Qeios

# Accelerating Vision Diffusion Transformers with Skip Branches

Guanjie Chen[1*] Xinyu Zhao[2*] Yucheng Zhou[3] Tianlong Chen[2†] Yu Cheng[4†]

[1]Shanghai Jiao Tong University [2]The University of North Carolina at Chapel Hill
[3]SKL-IOTSC, CIS, University of Macau [4]The Chinese University of Hong Kong

*chenguanjie@sjtu.edu.cn*, *xinyu@cs.unc.edu*

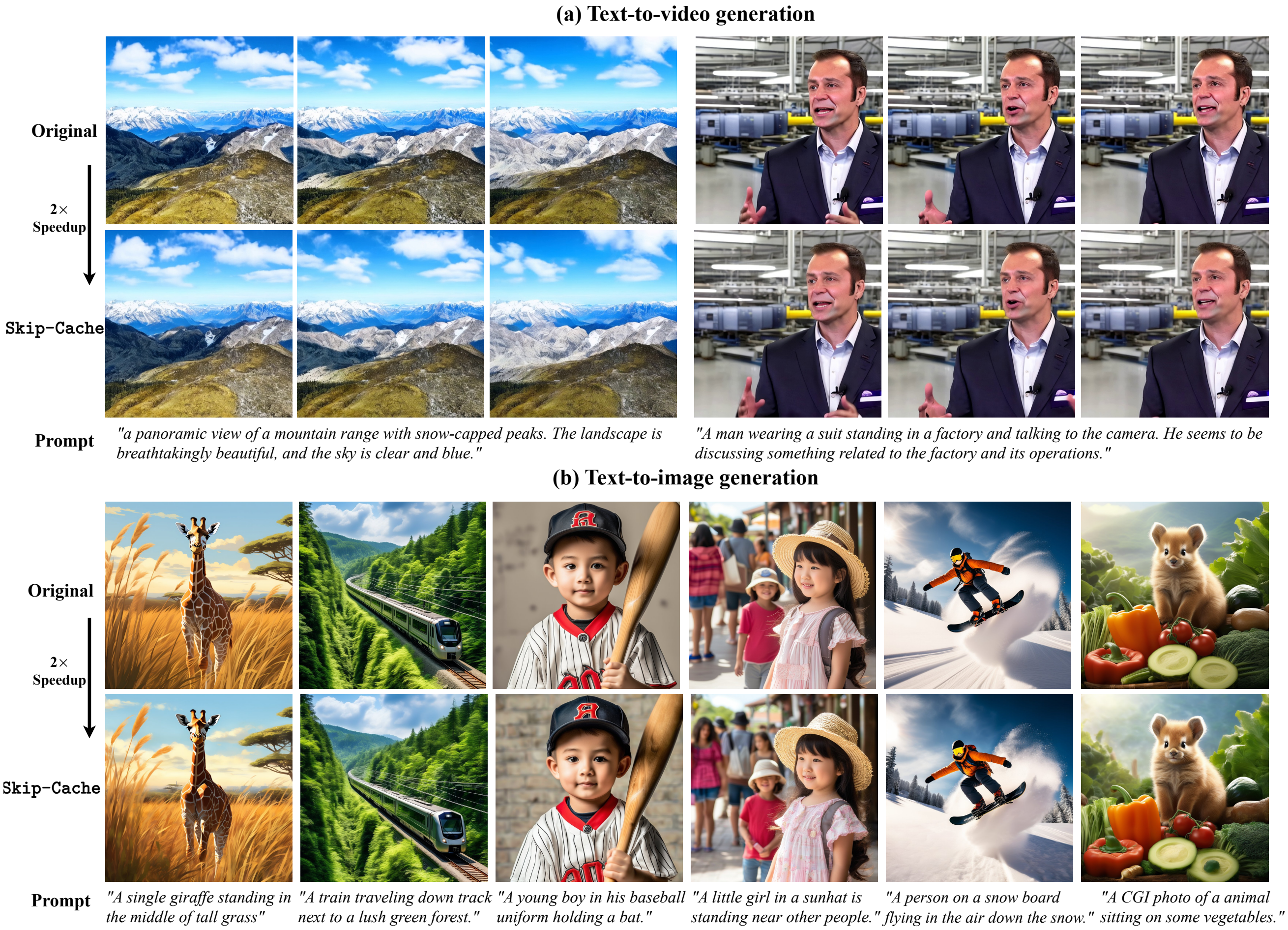

(a) Text-to-video generation
Original
2× Speedup
Skip-Cache
Prompt
"a panoramic view of a mountain range with snow-capped peaks. The landscape is breathtakingly beautiful, and the sky is clear and blue."
"A man wearing a suit standing in a factory and talking to the camera. He seems to be discussing something related to the factory and its operations."
(b) Text-to-image generation
Original
2× Speedup
Skip-Cache
Prompt
"A single giraffe standing in the middle of tall grass"
"A train traveling down track next to a lush green forest."
"A young boy in his baseball uniform holding a bat."
"A little girl in a sunhat is standing near other people."
"A person on a snow board flying in the air down the snow."
"A CGI photo of a animal sitting on some vegetables."

Figure 1. Results and generation speed comparison of `Skip-DiT` cached with `Skip-Cache` and its original version. Utilizing the skipping branch, `Skip-DiT` can significantly boost inference speed while preserving original quality. Latency is measured on one A100. We use Latte as the backbone model for video generation, and Hunyuan-DiT for image generation.

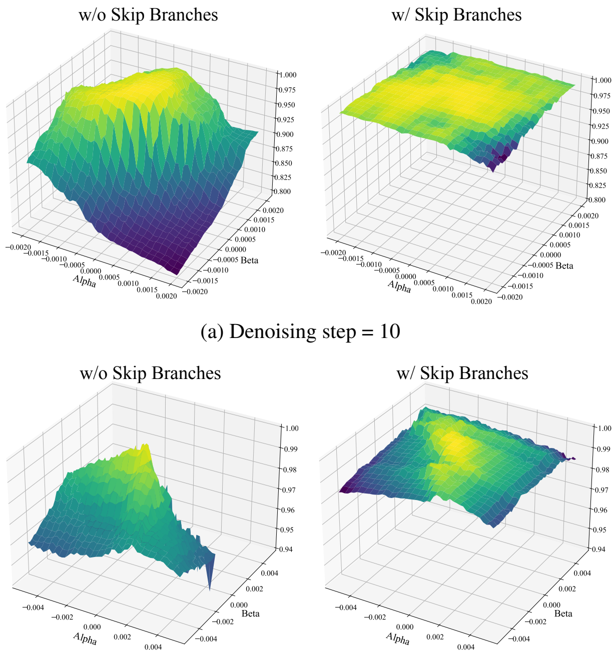
## Abstract

*Diffusion Transformers (DiT), an emerging image and video generation model architecture, has demonstrated great potential because of its high generation quality and scalability. Despite the impressive performance, its practical deployment is constrained by computational complexity and redundancy in the sequential denoising process. While feature caching across timesteps has proven effective in accelerating diffusion models, its application to DiT is limited by fundamental architectural differences from U-Net-based approaches. Through empirical analysis of DiT feature dynamics, we identify that significant feature variation between DiT blocks presents a key challenge for feature* *reusability. To address this, we convert standard DiT into `Skip-DiT` with skip branches to enhance feature smoothness. Further, we introduce `Skip-Cache` which utilizes the skip branches to cache DiT features across timesteps at the inference time. We validated the effectiveness of our proposal on different DiT backbones for video and image generation, showcasing skip branches to help preserve generation quality and achieve higher speedup. Experimental results indicate that `Skip-DiT` achieves a $1.5\times$ speedup almost for free and a $2.2\times$ speedup with only a minor reduction in quantitative metrics. Code is available at `https://github.com/OpenSparseLLMs/Skip-DiT.git`.*

# 1. Introduction

Diffusion models [3, 9, 24, 46] have emerged as the de-facto solution for visual generation, owing to their high fidelity outputs and ability to incorporate various conditioning signals, particularly natural language. Classical diffusion models, which adopt U-Net [27] as their denoising backbone, have dominated image and video generation applications. More recently, Diffusion Transformers (DiT) [4, 23] have introduced an alternative architecture that replaces traditional sequential convolutional networks with Vision Transformers, offering enhanced scalability potential. While initially designed for image generation, DiT has demonstrated remarkable effectiveness when extended to video generation tasks [16, 19, 25]. However, despite these advances, significant challenges remain in scaling diffusion models efficiently, particularly for applications involving large numbers of input tokens such as video generation. This scaling challenge is especially pronounced in DiT architectures, where the attention mechanism's computational complexity grows quadratically with input size. The magnitude of this challenge is illustrated by Sora [20], a state-of-the-art video generation model that remains unavailable to the public.

Numerous approaches have been proposed to enhance the efficiency of diffusion models, including reduced sampling techniques [33], distillation methods [29, 41], and quantization strategies [5]. Caching mechanisms, which reuse noise latent across timesteps, have emerged as a particularly promising direction as they do not require extensive model retraining [6, 15, 18, 38, 47]. However, many existing caching approaches [18, 38] are specifically tailored to U-Net architectures, leveraging their unique structural properties—particularly the skip connections between paired downsampling and upsampling blocks that enable high-level feature reuse while selectively updating low-level features. While some recent studies [6, 47] have attempted to adapt caching mechanisms for DiT acceleration, they have not achieved the same level of efficiency gains and performance preservation as their U-Net counterparts.



(a) Denoising step = 10

(b) Denoising step = 250

Figure 2. Feature smoothness analysis of DiT with class-to-video generation on `Taichi` using DDPM. Normalized disturbances (with strength coefficients $\alpha$ and $\beta$) are added to the model with and without skip connections. We compare the similarity between the original and perturbed features. The feature difference surface of Latte, with and without skip connections, is visualized in steps 10 and 250 of DDPM.

To understand the key challenges of feature caching in DiT, we analyze the feature dynamics during the denoising process. Drawing inspiration from loss landscape visualization techniques [11, 13], we examine feature changes using the early and late timesteps in denoising as a case study. For effective caching, features should exhibit minimal variation between timesteps, allowing us to reuse features from previous steps and bypass computation in subsequent Transformer blocks. We term this property "feature smoothness", which manifests as flatness in the landscape visualization. However, as illustrated in Figure 2, we observe that vanilla DiT (w/o skip branches) exhibits high feature variance across timesteps, contrary to the desired characteristics for effective caching. Thus, we ask:

> ***(Q)** What are the new insights can we gain from feature smoothness in DiT architectures, and how can these insights inform the design of more effective inference caching mechanisms?*

Drawing from the findings in [13], where residual connec-

[*] Equal Contribution.

