# OUSAC: OPTIMIZED GUIDANCE SCHEDULING WITH ADAPTIVE CACHING FOR DIT ACCELERATION

**Anonymous authors**
Paper under double-blind review

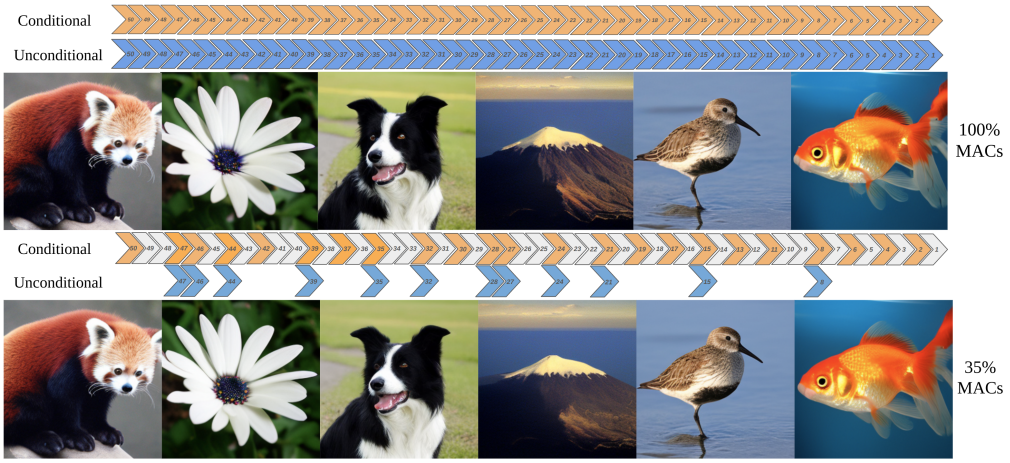


Figure 1: Top: Standard DDIM with 50 steps with CFG uniformly across all timesteps, requiring both conditional (orange) and unconditional (blue) forward pass at each step. Bottom: OUSAC achieves a similar visual quality with only **35%** of the original computational cost via two-stage optimization: optimized sparse guidance scheduling (empty white space) with caching (gray).

## ABSTRACT

Diffusion models have emerged as the dominant paradigm for high-quality image generation, yet their computational expense remains substantial due to iterative denoising. Classifier-Free Guidance (CFG) significantly enhances generation quality and controllability but doubles the computation size by requiring both conditional and unconditional forward passes at every timestep. We present OUSAC (**O**ptimized g**U**idance **S**cheduling with **A**daptive **C**aching), a framework that accelerates diffusion transformers (DiT) through systematic optimization. We begin with two key observations that reveal acceleration opportunities: first, the importance of guidance varies dramatically across timesteps – while a few critical steps require strong guidance, most steps need minimal or even no guidance; second, variable guidance patterns introduce denoising deviations that undermine the standard caching methods, which assume constant CFG scales and future similarity across steps. Moreover, different transformer blocks are affected with different levels under dynamic conditions. This paper develop a two stage approach leveraging these insights. Stage-1 employs evolutionary algorithms to discover sparse guidance schedules that apply CFG only at *critical* timesteps, which eliminates up to 82% of unconditional passes. Stage-2 introduces an adaptive rank allocation strategy that tailors calibration efforts per transformer block, maintaing caching effectiveness under variable guidance. Experiments demonstrate that OUSAC significantly outperforms the state-of-the-art acceleration methods. Specifically, it achieves 53% computational savings and a 15% improvement in generation quality on DiT-XL/2 (ImageNet 512×512), as well as 60% savings with 16.1% quality improvement on PixArt-$\alpha$ (MSCOCO) .

# 1 INTRODUCTION

Diffusion models (Ho et al., 2020; Song et al., 2021; Dhariwal & Nichol, 2021; Peebles & Xie, 2023; Chen et al., 2023) have revolutionized generative modeling, achieving unprecedented quality in image synthesis. Yet their widespread adoption remains limited by computational demands: generating a single high-quality image requires trillions of floating-point operations (TeraFLOPs) due to iterative denoising. This cost even doubles when using Classifier-Free Guidance (CFG) (Ho & Salimans, 2022), which improves generation quality by interpolating between conditional and unconditional predictions to strengthen adherence to input conditions. Though CFG is essential for balancing sample diversity and conditional fidelity, it applies a constant guidance scale uniformly across all timesteps – ignoring whether each step equally benefits from guidance. Current optimization methods often overlook this inefficiency, applying the same guidance scale uniformly across all denoising steps, from high-noise to low-noise regions. This raises a fundamental question: *can we retain the quality benefits of CFG while drastically reducing its computational cost?* The challenge is non-trivial – as naively skipping guidance at arbitrary timesteps leads to severe quality degradation. As illustrated in Figure 2, only through carefully optimized sparse guidance patterns can match the performance of full CFG while eliminating most computational overhead, indicating that many guidance computations in existing approaches are redundant.

We present OUSAC: Optimized gUidance Scheduling with Adaptive Caching for Diffusion Transformer Acceleration, a framework that addresses an unexplored challenge of integrating guidance scheduling with feature caching to accelerate DiT. We focus on transformers due to their growing adoption in state-of-the-art models (Peebles & Xie, 2023; Chen et al., 2023) and their uniform block structure that enables systematic optimization. While existing methods achieve efficiency gains through either guidance scheduling (Wang et al., 2024; Castillo et al., 2023) or feature caching (Ma et al., 2023; 2024; Zou et al., 2025) in isolation, no method has effectively combined the two.

This integration is challenging because variable guidance patterns violate the key assumption of caching methods – feature similarity across timesteps. To tackle this, OUSAC employs a two-stage optimization approach. In *Stage 1*, we discover sparse guidance schedules that identify which denoising steps truly require CFG and determine the optimal guidance scale at each. This poses a complex discrete-continuous optimization problem, as timestep decisions non-linearly interact throughout the denoising trajectory. Gradient-based optimization cannot apply directly to this due to memory constraints and vanishing gradients over $T$ steps. Instead, we use evolutionary strategies to efficiently explore the guidance space, discovering extremely space patterns that preserve quality while skipping guidance at most timesteps. In *Stage 2*, we introduce adaptive rank allocation for incremental calibration under variable guidance. The sparse schedules from *Stage 1* introduce two types of denosing deviations: guidance scale variations between consecutive steps, and branch switching when alternating betwen CFG and conditional-only passes. These break the feature consistency assumed by standard caching. We address this by assigning different calibration ranks to different transformer blocks, adapting to their sensitivity to guidance changes, an significant departure from the uniform calibration used in prior methods (Chen et al., 2025).

The synergy between optimized scheduling and adaptive caching enables gains beyond what each technique achieves alone. OUSAC reduces computational cost by 53% while improving FID by 15% on DiT-XL/2 (ImageNet 512×512), and achieves 60% cost reduction with a 16.1% FID improvement on PixArt-$\alpha$ (MSCOCO). Here are our key contributions:

- A systematic framework for optimizing per-step guidance schedules through evolutionary strategies, discovering that extremely sparse patterns (eliminating up to 82% of unconditional forward passes) match or exceed the quality of constant guidance.
- Adaptive rank allocation via coordinate descent that assigns calibration ranks to different transformer regions under the impact of variable guidance, achieving 15% better FID than uniform calibration approaches at equivalent computational budgets.

# 2 RELATED WORK

**Diffusion model acceleration.** Recent acceleration methods fall into three categories. Sampling acceleration reduces denoising steps through improved numerical solvers. DDIM (Song et al., 2022)

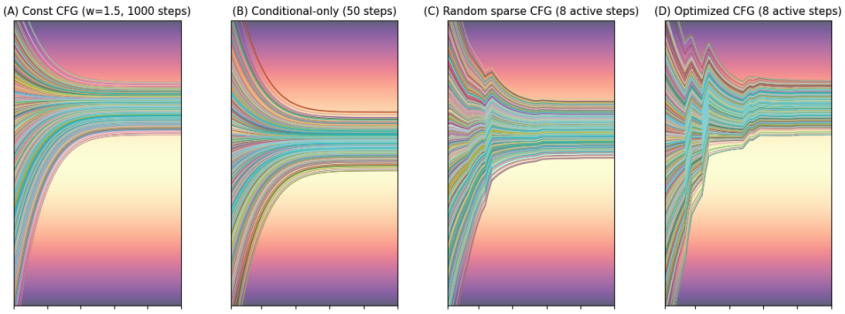


Figure 2: **1D example of OUSAC uses fewer steps to converge to the same solution as constant CFG.** (A) Constant CFG ($w$=1.5, 1000 steps) requires 2000 forward passes to map from a prior distribution (left) to a target distribution (right). (B) Conditional-only (50 steps) converges to an incorrect distribution. (C) Random sparse CFG with randomly assigned guidance scales at 8 steps also fails. (D) Our optimized sparse CFG with carefully tuned guidance scales at 8 steps matches the target distribution while using only 58 forward passes.

enables deterministic sampling with fewer steps, while DPM-Solver (Lu et al., 2022) uses exponential integrators for faster convergence. Higher-order methods (Zhang & Chen, 2023) and progressive distillation (Salimans & Ho, 2022) further reduce steps, though distillation requires expensive training. Recent inference-time distillation (Park et al., 2024) eliminates separate training but still cannot generalize across guidance scales. Architectural optimizations reduce per-step costs through efficient designs (Peebles & Xie, 2023), pruning (Zhu et al., 2024), and quantization (Liu et al., 2025; Yang et al., 2025). Our work maintain full denoising steps while reducing per-step cost through selective CFG forward passes and adaptive caching, without architectural modifications.

