

ResCa: Residual Caching for Diffusion Transformers Acceleration

Haipeng Fang^{1,2}, Yu Li^{1,2}, Fan Tang^{✉,1,2}, Yixing Lu^{1,2}, Juan Cao^{1,2}, Sheng Tang^{1,2}

¹ Institute of Computing Technology, Chinese Academy of Sciences

² University of Chinese Academy of Sciences

Abstract

Diffusion transformers have achieved remarkable progress in high-quality image and video generation, but their high computational overhead remains a significant issue. Existing token reduction-based acceleration techniques, such as caching and merging, attempt to reduce this cost from both temporal and spatial perspectives but often compromise generation quality by introducing non-updated or non-self denoising directions. In this paper, we propose Residual Caching (ResCa), a novel, training-free framework that introduces a **proxy denoising** perspective to overcome these limitations. ResCa achieves acceleration while maintaining a denoising trajectory that is both self and updated. The core idea is to perform true denoising on only one “proxy token” within each trajectory-based cluster, and use its computed multi-order residuals to guide the “simulated denoising” of all other tokens. ResCa can be seamlessly integrated into various diffusion models, including DiT, FLUX, and HunyuanVideo. Extensive quantitative and qualitative experiments demonstrate the effectiveness of our method, achieving up to a $5.5\times$ acceleration in GFLOPs while maintaining near-lossless generation quality on FLUX. Project page: <https://fanghaipeng.github.io/ResCa>.

1. Introduction

Diffusion transformers (DiTs) [30] have achieved remarkable success in high-fidelity image [4, 7, 19, 41, 43] and video [10, 18, 37] generation. Scaling model capacity or adding controllable modules [42, 44, 45] improves generation quality but increases computational cost, motivating dedicated acceleration techniques. Among them, sampler-based techniques optimize the denoising process [25, 29, 32, 35, 36], model-based techniques reduce architectural redundancy [2, 8, 16, 21], and feature-based techniques [1, 6, 9, 12, 24, 28, 39, 51] prune feature redundancy.

✉Corresponding author: Fan Tang.

In this manuscript, we use *skip-residuals* to denote residuals in skip connections, and *residuals* to denote the feature deviation across timesteps.

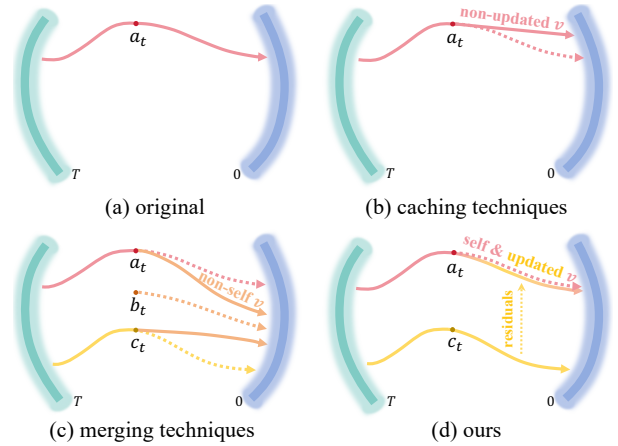


Figure 1. Illustration of denoising trajectories. Solid and dashed line denotes the *realized* and the *target* trajectories. (a) Original trajectory. (b) Caching: reuse the previous step x_t^a , forming a *non-updated* denoising direction (does not stay fixed due to residual connections). (c) Merging: merge similar a_t and c_t into b_t , reusing the x_t^b , forming a *non-self* denoising direction. (d) Ours: computes multi-order residuals $\Delta^k x_t^c$ to guide the cached x_t^a , thus achieving a both *self* and *updated* trajectory.

Within the feature-based techniques, token reduction [1, 9, 11, 24, 27, 51] aim to decrease the number of tokens. As shown in Fig. 1, caching methods such as ToCa [51], DuCa [52] and TokenCache [24], address temporal redundancy by reusing token features from previous timesteps, leading to a *non-updated* denoising direction. Merging techniques such as ToMeSD [1], ToMA [27], and SDTM [9] merge similar tokens and use the mixed features, creating a *non-self* denoising direction. We observe that residual variations along similar denoising paths often follow similar trends, which motivates our “proxy denoising” perspective.

Our core idea is to first cluster tokens based on their historical trajectories. Then, within each cluster, we perform “*true denoising*” on only one proxy token and use its computed multi-order residuals to guide the “*simulated denoising*” of all other unprocessed tokens. As illustrated in Fig. 1 (d), this proxy denoising design enables an efficient

simulated denoising. It uses the token’s own cached feature as the primary direction (the *self* component), and applies the multi-order residuals from the proxy’s true denoising as a correction (the *updated* component), thereby estimating trajectories that more closely align with the original, full-computation denoising paths.

We further examine this idea in preliminary experiments, as shown in Fig. 2, by asking (a) *where to find the similar residuals* and (b) *how to use the similar residuals*. For question (a), we visualize denoising trajectories for feature-based clusters (merging-oriented) and trajectory-based clusters, and compare the intra-cluster pairwise distances of residuals. Results show that trajectory-based clustering more reliably identifies similar residuals. For question (b), we analyze intra-cluster pairwise distances of multi-order residuals and their evolution across timesteps, highlighting that lower-order residuals exhibit higher reusability and that their utility can be estimated from the preceding timesteps.

Based on the above priors, we formalize our ResCa, consisting of two key components: Temporal-Enhanced Trajectory Clustering (TETC), which implements the trajectory-based clustering; and Proxy-Driven Denoising Simulation (PDDS), which performs the residual-corrected simulated denoising for non-proxy tokens with implicit ODE update. In summary, the contributions of this paper are as follows:

- We propose a new residual-based perspective for acceleration and systematically analyze the reusability and utility of multi-order residuals by investigating where to find similar residuals and how to use them.
- We propose a novel framework ResCa that uses a denoised trajectory to guide other unprocessed trajectories within clusters through temporal-enhanced trajectory clustering and proxy-driven denoising simulation, enabling adaptive and stable multi-step predictions.
- The proposed ResCa is training-free and can be seamlessly integrated into diverse diffusion transformers, and comprehensive experiments validate our effectiveness, achieving up to $5.5\times$ acceleration in GFLOPs on FLUX with near-lossless generation quality.

2. Related Works

2.1. Efficient Diffusion Models

Despite strong generative capabilities, diffusion models incur substantial computation, which limits real-time and large-scale applications. To accelerate DMs, recent work explores three directions: (1) *Sampler-based*: these methods aim to reduce the number of denoising steps. DDIM [35] introduces deterministic non-Markovian sampling; the DPM-Solver family [25, 26, 48] formulates sampling as solving an ODE; Consistency Models [36] enforce self-consistency to enable single- or few-step generation; and distillation approaches [17, 29, 32] compress the sam-

pler into fewer steps. (2) *Model-based*: these methods aim to reduce architectural redundancy. Structured pruning [2, 8, 16, 21] removes unimportant weights, while quantization [12, 20, 34, 46] reduces redundancy via low-bit weights. (3) *Feature-based*: these methods cut token-level computation, including quantization techniques [20, 46] and token-reduction methods [1, 5, 6, 22, 24, 28, 33, 38, 39, 47]. Owing to their training-free potential, feature-based methods have attracted increasing attention.

