
Foresight: Adaptive Layer Reuse for Accelerated and High-Quality Text-to-Video Generation

Muhammad Adnan*
The University of British Columbia
adnan@ece.ubc.ca

Nithesh Kurella
d-Matrix
nitheshk@d-matrix.ai

Akhil Arunkumar
d-Matrix
aarunkumar@d-matrix.ai

Prashant J. Nair
The University of British Columbia
prashantnair@ece.ubc.ca

Abstract

Diffusion Transformers (DiTs) achieve state-of-the-art results in text-to-image, text-to-video generation, and editing. However, their large model size and the quadratic cost of spatial-temporal attention over multiple denoising steps make video generation computationally expensive. Static caching mitigates this by reusing features across fixed steps but fails to adapt to generation dynamics, leading to suboptimal trade-offs between speed and quality.

We propose Foresight, an adaptive layer-reuse technique that reduces computational redundancy across denoising steps while preserving baseline performance. Foresight dynamically identifies and reuses DiT block outputs for all layers across steps, adapting to generation parameters such as resolution and denoising schedules to optimize efficiency. Applied to OpenSora, Latte, and CogVideoX, Foresight achieves up to $1.63\times$ end-to-end speedup, while maintaining video quality. The source code of Foresight is available at <https://github.com/STAR-Laboratory/foresight>.

1 Introduction

Diffusion Transformers (DiTs) [Bao et al., 2023, Peebles and Xie, 2023] are the state-of-the-art architecture for text-to-image [Chen et al., 2024a] and text-to-video [Brooks et al., 2024, Genmo, 2024, Ma et al., 2024, Liu et al., 2024c] generation. By leveraging the scaling properties of self-attention [Vaswani et al., 2017], DiTs enable high-fidelity video synthesis. However, this comes at a high computational cost. Self-attention has quadratic complexity $\mathcal{O}(L^2)$ in the token length L , which increases with both spatial resolution (more patches per frame) and temporal extent (more frames). Combined with the *tens* of denoising steps in typical diffusion pipelines, this leads to prohibitive inference latency for high-resolution or long-duration videos.

Inference latency in diffusion models is further increased by classifier-free guidance (CFG) [Ho and Salimans, 2022] and full-precision weights. Techniques like pruning [Ma et al., 2023b], quantization [Lin et al., 2024b], distillation [Hsieh et al., 2023], and architectural optimizations [Li et al., 2023b, Zhao et al., 2024c,a, Fang et al., 2023, Bolya and Hoffman, 2023] can reduce model size and compute cost. However, they often require retraining or hardware-specific tuning, making them impractical for deploying large, pretrained DiTs across diverse hardware environments.

Feature caching offers a complementary strategy for accelerating diffusion by reusing intermediate activations across adjacent denoising steps. Early methods for U-Net models cache skip-connection

*A large part of this work was done while the author was interning with d-Matrix.

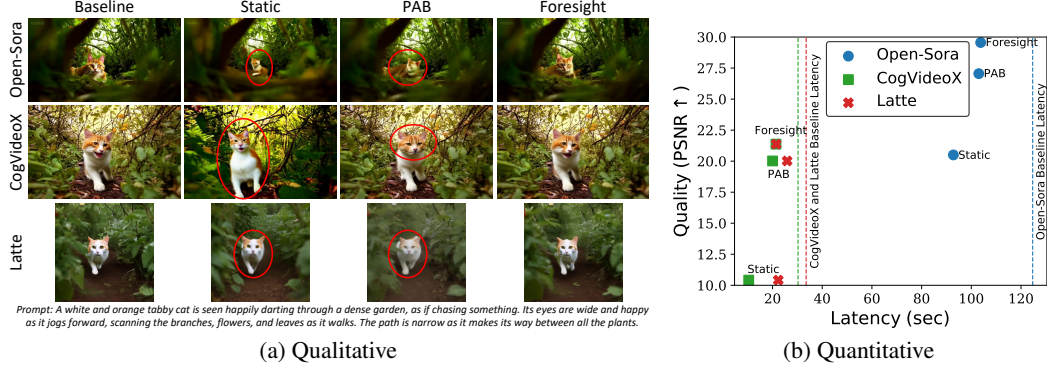


Figure 1: We compare the effectiveness of Adaptive Layer Reuse (Foresight) with prior state-of-the-art static techniques. The **red circle** indicates the frame variations in prior techniques. Videos are generated using OpenSora Zheng et al. [2024] at 720p, Latte Ma et al. [2024] at 512×512, and CogVideoX Yang et al. [2024] at 480×720, all with a fixed length of 2 seconds. We report average inference time on a single A100 GPU using Open-Sora prompts.

outputs between steps t and $t-1$ [Ma et al., 2023a, Wimbauer et al., 2024], while recent work extends this idea to DiTs [Chen et al., 2024b, Liu et al., 2024b, Selvaraju et al., 2024, Yuan et al., 2024, Zou et al., 2024, Kahatapitiya et al., 2024, Liu et al., 2024a]. However, these approaches apply a *static*, uniform reuse policy across all layers and steps, which yields limited speedup and often degrades video quality and temporal coherence. As shown in Figure 1, static methods [Chen et al., 2024b, Liu et al., 2024b, Zhao et al., 2024b] introduce visible artifacts (highlighted in red), driven by a latency-first design that ignores layer- and step-specific reuse dynamics.

To better balance inference speed and generation quality, we analyze reuse patterns in DiT blocks and find that static reuse fails to preserve critical updates. Our analysis reveals two key findings: (1) the potential for reuse varies significantly across both layers and denoising steps, and (2) even minor changes in generation conditions, such as prompt content, resolution, or step count, can dramatically alter activation behavior. Figure 2(Left) quantifies the layer-wise Mean Squared Error (MSE) of spatial DiT features between successive timesteps in OpenSora, revealing pronounced layer-wise heterogeneity. Figure 2(Middle, Right) further highlights resolution-dependent reuse variation within the same layer. Motivated by these observations, we propose Foresight, a *training-free*, per-layer adaptive reuse scheduler. Foresight *dynamically* decides, at runtime, whether to recompute or reuse each block’s activation, thereby providing accelerated inference while maintaining video quality.

Our comprehensive experiments on NVIDIA A100 GPUs for text-to-video generation benchmarks demonstrate the effectiveness of Foresight compared to static reuse baselines on OpenSora, Latte, and CogVideoX. Foresight achieves up to $1.63\times$ end-to-end inference speedup, with no additional training, while preserving video quality.

2 Related Work

2.1 Diffusion Transformers

Diffusion Models (DMs) [Ho et al., 2020, Chen et al., 2024a] generate images and videos by iteratively denoising random noise across multiple timesteps. Early models used convolutional U-Net backbones [Sohl-Dickstein et al., 2015, Rombach et al., 2022], achieving strong results but with limited scalability. Diffusion Transformers (DiTs) [Bao et al., 2023, Peebles and Xie, 2023] replace convolutions with self-attention, offering better scalability and generation quality. DiTs now set state-of-the-art benchmarks in text-to-image [Liu et al., 2024c] and text-to-video generation [Tim Brooks and Ramesh.]. Notably, Sora [Brooks et al., 2024] demonstrated high-quality video synthesis, motivating open-source projects like OpenSora [Zheng et al., 2024], Latte [Ma et al., 2024], CogVideoX [Yang et al., 2024], and Mochi [Genmo, 2024]. However, iterative denoising and spatial-temporal attention introduce high inference latency, limiting practical deployment.

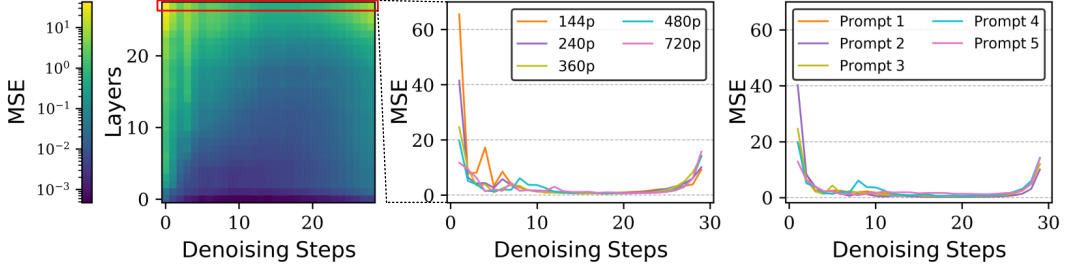


Figure 2: Mean squared error (MSE) between consecutive Spatial DiT block outputs in OpenSora [Zheng et al., 2024] during the generation of a 2-second, 480p video. Lower MSE values indicate greater similarity and thus higher reuse potential. **(Left)** Heatmap across layers and denoising steps reveals distinct, layer-specific reuse patterns. **(Middle)** Variation of MSE for *Layer 28* across different resolutions. **(Right)** Variation of MSE for *Layer 28* across different text prompts. Note that Temporal DiT blocks demonstrate similar reuse trends as Spatial DiT blocks.

2.2 Accelerating Diffusion Transformers

Efforts to accelerate diffusion models fall into two main categories: (1) *reducing denoising steps* and (2) *speeding up the denoising network*. The first reduces inference latency by shortening the sampling schedule. Deterministic methods like DDIM [Song et al., 2020] preserve quality with fewer steps, while higher-order solvers such as DPM-Solvers [Lu et al., 2022a,b] improve efficiency through advanced numerical integration. Rectified Flow [Liu et al., 2022] enhances ODE-based sampling for faster inference. Step distillation [Meng et al., 2023, Salimans and Ho, 2022] compresses multi-step models into fewer-step variants. Consistency Models [Song et al., 2023] enable single-step generation by mapping noise directly to data while maintaining consistency across timesteps.

The second direction accelerates the denoising network itself using techniques such as quantization [Li et al., 2023a, Zhao et al., 2024a, Shang et al., 2023], token reduction [Bolya and Hoffman, 2023, Zhang et al., 2025], knowledge distillation [Li et al., 2023b], and pruning [Wang et al., 2023]. More recently, training-free caching methods have gained attention for reusing intermediate activations across timesteps. Methods like DeepCache [Ma et al., 2023a], BlockCache [Wimbauer et al., 2024], and Faster Diffusion [Li et al., 2024] achieve speedups via feature reuse but are designed for U-Net architectures, limiting their applicability to transformer-based models.

To overcome the limitations of U-Net, recent work adapts caching techniques to Diffusion Transformers (DiTs). Methods like Δ -DiT [Chen et al., 2024b], T-GATE [Liu et al., 2024b], and FORA [Selvaraju et al., 2024] apply caching to image synthesis. For video, PAB [Zhao et al., 2024b] hierarchically broadcasts attention, DiTFastAttn [Yuan et al., 2024] reduces redundancy across condition and timestep dimensions, and ToCA [Zou et al., 2025] selectively caches tokens. TeaCache [Liu et al., 2024a] adds timestep embedding awareness to caching. Other methods [Kahatapitiya et al., 2024, Xia et al., 2025, Lv et al., 2024, Anagnostidis et al., 2025] explore different reuse granularities. However, they all apply static reuse uniformly across layers and steps. Notably, these techniques prioritize speedup often at the cost of degraded output quality.

