SpeCa: Accelerating Diffusion Transformers with Speculative Feature Caching

Anonymous Author(s)

Abstract

Diffusion models have revolutionized high-fidelity image and video synthesis, yet their computational demands remain prohibitive for real-time applications. These models face two fundamental challenges: strict temporal dependencies preventing parallelization, and computationally intensive forward passes required at each denoising step. Drawing inspiration from speculative decoding in large language models, we present *SpeCa*, a novel "***Forecast-then-verify***" acceleration framework that effectively addresses both limitations. *SpeCa*'s core innovation lies in introducing Speculative Sampling to diffusion models, predicting intermediate features for subsequent timesteps based on fully computed reference timesteps. Our approach implements a parameter-free verification mechanism that efficiently evaluates prediction reliability, enabling real-time decisions to accept or reject each prediction while incurring negligible computational overhead. Furthermore, *SpeCa* introduces sample-adaptive computation allocation that dynamically modulates resources based on generation complexity—allocating reduced computation for simpler samples while preserving intensive processing for complex instances. Experiments demonstrate 6.34× acceleration on FLUX with minimal quality degradation (5.5% drop), 7.3× speedup on DiT while preserving generation fidelity, and 79.84% VBench score at 6.1× acceleration for HunyuanVideo. The verification mechanism incurs minimal overhead (1.67%-3.5% of full inference costs), establishing a new paradigm for efficient diffusion model inference while maintaining generation quality even at aggressive acceleration ratios. Codes are available in the supplementary material and will be released in Github.

CCS Concepts

• **Generative Multimedia → Multimedia Foundation Models**.

Keywords

Diffusion Models, Feature Cache, Speculative Sampling

1 Introduction

Diffusion Models (DMs) [12] have established themselves as a dominant force in generative AI, achieving state-of-the-art performance across image synthesis [35] and video generation [1]. While architectural breakthroughs like Diffusion Transformers (DiT) [32] have substantially enhanced generation fidelity through scalable transformer architectures, they remain constrained by an intrinsic limitation of diffusion-based approaches: the sequential dependency in sampling. Each generation requires executing dozens to hundreds of denoising steps, with each step necessitating a complete forward pass through increasingly complex models. This computational paradigm creates an escalating tension between generation quality and inference efficiency - a dilemma that becomes particularly acute as model sizes expand and generation tasks grow more sophisticated. For instance, modern video generation architectures like HunyuanVideo require 595.46 TFLOPs per forward pass (480p, 2s, 50steps), making real-time generation computationally prohibitive.

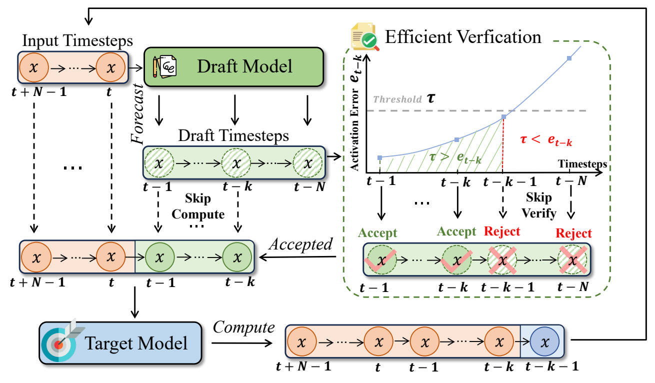

Figure 1: SpeCa's speculative execution workflow. The draft model forecasts N future timesteps ($t-1$ to $t-N$), followed by lightweight error verification across these steps. Predictions are accepted sequentially until a rejection occurs at $t-k$ (when activation error e_{t-k} exceeds threshold τ). Accepted steps ($t-1$ to $t-k$) are appended to inputs, while the target model recomputes $t-k-1$ to recover trajectory fidelity.

The computational challenge originates from two fundamental characteristics of diffusion sampling: (1) **Strict temporal dependencies** enforcing step-by-step execution that prohibits parallelization, and (2) **Full-model forward passes** required at each timestep, becoming prohibitively expensive for modern architectures. While reduced-step samplers like DDIM [43] attempt to address the first challenge through non-Markovian processes, they still face an inescapable trade-off - aggressive step reduction (e.g., from 1000 to 50 steps) inevitably degrades output quality due to trajectory truncation. Recent caching acceleration methods have emerged to tackle primarily the second challenge, with approaches exploring different perspectives - token-based methods [40], [54], [55] and residual-based approaches [6] focus on reusing computations across timesteps to reduce per-step computational burden. However, these methods are inherently limited by their reliance on feature similarity between adjacent timesteps, with efficacy rapidly diminishing as acceleration ratios increase. Methods like TaylorSeer [24] attempt to leverage temporal dependencies more effectively through Taylor series approximations to predict multi-step evolution, but critically

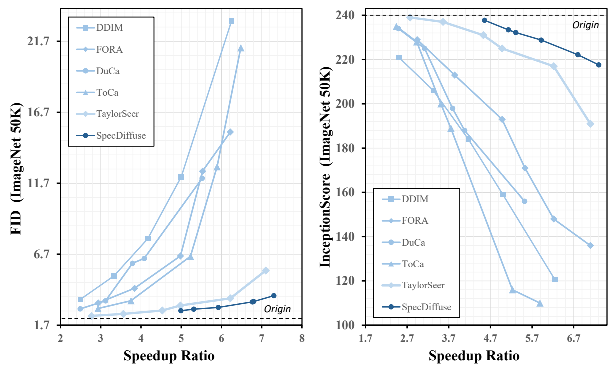

Figure 2: Comparison of caching methods on Inception Score (IS) and FID. *SpeCa* outperforms previous methods, particularly at high-acceleration ratios.

lack error correction mechanisms. This absence of validation allows prediction errors to compound exponentially, particularly at high acceleration ratios where minor inaccuracies in early steps catastrophically distort the generation trajectory. Furthermore, existing approaches uniformly apply the same acceleration ratio to all samples and maintain fixed sampling intervals, resulting in computational inefficiency where simple samples receive excessive computation while complex samples remain underserved.

Drawing inspiration from the success of Speculative Decoding in large language models, we present *SpeCa*, an acceleration framework based on a "***Forecast-then-verify***" mechanism. Unlike traditional step-by-step inference, *SpeCa* adopts a speculative sampling paradigm that implements efficient inference through a three-step core workflow: (1) performing complete forward computations at strategically selected key timesteps to obtain accurate feature representations; (2) predicting features for multiple subsequent timesteps based on these high-fidelity representations; and (3) evaluating the reliability of each predicted feature through a rigorous error verification mechanism at each timestep, dynamically deciding whether to accept or reject predictions. This approach not only addresses the limitations of existing caching methods but also introduces adaptive computation allocation based on sample complexity, providing higher acceleration for simpler samples while ensuring complex samples receive sufficient computational resources to maintain high-fidelity generation quality.

Through extensive experimental evaluation across multiple model architectures and generation tasks, we demonstrate that *SpeCa* significantly outperforms existing acceleration methods in both efficiency and output fidelity. On FLUX.1-dev model, our approach maintains remarkable generation quality with only a 5.5% quality degradation at 6.34× acceleration, while the current SOTA method TaylorSeer suffers a substantial 17.5% quality loss at the same acceleration ratio. Similarly, for DiT model, where competing methods exhibit catastrophic quality deterioration beyond 3× acceleration, *SpeCa* preserves high-fidelity generation even at an impressive 7.3× acceleration ratio. Furthermore, when applied to the computationally intensive HunyuanVideo architecture, our method achieves 6.16× acceleration while maintaining SOTA Vbench scores 79.84%. Notably, these significant performance gains are achieved with an extremely **lightweight verification mechanism that consumes merely 3.5% (DiT), 1.75% (FLUX), and 1.67% (HunyuanVideo) of the complete forward pass computation**, yet effectively prevents error accumulation even at aggressive acceleration ratios.

In summary, our contributions are as follows:

- ***SpeCa* Framework**: We propose the ***"Forecast-then-verify"*** acceleration framework for diffusion models, *SpeCa*, inspired by Speculative Decoding in large language models, addressing the critical efficiency bottleneck in diffusion model inference. This framework, through precise forward prediction and a rigorous lightweight verification mechanism, transcends the theoretical limitations of traditional acceleration methods, resolving the quality collapse issue at high acceleration ratios.
- **Sample-adaptive Computation Allocation**: *SpeCa* dynamically allocates computational steps by sample complexity. **On HunyuanVideo**, it achieves computation reduction for low-error samples (57.5% cases accelerated 6.48×) while reserving resources for complex cases (42.5% cases accelerated 5.82×). This model-specific distribution enables **1% VBench quality drop** compared to full computation, outperforming fixed-step methods in quality-efficiency tradeoffs.
- **State-of-the-Art Performance**: Extensive experiments across various model architectures (DiT, FLUX, HunyuanVideo) demonstrate that *SpeCa* significantly outperforms existing acceleration methods. Notably, it achieves only a 5.5% quality degradation at 6.34× acceleration on FLUX.1-dev; maintains high-quality generation even at 7.3× acceleration on DiT; and sustains a SOTA Vbench score of 79.84% on the computationally intensive HunyuanVideo at 6.1× acceleration. These results establish *SpeCa* as a new benchmark for efficient diffusion model inference.

2 Related Works

Diffusion models [13, 42] have shown exceptional capabilities in image and video generation. Early architectures, primarily based on U-Net [36], faced scalability limitations that hindered large model training and deployment. The introduction of Diffusion Transformer (DiT) [33] overcame these issues, leading to significant advancements and state-of-the-art performance across various domains [4, 5, 49, 52]. However, the sequential sampling process in diffusion models remains computationally demanding, prompting the development of acceleration techniques.

2.1 Sampling Timestep Reduction

A key approach to accelerating diffusion models is *minimizing sampling steps while preserving output quality.* DDIM [43] introduced a deterministic sampling method that reduced denoising iterations without sacrificing fidelity. The DPM-Solver series [27, 28, 51] further advanced this with high-order ODE solvers. Other strategies, such as Rectified Flow [25] and knowledge distillation [30, 39], reduce the number of denoising steps. Consistency Models [44] enable few-step sampling by directly mapping noisy inputs to clean data, removing the need for sequential denoising.

2.2 Denoising Network Acceleration

To accelerate inference, *optimizing the denoising network's computational efficiency* is crucial. This can be classified into *Model Compression-based* and *Feature Caching-based* techniques.

Model Compression-based Acceleration. Model compression techniques, including network pruning [9, 53], quantization [16, 20, 41], knowledge distillation [21], and token reduction [2, 15], aim to reduce model complexity while maintaining performance. These methods require retraining or fine-tuning to minimize quality loss and achieve faster inference, though they often involve trade-offs, reducing model size at the cost of expressive power and accuracy.

Feature Caching-based Acceleration. Feature caching is particularly useful for DiT models [19, 29]. Techniques like FORA [40] and Δ-DiT [6] reuse attention and MLP representations, while TeaCache [23] dynamically estimates timestep-dependent differences. DiTFastAttn [50] reduces redundancies in self-attention, and ToCa [54] updates features dynamically. EOC [34] optimizes using prior knowledge. Innovations like UniCP [45] and RAS [26] further improve efficiency. However, existing methods rely on a "*cache-then-reuse*" paradigm, which loses effectiveness as timestep gaps grow. TaylorSeer [24] introduced a "*cache-then-forecast*" paradigm, predicting future features, but lacks mechanisms to verify prediction accuracy, which may lead to error accumulation.