2.2. Token Reduction in Diffusion Models

Token reduction methods aim to reduce inference tokens and can be grouped into three categories. (1) *Merging-based*: This category includes explicit and implicit forms. In explicit merging, ToMeSD [1], SDTM [9], and ToMA [27] merge tokens based on similarity, attention, or frequency, process the reduced set, and then restore features to their original positions. In implicit merging, AT-EDM [38] and DyDiT [47] prune tokens by importance and fill missing positions with features copied from similar retained tokens. In both forms, pruned tokens reuse shared rather than individual features, yielding a *non-self* direction. (2) *Caching-based*: This category includes static reuse and dynamic prediction. In static reuse, ToCa [51] performs fine-grained caching with dynamic token selection, Token-Cache [24] uses a predictor to assign scores and cache tokens, and DuCa [52] alternates conservative and aggressive caching across timesteps. By reusing features from preceding timesteps, these approaches induce a *non-updated* denoising direction. In contrast, dynamic prediction replaces caching with estimation: TaylorSeer [23] uses Taylor extrapolation, while FoCa [49] models prediction with ODEs. However, diffusion dynamics are non-stationary, and timestep-specific signals can shift the denoising direction, making history-only extrapolation insufficient. (3) *Mixed-based*: SDTM [9] and ClusCa [50] linearly weight temporal cache with spatially similar features; however, the spatial term is often underweighted, leaving room for further exploration. To address this, We denoises a small token subset, decouples informative components, and uses them as on-step feedback to guide the remaining tokens, yielding a denoising direction that is both *self* and *updated*.

3. Preliminary and Motivations

Our approach is driven by a “proxy denoising” perspective. The core idea is to first cluster tokens based on their historical trajectories. Then, within each cluster, we perform *true denoising* on only one proxy token and use its computed multi-order residuals to guide the *simulated denoising* of all other unprocessed tokens. This idea is further examined by addressing the following questions. Experiments are conducted on FLUX.1-dev [19] using 10k samples, with additional details provided in the *Supplementary Materials*.

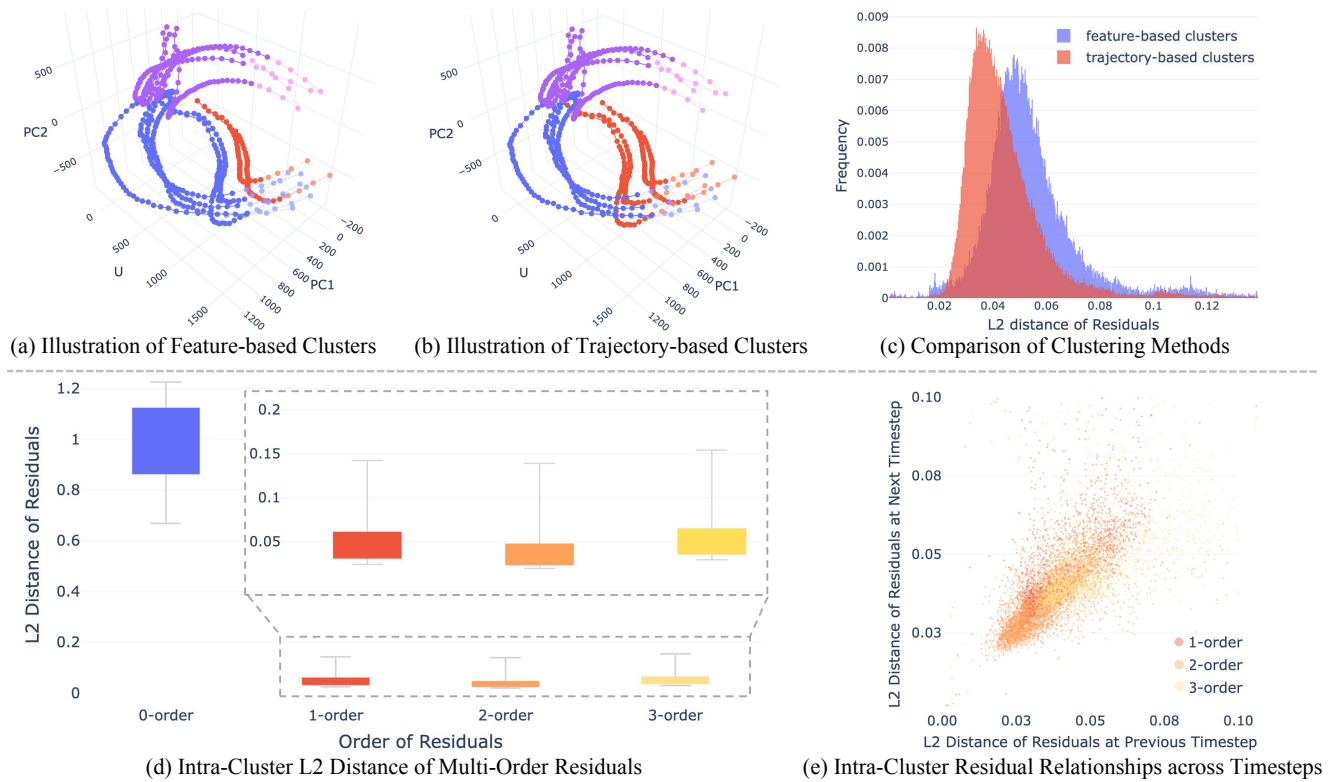


Figure 2. Preliminaries. *Upper: Where to find the similar residuals?* Illustration of (a) feature- and (b) trajectory-based clusters, where the trajectory-based clusters groups the blue and red trajectories more accurately. (c) Comparison of the two methods showing that trajectory-based clusters have lower pairwise distance of residuals within each cluster. *Lower: How to use the similar residuals?* (d) Intra-cluster L2 distance of multi-order residuals, indicating that 1st-, 2nd-, and 3rd-order residuals are more reusable than 0th-order ones. (e) Intra-cluster residual relationships across timesteps, showing that smaller historical residual distance implies higher reuse value for future residuals.