3 Foresight: From Static to Adaptive Layer Reuse

We begin by outlining background on diffusion models and static caching for inference acceleration. We then analyze the limitations of static reuse in Diffusion Transformers (DiTs), motivating Foresight, a training-free framework for adaptive layer reuse.

3.1 Preliminary

Diffusion models generate images and videos through two complementary processes. A forward process that gradually introduces noise into clean data, and a reverse process that iteratively denoises to reconstruct data conditioned on input prompts. Starting from an initial clean sample $\mathbf{x}_0$, the forward process adds Gaussian noise over T timesteps using a variance schedule $\{\beta_t\}_{t=1}^T$:

$$\mathbf{x}_t = \sqrt{1 - \beta_t} \mathbf{x}_{t-1} + \sqrt{\beta_t} \mathbf{z}_t, \quad \mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad t = 1, \dots, T \quad (1)$$

As $t \rightarrow T$, the distribution of $\mathbf{x}_t$ approaches $\mathcal{N}(\mathbf{0}, \mathbf{I})$. The reverse denoising process seeks to recover the original sample $\mathbf{x}_0$ by iteratively estimating:

$$p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)) \quad (2)$$

where μ_θ and Σ_θ are the mean and variance predicted by a denoising neural network (p) that is parameterized by θ .

State-of-the-art text-to-video models adopt Spatial–Temporal DiT (ST-DiT) layers that alternate spatial and temporal transformer blocks:

$$\text{DiT}_{\text{spatial}} = \{f_{\text{SA}}, f_{\text{CA}}, f_{\text{MLP}}\}, \quad \text{DiT}_{\text{temporal}} = \{f_{\text{TA}}, f_{\text{CA}}, f_{\text{MLP}}\}$$

where f_{SA} and f_{TA} apply spatial and temporal self-attention, and f_{CA} performs cross-attention with text tokens. At timestep t , the input is a token sequence obtained by flattening all $H \times W$ latent patches in each of the F frames:

$$\mathbf{x}_t = \{x_{1:H \times W}^{(f)}\}_{f=1}^F$$

3.2 Static Layer Reuse in Diffusion Models

Recent methods accelerate DiTs via *static caching* [Zhao et al., 2024b, Chen et al., 2024b, Liu et al., 2024b,a, Zou et al., 2024]. For a model with L layers and T denoising steps, let $\mathbf{x}_t^l$ denote the output of layer l at step t , comprising spatial and temporal components $\{\mathbf{x}_{\text{spatial}}^l(t), \mathbf{x}_{\text{temporal}}^l(t)\}$. These activations are stored in a cache $\mathcal{C}$:

$$\mathcal{C}(\mathbf{x}_t^l) \leftarrow \{\mathbf{x}_{\text{spatial}}^l(t), \mathbf{x}_{\text{temporal}}^l(t)\}, \quad l = 1, \dots, L, \quad t = 1, \dots, T \quad (3)$$

All layers are computed and cached at a chosen step t_1 . For the next N timesteps, the model reuses those activations:

$$\mathbf{x}_t^l = \mathcal{C}(\mathbf{x}_{t_1}^l), \quad l = 1, \dots, L, \quad t = t_1 + 1, \dots, t_1 + N \quad (4)$$

This reuse significantly reduces computational costs by skipping redundant calculations across spatial and temporal DiT blocks for N consecutive steps. Once the reuse window expires (at timestep $t_1 + N + 1$), all layers are recomputed, and the cache is refreshed with the latest outputs $\mathcal{C}(\mathbf{x}_{t_1+N+1}^l)$, enabling reuse in subsequent steps.

After step $t_1 + N$, every layer is recomputed, the cache refreshed, and the cycle can repeat. Caching strategies can be applied at different granularities: fine-grained reuse (e.g., individual attention or MLP blocks) or coarse-grained reuse (entire DiT blocks). Prior methods, such as [Zhao et al., 2024b], emphasize fine-grained reuse. In contrast, our approach investigates coarse-grained reuse at the DiT block level, motivated by the observation that adjacent-step features exhibit significant similarity, as illustrated in Figure 2.

3.3 Limitations of Static Reuse

Reuse suitability in text-to-video diffusion varies along three key axes: prompt, layer, and configuration dynamics. Prompts differ in visual complexity. For instance, some prompts induce static scenes while others cause rapid changes. This leads to significant variation in feature similarity across timesteps (Figure 3a). Static reuse applies uniform caching and cannot adapt to such prompt-specific behavior. Layer-wise sensitivity analysis shows that reusing late layers degrades quality most, as these layers exhibit greater feature variation (Figure 3b, Figure 2 (left)), yet static methods fail to account for this variation. Moreover, video configuration parameters such as resolution, length, and denoising schedule can drastically alter reuse patterns, even under the same prompt (Figure 2 (middle); Appendix C). These findings underscore the limitations of static reuse under dynamic generation conditions.

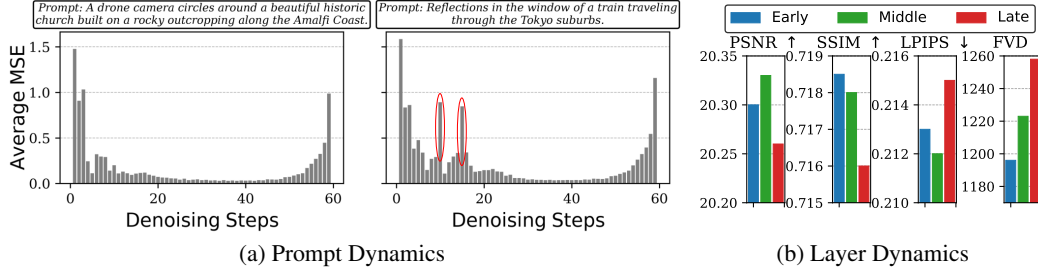


Figure 3: **(a)** Prompt-dependent dynamics of intermediate spatial features (Spatial DiT block outputs) across layers. Prompts featuring rapid scene changes exhibit sharper variations between consecutive steps, indicating higher sensitivity and less reuse potential. The **red circles** indicate these variations. **(b)** Sensitivity analysis across DiT layers grouped into early, middle, and late stages. Applying static reuse ($N = 1$) within each group shows that later-stage layers disproportionately contribute to quality degradation under static caching. All experiments use the OpenSora model for 2-second, 720p videos with prompts selected from the OpenSora prompt set.

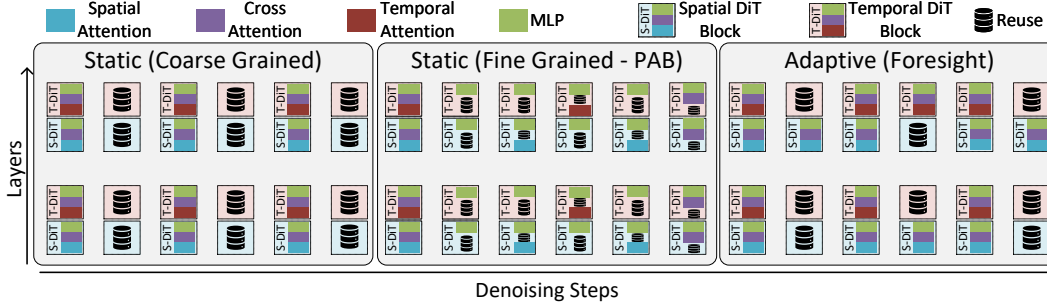


Figure 4: Overview of Foresight as compared to Static Reuse Techniques. Both Coarse-Grained and Fine-Grained [Zhao et al., 2024b] techniques have a preset reuse strategy and do not adapt reuse dynamically with layers and denoising steps.

3.4 Foresight: Adaptive Layer Reuse Framework

To address the limitations of static reuse, we propose Foresight, an adaptive layer reuse framework that balances inference speed and generation quality. Figure 4 shows the overall flow, contrasting Foresight with coarse-grained static caching and fine-grained methods like PAB [Zhao et al., 2024b]. Unlike static approaches, Foresight makes dynamic reuse decisions through a two-phase process: a *warmup phase* and a *reuse phase*, driven by layer-specific reuse thresholds (λ) and reuse metrics (δ).

Warmup Phase: In contrast to prior caching methods, Foresight does not initialize the cache ($\mathcal{C}$) immediately after the first step. Instead, all spatial, temporal, cross-attention, and MLP blocks are computed for the first W denoising steps, allowing intermediate features to stabilize. At timestep $t = W$, the cache is initialized with the latest outputs from each layer as described by Eq. 3.

Foresight also establishes a reuse threshold (λ) for each layer based on the mean squared error (MSE) of intermediate features between consecutive steps during the warmup phase. Let $\mathbf{x}_t$ denote features at step t . Given stabilized features by $t = W$, we compute thresholds as a weighted sum of MSEs from the final three warmup steps, scaled geometrically to reduce bias:

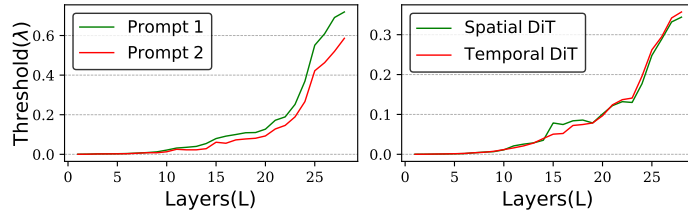


Figure 5: **(Left)** Spatial block thresholds for two different prompts using OpenSora (240p, 2s). **(Right)** Spatial and temporal block thresholds for the same prompt using OpenSora (720p, 2s).

$$\lambda_{\mathbf{x}}^l = \sum_{t=W-2}^W \frac{1}{10^{W-t}} \left(\frac{1}{P} \sum_{i=1}^P (\mathbf{x}_i^l(t) - \mathbf{x}_i^l(t-1))^2 \right), \quad \mathbf{x} \in \{\mathbf{x}_{\text{spatial}}, \mathbf{x}_{\text{temporal}}\}, \quad l = 1, \dots, L \quad (5)$$

Here, $P = H \times W$ for spatial blocks and $P = F$ for temporal blocks. Figure 5 shows the adaptive thresholds, showing variations for different prompts and resolutions. It shows how the reuse threshold varies if the resolution changes from 240p to 720p.

Reuse Phase: The reuse phase employs the initialized cache ($\mathcal{C}$) and thresholds (λ), introducing a dynamic reuse metric (δ) for each layer and block to guide reuse decisions. This phase alternates between reuse and recomputation steps. Specifically, reuse occurs for N steps, after which all layers are recomputed for every R step. The reuse metric is updated based on MSE between the current and cached features at each recomputation step, as defined in Eq 6. This update enables Foresight to adapt and reuse dynamically based on feature changes.

$$\delta_{\mathbf{x}}^l(t) = \frac{1}{P} \sum_{i=1}^P (\mathbf{x}_i^l(t) - \mathcal{C}_i^l(t-1))^2, \quad \mathbf{x} \in \{\mathbf{x}_{\text{spatial}}, \mathbf{x}_{\text{temporal}}\}, \quad l = 1, \dots, L \quad (6)$$

After updating the reuse metric, the cache is refreshed with the latest intermediate features defined in Eq. 3. At the subsequent timestep ($t + 1$), reuse decisions are made by comparing the reuse metric (δ) against the threshold (λ):

$$\mathbf{x}_{t+1}^l = \begin{cases} \mathcal{C}(\mathbf{x}_t^l), & \text{if } \delta_{\mathbf{x}}^l(t) \leq \gamma \lambda_{\mathbf{x}}^l \\ \text{Compute}, & \text{otherwise} \end{cases}, \quad \gamma \in (0, 2], \quad l = 1, \dots, L \quad (7)$$

We introduce a scaling factor (γ) to control the reuse threshold, enabling adjustable trade-offs between speed and quality: higher γ emphasizes quality (less reuse), while lower γ favors speed (more reuse). Algorithm 1 describes the end-to-end flow of the Foresight framework.