2.3 Speculative Sampling

Speculative decoding has emerged as an effective approach for accelerating large language models (LLMs) while preserving output quality. The core idea, introduced by Leviathan et al.[18], employs a draft-then-verify mechanism where a smaller model proposes candidate tokens that are efficiently verified in parallel by the main model. Subsequent research has refined this paradigm: SpecInfer[31] introduced adaptive draft lengths, while Medusa [3] enhanced parallelism through multiple decoding heads. Recent advances have expanded the efficiency frontier of speculative methods. SpecTr [47] incorporated optimal transport theory for improved batch verification, and Sequoia [7] developed hardware-optimized tree-based verification. These approaches effectively leverage parallel computation to circumvent sequential dependencies. The speculative paradigm has demonstrated versatility beyond text generation domains. SpecVidGen [38] successfully adapted these techniques to video and image generation tasks, respectively, establishing the broader applicability of parallel prediction and verification mechanisms across different generative modalities.

Despite these advances, applying speculative sampling to diffusion models presents unique challenges due to the continuous nature of diffusion states and complex feature dependencies across timesteps. Current diffusion acceleration methods primarily rely on caching approaches, which often struggle with error accumulation and quality degradation. This gap motivates our work, which introduces *SpeCa*, a novel framework that bridges feature prediction with speculative principles through a coherent "*forecast-then-verify*" mechanism, enabling efficient validation and dynamic acceptance of predicted features across multiple timesteps.

3 Method

3.1 Preliminary

3.1.1 Diffusion Models. Diffusion models generate structured data by progressively transforming noise into meaningful data through iterative denoising steps. The core mechanism models the conditional probability distribution at each timestep as a Gaussian. Specifically, the model predicts the mean and variance for x_{t-1} given x_t at timestep t. The process is expressed as:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}\left(x_{t-1}; \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\tau_\theta(x_t,t)\right), \beta_t \mathbf{I}\right), \quad (1)$$

where $\mathcal{N}$ denotes a normal distribution, α_t and β_t are time-dependent parameters, and $\tau_\theta(x_t, t)$ is the predicted noise at timestep t. The process starts with a noisy image and iteratively refines it by sampling from these distributions until a clean sample is obtained.

3.1.2 Diffusion Transformer Architecture. The Diffusion Transformer (DiT) [32] employs a hierarchical structure $\mathcal{G} = g_1 \circ g_2 \circ \cdots \circ g_L$, where each module $g_l = \mathcal{F}^l_{\text{SA}} \circ \mathcal{F}^l_{\text{CA}} \circ \mathcal{F}^l_{\text{MLP}}$ consists of self-attention (SA), cross-attention (CA), and multilayer perceptron (MLP) components. In DiT, these components are dynamically adapted over time to accommodate varying noise levels during the image generation process. The input $\mathbf{x}t = \{x_i\}_{i=1}^{H\times W}$ is represented as a sequence of tokens that correspond to image patches. Each module integrates information through residual connections, defined as $\mathcal{F}(\mathbf{x}) = \mathbf{x} + \text{AdaLN} \circ f(\mathbf{x})$, where AdaLN refers to adaptive layer normalization, which facilitates more effective learning.

3.1.3 Speculative Decoding. Diffusion models incur high computational costs due to their sequential sampling process, requiring full model computation at each timestep. To address this, we introduce *SpeCa*, a framework that adapts the predict-validate paradigm to accelerate diffusion model inference.

$$P(y_1|x) \approx \prod_{i=1}^{n} P_{\text{draft}}(y_i|x, y_{<i}) \cdot \frac{P_{\text{main}}(y_i|x, y_{<i})}{P_{\text{draft}}(y_i|x, y_{<i})} \quad (2)$$

Key benefits include **efficient draft predictions**, **streamlined validation**, and **dynamic sequence length adjustment**. Efficiency is maximized when draft predictions align closely with the main model's distribution, enabling up to γ-fold acceleration, where γ is the number of units predicted per iteration.

3.2 SpeCa Framework Overview

Diffusion models incur high computational costs due to their sequential sampling, requiring full computation at each timestep. To mitigate this, we introduce *SpeCa*, a framework that adapts the predict-validate paradigm to accelerate inference.

SpeCa intelligently forecasts feature representations for multiple future timesteps and employs a rigorous validation mechanism, significantly reducing computational overhead while preserving generation quality. Unlike existing feature caching methods that often accumulate errors by ignoring temporal dynamics, our approach explicitly models feature evolution across timesteps. The core *SpeCa* workflow follows a "forecast-verify" paradigm:

(1) Execute a complete forward computation at a key time step t to obtain accurate feature representation $\mathcal{F}(x_t^l)$
(2) Based on this feature representation, predict the features for subsequent k time steps $\{t-1, t-2, \ldots, t-k\}$
(3) Evaluate the reliability of the predicted features through an error validation mechanism to decide whether to accept or reject each prediction step

This framework generates predicted features using a lightweight draft model and validates them using the main model, forming a dynamically adaptive inference path.

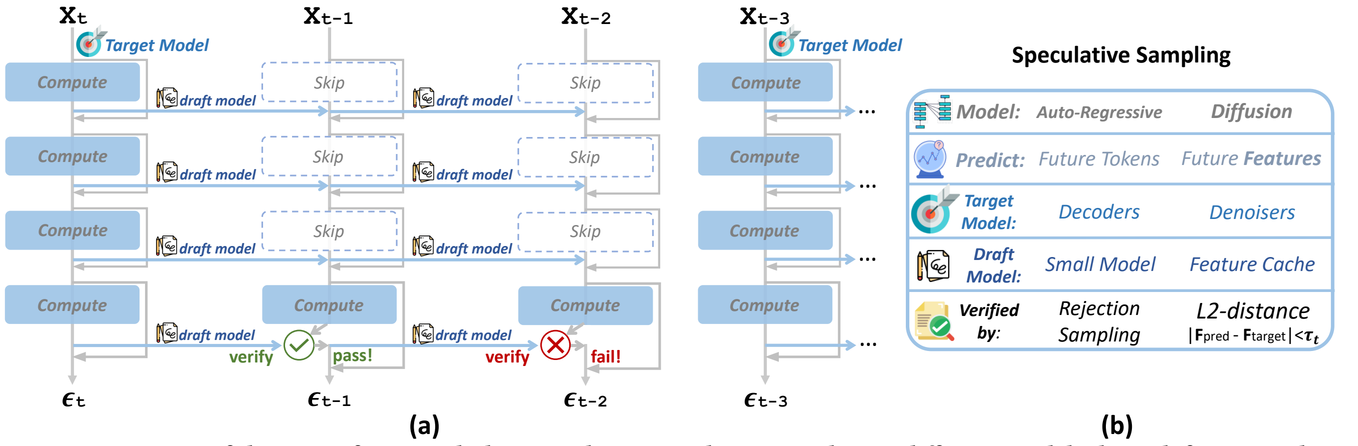
Xt
Xt-1
Xt-2
Xt-3
Target Model
Compute
Skip
draft model
verify
pass!
fail!
ϵt
ϵt-1
ϵt-2
ϵt-3
(a)
Speculative Sampling
Model: Auto-Regressive Diffusion
Predict: Future Tokens Future Features
Target Model: Decoders Denoisers
Draft Model: Small Model Feature Cache
Verified by: Rejection Sampling L2-distance |Fpred - Ftarget|<τt
(b)

Figure 3: Overview of the SpeCa framework that introduces speculative sampling to diffusion models through feature caching. (a) Our approach employs TaylorSeer as a lightweight draft model to forecast activation features across future timesteps ($t-1$ to $t-N$), with a verification mechanism that computes relative errors at the final network layer. When error exceeds threshold τ (at $t-2$), predictions are rejected and full computation resumes. (b) Comparison between speculative sampling in language models vs. diffusion models, highlighting our method's unique components: feature-based prediction, L2-distance verification, and adaptive computation allocation. This strategy significantly reduces inference costs while preserving generation quality.

3.3 Draft Model: TaylorSeer Predictor

SpeCa uses TaylorSeer as its draft model, which applies Taylor series expansion to predict features across time steps using current features and their derivatives. TaylorSeer requires no training while providing a rigorous and efficient prediction solution. For a feature $\mathcal{F}(x_t^l)$ at the current time step t, TaylorSeer effectively predicts the feature evolution for subsequent k time steps by leveraging temporal patterns in the diffusion process. This mathematical approach captures the underlying dynamics of feature transformation as:

$$\mathcal{F}_{\text{pred}}(x_{t-k}^l) = \mathcal{F}(x_t^l) + \sum_{i=1}^{m} \frac{\Delta^i \mathcal{F}(x_t^l)}{i! \cdot N^i}(-k)^i \tag{3}$$

where $\Delta^i \mathcal{F}(x_t^l)$ represents the i-th order finite difference of the feature, capturing essential patterns across sampling points. This term is computed through the expression:

$$\Delta^i \mathcal{F}(x_t^l) = \sum_{j=0}^{i} (-1)^{i-j} \binom{i}{j} \mathcal{F}(x_{t+jN}^l) \tag{4}$$

This formulation approximates higher-order derivatives with N as the sampling interval and k as the time step difference, enabling accurate and efficient feature prediction.

3.4 Error Computation and Validation

3.4.1 Relative Error Measure. We quantify prediction quality using relative error e_k, which measures deviation between predicted and actual features:

$$e_k = \frac{|\mathcal{F}_{\text{pred}}(x_{t-k}^l) - \mathcal{F}(x_{t-k}^l)|_2}{|\mathcal{F}(x_{t-k}^l)|_2 + \varepsilon} \tag{5}$$

where $\varepsilon = 10^{-8}$ prevents division by zero. This metric better reflects generation quality impact than absolute errors.

3.4.2 Sequential Validation Mechanism. Our validation follows a sequential decision procedure for predictions $\{t-1, t-2, ..., t-K\}$. At each step, we compare error e_k with threshold τ_t:

(1) If $e_k \le \tau_t$, the prediction for that time step is accepted, and validation proceeds to the next step;
(2) If $e_k > \tau_t$, the current and all subsequent predictions are rejected, reverting to standard computation.

Formally, for the prediction sequence $\{t-1, t-2, ..., t-K\}$, the process produces an acceptance set $\mathcal{A}$ and a rejection set $\mathcal{R}$:

$$\mathcal{A} = \{t-k \mid e_k \le \tau_t, 1 \le k \le j\} \tag{6}$$
$$\mathcal{R} = \{t-k \mid k > j\} \cup \{t-j\} \text{ if } e_j > \tau_t \tag{7}$$

To accommodate varying difficulty across diffusion stages, we apply an adaptive threshold: $\tau_t = \tau_0 \cdot \beta^{\frac{T-t}{T}}$, where τ_0 is initial threshold, $\beta \in (0,1)$ controls decay, and T is total steps. This allows more speculative execution in early noisy stages while enforcing stricter checks as fine details emerge.