Where to find the similar residuals? We first qualitatively inspect denoising trajectories using the trajectory approximation method [3]. As shown in Fig. 2 (a) and (b), we project the high-dimensional feature trajectory into a 3D trajectory, where darker points denote sampled timesteps and lighter points denote unsampled ones. For merging-based techniques such as ToMeSD [1], tokens are typically grouped according to feature similarity at the last timestep, which can cause tokens with dissimilar trajectories to be clustered together, for example the blue trajectory in Fig. 2 (a). To mitigate this issue, we instead perform clustering based on the entire historical trajectory so that truly similar trajectories can be grouped together, as illustrated by the blue and red trajectories in Fig. 2 (b). In Fig. 2 (c), we further provide a quantitative comparison between the two clustering strategies by plotting histograms of the intra-cluster pairwise L2 distance of first-order residuals (differences between adjacent timesteps). The results show that trajectory-based clustering yields smaller residual distances, highlighting that *historical trajectory-based clusters can more reliably identify similar residuals.*

How to use the similar residuals? After clustering tokens

based on the historical denoising trajectories, we further analyze the reusability and utility of intra-cluster residuals. For each trajectory at each timestep, we compute multi-order residuals: the 0th-order residual corresponds to the feature itself, the 1st-order residual represents the difference between features at adjacent timesteps, and the k -th-order residual captures the change in the $(k - 1)$ -th-order residual. Fig. 2 (d) reports the intra-cluster pairwise L2 distance of these multi-order residuals. The results show that 0th-order residuals have larger distances, while higher-order residuals exhibit smaller distances, since 0th-order residuals still encode the token representation itself, whereas higher-order residuals partially decouple the directional component of the denoising trajectory. In contrast, 3rd-order inter-residual distances are slightly larger than 2nd-order ones, because higher-order residuals further amplify small variations and noise. Fig. 2 (e) further illustrates the relationships of residuals across timesteps: the inter-residual distance at the preceding timestep and the next timestep follow an approximately linear trend along $y = x$. Overall, *lower-order residuals exhibit higher reusability, and their utility can be estimated based on the preceding trajectory.*

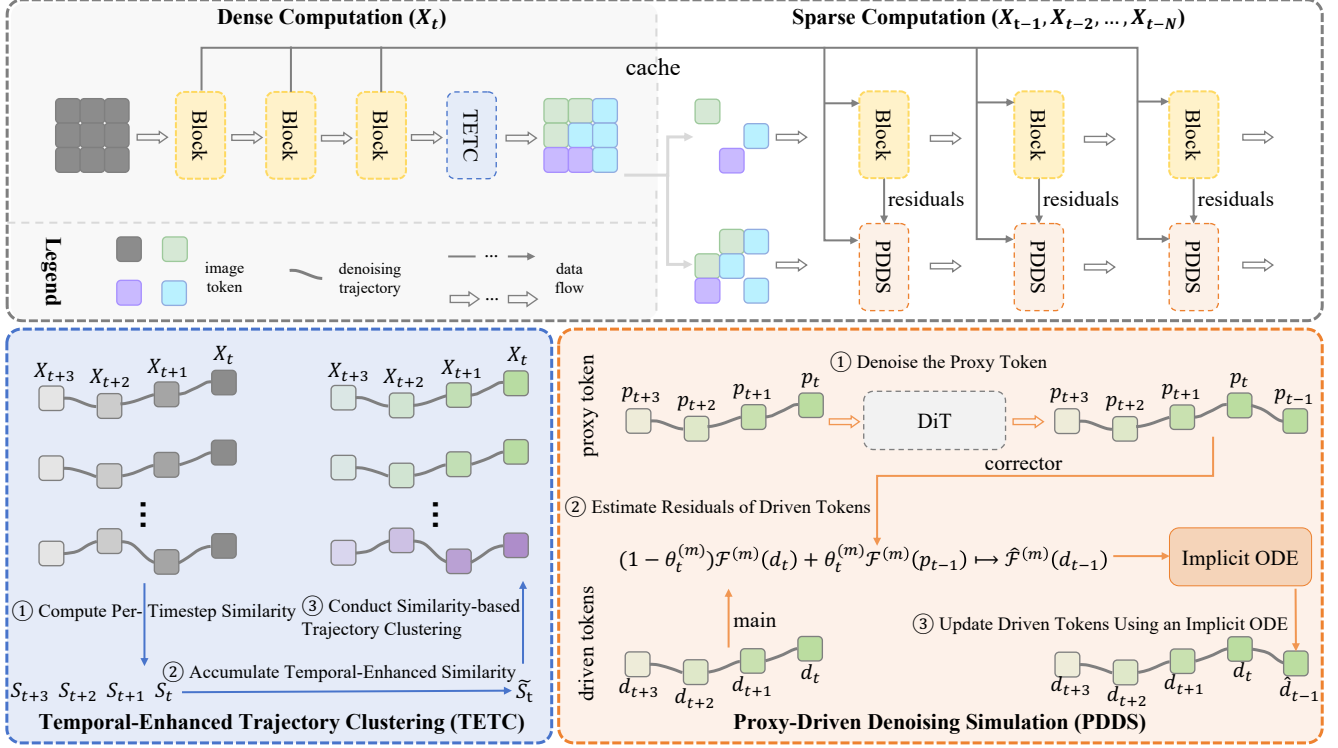


Figure 3. Overview. *grey*: In the dense timestep, all tokens are processed to cache their features, followed by TETC to group tokens. In the sparse timestep, one proxy token from each group is selected for denoising, which then uses PDDS to guide the denoising of other tokens. *blue*: Temporal-Enhanced Trajectory Clustering (TETC): perform clustering using a similarity metric computed by token trajectories. *orange*: Proxy-driven Denoising Simulation (PDDS): use the driven token’s residuals as the main direction; the proxy token’s residuals provide directional corrections to estimate the future direction, and subsequently simulate the denoising process using an implicit ODE.

4. Method

4.1. Overview

We introduce ResCa, a residual caching approach that leverages a proxy token to guide the simulated denoising of driven tokens, shown in Fig. 3. The denoising process is divided into dense and sparse computation timesteps. In the dense computation timestep, all tokens are processed to cache their features, followed by temporal-enhanced trajectory clustering to group similar tokens. In the sparse computation timestep, one proxy token is selected from each cluster for denoising, which then uses the proxy-driven denoising simulation to guide the denoising of other tokens.

4.2. Temporal-Enhanced Trajectory Clustering

Motivated by question (a) in Sec. 3, we observe that clusters constructed from historical denoising trajectories can more reliably group tokens with similar residuals. To further encourage the alignment of future residuals, we assign larger weights to more recent timesteps. To this end, we introduce the temporal-enhanced trajectory clustering (TETC) method. As illustrated in Fig. 3, TETC consists of three stages: compute per-timestep cosine similarity, accumu-

late temporal-enhanced multi-step similarity, and conduct similarity-based trajectory clustering.

Step 1: Compute Per-Timestep Similarity. Given a historical trajectory sequence $X = \{X_T, X_{T-1}, \dots, X_t\}$, where each $X_t \in \mathbb{R}^{N \times D}$, with N denoting the number of tokens and D representing the dimension, we first compute the pairwise cosine similarity between tokens at each timestep to obtain the similarity matrix S_t :

$$S_t = \frac{X_t X_t^T}{\|X_t\|_2 \|X_t^T\|_2}. \quad (1)$$

Step 2: Accumulate Temporal-Enhanced Similarity. We then apply a temporal moving average over S_t to obtain the cumulative similarity matrix $\tilde{S}_t$:

$$\tilde{S}_t = \alpha_S \cdot S_t + (1 - \alpha_S) \cdot \tilde{S}_{t+1}, \quad (2)$$

where α_S is a smoothing factor that controls the temporal weighting of recent timesteps.