[†] Corresponding Authors.

tions are shown capable of mitigating sharp loss landscapes, in this study, ❶ we first conduct preliminary experiments adding skip branches to pre-trained DiT model from shallow to deep blocks, named `Skip-DiT` . which achieves significantly improved feature smoothness with minimal continuous pre-training, as demonstrated in Figure 2 (w/ skip branches). ❷ We then leverage these skip branches during inference to implement an efficient caching mechanism, `Skip-Cache` , where only the first DiT block's output needs to be computed for subsequent timesteps while deep block outputs are cached and reused. ❸ To evaluate our proposal, we conduct extensive experiments across multiple DiT backbones, covering image and video generation, class-conditioned and text-conditioned generation. We demonstrate that `Skip-DiT` consistently outperforms both dense baselines and existing caching mechanisms in both qualitative and quantitative evaluations. Our contributions are three-fold:

- We identify feature smoothness as a critical factor limiting the effectiveness of cross-timestep feature caching in DiT, which helps better understand caching efficiency.
- We build `Skip-DiT` , a skip-branch augmented DiT architecture that enhances feature smoothness, and `Skip-Cache` that efficiently leverages skip branches for feature caching across timesteps. `Skip-Cache` facilitates accelerated inference via caching while maintaining visual generation performance.
- Extensive empirical evaluations demonstrate that `Skip-Cache` achieves substantial acceleration: up to $1.5\times$ speedup is achieved almost for free and a $2.2\times$ speedup with only a minor reduction in quantitative metrics.

## 2. Related Works

**Transformer-based Diffusion Models** The diffusion model has become the dominating architecture for image and video generation, whose main idea is iterative generate high-fidelity images or video frames from noise [26]. Early diffusion models mainly employ U-Net as their denoising backbone [3, 24]. However, U-Net architectures struggle to model long-range dependencies due to the local nature of convolutions. Researchers proposing diffusion transformer model (DiT) for image generation [2, 4, 23]. Recent years have witnessed a significant growth in studies of video DiT. Proprietary DiT such as Sora [20] and Movie-Gen [25] show impressive generation quality, also evidenced by open-sourced implementation [12, 48]. Latte decomposes the spatial and temporal dimensions into four efficient variants for handling video tokens, allowing effective modeling of the substantial number of tokens extracted from videos in the latent space [19]. CogvideoX adds a 3D VAE combined with an expert transformer using adaptive LayerNorm, which enables the generation of longer, high-resolution videos [40]. However, as the number of tokens grows exponentially with video length and spatial resolution, the computational complexity of DiT especially the self-attention mechanism remains a significant bottleneck for video generation.

**Diffusion Acceleration with Feature Caching** Since the diffusion model involves iterative denoising, caching features across time-steps, model layers, and modules has been found an effective way to save inference computation costs. For U-Net Diffusion, DeepCache [18] and FRDiff [32] exploit temporal redundancy by reusing features across adjacent denoising steps. While other works take a more structured approach by analyzing and caching specific architectural components–Faster Diffusion [15] specifically targets encoder feature reuse while enabling parallel decoder computation, and Block Caching [38] introduces automated caching schedules for different network blocks based on their temporal stability patterns. Recently, cache-based acceleration has also been applied to DiT. PAB [47] introduces a pyramid broadcasting strategy for attention outputs. $\Delta$-DiT [6] proposes adaptive caching of different DiT blocks based on their roles in a generation–rear blocks during early sampling for details and front blocks during later stages for outlines. T-Gate [45] identifies a natural two-stage inference process, enabling the caching and reuse of text-oriented semantic features after the initial semantics-planning stage. While these caching techniques have shown promise, they are primarily limited to inference-time optimization, and there remains significant potential for improving their acceleration factors.

## 3. Methodology

### 3.1. Preliminaries

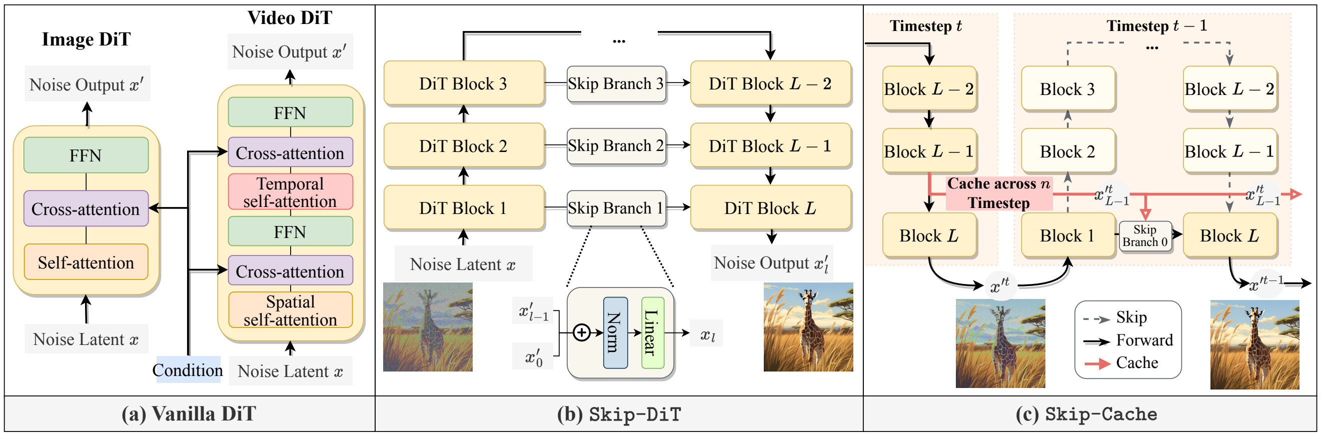
**Diffusion model** The concept of diffusion models mirrors particle dispersion physics, where particles spread out with random motion. It involves forward and backward diffusion. The forward phase adds noise to data across $T$ timesteps. Starting from data $\mathbf{x}_0 \sim q(\mathbf{x})$, noise is added to the data at each timestep $t \in \{1 \ldots T\}$.

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\epsilon_{t-1} \tag{1}$$

where $\alpha$ determines noise level while $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ represents Gaussian noise. The data $\mathbf{x}_t$ becomes increasingly noisy with time, reaching $\mathcal{N}(0, \mathbf{I})$ at $t = T$. Reverse diffusion then reconstructs the original data as follows, where $\mu_\theta$ and $\Sigma_\theta$ refer to the learnable mean and covariance:

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

**Diffusion Transformer** In our study, we consider two types of DiT models processing different visual information: image DiT and video DiT, presented in Figure 3 (a). ❶



(a) Vanilla DiT

(b) Skip-DiT

(c) Skip-Cache

Figure 3. Illustration of Skip-DiT and Skip-Cache for DiT visual generation caching. (a) The vanilla DiT block for image and video generation. (b) Skip-DiT modifies the vanilla DiT model using skip branches to connect shallow and deep DiT blocks. (c) Given a Skip-DiT with $L$ layers, during inference, at the $t-1$ step, the first layer output $x'^{t-1}_0$ and cached $L-1$ layer output $x'^{t}_{L-1}$ are forwarded through the skip branches to the final DiT block to generate the denoising output, without executing DiT blocks 2 to $L-1$.

Image DiT model follows Vision Transformer, each block contains self-attention, cross-attention, and feed-forward networks (FFN), where the cross-attention module incorporates text and timestep conditions. ❷ Video DiT adopts dual-subblock architecture: the spatial Transformer subblock processes tokens within the same time frame, while the temporal subblock manages cross-frame relationships. These blocks alternate in sequence, with cross-attention integrating conditional inputs. A complete video DiT block pairs one spatial and one temporal component in an interleaved pattern, following [19, 47].

## 3.2. Visualizing the Feature Smoothness of DiT

Section 1 introduces the concept of feature smoothness, and provides an intuitive explanation. In this section, we will detail the feature smoothness visualization and analysis.

To use this approach, we denote the original module parameters of the base model as $\theta^*$ in the graph and choose two random direction vectors,$\delta$ and $\eta$, each direction vectors share the same dimension as $\theta$. Firstly, these directions are normalized according to the original parameters it correspond to. We take $\delta$ and $\eta$ to disturb the model with strength coefficients $\alpha$ and $\beta$. After updating, we get a new model with parameter $\theta^{'}$:

$$\theta^{'} = \theta^* + \alpha\delta + \beta\eta \tag{3}$$

We denote the predicted noise after $k$ denoising steps of model before and after adding disturbs as $\boldsymbol{x}^k_{\theta^*}$ and $\boldsymbol{x}^k_{\theta'}$. We then define the feature difference with function;

$$L(\theta^*) = \frac{\boldsymbol{x}^k_{\theta^*} \cdot \boldsymbol{x}^k_{\theta'}}{\|\boldsymbol{x}^k_{\theta^*}\| \|\boldsymbol{x}^k_{\theta'}\|} \tag{4}$$

we then plot the feature difference surface feature according to the 2-D function:

$$f(\alpha, \beta) = L(\theta^* + \alpha\delta + \beta\eta) \tag{5}$$

This approach was employed in [8] and [14], where $L(\theta^*)$ represents the loss of a model with parameters $\theta^*$, used to analyze trajectories of various minimization methods and model structures. Similarly, [11, 13] utilized this approach to demonstrate that different optimization algorithms converge to distinct local minima in the 2D projected space.

## 3.3. Skip-DiT: Improving Feature Smoothness

We visualize vanilla DiT feature smoothness in Figure 2 (w/o Skip branches) which is trained and generated on Taichi dataset. We can observe drastic feature changes at the two DDPM steps, indicating they are not ideal states to cache features, thus shortening the space for caching. According to the insights from [13] and Diffusion caching study utilizing U-Net residual connection feature [18], we next investigate whether DiT feature smoothness can be improved by minimal modification to its structure to insert residual property, *i.e.* skip branches.

