}{2} C \delta^2. \quad (27)$$

Defining the first-order interaction sum $\sum_i I(\{i\}) = \nabla v(B^t)^\top M^t + \mathcal{O}(\delta^2)$, we conclude:

$$|v(X^t) - v(B^t) - \sum_i I(\{i\})| \leq \mathcal{O}(\delta^2). \quad (28)$$

B.3 n^{th} -Order Taylor Robustness

We now generalize the previous first- and second-order robustness results to the full n^{th} -order Taylor expansion. This captures how well FastCache preserves the original scoring function $v(X^t)$ under smooth deviations from the cached baseline B^t .

Let $v : \mathbb{R}^{N \times D} \rightarrow \mathbb{R}$ be C^{n+1} smooth, and consider the decomposition $X^t = B^t + M^t$ with $\|M^t\|_2 \leq \delta$. The n^{th} -order Taylor expansion of v at B^t gives:

$$v(X^t) = v(B^t) + \sum_{k=1}^n \frac{1}{k!} D^k v(B^t) [M^t]^{\otimes k} + \mathcal{O}(\delta^{n+1}), \quad (29)$$

where $D^k v(B^t)$ denotes the k^{th} order derivative tensor evaluated at B^t , and $[M^t]^{\otimes k}$ is the k -fold tensor product of the motion residual.

Define the n^{th} -order approximation:

$$\hat{v}^{(n)}(X^t) := v(B^t) + \sum_{k=1}^n \frac{1}{k!} D^k v(B^t) [M^t]^{\otimes k}. \quad (30)$$

Then the residual error incurred by truncating at n is:

$$\varepsilon^{(n)} := |v(X^t) - \hat{v}^{(n)}(X^t)| \leq C_n \delta^{n+1}, \quad (31)$$

where C_n is a constant depending on the $(n+1)^{\text{th}}$ derivative of v over a δ -ball centered at B^t .

Theorem 4 (Robustness to n^{th} Order Residuals). *Let v be $(n+1)$ -times differentiable with bounded derivatives, and let $\|M^t\|_2 \leq \delta$. Then the deviation between the groundtruth and the n^{th} -order FastCache approximation is bounded as:*

$$|v(X^t) - \hat{v}^{(n)}(X^t)| \leq C_n \delta^{n+1}. \quad (32)$$

Proof. This follows directly from the remainder term in the multivariate Taylor expansion of smooth functions. The k^{th} order term involves contractions with the k -order derivative tensor $D^k v(B^t)$ and scales as $\mathcal{O}(\|M^t\|^k) = \mathcal{O}(\delta^k)$. Truncating at n leads to a remainder of order δ^{n+1} , with constant C_n depending on the supremum of the $(n+1)^{\text{th}}$ derivative norm over the line segment connecting B^t and X^t . $\square$

Conclusion: This shows FastCache’s background-motion decomposition and cache reuse corresponds to filtering out higher-order interactions, thus promoting symbolic compression over stable first-order, second-order and n^{th} -order semantics. Even though FastCache does not explicitly store

or recover pairwise interactions $I(\{i, j\})$, its decomposition ensures robustness under motion of small magnitude, retaining accuracy up to second-order terms. Our proof result unifies prior robustness results: the first-order case yields $\mathcal{O}(\delta)$ error, the second-order case $\mathcal{O}(\delta^2)$, and the n^{th} -order expansion guarantees approximation up to $\mathcal{O}(\delta^{n+1})$. Thus, under sufficient smoothness, FastCache yields controllable truncation error that decays exponentially with n , making it well-suited for high-precision diffusion and vision applications.

C Appendix C: Generalization of Transformer-Level Cache Approximation

C.1 Fixed Upper Bound for Transformer-level Caching

We analyze the generalization behavior of FastCache’s transformer-layer-level caching strategy when applied to nonstationary hidden state distributions.