Step 3: Conduct Similarity-based Trajectory Clustering. Since features in diffusion models are typically high-dimensional and sparse, we adopt a K-medoids clustering based on cosine distance:

$$\min_{\{C_1, C_2, \dots, C_K\}} \sum_{t=1}^N \sum_{k=1}^K \mathbb{I}(X_t \in C_k) \cdot (1 - \tilde{\mathcal{S}}_t(X_t, C_k)), \quad (3)$$

where $\mathbb{I}(X_i \in C_k)$ is an indicator function that equals 1 if the data point X_i belongs to the cluster C_k and 0 otherwise, and $\tilde{\mathcal{S}}_t(X_i, C_k)$ denotes the cosine similarity between X_i and the medoid of C_k under the cumulative similarity measure. The K-medoids algorithm constrains each cluster center to be an actual data point. We random select one token from C_k as the *proxy token*, denoted by p_k , where $p_k \in C_k$. The remaining data points in the cluster are referred to as *driven tokens*, denoted by $D_k = \{X_i \mid X_i \in C_k, X_i \neq p_k\}$, so that D_k represents all tokens in the cluster C_k except the proxy token and satisfies $D_k \cap \{p_k\} = \emptyset$.

4.3. Proxy-Driven Denoising Simulation

Based on Sec. 3, we observe that lower-order residuals exhibit higher reusability and that their utility can be estimated based on the preceding denoising trajectory. Instead of directly reusing the feature values of similar trajectories, we reuse their low-order residuals and weight them according to a confidence score derived from the historical relationships between trajectories. Building on this idea, we design a proxy-driven denoising simulation scheme that leverages a denoised trajectory to guide the denoising of the remaining tokens. As illustrated in Fig. 3 (c), the method consists of three main steps: denoising the proxy token, estimating residuals of the driven tokens, and update driven tokens using an implicit ODE.

Step 1: Denoise the Proxy Token. For each cluster C_k and each network layer l , we denote the proxy feature at timestep t by $p_{k,t}^l$. A reverse-time denoising step of the diffusion model updates this proxy token via $\mathcal{F}(\cdot)$:

$$p_{k,t}^l = \mathcal{F}(p_{k,t+1}^l, t+1). \quad (4)$$

For notational simplicity, we omit the cluster index k and layer index l in the following and write p_t instead of $p_{k,t}^l$. Then, we yield a discrete trajectory along the timesteps. To connect with an ODE-style description, we construct multi-order residuals through recursive finite differences. We first treat the zeroth-order term as the feature itself,

$$\mathcal{F}^{(0)}(p_t) = p_t, \quad (5)$$

and define higher-order terms by

$$\mathcal{F}^{(m)}(p_t) = \mathcal{F}^{(m-1)}(p_t) - \mathcal{F}^{(m-1)}(p_{t+1}), \quad m \geq 1. \quad (6)$$

The collection $\{\mathcal{F}^{(m)}(p_t)\}_{m=1}^M$ serves as a multi-order derivative-like descriptor of the proxy trajectory and will be used to estimate the residuals of the driven tokens.

Step 2: Estimate Residuals of Driven Tokens. For each group C_k , it contains one proxy token p_k and a set of driven

tokens $D_k = \{X_i \mid X_i \in C_k, X_i \neq p_k\}$. Let $d_{k,i,t}$ denote the feature of a driven token at the timestep t . For notational simplicity, in the following we omit the cluster k and token indices i and write d_t for the generic driven token. Using the multi-order residuals proxies $\{\mathcal{F}^{(m)}(\cdot)\}_{m=1}^M$ defined in Step 1, we first compute an order-specific confidence weight that measures how well the m -th order residuals of the two trajectories align at timestep t :

$$\theta_t^{(m)} = \max\left(0, \cos\left(\mathcal{F}^{(m)}(p_t), \mathcal{F}^{(m)}(d_t)\right)\right), \quad (7)$$

with $\theta_t^{(m)} \in [0, 1]$. We then estimate the m -th order residuals of the driven token at the next step $t+1$ by modulating its own residuals using the proxy residuals at t :

$$\hat{\mathcal{F}}^{(m)}(d_{t+1}) = (1 - \theta_t^{(m)}) \cdot \mathcal{F}^{(m)}(d_t) + \theta_t^{(m)} \cdot \mathcal{F}^{(m)}(p_{t+1}). \quad (8)$$

Here $\mathcal{F}^{(m)}(d_t)$ serves as the base residual direction of the driven token, while the term $\mathcal{F}^{(m)}(p_{t+1})$ encodes a temporal correction driven by the proxy trajectory at timestep $t+1$. The confidence $\theta_t^{(m)}$ controls how strongly this look-ahead correction is applied: when the trajectories are highly consistent, $\theta_t^{(m)}$ is close to 1 and the driven tokens' residuals are strongly aligned to the proxy residuals at $t+1$; otherwise, the update falls back to its own residuals.

Step 3: Update Driven Tokens Using an Implicit ODE.

From an ODE perspective, the driven token trajectory $d(t)$ evolves along a reverse-time dynamics. Instead of explicitly introducing a continuous drift function, we directly reuse the multi-order residuals $\{\hat{\mathcal{F}}^{(m)}(d_{t+1})\}_{m=1}^M$ obtained in Eq. 8 as discrete approximations of time derivatives at timestep $t+1$. Following the implicit Taylor method for ODEs and adopting its unit-step discrete form, we advance d_t to d_{t+1} via a multi-order implicit update:

$$d_{t+1} = d_t + \sum_{m=1}^M \frac{1}{m!} \hat{\mathcal{F}}^{(m)}(d_{t+1}). \quad (9)$$

When $M=1$, Eq. (9) reduces to an implicit Euler-style step,

$$d_{t+1} = d_t + \hat{\mathcal{F}}^{(1)}(d_{t+1}). \quad (10)$$

Moreover, the estimated multi-order residuals $\{\hat{\mathcal{F}}^{(m)}(d_{t+1})\}_{m=1}^M$ are compatible with standard implicit linear multistep schemes. Taking the BDF2 with unit step size as an example, we obtain the following BDF2-style implicit update for d_{t+1} :

$$d_{t+1} = \frac{4}{3}d_t - \frac{1}{3}d_{t-1} + \frac{2}{3}\hat{\mathcal{F}}^{(1)}(d_{t+1}), \quad (11)$$

where d_{t-1} is the driven token's feature at the previous timestep. Thus, our estimated multi-order residuals can be directly plugged into implicit high-order or multi-step ODE schemes without extra process of the diffusion model. Additional implementation details and discussion are provided in the *Supplementary Materials*.