A vanilla DiT can be converted into Skip-DiT by connecting shallow blocks to deep blocks with skip branches, as shown in Figure 3 (b). Let $\boldsymbol{x}$ denote the input noise embedding, and $\boldsymbol{x}'_l$ represents the output at the $l$-th layer of Skip-DiT. The architecture consists of $L$ sequential DiT blocks with skip connections. Each DiT block $\mathcal{F}^l_{\text{DiT}}$ at block $l$ processes the features as $\boldsymbol{x}' = \mathcal{F}^l_{\text{DiT}}(\boldsymbol{x})$. The $i$-th skip branch ($i \in \{1 \ldots L//2\}$) connects $i$-th block to ($L+1-i$)-th block, which can be denoted as $\mathcal{F}^i_{\text{Skip}}(\cdot,\cdot)$. Given output $\boldsymbol{x}'_i$ from the start of the skip branch and $\boldsymbol{x}'_l$ from the previous layer, the skip branch aggregates them to the input to $l$-th block as:

$$\boldsymbol{x}_l = \mathcal{F}^i_{\text{skip}}(\boldsymbol{x}'_i, \boldsymbol{x}'_{l-1}) = \texttt{Linear}(\texttt{Norm}(\boldsymbol{x}'_i \oplus \boldsymbol{x}'_{l-1})) \quad (6)$$

where $\oplus$ denotes concatenation, `Norm` represents the layer normalization, and `Linear` is a linear fully-connected layer. The final output $x'_l$ of `Skip-DiT` represents the processed noise output. Each skip branch creates a shortcut path that helps preserve and process information from earlier layers, enabling better gradient flow and feature reuse throughout the network. The combination of DiT blocks and skip branches allows the model to effectively learn the underlying noise distribution while maintaining stable training dynamics.

Similarly, we initialize a class-to-video DiT with skip branches and train it from scratch on `Taichi` dataset, then visualize its feature smoothness as shown in Figure 2 (w/ Skip branches). The results show `Skip-DiT` has a more flat feature-changing landscape at both the beginning and finalizing timesteps. This justifies our initiative to enhance DiT feature smoothness with skip connections.

### 3.4. `Skip-Cache`: Caching with Skip Branches

As we have shown better feature smoothness in `Skip-DiT`, we can use the feature stability and skip branch property of `Skip-DiT` to implement efficient DiT caching, namely `Skip-Cache`. The inference process for global timestep $t$ of full inference can be expressed as follows:

$$\boldsymbol{x}'^t = \mathcal{F}^L_{\text{DiT}}(\mathcal{F}^{L-1}_{\text{DiT}}(...\mathcal{F}^1_{\text{DiT}}(\boldsymbol{x}))) \quad (7)$$

Consider timesteps $t-1$ and $t$ where the model generates $\boldsymbol{x}^{t-1}$ conditioned on $\boldsymbol{x}^t$. During this step, we cache the intermediate output from the last second layer as

$$\mathcal{C}^t_{L-1} = x^t_{L-1} \quad (8)$$

For local timestep $t-1$, with cached feature $\mathcal{C}^t_{l-1}$ and first skip branch $\mathcal{F}^1_{\text{Skip}}(\cdot,\cdot)$, the inference process can be formulated as:

$$x'_{t-1} = \mathcal{F}^L_{\text{DiT}}(\mathcal{F}^1_{\text{Skip}}(\boldsymbol{x}'^{t-1}_1, \mathcal{C}^t_{L-1})) \quad (9)$$

At each local timestep, only 1-th and $L$-th blocks are executed while reusing cached features from the previous global timestep through the skip branch, significantly reducing computational overhead while maintaining generation quality. This can be extended to 1:$N$ inference pattern where $\mathcal{C}^t_{L-1}$ will be reused for the next $N$-1 timesteps.

## 4. Experiments

### 4.1. Implementation Details

**Models** To demonstrate the remarkable effectiveness of `Skip-DiT` and `Skip-Cache` in video generation, we employ the classic and open-source DiT model, Latte [19], as our base model. Latte consists of spatial and temporal transformer blocks, making it suitable for both class-to-video and text-to-video tasks. Hunyuan-DiT [16], the first text-to-image DiT model with skip branches, modifies the model structure by splitting transformer blocks into encoder and decoder blocks, which are connected via long skip connections, similar to UNets. In this work, we leverage Hunyuan-DiT to investigate the effectiveness of skip connections in text-to-image tasks. Additionally, we modify the structure of Latte, following the guidance in Figure 3, to evaluate the performance of skip connections in video generation tasks and explore techniques for integrating skip connections into a pre-trained DiT model.

**Datasets** In the class-to-video task, we conduct comprehensive experiments on four public datasets: FaceForensics [28], SkyTimelapse [39], UCF101 [34], and Taichi-HD [31]. Following the experimental settings in Latte, we extract 16-frame video clips from these datasets and resize all frames to a resolution of 256×256.

For the text-to-video task, the original Latte is trained on Webvid-10M [1] and Vimeo-2M [36], comprising approximately 330k text-video pairs in total. Considering that the resolution of Webvid-10M is lower than 512×512 and is predominantly used in the early stages of pre-training [20], we utilize only Vimeo-2M for training `Skip-DiT`. To align with Latte, we sample 330k text-video pairs from Vimeo-2M. All training data are resized to a resolution of 512×512, with 16 frames per video and a frame rate of 8.

**Training Details** For the training of class-to-video tasks, we train all instances of `Skip-DiT` from scratch without any initialization. During training, we update all parameters in `Skip-DiT`, including skip branches. For text-to-video tasks, we propose a two-stage continual training strategy as follows:

- **Skip-branch training:** The models are initialized with the weights of the original text-to-video Latte model, while the weight of skip branches are initialized randomly. During this stage, we train only the skip branches until the model can roughly generate items. This stage takes approximately one day.
- **Overall training:** After fully training the skip branches, we unfreeze all other parameters and perform overall training. At this stage, `Skip-DiT` rapidly recovers its generation capability within approximately two days and can generate content comparable to the original Latte with an additional three days of training.

This strategy significantly reduces training costs compared to training from scratch. All our training experiments are conducted on 8 H100 GPUs, employing the video-image joint training strategy proposed in Ma et al. [19]. We find that this approach significantly enhances training stability.

Table 1. Class-to-video generation performance. The definition of the cache step $n$ follows that in Table 3. For the UCF101 task, videos are generated using DDIM with 50 steps, while for the remaining tasks, DDPM with 250 steps is employed. Latency and speedup are calculated on one A100 GPU, with solver 250 steps DDPM. $n = i$ indicates caching the high-level feature in $x_t$ for reuse during the inference of $x_{t-1}, x_{t-2}, \ldots, x_{t-n+1}$. We highlight Baseline DiT models (without caching) in grey , and the best metrics in **bold**.

| Method | UCF101 | | FFS | | Sky | | Taichi | | FLOPs (T) | Latency (s) | Speedup |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | FVD (↓) | FID (↓) | FVD (↓) | FID (↓) | FVD (↓) | FID (↓) | FVD (↓) | FID (↓) | | | |
| Latte | 155.22 | 22.97 | 28.88 | 5.36 | 49.46 | 11.51 | 166.84 | 11.57 | 278.63 | 9.90 | 1.00× |
| Δ-DiT | 161.62 | 25.33 | 25.80 | 4.46 | 51.70 | **11.67** | 188.39 | **12.09** | 226.10 | 8.09 | 1.22× |
| FORA | 160.52 | 23.52 | 27.23 | 4.64 | 52.90 | 11.96 | 198.56 | 13.68 | 240.26 | 9.00 | 1.10× |
| PAB$_{23}$ | 213.50 | 30.96 | 58.15 | 5.94 | 96.97 | 16.38 | 274.90 | 16.05 | 233.87 | 7.63 | 1.30× |
| PAB$_{35}$ | 1176.57 | 93.30 | 863.18 | 128.34 | 573.72 | 55.66 | 828.40 | 42.96 | 222.90 | 7.14 | 1.39× |
| Skip-Cache | | | | | | | | | | | |
| Skip-DiT | 141.30 | 23.78 | 20.62 | 4.32 | 49.21 | 11.92 | 163.03 | 13.55 | 290.05 | 10.02 | 1.00× |
| $n = 2$ | 141.42 | 21.46 | **23.55** | **4.49** | **51.13** | 12.66 | **167.54** | 13.89 (**0.34**↑) | 180.68 | 6.40 | 1.56× |
| $n = 3$ | **137.98** | 19.93 | 26.76 | 4.75 | 54.17 | 13.11 | 179.43 | 14.53 | 145.87 | 5.24 | 1.91× |
| $n = 4$ | 143.00 | 19.03 | 30.19 | 5.18 | 57.36 | 13.77 | 188.44 | 14.38 | 125.99 | 4.57 | 2.19× |
| $n = 5$ | 145.39 | **18.72** | 35.52 | 5.86 | 62.92 | 14.18 | 209.38 | 15.20 | 121.02 | 4.35 | 2.30× |
| $n = 6$ | 151.77 | 18.78 | 42.41 | 6.42 | 68.96 | 15.16 | 208.04 | 15.78 | **111.07** | **4.12** | **2.43**× |

**Evaluation Details** Following previous works [18, 47] and Latte, we evaluate text-to-video models using VBench [10], Peak Signal-to-Noise Ratio (PSNR), Learned Perceptual Image Patch Similarity (LPIPS) [43], and Structural Similarity Index Measure (SSIM) [37]. VBench is a comprehensive benchmark suite comprising 16 evaluation dimensions. PSNR is a widely used metric for assessing the quality of image reconstruction, LPIPS measures feature distances extracted from pre-trained networks, and SSIM evaluates structural information differences. All the videos generated for evaluation are sampled with 50 steps DDIM [33], which is the default setting used in Latte.

For class-to-image tasks, we evaluate the similarity between generated and real videos using Fréchet Video Distance (FVD) [35] and Fréchet Inception Distance (FID) [21], following the evaluation guidelines of StyleGAN-V [42]. Latte uses 250-step DDPM [9] as the default solver for class-to-video tasks, which we adopt for all tasks except UCF101. For UCF101, we employ 50-step DDIM [33], as it outperforms 250-step DDPM on both Latte and Skip-DiT. Table 2 highlights this phenomenon, showing our methods consistently outperform DDPM-250 under comparable throughput, except for UCF101, where DDIM performs better than 250 steps DDPM.

**Implementation Details of Other Caching Methods** We compare with 4 other DiT caching methods in video and image generation: ❶ T-GATE [44] reuses self-attention in the semantics-planning phase and skips cross-attention in the fidelity-improving phase. We follow Zhao et al. [47] to split these two phases. ❷ Δ-DiT identifies high similarity in deviations between feature maps and reuses them at the next timestep. While this method is originally designed for images, we extend it to video DiTs by caching only the front blocks, as we observe significant degeneration when caching the back blocks. ❸ FORA [30] reuses attention features across timesteps. ❹ PAB [47] further extends it by broadcasting cross, spatial, and temporal attention features separately. All these caching methods are performed on Latte for equal comparison with Skip-Cache on Skip-DiT .

