# Memory-Efficient Visual Autoregressive Modeling with Scale-Aware KV Cache Compression

**Kunjun Li**[♣,◇] **Zigeng Chen**[◇] **Cheng-Yen Yang**[♣] **Jenq-Neng Hwang**[♣]
University of Washington[♣] National University of Singapore[◇]
{kunjun, zigeng99}@u.nus.edu, {cycyang, hwang}@uw.edu


Figure 1: We introduce a new KV cache compression framework for Visual Autoregressive modeling that preserves pixel-level fidelity. On Infinity-8B, it achieves **10x** memory reduction from **85 GB** to **8.5 GB** with negligible quality degradation (GenEval score **remains** at **0.79** and DPG score marginally decreases from **86.61** to **86.49**).

## Abstract

Visual Autoregressive (VAR) modeling has garnered significant attention for its innovative next-scale prediction approach, which yields substantial improvements in efficiency, scalability, and zero-shot generalization. Nevertheless, the coarse-to-fine methodology inherent in VAR results in exponential growth of the KV cache during inference, causing considerable memory consumption and computational redundancy. To address these bottlenecks, we introduce ScaleKV, a novel KV cache compression framework tailored for VAR architectures. ScaleKV leverages two critical observations: varying cache demands across transformer layers and distinct attention patterns at different scales. Based on these insights, ScaleKV categorizes transformer layers into two functional groups: drafters and refiners. Drafters exhibit dispersed attention across multiple scales, thereby requiring greater cache capacity. Conversely, refiners focus attention on the current token map to process local details, consequently necessitating substantially reduced cache capacity. ScaleKV optimizes the multi-scale inference pipeline by identifying scale-specific drafters and refiners, facilitating differentiated cache management tailored to each scale. Evaluation on the state-of-the-art text-to-image VAR model family, Infinity, demonstrates that our approach effectively reduces the required KV cache memory to **10%** while preserving **pixel-level fidelity**. Code is available at `https://github.com/StargazerX0/ScaleKV`.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

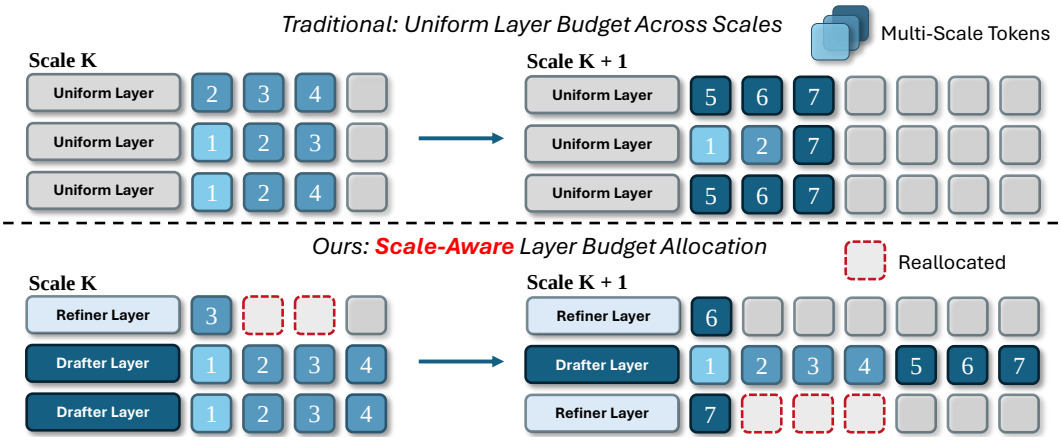



Figure 2: By implementing scale-aware layer budget allocation, ScaleKV enables differentiated cache management tailored to each layer's computational demands at every scale.

## 1 Introduction

Recent advances in Autoregressive (AR) models [51, 23, 68, 31] have achieved impressive image quality and multi-modal capabilities [34, 63, 61, 58, 60] for unified vision understanding and generation. However, the token-by-token generation approach requires numerous decoding steps. Visual Autoregressive (VAR) modeling [53] has revolutionized this process through next-scale prediction, enabling models to decode multiple tokens in parallel. Building upon this framework, several approaches [16, 52] have demonstrated promising results for VAR-based text-to-image generation.

Despite these advancements, VAR models face a fundamental scalability challenge due to exponential growth in token sequence across scales. Unlike traditional next-token prediction models, which process one token per step with linear KV cache growth, VAR models must preserve KV states from all previous token maps. Generating a 1024×1024 image requires processing over 10K tokens across multiple scales, creating severe memory bottlenecks—with KV cache alone consuming approximately 85 GB of memory when generating 1024×1024 images with a batch size of 8 using Infinity-8B [16], a text-to-image VAR model. Figure 3(a) illustrates this complexity, where each scale contributes a new KV cache entry of size $h_k \times w_k$. The total cache requirement grows cubically with the number of scales $n$, while the computational complexity of attention reaches $\mathcal{O}(n^4)$. These memory constraints and increased inference latency significantly impede practical deployment.