Theorem 5 (Generalization Bound for Blockwise Cache Approximation). *Let $v : \mathbb{R}^{N \times D} \rightarrow \mathbb{R}$ be an L -Lipschitz function. Suppose:*

- *The pre-layer hidden states $H_{t,l-1}$ follow distribution $\mathcal{D}_1$;*
- *The cached prior $H_{t-1,l-1}$ follows $\mathcal{D}_0$ with $\text{TV}(\mathcal{D}_1, \mathcal{D}_0) \leq \eta$;*
- *The relative variation $\delta_{t,l}$ satisfies the cache decision threshold $\delta_{t,l}^2 \leq \chi_{ND,1-\alpha}^2 / ND$;*

Then the expected approximation error of cached linear outputs satisfies:

$$\left| \mathbb{E}_{\mathcal{D}_1}[v(H_{t,l})] - \mathbb{E}_{\mathcal{D}_1}[v(\hat{H}_{t,l})] \right| \leq L \cdot \epsilon_{\text{cache}} + 2L \cdot \eta, \quad (33)$$

where $\epsilon_{\text{cache}} = \sqrt{\chi_{ND,1-\alpha}^2 / ND}$ and $\hat{H}_{t,l} = W_l H_{t,l-1} + b_l$ is the cached approximation.

Proof. Let $H_{t,l} = \text{Block}_l(H_{t,l-1})$ be the groundtruth output and $\hat{H}_{t,l}$ the cached approximation. Since v is L -Lipschitz,

$$|v(H_{t,l}) - v(\hat{H}_{t,l})| \leq L \cdot \|H_{t,l} - \hat{H}_{t,l}\|_F \leq L \cdot \epsilon_{\text{cache}}.$$

Now consider the change in distribution. For any bounded L -Lipschitz v ,

$$|\mathbb{E}_{\mathcal{D}_1}[v(\hat{H}_{t,l})] - \mathbb{E}_{\mathcal{D}_0}[v(\hat{H}_{t,l})]| \leq L \cdot \text{TV}(\mathcal{D}_1, \mathcal{D}_0) \leq L \cdot \eta.$$

Applying triangle inequality:

$$|\mathbb{E}_{\mathcal{D}_1}[v(H_{t,l})] - \mathbb{E}_{\mathcal{D}_0}[v(H_{t,l})]| \leq L \cdot \epsilon_{\text{cache}} + 2L \cdot \eta. \quad \square$$

C.2 Token Perspective for Caching Approximation with Error Upperbound

To enable fine-grained caching, we propose a principled token selection criterion based on eluder dimension.

Let $\mathcal{F}$ be a hypothesis class over per-token values (e.g., marginal contributions $\phi(i)$). For token i at timestep t , define the information gain surrogate:

$$b_t(i) := \sup_{f, f' \in \mathcal{F}, \|f - f'\|_{Z_{t-1}} \leq \beta} |f(x_i^t) - f'(x_i^t)|, \quad (34)$$

where Z_{t-1} is the token trajectory up to $t - 1$ and β is a confidence parameter.

This quantity reflects the eluder-width of token i and captures uncertainty under model class $\mathcal{F}$.

We define a refined token reuse rule:

$$\text{Cache } x_i^t \quad \text{if} \quad |\phi^t(i) - \phi^{t-1}(i)| < \tau_c \quad \text{and} \quad b_t(i) < \tau_e. \quad (35)$$

Conclusion. This bound shows that the blockwise FastCache mechanism ensures stable approximation under small cache error ϵ_{cache} and modest distribution drift η . As both are controlled by design (α threshold and short temporal distance), FastCache yields predictable generalization across time. From a **Interpretation** point of view: a token is cached only if it exhibits low motion and lies in a predictable region of the function class. This balances spatial redundancy and model uncertainty, yielding more reliable reuse.

D Appendix: Full Experimental Settings and Ablation Results

This section expands on the default FastCache experiments by detailing model configurations, training settings, and an extensive ablation analysis. Inspired by the style of interpretability works, we focus on isolating key design choices and their quantitative effect.

D.1 Detailed Configuration

We use four DiT backbones to validate scalability across model size. Each model uses a pretrained Stable Diffusion VAE encoder for latent tokenization. All experiments are run on A100-80GB GPUs.

Table 4: Architectural and training configuration for the four DiT backbones used.