Algorithm 1 Foresight

```

1: Inputs: Video tokens  $\mathbf{x}_{H \times W}^F$ , Steps  $T$ , Warmup steps  $W$ , Reuse steps  $N$ , Compute interval  $R$ 
2: Parameters: Reuse threshold  $\lambda$ , scaling factor  $\gamma$ 
3: Initialize cache  $\mathcal{C}_{\mathbf{x}}^l \leftarrow \emptyset$ 
4: for  $t = 1$  to  $W$  do
5:    $\mathbf{x}^l(t) \leftarrow \text{Compute}(\mathbf{x}^l)$ ,  $\forall l = 1, \dots, L$ 
6:    $\lambda_{\mathbf{x}}^l \leftarrow \text{MSE}[\mathbf{x}^l(t), \mathbf{x}^l(t-1)]$ 
7: end for
8:  $\mathcal{C}_{\mathbf{x}}^l \leftarrow \mathbf{x}^l(W)$ ,  $\delta_{\mathbf{x}}^l \leftarrow \lambda_{\mathbf{x}}^l$ 
9: for  $t = W + 1$  to  $T$  do
10:  if  $t \bmod R = 0$  then
11:     $\mathbf{x}^l(t) \leftarrow \text{Compute}(\mathbf{x}^l)$ ,  $\forall l = 1, \dots, L$ 
12:     $\delta_{\mathbf{x}}^l \leftarrow \text{MSE}[\mathbf{x}^l(t), \mathcal{C}_{\mathbf{x}}^l]$ 
13:     $\mathcal{C}_{\mathbf{x}}^l \leftarrow \mathbf{x}^l(t)$ 
14:  else
15:    for  $l = 1$  to  $L$  do
16:      if  $\delta_{\mathbf{x}}^l \leq \gamma \lambda_{\mathbf{x}}^l$  then
17:         $\mathbf{x}^l(t) \leftarrow \mathcal{C}_{\mathbf{x}}^l$ 
18:      else
19:         $\mathbf{x}^l(t) \leftarrow \text{Compute}(\mathbf{x}^l)$ 
20:         $\delta_{\mathbf{x}}^l \leftarrow \text{MSE}[\mathbf{x}^l(t), \mathcal{C}_{\mathbf{x}}^l]$ 
21:         $\mathcal{C}_{\mathbf{x}}^l \leftarrow \mathbf{x}^l(t)$ 
22:      end if
23:    end for
24:  end if
25: end for

```

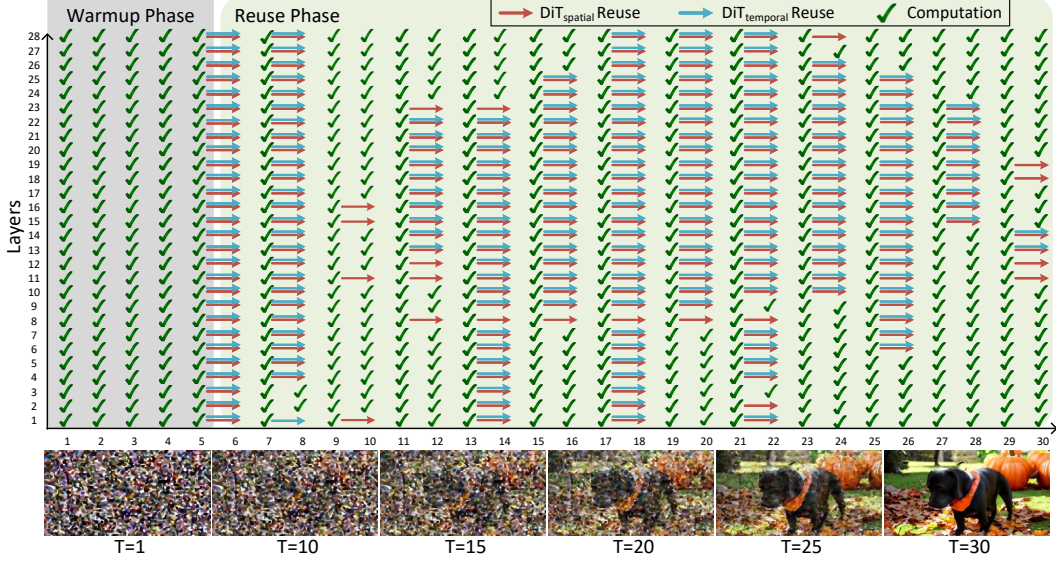


Figure 6: Example of Foresight’s adaptive layer reuse on the OpenSora model (240p, 4s) with a warmup phase of $W = 15\%$, reuse window of $N = 1$, computation of $R = 2$ and $\gamma = 0.5$. The evolution of the video frame is shown at the bottom. The prompt is “A playful black Labrador, adorned in a vibrant pumpkin-themed Halloween costume, surrounded by fallen leaves.”

Intuition with an Example: Figure 6 shows how Foresight works using OpenSora using an example prompt (as showcased in the caption). Checkmarks (✓) indicate computation, arrows (→) represent reuse. During the warmup phase ($W = 15\%$), all blocks are computed, initializing the cache and thresholds (λ) scaled by $\gamma = 0.5$. In the reuse phase, a reuse window of $N = 1$ and recomputation interval $R = 2$ results in alternating reuse and recomputation steps. Notably, later layers (e.g., 24–28) are recomputed more frequently, demonstrating Foresight’s adaptive capability to maintain quality. The video frame evolution is shown at the bottom of Figure 6.

3.5 Foresight: Convergence Analysis

We provide a formal analysis showing that *adaptive reuse*, the key mechanism of Foresight, does not affect model convergence. Let $\mathbf{x}_0$ be the generated (denoised) video, $\mathbf{x}_t$ be the intermediate latent at denoising step t ($T \leq t \leq 1$). For layer l of the spatio-temporal diffusion transformer, denote its output by $f_\theta^l(\mathbf{x}_t)$. During timesteps $T_{\text{reuse}} \subseteq \{1, \dots, T\}$, adaptive reuse approximates this output using the cached value from the previous step as per Eq 8.

$$\tilde{f}_\theta^l(\mathbf{x}_t) = \begin{cases} f_\theta^l(\mathbf{x}_{t-1}) & \text{if } \text{MSE}(\mathbf{x}_t, \mathbf{x}_{t-1}) < \lambda_l \\ f_\theta^l(\mathbf{x}_t) & \text{otherwise} \end{cases}, \quad t = W + 1, \dots, T, \quad l = 1, \dots, L \quad (8)$$

Since reuse preserves weights θ and transition kernels $p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)$, the original diffusion process is guaranteed to hold. Even when recomputed (in the worst case), the baseline forward pass is recovered. For any reused layer l at denoising step t , the error is bounded as shown in Eq 9.

$$\|f_\theta^l(\mathbf{x}_t) - \tilde{f}_\theta^l(\mathbf{x}_t)\|_2 \leq \epsilon_l, \quad \epsilon_l \leq \sqrt{\gamma \lambda_l} \quad (9)$$

Activations vary slowly in space/time when reused, making perturbations negligible. The reuse error at layer l and step t is shown in Eq 10.

$$\epsilon_t^l = \|f_\theta^l(\mathbf{x}_t) - \tilde{f}_\theta^l(\mathbf{x}_t)\|_2 \leq \underbrace{\sqrt{\gamma \lambda_x^l}}_{\epsilon_{\max}^l} \quad (10)$$

Thus, ε_t^l is uniformly bounded by a constant controllable via γ . Let $\mathbf{x}_t^*$ be the baseline latent and $\hat{\mathbf{x}}_t$ the latent under adaptive reuse. After one reverse step term 1 contracts by $\sqrt{1 - \beta_t} < 1$ and term 2 is bounded by $\varepsilon_{\text{tot}} = \sum_l L_l \varepsilon_{\text{max}}^l$ in Eq 11.

$$\|\hat{\mathbf{x}}_{t-1} - \mathbf{x}_{t-1}^*\| \leq \sqrt{1 - \beta_t} \|\hat{\mathbf{x}}_t - \mathbf{x}_t^*\| + \sum_{l=1}^L L_l \varepsilon_t^l \quad (11)$$

Assuming each block is L_l -Lipschitz with $L_l < 1$ and $\varepsilon_t^l \leq \varepsilon_{\text{max}}^l$ for all (t, l) , unrolling the recursion for k steps is shown in Eq 12.

$$\|\hat{\mathbf{x}}_{t-k} - \mathbf{x}_{t-k}^*\| \leq \underbrace{\left(\prod_{s=t-k+1}^t \sqrt{1 - \beta_s} \right) \|\hat{\mathbf{x}}_t - \mathbf{x}_t^*\|}_{(1)} + \underbrace{\varepsilon_{\text{tot}} \sum_{j=0}^{k-1} \prod_{s=t-j+1}^t \sqrt{1 - \beta_s}}_{(2)} \quad (12)$$

1. As $k \rightarrow t$, $\prod_s \sqrt{1 - \beta_s} \rightarrow 0$ since $\sum_s \beta_s \rightarrow \infty$ implies $\prod_s (1 - \beta_s) \rightarrow 0$.
2. Geometric series bounded by $\varepsilon_{\text{tot}} / (1 - \rho)$ where $\rho = \max_s \sqrt{1 - \beta_s} < 1$.

As $t \rightarrow T$ and $T \rightarrow \infty$, (1) vanishes and (2) remains finite. Tightening γ reduces ε_{tot} , yielding arbitrary closeness to the baseline. Adaptive reuse introduces bounded, vanishing perturbations into a contractive reverse Markov chain. Thus, Foresight maintains theoretical fidelity to the original diffusion model.

4 Experiments

4.1 Experimental Setup

Models: We evaluated Foresight on three popular text-to-video models: Open-Sora-v1.2 [Zheng et al., 2024], Latte-1.0 [Ma et al., 2024], and CogVideoX-2b [Yang et al., 2024], using **NVIDIA A100** 80GB GPUs with FlashAttention [Dao et al., 2022] enabled. Open-Sora used rflow [Liu et al., 2022] sampling with 30 denoising steps and a CFG scale of 7.5. Latte and CogVideoX used DDIM [Song et al., 2020] with 50 steps, and CFG scales of 7.5 and 6.0, respectively.

Evaluation Datasets and Metrics: For comprehensive evaluation, we assess Foresight on the VBench benchmark suite [Huang et al., 2024], using VBench accuracy, PSNR, SSIM [Wang and Bovik, 2002], LPIPS [Zhang et al., 2018], and FVD [Unterthiner et al., 2019] to quantify video quality. We report results on the UCF-101 [Soomro et al., 2012] and EvalCrafter [Liu et al., 2024d] prompt sets to evaluate performance on dynamic scenes. Following EvalCrafter, we also include CLIPSIM, CLIP-Temp, and DOVER’s VQA score [Wu et al., 2023] for fine-grained quality assessment.