**Dynamic guidance scheduling.** Evidence shows constant CFG wastes computation. Kynkäänniemi et al. (2024) find guidance harmful at extreme noise levels, while Wang et al. (2024) shows monotonic schedules outperform constant guidance. Theoretical advances include progressive guidance (Xi et al., 2024), characteristic guidance with non-linear corrections (Zheng & Lan, 2024), and gradient artifact correction (Gao et al., 2025). Training-free methods achieve partial speedups through convergence detection (Castillo et al., 2023), early-stage compression (Dinh et al., 2024), and adaptive scaling (Malarz et al., 2025; Li et al., 2025). Zhang et al. (2025) and Yehezkel et al. (2025) showed optimal schedules vary across architectures. Alternative approaches include autoguidance (Karras et al., 2024) and condition annealing (Sadat et al., 2023). We are the first to use evolutionary optimization to discover hybrid discrete-continuous guidance schedules.

**Feature caching and calibration.** Caching exploits temporal redundancy between timesteps. DeepCache (Ma et al., 2023) pioneered feature reuse for U-Nets, extended to transformers through block-level caching (Wimbauer et al., 2024) and training-inference harmonization (Huang et al., 2025). Learning-to-Cache (Ma et al., 2024) uses learned routing but produces fixed patterns. Token-wise caching (Zou et al., 2025) achieves 2.36x speedup through selective token reuse. ICC (Chen et al., 2025) combines caching with uniform SVD calibration across all blocks. However, no existing work addresses how calibration should adapt when guidance varies.

## 3 Preliminaries

**Diffusion models and sampling.** Diffusion models learn to reverse a forward noising process defined as $q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I)$, where $x_t$ is the noised image at timestep $t \in \{1, ..., T\}$, $x_0$ is the clean image, and $\bar{\alpha}_t$ represents the cumulative noise schedule. The denoising process can be accelerated using DDIM (Song et al., 2022):

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{x}_0(x_t, t) + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2} \cdot \epsilon_\theta(x_t, t) + \sigma_t\epsilon_t, \tag{1}$$

where $\hat{x}_0$ is the predicted clean image from $x_t$, $\epsilon_\theta$ is the learned noise predictor network, $\sigma_t$ controls the stochasticity of sampling (with $\sigma_t = 0$ for deterministic generation), and $\epsilon_t \sim \mathcal{N}(0, I)$.

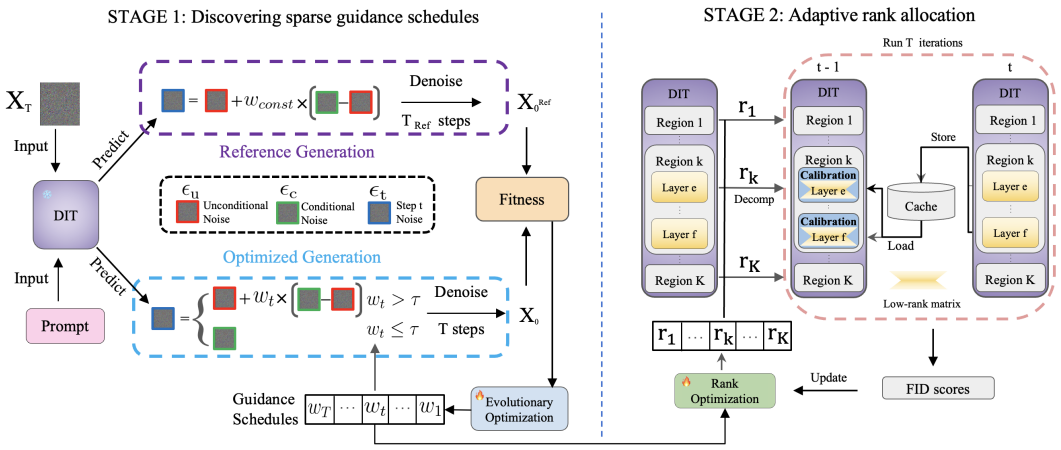


Figure 3: **The two-stage OUSAC optimization framework. Stage 1 (Left):** Evolutionary optimization discovers sparse guidance schedules by refining per-step guidance $\mathbf{w} = [w_T, \ldots, w_1]$. Starting from noise $x_T$, the framework generates a reference $x_0^{\text{Ref}}$ via $T_{\text{Ref}}$ denoising steps. At each timestep, full CFG is applied if $w_t > \tau$, otherwise only conditional forward passes are performed. The fitness function balances quality and sparsity, iteratively improving $\mathbf{w}$ through population sampling and evaluation. **Stage 2 (Right):** Adaptive rank allocation optimizes caching. DiT blocks are partitioned into $K$ regions, each assigned a calibration rank $r_k$. Cached features are corrected via SVD-based calibration. Coordinate descent with binary search tunes ranks to minimize FID.

**Classifier-free guidance.** To improve the quality of conditional generation, CFG (Ho & Salimans, 2022) interpolates between conditional and unconditional predictions:

$$\tilde{\epsilon}_\theta(x_t, c, t) = \epsilon_\theta(x_t, \emptyset, t) + w \cdot (\epsilon_\theta(x_t, c, t) - \epsilon_\theta(x_t, \emptyset, t)), \tag{2}$$

where $c$ denotes the conditioning information, $\emptyset$ represents null conditioning, and $w$ controls the guidance scale. By applying CFG at each timestep, the total computation is *doubled*.

**Caching for diffusion models.** Recent work (Chen et al., 2025) accelerates diffusion transformers by caching and reusing features across timesteps. The method corrects cached features through layer-wise calibration:

$$\hat{\mathbf{h}}_{\text{out}}^{\ell} = \mathcal{P}(\mathbf{h}_{\text{out}}^{\ell,\text{prev}}) + \mathbf{A}^{\ell}(\mathbf{h}_{\text{in}}^{\ell} - \mathcal{P}(\mathbf{h}_{\text{in}}^{\ell,\text{prev}})), \tag{3}$$

where $\ell$ is the layer index, $\mathcal{P}(\cdot)$ denotes the caching operation from the previous timestep, $\mathbf{h}_{\text{in}}^{\ell}$ is the current input to layer $\ell$, and $\mathcal{P}(\mathbf{h}_{\text{in}}^{\ell,\text{prev}})$ and $\mathcal{P}(\mathbf{h}_{\text{out}}^{\ell,\text{prev}})$ are the cached input and output from the previous timestep. Each layer has its own calibration matrix $\mathbf{A}^{\ell}$ that transforms the input increment to correct the cached output. To reduce computation, each $\mathbf{A}^{\ell}$ is approximated using SVD decomposition: $\mathbf{A}^{\ell} = \mathbf{U}^{\ell}\mathbf{\Sigma}^{\ell}\mathbf{V}^{\ell T} \approx \mathbf{U}_r^{\ell}\mathbf{\Sigma}_r^{\ell}\mathbf{V}_r^{\ell T}$, where the subscript $r$ denotes truncation to rank $r$. Prior increment-calibrated caching methods use uniform rank $r$ across all layers in all transformer blocks.

## 4 OUSAC

OUSAC accelerates diffusion transformers through two-stage optimization. *Stage 1* uses evolutionary algorithms to discover sparse guidance schedules that eliminate unconditional forward passes at non-critical timesteps where guidance contributes minimally to generation quality. *Stage 2* develops adaptive rank allocation for feature caching, where different transformer regions receive different calibration ranks to handle the varying feature differences introduced by variable guidance patterns. We optimize these components once per pre-trained model to discover optimal configurations. During inference, no optimization occurs—we simply apply the discovered sparse guidance schedule and adaptive caching configuration to accelerate generation. The discovered patterns generalize across different prompts and conditions. Sections 4.1 and 4.2 detail each optimization stage.

### 4.1 Stage 1: Discovering Sparse Guidance Schedules

#### 4.1.1 Problem Formulation

CFG's computational cost comes from applying guidance uniformly at every denoising timestep. Recent empirical studies (Kynkäänniemi et al., 2024) have provided valuable insights showing that guidance can be harmful at extreme noise levels and unnecessary near convergence. These pioneering works establish important foundations through interval-based strategies that significantly improve efficiency. In this work, we take it one step further and explore if we can find more complex, flexible, and task-specific patterns for CFG. We start by replacing the constant guidance scale $w$ with a per-timestep guidance schedule to reformulate Equation 2:

$$\tilde{\epsilon}_\theta(x_t, c, t) = \epsilon_\theta(x_t, \emptyset, t) + w_t \cdot (\epsilon_\theta(x_t, c, t) - \epsilon_\theta(x_t, \emptyset, t)), \tag{4}$$

where $w_t$ is now timestep-dependent. We optimize a guidance schedule $\mathbf{w} = [w_1, w_2, ..., w_T]$ where each $w_t \in [0, w_{\max}]$. When $w_t$ falls below a threshold $\tau$, we set $w_t = 0$ and skip the unconditional forward pass entirely, performing only the conditional forward pass.