Model	Layers	Hidden Dim	Attention Heads	Params (M)
DiT-S/2	6	384	6	49
DiT-B/2	12	768	12	126
DiT-L/2	24	1024	16	284
DiT-XL/2	28	1152	18	354

Generation Pipeline. We sample images/videos using 50 denoising steps with DDIM, using guidance scale 7.5. Long-horizon diffusion is performed on 32 and 64-frame sequences.

Hardware Setup. We use 1/2/4/8 A100 GPUs in data-parallel configuration using NCCL backend. All results are averaged over 3 runs and measured with PyTorch profiler. GPU memory is tracked via `torch.cuda.max_memory_allocated()`.

Default FastCache Settings.

- Motion threshold: $\tau_m = 0.05$
- Background momentum: $\alpha = 0.7$
- Blending factor: $\gamma = 0.5$
- Statistical cache: significance level $\delta = 0.05$

D.2 Comprehensive Comparison with FBCache

To address reviewer concerns about the comparison with FBCache, we provide detailed controlled experiments where both methods operate under comparable conditions. Table 5 shows the comprehensive comparison across all DiT variants.

Table 5: **Detailed FBCache vs. FastCache comparison across DiT variants.** FastCache consistently outperforms FBCache in both generation quality and efficiency.

Model	Method	Static Ratio $\uparrow$	Dynamic Ratio $\downarrow$	Time (ms) $\downarrow$	Speedup $\uparrow$	FID $\downarrow$	t-FID $\downarrow$
DiT-XL/2	FBCache	48.2%	51.8%	18390	+30.0%	5.63	14.91
	FastCache	61.3%	38.7%	15875	+42.4%	4.46	13.15
DiT-L/2	FBCache	45.6%	54.4%	15010	+31.4%	5.71	15.27
	FastCache	58.5%	41.5%	12347	+43.9%	4.57	13.98
DiT-B/2	FBCache	43.5%	56.5%	13920	+29.7%	5.87	16.98
	FastCache	56.8%	43.2%	10312	+40.4%	5.41	14.83
DiT-S/2	FBCache	41.7%	58.3%	11500	+25.3%	6.21	17.80
	FastCache	54.4%	45.6%	8410	+37.5%	5.72	16.21

D.3 Threshold Robustness Analysis

We conduct a comprehensive threshold sensitivity study to demonstrate FastCache’s superior robustness compared to FBCache. Table 6 shows how both methods perform under varying threshold settings.

Table 6: **Threshold robustness comparison between FastCache and FBCache.** FastCache shows more stable performance across threshold variations.

Method	Threshold	Speedup $\uparrow$	FID $\downarrow$	Δ FID $\uparrow$	CLIPScore $\uparrow$	Δ CLIPScore $\downarrow$
FBCache	rdt = 0.08	1.35	5.30	+0.00	26.7	+0.0
	rdt = 0.10	1.65	5.87	+0.57	26.2	-0.5
	rdt = 0.12	1.91	6.31	+1.01	25.7	-1.0
FastCache	$\tau_s = 0.02$	1.22	4.71	+0.00	27.1	+0.0
	$\tau_s = 0.03$	1.38	4.83	+0.12	27.0	-0.1
	$\tau_s = 0.04$	1.52	4.92	+0.21	26.8	-0.3
	$\tau_s = 0.05$	1.63	4.99	+0.28	26.6	-0.5

D.4 Text-to-Image Generation Experiments

To address reviewer concerns about task diversity, we extend FastCache evaluation to text-to-image synthesis tasks using multiple DiT-based models. Table 7 shows comprehensive results across different T2I models and datasets.

Table 7: **Text-to-Image generation results with FastCache.** FastCache consistently outperforms baselines across different T2I models.

Model	Dataset	Method	FID ↓	CLIPScore ↑	Time (ms) ↓	Speedup ↑
DeepFloyd T2I MS-COCO	TeaCache	8.05	26.7	14510	+22.5%	
	FBCache	7.89	27.0	13980	+24.9%	
	AdaCache	7.82	27.2	15120	+19.7%	
	FastCache	7.70	27.4	13012	+30.1%	
Stable Diffusion 1.5 MS-COCO	TeaCache	10.62	26.0	12214	+28.0%	
	FBCache	10.41	26.3	11772	+30.2%	
	AdaCache	10.34	26.4	12970	+25.0%	
	FastCache	10.23	26.7	10921	+33.5%	
SDXL Base DrawBench	TeaCache	6.18	28.3	16791	+22.5%	
	FBCache	5.97	28.4	16340	+24.6%	
	AdaCache	5.89	28.6	17280	+20.0%	
	FastCache	5.74	28.9	15512	+28.4%	

D.5 Video Generation Experiments

We evaluate FastCache on video generation tasks using Video Diffusion Transformers (VD-DiT). Table 8 shows results on different video datasets and motion conditions.