Baselines: For a comprehensive comparison, we evaluate Foresight against four training-free caching methods for accelerating diffusion models: Static, Δ -DiT, T-GATE, and PAB. Implementation details for each baseline appear in Appendix D.2.

4.2 Results

Quality Comparison: Table 1 compares the video quality of Foresight with static reuse methods. We generate 550 videos from the VBench Huang et al. [2024] prompt set, covering 550 prompts across 11 dimensions (50 per dimension). Video quality is evaluated using VBench accuracy (%) and standard metrics: PSNR, SSIM, LPIPS, and FVD, all relative to the baseline. We report two Foresight configurations, varying reuse steps from $N = 1$ to $N = 2$ and computation intervals from $R = 2$ to $R = 3$, with a fixed scaling factor ($\gamma = 0.5$) balancing speed and quality. Additional results are provided in Appendix E.

Qualitative results further highlight that Foresight surpasses static reuse methods by adaptively selecting which layers to reuse at each denoising step. This strategy preserves critical intermediate features, maintaining video quality and achieving up to $1.63\times$ speedup. Foresight consistently outperforms

Model	Method	VBench(%) $\uparrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	FVD $\downarrow$	Latency(s)	Speedup $\uparrow$
Open-Sora	Baseline	75.63	-	-	-	-	12.65 (± 0.06)	-
	Static	73.10	20.25	0.71	0.21	1257.50	9.27 (± 0.03)	1.36 $\times$
	Δ -DiT	74.20	22.45	0.73	0.18	1009.65	11.95 (± 0.03)	1.06 $\times$
	T-GATE	73.80	21.78	0.71	0.20	890.45	11.1 (± 0.02)	1.14 $\times$
	PAB	75.32	25.67	0.85	0.10	541.53	10.01 (± 0.03)	1.26 $\times$
	Foresight ($N_1 R_2$)	75.90	29.67	0.90	0.05	306.66	9.92 (± 0.92)	1.28 $\times$
	Foresight ($N_2 R_3$)	75.62	27.49	0.87	0.07	457.69	8.80 (± 1.12)	1.44$\times$
Latte	Baseline	73.02	-	-	-	-	33.53 (± 0.09)	-
	Static	61.56	10.41	0.36	0.65	3937.29	25.90 (± 0.09)	1.29 $\times$
	Δ -DiT	65.14	14.85	0.51	0.48	2751.64	32.84 (± 0.02)	1.02 $\times$
	T-GATE	62.48	12.56	0.48	0.56	1894.45	30.54 (± 0.05)	1.09 $\times$
	PAB	72.64	21.22	0.75	0.23	882.96	25.93 (± 0.02)	1.29 $\times$
	Foresight ($N_1 R_2$)	73.08	26.02	0.86	0.12	443.05	28.40 (± 1.36)	1.18 $\times$
	Foresight ($N_2 R_3$)	73.05	23.12	0.82	0.18	683.18	25.54 (± 1.56)	1.31$\times$
CogVideoX	Baseline	77.78	-	-	-	-	30.23 (± 0.03)	-
	Static	76.65	14.05	0.63	0.42	1811.77	22.44 (± 0.01)	1.35 $\times$
	Δ -DiT	54.16	11.94	0.49	0.65	3085.41	29.12 (± 0.01)	1.03 $\times$
	T-GATE	65.14	12.56	0.45	0.64	2845.64	27.64 (± 0.03)	1.09 $\times$
	PAB	77.89	29.04	0.91	0.07	340.24	22.05 (± 0.01)	1.37 $\times$
	Foresight ($N_1 R_2$)	77.94	34.75	0.95	0.03	130.65	20.67 (± 1.10)	1.46 $\times$
	Foresight ($N_2 R_3$)	77.84	28.45	0.87	0.12	531.99	18.53 (± 1.49)	1.63$\times$

Table 1: Qualitative comparison of Foresight and static reuse methods on the VBench prompt set, which spans 11 dimensions with 50 prompts each. Videos are generated at 240p with OpenSora, 512x512 with Latte, and 480x720 with CogVideoX, all with a fixed duration of 2 seconds. Metrics including PSNR, SSIM, LPIPS, and FVD are reported relative to the baseline.

all baselines across varying noise schedulers, denoising steps, and architectures, demonstrating robustness and generalizability.

Memory Overhead:

While caching techniques require no training, they introduce memory overhead from storing intermediate features. This overhead grows with model depth, resolution, and video length. Prior work [Zhao et al., 2024b] employs fine-grained caching across spatial, temporal, cross-

attention, and MLP blocks in all layers, resulting in a cache size of $\mathcal{C} = 6LHWF$, where L is the number of layers, F the number of frames, and each frame has spatial size $H \times W$. In contrast, Foresight adopts a coarser strategy, caching only full DiT blocks (two per layer), reducing the cost to $\mathcal{C} = 2LHWF$. Thus, Foresight provides a $3\times$ higher memory reduction.

4.3 Ablation Studies

We conduct ablation studies to analyze Foresight’s behavior by varying its key hyperparameters and examining the trade-offs between speed and video quality.

Varying Warmup Phase (W): To study the impact of the warmup phase, we vary its length from 5% to 40%, keeping all other Foresight settings fixed. As shown in Figure 7, increasing the warmup percentage reduces the number of denoising steps available for reuse. This leads to higher video quality but limits performance gains, as reuse is applied to fewer steps.

Varying Reuse Settings (N, R): Reuse settings determine the maximum reuse across adjacent timesteps. We conducted an ablation study to analyze their impact by varying the reuse steps and the compute interval. Table 2 summarizes the inference latency and video quality trade-off of Foresight relative to PAB as we vary the reuse window (N) and compute interval (R). Larger N increases reuse

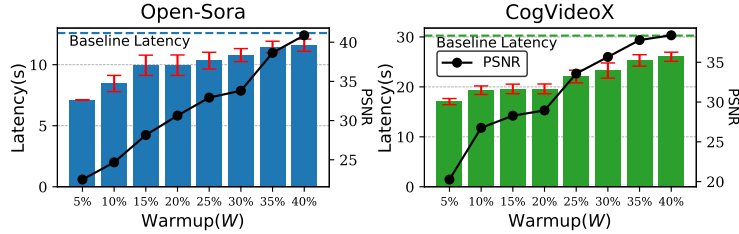


Figure 7: Varying warmup steps (W) with fixed reuse settings: $N = 1$, $R = 2$, and scaling factor $\gamma = 0.5$ in Foresight.

Table 2: Effect of reuse settings on Open-Sora (240p, 2s, $T=60$, $W=15\%$, $\gamma=0.5$). Compared to PAB (Latency = 19.88s, PSNR = 28.12).

Settings	Latency ↓	PSNR ↑
$N=1, R=2$	18.70 (-1.17)	32.38 (+4.26)
$N=2, R=3$	16.37 (-3.50)	30.42 (+2.30)
$N=3, R=4$	14.79 (-5.08)	29.03 (+0.91)
$N=4, R=5$	13.49 (-6.38)	27.91 (-0.20)

Table 3: Effect of scaling factor on Open-Sora ($N=1, R=2$, 240p, 2s, $T=60$, $W=15\%$). Compared to PAB (Latency = 19.88s, PSNR = 28.12).

γ	Latency ↓	PSNR ↑
0.25	20.50 (+0.62)	38.09 (+9.97)
0.5	18.70 (-1.17)	32.38 (+4.26)
1.0	17.03 (-2.84)	30.43 (+2.31)
2.0	16.02 (-3.85)	29.51 (+1.39)

opportunities but lowers the frequency of cache and reuse metric updates, eventually harming fidelity. In practice, we observe that for the range of $N=1$ to $N=3$ delivers the best balance, after which the video quality drops compared to the PAB baseline, mainly due to not updating the reuse metric. For $N=3$, Foresight yields a $1.35\times$ speed-up while surpassing PAB in video quality.

Varying Scaling Factor (γ): The scaling factor γ governs the speed-quality balance by adjusting how strictly reuse is applied. The reuse threshold is dynamic and set at the end of the warmup phase. The scaling factor can be applied uniformly across all layers or adjusted per layer to account for varying sensitivity. Table 3 shows that smaller values increase latency but enhance fidelity. With $\gamma=0.25$, Foresight reaches a PSNR of 39, which is 9.97 points above PAB, at the cost of just a 0.62 s latency increase. This reflects a slight reduction in the reuse profile.

5 Limitations and Discussion

Foresight operates at runtime without altering model architecture or application. Its plug-and-play design enables efficient deployment of text-to-video models, reducing computational costs and expanding access to generative AI for creative and educational purposes. However, broader accessibility may also increase the risk of misuse, such as creating deepfakes for misinformation or fraud. While Foresight does not inherently heighten these risks, its widespread use underscores the need for safeguards and ethical guidelines with controlled use of the model and usage guidelines. Foresight accelerates inference by reusing intermediate activations, its speed and video quality envelope are governed by the reuse window N and warm-up length W . Essentially, a larger N or a smaller W could yield greater speed-up at the risk of quality loss. Furthermore, Foresight can be expanded. Our experiments adopt coarse, block-level caching, yet the framework can operate at finer granularity.

6 Conclusions

We introduced Foresight, a training-free framework that accelerates diffusion transformers for text-to-video generation by making layer-wise reuse decisions at run time. Rather than relying on static caching, Foresight dynamically tracks mean-squared-error (MSE) feature change, maintains per-layer reuse thresholds, and selectively decides whether to reuse or recompute each block. Evaluated on Open-Sora-v1.2, Latte-1.0, and CogVideoX-2b, Foresight achieves up to $1.63\times$ speed-up in inference latency while consistently improving PSNR, SSIM, LPIPS, and FVD compared to state-of-the-art baselines. Its coarse, block-level caching strategy also reduces memory usage by $3\times$, without requiring retraining or architectural modifications.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback and Haoran Qiu for his insightful comments. Muhammad Adnan’s Ph.D. is supported by the Intel TSA and Natural Sciences and Engineering Research Council of Canada (NSERC) [RGPIN-2019-05059] Grants.

References

- Sotiris Anagnostidis, Gregor Bachmann, Yeongmin Kim, Jonas Kohler, Markos Georgopoulos, Artsiom Sanakoyeu, Yuming Du, Albert Pumarola, Ali Thabet, and Edgar Schönfeld. Flexidit: Your diffusion transformer can easily generate high-quality samples with less compute. *arXiv preprint arXiv:2502.20126*, 2025.
- Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *CVPR*, pages 22669–22679, 2023. URL <https://doi.org/10.1109/CVPR52729.2023.02171>.
- Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion, 2023. URL <https://arxiv.org/abs/2303.17604>.
- Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. URL <https://openai.com/research/video-generation-models-as-world-simulators>.
- Junsong Chen, Chongjian Ge, Enze Xie, Yue Wu, Lewei Yao, Xiaozhe Ren, Zhongdao Wang, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-sigma: Weak-to-strong training of diffusion transformer for 4k text-to-image generation, 2024a. URL <https://arxiv.org/abs/2403.04692>.
- 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, 2024b. URL <https://arxiv.org/abs/2406.01125>.
- Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35: 16344–16359, 2022.
- Weichen Fan, Chenyang Si, Junhao Song, Zhenyu Yang, Yinan He, Long Zhuo, Ziqi Huang, Ziyue Dong, Jingwen He, Dongwei Pan, et al. Vchitect-2.0: Parallel transformer for scaling up video diffusion models. *arXiv preprint arXiv:2501.08453*, 2025.
- Gongfan Fang, Xinyin Ma, and Xinchao Wang. Structural pruning for diffusion models, 2023. URL <https://arxiv.org/abs/2305.10924>.
- Genmo. Mochi 1. <https://github.com/genmoai/models>, 2024.
- Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. URL <https://arxiv.org/abs/2207.12598>.
- Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS '20*, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.
- Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*, 2023.
- 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*, pages 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.
- 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.

- Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. Flux.1 kontext: Flow matching for in-context image generation and editing in latent space, 2025. URL <https://arxiv.org/abs/2506.15742>.
- Senmao Li, Taihang Hu, Joost van de Weijer, Fahad Shahbaz Khan, Tao Liu, Linxuan Li, Shiqi Yang, Yaxing Wang, Ming-Ming Cheng, et al. Faster diffusion: Rethinking the role of the encoder for diffusion model inference. *Advances in Neural Information Processing Systems*, 37:85203–85240, 2024.
- 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*, pages 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. *Advances in Neural Information Processing Systems*, 36:20662–20678, 2023b.
- Bin Lin, Yunyang Ge, Xinhua Cheng, Zongjian Li, Bin Zhu, Shaodong Wang, Xianyi He, Yang Ye, Shenghai Yuan, Liuhan Chen, et al. Open-sora plan: Open-source large video generation model. *arXiv preprint arXiv:2412.00131*, 2024a.
- 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, 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. *arXiv preprint arXiv:2411.19108*, 2024a.
- Haozhe Liu, Wentian Zhang, Jinheng Xie, Francesco Faccio, Mengmeng Xu, Tao Xiang, Mike Zheng Shou, Juan-Manuel Perez-Rua, and Jürgen Schmidhuber. Faster diffusion via temporal attention decomposition, 2024b. URL <https://arxiv.org/abs/2404.02747>.
- Lin Liu, Quande Liu, Shengju Qian, Yuan Zhou, Wengang Zhou, Houqiang Li, Lingxi Xie, and Qi Tian. Text-animator: Controllable visual text video generation, 2024c.
- Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.
- Yaofang Liu, Xiaodong Cun, Xuebo Liu, Xintao Wang, Yong Zhang, Haoxin Chen, Yang Liu, Tieyong Zeng, Raymond Chan, and Ying Shan. Evalcrafter: Benchmarking and evaluating large video generation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22139–22149, 2024d.
- Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022a.
- Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022b.
- Zhengyao Lv, Chenyang Si, Junhao Song, Zhenyu Yang, Yu Qiao, Ziwei Liu, and Kwan-Yee K Wong. Fastercache: 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, 2024. URL <https://arxiv.org/abs/2401.03048>.

- Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free, 2023a. URL <https://arxiv.org/abs/2312.00858>.
- 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, 2023b.
- Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14297–14306, 2023.
- William Peebles and Saining Xie. Scalable diffusion models with transformers, 2023. URL <https://arxiv.org/abs/2212.09748>.
- Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.
- Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.
- Pratheba Selvaraju, Tianyu Ding, Tianyi Chen, Ilya Zharkov, and Luming Liang. Fora: Fast-forward caching in diffusion transformer acceleration, 2024. URL <https://arxiv.org/abs/2407.01425>.
- Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. Post-training quantization on diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1972–1981, 2023.
- Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015.
- Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.
- Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models, 2023. URL <https://arxiv.org/abs/2303.01469>.
- Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.
- Connor Holmes Will DePue Yufei Guo Li Jing David Schnurr Joe Taylor Troy Luhman Eric Luhman Clarence Ng Ricky Wang Tim Brooks, Bill Peebles and Aditya Ramesh. Video generation models as world simulators. URL <https://openai.com/index/video-generation-models-as-world-simulators/>.
- Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. Fvd: A new metric for video generation. 2019.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.
- Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Fei Wu, Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wenting Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You

- Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.
- Zhendong Wang, Yifan Jiang, Huangjie Zheng, Peihao Wang, Pengcheng He, Zhangyang Wang, Weizhu Chen, Mingyuan Zhou, et al. Patch diffusion: Faster and more data-efficient training of diffusion models. *Advances in neural information processing systems*, 36:72137–72154, 2023.
- Zhou Wang and Alan C Bovik. A universal image quality index. *IEEE signal processing letters*, 9(3): 81–84, 2002.
- Felix Wimbauer, Bichen Wu, Edgar Schoenfeld, 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. URL <https://arxiv.org/abs/2312.03209>.
- Haoning Wu, Erli Zhang, Liang Liao, Chaofeng Chen, Jingwen Hou, Annan Wang, Wenxiu Sun, Qiong Yan, and Weisi Lin. Exploring video quality assessment on user generated contents from aesthetic and technical perspectives. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20144–20154, 2023.
- Yifei Xia, Suhan Ling, Fangcheng Fu, Yujie Wang, Huixia Li, Xuefeng Xiao, and Bin Cui. Training-free and adaptive sparse attention for efficient long video generation. *arXiv preprint arXiv:2502.21079*, 2025.
- Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.
- Zhihang Yuan, Hanling Zhang, Pu Lu, Xuefei Ning, Linfeng Zhang, Tianchen Zhao, Shengen Yan, Guohao Dai, and Yu Wang. Ditfastattn: Attention compression for diffusion transformer models, 2024. URL <https://arxiv.org/abs/2406.08552>.
- Evelyn Zhang, Jiayi Tang, Xuefei Ning, and Linfeng Zhang. Training-free and hardware-friendly acceleration for diffusion models via similarity-based token pruning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 9878–9886, 2025.
- 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*, pages 586–595, 2018.
- Tianchen Zhao, Tongcheng Fang, Enshu Liu, Rui Wan, Widyadewi Soedarmadji, Shiyao Li, Zinan Lin, Guohao Dai, Shengen Yan, Huazhong Yang, Xuefei Ning, and Yu Wang. Vedit-q: Efficient and accurate quantization of diffusion transformers for image and video generation, 2024a. URL <https://arxiv.org/abs/2406.02540>.
- Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-time video generation with pyramid attention broadcast, 2024b. URL <https://arxiv.org/abs/2408.12588>.
- Yang Zhao, Yanwu Xu, Zhisheng Xiao, Haolin Jia, and Tingbo Hou. Mobilediffusion: Instant text-to-image generation on mobile devices. In *European Conference on Computer Vision*, pages 225–242. Springer, 2024c.
- Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all, March 2024. URL <https://github.com/hpcaitech/Open-Sora>.
- Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching, 2024. URL <https://arxiv.org/abs/2410.05317>.
- Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching, 2025. URL <https://arxiv.org/abs/2410.05317>.

NeurIPS Paper Checklist

1. Claims

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

Answer: [\[Yes\]](#)

Justification: The main claim is that static reuse methods are limited, and adaptive reuse is needed to handle the dynamic nature of reuse, as shown both qualitatively and quantitatively.

Guidelines:

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

2. Limitations

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

Answer: [\[Yes\]](#)

Justification: Section 5 discuss the limitations of Foresight.

Guidelines:

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

3. Theory assumptions and proofs

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

Answer: [\[Yes\]](#)

Justification: Section 3.5 provides the assumptions along with theoretical convergence analysis.

Guidelines:

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

4. Experimental result reproducibility

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

Answer: [\[Yes\]](#)

Justification: We provide a detailed implementation of Foresight, including the hyperparameters used in each experiment. All experiments use publicly available datasets and models, ensuring reproducibility. The source code is included in the supplementary material.

Guidelines:

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

5. Open access to data and code

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

Answer: [Yes]

Justification: We provide a detailed implementation of Foresight, including the hyperparameters used in each experiment. All experiments use publicly available datasets and models, ensuring reproducibility. The source code is included in the supplementary material.

Guidelines:

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

6. Experimental setting/details

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

Answer: [Yes]

Justification: Section 4.1 outlines the experimental setup used for all results. Each result includes its corresponding hyperparameters.

Guidelines:

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

7. Experiment statistical significance

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

Answer: [Yes]

Justification: For all experiments, we report the mean and standard deviation of inference time across over 500 prompts, as shown in Table 1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

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

8. Experiments compute resources

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

Answer: [Yes]

Justification: We present the resources used for experiments in Section 4.1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

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

Answer: [Yes]

Justification: This paper conforms with the NeurIPS code of ethics in every respect.

Guidelines:

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

10. Broader impacts

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

Answer: [Yes]

Justification: Section 5 discusses both positive and negative societal impacts related with efficient deployment of video generation models.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.

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

11. Safeguards

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

Answer: [\[Yes\]](#)

Justification: Section 5 discusses the potential safeguards associated with wide deployment of video generation models.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

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

Answer: [\[Yes\]](#)

Justification: All the models and datasets are properly cited.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

Answer: [\[Yes\]](#)

Justification: We have included the source code as an anonymized zip file in the supplementary material.

Guidelines:

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

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

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

Answer: [\[NA\]](#)

Justification: Paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

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

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

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

Answer: [\[NA\]](#)

Justification: Paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

Answer: [NA]

Justification: Foresight’s plug and play framework does not involve LLMs.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

Appendix

A Background

The success of Sora Brooks et al. [2024] has shown the strong potential of diffusion transformers for text-to-video generation, inspiring a wave of open-source models such as Open-Sora [Zheng et al., 2024], Open-Sora-Plan [Lin et al., 2024a], Latte [Ma et al., 2024], CogVideoX [Yang et al., 2024], Vchitect [Fan et al., 2025], and Mochi [Genmo, 2024].

Figure 8 shows a high-level overview of DiT-based text-to-video models. These models take as input Gaussian noise-initialized latent video frames, timesteps for scheduling, and a text prompt. They use Spatial-Temporal DiT (ST-DiT) blocks: Spatial-DiT captures intra-frame spatial structure, while Temporal-DiT captures inter-frame temporal dynamics. Cross-attention injects text conditioning into each block, and feedforward networks (FFNs) apply learned nonlinearities. Timestep embeddings guide denoising across both spatial and temporal layers.

B Workload Characterization

To analyze inference bottlenecks in text-to-video generation, we profile the Open-Sora [Zheng et al., 2024] model across various resolutions and timeframes, using a fixed 30-step RFlow scheduler using a single NVIDIA A100 (80GB) GPU with flash attention [Dao et al., 2022].

Figure 9 shows the end-to-end latency and its breakdown by operator type. As resolution increases from 480p to 720p, latency rises $2.5\times$, driven largely by the quadratic complexity of attention operations. Attention modules (spatial, temporal, and cross-attention) account for 50% of inference time, FFNs for 15%, and non-linear layers—LayerNorm, scaling, and residuals—for 35%. The sizable cost of non-linear operations highlights the need to target them in optimization efforts.

To identify system bottlenecks during inference, we measured compute and memory throughput for Spatial and Temporal Attention blocks in the Open-Sora [Zheng et al., 2024] model using a single NVIDIA A100 (80GB) GPU with batch size 1 (Figure 10). For Spatial Attention, we varied resolution from 144p to 1080p with a fixed timeframe of 8 seconds. For Temporal Attention, we varied timeframes from 2 to 16 seconds at a fixed resolution of 720p.