Method FLUX.1	Efficient Attention	Acceleration				Image Reward $\uparrow$ DrawBench	CLIP $\uparrow$ Score
		Latency(s) $\downarrow$	Speed $\uparrow$	FLOPs(T) $\downarrow$	Speed $\uparrow$		
[dev]: 50 steps	✓	25.82	1.00×	3719.5	1.00×	0.9898	19.761
34% steps	✓	9.07	2.85×	1264.6	3.13×	0.9453	18.870
Δ -DiT ($\mathcal{N} = 3$)	✓	13.02	1.98×	1686.8	2.21×	0.8721	18.742
Chipmunk	✓	12.72	2.02×	1505.9	2.47×	0.9936	19.534
FORA ($\mathcal{N} = 3$)	✓	10.16	2.54×	1320.1	2.82×	0.9776	19.339
ToCa ($\mathcal{N} = 6$)	✗	13.16	1.96×	924.3	4.02×	0.9802	18.688
DuCa ($\mathcal{N} = 5$)	✓	8.18	3.15×	978.8	3.80×	0.9955	19.314
TaylorSeer ($\mathcal{N} = 4, O = 2$)	✓	9.24	2.80×	1042.3	3.57×	0.9857	19.496
ClusCa ($\mathcal{N} = 5, O = 1, K = 16$)	✓	8.12	3.18×	897.0	4.14×	0.9825	19.481
ResCa-IE ($\mathcal{N} = 5, K = 16$)	✓	8.19	3.15×	898.1	4.14×	0.9958	19.537
ResCa-IT ($\mathcal{N} = 6, O = 2, K = 16$)	✓	7.17	3.60×	749.5	4.96×	0.9937	19.452
22% steps	✓	6.04	4.28×	818.3	4.55×	0.8183	18.224
DuCa ($\mathcal{N} = 7$)	✓	6.74	3.83×	760.1	4.89×	0.9757	18.962
TeaCache ($l = 0.8$)	✓	7.21	3.58×	892.4	4.17×	0.8683	18.500
TaylorSeer ($\mathcal{N} = 5, O = 2$)	✓	7.46	3.46×	893.5	4.16×	0.9864	19.406
ClusCa ($\mathcal{N} = 6, O = 1, K = 16$)	✓	7.10	3.63×	748.5	4.96×	0.9762	19.533
ResCa-IE ($\mathcal{N} = 7, K = 16$)	✓	6.79	3.80×	675.1	5.51×	0.9737	19.336
ResCa-IB ($\mathcal{N} = 7, K = 16$)	✓	6.82	3.79×	675.2	5.51×	0.9889	19.441
TeaCache ($l = 1.2$)	✓	6.12	4.22×	669.3	5.56×	0.7394	18.432
TaylorSeer ($\mathcal{N} = 7, O = 2$)	✓	6.38	4.05×	670.4	5.54×	0.9128	18.983
ClusCa ($\mathcal{N} = 7, O = 2, K = 32$)	✓	7.06	3.66×	678.4	5.48×	0.9364	18.994
ResCa-IT ($\mathcal{N} = 8, O = 2, K = 32$)	✓	6.54	3.95×	604.6	6.15×	0.9468	19.039

Table 1. Quantitative comparison in text-to-image generation for FLUX on Image Reward. The best results are highlighted in bold.

5. Experiments

5.1. Experiments Settings

Implementation details. Our ResCa framework includes three versions: IE, IB, and IT, which represent the implicit Euler, implicit BDF2, and implicit Taylor methods, respectively. For the implicit Euler and implicit BDF2, the order is fixed at 1, while the other methods depend on O . The cache interval is set to N , and the number of clusters is K .

Evaluations and Metrics. We evaluated the effectiveness of our proposed method across several commonly used vision generation models. For class-conditional image generation with DiT-XL/2 [30], we uniformly sampled 50,000 images from all 1,000 ImageNet categories at a resolution of 256×256 , using FID-50k [14] as the primary evaluation metric and sFID for supplementary analysis. For text-to-image generation with FLUX.1-dev [19], we utilized 200 prompts from DrawBench [31] and evaluated image quality using Image Reward [40], while assessing the alignment between text and image using the CLIP score [13]. In the case of text-to-video generation with Hunyuan-Video [18], we employed the VBench [15] evaluation framework, generating five videos per prompt across 946 distinct prompts, each with different random seeds. Detailed configurations of the models are provided in the *Supplementary Materials*.

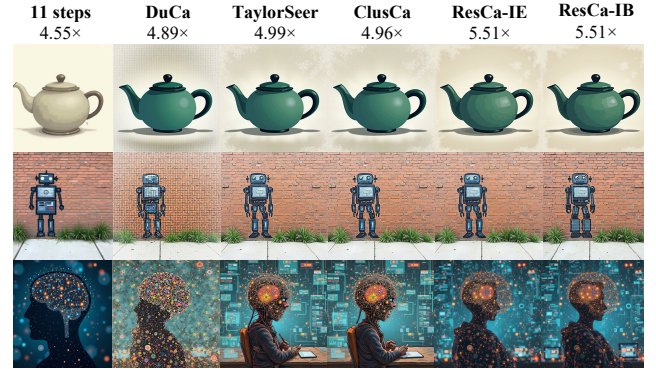


Figure 4. Qualitative comparison in text-to-image generation on FLUX. ResCa show stronger texture details in the kettle reflection, robotic limbs, and human brain, with a $5.51\times$ acceleration.

5.2. Results on Text-to-Image Generation

We compare our ResCa with SoTA caching approaches in Table 1. When $N = 5$ and $O = 1$, ResCa-IE achieves the best performance with an Image Reward of 0.9958 and a Clip Score of 19.537, outperforming ClusCa at $N = 5$ and $O = 1$. This highlights that first-order implicit Euler predictions, based on future residuals estimation, achieve higher accuracy than first-order mixed methods. When $N = 6$ and $O = 2$, the speedup ratio further improves to $4.96\times$. Additionally, at $N = 7$, implicit BDF2 reaches an Image

Method	Efficient Attention	Acceleration				VBench $\uparrow$ Score(%)
		Latency(s) $\downarrow$	Speed $\uparrow$	FLOPs(T) $\downarrow$	Speed $\uparrow$	
Original	✓	334.96	1.00 $\times$	29773.0	1.00 $\times$	80.66
22% steps	✓	87.01	3.85 $\times$	6550.1	4.55 $\times$	78.74
FORA ($\mathcal{N} = 5$)	✓	83.78	4.00 $\times$	5960.4	5.00 $\times$	78.83
ToCa ($\mathcal{N} = 5$)	✗	93.80	3.57 $\times$	7006.2	4.25 $\times$	78.86
DuCa ($\mathcal{N} = 5$)	✓	87.48	3.83 $\times$	6483.2	4.62 $\times$	78.72
TeaCache ($l = 0.4$)	✓	70.43	4.76$\times$	6550.1	4.55 $\times$	79.36
TaylorSeer ($\mathcal{N} = 6, O = 1$)	✓	79.46	4.22 $\times$	5359.1	5.56 $\times$	79.78
ClusCa ($\mathcal{N} = 7, K = 32$)	✓	74.88	4.47 $\times$	4796.2	6.21$\times$	79.60
ResCa-IE ($\mathcal{N} = 6, K = 32$)	✓	83.42	4.02 $\times$	5383.2	5.53 $\times$	79.98
ResCa-IT ($\mathcal{N} = 7, O = 2, K = 64$)	✓	75.04	4.46 $\times$	4814.1	6.19 $\times$	79.78

Table 2. **Quantitative comparison in text-to-video generation** for HunyuanVideo on VBench. Best results are highlighted in **bold**.