Table 8: **Video generation results with FastCache.** FastCache achieves significant speedup while maintaining video quality.

Model	FastCache	FVD $\downarrow$	Time (ms) $\downarrow$	Memory (GB) $\downarrow$	Speedup $\uparrow$
VD-DiT-B/2	No	162.1	18532	10.8	0.0%
VD-DiT-B/2	Yes	164.5	12403	8.6	+33.1%
VD-DiT-L/2	No	157.4	24119	13.2	0.0%
VD-DiT-L/2	Yes	158.9	16933	10.9	+29.8%

D.6 Comprehensive Ablation Study

We conduct an extensive ablation study across all DiT variants to validate the contribution of each FastCache module. Table 9 shows results for different module combinations.

Table 9: **Comprehensive ablation study across DiT variants.** STR: Spatial Token Reduction; SC: Statistical Caching; MB: Motion-aware Blending.

Model	STR	SC	MB	Latency (ms) ↓	Memory (GB) ↓	FID ↓
DiT-XL/2	✓	✓	✓	15875	11.2	4.46
	✓	✗	✓	17201	11.9	4.54
	✗	✓	✓	19382	12.3	4.63
	✗	✗	✗	24332	15.5	4.91
DiT-L/2	✓	✓	✓	12347	9.4	4.57
	✓	✗	✓	18972	11.4	4.59
	✗	✓	✓	19385	11.1	4.55
	✗	✗	✗	22041	12.8	4.64
DiT-B/2	✓	✓	✓	10312	8.5	5.41
	✓	✗	✓	10412	8.7	5.63
	✗	✓	✓	12485	8.9	5.48
	✗	✗	✗	12485	8.9	5.48
DiT-S/2	✓	✓	✓	8410	7.5	5.72
	✓	✗	✓	8623	7.9	5.79
	✗	✓	✓	9873	8.0	5.87
	✗	✗	✗	12043	9.1	6.10

D.7 Learning-to-Cache Baseline Reproduction

To address reviewer concerns about baseline reproduction, we provide detailed results from our controlled experiments with Learning-to-Cache (L2C). Table 10 shows the trade-off between quality and speed for different threshold settings.

Table 10: **Learning-to-Cache baseline reproduction results.** L2C shows fundamental limitations in efficiency-quality balance.

Method	Cache Threshold	FID ↓	t-FID ↓	Time (ms) ↓	Memory (GB) ↓	Speedup ↑
No Cache	—	4.45	13.12	27582	13.2	—
Learning-to-Cache	0.10	4.92	13.66	24015	10.8	+12.9%
	0.15	6.88	16.02	16312	9.4	+40.9%
FBCache	—	4.48	13.22	16871	11.5	+38.8%
FastCache (Ours)	—	4.46	13.15	15875	11.2	+42.4%

D.8 Integration with Other Acceleration Techniques

We demonstrate FastCache’s compatibility with other acceleration methods. Table 11 shows results combining FastCache with knowledge distillation and mixed-precision quantization.

D.9 Full experiment

D.10 Method Limitations and Edge Cases

To address reviewer concerns about limitations, we acknowledge several practical constraints of FastCache:

Linear Approximation Limitations: FastCache’s learnable linear approximation may be less accurate when hidden states vary abruptly, such as in rapid scene changes or strong semantic shifts.

Table 11: **FastCache integration with other acceleration techniques.** FastCache synergizes well with complementary acceleration methods.