The best schedule $\mathbf{w}$ can be found by solving the following optimization problem:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \mathcal{L}_{\text{total}}(\mathbf{w}) = \mathcal{L}_{\text{quality}}(\mathbf{w}) + \lambda \mathcal{L}_{\text{sparse}}(\mathbf{w}) \tag{5}$$

The quality preservation term $\mathcal{L}_{\text{quality}}$ ensures our sparse schedule maintains generation fidelity through output matching:

$$\mathcal{L}_{\text{quality}}(\mathbf{w}) = \mathbb{E}_{x_T, c}\left[\|\mathcal{G}_T(x_T, c; \mathbf{w}) - \mathcal{G}_{T_{\text{ref}}}(x_T, c; w_{\text{const}})\|_2^2\right] \tag{6}$$

where $\mathcal{G}_T$ denotes the $T$-step generation process starting from initial noise $x_T$ with our optimized schedule $\mathbf{w}$, and $\mathcal{G}_{T_{\text{ref}}}$ represents reference generation from the same $x_T$ using constant guidance $w_{\text{const}}$. The reference uses substantially more steps ($T_{\text{ref}} \gg T$, typically 1000 vs 20-50) to provide smooth denoising process and high-quality targets. Starting from identical noise ensures fair comparison and helps identify critical timesteps for guidance.

The sparsity term directly penalizes the number of timesteps requiring full CFG forward pass: $\mathcal{L}_{\text{sparse}}(\mathbf{w}) = \sum_{t=1}^{T} \mathbb{I}[w_t > \tau]$ where $\tau$ serves as an activation threshold below which guidance is completely disabled, eliminating the unconditional forward pass. This binary decision at each timestep transforms the optimization into a hybrid continuous-discrete problem (Barton et al., 2000).

#### 4.1.2 Evolutionary Optimization Strategy

Direct gradient-based optimization of this objective is intractable as it would require backpropagation through the entire $T$-step generation trajectory, creating prohibitive memory requirements and suffering from vanishing gradients. Instead, we employ a tailored evolutionary strategy that operates in a transformed space for numerical stability.

We maintain a population center $\boldsymbol{\mu} \in \mathbb{R}^T$ where $\boldsymbol{\mu} = [\mu_1, \ldots, \mu_T]^T$ with each $\mu_t \in \mathbb{R}$. This center represents the mean of our search distribution in the parameter space, a fundamental concept in both CMA-ES (Hansen, 2023) and Natural Evolution Strategies (Wierstra et al., 2014; Yi et al., 2009). At each generation $g \in \{1, \ldots, G\}$, we decode the center to get base guidance values:

$$\mathbf{w}_{\text{base}} = w_{\max} \cdot \text{sigmoid}(\boldsymbol{\mu}_{\boldsymbol{g}}) \tag{7}$$

We construct a population by perturbing these base values. For each candidate $i \in \{1, \ldots, P\}$: $\mathbf{w}^{(i)} = \mathbf{w}_{\text{base}} + \boldsymbol{\delta}^{(i)}$, where $\boldsymbol{\delta}^{(i)} \sim \mathcal{N}(0, \sigma_{\text{noise}}^2 I)$ and $\sigma_{\text{noise}} = \sigma_0(1 - g/G)$ decreases across generations to refine exploration, with $\sigma_0$ being the initial noise scale.

We apply a threshold $\tau$ to sparsify the guidance schedule and determine the noise prediction at $t$:

$$\epsilon_t = \begin{cases} \epsilon_u + w_t^{(i)} \cdot (\epsilon_c - \epsilon_u) & \text{if } w_t^{(i)} \geq \tau \\ \epsilon_c & \text{if } w_t^{(i)} < \tau \end{cases} \tag{8}$$

Each candidate is evaluated using: $f^{(i)} = -\mathcal{L}_{\text{quality}}(\mathbf{w}^{(i)}) + \lambda \cdot S(\mathbf{w}^{(i)})$, where $S(\mathbf{w}^{(i)}) = (T - \|\mathbf{w}^{(i)}\|_0)/T$ measures sparsity. With fitness for all candidates $\{f^{(1)}, \cdots, f^{(P)}\}$, we compute rank-based weights: $a_i = d_i/(P-1) - 0.5$, where $d_i \in \{0, 1, \cdots, P-1\}$ is the rank of candidate $i$,

with 0 for the lowest fitness and $P-1$ for the highest. This rank-based weighting scheme follows established practices in evolution strategies (Hansen & Ostermeier, 2001; Hansen et al., 2003).

The population center evolves through natural gradient estimation:

$$\boldsymbol{\mu_{g+1}} \leftarrow \boldsymbol{\mu_g} + \frac{\eta}{P}\sum_{i=1}^{P} a_i \cdot (\text{sigmoid}^{-1}(\mathbf{w}^{(i)}/w_{\max}) - \boldsymbol{\mu_g}) \tag{9}$$

After $G$ generations, the converged center $\boldsymbol{\mu}^*$ yields: $\mathbf{w}^* = w_{\max} \cdot \text{sigmoid}(\boldsymbol{\mu}^*)$. This sparse schedule $\mathbf{w}^*$ applies guidance selectively at critical timesteps to reduce redundant computations.

### 4.2 Stage 2: Adaptive Caching Under Denoising Fluctuations

#### 4.2.1 The Challenge of Caching with Variable Guidance

The sparse guidance schedules discovered in *Stage 1* fundamentally challenge existing caching methods. Standard caching approaches such as ICC (Chen et al., 2025) assume that features between consecutive timesteps remain similar. This assumption holds when guidance remains constant throughout denoising. However, our variable guidance patterns from *Stage 1* break this assumption, causing the incremental calibration in Equation 3 to become less effective when correcting the larger feature differences introduced by changing guidance scales.

As shown in Figure 5, when we apply variable guidance from *Stage 1* with naive caching, feature reconstruction errors increase significantly compared to constant CFG with the same caching method. The variable guidance causes higher MSE across all transformer blocks, with particularly severe degradation in deeper blocks. Since these elevated reconstruction errors accumulate through the network and degrade final image quality, which motivates us to explore the reason behind this.

To address this challenge, we first need to understand how variable guidance affects the denoising process. We consider two scenarios: when consecutive timesteps both use CFG but with different guidance scales, and when timesteps switch between using CFG and using only conditional prediction. Each scenario introduces distinct types of denoising fluctuations that degrade caching.

*Scenario 1: Both timesteps use CFG with different scales.* When both $w_t^* > \tau$ and $w_{t-1}^* > \tau$, the denoising process deviates by:

$$\Delta x_{t-1}^{\text{strength}} = \sqrt{\bar{\alpha}_{t-1}/\bar{\alpha}_t} \cdot (w_{t-1}^* - w_t^*) \cdot [\epsilon_c(x_t, t) - \epsilon_u(x_t, t)] \tag{10}$$

where $\bar{\alpha}_t$ is the cumulative noise schedule coefficient. This deviation grows with the guidance difference $(w_{t-1}^* - w_t^*)$ and the gap between conditional and unconditional predictions.

*Scenario 2: CFG at timestep $t$ transitions to no guidance at timestep $t-1$.* When $t$ uses CFG ($w_t^* > \tau$) but $t-1$ uses only conditional prediction ($w_{t-1}^* < \tau$), the deviation becomes:

$$\Delta x_{t-1}^{\text{switch}} = \beta_{t-1,t} \cdot (1 - w_t^*) \cdot [\epsilon_c(x_t, t) - \epsilon_u(x_t, t)], \tag{11}$$

where $\beta_{t-1,t} = \sqrt{1-\bar{\alpha}_{t-1}} - \sqrt{\bar{\alpha}_{t-1}(1-\bar{\alpha}_t)/\bar{\alpha}_t}$ is the noise coefficient for DDIM sampling. These deviations exceed what standard caching methods can handle, as cached features from timestep $t$ no longer match the expected input for timestep $t-1$. This mismatch causes higher reconstruction errors that accumulate through the transformer blocks. This heterogeneous error distribution reveals that different transformer regions require different calibration to effectively handle variable guidance.

#### 4.2.2 Region-Adaptive Rank Allocation

Caching with variable guidance causes larger denoising deviations than caching under a constant CFG. While incremental calibration was designed to alleviate caching errors under constant CFG, we now examine how it performs under variable guidance patterns. In our experiment, we find that different blocks benefit from different calibration ranks. Early (block 0) and late blocks (block 26) maintain lower MSE with higher ranks (=512), while middle blocks (block 12) achieve better performance with lower ranks (=256). This heterogeneous pattern shows that uniform rank allocation cannot handle the varying error magnitudes introduced by variable guidance. For detailed analysis across all transformer blocks, please refer to the appendix A.2. Since higher ranks directly increase computational cost during denoising, we face a trade-off between feature consistency and

efficiency. These observations motivate our systematic approach to discovering optimal rank distributions, where we assign each region the rank that best balances error reduction and compute.