3.5 Computational Complexity Analysis

Diffusion models traditionally require T sequential sampling steps, each with a complete forward computation, resulting in a total complexity of $O(T \cdot C)$, where C represents the per-step computational cost. *SpeCa* significantly reduces this computational burden through a strategic combination of speculative sampling and efficient validation mechanisms. In the *SpeCa* framework, the T sampling steps are dynamically divided into two categories:

(1) **Complete computation steps**: Execute the complete forward computation of the diffusion model
(2) **Speculative prediction steps**: Use TaylorSeer to predict features and perform lightweight validation

Let T_{full} and T_{spec} denote the number of complete computation steps and speculative prediction steps, respectively, with $T = T_{\text{full}} + T_{\text{spec}}$. The computational cost of *SpeCa* can be formally expressed as: $O(T_{\text{full}} \cdot C + T_{\text{spec}} \cdot C_{\text{pred}} + T_{\text{spec}} \cdot C_{\text{verify}})$. C represents the cost of a full computation step, while C_{pred} and C_{verify} represent the computational complexity of prediction and validation operations.

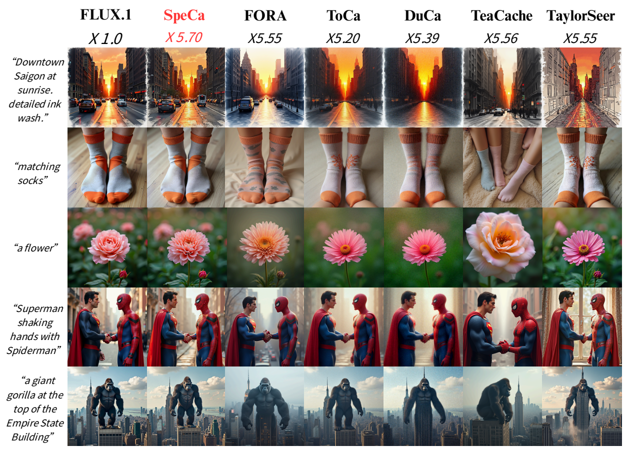
FLUX.1
SpeCa
FORA
ToCa
DuCa
TeaCache
TaylorSeer
X 1.0
X 5.70
X5.55
X5.20
X5.39
X5.56
X5.55
"Downtown Saigon at sunrise, detailed ink wash."
"matching socks"
"a flower"
"Superman shaking hands with Spiderman"
"a giant gorilla at the top of the Empire State Building"

Figure 4: Text-to-image generation comparison across acceleration methods. SpeCa (5.70×) achieves visual fidelity comparable to the baseline FLUX.1-dev, while surpassing other methods in structure, detail, and semantic accuracy across prompts. SpeCa better preserves object relationships, color consistency, and structure in complex scenes.

Let the prediction acceptance rate be $\alpha = \frac{T_{\text{spec}}}{T}$, and we express $T_{\text{full}} = T \cdot (1 - \alpha)$. The TaylorSeer predictor incurs negligible overhead ($C_{\text{pred}} \ll C$) and requires no additional training. In modern diffusion architectures, validation only involves comparing features in the final layer, costing $C_{\text{verify}} \approx \gamma \cdot C$, where $\gamma \ll 1$ (typically $\gamma < 0.05$). Thus, the total complexity simplifies to:

$$O(T \cdot (1-\alpha) \cdot C + T \cdot \alpha \cdot \gamma \cdot C) = O(T \cdot C \cdot (1 - \alpha + \alpha \cdot \gamma)) \tag{8}$$

Consequently, the theoretical acceleration ratio of SpeCa over traditional diffusion inference is:

$$\mathcal{S} = \frac{T \cdot C}{T \cdot C \cdot (1 - \alpha + \alpha \cdot \gamma)} = \frac{1}{1 - \alpha + \alpha \cdot \gamma} \tag{9}$$

Since $\gamma \ll 1$, when the prediction acceptance rate α is relatively high, the speedup can be approximated as: $\mathcal{S} \approx \frac{1}{1-\alpha}$. This efficiency arises from *SpeCa*'s ability to reduce full computations through accurate predictions, while using lightweight validation to maintain generation quality. Unlike methods that require retraining, *SpeCa* can be directly applied to existing diffusion models, providing significant computational savings in practical use.

4 Experiments

4.1 Experiment Settings

4.1.1 Model Configurations. For text-to-image generation, we employ **FLUX.1-dev**[17], which utilizes Rectified Flow sampling with 50 steps. This model is evaluated on NVIDIA H800 GPUs. For text-to-video generation, we use **HunyuanVideo**[22, 46], evaluated on the Hunyuan-Large architecture with 50-step inference, using NVIDIA H800 80GB GPUs. Lastly, for class-conditional image generation, we utilize **DiT-XL/2** [32], which employs 50-step DDIM sampling and is evaluated on NVIDIA A800 80GB GPUs. *For detailed model configurations, please refer to the Supplementary Material.*

4.1.2 Evaluation and Metrics. For the FLUX.1-dev, operating at a resolution of 1000 × 1000, we used 200 high-quality prompts from DrawBench [38] to generate diverse, photorealistic images. The overall quality was comprehensively assessed using ImageReward [48], and their alignment with the text was evaluated using the GenEval [10] benchmark. For the text-to-video generation task, we employed the VBench [14] framework with 946 prompts, generating 4,730 videos, which were evaluated across 16 core dimensions. For class-conditional image generation, we sampled 1,000 categories from ImageNet [37], producing a total of 50,000 images, which were evaluated using FID-50k [11], sFID, and Inception Score.

4.2 Text-to-Image Generation

Quantitative Study. SpeCa demonstrates significant superiority over other methods, especially at high acceleration rates. At a 6.34× speedup, *SpeCa* achieves an Image Reward of **0.9355**, far surpassing other methods such as TeaCache (0.7346) and TaylorSeer (0.8168). Additionally, *SpeCa* achieves a Geneval Overall score of **0.5922**, also outperforming other high-acceleration methods. These results highlight *SpeCa*'s ability to maintain generation quality while achieving high acceleration, with its advantages becoming more pronounced at higher acceleration rates. In contrast, many traditional methods such as TeaCache and TaylorSeer experience significant degradation in image quality when aiming for high acceleration, leading to a substantial decrease in Image Reward (e.g., TeaCache at 0.6356, TaylorSeer at 0.5886). This performance difference underscores *SpeCa*'s robustness and efficiency in high-speed inference, demonstrating its ability to minimize quality loss while enabling substantial acceleration, making it a clear leader in efficient text-to-image generation.

Qualitative Study. SpeCa maintains visual quality comparable to the baseline model FLUX.1-dev even at an acceleration ratio of **5.70**×, significantly outperforming other high-acceleration methods. Other models at similar acceleration ratios typically exhibit reduced structural integrity (as shape distortion in "*matching socks*"), decreased detail precision (like texture blurring in "*downtown Saigon*"), and object boundary ambiguity (like extra limbs appearing in "*Superman shaking hands with Spiderman*"). In contrast, *SpeCa* demonstrates remarkable advantages in preserving object morphology, textural details, semantic accuracy, and object boundary integrity, proving its excellence in text-to-image generation.

4.3 Text-to-Video Generation

Quantitative Study. SpeCa demonstrates exceptional computational efficiency while maintaining high quality. As shown in Table 2, our base configuration achieves **5692.7 TFLOPs** with **5.23**× acceleration and **79.98**% VBench, outperforming TaylorSeer ($\mathcal{N}$=5, $\mathcal{O}$=1) at 5.00× acceleration and 79.93% quality. *SpeCa* outperforms step reduction (78.74%), TeaCache (79.36%), ToCa (78.86%), and DuCa (78.72%), despite similar acceleration ratios. In the enhanced configuration, computational cost is further reduced to 4834.8 TFLOPs with 6.16× speedup, while maintaining 79.84% quality, showcasing the effectiveness of speculative sampling and sample-adaptive computation in balancing efficiency and output quality.

Qualitative Study. In the qualitative experiments of *SpeCa*, the model successfully overcomes three common generation errors and demonstrates significant advantages in accelerating inference. First, it addresses the Clock Face Scale and Text Spelling Error, accurately generating clock face symbols and avoiding issues like symbol distortion or text errors, ensuring visual clarity and semantic accuracy.

Table 1: Quantitative comparison in text-to-image generation for FLUX on Image Reward.

Method FLUX.1[17]	Efficient Attention [8]	Acceleration Latency(s) ↓	Speed ↑	FLOPs(T) ↓	Speed ↑	Image Reward ↑ DrawBench	Geneval↑ Overall
[dev]: 50 steps	✔	17.84	1.00×	3719.50	1.00×	0.9898 (+0.0000)	0.6752 (+0.0000)
60% steps	✔	10.84	1.65×	2231.70	1.67×	0.9739 (-0.0159)	0.6546 (-0.0206)
50% steps	✔	9.03	1.98×	1859.75	2.00×	0.9429 (-0.0469)	0.6655 (-0.0097)
40% steps	✔	7.15	2.50×	1487.80	2.62×	0.9317 (-0.0581)	0.6606 (-0.0146)
34% steps	✔	6.12	2.92×	1264.63	3.13×	0.9346 (-0.0552)	0.6501 (-0.0251)
Δ-DiT ($\mathcal{N} = 2$)[6]	✔	12.07	1.48×	2480.01	1.50×	0.9316 (-0.0582)	0.6700 (-0.0052)
Δ-DiT ($\mathcal{N} = 3$) † [6]	✔	9.21	1.94×	1686.76	2.21×	0.8561 (-0.1337)	0.6379 (-0.0373)
FORA ($\mathcal{N} = 6$) † [40]	✔	4.02	4.44×	893.54	4.16×	0.8235 (-0.1663)	0.5940 (-0.0812)
ToCa ($\mathcal{N} = 8$, $\mathcal{N} = 90\%$) † [54]	✘	5.94	3.00×	784.54	4.74×	0.9086 (-0.0812)	0.6347 (-0.0405)
DuCa ($\mathcal{N} = 8$, $\mathcal{N} = 70\%$) [55]	✔	7.59	2.35×	838.61	4.44×	0.8905 (-0.0993)	0.6361 (-0.0391)
TeaCache ($l = 0.8$) † [23]	✔	4.98	3.58×	892.35	4.17×	0.8683 (-0.1215)	0.6356 (-0.0396)
TaylorSeer ($\mathcal{N} = 5, \mathcal{O} = 2$) [24]	✔	6.17	2.89×	893.54	4.16×	0.9992 (+0.0094)	0.6266 (-0.0486)
SpeCa	✔	4.93	3.62×	791.40	4.70×	**0.9998 (+0.0100)**	**0.6397 (+0.0355)**
FORA ($\mathcal{N} = 7$) † [40]	✔	3.98	4.48×	670.44	5.55×	0.7398 (-0.2500)	0.5678 (-0.1074)
ToCa ($\mathcal{N} = 10$, $\mathcal{N} = 90\%$) † [54]	✘	5.65	3.16×	714.66	5.20×	0.7131 (-0.2767)	0.6026 (-0.0726)
DuCa, ($\mathcal{N} = 9$, $\mathcal{N} = 90\%$) † [55]	✔	4.69	3.80×	690.26	5.39×	0.8601 (-0.1297)	0.6189 (-0.0563)
TeaCache($l = 1.2$) † [23]	✔	3.98	4.48×	669.27	5.56×	0.7351 (-0.2547)	0.5845 (-0.0907)
TaylorSeer ($\mathcal{N} = 7, \mathcal{O} = 2$) † [24]	✔	5.02	3.55×	670.44	5.55×	0.9331 (-0.0567)	0.5886 (-0.0866)
SpeCa	✔	4.14	4.31×	**652.14**	**5.70×**	**0.9717 (-0.0181)**	**0.6338 (-0.0414)**
FORA ($\mathcal{N} = 9$) † [40]	✔	3.36	5.31×	596.07	6.24×	0.5442 (-0.4456)	0.5199 (-0.1553)
ToCa ($\mathcal{N} = 12$, $\mathcal{N} = 90\%$) † [54]	✘	5.27	3.39×	644.70	5.77×	0.7131 (-0.2767)	0.5479 (-0.1273)
DuCa ($\mathcal{N} = 12$, $\mathcal{N} = 80\%$) † [55]	✔	4.18	4.27×	606.91	6.13×	0.6753 (-0.3145)	0.5561 (-0.1191)
TeaCache ($l = 1.4$) † [23]	✔	3.63	4.91×	594.90	6.25×	0.7346 (-0.2552)	0.5524 (-0.1228)
TaylorSeer ($\mathcal{N} = 9, \mathcal{O} = 2$) † [24]	✔	4.72	3.78×	596.07	6.24×	0.8168 (-0.1730)	0.5380 (-0.1372)
SpeCa	✔	3.81	4.68×	**586.93**	**6.34×**	**0.9355 (-0.0543)**	**0.5922 (-0.0830)**