In Spatial Attention, increasing resolution increases the number of spatial tokens, leading to higher compute demands due to the quadratic complexity of attention. This keeps the block compute-bound. In contrast, Temporal Attention sees a smaller increase in sequence length with longer timeframes, while the batch size—equal to the number of token patches in a frame—remains large. As a result, flash attention becomes suboptimal and performance becomes memory-bound. Overall, ST-DiT layers are primarily compute-bound, suggesting that reusing redundant computations can improve inference efficiency.

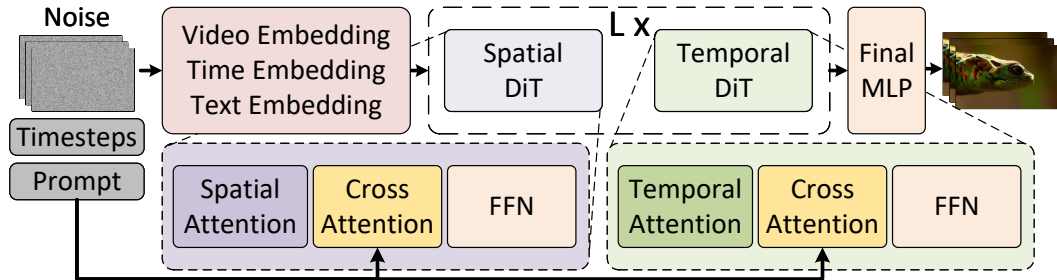


Figure 8: Overview of diffusion transformer (DiT) based text-to-video generation models. It comprises of mainly spatial and temporal diffusion transformer blocks with spatial and temporal attention. Cross attention takes into account prompt conditioning for generated video.

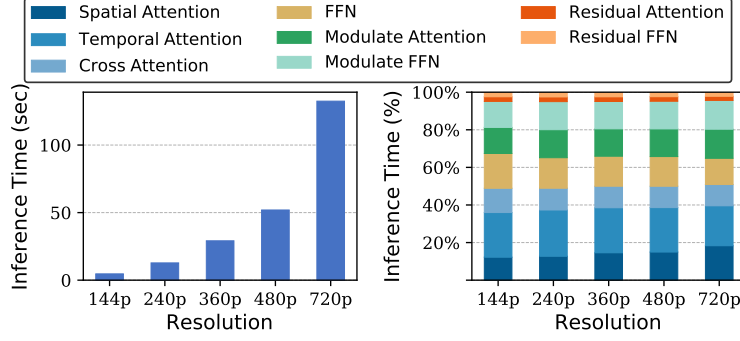


Figure 9: **Left:** End-to-end inference time across resolutions. **Right:** Inference time breakdown by operator. Results based on Open-Sora [Zheng et al., 2024] using a single NVIDIA A100 (80GB) GPU with batch size 1.

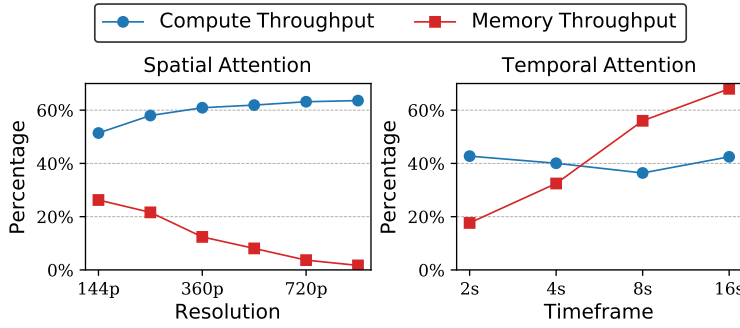


Figure 10: Compute vs. memory throughput for Spatial and Temporal Attention blocks in Open-Sora [Zheng et al., 2024], measured on a single NVIDIA A100 (80GB) GPU with batch size 1.

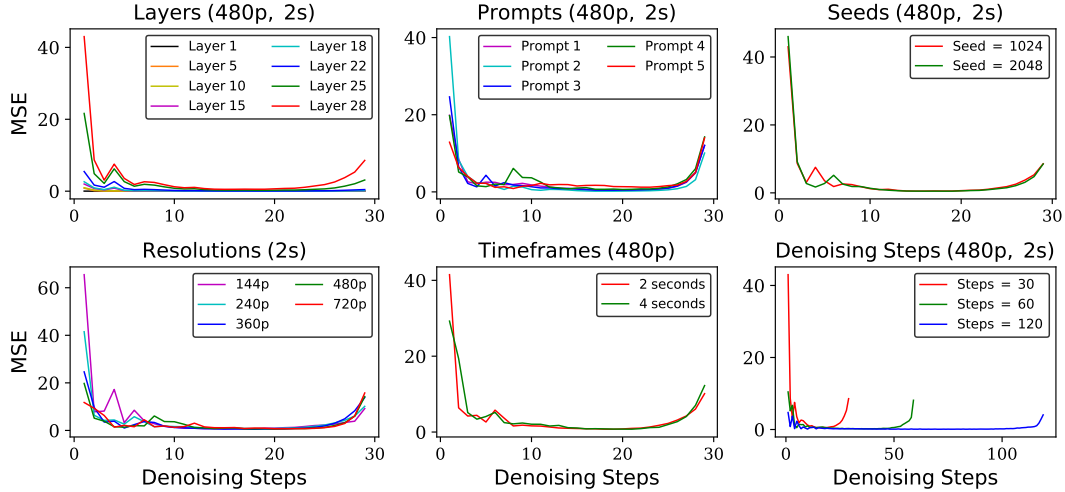


Figure 11: Quantitative analysis of Spatial DiT output using mean squared difference across different layer, prompts, seeds, video resolutions, video timeframes and denoising steps for OpenSora Zheng et al. [2024] model Layer 28 if layer not specified. The prompt is “A narrow, cobblestone alleyway, bathed in the soft glow of vintage street lamps, stretches between tall, weathered brick buildings adorned with ivy. The scene begins with a gentle drizzle, creating a reflective sheen on the cobblestones. As the camera pans, a black cat with piercing green eyes darts across the path, adding a touch of mystery. The alley is lined with quaint, shuttered windows and wooden doors, some slightly ajar, hinting at hidden stories within. A soft breeze rustles the leaves of potted plants and hanging flower baskets, while distant, muffled sounds of city life create a serene yet vibrant atmosphere.”

C Feature Variations and Reuse Metric

To analyze feature variation across video configurations, we vary one parameter at a time while keeping others fixed, and measure changes in intermediate features at specific layers. Figure 11 shows the effect of varying prompts, noise seeds, resolution, timeframes, and denoising steps. The results indicate that intermediate features are sensitive to these configurations. Therefore, adaptive reuse must account for such variations to minimize quality loss in video generation.

C.1 Reuse Metric

To analyze the dynamic behavior of reuse and intermediate feature variation, we measure how features evolve across layers and timesteps using cosine similarity.

C.1.1 Across Condition

Figure 12 illustrates the evolution of spatial features during conditional generation, across layers and denoising steps.

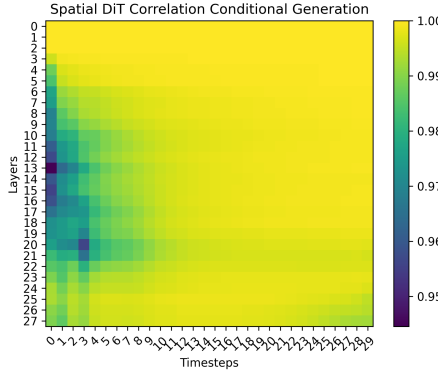


Figure 12: Cosine similarity of Spatial-DiT features across conditioning steps in the OpenSora model.

C.1.2 Across Layers

Figure 13 shows the cosine similarity of spatial-DiT features across layers and denoising steps for the OpenSora model.

C.1.3 Across Denoising Steps

Figure 14 shows the cosine similarity of Spatial-DiT features across denoising steps for different layers of the OpenSora model. Later layers exhibit greater feature variation than early and middle layers.

D Evaluation

D.1 Evaluation Metrics

We evaluate video generation quality using established metrics that assess both perceptual quality and similarity to baseline outputs. Specifically, we use VBench [Huang et al., 2024], a comprehensive benchmark capturing multiple perceptual dimensions, and complement it with similarity metrics comparing generated videos to baseline outputs without reuse.

VBench [Huang et al., 2024] is a video generation benchmark that evaluates model quality across 16 well-defined dimensions. It uses 11 prompt categories, each designed to probe specific aspects of these dimensions, with weighted scores assigned to each. For evaluation, we select the first 50 prompts from each category, totaling 550 prompts.

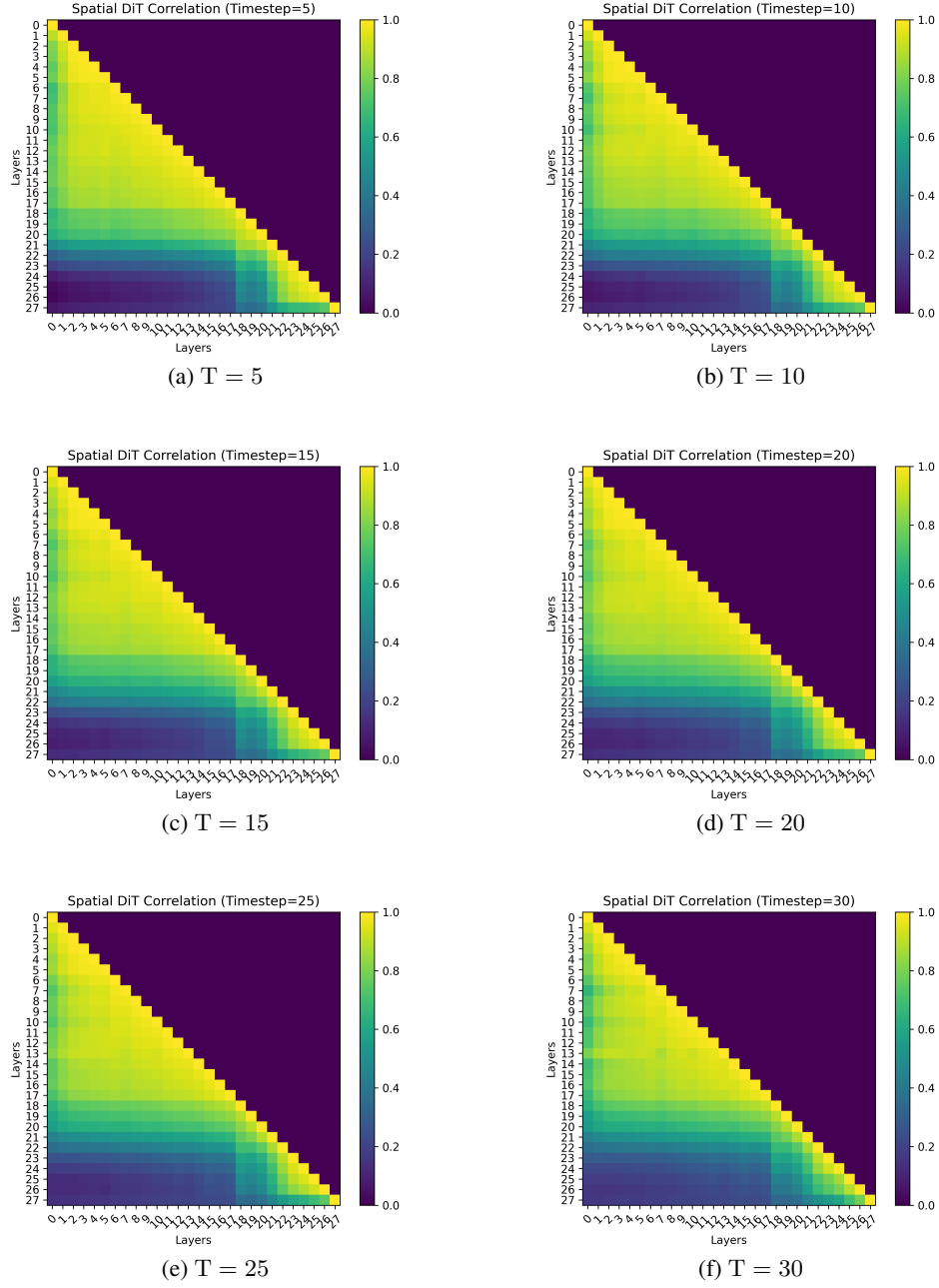


Figure 13: Cosine similarity of Spatial-DiT features across layers at different denoising steps in the OpenSora model.