Formally, we partition the $N$ transformer blocks into $K$ regions $\mathcal{R} = \{R_1, \ldots, R_K\}$ based on their network position. We divide blocks uniformly such that each region contains $\lfloor N/K \rfloor$ consecutive blocks, with region $R_k$ containing blocks from index $(k-1) \cdot \lfloor N/K \rfloor$ to $k \cdot \lfloor N/K \rfloor - 1$. Each region $k$ receives a tailored calibration rank $r_k$ to obtain a region-specific calibration matrix:

$$\mathbf{A}_\ell \approx \mathbf{U}_{\ell,r_k} \mathbf{\Sigma}_{\ell,r_k} \mathbf{V}_{\ell,r_k}^T \quad \text{for layer } \ell \in R_k \tag{12}$$

We optimize the rank configuration $\mathbf{r} = [r_1, r_2, \ldots, r_K]$ where each $r_k \in [r_{\min}, r_{\max}]$.

#### 4.2.3 Rank Optimization via Coordinate Descent.

Finding the optimal rank configuration $\mathbf{r}^* = [r_1, \ldots, r_K]$ requires searching over a large discrete space where each region can take ranks from $[r_{\min}, r_{\max}]$. This is challenging because rank assignments across regions interact through the sequential nature of the transformer: early blocks affect later blocks' inputs, creating complex dependencies that make the relationship between rank configuration and final generation quality non-linear. This is a constrained optimization problem:

$$\mathbf{r}^* = \arg\min_{\mathbf{r}} \text{FID}(\mathcal{G}_T(\mathbf{r}, \mathbf{w}^*)) \quad \text{s.t.} \quad r_k \in [r_{\min}, r_{\max}], \quad \textstyle\sum_{k=1}^{K} r_k \leq B, \tag{13}$$

where $\mathcal{G}_T(\mathbf{r}, \mathbf{w}^*)$ denotes the $T$-step generation process using rank configuration $\mathbf{r}$ with the optimized guidance schedule $\mathbf{w}^*$ from *Stage 1*, and $B$ represents the total computational budget. The generation process follows the same $T$-step denoising trajectory as in *Stage 1*, but now using regional calibration matrices based on the rank configuration $\mathbf{r}$.

We solve this optimization through coordinate descent (Wright, 2015), which naturally decomposes the problem into a sequence of single-variable optimizations. For each region $k$, we fix the ranks of all other regions and search for the optimal $r_k$ that minimizes the FID score:

$$r_k^* = \arg\min_{r_k} \text{FID}(\mathcal{G}(r_1, \ldots, r_k, \ldots, r_K, \mathbf{w}^*)) \tag{14}$$

Within each coordinate optimization step, we use binary search to efficiently explore the rank space $[r_{\min}, r_{\max}]$. This procedure iterates across all regions until the overall rank configuration converges.

## 5 Experiments

### 5.1 Experimental Setup

**Models and datasets.** We evaluate OUSAC on two state-of-the-art diffusion transformers. For class-conditional generation, we employ DiT-XL/2 (Peebles & Xie, 2023) on ImageNet (Deng et al., 2009), generating 50,000 images at both $256\times256$ and $512\times512$ resolutions across all 1,000 classes. For text-to-image synthesis, we utilize PixArt-$\alpha$ (Chen et al., 2023) on MSCOCO 2014 (Lin et al., 2015), producing 30,000 images at $256\times256$ resolution using the caption set from (Zou et al., 2025).

**Evaluation metrics.** For ImageNet generation, we adopt the standard evaluation suite: Inception Score (IS) (Salimans et al., 2016) for sample quality, Fréchet Inception Distance (FID) (Nash et al., 2021) and spatial FID (sFID) for distribution matching, and Precision-Recall (Kynkäänniemi et al., 2019) for mode coverage and fidelity. For MSCOCO text-to-image synthesis, we report FID-30k and CLIP Score (Hessel et al., 2022) to assess both visual quality and text-image alignment. Computational efficiency is quantified through multiply-accumulate operations (MACs), providing hardware-agnostic performance measurements. Latency is measured with batch size 8 on a single H100 GPU.

**Implementation details.** For DiT-XL/2, we use DDIM sampling with timesteps from 20 to 50 and classifier-free guidance with a baseline scale of 1.5. The evolutionary optimization discovers sparse schedules over 15 generations. For adaptive rank allocation, transformer blocks are partitioned into regions based on their position in the network, with coordinate descent exploring calibration ranks from 16 to 1024. Reference trajectories use longer sampling steps (250-1000) to ensure high-quality supervision. PixArt-$\alpha$ follows a similar protocol but employs DPM-Solver with 20 steps.

Table 1: Quantitative comparison of acceleration methods on DiT-XL/2 for ImageNet generation.

| Methods | Steps | CFG-Steps↓ | MACs (T)↓ | Latency (s)↓ | IS↑ | FID↓ | sFID↓ | Prec.↑ | Recall↑ |
|---|---|---|---|---|---|---|---|---|---|
| **DiT-XL/2 (ImageNet 512×512)** | | | | | | | | | |
| DDIM | 1000 | 1000 | 1049.1 | 416.79 | 210.6 | 2.99 | 4.38 | 83.17 | 55.6 |
| DDIM | 50 | 50 | 52.45 | 20.86 | 203.8 | 3.20 | 4.53 | 83.27 | 56.4 |
| L2C | 50 | 50 | 40.62 | 16.30 | 199.5 | 3.98 | 5.66 | 82.46 | 53.3 |
| ICC | 50 | 50 | 33.43 | 15.88 | 200.0 | 3.73 | 5.39 | 83.30 | 55.6 |
| OUSAC w/o cache | 50 | **9** | 30.94 | 12.91 | 228.3 | **2.71** | 4.38 | 83.43 | **57.0** |
| OUSAC | 50 | **9** | **24.93** | **11.28** | **228.6** | 2.72 | **4.13** | **83.62** | 55.3 |
| DDIM | 30 | 30 | 31.47 | 12.52 | 198.0 | 3.86 | 4.94 | **83.12** | 54.4 |
| L2C | 30 | 30 | 25.72 | 10.31 | 189.4 | 4.93 | 6.72 | 82.14 | 54.5 |
| ICC | 30 | 30 | 20.05 | 9.54 | 171.0 | 6.85 | 6.72 | 79.84 | 53.5 |
| OUSAC w/o cache | 30 | **8** | 19.93 | 8.20 | **214.8** | **3.33** | 4.85 | 82.99 | 55.3 |
| OUSAC | 30 | **8** | **16.01** | **7.19** | 209.7 | 3.37 | **4.66** | 82.62 | **56.5** |
| **DiT-XL/2 (ImageNet 256×256)** | | | | | | | | | |
| DDIM | 1000 | 1000 | 237.2 | 106.42 | 245.0 | 2.12 | 4.66 | 80.66 | 59.7 |
| DDIM | 50 | 50 | 11.86 | 5.33 | 239.4 | 2.23 | 4.29 | 80.06 | 59.2 |
| HarmoniCa | 50 | 50 | 10.58 | 4.78 | 210.1 | 3.33 | 5.03 | 77.40 | **60.1** |
| L2C | 50 | 50 | 9.76 | 4.43 | 245.5 | 2.23 | **4.27** | 80.95 | 59.1 |
| ICC | 50 | 50 | 8.05 | 4.18 | 258.5 | 2.16 | 4.28 | 82.08 | 58.1 |
| OUSAC w/o cache | 50 | **8** | 6.63 | 3.37 | 263.0 | 2.10 | 4.29 | 81.85 | 58.8 |
| OUSAC | 50 | **8** | **5.07** | **2.81** | **266.5** | **2.04** | 4.29 | **82.09** | 58.7 |

**Baselines.** We compare against representative acceleration techniques: (1) ICC (Chen et al., 2025), which applies uniform-rank increment-calibrated caching to transformer blocks; (2) Learning-to-Cache (L2C) (Ma et al., 2024), which uses learned routing for adaptive feature caching; (3) HarmoniCa (Huang et al., 2025), which harmonizes training and inference through optimized caching strategies; and (4) standard DDIM sampling at various step counts as reference points. These baselines represent both training-free methods (ICC) and approaches requiring additional training (L2C, HarmoniCa), enabling comprehensive evaluation of our gradient-free optimization approach. Since parts of L2C and HarmoniCa lack publicly available checkpoints for our experimental settings, we reimplemented them following their published protocols.

**Guidance-driven caching protocol.** We adapt the caching strategy to handle variable guidance patterns from *Stage 1*. The key modification is that when timestep $t$ uses only the conditional forward pass and timestep $t-1$ requires full CFG, we cannot reuse cached features since the unconditional forward pass was never computed at timestep $t$. Otherwise, standard caching applies. This ensures correct CFG application while maximizing cache reuse when guidance is inactive.