- † Methods exhibit significant degradation in Image Reward, leading to severe deterioration in image quality.
- gray: Deviation from FLUX.1-dev 50stpes baseline (ImageReward=0.9898, Geneval Overall=0.6752). Blue: **SpeCa**'s superior performance with minimal quality loss

Table 2: Quantitative comparison in text-to-video generation for HunyuanVideo on VBench.

Method	Latency(s) ↓	Speed ↑	FLOPs(T) ↓	Speed ↑	VBench ↑
50-steps	145.00	1.00×	29773.0	1.00×	80.66
22% steps	31.87	4.55×	6550.1	4.55×	78.74
TeaCache^{1}	30.49	4.76×	6550.1	4.55×	79.36
FORA	34.39	4.22×	5960.4	5.00×	78.83
ToCa	38.52	3.76×	7006.2	4.25×	78.86
DuCa	31.69	4.58×	6483.2	4.62×	78.72
TeaCache^{2}	26.61	5.45×	5359.1	5.56×	78.32
TaylorSeer^{1}	34.84	4.16×	5960.4	5.00×	79.93
SpeCa^{1}	34.58	4.19×	**5692.7**	**5.23×**	**79.98**
TaylorSeer^{2}	31.69	4.58×	5359.1	5.56×	79.78
SpeCa^{2}	31.45	4.61×	**4834.8**	**6.16×**	**79.84**

Second, for the Vase Pattern and Color Mismatch, *SpeCa* preserves intricate patterns and color matching, unlike other methods that often suffer from detail loss or inaccurate colors in complex patterns. Finally, when tackling the Bicycle Structural Deformation and Breakdown issue under high acceleration, *SpeCa* maintains structural integrity and high-quality content generation, while other methods tend to lose motion blur and exhibit structural deformation. Through its innovative speculative sampling mechanism, *SpeCa* ensures high-efficiency inference without compromising generation stability or content accuracy.

4.4 Class-Conditional Image Generation

We evaluated *SpeCa* against established acceleration methods including ToCa[54], FORA[40], DuCa[55], state-of-the-art TaylorSeer, and baseline DDIM with reduced sampling steps on DiT-XL/2[32] architecture. Results demonstrate that *SpeCa* consistently outperforms existing methods, particularly at high acceleration ratios. At **5**× acceleration, conventional methods struggle: DDIM-10 (FID 12.15), FORA (9.24), ToCa (12.86), and DuCa (12.05) show substantial degradation, while SpeCa achieves an exceptional FID of only **2.72**, outperforming TaylorSeer (3.09) by 12% with 5× faster inference. The disparity becomes more pronounced at **6.76** × acceleration, where traditional approaches severely degrade: DDIM-8 (FID 23.13), ToCa (21.24), and DuCa (31.97). Even TaylorSeer shows significant quality loss (FID 4.42). Under these extreme conditions, *SpeCa* maintains remarkable robustness with an FID of only **3.76** while preserving advantages across sFID and Inception Score. These results empirically validate our method's superiority, effectively mitigating error accumulation and addressing fundamental limitations in diffusion model acceleration.

4.5 Ablation Studies

Hyperparameter Analysis. Figure 8 demonstrates the impact of threshold (τ_0) and decay rate (β) on *SpeCa* performance. Our analysis reveals that increasing τ_0 substantially reduces computational demands (measured in FLOPs) at the expense of generation quality, with FID and sFID metrics showing marked degradation beyond $\tau_0 > 0.5$. Meanwhile, higher decay rates (β) yield marginal improvements in acceleration with negligible impact on output fidelity. Our adaptive threshold scheduling enables more aggressive time step skipping in the early denoising stages and more conservative progression in later stages, effectively balancing efficiency and fidelity

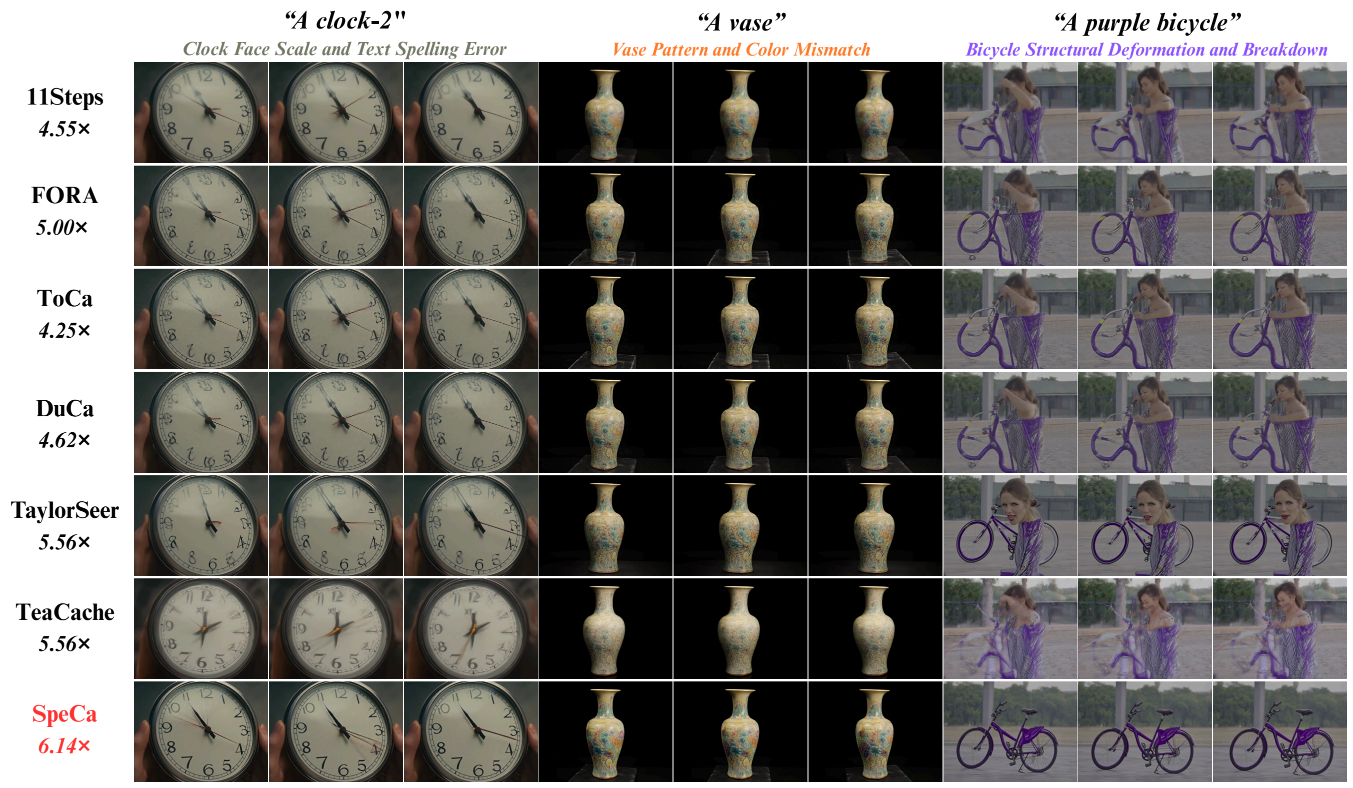
"A clock-2"
Clock Face Scale and Text Spelling Error
"A vase"
Vase Pattern and Color Mismatch
"A purple bicycle"
Bicycle Structural Deformation and Breakdown
11Steps
4.55×
FORA
5.00×
ToCa
4.25×
DuCa
4.62×
TaylorSeer
5.56×
TeaCache
5.56×
SpeCa
6.14×

Figure 5: Visualization of different generation methods on various objects. While achieving better accuracy in object creation, other methods suffer from issues such as *spelling errors*, *missing details*, and *structural distortions*. In contrast, our method excels in maintaining high-quality, precise generation without these errors.

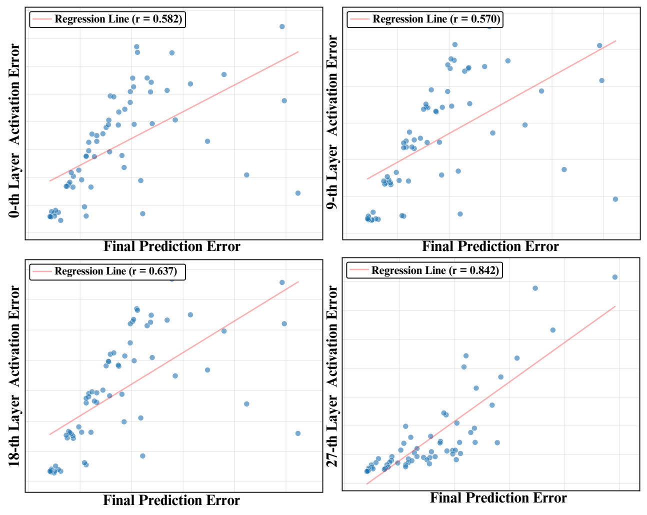

Figure 6: Strong correlation (r=0.842) between 27-th Layer errors and final output errors, validating its use as an efficient monitoring point. Results align with our Taylor-based analysis of error propagation in DiT's deeper layers.

throughout the generation process. Tables 4 and 5 provide comprehensive quantification of these relationships, illustrating how these parameters enable optimal trade-offs between computational efficiency and generation quality.