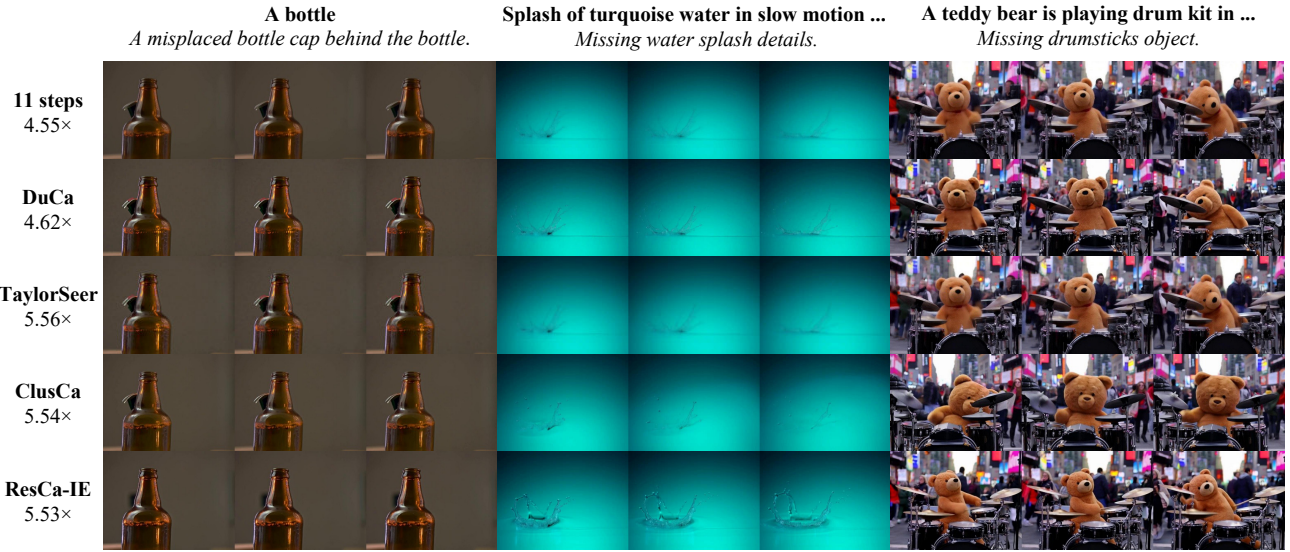


Figure 5. **Qualitative comparison in text-to-video generation** on HunyuanVideo. ResCa generates high-quality, semantically aligned videos, while others suffer from issues such as misplaced bottle caps, missing water splash details, and missing drumstick objects.

Reward of 0.9889, demonstrating the compatibility of our method with multi-step ODEs. Even at extreme intervals ($N = 8$), ResCa maintains high-quality generation, showcasing its adaptability to long-step caching. As shown in Fig. 4, our qualitative comparison reveals that ResCa outperforms other methods, delivering enhanced texture details in the kettle reflection, robotic limbs, and human brain, further proving its superiority in high-speed image generation.

5.3. Results on Text-to-Video Generation

In Table 2, we report the comparison of text-to-video generation. With $N = 6$ and $K = 32$, ResCa-IE attains the best VBench score of 79.98, outperforming TaylorSeer with a comparable speedup at $N = 6, O = 1$ by about 0.2%. When $N = 7$, we adopt $O = 2$ and $K = 64$ to enhance the modeling capacity; under a 6.19 $\times$ FLOPs speedup, ResCa-IT surpasses ClusCa with $N = 7, K = 64$ by approximately 0.18% on VBench. These results show

that ResCa delivers higher video generation quality than previous SoTA methods under comparable acceleration settings. We present a qualitative comparison in Fig. 5, where the results show that our ResCa generates high-quality, semantically aligned videos, while other methods suffer from issues such as misplaced bottle caps, missing water splash details, and missing drumstick objects.

5.4. Results on Class-to-Image Generation

In the quantitative comparison of class-conditional image generation (shown in Table 3), ResCa shows a superior trade-off between generation quality and acceleration. For instance, when $N = 3, K = 16$, ResCa-IE achieves the best FID of 2.37 and sFID of 4.63. When $N = 4, K = 16$, ResCa-IE achieves a FID of 2.49, outperforming TaylorSeer with a higher speedup (3.23 $\times$ v.s. 2.77 $\times$). Additionally, when $N = \{5, 6\}, K = 16$, ResCa-IE achieves FID reductions of 0.03 and 0.08 compared to ClusCa, despite similar

Method	Latency(s) ↓	FLOPs(T) ↓	Speed ↑	FID ↓	sFID ↓
DDIM-50 steps	0.428	23.74	1.00×	2.32	4.32
DDIM-25 steps	0.230	11.87	2.00×	3.18	4.74
FORA ($\mathcal{N} = 2$)	0.278	12.35	1.92×	2.66	4.88
ToCa ($\mathcal{N} = 3$)	0.216	9.73	2.44×	2.87	4.76
DuCa ($\mathcal{N} = 3$)	0.208	9.54	2.49×	2.85	4.64
ResCa-IE ($\mathcal{N} = 3, K = 16$)	0.234	9.20	2.58×	2.37	4.63
DDIM-20 steps	0.191	9.49	2.50×	3.81	5.15
FORA ($\mathcal{N} = 3$)	0.222	8.58	2.77×	3.55	6.36
ToCa ($\mathcal{N} = 4$)	0.197	8.10	2.93×	3.42	5.12
DuCa ($\mathcal{N} = 4$)	0.175	7.61	3.12×	3.42	4.94
TaylorSeer ($\mathcal{N} = 3, O = 1$)	0.215	8.56	2.77×	2.49	4.81
ResCa-IE ($\mathcal{N} = 4, K = 16$)	0.191	7.36	3.23×	2.49	4.99
DDIM-15 steps	0.151	7.12	3.33×	5.17	6.11
ToCa ($\mathcal{N} = 5$)	0.176	6.77	3.51×	6.20	7.17
DuCa ($\mathcal{N} = 5$)	0.152	6.27	3.79×	6.06	6.72
TaylorSeer ($\mathcal{N} = 4, O = 1$)	0.186	6.66	3.56×	2.71	5.45
ClusCa ($\mathcal{N} = 5, K = 16$)	0.166	5.98	3.97×	2.65	5.13
ResCa-IE ($\mathcal{N} = 5, K = 16$)	0.168	5.99	3.96×	2.62	5.08
DDIM-12 steps	0.128	5.70	4.16×	7.80	8.03
FORA ($\mathcal{N} = 5$)	0.171	5.24	4.53×	6.58	11.29
DuCa ($\mathcal{N} = 6$)	0.145	5.81	4.08×	6.40	6.71
ClusCa ($\mathcal{N} = 6, K = 16$)	0.159	5.52	4.29×	3.16	5.65
ResCa-IE ($\mathcal{N} = 6, K = 16$)	0.160	5.53	4.29×	3.08	5.49

Table 3. **Quantitative comparison in class-to-image generation** for DiT-XL/2 on ImageNet. Best results are highlighted in **blod**.

speedup ratios. These results demonstrate that, compared to explicit methods relying on historical state reusing or forecasting, our implicit ODE-based method, which estimates future states, consistently delivers superior generation performance under comparable or higher acceleration ratios.

5.5. Ablation Studies

We conduct ablation studies to evaluate the key components of our method, as described below. Additional experiments and analyses can be found in the *Supplementary Material*.