### 5.2 Main Results

**DiT-XL/2 on ImageNet.** Table 1 demonstrates substantial efficiency gains across both resolutions. At $512\times512$, OUSAC matches 50-step DDIM quality (FID 2.72 vs 3.20) with 47% less computation (24.97T vs 52.45T MACs) by applying guidance at only 9 of 50 timesteps. It even surpasses 1000-step DDIM (FID 2.72 vs 2.99) while using 97% less computation. At $256 \times 256$, OUSAC achieves the best FID (2.04) among all baselines, including 1000-step DDIM (2.12), while using only 5.07T vs 11.86T MACs for standard 50-step sampling.

**PixArt-$\alpha$ on MSCOCO.** Text-to-image generation with DPM-Solver shows a minimal quality gap between 20- and 1000-step sampling (FID 24.60 vs 22.97), providing weak learning signals for our evolutionary optimization. Despite this challenge, OUSAC discovers that only 6 out of 20 timesteps need guidance on average, achieving FID 19.27 (21.7% improvement) with 60% computational reduction (2.67 vs 6.72 MACs), while maintaining text-image alignment (CLIP score 16.48 vs 16.31).

**Comparison with other CFG redundancy reduction methods.** Table 3 compares OUSAC with Adaptive Guidance (Castillo et al., 2023), which uses discrete selection among $k+2$ predetermined

Table 2: Performance evaluation on MSCOCO 2014 with PixArt-$\alpha$ at $256 \times 256$ resolution.

| Method | Steps | CFG-Steps↓ | MACs (T)↓ | FID↓ | CLIP Score↑ |
|---|---|---|---|---|---|
| DPM-Solver | 1000 | 1000 | 336.11 | 22.97 | 16.42 |
| DPM-Solver | 20 | 20 | 6.72 | 24.60 | 16.31 |
| ICC | 20 | 20 | 3.70 | 21.86 | 16.47 |
| OUSAC w/o cache | 20 | **6** | 4.37 | 22.69 | 16.39 |
| OUSAC | 20 | **6** | **2.67** | **19.27** | **16.48** |

Table 3: Comparison of CFG redundancy reduction methods on DiT-XL/2 (ImageNet 512×512).

| Methods | Strategy | CFG Steps↓ | MACs (T)↓ | ΔMACs↓ | IS↑ | FID↓ | sFID↓ | Precision↑ | Recall↑ |
|---|---|---|---|---|---|---|---|---|---|
| DDIM | Constant | 50 | 52.45 | – | 203.8 | 3.25 | 4.53 | 83.27 | 56.4 |
| Adaptive Guidance | Discrete | 32 | 43.00 | -18% | 179.2 | 4.15 | 4.68 | 82.88 | 55.9 |
| OUSAC | Continuous | **9** | **30.94** | **-41%** | **228.3** | **2.71** | **4.38** | **83.43** | **57.0** |

options per timestep. While Adaptive Guidance achieves 18% computational reduction (32 CFG steps), OUSAC's continuous optimization ($w_t \in [0, w_{\max}]$) discovers only 9 timesteps need guidance, achieving 41% reduction and improving FID from 3.25 to 2.71.

### 5.3 Ablation Study

We systematically investigate the design choices in OUSAC through controlled experiments, analyzing how each component contributes to the overall performance gains.

Table 4: Ablation study of rank allocation strategies for DiT-XL/2 (ImageNet 512×512).

| Methods | Steps | MACs (T)↓ | IS↑ | FID↓ | sFID↓ | Precision↑ | Recall↑ |
|---|---|---|---|---|---|---|---|
| DDIM | 50 | 52.45 | 203.8 | 3.25 | 4.53 | 83.27 | 56.4 |
| ICC | 50 | 33.43 | 200.0 | 3.73 | 5.39 | 83.30 | 55.6 |
| OUSAC w/o cache | 50 | 30.94 | **228.3** | **2.71** | **4.38** | 83.43 | **57.0** |
| OUSAC w/ uniform $r$=1024 | 50 | 34.68 | **229.8** | 2.92 | 4.96 | 82.12 | 54.6 |
| OUSAC w/ uniform $r$=512 | 50 | 26.16 | 205.0 | 4.46 | 5.19 | 78.19 | 55.4 |
| OUSAC w/ uniform $r$=256 | 50 | 23.94 | 213.7 | 3.68 | 6.47 | 82.61 | 54.8 |
| **OUSAC** | 50 | **22.37** | 228.1 | 3.01 | 4.63 | **83.82** | 54.9 |

**Does adaptive rank allocation outperform uniform rank allocation?** Table 4 validates our adaptive rank allocation strategy. If we replace OUSAC's adaptive ranks with uniform ranks for all blocks, performance drops. Uniform $r$=256 increases FID to 3.68, while uniform $r$=1024 achieves FID 2.92 but requires 55% more computation (34.68T vs 22.37T MACs). OUSAC discovers region-specific rank distributions that achieve FID 3.01 with only 22.37T MACs, demonstrating that different transformer regions require different calibration levels under variable guidance patterns.

Additional ablation studies can be found in Appendix A.3.

## 6 Conclusion

This paper establishes that guidance is only important at a small subset of denoising steps, contrary to the traditional practice of uniform Classifier-Free-Guidance (CFG).We propose OUSAC with this critical insight, which integrates sparse guidance scheduling with adaptive feature caching. Using evolutionary search, we find that only 18% of timesteps in DiT-XL/2 require guidance, yet this selective use yields better FID than standard CFG. To address feature reconstruction errors from variable guidance, we introduce adaptive rank allocation via coordinate descent with binary search. Together, sparse guidance and adaptive caching enable OUSAC to achieve both improved FID and reduced computation compared to baselines.

## REFERENCES


PI Barton, Julio R Banga, and S Galan. Optimization of hybrid discrete/continuous dynamic systems. *Computers & Chemical Engineering*, 24(9-10):2171–2182, 2000.

Angela Castillo, Jonas Kohler, Juan C. Pérez, Juan Pablo Pérez, Albert Pumarola, Bernard Ghanem, Pablo Arbeláez, and Ali Thabet. Adaptive guidance: Training-free acceleration of conditional diffusion models, 2023. URL https://arxiv.org/abs/2312.12487.

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

Zhiyuan Chen, Keyi Li, Yifan Jia, Le Ye, and Yufei Ma. Accelerating diffusion transformer via increment-calibrated caching with channel-aware singular value decomposition, 2025. URL https://arxiv.org/abs/2505.05829.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021. URL https://arxiv.org/abs/2105.05233.

Anh-Dung Dinh, Daochang Liu, and Chang Xu. Compress guidance in conditional diffusion sampling, 2024. URL https://arxiv.org/abs/2408.11194.

Zhengqi Gao, Kaiwen Zha, Tianyuan Zhang, Zihui Xue, and Duane S. Boning. Reg: Rectified gradient guidance for conditional diffusion models, 2025. URL https://arxiv.org/abs/2501.18865.

Nikolaus Hansen. The cma evolution strategy: A tutorial, 2023. URL https://arxiv.org/abs/1604.00772.

Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, 9(2):159–195, 2001.

Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary computation*, 11(1):1–18, 2003.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning, 2022. URL https://arxiv.org/abs/2104.08718.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. URL https://arxiv.org/abs/2207.12598.

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

Yushi Huang, Zining Wang, Ruihao Gong, Jing Liu, Xinjie Zhang, Jinyang Guo, Xianglong Liu, and Jun Zhang. Harmonica: Harmonizing training and inference for better feature caching in diffusion transformer acceleration, 2025. URL https://arxiv.org/abs/2410.01723.

Tero Karras, Miika Aittala, Tuomas Kynkäänniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself, 2024. URL https://arxiv.org/abs/2406.02507.

Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models, 2019. URL https://arxiv.org/abs/1904.06991.

Tuomas Kynkäänniemi, Miika Aittala, Tero Karras, Samuli Laine, Timo Aila, and Jaakko Lehtinen. Applying guidance in a limited interval improves sample and distribution quality in diffusion models, 2024. URL https://arxiv.org/abs/2404.07724.

Pengxiang Li, Shilin Yan, Joey Tsai, Renrui Zhang, Ruichuan An, Ziyu Guo, and Xiaowei Gao. Adaptive classifier-free guidance via dynamic low-confidence masking, 2025. URL https://arxiv.org/abs/2505.20199.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015. URL https://arxiv.org/abs/1405.0312.

Xuewen Liu, Zhikai Li, and Qingyi Gu. Cachequant: Comprehensively accelerated diffusion models, 2025. URL https://arxiv.org/abs/2503.01323.

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, 2022. URL https://arxiv.org/abs/2206.00927.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free, 2023. URL https://arxiv.org/abs/2312.00858.

Xinyin Ma, Gongfan Fang, Michael Bi Mi, and Xinchao Wang. Learning-to-cache: Accelerating diffusion transformer via layer caching, 2024. URL https://arxiv.org/abs/2406.01733.

Dawid Malarz, Artur Kasymov, Maciej Zieba, Jacek Tabor, and Przemysław Spurek. Classifier-free guidance with adaptive scaling, 2025. URL https://arxiv.org/abs/2502.10574.

Charlie Nash, Jacob Menick, Sander Dieleman, and Peter W. Battaglia. Generating images with sparse representations, 2021. URL https://arxiv.org/abs/2103.03841.