## 4.2. Main Results

**Class-to-video Generation** We compare the quantitative performance of Latte and Skip-DiT on four class-to-video tasks, as shown in Table 1. Skip-DiT consistently outperforms Latte in terms of FVD scores across all tasks while achieving comparable performance in FID scores, demonstrating its strong video generation capabilities. Furthermore, we observe that Skip-Cache significantly outperforms other caching methods across most metrics, incurring only an average loss of 2.37 in the FVD score and 0.27 in the FID score while achieving a 1.56× speedup. In comparison, only PAB [47] achieves a speedup of more than 1.3×, but at the cost of a substantial average loss of 60.78 in FVD score and 4.48 in FID score. Notably, in the Taichi [31] task, all other caching methods exhibit significant degradation in FVD scores($\geq$ 21.5), whereas Skip-Cache experiences only a slight loss (163.06→167.54). To achieve even higher speedup on the other three class-to-video tasks, we accelerated Skip-Cache with a larger cache timesteps (N=3), resulting in a 2.19× speedup with an average loss of 6.47 in FVD and a 0.68 improvement in FID.

**Text-to-video Generation** In Table 3, we present a quantitative evaluation of all text-to-video models and caching methods. Videos are generated using the prompts from VBench [10], which is considered a more generalized benchmark [10, 47, 48]. Compared with the original Latte, Skip-Cache achieves a comparable VBench score (75.60

Table 2. Comparison with the faster DDIM sampler. Skip-Cache is evaluated with a 250-steps DDPM and compared to the DDIM sampler under similar throughput. Baseline DiT models (without caching) are highlighted in grey, and the best metrics are indicated in **bold**. Notably, DDIM outperforms the 250-step DDPM in the UCF101 task for both Latte and Skip-DiT .

| Method | UCF101 | | FFS | | Sky | | Taichi | |
|---|---|---|---|---|---|---|---|---|
| | FVD ↓ | FID ↓ | FVD ↓ | FID ↓ | FVD ↓ | FID ↓ | FVD ↓ | FID ↓ |
| Latte | 165.04 | 23.75 | 28.88 | 5.36 | 49.46 | 11.51 | 166.84 | 11.57 |
| Skip-DiT | 173.70 | 22.95 | 20.62 | 4.32 | 49.22 | 12.05 | 163.03 | 13.55 |
| Skip-Cache$_{n=2}$ | 165.60 | **22.73** | **23.55** | **4.49** | **51.13** | **12.66** | **167.54** | **13.89** |
| DDIM+Skip-DiT | **134.22** | 24.60 | 37.28 | 6.48 | 86.39 | 13.67 | 343.97 | 21.01 |
| DDIM+Latte | 146.78 | 23.06 | 39.10 | 6.47 | 78.38 | 13.73 | 321.97 | 21.86 |
| Skip-Cache$_{n=3}$ | 169.37 | **22.47** | **26.76** | **4.75** | **54.17** | **13.11** | **179.43** | **14.53** |
| DDIM+Skip-DiT | **139.52** | 24.71 | 39.20 | 6.49 | 90.62 | 13.80 | 328.47 | 21.33 |
| DDIM+Latte | 148.46 | 23.41 | 41.00 | 6.54 | 74.39 | 14.20 | 327.22 | 22.96 |

Table 3. Text-to-video generation performance. Latency and speedup are calculated using a single A100 GPU. The notation $n = i$ represents caching high-level features in $x_t$ for reuse during the inference of $x_{t-1}, x_{t-2}, \ldots, x_{t-n+1}$. Skip-Cache 65% and Skip-Cache 75% correspond to performing caching at 65% and 75% of the timesteps, respectively. Baseline DiT models (without caching) are highlighted in grey, while the best metrics are emphasized in **bold**.

| Method | VBench(%) ↑ | PSNR ↑ | LPIPS ↓ | SSIM ↑ | FLOPs (T) | latency (s) | speedup |
|---|---|---|---|---|---|---|---|
| Latte | 76.14 | – | – | – | 1587.25 | 27.11 | 1.00× |
| T-GATE | 75.68 (↓0.46) | 22.78 | 0.19 | 0.78 | 1470.72 | 24.15 | 1.12× |
| Δ-DiT | **76.06** (↓0.08) | 24.01 | 0.17 | 0.81 | 1274.36 | 21.40 | 1.27× |
| FORA | **76.06** (↓0.08) | 22.93 | 0.14 | 0.79 | 1341.72 | 24.21 | 1.19× |
| PAB$_{235}$ | 73.79 (↓2.35) | 19.18 | 0.27 | 0.66 | 1288.08 | 23.24 | 1.24× |
| PAB$_{347}$ | 72.08 (↓4.06) | 18.20 | 0.32 | 0.63 | 1239.35 | 22.23 | 1.29× |
| PAB$_{469}$ | 71.64 (↓4.50) | 17.40 | 0.35 | 0.60 | 1210.11 | 21.60 | 1.33× |
| Skip-DiT | 75.60 | – | – | – | 1648.13 | 28.72 | 1.00× |
| Skip-Cache 75% | | | | | | | |
| $n = 2$ | 75.36 (↓0.24) | 26.02 | 0.10 | 0.84 | 1066.62 | 18.25 | 1.57× |
| $n = 3$ | 75.07 (↓0.53) | 22.85 | 0.18 | 0.76 | 852.38 | 14.88 | 1.93× |
| $n = 4$ | 74.43 (↓1.17) | 22.08 | 0.22 | 0.73 | **760.56** | **13.03** | **2.20**× |
| Skip-Cache 65% | | | | | | | |
| $n = 2$ | 75.51 (↓0.09) | **29.52** | **0.06** | **0.89** | 1127.83 | 19.28 | 1.49× |
| $n = 3$ | 75.26 (↓0.34) | 27.46 | 0.09 | 0.85 | 974.80 | 16.67 | 1.72× |
| $n = 4$ | 74.73 (↓0.87) | 25.97 | 0.13 | 0.81 | 882.98 | 15.12 | 1.90× |

*vs.* 76.14) with only six days of continual pre-training on 330k training samples.

To demonstrate the superiority of the caching mechanism in Skip-Cache, we evaluate two caching settings: caching at timesteps 700–50 and 800–50 (out of 1000 timesteps in total). In both settings, Skip-Cache achieves the highest speedup while maintaining superior scores in PSNR, LPIPS, and SSIM, with only a minor loss in VBench score.

In the first setting, our caching mechanism achieves 1.49× and 1.72× speedup with only a 0.12% and 0.22% loss in VBench score, respectively. Among other caching methods, only PAB$_{469}$ achieves a speedup of more than 1.30×, but at the cost of a 4.50% drop in VBench score. Moreover, our caching method can achieve a 1.90× speedup while still maintaining absolutely better PSNR, LPIPS, and SSIM scores compared to other caching methods.

Furthermore, in the second setting, we achieve a 2.20× speedup with just a 1.17% sacrifice in VBench score, representing the highest speedup among current training-free DiT acceleration works.

## 4.3. Generalize Skip-Cache to image Generation

Hunyuan-DiT [16] is a powerful text-to-image DiT model featuring skip branches, whose effectiveness has been demonstrated in [16]. However, its skip branches have not been explored for accelerating image generation. We leverage these skip branches using the same caching mechanism as Skip-Cache and compare our caching strategy with other training-free acceleration methods. Furthermore, we extend our proposed Skip-Cache to class-to-image task in Appendix 6, where Skip-DiT exceeds vanilla DiT model with only around 38% of its training cost.

**Evaluation Details** To evaluate the generalization of the caching mechanism in Skip-Cache for text-to-image tasks, we use the zero-shot Fréchet Inception Distance (FID) on the MS COCO [17] 256×256 validation dataset by generating 30,000 images based on its prompts, following the evaluation guidelines established by Hunyuan-DiT. Additionally, we employ Peak Signal-to-Noise Ratio (PSNR), Learned Perceptual Image Patch Similarity (LPIPS) [43], and Structural Similarity Index Measure (SSIM) [37] to assess the changes introduced by the caching methods. To ensure a fair comparison, we disable the prompt enhancement feature of Hunyuan-DiT. All videos are generated at a resolution of 1024×1024 and subsequently resized to 256×256 for evaluation.

**Evaluation Results** Table 4 provides a comprehensive comparison of Hunyuan-DiT and various caching methods. Notably, our caching mechanism achieves a 2.09× speedup without any degradation in FID or CLIP scores. Furthermore, it outperforms all other caching methods in terms of PSNR, LPIPS, and SSIM scores, consistently maintaining the highest performance even with a 1.96× speedup. These findings underscore the robustness and adaptability of our caching mechanism to image generation tasks.

## 4.4. Ablation studies

**Select the best timesteps to cache** A heat map visualizing feature dynamics across blocks is shown in Figure 4. After incorporating skip branches into Latte, we observe that major changes are concentrated in the early timesteps, with feature dynamics becoming considerably smoother in the later timesteps (700→50). In contrast, features in Latte exhibit rapid changes across all timesteps. This finding highlights that caching during smoother timesteps leads to

Table 4. Text-to-image generation performance. Latency and speedup are calculated on one A100 GPU. $n_i$ indicates caching the high-level feature in $x_t$ for reuse during the inference of $x_{t-1}, x_{t-2}, \ldots, x_{t-n+1}$. We highlight baseline DiT models (without caching) in grey, and the best metrics in **bold**.

| Method | FID (↓) | CLIP (↑) | PSNR (↑) | LPIPS (↓) | SSIM (↑) | FLOPs (T) | latency (s) | speedup |
|---|---|---|---|---|---|---|---|---|
| HunYuan-DiT | 32.64 | 30.51 | – | – | – | 514.02 | 18.69 | 1.00 |
| TGATE | 32.71 | 30.64 | 16.80 | 0.24 | 0.61 | 378.94 | 13.21 | 1.41 |
| Δ-Cache | 28.35 | 30.35 | 16.56 | 0.21 | 0.65 | 362.67 | 13.58 | 1.38 |
| FORA | 31.21 | 30.53 | 19.58 | 0.14 | 0.75 | 330.68 | 13.20 | 1.42 |
| Skip-Cache | | | | | | | | |
| $n = 2$ | 31.30 | 30.52 | **22.09** | **0.10** | **0.84** | 348.24 | 12.76 | 1.46 |
| $n = 3$ | 29.53 | 30.55 | 21.25 | 0.11 | 0.81 | 299.48 | 10.91 | 1.71 |
| $n = 4$ | 27.49 | 30.55 | 20.55 | 0.13 | 0.78 | 270.22 | 10.02 | 1.87 |
| $n = 5$ | 28.37 | 30.56 | 19.94 | 0.14 | 0.76 | 260.47 | 9.51 | 1.96 |
| $n = 6$ | **27.21** | **30.71** | 19.18 | 0.18 | 0.70 | **240.96** | **8.96** | **2.09** |