Validation Layer Selection and Trajectory Analysis. Our analysis of layer-wise feature correlations reveals that deeper layers provide more reliable indicators of final output quality. Figure 6 presents correlation analysis between activation errors measured at different DiT network layers and final generated image errors. Each scatter plot represents a generated sample, with the X-axis depicting the error between the final predicted output and the original full-step diffusion model output, while the Y-axis represents the activation error at specific network layers. Results demonstrate that 27-th Layer (deep layer) activation errors exhibit the **strongest correlation (*r*=0.842)** with final output errors, substantially higher than shallow layers (0-th Layer: *r*=0.582, 9-th Layer: *r*=0.570) and middle layers (18-th Layer: *r*=0.637). This finding provides robust support for our validation strategy: we can efficiently predict final generation quality by monitoring deep layer feature errors without computing the entire network. This aligns precisely with our theoretical analysis of error propagation based on Taylor expansion, confirming that deeper features have a more direct and deterministic influence on final output quality. Additionally, trajectory analysis in feature space confirms that *SpeCa* maintains evolution paths closely aligned with standard sampling, even at high acceleration ratios (5×). This trajectory preservation explains why our method maintains generation quality while significantly reducing computation. The consistent correlation patterns observed across diverse visual domains further validate our approach's generalizability and robustness in practical generation scenarios. *Refer to Appendices for detailed analyses.*

Table 3: Quantitative comparison on class-to-image generation on ImageNet with DiT-XL/2.

Method	**Efficient Attention**	**Latency(s)** ↓	**FLOPs(T)** ↓	**Speed** ↑	**FID** ↓	**sFID** ↓	**Inception Score** ↑
DDIM-50 steps	✔	0.995	23.74	1.00×	2.32(+0.00)	4.32(+0.00)	241.25(+0.00)
DDIM-25 steps	✔	0.537	11.87	2.00×	3.18 (+0.86)	4.74 (+0.42)	232.01 (-9.24)
DDIM-20 steps†	✔	0.406	9.49	2.50×	3.81 (+1.49)	5.15 (+0.83)	221.43 (-19.82)
DDIM-12 steps †	✔	0.128	5.70	4.17×	7.80 (+5.48)	8.03 (+3.71)	184.50 (-56.75)
DDIM-10 steps †	✔	0.224	4.75	5.00×	12.15 (+9.83)	11.33 (+7.01)	159.13 (-82.12)
DDIM-8 steps†	✔	0.189	3.80	6.25×	23.13 (+20.81)	19.23 (+14.91)	120.58 (-120.67)
DDIM-7 steps†	✔	0.168	3.32	7.14×	33.65 (+31.33)	27.51 (+23.19)	92.74 (-148.51)
Δ-DiT($\mathcal{N}=2$)	✔	0.740	18.04	1.31×	2.69 (+0.37)	4.67 (+0.35)	225.99 (-15.26)
Δ-DiT($\mathcal{N}=3$) †	✔	0.658	16.14	1.47×	3.75 (+1.43)	5.70 (+1.38)	207.57 (-33.68)
FORA ($\mathcal{N}=6$) †	✔	0.427	5.24	4.98×	9.24 (+6.92)	14.84 (+10.52)	171.33 (-69.92)
ToCa ($\mathcal{N}=9$, $\mathcal{N}=95\%$) †	✘	1.016	6.34	3.75×	6.55 (+4.23)	7.10 (+2.78)	189.53 (-51.72)
DuCa ($\mathcal{N}=6$, $\mathcal{N}=95\%$) †	✔	0.817	5.81	4.08×	6.40 (+4.08)	6.71 (+2.39)	188.42 (-52.83)
TaylorSeer ($\mathcal{N}=6, \mathcal{O}=4$)	✔	0.603	4.76	4.98×	3.09 (+0.77)	6.50 (+2.18)	225.16 (-16.09)
SpeCa	✔	0.476	**4.76**	**4.99**×	**2.72** (+0.40)	**5.51** (+1.19)	**233.85** (-7.40)
FORA ($\mathcal{N}=7$) †	✔	0.405	3.82	6.22×	12.55 (+10.23)	18.63 (+14.31)	148.44 (-92.81)
ToCa ($\mathcal{N}=13$, $\mathcal{N}=95\%$) †	✘	1.051	4.03	5.90×	21.24 (+18.92)	19.93 (+15.61)	116.08 (-125.17)
DuCa ($\mathcal{N}=12$, $\mathcal{N}=95\%$)†	✔	0.698	3.94	6.02×	31.97 (+29.65)	27.26 (+22.94)	87.94 (-153.31)
TaylorSeer ($\mathcal{N}=8, \mathcal{O}=4$) †	✔	0.580	3.82	6.22×	4.42 (+2.10)	7.75 (+3.43)	205.16 (-36.09)
SpeCa	✔	0.436	**3.51**	**6.76**×	**3.76** (+1.44)	**7.13** (+2.81)	**217.10** (-24.15)
FORA ($\mathcal{N}=8$, $\mathcal{N}=95\%$) †	✔	0.405	3.34	7.10×	15.31 (+12.99)	21.91 (+17.59)	136.21 (-105.04)
ToCa ($\mathcal{N}=13$, $\mathcal{N}=98\%$) †	✘	1.033	3.66	6.48×	22.18 (+19.86)	20.68 (+16.36)	110.91 (-130.34)
DuCa ($\mathcal{N}=18$, $\mathcal{N}=95\%$)†	✔	0.714	3.59	6.61×	133.06 (+130.74)	98.13 (+93.81)	15.87 (-225.38)
TaylorSeer ($\mathcal{N}=9, \mathcal{O}=4$)†	✔	0.571	3.34	7.10×	5.55 (+3.23)	8.45 (+4.13)	191.19 (-50.06)
SpeCa	✔	0.431	**3.26**	**7.30**×	**3.78** (+1.46)	**6.36** (+2.04)	**217.61** (-23.64)

- † Methods exhibit significant degradation in FID, leading to severe deterioration in image quality.
- gray: Deviation from DDIM-50 baseline (FID=2.32, sFID=4.32, IS=241.25). Blue: **SpeCa**'s superior performance with minimal quality loss

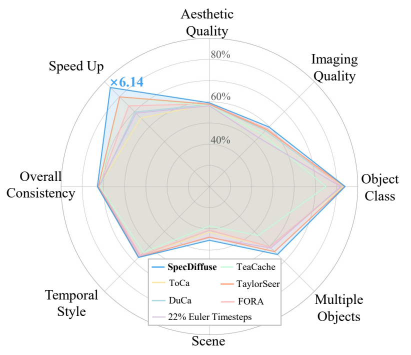

Figure 7: VBench performance of *SpeCa* versus baselines.

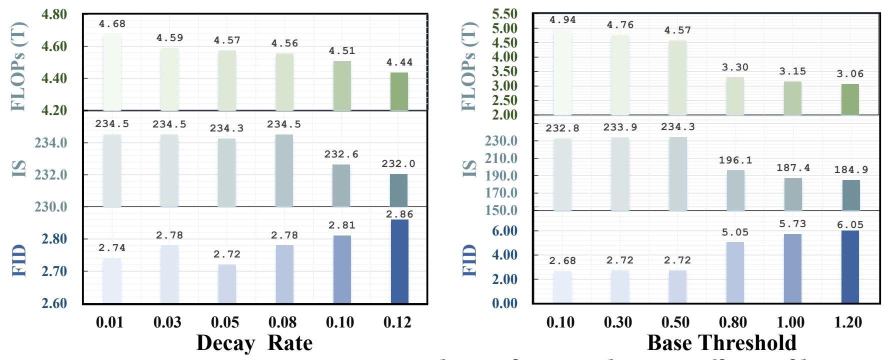

Figure 8: Hyperparameter sensitivity analysis of SpeCa showing effects of base ratio (τ_0) and decay rate (β) on computational efficiency and generation quality in SpeCa.

5 Conclusion

In this paper, we introduce *SpeCa*, a novel acceleration framework based on the "**forecast-then-verify**" mechanism for diffusion models. By extending speculative sampling principles to the diffusion domain, we successfully address the quality collapse issue that plagues traditional methods at high acceleration ratios. *SpeCa* employs a lightweight verification mechanism (consuming merely 1.67%-3.5% of full forward pass computation) to effectively evaluate prediction reliability, while introducing sample-adaptive computation allocation that dynamically adjusts computational resources based on generation complexity. Extensive experiments demonstrate significant performance improvements across various architectures: achieving 6.34× acceleration with only 5.5% quality degradation on FLUX models, maintaining high-fidelity generation at 7.3× acceleration on DiT models, and attaining 79.84% VBench score at 6.1× acceleration on the computationally intensive HunyuanVideo. As a plug-and-play solution, *SpeCa* seamlessly integrates into existing diffusion architectures without additional training, opening new avenues for diffusion model deployment in resource-constrained environments. Future work will explore applications to additional modalities and investigate synergies with other acceleration techniques, further enhancing the efficiency and applicability of diffusion models in practical scenarios.

References

[1] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. 2023. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127* (2023).

[2] Daniel Bolya and Judy Hoffman. 2023. Token merging for fast stable diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 4599–4603.

[3] Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. 2024. *Medusa: Simple LLM Inference Acceleration Framework with Multiple Decoding Heads*. doi:10.48550/arXiv.2401.10774 arXiv:2401.10774 [cs]

[4] Junsong Chen, Chongjian Ge, Enze Xie, Yue Wu, Lewei Yao, Xiaozhe Ren, Zhongdao Wang, Ping Luo, Huchuan Lu, and Zhenguo Li. 2024. PixArt-Σ: Weak-to-Strong Training of Diffusion Transformer for 4K Text-to-Image Generation. arXiv:2403.04692 [cs.CV]

[5] Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Yue Wu, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, and Zhenguo Li. 2024. PixArt-α: Fast Training of Diffusion Transformer for Photorealistic Text-to-Image Synthesis. In *International Conference on Learning Representations*.

[6] Pengtao Chen, Mingzhu Shen, Peng Ye, Jianjian Cao, Chongjun Tu, Christos-Savvas Bouganis, Yiren Zhao, and Tao Chen. 2024. Δ-DiT: A Training-Free Acceleration Method Tailored for Diffusion Transformers. *arXiv preprint arXiv:2406.01125* (2024).

[7] Zhuoming Chen, Avner May, Ruslan Svirschevski, Yuhsun Huang, Max Ryabinin, Zhihao Jia, and Beidi Chen. 2024. *Sequoia: Scalable, Robust, and Hardware-aware Speculative Decoding*. doi:10.48550/arXiv.2402.12374 arXiv:2402.12374 [cs]

[8] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher R'e. 2022. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. *ArXiv* abs/2205.14135 (2022). https://api.semanticscholar.org/CorpusID:249151871

[9] Gongfan Fang, Xinyin Ma, and Xinchao Wang. 2023. Structural Pruning for Diffusion Models. *arXiv preprint arXiv:2305.10924* (2023).

[10] Dhruba Ghosh, Hanna Hajishirzi, and Ludwig Schmidt. 2023. *GenEval: An Object-Focused Framework for Evaluating Text-to-Image Alignment*. doi:10.48550/arXiv.2310.11513 arXiv:2310.11513 [cs]

[11] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2018. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. doi:10.48550/arXiv.1706.08500 arXiv:1706.08500 [cs].

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. doi:10.48550/arXiv.2006.11239 arXiv:2006.11239 [cs].

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems* 33 (2020), 6840–6851.

[14] 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. 2023. VBench: Comprehensive Benchmark Suite for Video Generative Models. doi:10.48550/arXiv.2311.17982 arXiv:2311.17982 [cs].

[15] Minchul Kim, Shangqian Gao, Yen-Chang Hsu, Yilin Shen, and Hongxia Jin. 2024. Token fusion: Bridging the gap between token pruning and token merging. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 1383–1392.