Effect of Number of Clusters. The number of clusters directly determines the similarity of residuals within each cluster, and thus plays an important role in the overall performance. Under the FLUX configuration with $N = 8$ and $O = 1$, we further analyze the relationship between the number of clusters and the influence on the generated results, as shown in Fig. 6. The results suggest that the best trade-off is achieved when $K = 16$ or $K = 32$. Taking both computational efficiency and generation quality into account, we use $K = 16$ as the default setting.

Effect of Clustering Methods. We compare the effects of feature-based clustering and trajectory-based clustering in Fig. 7. To ensure a clearer comparison, we set $N = 8$ to amplify the influence of residual reuse. The results indicate that feature-based clustering causes noticeable quality degradation in the generated images, including overexposure and missing details. This degradation arises because less similar residuals are reused, which introduces erroneous guidance into the denoising process.

Effect of Multi-order Residuals We compare the effects

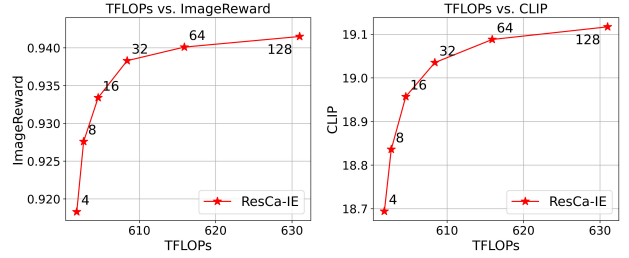


Figure 6. **Ablation on the number of clusters.**

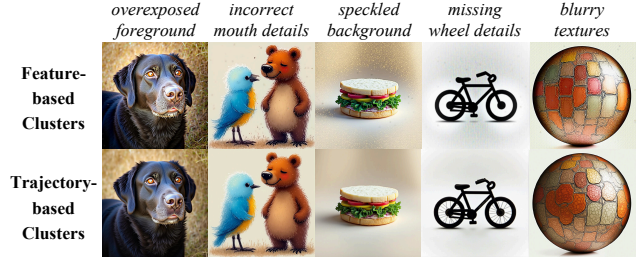


Figure 7. **Visual comparisons of clustering methods.**

Method	Latency(s) ↓	FLOPs(T) ↓	FID ↓	sFID ↓
DDIM-15 steps	0.193	6.66	4.75	8.43
ResCa-IT ($\mathcal{N} = 5, O = 1$)	0.168	5.99	2.62	5.08
ResCa-IT ($\mathcal{N} = 5, O = 2$)	0.170	6.00	2.57	4.98
ResCa-IT ($\mathcal{N} = 5, O = 3$)	0.171	6.00	2.58	5.02
ResCa-IT ($\mathcal{N} = 5, O = 4$)	0.171	6.01	2.58	5.00

Table 4. **Ablation on the multi-order residuals.**

of multi-order residuals on the results, as shown in Table 4. The results reveal that using first-order and second-order residuals significantly improves FID. However, increasing the order leads to a slight decrease in FID while increasing the computational cost. Therefore, we use first-order residuals to maintain the simplicity by default.

6. Conclusion

In this paper, we introduce Residual Caching (ResCa), a novel framework that accelerates diffusion transformers from a “proxy denoising” perspective. Specifically, ResCa perform true denoising on one proxy token in each cluster to obtain its residuals at the next timestep, which are then used to guide the simulated denoising of the remaining tokens. This process is realized through the Temporal-Enhanced Trajectory Clustering and Proxy-Driven Denoising Simulation modules. ResCa enables stable and adaptive denoising, and can be seamlessly integrated into a wide range of diffusion models, including DiT, FLUX, and HunyuanVideo. Extensive quantitative and qualitative experiments demonstrate the effectiveness of our method. In particular, ResCa achieving up to a $5.5\times$ acceleration in GFLOPs while maintaining near-lossless generation quality on FLUX. We hope this work can inspire further exploration of the “proxy denoising” paradigm in feature caching for diffusion models.

Acknowledgements. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 62572458, in part by the Beijing Major Science and Technology Project under No. Z251100008125009, and in part by the Innovation Funding of the Institute of Computing Technology, Chinese Academy of Sciences under Grant No. E561090.

References

- [1] Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion. In *IEEE Conf. Comput. Vis. Pattern Recog. Worksh.*, pages 4599–4603, 2023. 1, 2, 3
- [2] Thibault Castells, Hyoung-Kyu Song, Bo-Kyeong Kim, and Shinkook Choi. LD-Pruner: Efficient pruning of latent diffusion models using task-agnostic insights. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 821–830, 2024. 1, 2
- [3] Defang Chen, Zhenyu Zhou, Can Wang, Chunhua Shen, and Siwei Lyu. On the trajectory regularity of ode-based diffusion sampling. In *Int. Conf. Mach. Learn.*, pages 7905–7934, 2024. 3
- [4] Junsong Chen, YU Jincheng, GE Chongjian, Lewei Yao, Enze Xie, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, and Zhenguo Li. PixArt- α : Fast training of diffusion transformer for photorealistic text-to-image synthesis. In *Int. Conf. Learn. Represent.*, 2024. 1
- [5] Pengtao Chen, Mingzhu Shen, Peng Ye, Jianjian Cao, Chongjun Tu, Christos-Savvas Bouganis, Yiren Zhao, and Tao Chen. Δ -DiT: A training-free acceleration method tailored for diffusion transformers. *arXiv preprint arXiv:2406.01125*, 2024. 2
- [6] Xinle Cheng, Zhuoming Chen, and Zhihao Jia. CAT Pruning: Cluster-aware token pruning for text-to-image diffusion models. *arXiv preprint arXiv:2502.00433*, 2025. 1, 2
- [7] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Int. Conf. Mach. Learn.*, 2024. 1
- [8] Gongfan Fang, Xinyin Ma, and Xinchao Wang. Structural pruning for diffusion models. In *Adv. Neural Inform. Process. Syst.*, 2023. 1, 2
- [9] Haipeng Fang, Sheng Tang, Juan Cao, Enshuo Zhang, Fan Tang, and Tong-Yee Lee. Attend to Not Attended: Structure-then-detail token merging for post-training dit acceleration. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 18083–18092, 2025. 1, 2
- [10] Haipeng Fang, Sheng Tang, Zhihao Sun, Ziyao Huang, Juan Cao, Fan Tang, and Yongdong Zhang. MoAnimate: Bridging the motion-oriented latent representation gaps in human video animation. *IEEE Trans. Circuit Syst. Video Technol.*, 2025. 1
- [11] Haipeng Fang, Ziheng Wu, Xinyi Zou, Jun Huang, Juan Cao, and Sheng Tang. FR²ViT: Finetuning-free token reduction for dense prediction through a refinement-reactivation architecture. In *IEEE Int. Conf. Acoust., Speech, Signal Process.*, pages 1–5, 2025. 1
- [12] Yefei He, Luping Liu, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. PTQD: Accurate post-training quantization for diffusion models. *Adv. Neural Inform. Process. Syst.*, 36: 13237–13249, 2023. 1, 2
- [13] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. In *Conf. Empir. Methods Nat. Lang. Process.*, pages 7514–7528, 2021. 6
- [14] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Adv. Neural Inform. Process. Syst.*, 30, 2017. 6
- [15] Ziqi Huang, Yanan 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 21807–21818, 2024. 6
- [16] Bo-Kyeong Kim, Hyoung-Kyu Song, Thibault Castells, and Shinkook Choi. BK-SDM: A lightweight, fast, and cheap version of stable diffusion. In *Eur. Conf. Comput. Vis.*, pages 381–399, 2024. 1, 2
- [17] Sanghwan Kim, Hao Tang, and Fisher Yu. Distilling ODE solvers of diffusion models into smaller steps. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9410–9419, 2024. 2
- [18] 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. 1, 6
- [19] Black Forest Labs. Flux. <https://github.com/black-forest-labs/flux>, 2024. 1, 2, 6
- [20] Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-Diffusion: Quantizing diffusion models. In *Int. Conf. Comput. Vis.*, pages 17535–17545, 2023. 2
- [21] 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. *Adv. Neural Inform. Process. Syst.*, 36: 20662–20678, 2023. 1, 2
- [22] Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep embedding tells: It’s time to cache for video diffusion model. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 7353–7363, 2025. 2
- [23] Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From reusing to forecasting: Accelerating diffusion models with taylorseers. In *Int. Conf. Comput. Vis.*, pages 15853–15863, 2025. 2
- [24] Jinming Lou, Wenyang Luo, Yufan Liu, Bing Li, Xinmiao Ding, Weiming Hu, Yuming Li, and Chenguang Ma. Token caching for diffusion transformer acceleration. *arXiv preprint arXiv:2409.18523*, 2024. 1, 2
- [25] 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. *Adv. Neural Inform. Process. Syst.*, 35:5775–5787, 2022. 1, 2