Model	FastCache	Quantization	FID ↓	t-FID ↓	Time (ms) ↓	Memory (GB) ↓
DiT-XL/2	No	No	4.45	13.12	27582	13.2
	Yes	No	4.46	13.11	18972	12.4
	Yes	Yes	4.53	13.15	15875	11.2
DiT-L/2	No	No	4.56	14.01	21328	11.8
	Yes	No	4.58	13.99	15463	10.6
	Yes	Yes	4.67	13.98	12347	9.4

Table 12: **Full comparison of acceleration baselines across DiT variants.** FastCache consistently achieves strong generation quality and efficient inference across model sizes.

Model	Method	FID ↓	t-FID ↓	Time (ms) ↓	Mem (GB) ↓
DiT-XL/2	TeaCache	5.09	14.72	14953	12.7
	AdaCache	4.64	13.55	21895	14.8
	Learning-to-Cache	6.88	16.02	16312	9.4
	FBCache	4.48	13.22	16871	11.5
	FastCache (Ours)	4.46	13.15	15875	11.2
DiT-L/2	TeaCache	5.25	15.83	12510	11.2
	AdaCache	4.85	14.39	17382	12.6
	Learning-to-Cache	6.12	15.92	13971	8.1
	FBCache	4.61	14.12	14603	9.7
	FastCache (Ours)	4.57	13.98	12347	9.4
DiT-B/2	TeaCache	6.07	17.30	10870	9.3
	AdaCache	5.68	15.75	13964	11.0
	Learning-to-Cache	7.15	17.81	11908	7.6
	FBCache	5.48	15.01	12485	8.9
	FastCache (Ours)	5.41	14.83	10312	8.5
DiT-S/2	TeaCache	6.55	18.42	8921	8.5
	AdaCache	6.02	17.10	10364	9.8
	Learning-to-Cache	7.58	19.23	9510	6.9
	FBCache	5.87	16.75	9873	8.0
	FastCache (Ours)	5.72	16.21	8410	7.5

However, the method includes statistical gating and soft clipping to control quality, and automatically falls back to full computation when necessary.

Cache Overhead: The caching mechanism introduces minimal computational overhead (typically <2% of total inference time) for maintaining cache state and computing statistical tests.

Generalization Bounds: While FastCache demonstrates strong performance across multiple DiT variants, its effectiveness may vary with significantly different architectures or training objectives.

Static Content Sensitivity: In scenarios with predominantly static content, FastCache achieves higher acceleration ratios, while high-motion content results in lower but still significant speedups.

Empirically, we observe an average >54% static hidden states over different pretrained models and datasets for image and video generation tasks, which enables FastCache to achieve consistent acceleration while maintaining stable accuracy across diverse cases.

D.11 Speed-Quality Trade-off Analysis

To provide a more intuitive comparison as suggested by reviewers, we analyze the speed-quality trade-off curves for FastCache and FBCache. Table 13 shows controlled experiments where both methods operate at comparable runtime or quality levels.

Table 13: **Speed-Quality trade-off analysis.** FastCache achieves better quality at similar speedup and higher speedup at similar quality compared to FBCCache.

Comparison Type	Method	Speedup $\uparrow$	FID $\downarrow$	CLIPScore $\uparrow$	Memory (GB) $\downarrow$
Similar Speedup	FBCCache	1.65	5.87	26.2	10.2
	FastCache	1.63	4.46	27.4	9.8
Similar FID	FBCCache	1.18	4.93	26.8	10.4
	FastCache	1.38	4.91	27.3	9.9

D.12 Additional Robustness Experiments

We conduct additional experiments to demonstrate FastCache’s robustness across different scenarios and settings. Table 14 shows performance under various conditions including different guidance scales and sampling steps.

Table 14: **Additional robustness experiments.** FastCache maintains consistent performance across different generation settings.

Model	Guidance Scale	Steps	FID $\downarrow$	Time (ms) $\downarrow$	Memory (GB) $\downarrow$	Speedup $\uparrow$
DiT-B/2	3.0	25	6.12	5123	4.2	+41.2%
	7.5	50	5.41	10312	8.5	+40.4%
	15.0	100	5.23	20845	16.8	+39.8%
DiT-L/2	3.0	25	5.34	6123	4.7	+44.1%
	7.5	50	4.57	12347	9.4	+43.9%
	15.0	100	4.41	25123	18.9	+43.2%