[16] Sungbin Kim, Hyunwuk Lee, Wonho Cho, Mincheol Park, and Won Woo Ro. 2025. Ditto: Accelerating Diffusion Model via Temporal Value Similarity. In *Proceedings of the 2025 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE.

[17] Black Forest Labs. 2024. FLUX. https://github.com/black-forest-labs/flux.

[18] Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. *Fast Inference from Transformers via Speculative Decoding*. doi:10.48550/arXiv.2211.17192 arXiv:2211.17192 [cs]

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

[20] Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. 2023. Q-Diffusion: Quantizing Diffusion Models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 17489–17499. doi:10.1109/ICCV51070.2023.01608

[21] Yanyu Li, Huan Wang, Qing Jin, Ju Hu, Pavlo Chemerys, Yun Fu, Yanzhi Wang, Sergey Tulyakov, and Jian Ren. 2024. Snapfusion: Text-to-image diffusion model on mobile devices within two seconds. *Advances in Neural Information Processing Systems* 36 (2024).

[22] Zhimin Li, Jianwei Zhang, and and others Lin. 2024. Hunyuan-DiT: A Powerful Multi-Resolution Diffusion Transformer with Fine-Grained Chinese Understanding. doi:10.48550/arXiv.2405.08748 arXiv:2405.08748 [cs]

[23] Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. 2024. Timestep Embedding Tells: It's Time to Cache for Video Diffusion Model. arXiv:2411.19108 [cs.CV]

[24] Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. 2025. *From Reusing to Forecasting: Accelerating Diffusion Models with TaylorSeers*. doi:10.48550/arXiv.2503.06923 arXiv:2503.06923 [cs]

[25] Xingchao Liu, Chengyue Gong, et al. 2023. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. In *The Eleventh International Conference on Learning Representations*.

[26] Ziming Liu, Yifan Yang, Chengruidong Zhang, Yiqi Zhang, Lili Qiu, Yang You, and Yuqing Yang. 2025. Region-Adaptive Sampling for Diffusion Transformers. arXiv:2502.10389 [cs.CV] https://arxiv.org/abs/2502.10389

[27] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 2022. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems* 35 (2022), 5775–5787.

[28] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 2022. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095* (2022).

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

[30] Chenlin Meng, Ruiqi Gao, Diederik P Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. 2022. On Distillation of Guided Diffusion Models. In *NeurIPS 2022 Workshop on Score-Based Methods*. https://openreview.net/forum?id=6QHpSQt6VR-

[31] Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, Chunan Shi, Zhuoming Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. 2024. SpecInfer: Accelerating Generative Large Language Model Serving with Tree-based Speculative Inference and Verification. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3* (2024-04-27). 932–949. doi:10.1145/3620666.3651335 arXiv:2305.09781 [cs]

[32] William Peebles and Saining Xie. 2023. Scalable Diffusion Models with Transformers. doi:10.48550/arXiv.2212.09748 arXiv:2212.09748 [cs].

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

[34] Junxiang Qiu, Shuo Wang, Jinda Lu, Lin Liu, Houcheng Jiang, and Yanbin Hao. 2025. Accelerating Diffusion Transformer via Error-Optimized Cache. arXiv:2501.19243 [cs.CV] https://arxiv.org/abs/2501.19243

[35] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. doi:10.48550/arXiv.2112.10752 arXiv:2112.10752 [cs].

[36] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, 234–241.

[37] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. doi:10.48550/arXiv.1409.0575 arXiv:1409.0575 [cs].

[38] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. 2022. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. doi:10.48550/arXiv.2205.11487 arXiv:2205.11487 [cs]

[39] Tim Salimans and Jonathan Ho. 2022. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512* (2022).

[40] Pratheba Selvaraju, Tianyu Ding, Tianyi Chen, Ilya Zharkov, and Luming Liang. 2024. FORA: Fast-Forward Caching in Diffusion Transformer Acceleration. *arXiv preprint arXiv:2407.01425* (2024).

[41] Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. 2023. Post-training quantization on diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 1972–1981.

[42] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*. PMLR, 2256–2265.

[43] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*.

[44] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. 2023. Consistency Models. In *International Conference on Machine Learning*. PMLR, 32211–32252.

[45] Wenzhang Sun, Qirui Hou, Donglin Di, Jiahui Yang, Yongjia Ma, and Jianxun Cui. 2025. UniCP: A Unified Caching and Pruning Framework for Efficient Video Generation. arXiv:2502.04393 [cs.CV] https://arxiv.org/abs/2502.04393

[46] Xingwu Sun, Yanfeng Chen, Huang, et al. 2024. Hunyuan-Large: An Open-Source MoE Model with 52 Billion Activated Parameters by Tencent. doi:10.48550/arXiv.2411.02265 arXiv:2411.02265 [cs]

[47] Ziteng Sun, Ananda Theertha Suresh, Jae Hun Ro, Ahmad Beirami, Himanshu Jain, and Felix Yu. 2024. *SpecTr: Fast Speculative Decoding via Optimal Transport.* doi:10.48550/arXiv.2310.15141 arXiv:2310.15141 [cs]

[48] Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. 2023. ImageReward: Learning and Evaluating Human Preferences for Text-to-Image Generation. doi:10.48550/arXiv.2304.05977 arXiv:2304.05977 [cs].

[49] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Xiaotao Gu, Yuxuan.Zhang, Weihan Wang, Yean Cheng, Bin Xu, Yuxiao Dong, and Jie Tang. 2025. CogVideoX: Text-to-Video Diffusion Models with An Expert Transformer. In *The Thirteenth International Conference on Learning Representations.* https://openreview.net/forum?id=LQzN6TRFg9

[50] Zhihang Yuan, Hanling Zhang, Lu Pu, Xuefei Ning, Linfeng Zhang, Tianchen Zhao, Shengen Yan, Guohao Dai, and Yu Wang. 2024. DiTFastAttn: Attention Compression for Diffusion Transformer Models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems.* https://openreview.net/forum?id=51HQpkQy3t

[51] Kaiwen Zheng, Cheng Lu, Jianfei Chen, and Jun Zhu. 2023. DPM-Solver-v3: Improved Diffusion ODE Solver with Empirical Model Statistics. In *Thirty-seventh Conference on Neural Information Processing Systems.* https://openreview.net/forum?id=9fWKExmKa0

[52] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. 2024. *Open-Sora: Democratizing Efficient Video Production for All.* https://github.com/hpcaitech/Open-Sora

[53] Haowei Zhu, Dehua Tang, Ji Liu, Mingjie Lu, Jintu Zheng, Jinzhang Peng, Dong Li, Yu Wang, Fan Jiang, Lu Tian, Spandan Tiwari, Ashish Sirasao, Jun-Hai Yong, Bin Wang, and Emad Barsoum. 2024. DiP-GO: A Diffusion Pruner via Few-step Gradient Optimization. arXiv:2410.16942 [cs.CV]

[54] Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. 2024. Accelerating Diffusion Transformers with Token-wise Feature Caching. *arXiv preprint arXiv:2410.05317* (2024).

[55] Chang Zou, Evelyn Zhang, Runlin Guo, Haohang Xu, Conghui He, Xuming Hu, and Linfeng Zhang. 2024. Accelerating Diffusion Transformers with Dual Feature Caching. arXiv:2412.18911 [cs.LG] https://arxiv.org/abs/2412.18911

A Experimental Details

In this section, more details of the experiments are provided. All models incorporate a unified forced activation period $\mathcal{N}$, while $\mathcal{O}$ represents the order of the Taylor expansion.

A.1 Model Configurations.

The experiments are conducted on three state-of-the-art visual generative models: the text-to-image generation model FLUX.1-dev [17], text-to-video generation model HunyuanVideo [46], and the class-conditional image generation model DiT-XL/2 [32].
FLUX.1-dev[17] predominantly employs the Rectified Flow [25] sampling method with 50 steps as the standard configuration. All experimental evaluations of FLUX.1-dev were conducted on NVIDIA H800 GPUs.
HunyuanVideo [22, 46] was evaluated on the Hunyuan-Large architecture, utilizing the standard 50-step inference protocol as the baseline while preserving all default sampling parameters to ensure experimental rigor and consistency. Comprehensive performance benchmarks were systematically conducted on NVIDIA H800 80GB GPUs for detailed latency assessment and thorough inference performance analysis. The configuration parameters for FORA were set to $\mathcal{N} = 5$; ToCa and DuCa were configured with $\mathcal{N} = 5, R = 90\%$; TaylorSeer[1] was parameterized with $\mathcal{N} = 5, \mathcal{O} = 1$, while TaylorSeer[2] used $\mathcal{N} = 6, \mathcal{O} = 1$; TeaCache[1] was configured with a threshold of $l = 0.4$, and TeaCache[2] with $l = 0.5$.
DiT-XL/2 [32] adopts a 50-step DDIM [43] sampling strategy to ensure consistency with other models. Experiments on DiT-XL/2 are conducted on NVIDIA A800 80GB GPUs.

B Hyperparameter Analysis

Figure 8 illustrates how the base threshold (τ_0) and decay rate (β) affect *SpeCa* performance. Increasing the base threshold significantly reduces computational cost (FLOPs) while elevating FID and sFID values, with quality metrics notably deteriorating when $\tau_0 > 0.5$. Higher decay rates slightly improve acceleration ratios with minimal impacts on generation quality. Tables 4 and 5 further quantify these trends, demonstrating how adjusting these parameters enables *SpeCa* to achieve various trade-offs between computational efficiency and generation quality.

C Trajectory Analysis

Figure 9 presents feature evolution trajectories of different acceleration methods during the generation process. These trajectories are visualized in a two-dimensional plane after applying Principal Component Analysis (PCA) to high-dimensional feature representations at each timestep. Each trajectory represents a specific sample (such as "*daisy*", "*horse*", "*tiger*", or "*flamingo*") progressing from noise to clear image, with star and square markers indicating the trajectory's beginning and end points, respectively. The visualization clearly demonstrates that despite achieving a 4.99× acceleration, *SpeCa*'s feature evolution trajectory (red line) almost perfectly overlaps with the original DiT (blue line, 1.0× speed), while significantly outperforming other acceleration methods such as ToCa (4.92×) and TaylorSeer (4.99×). This result conclusively demonstrates our method's capability to accurately maintain the dynamic characteristics of the original diffusion process while substantially reducing computational steps, ensuring high-quality image generation.

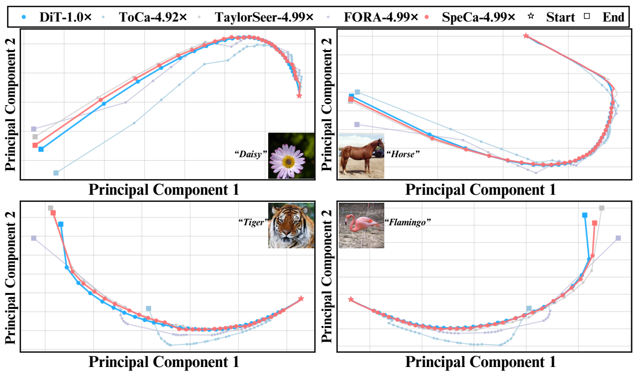

Figure 9: Scatter plot of the trajectories of different diffusion acceleration methods after performing Principal Component Analysis (PCA). The figure illustrates how the trajectories evolve across different methods, highlighting their relative efficiencies in terms of feature evolution.