Table 5. Ablation study on different timestep cached. Total timesteps is 1000. During early timesteps (1000→700), the feature changes rapidly. In the latter timesteps (700→50) feature dynamics become considerably smoother. We only perform Skip-Cache in caching timesteps We highlight the best metrics in **bold**.

| **Caching Timestep** | **VBench** (%) ↑ | **PSNR** ↑ | **LPIPS** ↓ | **SSIM** ↑ |
|---|---|---|---|---|
| 700→50 | **75.51** | **29.52** | **0.06** | **0.89** |
| 950→300 | 75.48 | 20.58 | 0.23 | 0.73 |
| 800→50 | 75.36 | 26.02 | 0.10 | 0.84 |
| 900→50 | 75.24 | 22.13 | 0.19 | 0.76 |

Table 6. Compatibility of Skip-DiT with other caching methods. All the methods are performed on Skip-DiT . $n = i$ indicates caching the high-level feature in $x_t$ for reuse during the inference of $x_{t-1}, x_{t-2}, \ldots, x_{t-n+1}$. We highlight baseline DiT models (without caching) in grey, and the best metrics in **bold**.

| **Method** | **VBench** (%) ↑ | **PSNR** ↑ | **LPIPS** ↓ | **SSIM** ↑ |
|---|---|---|---|---|
| Skip-DiT | 75.60 | – | – | – |
| T-GATE | 75.16 | **24.09** | **0.16** | 0.78 |
| Δ-DiT | **75.48** | 22.64 | 0.17 | **0.79** |
| FORA | 75.38 | 23.26 | 0.16 | 0.79 |
| $\text{PAB}_{235}$ | 73.79 | 19.92 | 0.29 | 0.68 |
| $\text{PAB}_{347}$ | 72.08 | 18.98 | 0.34 | 0.65 |
| $\text{PAB}_{469}$ | 71.64 | 18.02 | 0.37 | 0.62 |

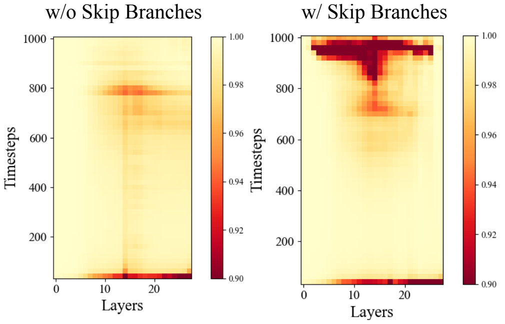


Figure 4. The feature dynamics of Latte with and without Skip Branches. We use cosine similarity to evaluate the differences in features at the same layers across timesteps.

significantly better performance, supporting the hypothesis that smooth features enhance caching efficiency in DiT. In Table 5, we further validate this observation: under equivalent throughput, caching in the later timesteps (700→50), where features are smoother, outperforms caching in the earlier timesteps (950→300), achieving superior PSNR, LPIPS, and SSIM scores. Additionally, we segmented the rapidly changing timesteps and experimented with three caching ranges: 900→50, 800→50, and 700→50. The results show that increasing the ratio of smoother regions significantly improves caching performance. These findings underscore the importance of leveraging smoother feature dynamics for optimal caching.

**Compatibility of Skip-DiT with other caching methods** As shown in Table 6, we extend the existing DiT caching methods to Skip-DiT and observe slight performance improvements. Specifically, in Δ-DiT, the middle blocks are cached instead of the front blocks, as discussed in Section 4.1, to leverage Skip-DiT' U-shaped structure. Taking PAB [47] as an example, it loses 1.15% less in VBench score and achieves noticeably better PSNR and SSIM scores on Skip-DiT compared to Latte, highlighting the potential of Skip-DiT to enhance cache-based methods, and proving the superior caching efficiency of the model with better feature smoothness.

## 5. Conclusion

In this work, we introduce Skip-DiT, a skip-branch-enhanced DiT model designed to produce smoother features and propose Skip-DiT cache to improve caching efficiency in video and image generation tasks. By enhancing feature smoothness across timesteps, Skip-DiT unlocks the potential to cache most blocks while maintaining high-generation quality. Additionally, Skip-Cache leverages its U-net-style architecture to enable cross-timestep feature caching. Our approach achieves maximum speedup in cache-based visual DiT generation while preserving the highest similarity to original outputs. Furthermore, we analyze feature dynamics before and after incorporating skip branches, demonstrating the effectiveness of caching at timesteps with smoother features. We also show that `Skip-DiT` is compatible with other caching methods, further extending its applicability. Overall, `Skip-DiT` can seamlessly integrate with various DiT backbones, enabling real-time, high-quality video and image generation while consistently outperforming baseline methods. We believe `Skip-Cache` offers a simple yet powerful foundation for advancing future research and practical applications in visual generation.

## References

[1] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1728–1738, 2021.

[2] 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. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22669–22679, 2022.

[3] James Betker, Gabriel Goh, Li Jing, † TimBrooks, Jianfeng Wang, Linjie Li, † LongOuyang, † JuntangZhuang, † JoyceLee, † YufeiGuo, † WesamManassra, † PrafullaDhariwal, † CaseyChu, † YunxinJiao, and Aditya Ramesh. Improving image generation with better captions.

[4] Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Yue Wu, Zhongdao Wang, James T. Kwok, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-$\alpha$: Fast training of diffusion transformer for photorealistic text-to-image synthesis. *ArXiv*, abs/2310.00426, 2023.

[5] Lei Chen, Yuan Meng, Chen Tang, Xinzhu Ma, Jingyan Jiang, Xin Wang, Zhi Wang, and Wenwu Zhu. Q-dit: Accurate post-training quantization for diffusion transformers. *arXiv preprint arXiv:2406.17343*, 2024.

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

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[8] Ian J. Goodfellow and Oriol Vinyals. Qualitatively characterizing neural network optimization problems. *CoRR*, abs/1412.6544, 2014.

[9] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[10] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. Vbench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21807–21818, 2024.

[11] Daniel Jiwoong Im, Michael Tao, and Kristin Branson. An empirical analysis of deep network loss surfaces. *ArXiv*, abs/1612.04010, 2016.

[12] PKU-Yuan Lab and Tuzhan AI etc. Open-sora-plan, 2024.

[13] Hao Li, Zheng Xu, Gavin Taylor, and Tom Goldstein. Visualizing the loss landscape of neural nets. *ArXiv*, abs/1712.09913, 2017.

[14] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6391–6401, 2018.

[15] Senmao Li, Taihang Hu, Fahad Shahbaz Khan, Linxuan Li, Shiqi Yang, Yaxing Wang, Ming-Ming Cheng, and Jian Yang. Faster diffusion: Rethinking the role of unet encoder in diffusion models. *arXiv preprint arXiv:2312.09608*, 2023.

[16] Zhimin Li, Jianwei Zhang, Qin Lin, Jiangfeng Xiong, Yanxin Long, Xinchi Deng, Yingfang Zhang, Xingchao Liu, Minbin Huang, Zedong Xiao, Dayou Chen, Jiajun He, Jiahao Li, Wenyue Li, Chen Zhang, Rongwei Quan, Jianxiang Lu, Jiabin Huang, Xiaoyan Yuan, Xiaoxiao Zheng, Yixuan Li, Jihong Zhang, Chao Zhang, Meng Chen, Jie Liu, Zheng Fang, Weiyan Wang, Jinbao Xue, Yangyu Tao, Jianchen Zhu, Kai Liu, Sihuan Lin, Yifu Sun, Yun Li, Dongdong Wang, Mingtao Chen, Zhichao Hu, Xiao Xiao, Yan Chen, Yuhong Liu, Wei Liu, Di Wang, Yong Yang, Jie Jiang, and Qinglin Lu. Hunyuan-dit: A powerful multi-resolution diffusion transformer with fine-grained chinese understanding, 2024.

[17] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, pages 740–755. Springer, 2014.

[18] Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15762–15772, 2023.

[19] 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*, 2024.

[20] OpenAI. Sora: Creating video from text. `https://openai.com/sora`, 2024.

[21] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On buggy resizing libraries and surprising subtleties in FID calculation. *CoRR*, abs/2104.11222, 2021.

[22] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference*

*on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 4172–4182. IEEE, 2023.

[23] William S. Peebles and Saining Xie. Scalable diffusion models with transformers. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4172–4182, 2022.

[24] Dustin Podell, Zion English, Kyle Lacey, A. Blattmann, Tim Dockhorn, Jonas Muller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *ArXiv*, abs/2307.01952, 2023.

[25] Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, David Yan, Dhruv Choudhary, Dingkang Wang, Geet Sethi, Guan Pang, Haoyu Ma, Ishan Misra, Ji Hou, Jialiang Wang, Kiran Jagadeesh, Kunpeng Li, Luxin Zhang, Mannat Singh, Mary Williamson, Matt Le, Matthew Yu, Mitesh Kumar Singh, Peizhao Zhang, Peter Vajda, Quentin Duval, Rohit Girdhar, Roshan Sumbaly, Sai Saketh Rambhatla, Sam Tsai, Samaneh Azadi, Samyak Datta, Sanyuan Chen, Sean Bell, Sharadh Ramaswamy, Shelly Sheynin, Siddharth Bhattacharya, Simran Motwani, Tao Xu, Tianhe Li, Tingbo Hou, Wei-Ning Hsu, Xi Yin, Xiaoliang Dai, Yaniv Taigman, Yaqiao Luo, Yen-Cheng Liu, Yi-Chiao Wu, Yue Zhao, Yuval Kirstain, Zecheng He, Zijian He, Albert Pumarola, Ali Thabet, Artsiom Sanakoyeu, Arun Mallya, Baishan Guo, Boris Araya, Breena Kerr, Carleigh Wood, Ce Liu, Cen Peng, Dimitry Vengertsev, Edgar Schonfeld, Elliot Blanchard, Felix Juefei-Xu, Fraylie Nord, Jeff Liang, John Hoffman, Jonas Kohler, Kaolin Fire, Karthik Sivakumar, Lawrence Chen, Licheng Yu, Luya Gao, Markos Georgopoulos, Rashel Moritz, Sara K. Sampson, Shikai Li, Simone Parmeggiani, Steve Fine, Tara Fowler, Vladan Petrovic, and Yuming Du. Movie Gen: A Cast of Media Foundation Models. *arXiv e-prints*, art. arXiv:2410.13720, 2024.