Geon Yeong Park, Sang Wan Lee, and Jong Chul Ye. Inference-time diffusion model distillation, 2024. URL https://arxiv.org/abs/2412.08871.

William Peebles and Saining Xie. Scalable diffusion models with transformers, 2023. URL https://arxiv.org/abs/2212.09748.

Seyedmorteza Sadat, Jakob Buhmann, Derek Bradley, Otmar Hilliges, and Romann M Weber. Cads: Unleashing the diversity of diffusion models through condition-annealed sampling. *arXiv preprint arXiv:2310.17347*, 2023.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models, 2022. URL https://arxiv.org/abs/2202.00512.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans, 2016. URL https://arxiv.org/abs/1606.03498.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2022. URL https://arxiv.org/abs/2010.02502.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations, 2021. URL https://arxiv.org/abs/2011.13456.

Xi Wang, Nicolas Dufour, Nefeli Andreou, Marie-Paule Cani, Victoria Fernández Abrevaya, David Picard, and Vicky Kalogeiton. Analysis of classifier-free guidance weight schedulers. *arXiv preprint arXiv:2404.13040*, 2024.

Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, Jan Peters, and Jürgen Schmidhuber. Natural evolution strategies. *The Journal of Machine Learning Research*, 15(1):949–980, 2014.

Felix Wimbauer, Bichen Wu, Edgar Schoenfeld, Xiaoliang Dai, Ji Hou, Zijian He, Artsiom Sanakoyeu, Peizhao Zhang, Sam Tsai, Jonas Kohler, Christian Rupprecht, Daniel Cremers, Peter Vajda, and Jialiang Wang. Cache me if you can: Accelerating diffusion models through block caching, 2024. URL https://arxiv.org/abs/2312.03209.

Stephen J. Wright. Coordinate descent algorithms, 2015. URL https://arxiv.org/abs/1502.04759.

WANG Xi, Nicolas Dufour, Nefeli Andreou, CANI Marie-Paule, Victoria Fernandez Abrevaya, David Picard, and Vicky Kalogeiton. To guide or not to guide: Improving diffusion sampling with progressive guidance. *OpenReview*, 2024.

Tianze Yang, Yucheng Shi, Mengnan Du, Xuansheng Wu, Qiaoyu Tan, Jin Sun, and Ninghao Liu. Concept-centric token interpretation for vector-quantized generative models. *arXiv preprint arXiv:2506.00698*, 2025.

Shai Yehezkel, Omer Dahary, Andrey Voynov, and Daniel Cohen-Or. Navigating with annealing guidance scale in diffusion space, 2025. URL https://arxiv.org/abs/2506.24108.

Sun Yi, Daan Wierstra, Tom Schaul, and Jürgen Schmidhuber. Stochastic search using the natural gradient. In *Proceedings of the 26th annual international conference on machine learning*, pp. 1161–1168, 2009.

Huixuan Zhang, Junzhe Zhang, and Xiaojun Wan. How much to guide: Revisiting adaptive guidance in classifier-free guidance text-to-vision diffusion models, 2025. URL https://arxiv.org/abs/2506.08351.

Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator, 2023. URL https://arxiv.org/abs/2204.13902.

Candi Zheng and Yuan Lan. Characteristic guidance: Non-linear correction for diffusion model at large guidance scale, 2024. URL https://arxiv.org/abs/2312.07586.

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. Dip-go: A diffusion pruner via few-step gradient optimization, 2024. URL https://arxiv.org/abs/2410.16942.

Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching, 2025. URL https://arxiv.org/abs/2410.05317.

# A Appendix

## A.1 The Use of Large Language Models

We used large language models as writing assistants to improve the clarity and grammatical correctness of our manuscript. Specifically, we used LLM to refine sentence structure, correct grammatical errors, and enhance readability for audiences.

## A.2 Motivation

### A.2.1 Denoising Deviations Under Variable Guidance - math

The sparse guidance schedules from Stage 1 introduce denoising deviations that affect caching effectiveness. We analyze two primary scenarios that arise from our variable guidance patterns.

**Scenario 1: Guidance Scale Variation** When consecutive timesteps both apply CFG but with different scales ($w_t^* > \tau$ and $w_{t-1}^* > \tau$), we derive the resulting deviation.

Starting from the DDIM update (Song et al., 2022) with deterministic sampling ($\sigma_t = 0$):

$$x_{t-1} = \sqrt{\frac{\bar{\alpha}_{t-1}}{\bar{\alpha}_t}} x_t + \beta_{t-1,t} \cdot \tilde{\epsilon}_\theta(x_t, t) \tag{15}$$

where $\beta_{t-1,t} = \sqrt{1 - \bar{\alpha}_{t-1}} - \sqrt{\frac{\bar{\alpha}_{t-1}(1-\bar{\alpha}_t)}{\bar{\alpha}_t}}$.

With CFG, the noise prediction is:

$$\tilde{\epsilon}_\theta(x_t, t) = \epsilon_u(x_t, t) + w_t^*[\epsilon_c(x_t, t) - \epsilon_u(x_t, t)] \tag{16}$$

When guidance scale changes from $w_t^*$ to $w_{t-1}^*$, the deviation becomes:

$$\Delta x_{t-1}^{\text{scale}} = \beta_{t-1,t}(w_{t-1}^* - w_t^*)[\epsilon_c(x_t, t) - \epsilon_u(x_t, t)] \tag{17}$$

For the approximation in Equation 10 of the main text, we use $\beta_{t-1,t} \approx \sqrt{\bar{\alpha}_{t-1}/\bar{\alpha}_t}$ for clarity.

**Scenario 2: Guidance Mode Switching** When timestep $t$ uses CFG ($w_t^* > \tau$) but timestep $t-1$ switches to conditional-only ($w_{t-1}^* < \tau$), this is equivalent to switching from $w_t^*$ to $w_{t-1}^* = 1$ (since conditional-only means $\tilde{\epsilon} = \epsilon_c$).

Following the same framework, the switching deviation is:

$$\Delta x_{t-1}^{\text{switch}} = \beta_{t-1,t}(1 - w_t^*)[\epsilon_c(x_t, t) - \epsilon_u(x_t, t)] \tag{18}$$

Note that when $w_t^* > 1$, this deviation has opposite sign compared to Scenario 1, creating an abrupt trajectory change.

**Impact on Feature Caching** These deviations directly affect cached feature validity. Equations 17 and 18 reveal that variable guidance creates trajectory discontinuities that exceed uniform calibration's correction capacity, motivating our adaptive rank allocation in Stage 2.

### A.2.2 Denoising Deviations Under Variable Guidance - Visualization

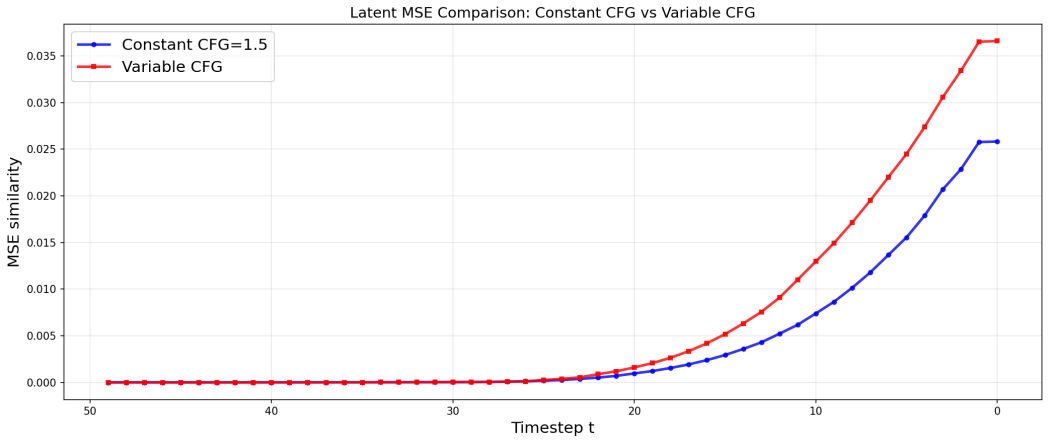


Figure 4: **Cumulative impact of variable guidance on final latent quality.** Mean squared error between 50-step generation with constant CFG=1.5 and variable CFG across denoising timesteps. Constant CFG (blue) maintains controlled error growth throughout denoising. Variable guidance from Stage 1 (red) causes accelerated error accumulation, resulting in 40% higher final latent error. This confirms that the block-wise reconstruction errors shown in Figure 5 accumulate through the network to degrade final MSE.

To empirically validate the theoretical deviations derived above, we conducted experiments measuring feature reconstruction errors under different guidance patterns. Figure 5 reveals that our sparse

guidance schedule from Stage 1 introduces substantially higher reconstruction errors compared to constant CFG across all transformer blocks, with particularly severe degradation in blocks 18-26. These block-wise errors are not isolated—Figure 4 demonstrates their cumulative impact, showing that variable guidance causes 40% higher final latent error compared to constant CFG, confirming that local reconstruction errors propagate through the network to degrade generation quality. While incremental calibration can mitigate these errors, Figure 6 exposes a critical limitation of uniform calibration: different transformer regions require different calibration ranks under variable guidance. Early and late blocks achieve better performance with lower ranks ($r = 256$), while middle blocks require higher ranks ($r = 512$) to minimize MSE. This heterogeneous error pattern across the network directly motivates our adaptive rank allocation strategy in Stage 2, which assigns region-specific calibration ranks rather than applying uniform calibration across all blocks.