PSNR measures pixel-level mean squared error between a baseline video (without reuse) and a reused-version. Higher PSNR indicates lower error and better quality.

SSIM [Wang and Bovik, 2002] measures image similarity by comparing structural information, luminance, and contrast between two images.

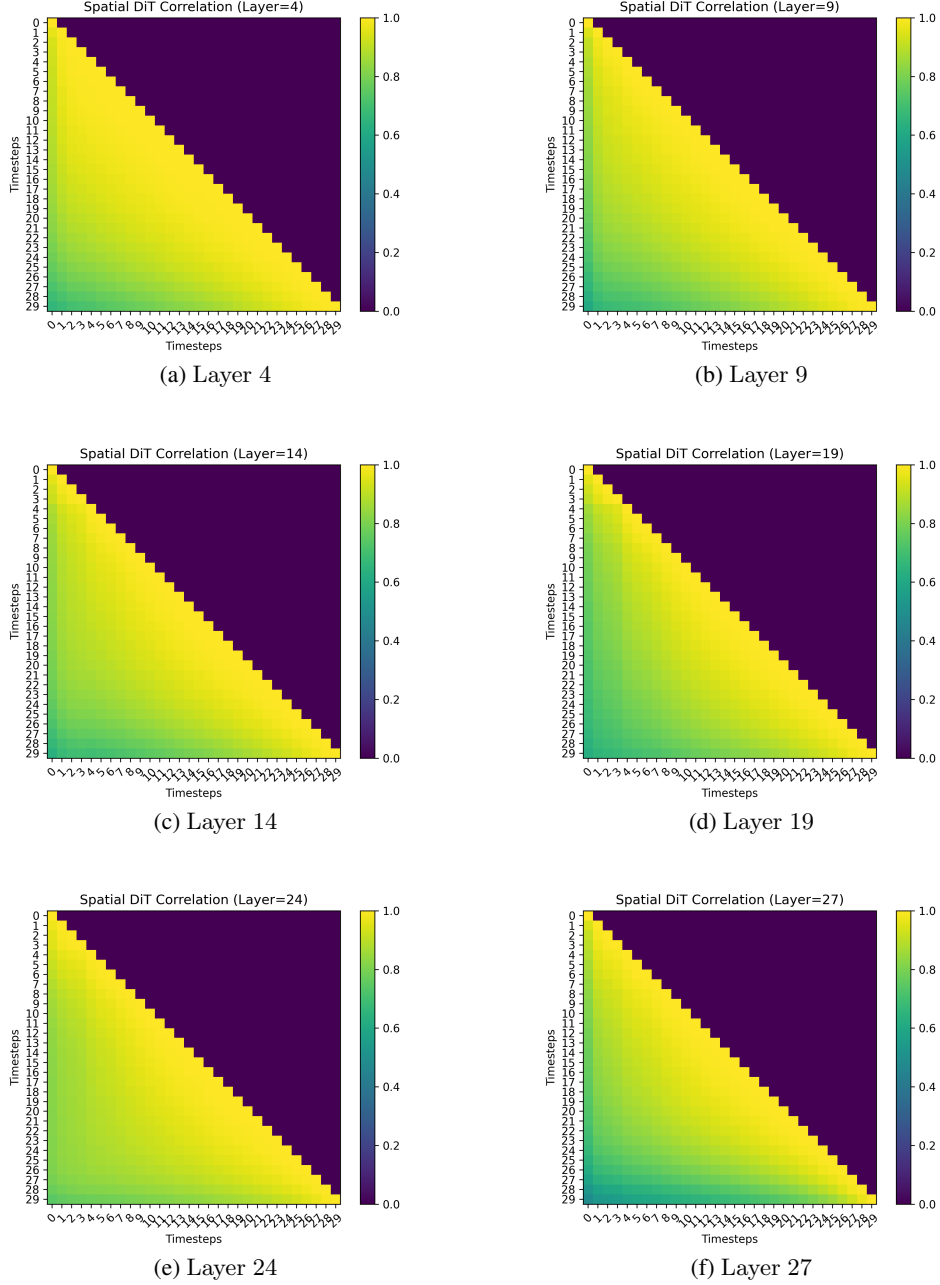


Figure 14: Cosine similarity of Spatial-DiT features across denoising steps for different layers of the OpenSora model.

LPIPS [Zhang et al., 2018] measures perceptual similarity between images using deep neural network features, capturing differences more effectively than pixel-based metrics. It computes the distance between features extracted from pre-trained models trained on large-scale datasets.

FVD [Unterthiner et al., 2019] based on the Fréchet Inception Distance (FID) for images, extends the concept to videos. It measures the distance between real and generated video distributions in the feature space of a pretrained network. By capturing both spatial quality and temporal consistency, FVD reflects how closely generated videos match real ones in appearance and motion.

We compute all the above similarity metrics per frame and report the average across all frames as the final video score.

D.2 Baselines Generation Settings

We evaluate four static cache reuse methods—Static, Δ -DiT[Chen et al., 2024b], T-GATE[Liu et al., 2024b], and PAB[Zhao et al., 2024b]—using the configurations detailed below.

Static method caches coarse-grained features using a reuse window of size N and updates the cache at a fixed compute interval R , as shown in Table 4.

Table 4: Static baseline Settings

Static	Denoising Steps (T)	Reuse Window (N)	Compute Interval (R)
Open-Sora	30	1	2
Latte	50	1	2
CogVideoX	50	1	2

Δ -DiT caches feature map deviations instead of full feature maps. It applies to back blocks during early outline generation and to front blocks during the detail refinement stage. Cache reuse is controlled by a gating hyperparameter b , which defines the boundary between front and back blocks, and a cache interval k as shown in Table 5.

Table 5: Δ -DiT baseline Settings

Δ -DiT	Denoising Steps (T)	Cache Interval (k)	Gate Step (b)	Block Range
Open-Sora	30	2	25	[0, 5]
Latte	50	2	48	[0, 2]
CogVideoX	50	2	48	[0, 2]

T-GATE divides the diffusion process into two phases: semantics planning and fidelity improvement, with the transition defined by gate step m . During the semantics-planning phase, cross-attention (CA) remains active, while self-attention (SA) is computed and reused every k steps after an initial warm-up. After step m , CA is replaced by cached features, while SA continues as shown in Table 6.

Table 6: T-GATE baseline Settings

T-GATE	Denoising Steps (T)	Cache Interval (k)	Gate Step (b)
Open-Sora	30	2	12
Latte	50	2	20
CogVideoX	50	2	20

PAB employs Pyramid Attention Broadcast, where the broadcast range forms a hierarchy: cross-attention (γ) at the base, temporal attention (β) in the middle, and spatial attention (α) at the top. Reuse occurs during designated broadcast timesteps. Each DiT block’s MLP follows a separate reuse schedule, as detailed in Table 7, based on empirical evaluation.

Table 7: PAB baseline Settings

PAB	Denoising Steps (T)	Broadcast Range and Timesteps				Reuse	MLP Broadcast	
		Spatial (α)	Temporal (β)	Cross (γ)	Timesteps		Blocks	Timesteps
Open-Sora	30	2	4	6	[930-450]	2	[0, 1, 2, 3, 4]	[864, 788, 676]]
Latte	50	2	4	6	[800-100]	2	[0, 1, 2, 3, 4]	[720, 640, 560, 480, 400]]
CogVideoX	50	2	-	-	[850, 100]	-	-	-

E Extended Results

E.1 Adaptive behavior for different prompts

To quantify the adaptive behavior of Foresight, we plot absolute latency for all methods across prompts from the Open-Sora set. As shown in Figure 15, static reuse methods exhibit consistent latency due to fixed reuse schedules. In contrast, Foresight balances speed and quality based on scene complexity, enabling dynamic reuse for improved video quality and inference speedup. Table 15 shows the VBench wall clock time for 500 prompts for OpenSora model for a more realistic comparison.

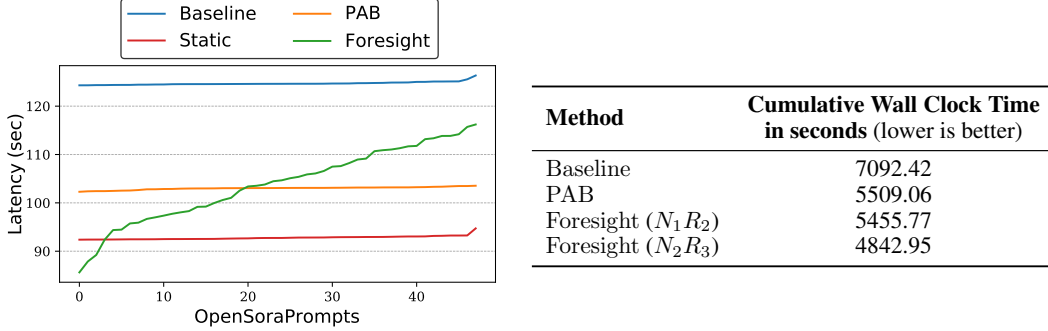


Figure 15: **Left:** Latency variation across prompts from the Open-Sora set. Baseline and Static methods yield constant inference latency due to fixed reuse schedules. In contrast, Foresight adapts reuse based on prompt complexity, improving video quality with dynamic latency. Prompts are sorted by latency (ascending) using the Open-Sora model at 720p for 2-second generations. **Right:** Open-Sora cumulative wall clock time in seconds for VBench prompt set 500 prompts.

E.2 Additional results

To assess the efficacy of Foresight, we evaluate it on state-of-the-art models HunyuanVideo [Kong et al., 2024] and Wan-2.1 [Wan et al., 2025] using the PenguinVideoBenchmark and compare it with TeaCache [Liu et al., 2024a]. As shown in Table 8, Foresight consistently outperforms prior work in quality while delivering competitive speedups. Unlike static methods, Foresight adapts to balance speed and quality. For fairness, we also report performance for the Foresight configuration that matches the quality of prior work.