[26] Robin Rombach, A. Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10674–10685, 2021.

[27] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *ArXiv*, abs/1505.04597, 2015.

[28] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics: A large-scale video dataset for forgery detection in human faces. *CoRR*, abs/1803.09179, 2018.

[29] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. *arXiv preprint arXiv:2311.17042*, 2023.

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

[31] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. First order motion model for image animation. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7135–7145, 2019.

[32] Junhyuk So, Jungwon Lee, and Eunhyeok Park. Frdiff: Feature reuse for exquisite zero-shot acceleration of diffusion models. *arXiv preprint arXiv:2312.03517*, 2023.

[33] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[34] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, abs/1212.0402, 2012.

[35] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *CoRR*, abs/1812.01717, 2018.

[36] Yaohui Wang, Xinyuan Chen, Xin Ma, Shangchen Zhou, Ziqi Huang, Yi Wang, Ceyuan Yang, Yinan He, Jiashuo Yu, Peiqing Yang, Yuwei Guo, Tianxing Wu, Chenyang Si, Yuming Jiang, Cunjian Chen, Chen Change Loy, Bo Dai, Dahua Lin, Yu Qiao, and Ziwei Liu. LAVIE: high-quality video generation with cascaded latent diffusion models. *CoRR*, abs/2309.15103, 2023.

[37] Zhou Wang and Alan C. Bovik. A universal image quality index. *IEEE Signal Process. Lett.*, 9(3):81–84, 2002.

[38] Felix Wimbauer, Bichen Wu, Edgar Schoenfeld, Xiaoliang Dai, Ji Hou, Zijian He, Artsiom Sanakoyeu, Peizhao Zhang, Sam Tsai, Jonas Kohler, et al. Cache me if you can: Accelerating diffusion models through block caching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6211–6220, 2024.

[39] Wei Xiong, Wenhan Luo, Lin Ma, Wei Liu, and Jiebo Luo. Learning to generate time-lapse videos using multi-stage dynamic generative adversarial networks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2364–2373. Computer Vision Foundation / IEEE Computer Society, 2018.

[40] 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.

[41] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6613–6623, 2024.

[42] Sihyun Yu, Jihoon Tack, Sangwoo Mo, Hyunsu Kim, Junho Kim, Jung-Woo Ha, and Jinwoo Shin. Generating videos with dynamics-aware implicit generative adversarial networks. In *International Conference on Learning Representations*, 2022.

[43] 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.

[44] 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. *CoRR*, abs/2404.02747, 2024.

[45] 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 preprint arXiv:2404.02747*, 2024.

[46] Yiming Zhang, Zhening Xing, Yanhong Zeng, Youqing Fang, and Kai Chen. Pia: Your personalized image animator via plug-and-play modules in text-to-image models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7747–7756, 2024.

[47] Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-time video generation with pyramid attention broadcast, 2024.

[48] 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, 2024.

# Accelerating Vision Diffusion Transformers with Skip Branches

## Supplementary Material

## 6. Class-to-image Generation Experiments

Peebles and Xie [22] proposed the first diffusion model based on the transformer architecture, and it outperforms all prior diffusion models on the class conditional ImageNet [7] 512×512 and 256×256 benchmarks. We add skip branches to its largest model DiT-XL/2 to get `Skip-DiT`. We train `Skip-DiT` on class conditional ImageNet with resolution 256×256 from scratch with completely the same experiments setting as DiT-XL/2, and far exceeds DiT-XL/2 with only around 38% of its training cost.

**Training of `Skip-DiT`** We modify the structure of DiT-XL/2 following the methodology outlined in Section 3 and train `Skip-DiT` for 2,900,000 steps on 8 A100 GPUs, compared to 7,000,000 steps for DiT-XL/2, which also uses 8 A100 GPUs. The datasets and other training settings remain identical to those used for DiT-XL/2, and we utilize the official training code of DiT-XL/2*. The performance comparison is presented in Table 7, which demonstrates that `Skip-DiT` significantly outperforms DiT-XL/2 while requiring only 38% of its training steps, highlighting the training efficiency and effectiveness of `Skip-DiT`.

Table 7. Comparison of training efficiency between DiT-XL/2 and `Skip-DiT` . Images are generated with a 250-step DDPM solver. The term *cfg* refers to classifier-free guidance scales, where metrics for *cfg=1.0* are computed without classifier-free guidance. The best metrics are highlighted in **bold**. `Skip-DiT` significantly exceeds DiT-XL/2 with much less training steps.

| Model | Steps | FID ↓ | sFID ↓ | IS ↑ | Precision ↑ | Recall ↑ |
|---|---|---|---|---|---|---|
| | | | ***cfg=1.0*** | | | |
| DiT-XL/2 | 7000k | 9.49 | 7.17 | 122.49 | 0.67 | **0.68** |
| `Skip-DiT` | **2900k** | **8.37** | **6.50** | **127.63** | **0.68** | **0.68** |
| | | | ***cfg=1.5*** | | | |
| DiT-XL/2 | 7000k | 2.30 | 4.71 | 276.26 | **0.83** | **0.58** |
| `Skip-DiT` | **2900k** | **2.29** | **4.58** | **281.81** | **0.83** | **0.58** |

**Accelerating Evaluation** We evaluate `Skip-Cache` on `Skip-DiT` and compare its performance against two other caching methods: Δ-DiT and FORA. As shown in Table 8, `Skip-Cache` achieves a 1.46× speedup with only a minimal FID loss of 0.02 when the classifier-free guidance scale is set to 1.5, compared to the 7–8× larger losses observed with Δ-DiT and FORA. Moreover, even with a 1.9× acceleration, `Skip-DiT` performs better than the other caching methods. These findings further confirm the effectiveness of `Skip-DiT` for class-to-image tasks.

Table 8. Class-to-image generation performance. The definition of the cache step $n$ follows that in Table 4. Images are generated with a 250-step DDPM solver. Speedups are calculated on an H100 GPU with a sample batch size of 8. $n = i$ indicates caching the high-level features in $x_t$ for reuse during the inference of $x_{t-1}, x_{t-2}, \ldots, x_{t-n+1}$. The term *cfg* refers to classifier-free guidance scales, where metrics for *cfg=1.0* are computed without classifier-free guidance. We highlight baseline DiT models (without caching) in grey and the best metrics in **bold**.

| Methods | FID ↓ | sFID ↓ | IS ↑ | Precision% | Recall% | Speedup |
|---|---|---|---|---|---|---|
| | | | ***cfg=1.5*** | | | |
| DiT-XL/2 | 2.30 | 4.71 | 276.26 | 82.68 | 57.65 | 1.00× |
| FORA | 2.45 | 5.44 | 265.94 | 81.21 | 58.36 | 1.57× |
| Delta-DiT | 2.47 | 5.61 | 265.33 | 81.05 | **58.83** | 1.45× |
| | | | `Skip-Cache` | | | |
| `Skip-DiT` | 2.29 | 4.58 | 281.81 | 82.88 | 57.53 | 1.00× |
| $n = 2$ | **2.31** | **4.76** | **277.51** | **82.52** | 58.06 | 1.46× |
| $n = 3$ | 2.40 | 4.98 | 272.05 | 82.14 | 57.86 | 1.73× |
| $n = 4$ | 2.54 | 5.31 | 267.34 | 81.60 | 58.31 | **1.93**× |
| | | | ***cfg=1.0*** | | | |
| DiT-XL/2 | 9.49 | 7.17 | 122.49 | 66.66 | 67.69 | 1.00× |
| FORA | 11.72 | 9.27 | 113.01 | 64.46 | 67.69 | 1.53× |
| Delta-DiT | 12.03 | 9.68 | 111.86 | 64.57 | 67.53 | 1.42× |
| | | | `Skip-Cache` | | | |
| `Skip-DiT` | 8.37 | 6.50 | 127.63 | 68.06 | 67.89 | 1.00× |
| $n = 2$ | **9.25** | **7.09** | **123.57** | **67.32** | 67.40 | 1.46× |
| $n = 3$ | 10.18 | 7.72 | 119.60 | 66.53 | **67.84** | 1.71× |
| $n = 4$ | 11.37 | 8.49 | 116.01 | 65.73 | 67.32 | **1.92**× |

## 7. Evaluation Details

**VBench [10]** is a novel evaluation framework for video generation models. It breaks down video generation assessment to 16 dimensions from video quality and condition consistency: subject consistency, background consistency, temporal flickering, motion smoothness, dynamic degree, aesthetic quality, imaging quality, object class, multiple objects, human action, color, spatial relationship, scene, temporal style, appearance style, overall consistency.

**Peak Signal-to-Noise Ratio (PSNR)** measures generated visual content quality by comparing a processed version $\mathbf{v}$ to the original reference $\mathbf{v}_r$ by:

$$\text{PSNR} = 10 \times \log_{10}\left(\frac{R^2}{\texttt{MSE}(\mathbf{v}, \mathbf{v}_r)}\right) \tag{10}$$

where $R$ is the maximum possible pixel value, and $\texttt{MSE}(\cdot, \cdot)$ calculates the Mean Squared Error between original and

[*] https://github.com/facebookresearch/DiT

processed images or videos. Higher PSNR indicates better reconstruction quality. However, PSNR does not always correlate with human perception and is sensitive to pixel-level changes.

**Structural Similarity Index Measure (SSIM)** is a perceptual metric that evaluates image quality by considering luminance, contrast, and structure:

$$\mathrm{SSIM} = [l(\mathbf{v}, \mathbf{v}_r)]^{\alpha} \cdot [c(\mathbf{v}, \mathbf{v}_r)]^{\beta} \cdot [s(\mathbf{v}, \mathbf{v}_r \mathbf{v}_r)]^{\gamma} \tag{11}$$

where $\alpha, \beta, \gamma$ are weights for luminance, contrast, and structure quality, where luminance comparison is $l(x, y) = \frac{2\mu_{\mathbf{v}}\mu_{\mathbf{v}_r} + C_1}{\mu_{\mathbf{v}}^2 + \mu_{\mathbf{v}_r}^2 + C_1}$, contrast comparison is $c(x, y) = \frac{2\sigma_{\mathbf{v}}\sigma_{\mathbf{v}_r} + C_2}{\sigma_{\mathbf{v}}^2 + \sigma_{\mathbf{v}_r}^2 + C_2}$, and structure comparison is $s(x, y) = \frac{\sigma_{xy} + C_3}{\sigma_{\mathbf{v}}\sigma_{\mathbf{v}_r} + C_3}$, with $C$ denoting numerical stability coefficients. SSIM scores range from -1 to 1, where 1 means identical visual content.