### A.3 Additional Results

Table 5: Effect of reference generation length on optimized guidance schedules for DiT-XL/2.

| Methods | Steps | MACs(T) | IS↑ | FID↓ | sFID↓ | Precision↑ | Recall↑ |
|---|---|---|---|---|---|---|---|
| DDIM | 1000 | 1049.1 | 210.6 | 2.99 | 4.38 | 83.17 | 55.6 |
| DDIM | 50 | 52.45 | 203.8 | 3.25 | 4.53 | 83.27 | 56.4 |
| OUSAC w/o cache ($T_{ref}$ = 50) | 50 | 33.56 | 229.6 | 2.84 | 4.40 | 83.80 | 56.0 |
| OUSAC w/o cache ($T_{ref}$ = 500) | 50 | 34.09 | 225.0 | 2.77 | 4.39 | 83.49 | 56.2 |
| OUSAC w/o cache ($T_{ref}$ = 1000) | 50 | 30.94 | 228.3 | 2.71 | 4.38 | 83.43 | 57.0 |

**Does reference generation quality impact schedule discovery?** Table 5 shows how reference generation length affects the discovered guidance schedules. Short reference generations ($T_{ref} = 50$) lead to denser schedules with 33.56T MACs. As reference length increases to $T_{ref} = 1000$, the discovered schedule becomes sparser (30.94T MACs) with better FID (2.71), using guidance at only 9 out of 50 timesteps. Longer reference generations provide better targets for the evolutionary optimization to discover which timesteps truly require guidance.

Table 6: Ablation study of adaptive rank allocation strategies for DiT-XL/2.

| Methods | Steps | MACs(T) | IS↑ | FID↓ | sFID↓ | Precision↑ | Recall↑ |
|---|---|---|---|---|---|---|---|
| DiT-XL/2 (ImageNet 256×256) | | | | | | | |
| DDIM | 50 | 11.86 | 239.4 | 2.23 | 4.29 | 80.06 | 59.27 |
| ICC | 50 | 8.05 | 258.5 | 2.16 | 4.28 | 82.08 | 58.1 |
| OUSAC w/o cache | 50 | 6.63 | 263.0 | 2.10 | 4.29 | 81.85 | 58.87 |
| OUSAC ($r = 768$) | 50 | 6.88 | 273.5 | 2.12 | 4.28 | 82.89 | 57.79 |
| OUSAC ($r = 512$) | 50 | 5.81 | 276.9 | 2.48 | 4.79 | 83.17 | 56.36 |
| OUSAC ($r = 256$) | 50 | 4.74 | 271.5 | 2.27 | 4.53 | 82.97 | 57.06 |
| OUSAC | 50 | 5.07 | 266.5 | 2.04 | 4.29 | 82.09 | 58.75 |

Table 7: Ablation study of adaptive rank allocation strategies for PixArt-$\alpha$.

| Method | MACs | FID↓ | CLIP Score↑ |
|---|---|---|---|
| DPM-Solver (1000 steps) | 336.11 | 22.97 | 16.42 |
| DPM-Solver (20 steps) | 6.72 | 24.60 | 16.31 |
| OUSAC w/o cache | 4.37 | 22.69 | 16.39 |
| OUSAC ($r = 32$) | 2.63 | 20.05 | 16.49 |
| OUSAC ($r = 64$) | 2.70 | 19.92 | 16.52 |
| OUSAC | 2.67 | 19.27 | 16.48 |

**Adaptive versus uniform rank allocation.** Tables 6 and 7 validate our adaptive rank allocation strategy across both DiT-XL/2 and PixArt-$\alpha$. For DiT-XL/2 at 256×256 resolution, replacing OUSAC's adaptive ranks with uniform ranks across all blocks degrades performance significantly. Uniform rank $r = 256$ increases FID from 2.04 to 2.27, while higher uniform rank $r = 768$ achieves slightly better FID of 2.12 but requires 36% more computation (6.88T vs 5.07T MACs). Our coordinate descent optimization discovers region-specific rank distributions that achieve the best FID of 2.04 with only 5.07T MACs, demonstrating that different transformer regions require different calibration levels under variable guidance patterns. This heterogeneous requirement becomes even

more pronounced compared to ICC, which uses uniform calibration and achieves FID 2.16 with 8.05T MACs. Similarly, for PixArt-$\alpha$, OUSAC with adaptive rank allocation achieves FID 19.27, outperforming both uniform $r = 32$ (FID 20.05) and $r = 64$ (FID 19.92) configurations while maintaining comparable computational cost at 2.67 MACs. These results confirm that uniform calibration cannot adequately handle the varying reconstruction errors introduced by sparse guidance schedules, validating our approach of assigning calibration ranks based on each region's specific requirements under variable guidance conditions.

### A.4 Additional Qualitative Results

Figure 7 provides additional visual comparisons across 15 ImageNet classes. These examples further demonstrate that OUSAC achieves comparable visual quality to standard DDIM while using only 35% of the computational budget.

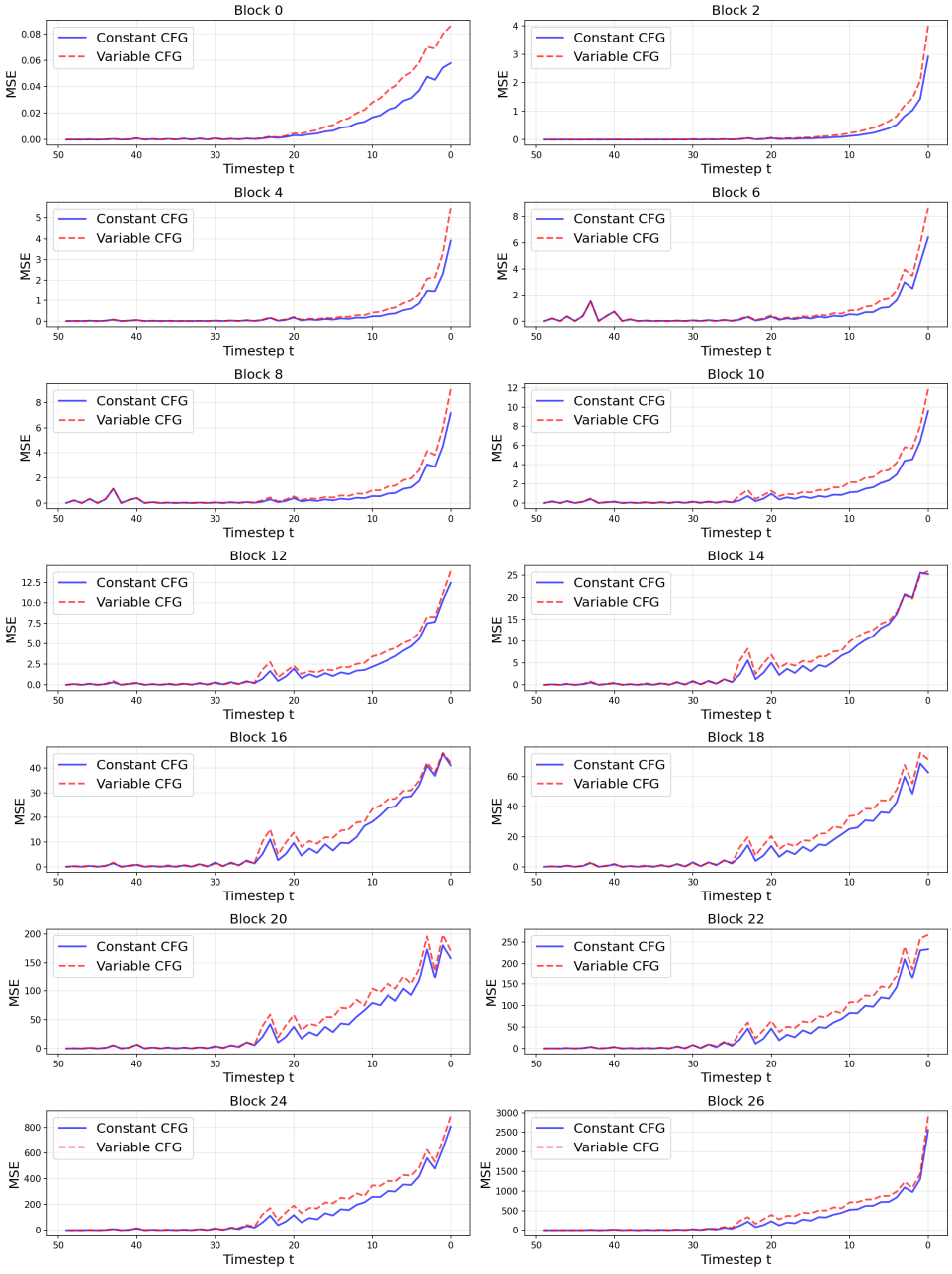


Figure 5: **Why we need incremental calibration for variable guidance patterns.** Each subplot shows the mean squared error between features at timestep $t$ and cached features from timestep $t+1$ in DiT-XL/2. With constant CFG at $w = 1.5$ (blue), reconstruction errors remain moderate across all blocks. However, our sparse guidance schedule from Stage 1 (red) causes substantially higher reconstruction errors across most transformer blocks, particularly in blocks 16 to 24. These elevated errors occur because variable guidance patterns create larger feature discrepancies between consecutive timesteps than standard caching assumes. This motivates us to use incremental calibration to correct these increased reconstruction errors, though the heterogeneous error distribution across blocks suggests that uniform calibration ranks may not be optimal.