Collectively, these hyperparameter analyses, deep layer validation strategies, and trajectory fidelity studies showcase *SpeCa*'s exceptional performance and flexibility in accelerating diffusion model inference, providing robust guarantees for efficient generation in practical applications.

Table 4: Ablation study on decay rate (base_threshold=0.5)

Decay Rate β	FLOPs(T) ↓	Speed ↑	FID ↓	sFID ↓	IS ↑
0.12	4.44	5.35	2.87	5.34	232.05
0.10	4.51	5.27	2.81	5.30	232.63
0.08	4.56	5.21	2.78	5.20	234.51
0.05	4.57	5.19	2.72	5.21	234.26
0.03	4.59	5.17	2.78	5.20	234.51
0.01	4.68	5.08	2.74	5.48	234.54

Table 5: Ablation study on base threshold (decay_rate=0.5)

Threshold τ_0	FLOPs(T) ↓	Speed ↑	FID ↓	sFID ↓	IS ↑
0.1	4.95	4.80	2.68	5.34	232.79
0.3	4.76	4.99	2.72	5.51	233.85
0.5	4.57	5.19	2.72	5.21	234.26
0.8	3.31	7.18	5.05	7.12	196.07
1.0	3.15	7.53	5.73	7.77	187.40
1.2	3.06	7.75	6.05	8.20	184.94

D Anonymous Page for Video Presentation

To further showcase the advantages of *SpeCa* in video generation, we have created an anonymous GitHub page. For a more detailed demonstration, please visit https://speca2025.github.io/SpeCa2025/. Additionally, the videos are also available in the Supplementary Material.

E Theoretical Analysis

E.1 Error Propagation Analysis

Theorem 1 (Prediction Error Bound). *For an m-th order Taylor expansion predictor with sampling interval N, the prediction error at step k satisfies:*

$$\|e_k\|_2 \leq \frac{L^{m+1}k^{m+1}}{(m+1)!N^{m+1}}\|\mathcal{F}(x_t^l)\|_2 + O\left(\frac{k^{m+2}}{(m+2)!N^{m+2}}\right) \tag{10}$$

where L is the Lipschitz constant of the feature evolution.

Proof. The Taylor expansion of $\mathcal{F}(x_{t-k}^l)$ around x_t^l gives:

$$\mathcal{F}(x_{t-k}^l) = \sum_{i=0}^{m} \frac{(-k)^i}{i!N^i}\Delta^i\mathcal{F}(x_t^l) + R_m(k) \tag{11}$$

$$\text{where } R_m(k) = \frac{(-k)^{m+1}}{(m+1)!N^{m+1}}\mathcal{F}^{(m+1)}(\xi),\ \xi \in [t-k, t] \tag{12}$$

For diffusion models, the feature evolution is governed by the underlying stochastic differential equation:

$$d\mathcal{F}(x_\tau^l) = \mu(\mathcal{F}(x_\tau^l), \tau)d\tau + \sigma(\tau)dW_\tau \tag{13}$$

where μ is the drift term, σ is the diffusion coefficient, and W_τ is the Wiener process.

Due to the smoothness properties of the diffusion process, we can bound the higher-order derivatives. Let $\mathcal{F}^{(i)}$ denote the i-th derivative of the feature with respect to time. For the drift and diffusion coefficients in diffusion models, we have:

$$\|\mu(\mathcal{F}(x_\tau^l), \tau)\|_2 \leq L\|\mathcal{F}(x_\tau^l)\|_2 \tag{14}$$

where L is the Lipschitz constant of the drift term.

Using the theory of finite differences, the approximation error of the finite difference operator is bounded by:

$$\|\Delta^i\mathcal{F}(x_t^l) - N^i\mathcal{F}^{(i)}(x_t^l)\|_2 \leq CN^{i+1}\|\mathcal{F}^{(i+1)}(x_t^l)\|_2 \tag{15}$$

for some constant C.

Furthermore, under the Lipschitz continuity assumption, we can establish:

$$\|\Delta^i\mathcal{F}(x_t^l)\|_2 \leq (2L)^i\|\mathcal{F}(x_t^l)\|_2 \tag{16}$$

This bound follows from the recursive application of the Lipschitz property across successive timesteps. For the i-th derivative, we have:

$$\|\mathcal{F}^{(i)}(x_t^l)\|_2 \leq L^i\|\mathcal{F}(x_t^l)\|_2 \tag{17}$$

Applying these bounds to the remainder term:

$$\|R_m(k)\|_2 \leq \frac{k^{m+1}}{(m+1)!N^{m+1}}L^{m+1}\|\mathcal{F}(x_t^l)\|_2 \tag{18}$$

The prediction error is thus:

$$\begin{aligned}\|\mathcal{F}_{\text{pred}}(x_{t-k}^l) - \mathcal{F}(x_{t-k}^l)\|_2 &= \|R_m(k)\|_2 \\ &\leq \frac{L^{m+1}k^{m+1}}{(m+1)!N^{m+1}}\|\mathcal{F}(x_t^l)\|_2 \\ &\quad + O\left(\frac{k^{m+2}}{(m+2)!N^{m+2}}\right)\end{aligned} \tag{19}$$

The relative error is defined as:

$$e_k = \frac{\|\mathcal{F}_{\text{pred}}(x_{t-k}^l) - \mathcal{F}(x_{t-k}^l)\|_2}{\|\mathcal{F}(x_{t-k}^l)\|_2 + \tau} \tag{20}$$

$$\tag{21}$$

To relate $\|\mathcal{F}(x_{t-k}^l)\|_2$ to $\|\mathcal{F}(x_t^l)\|_2$, we use the reverse Lipschitz condition:

$$\|\mathcal{F}(x_{t-k}^l)\|_2 \geq (1 - \frac{Lk}{N})\|\mathcal{F}(x_t^l)\|_2 \tag{22}$$

which follows from the fact that the feature evolution in diffusion models exhibits controlled decay properties.

Substituting this lower bound:

$$\begin{aligned}e_k &\leq \frac{L^{m+1}k^{m+1}}{(m+1)!N^{m+1}}\frac{\|\mathcal{F}(x_t^l)\|_2}{(1-\frac{Lk}{N})\|\mathcal{F}(x_t^l)\|_2} + O(k^{m+2}) && (23)\\ &= \frac{L^{m+1}k^{m+1}}{(m+1)!N^{m+1}}\frac{1}{1-\frac{Lk}{N}} + O(k^{m+2}) && (24)\end{aligned}$$

For small values of $\frac{Lk}{N}$ (which is typically the case in our setting), we can approximate $\frac{1}{1-\frac{Lk}{N}} \approx 1 + \frac{Lk}{N}$, yielding:

$$\begin{aligned}e_k &\leq \frac{L^{m+1}k^{m+1}}{(m+1)!N^{m+1}}\left(1 + \frac{Lk}{N}\right) + O(k^{m+2}) && (25)\\ &= \frac{L^{m+1}k^{m+1}}{(m+1)!N^{m+1}} + \frac{L^{m+2}k^{m+2}}{(m+1)!N^{m+2}} + O(k^{m+2}) && (26)\\ &= \frac{L^{m+1}k^{m+1}}{(m+1)!N^{m+1}} + O\left(\frac{k^{m+2}}{N^{m+2}}\right) && (27)\end{aligned}$$

This completes the proof of the error bound. □

E.2 Convergence Guarantee

Theorem 2 (Distributional Convergence). *When the verification threshold schedule satisfies:*

$$\tau_t = \tau_0 \cdot \beta^{\frac{T-t}{T}} \leq \sqrt{\frac{\beta_t(1-\bar{\alpha}_t)}{1+\|\mathcal{F}(x_t^l)\|_2^2}} \tag{28}$$

the generated distribution p_{Spec} converges to p_{orig} in total variation distance:

$$\lim_{T\to\infty} \|p_{Spec} - p_{orig}\|_{TV} = 0 \tag{29}$$

Proof. We decompose the proof into three main components: first establishing the relationship between feature errors and output image errors, then analyzing the error propagation through the diffusion process, and finally proving convergence in the distribution space.

1. Noise Schedule Compatibility:

In diffusion models, the denoising process at timestep t is given by:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sigma_t z \tag{30}$$

where $z \sim \mathcal{N}(0, I)$ is the random noise added during sampling, and $\sigma_t = \sqrt{\beta_t}$.

Let x^*_{t-1} be the output generated using the original model and x_{t-1} be the output from our *SpeCa* framework. The error between these outputs can be expressed as:

$$\begin{aligned}\|x_{t-1} - x^*_{t-1}\|_2 &= \left\| \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \tau_\theta(x_t, t) \right) \right. \\ &\quad \left. - \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \tau^*_\theta(x_t, t) \right) \right\|_2 \\ &= \frac{1}{\sqrt{\alpha_t}} \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \|\tau^*_\theta(x_t, t) - \tau_\theta(x_t, t)\|_2 \end{aligned} \tag{31}$$

The predicted noise $\tau_\theta(x_t, t)$ is a function of the feature representation $\mathcal{F}(x^l_t)$. Under the Lipschitz continuity of the mapping from features to noise predictions, we have:

$$\|\tau^*_\theta(x_t, t) - \tau_\theta(x_t, t)\|_2 \leq K \|\mathcal{F}^*(x^l_t) - \mathcal{F}(x^l_t)\|_2 \tag{32}$$

where K is the Lipschitz constant of the mapping.

Substituting our definition of relative error e_t:

$$\|\mathcal{F}^*(x^l_t) - \mathcal{F}(x^l_t)\|_2 = e_t \cdot (\|\mathcal{F}(x^l_t)\|_2 + \tau) \tag{33}$$

Therefore:

$$\|x_{t-1} - x^*_{t-1}\|_2 \leq \frac{1}{\sqrt{\alpha_t}} \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} K \cdot e_t \cdot (\|\mathcal{F}(x^l_t)\|_2 + \tau) \tag{34}$$

$$= C_t \cdot e_t \cdot (\|\mathcal{F}(x^l_t)\|_2 + \tau) \tag{35}$$

where $C_t = \frac{1}{\sqrt{\alpha_t}} \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} K$.

Our verification condition ensures:

$$e_t \leq \tau_t = \tau_0 \cdot \beta^{\frac{T-t}{T}} \leq \sqrt{\frac{\beta_t (1-\bar{\alpha}_t)}{1 + \|\mathcal{F}(x^l_t)\|_2^2}} \tag{36}$$

This implies:

$$\|x_{t-1} - x^*_{t-1}\|_2 \leq C_t \cdot \sqrt{\frac{\beta_t (1-\bar{\alpha}_t)}{1 + \|\mathcal{F}(x^l_t)\|_2^2}} \cdot (\|\mathcal{F}(x^l_t)\|_2 + \tau) \tag{37}$$

$$\approx C_t \cdot \sqrt{\beta_t (1-\bar{\alpha}_t)} \cdot \frac{\|\mathcal{F}(x^l_t)\|_2}{\sqrt{1 + \|\mathcal{F}(x^l_t)\|_2^2}} \tag{38}$$

$$\leq C_t \cdot \sqrt{\beta_t (1-\bar{\alpha}_t)} \tag{39}$$

This bounds the per-step error in terms of the diffusion noise schedule.