**Learned Perceptual Image Patch Similarity (LPIPS)** is a deep learning-based metric that measures perceptual similarity using L2-Norm of visual features $v \in \mathbb{R}^{H \times W \times C}$ extracted from pretrained CNN $\mathcal{F}(\cdot)$. LPIPS captures semantic similarities and is therefore more robust to small geometric transformations than PSNR and SSIM.

$$\mathrm{LPIPS} = \frac{1}{HW} \sum_{h,w} ||\mathcal{F}(v_r) - \mathcal{F}(v)||_2^2 \tag{12}$$

**Fréchet Inception Distance (FID) and Fréchet Video Distance (FVD)** FID measures the quality and diversity of generated images by computing distance between feature distributions of reference $\mathcal{N}(\mu_r, \Sigma_r)$ and generated images $\mathcal{N}(\mu, \Sigma)$ using inception architecture CNNs, where $\mu, \Sigma$ are mean and covariance of features.

$$\mathrm{FID} = ||\mu_r - \mu||^2 + Tr(\Sigma_r + \Sigma - 2(\Sigma_r \Sigma)^{1/2}) \tag{13}$$

FVD is a video extension of FID. Lower FID and FVD indicate higher generation quality.

## 8. Implementation Details

**DeepCache** DeepCache [18] is a training-free caching method designed for U-Net-based diffusion models, leveraging the inherent temporal redundancy in sequential denoising steps. It utilizes the skip connections of the U-Net to reuse high-level features while updating low-level features efficiently. `Skip-Cache` shares significant similarities with DeepCache but extends the method to DiT models. Specifically, we upgrade traditional DiT models to `Skip-DiT` and cache them using `Skip-Cache` . In the work of DeepCache, two key caching decisions are introduced: (1) `N`: the number of steps for reusing cached high-level features. Cached features are computed once and reused for the next `N-1` steps. (2) The layer at which caching is performed. For instance, caching at the first layer ensures that only the first and last layers of the U-Net are recomputed. In `Skip-Cache`, we adopt these two caching strategies and additionally account for the timesteps to cache, addressing the greater complexity of DiT models compared to U-Net-based diffusion models. For all tasks except the class-to-image task, caching is performed at the first layer, whereas for the class-to-image task, it is applied at the third layer.

**$\Delta$-DiT** $\Delta$-DiT [6] is a training-free caching method designed for image-generating DiT models. Instead of caching the feature maps directly, it uses the offsets of features as cache objects to preserve input information. This approach is based on the observation that the front blocks of DiT are responsible for generating the image outlines, while the rear blocks focus on finer details. A hyperparameter $b$ is introduced to denote the boundary between the outline and detail generation stages. When $t \leq b$, $\Delta$-Cache is applied to the rear blocks; when $t > b$, it is applied to the front blocks. The number of cached blocks is represented by $N_c$.

While this caching method was initially designed for image generation tasks, we extend it to video generation tasks. In video generation, we observe significant degradation in performance when caching the rear blocks, so we restrict caching to the front blocks during the outline generation stage. For Hunyuan-DiT [16], we cache the middle blocks due to the U-shaped transformer architecture. Detailed configurations are provided in Table 9.

Table 9. Configuration details for $\Delta$-Cache in different models and tasks. *t2v* denotes text-to-video, *c2v* denotes class-to-video, *t2i* denotes text-to-image, and *c2i* denotes class-to-image.

| $\Delta$-DiT | Task | Diffusion steps | $b$ | All layers | $N_c$ |
|---|---|---|---|---|---|
| Latte | *t2v* | 50 | 12 | 28 | 21 |
| Latte | *c2v* | 250 | 60 | 14 | 10 |
| Hunyuan | *t2i* | 50 | 12 | 28 | 18 |
| DiT-XL/2 | *c2i* | 250 | 60 | 28 | 21 |

**PAB** PAB (Pyramid Attention Broadcast) [47] is one of the most promising caching methods designed for real-time video generation. The method leverages the observation that attention differences during the diffusion process follow a U-shaped pattern, broadcasting attention outputs to subsequent steps in a pyramid-like manner. Different broadcast ranges are set for three types of attention—spatial, temporal, and cross-attention—based on their respective differences. $PAB_{\alpha\beta\gamma}$ denotes the broadcast ranges for spatial ($\alpha$), temporal ($\beta$), and cross ($\gamma$) attentions.

In this work, we use the official implementation of PAB for text-to-video tasks on Latte and adapt the caching method to other tasks in-house. For the class-to-video task, where cross-attention is absent, $PAB_{\alpha\beta}$ refers to the broadcast ranges of spatial ($\alpha$) and temporal ($\beta$) attentions. In the text-to-image task, which lacks temporal attention, $PAB_{\alpha\beta}$ instead denotes the broadcast ranges of spatial ($\alpha$) and cross ($\beta$) attentions. We do not apply PAB to the class-to-image task, as it involves only spatial attention.

Table 10. Configuration details for T-GATE in different settings. *t2v* denotes text-to-video, *t2i* denotes text-to-image.

| T-GATE | Task | Diffusion steps | m | k |
|---|---|---|---|---|
| Latte | *t2v* | 50 | 20 | 2 |
| Hunyuan-DiT | *t2i* | 50 | 20 | 2 |

**T-Gates** T-Gates divide the diffusion process into two phases: (1) the Semantics-Planning Phase and (2) the Fidelity-Improving Phase. In the first phase, self-attention is computed and reused every $k$ step. In the second phase, cross-attention is cached using a caching mechanism. The hyperparameter $m$ determines the boundary between these two phases. For our implementation, we use the same hyperparameters as PAB [47]. Detailed configurations are provided in Table 10.

**FORA** FORA (Fast-Forward Caching) [30] stores and reuses intermediate outputs from attention and MLP layers across denoising steps. However, in the original FORA paper, features are cached in advance before the diffusion process. We do not adopt this approach, as it is a highly time-consuming process. Instead, in this work, we skip the "Initialization" step in FORA and calculate the features dynamically during the diffusion process.

## 9. Case Study

**Video Generation** In Figure 5, we showcase the generated video frames from text prompts with `Skip-Cache` , PAB, and comparing them to the original model. From generating portraits to scenery, `Skip-Cache` consistently demonstrates better visual fidelity along with faster generation speeds. Figure 6 presents class-to-video generation examples with `Skip-Cache` with varying caching steps $\in \{2, 4, 6\}$. By comparing `Skip-Cache` to Original model, we see `Skip-Cache` maintain good generation quality across different caching steps.

**Image Generation** Figure 7 compares qualitative results of `Skip-Cache` compared to other caching-based acceleration methods ($\Delta$-DiT, FORA, T-GATE) on Hunyuan-DiT. In Figure 8, `Skip-Cache` show distinct edges in higher speedup and similarity to the original generation, while other baselines exist with different degrees of change in details such as color, texture, and posture. Similarly, we present `Skip-Cache` with varying caching steps in Figure 8, showing that with more steps cached, it still maintains high fidelity to the original generation.

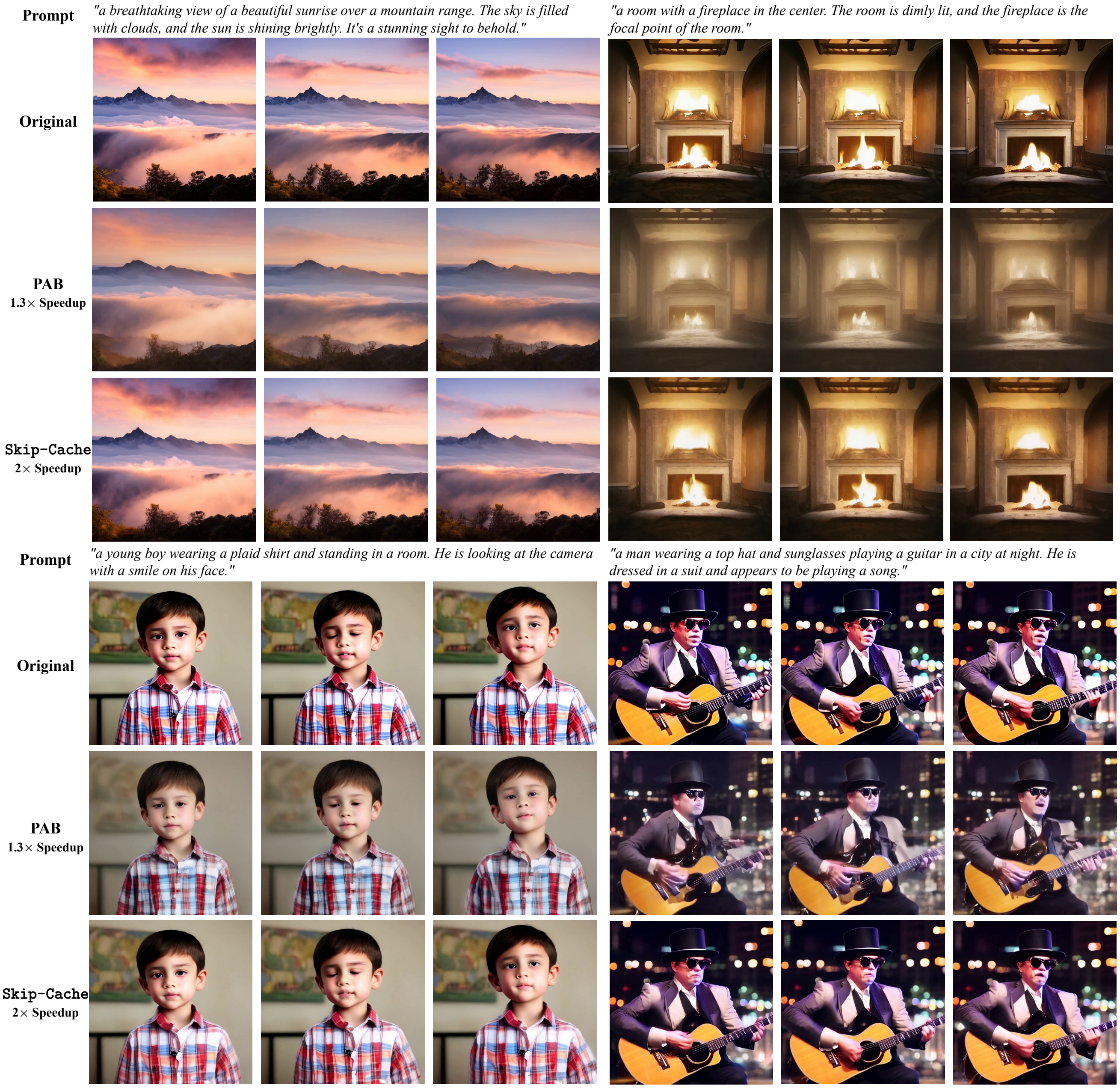

Prompt
"a breathtaking view of a beautiful sunrise over a mountain range. The sky is filled with clouds, and the sun is shining brightly. It's a stunning sight to behold."
"a room with a fireplace in the center. The room is dimly lit, and the fireplace is the focal point of the room."
Original
PAB
1.3× Speedup
Skip-Cache
2× Speedup
Prompt
"a young boy wearing a plaid shirt and standing in a room. He is looking at the camera with a smile on his face."
"a man wearing a top hat and sunglasses playing a guitar in a city at night. He is dressed in a suit and appears to be playing a song."
Original
PAB
1.3× Speedup
Skip-Cache
2× Speedup

Figure 5. Qualitative results of text-to-video generation. We present `Skip-Cache` , PAB$_{469}$, and the original model. We apply `Skip-DiT` on Latte as the backbone model for video generation. The frames are randomly sampled from the generated video.