- [26] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver++: Fast solver for guided sampling of diffusion probabilistic models. *Mach. Intell. Res.*, 22(4):730–751, 2025. 2
- [27] Wenbo Lu, Shaoyi Zheng, Yuxuan Xia, and Shengjie Wang. ToMA: Token merge with attention for diffusion models. In *Int. Conf. Mach. Learn.*, 2025. 1, 2
- [28] Xinyin Ma, Gongfan Fang, and Xinchao Wang. DeepCache: Accelerating diffusion models for free. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 15762–15772, 2024. 1, 2
- [29] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 14297–14306, 2023. 1, 2
- [30] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Int. Conf. Mach. Learn.*, pages 4195–4205, 2023. 1, 6
- [31] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Adv. Neural Inform. Process. Syst.*, 35:36479–36494, 2022. 6
- [32] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *Int. Conf. Learn. Represent.*, 2022. 1, 2
- [33] 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. 2
- [34] Junhyuk So, Jungwon Lee, Daehyun Ahn, Hyungjun Kim, and Eunhyeok Park. Temporal dynamic quantization for diffusion models. *Adv. Neural Inform. Process. Syst.*, 36:48686–48698, 2023. 2
- [35] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *Int. Conf. Learn. Represent.*, 2021. 1, 2
- [36] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Int. Conf. Mach. Learn.*, pages 32211–32252, 2023. 1, 2
- [37] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 1
- [38] Hongjie Wang, Difan Liu, Yan Kang, Yijun Li, Zhe Lin, Niraj K Jha, and Yuchen Liu. Attention-driven training-free efficiency enhancement of diffusion models. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 16080–16089, 2024. 2
- [39] 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6211–6220, 2024. 1, 2
- [40] Jiazhen Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. ImageReward: Learning and evaluating human preferences for text-to-image generation. *Adv. Neural Inform. Process. Syst.*, 36:15903–15935, 2023. 6
- [41] Yu Xu, Fan Tang, Juan Cao, Xiaoyu Kong, Yuxin Zhang, Jintao Li, Oliver Deussen, and Tong-Yee Lee. HeadRouter: A training-free image editing framework for mm-dits by adaptively routing attention heads. *ACM Trans. Graph.*, 2024. 1
- [42] Yu Xu, Fan Tang, Juan Cao, Yuxin Zhang, Oliver Deussen, Weiming Dong, Jintao Li, and Tong-Yee Lee. B4M: Breaking low-rank adapter for making content-style customization. *ACM Trans. Graph.*, 44(2):1–17, 2025. 1
- [43] Yu Xu, Fan Tang, You Wu, Lin Gao, Oliver Deussen, Hongbin Yan, Jintao Li, Juan Cao, and Tong-Yee Lee. In-Context Brush: Zero-shot customized subject insertion with context-aware latent space manipulation. In *SIGGRAPH Asia Conf. Pap.*, pages 1–12, 2025. 1
- [44] Yu Xu, Hongbin Yan, Juan Cao, Yiji Cheng, Tiankai Hang, Runze He, Zijin Yin, Shiyi Zhang, Yuxin Zhang, Jintao Li, et al. TAG-MoE: Task-aware gating for unified generative mixture-of-experts. *arXiv preprint arXiv:2601.08881*, 2026. 1
- [45] Ziyi Xu, Ziyao Huang, Juan Cao, Yong Zhang, Xiaodong Cun, Qing Shuai, Yuchen Wang, Linchao Bao, and Fan Tang. Anchorcrafter: Animate cyber-anchors selling your products via human-object interacting video generation. *ACM Trans. Graph.*, 2026. 1
- [46] Tianchen Zhao, Tongcheng Fang, Haofeng Huang, Rui Wan, Widyadewi Soedarmadji, Enshu Liu, Shiyao Li, Zinan Lin, Guohao Dai, Shengen Yan, et al. ViDiT-Q: Efficient and accurate quantization of diffusion transformers for image and video generation. In *Int. Conf. Learn. Represent.*, 2025. 2
- [47] Wangbo Zhao, Yizeng Han, Jiasheng Tang, Kai Wang, Yibing Song, Gao Huang, Fan Wang, and Yang You. Dynamic diffusion transformer. In *Int. Conf. Learn. Represent.*, 2025. 2
- [48] Kaiwen Zheng, Cheng Lu, Jianfei Chen, and Jun Zhu. Dpm-Solver-v3: Improved diffusion ode solver with empirical model statistics. *Adv. Neural Inform. Process. Syst.*, 36:55502–55542, 2023. 2
- [49] Shikang Zheng, Liang Feng, Xinyu Wang, Qinming Zhou, Peiliang Cai, Chang Zou, Jiacheng Liu, Yuqi Lin, Junjie Chen, Yue Ma, et al. Forecast then calibrate: Feature caching as ode for efficient diffusion transformers. In *AAAI Conf. Artif. Intell.*, pages 13449–13457, 2026. 2
- [50] Zhixin Zheng, Xinyu Wang, Chang Zou, Shaobo Wang, and Linfeng Zhang. Compute only 16 tokens in one timestep: Accelerating diffusion transformers with cluster-driven feature caching. In *ACM Int. Conf. Multimedia*, pages 10181–10189, 2025. 2
- [51] Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. In *Int. Conf. Learn. Represent.* 1, 2
- [52] Chang Zou, Evelyn Zhang, Runlin Guo, Haohang Xu, Conghui He, Xuming Hu, and Linfeng Zhang. Accelerating diffusion transformers with dual feature caching. *arXiv preprint arXiv:2412.18911*, 2024. 1, 2