2. Martingale Analysis:

We now analyze the error accumulation across multiple steps. Define the error process:

$$M_t = \|x_t - x^*_t\|_2^2 - \sum_{s=1}^{t} \mathbb{E}[\|\tau_s\|_2^2] \tag{40}$$

where $\tau_s = x_s - \mathbb{E}[x_s | x_{s+1}]$ represents the error introduced at step s.

We can show that $\{M_t\}$ forms a martingale with respect to the filtration $\mathcal{F}_t$ (the information available up to time t):

$$\mathbb{E}[M_{t+1} | \mathcal{F}_t] = \mathbb{E}[\|x_{t+1} - x^*_{t+1}\|_2^2 | \mathcal{F}_t] - \sum_{s=1}^{t+1} \mathbb{E}[\|\tau_s\|_2^2] \tag{41}$$

$$= \|x_t - x^*_t\|_2^2 + \mathbb{E}[\|\tau_{t+1}\|_2^2 | \mathcal{F}_t] \tag{42}$$

$$- \sum_{s=1}^{t} \mathbb{E}[\|\tau_s\|_2^2] - \mathbb{E}[\|\tau_{t+1}\|_2^2] \tag{43}$$

$$= M_t + \mathbb{E}[\|\tau_{t+1}\|_2^2 | \mathcal{F}_t] - \mathbb{E}[\|\tau_{t+1}\|_2^2] \tag{44}$$

$$\tag{45}$$

Under our verification threshold condition, the error τ_{t+1} is bounded such that:

$$\mathbb{E}[\|\tau_{t+1}\|_2^2 | \mathcal{F}_t] = \mathbb{E}[\|\tau_{t+1}\|_2^2] \tag{46}$$

Therefore:

$$\mathbb{E}[M_{t+1} | \mathcal{F}_t] = M_t \tag{47}$$

This confirms that $\{M_t\}$ is indeed a martingale.

For the martingale to converge, we need to verify the Lindeberg condition:

$$\sum_{t=1}^{T} \frac{\mathbb{E}[\|\tau_t\|_2^4]}{\beta_t^2} < \infty \tag{48}$$

Under our threshold schedule, we have:

$$\mathbb{E}[\|\tau_t\|_2^4] \leq C_t^4 \cdot (\beta_t (1-\bar{\alpha}_t))^2 \tag{49}$$

$$= \frac{K^4 (1-\alpha_t)^4}{\alpha_t^2 (1-\bar{\alpha}_t)^2} \cdot (\beta_t (1-\bar{\alpha}_t))^2 \tag{50}$$

$$= K^4 \frac{(1-\alpha_t)^4 \beta_t^2}{\alpha_t^2} \tag{51}$$

Therefore:

$$\sum_{t=1}^{T} \frac{\mathbb{E}[\|\tau_t\|_2^4]}{\beta_t^2} = \sum_{t=1}^{T} K^4 \frac{(1-\alpha_t)^4}{\alpha_t^2} \tag{52}$$

$$\tag{53}$$

In diffusion models, the schedule typically satisfies $\alpha_t \to 1$ as $t \to T$, ensuring the above sum is finite.

By the martingale convergence theorem, M_t converges almost surely, implying:

$$\lim_{t \to T} \left(\|x_t - x^*_t\|_2^2 - \sum_{s=1}^{t} \mathbb{E}[\|\tau_s\|_2^2] \right) = M_\infty < \infty \text{ a.s.} \tag{54}$$

3. Convergence in Distribution:

Finally, we need to translate the almost sure convergence of the error process to convergence in distribution. For diffusion models, the total variation distance between the generated distributions can be bounded by:

$$\|p_{\text{Spec}} - p_{\text{orig}}\|_{\text{TV}} \leq \mathbb{E}[\|x_0 - x^*_0\|_2] \leq \sqrt{\mathbb{E}[\|x_0 - x^*_0\|_2^2]} \tag{55}$$

From our martingale analysis:

$$\mathbb{E}[\|x_0 - x^*_0\|_2^2] = \mathbb{E}[M_0] + \sum_{s=1}^{T} \mathbb{E}[\|\tau_s\|_2^2] = \mathbb{E}[M_T] + \sum_{s=1}^{T} \mathbb{E}[\|\tau_s\|_2^2] \tag{56}$$

Under our threshold schedule, we have:

$$\sum_{s=1}^{T} \mathbb{E}[\|\tau_s\|_2^2] \leq \sum_{s=1}^{T} C_s^2 \cdot \beta_s(1-\bar{\alpha}_s) \longrightarrow 0 \text{ as } T \longrightarrow \infty \tag{57}$$

Therefore:

$$\lim_{T\to\infty} \|p_{\text{Spec}} - p_{\text{orig}}\|_{\text{TV}} = 0 \tag{58}$$

This completes the proof of distributional convergence. □

E.3 Computational Complexity

Theorem 3 (Speedup Lower Bound). *With acceptance rate α and verification cost ratio γ, the speedup ratio $\mathcal{S}$ satisfies:*

$$\mathcal{S} \geq \frac{1}{1-\alpha(1-\gamma-\frac{C_{overhead}}{C})} \tag{59}$$

where $C_{overhead}$ is the constant-time overhead.

Proof. To establish a comprehensive understanding of the computational complexity, we analyze the component-wise breakdown of both the traditional diffusion sampling and our *SpeCa* approach.

1. Traditional Diffusion Sampling: The standard diffusion model inference requires T sequential denoising steps, each requiring a full forward pass through the model. Let C denote the computational cost of a single forward pass. The total computational complexity is:

$$T_{\text{standard}} = T \cdot C \tag{60}$$

2. *SpeCa* Computational Components:

Let us break down the computation in *SpeCa*:

(a) Full computation steps: These are the steps where we perform a complete forward pass through the diffusion model. If we denote the number of such steps as T_{full}, the computational cost is:

$$T_{\text{full}} \cdot C \tag{61}$$

(b) Prediction steps: For the speculative steps, we use TaylorSeer to predict features for subsequent timesteps. The computational cost of prediction is negligible compared to a full forward pass, but we include it for completeness:

$$T_{\text{spec}} \cdot C_{\text{pred}} \tag{62}$$

where $C_{\text{pred}} \ll C$.

(c) Verification steps: For each speculative prediction, we need to validate its accuracy using our lightweight verification mechanism. This involves computing the relative error between the predicted features and the actual features. The cost per verification is:

$$C_{\text{verify}} = \gamma \cdot C + C_{\text{overhead}} \tag{63}$$

where γ is the ratio of the verification cost to the full forward pass cost, and C_{overhead} is a constant overhead independent of the model size.

In modern diffusion architectures (DiT, FLUX, HunyuanVideo), the verification only requires comparing features in the final layer, resulting in $\gamma \ll 1$. Our empirical measurements show $\gamma \approx 0.035$ for DiT, $\gamma \approx 0.0175$ for FLUX, and $\gamma \approx 0.0167$ for HunyuanVideo.

3. Total Computation for *SpeCa*:

The total computational cost for *SpeCa* is:

$$\begin{aligned} T_{\text{total}} &= T_{\text{full}} \cdot C + T_{\text{spec}} \cdot C_{\text{pred}} + T_{\text{spec}} \cdot C_{\text{verify}} &(64)\\ &\approx T_{\text{full}} \cdot C + T_{\text{spec}} \cdot (\gamma \cdot C + C_{\text{overhead}}) &(65)\end{aligned}$$

where we have neglected C_{pred} as it is significantly smaller than other terms.

4. Acceptance Rate Analysis:

Define the acceptance rate α as the ratio of speculative steps to total steps:

$$\alpha = \frac{T_{\text{spec}}}{T} \tag{66}$$

This means $T_{\text{full}} = T - T_{\text{spec}} = T(1-\alpha)$. Substituting into our total computation formula:

$$\begin{aligned} T_{\text{total}} &= T(1-\alpha)\cdot C + T\alpha \cdot (\gamma \cdot C + C_{\text{overhead}}) &(67)\\ &= TC(1-\alpha) + T\alpha\gamma C + T\alpha C_{\text{overhead}} &(68)\\ &= TC[1-\alpha+\alpha\gamma] + T\alpha C_{\text{overhead}} &(69)\\ &= TC[1-\alpha(1-\gamma)] + T\alpha C_{\text{overhead}} &(70)\end{aligned}$$

5. Speedup Ratio:

The speedup ratio $\mathcal{S}$ is defined as the ratio of the standard computation time to the *SpeCa* computation time:

$$\begin{aligned} \mathcal{S} &= \frac{T_{\text{standard}}}{T_{\text{total}}} &(71)\\ &= \frac{TC}{TC[1-\alpha(1-\gamma)] + T\alpha C_{\text{overhead}}} &(72)\\ &= \frac{1}{1-\alpha(1-\gamma) + \frac{T\alpha C_{\text{overhead}}}{TC}} &(73)\\ &= \frac{1}{1-\alpha(1-\gamma) + \alpha\frac{C_{\text{overhead}}}{C}} &(74)\\ &= \frac{1}{1-\alpha+\alpha\gamma + \alpha\frac{C_{\text{overhead}}}{C}} &(75)\\ &= \frac{1}{1-\alpha(1-\gamma-\frac{C_{\text{overhead}}}{C})} &(76)\end{aligned}$$

6. Practical Approximation:

For modern architectures where $C \gg C_{\text{overhead}}$ (the model computation dominates over any constant overhead), and with $\gamma \approx 0.05$, we can approximate:

$$\begin{aligned} \mathcal{S} &\approx \frac{1}{1-\alpha(1-\gamma)} &(77)\\ &\approx \frac{1}{1-\alpha(0.95)} &(78)\\ &\approx \frac{1}{1-0.95\alpha} &(79)\end{aligned}$$

When α approaches 1 (high acceptance rate), the speedup approaches $\frac{1}{0.05} = 20\times$. In practice, we observe acceptance rates of approximately $\alpha = 0.85$ across various models and datasets, resulting in speedups around $\frac{1}{1-0.95\cdot 0.85} \approx \frac{1}{0.1925} \approx 5.2\times$, which aligns with our experimental results.

This analysis demonstrates that *SpeCa*'s efficiency directly correlates with the acceptance rate of speculative predictions, with theoretical maximum speedups approaching $\frac{1}{\gamma}$ as $\alpha \longrightarrow 1$. □

E.3.1 Adaptive Threshold Selection. The adaptive threshold is a critical component in balancing the trade-off between acceleration and quality. We adopt a timestep-dependent threshold schedule:

$$\tau_t = \tau_0 \cdot \beta^{\frac{T-t}{T}} \tag{80}$$

where τ_0 is the initial threshold, $\beta \in (0, 1)$ is the decay factor, and T is the total number of timesteps.

This schedule is motivated by the following observations:

1. **Noise Level Correlation**: In early timesteps (large t), the diffusion process is dominated by random noise, making feature predictions more reliable and allowing for higher thresholds.

2. **Detail Sensitivity**: As t decreases, the diffusion process focuses on generating fine details, requiring stricter thresholds to maintain visual quality.

3. **Error Propagation**: Errors in early timesteps have greater impact on final quality due to error accumulation, necessitating an exponential decay in threshold value.