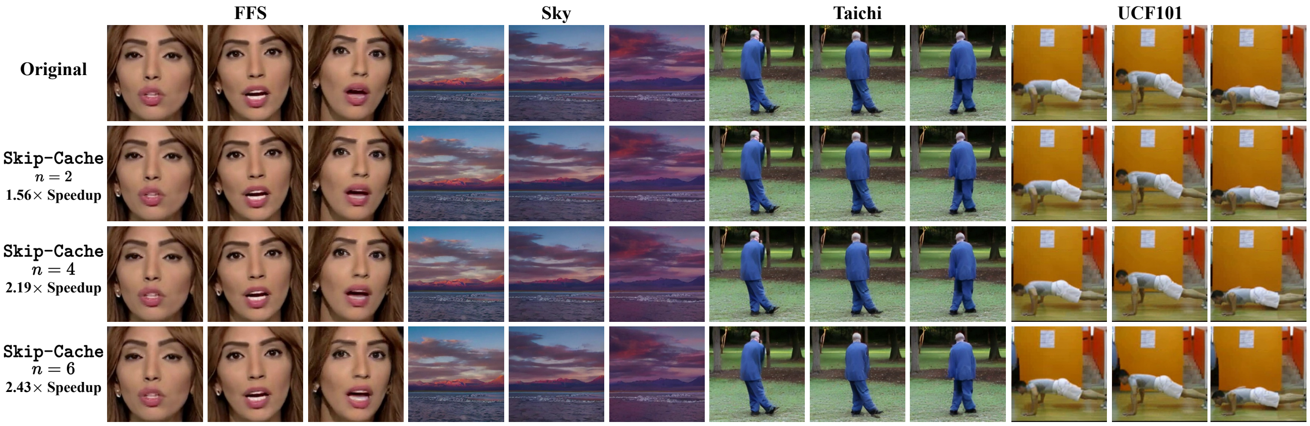


Figure 6. Qualitative results of class-to-video generation. We present the original video generation model and `Skip-Cache` with different caching steps $n$. We apply `Skip-DiT` on Latte as the backbone model for video generation. The frames are randomly sampled from the generated video.

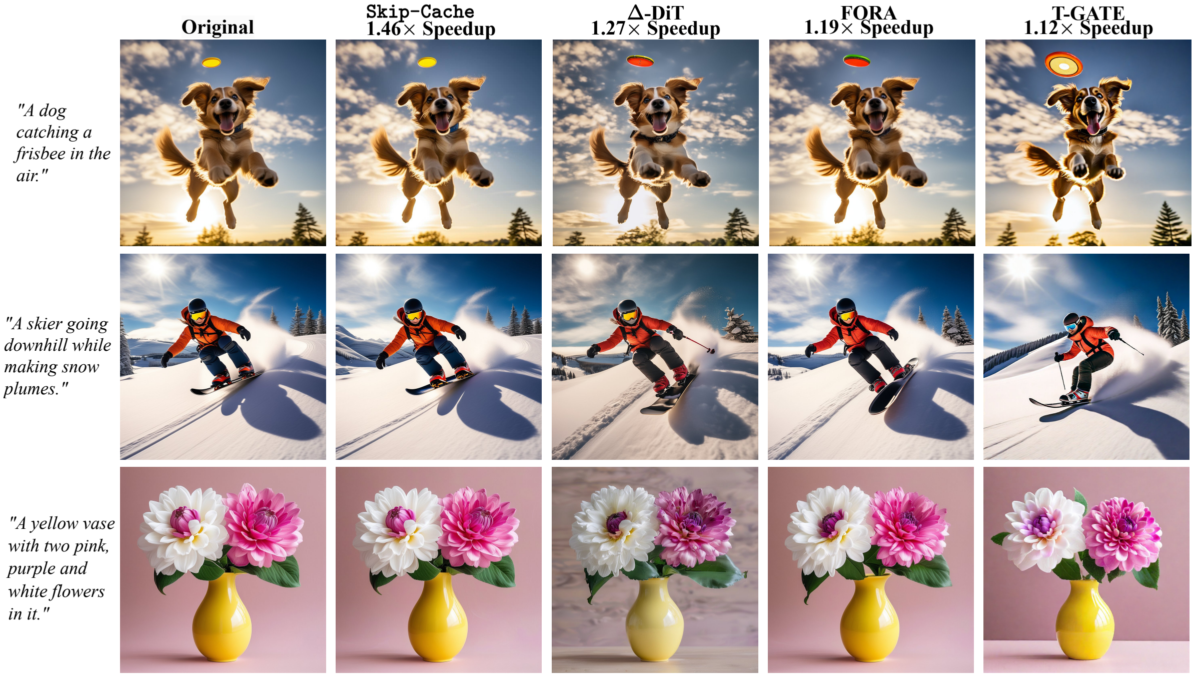

Original
Skip-Cache
1.46× Speedup
Δ-DiT
1.27× Speedup
FORA
1.19× Speedup
T-GATE
1.12× Speedup
"A dog catching a frisbee in the air."
"A skier going downhill while making snow plumes."
"A yellow vase with two pink, purple and white flowers in it."

Figure 7. Qualitative results of text-to-image generation. We present `Skip-Cache` , Δ-DiT, FORA, T-GATE, and the original model. We apply `Skip-DiT` on Hunyuan-DiT as the backbone model for image generation.

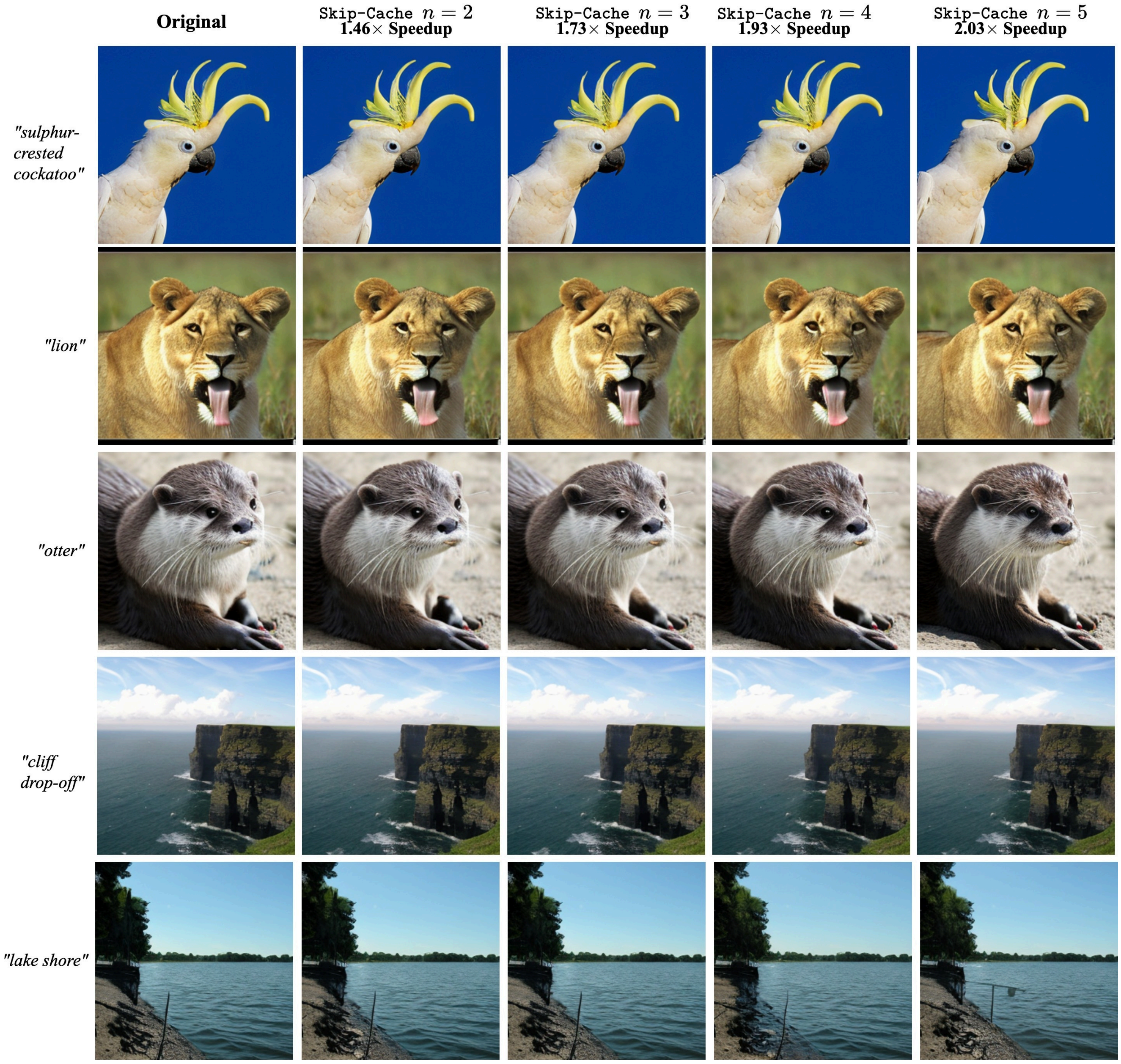


Figure 8. Qualitative results of class-to-image generation. We present the original image generation model and Skip-Cache with different caching steps $n$. We apply Skip-DiT on Hunyuan-DiT as the backbone model for image generation.