Model	Method	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	FVD $\downarrow$	Latency(s)	Speedup $\uparrow$
HunyuanVideo	Baseline	-	-	-	-	415.94 (± 1.04)	-
	TeaCache(0.1)	37.31	0.96	0.02	170.16	258.08 (± 3.28)	1.61 $\times$
	Foresight ($N_1 R_2$)	41.79	0.97	0.01	47.45	256.75 (± 6.54)	1.62 $\times$
	Foresight ($N_2 R_3$)	37.25	0.96	0.02	172.85	231.08 (± 7.42)	1.80 $\times$
Wan-2.1	Baseline	-	-	-	-	74.87 (± 0.45)	-
	TeaCache(0.1)	21.49	0.77	0.14	1222.56	38.9 (± 0.25)	1.90 $\times$
	Foresight ($N_1 R_2$)	25.12	0.86	0.08	543.40	50.93 (± 1.14)	1.47 $\times$
	Foresight ($N_2 R_3$)	21.64	0.77	0.15	1234.48	33.57 (± 1.32)	2.23 $\times$

Table 8: Qualitative comparison of Foresight with recent related work and state-of-the-art models on the PenguinVideoBenchmark. Results are shown for 10 randomly sampled prompts: 720p videos of 2 seconds with HunyuanVideo, and 480p videos of 5 seconds with Wan-2.1.

E.3 Performance comparison with same generation quality

Foresight uses fixed hyperparameters across all experiments and adapts consistently across models, prompts, and configurations. These parameters provide a practical way to balance quality and performance without additional tuning. By contrast, prior work [Zhao et al., 2024b, Liu et al., 2024a]

requires model-specific adjustments: PAB fixes MLP reuse empirically, while TeaCache thresholds vary by model. For a fair comparison, Table 9 reports speedups for both methods using a Foresight configuration that matches PAB’s output quality across models.

Table 9: Performance improvement measured at matched video quality

Model	PSNR	PAB Speedup	Foresight Speedup
OpenSora	25.67	1.26×	1.68×
Latte	21.22	1.29×	1.58×
CogVideoX	29.04	1.37×	1.95×

E.4 Additional results across other benchmarks and prompts set

To complement Section 4.2, which reports results using VBench [Huang et al., 2024] prompts and standard similarity metrics, we further evaluate Foresight on two additional prompt sets: UCF-101 [Soomro et al., 2012] and EvalCrafter [Liu et al., 2024d]. We include CLIPSIM and CLIP-Temp to assess text-video alignment and temporal consistency, along with DOVER’s [Wu et al., 2023] VQA metrics for both aesthetic and technical quality, as shown in Table 10.

E.5 Extension to other diffusion tasks

To evaluate the efficacy of Foresight across other diffusion tasks, we apply Foresight to the FLUX [Labs et al., 2025] text-to-image generation model, which does not include a temporal dimension. In this setting, Foresight reuses spatial and cross-attention layers and MLPs.

Model	Method	CLIP SIM	CLIP Temp	VQA Aesthetic	VQA Technical	VQA Overall	Latency(s)	Speedup
UCF-101 [Soomro et al., 2012] (101 Prompts)								
Open-Sora	Baseline	20.51	99.64	14.56	25.02	19.75	12.71 (± 0.04)	-
	PAB	20.47	99.76	6.57	15.66	10.32	10.05 (± 0.06)	1.26×
	Foresight ($N_1 R_2$)	20.73	99.88	15.17	26.68	21.11	9.72 (± 0.84)	1.30×
	Foresight ($N_2 R_3$)	20.70	99.86	14.01	26.11	20.32	7.32 (± 0.80)	1.73×
Latte	Baseline	20.09	99.39	20.71	12.93	14.10	32.48 (± 0.02)	-
	PAB	20.09	99.39	20.71	12.69	13.79	25.11 (± 0.01)	1.29×
	Foresight ($N_1 R_2$)	20.12	99.14	22.72	21.04	20.34	21.48 (± 0.05)	1.51×
	Foresight ($N_2 R_3$)	20.09	99.16	21.95	19.14	18.92	15.56 (± 0.22)	2.08×
CogVideoX	Baseline	20.22	99.43	40.25	42.34	41.19	29.35 (± 0.02)	-
	PAB	20.20	99.43	39.44	41.86	40.52	22.65 (± 0.05)	1.29×
	Foresight ($N_1 R_2$)	20.21	99.42	40.64	40.65	40.17	19.57 (± 0.96)	1.49×
	Foresight ($N_2 R_3$)	20.17	99.44	41.46	43.34	42.15	17.00 (± 1.18)	1.72×
EvalCrafter [Liu et al., 2024d] (150 prompts)								
Open-Sora	Baseline	20.07	99.55	17.48	29.94	23.49	12.59 (± 0.09)	-
	PAB	19.91	99.76	9.16	20.82	14.15	10.07 (± 0.03)	1.24×
	Foresight ($N_1 R_2$)	20.05	99.54	16.85	29.56	23.10	9.36 (± 0.77)	1.34×
	Foresight ($N_2 R_3$)	20.05	99.50	15.67	29.21	22.24	7.06 (± 0.93)	1.78×
Latte	Baseline	20.69	99.50	53.05	45.59	48.37	32.48 (± 0.02)	-
	PAB	19.98	99.65	53.35	35.39	40.84	25.11 (± 0.01)	1.29×
	Foresight ($N_1 R_2$)	20.60	99.51	55.85	45.80	49.27	28.37 (± 1.14)	1.14×
	Foresight ($N_2 R_3$)	20.54	99.50	54.00	44.03	47.29	25.59 (± 1.36)	1.26×
CogVideoX	Baseline	20.66	99.51	54.72	57.67	56.53	30.43 (± 0.12)	-
	PAB	20.66	99.51	52.49	56.10	54.85	22.95 (± 0.07)	1.32×
	Foresight ($N_1 R_2$)	20.66	99.55	53.34	56.73	55.64	25.80 (± 0.84)	1.17×
	Foresight ($N_2 R_3$)	20.64	99.55	52.54	56.34	54.98	24.56 (± 1.08)	1.23×

Table 10: Qualitative comparison of Foresight and PAB on UCF [Soomro et al., 2012] and EvalCrafter [Liu et al., 2024d] prompts set. Videos are generated at 240p with OpenSora, 512x512 with Latte, and 480x720 with CogVideoX, all with a fixed duration of 2 seconds. Metrics including EvalCrafter’s Liu et al. [2024d] CLIP score and DOVER’s [Wu et al., 2023] VQA in Aesthetics, Technical and Overall score.

Table 11 reports the performance, showing that Foresight reduces wall-clock time by approximately $2\times$ while maintaining comparable image quality to the baseline.

Table 11: Open-Sora cumulative wall clock time in seconds

Method	Latency (seconds)
Baseline	14.02
Foresight ($N_1 R_2$)	7.1

E.6 Qualitative Comparison

Figures 16 and 17 present a qualitative comparison of videos generated by Foresight and the baseline across multiple frames. Foresight maintains consistent video quality while achieving notable performance improvements.



Figure 16: Qualitative Comparison for Open-Sora [Zheng et al., 2024] with 720p, 2sec video generation across multiple frames.

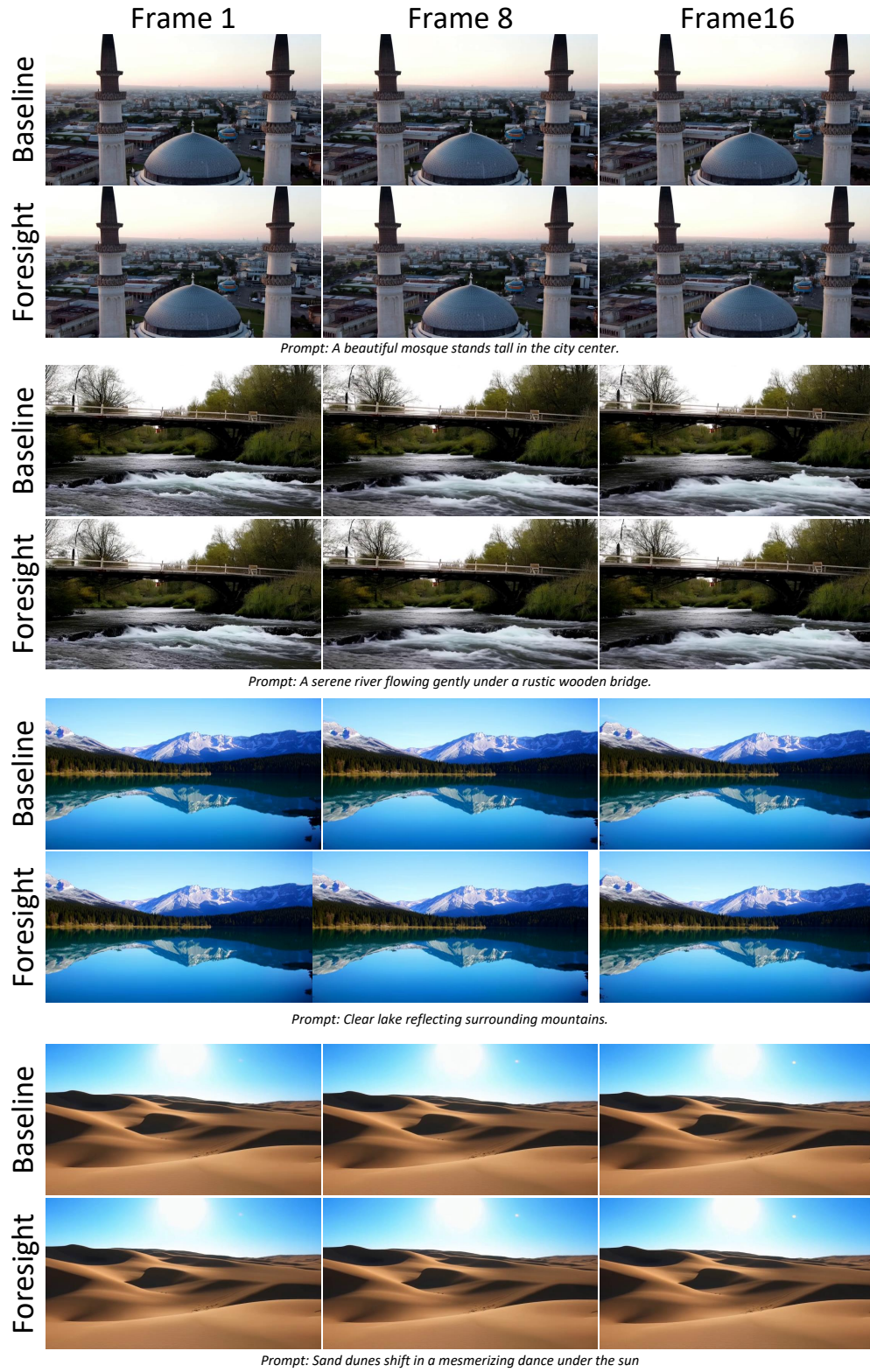


Figure 17: Qualitative Comparison for CogVideoX [Yang et al., 2024] with 720p, 2sec video generation across multiple frames.