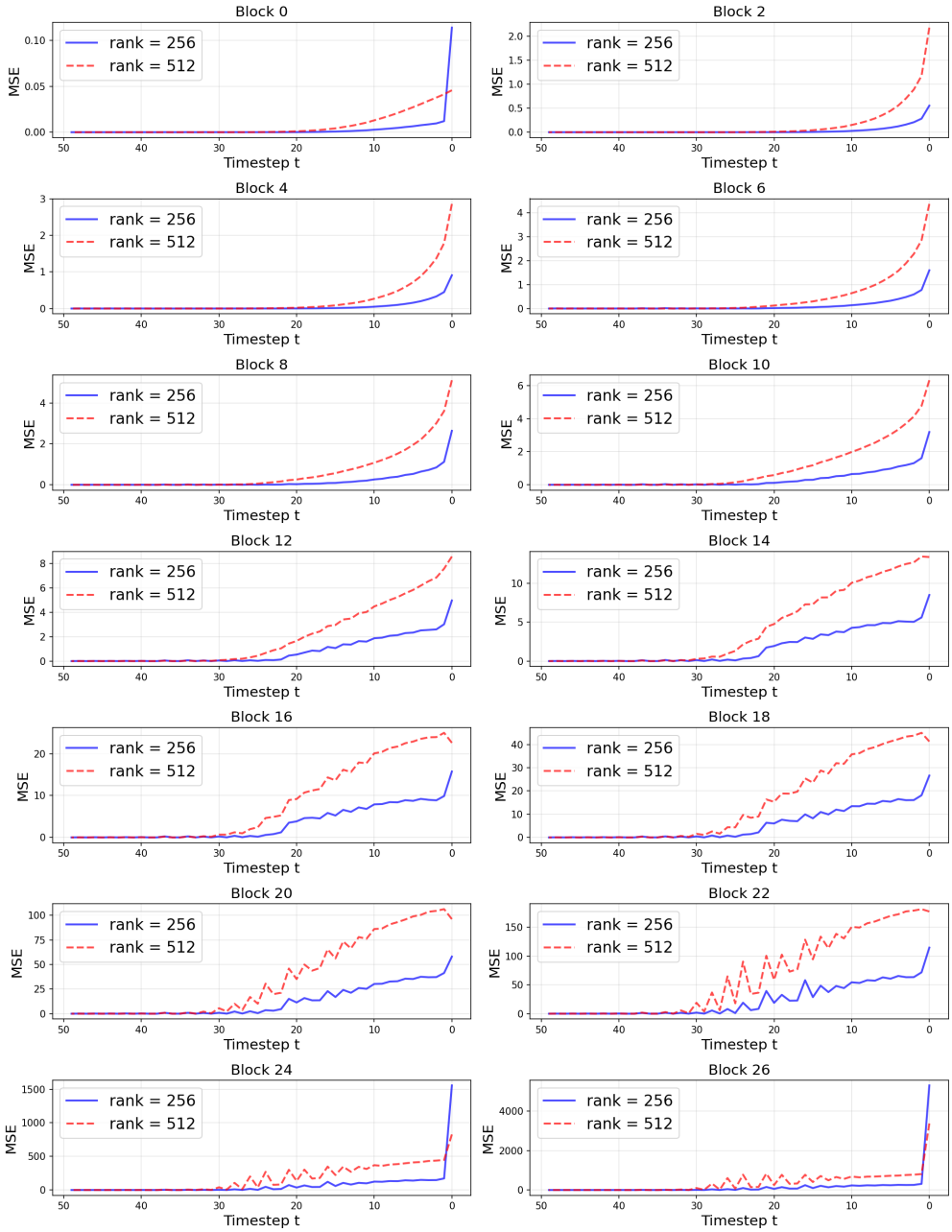


Figure 6: **Heterogeneous calibration requirements across transformer regions under variable guidance.** Mean squared error between features at timestep $t$ and cached features from timestep $t+1$ for DiT-XL/2 blocks with uniform rank $r = 256$ (blue) versus $r = 512$ (red). No single rank performs optimally across all blocks: early (0-2) and late blocks (24-26) achieve lower error with $r = 256$, while middle blocks (4-22) require $r = 512$ to preserve semantic information under variable guidance. This heterogeneous pattern demonstrates that uniform calibration cannot address the varying reconstruction errors introduced by sparse guidance schedules, motivating our adaptive rank allocation that assigns region-specific calibration ranks.

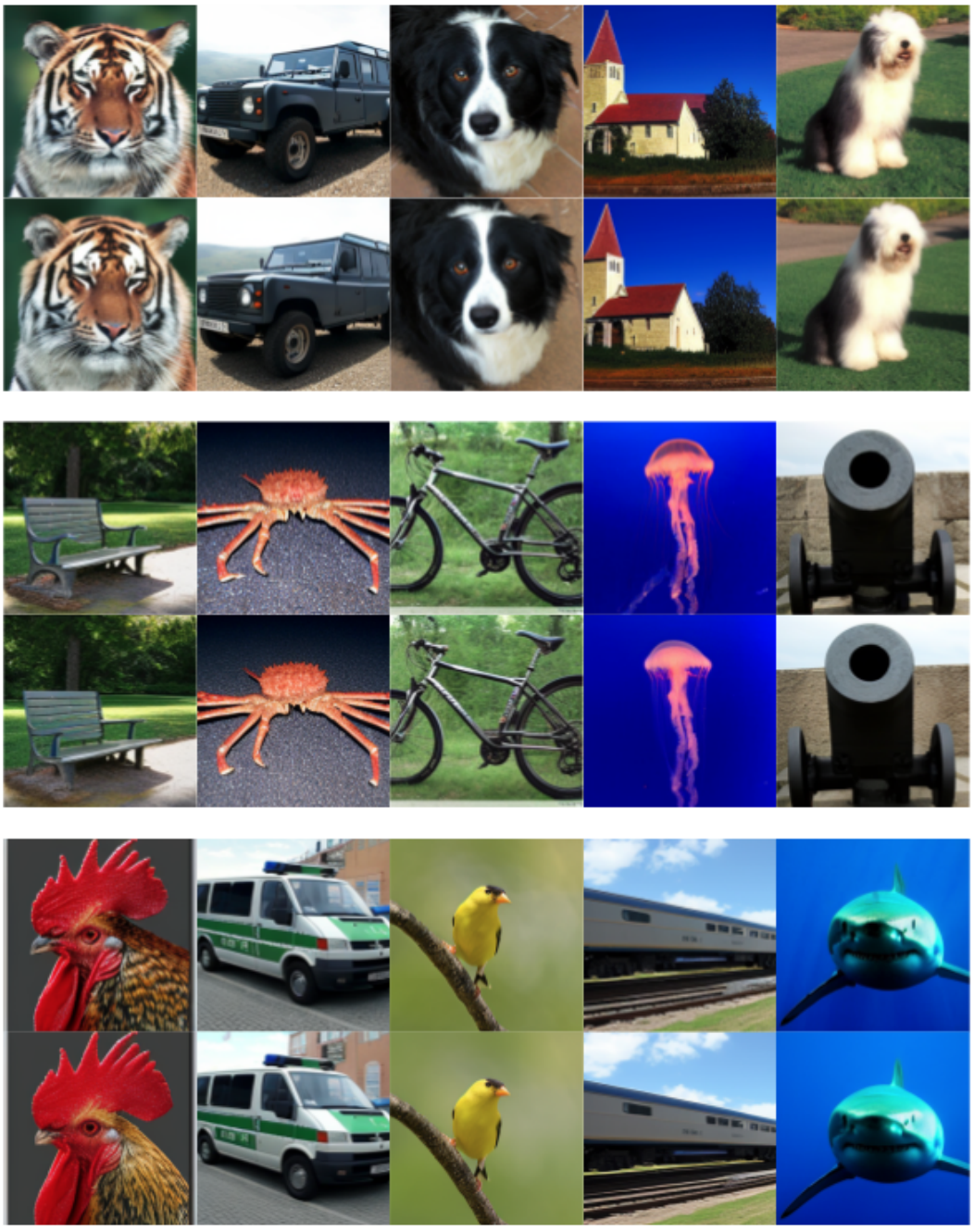

Figure 7: Visual quality comparison between standard DDIM (50 steps, 100% MACs) and OUSAC (50 steps, 35% MACs) on DiT-XL/2 at 256×256 resolution across diverse ImageNet classes. Each pair shows DDIM (top) and OUSAC (bottom) outputs from identical initial noise.