To address these inefficiencies, we analyze the specific properties of VAR's next-scale prediction paradigm. First, we observe that cache demands vary significantly across transformer layers. Next, we find that different scales exhibit distinct attention patterns. These findings indicate that VAR requires both layer-adaptive and scale-specific cache management strategies for optimal performance.

**Our Approach.** Inspired by these observations, we propose *Scale-Aware KV Cache* (ScaleKV), a simple yet highly effective method that significantly reduces inference memory while maintaining high generation quality. Our approach categorizes transformer layers into two functional groups: **Drafters** and **Refiners**. Drafters distribute attention across multiple scales to access global information from preceding tokens, requiring greater cache capacity. In contrast, refiners focus attention on current token map to process local details, necessitating substantially reduced cache storage. As illustrated in Figure 2, ScaleKV optimizes multi-scale inference by identifying scale-specific drafters and refiners, facilitating differentiated cache management to their computational demands at each scale.

Extensive evaluation demonstrates the effectiveness of our method. As shown in Figure 1, compared to the original Infinity-8B model, ScaleKV achieves negligible quality degradation (GenEval score remains at 0.79 and DPG score decreases slightly from 86.61 to 86.49) while requiring merely **10%** of the original GPU memory consumption. These results validate that ScaleKV effectively addresses the fundamental memory bottlenecks that have constrained the practical deployment of VAR models.

In conclusion, we introduce ScaleKV, a novel KV cache compression framework for VAR. ScaleKV categorizes transformer layers into drafters and refiners and implements scale-aware layer budget

allocation. Through extensive experiments, our method achieves significant memory reduction while preserving pixel-level fidelity, enabling efficient deployment in resource-constrained environments.

## 2 Related Works

**Autoregressive Visual Generation.** Early works [6, 54] pioneered pixel-by-pixel image generation, later enhanced by VQVAE [55] and VQGAN [8] through image patch quantization. Recent advances include GPT-style models [51, 34], mixture-of-experts [23], linear attention [26], diffusion-autoregressive hybrids [63, 81, 14], and masked approaches [26, 4, 38]. However, autoregressive approaches suffer from substantial inference latency due to sequential token generation. VAR [53] overcomes this limitation through hierarchical parallel decoding, and has been extended to text-to-image synthesis [16, 74, 52, 40, 27, 31], audio synthesis [45], and 3D content creation [73].

**Efficient Visual Generation.** For diffusion models, efficiency optimization methods are already well-developed. [47, 71, 37, 48, 69] focus on reducing sampling steps while [29, 80, 9, 72, 67, 28, 49, 24] optimize models through quantization [24], pruning [9] or knowledge distillation [19]. Several approaches [41, 59, 76, 66, 50, 25, 39, 79] skip redundant computations during the denoising process.

However, research on memory optimization for VAR image generation remains in its early stages. LiteVAR [64] and FastVAR [15] enhance inference speed but do not address fundamental memory bottlenecks, while CoDe [7] improves memory efficiency through collaborative decoding but requires an additional VAR model. Hack [44] reduces memory via per-head budgets, but its irregular tensor shapes require specialized kernels. In contrast, our approach directly reduces memory consumption and integrates with existing techniques to enhance efficiency.

**KV Cache Compression.** Current KV cache compression techniques for large language models (LLMs) and vision-language models (VLMs) primarily utilize quantization [36, 70, 22, 17], eviction [78, 35, 42, 46, 30], and merging [77, 33, 56, 57] strategies. Quantization reduces precision but faces granularity limitations; eviction removes less important tokens using attention metrics while optimizing allocation within budgets [12, 65, 10, 11]; and merging consolidates redundant KV pairs.

However, primarily designed for single-sequence processing in LLMs and VLMs, existing techniques fail to accommodate the multi-scale operational characteristics of VAR and the varying attention patterns across layers and scales.

## 3 Methods

### 3.1 Preliminary

Visual Autoregressive modeling [53] advances traditional AR approaches by shifting the prediction paradigm from "next token" to "next scale." Within this framework, each autoregressive operation produces a token map corresponding to a specific resolution scale, rather than individual tokens.

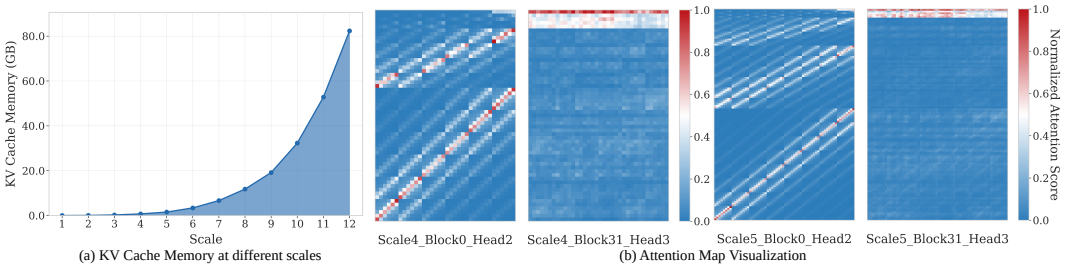
Given an image feature map $f \in \mathbb{R}^{h \times w \times C}$, VAR quantizes it into $K$ multi-scale token maps $R = (r_1, r_2, \ldots, r_K)$ with increasingly finer resolutions. The joint probability distribution over these multi-scale maps is decomposed autoregressively according to:

$$p(r_1, r_2, \ldots, r_K) = \prod_{k=1}^{K} p(r_k \mid r_1, r_2, \ldots, r_{k-1}), \tag{1}$$

where each token map $r_k \in \{1, \ldots, V\}^{h_k \times w_k}$ contains $h_k \times w_k$ discrete tokens, selected from a vocabulary of size $V$ at scale $k$. At each autoregressive step $k$, the model generates all $h_k \times w_k$ tokens comprising $r_k$ in parallel, conditioning on previously generated scales $(r_1, \ldots, r_{k-1})$. While VAR provides substantial improvements in both inference efficiency and generation quality, this coarse-to-fine generation strategy significantly expands the sequence length.

### 3.2 Key Observations

VAR represents an innovative paradigm that diverges from traditional autoregressive approaches. In this work, we examine the next-scale prediction process to identify properties that can be leveraged to reduce computational redundancy. Our analysis focuses specifically on attention patterns in VAR.



(a) KV Cache Memory at different scales (b) Attention Map Visualization

Figure 3: (a) Exponential KV cache growth. (b) Visualization of two distinct attention patterns.

**Cache demands vary significantly across transformer layers.** Our analysis revealed two distinct attention pattern typologies in VAR models, visualized in Figure 3(b). The left pattern (Block0_Head2) exhibits a diagonal structure with dispersed attention that spans preceding scales, indicating broad contextual integration and high cache demand. In contrast, the right pattern (Block31_Head3) demonstrates highly concentrated attention focused predominantly within the current token map, suggesting localized processing with minimal cache requirements. Through systematic analysis of these attention maps, we identified that most layers fall into one of these two categories, which we term **Drafters** and **Refiners**. Drafters distribute attention broadly across historical context, necessitating substantial KV cache capacity to access multiple scales. Refiners, however, concentrate attention primarily on local information within the current token map, requiring significantly reduced cache storage.

**Different scales exhibit distinct attention patterns.** As VAR progresses through its hierarchy, both groups display scale-dependent evolution. Drafter layers exhibit increasingly dispersed attention patterns at higher scales to integrate broader contextual information. Conversely, refiners grow progressively more concentrated, as evidenced by the heightened focus in Scale5_Block31_Head3 compared to its Scale4 counterpart (Figure 3(b)). This bidirectional evolution reveals a specialized hierarchical process where drafters gather global context while refiners perform localized processing.

These findings challenge both uniform cache allocation [62] and position-based cache reduction [3] employed by current methods, suggesting that VAR models would benefit from adaptive allocation strategies accounting for both layer-specific requirements and scale-dependent characteristics.

### 3.3 Scale-Aware KV Cache

Based on our observations, we propose a simple yet highly effective KV cache compression framework for next-scale prediction called Scale-Aware KV Cache. As illustrated in Figure 4, ScaleKV categorizes transformer layers into two functional groups termed **Drafters** and **Refiners**, implementing adaptive cache management strategies based on these roles. This approach optimizes multi-scale inference by identifying each layer's function at every scale, enabling adaptive cache allocation that aligns with specific computational demands of each layer.

**Identifying Drafter and Refiner Layers.** To systematically distinguish between drafter and refiner layers across different scales, we introduce the **Attention Selectivity Index (ASI)**. This metric quantifies each layer's attention patterns by considering two critical factors: (1) the proportion of attention directed to current token map and (2) the concentration of attention in history sequence.

Let $\alpha_{i,j}^{(l,k)}$ represent the normalized attention score from query position $i$ in the current map $r_k$ to key/value position $j$ in layer $l$ when processing scale $k$. The token indices can be partitioned into history indices $\mathcal{P}_{k-1}$ (from previously generated maps $r_1, \ldots, r_{k-1}$) and current map indices $\mathcal{C}_k$ (from $r_k$). The ASI for layer $l$ at scale $k$ is defined as:

$$ASI^{(l,k)} = \underbrace{\mathbb{E}_{i\sim\mathcal{U}(H_k)}\left[\sum \alpha_{i,j}^{(l,k)},\ j \in \mathcal{C}_k\right]}_{\text{Current Attention Ratio}} \cdot \underbrace{\mathbb{E}_{i\sim\mathcal{U}(H_k)}\left[\text{TopKSum}'\left(\{\alpha_{i,j}^{(l,k)} \mid j \in \mathcal{P}_{k-1}\}\right)\right]}_{\text{History Top-K Ratio}}, \quad (2)$$

where $\mathbb{E}_{i\sim\mathcal{U}(H_k)}[\cdot]$ denotes expectation over query positions $i$ sampled uniformly from the current map $r_k$, and $\text{TopKSum}'(\cdot)$ computes the sum of the top-$K'$ attention scores directed toward the history tokens. The parameter $K'$ controls the number of top scores included in the selectivity term.

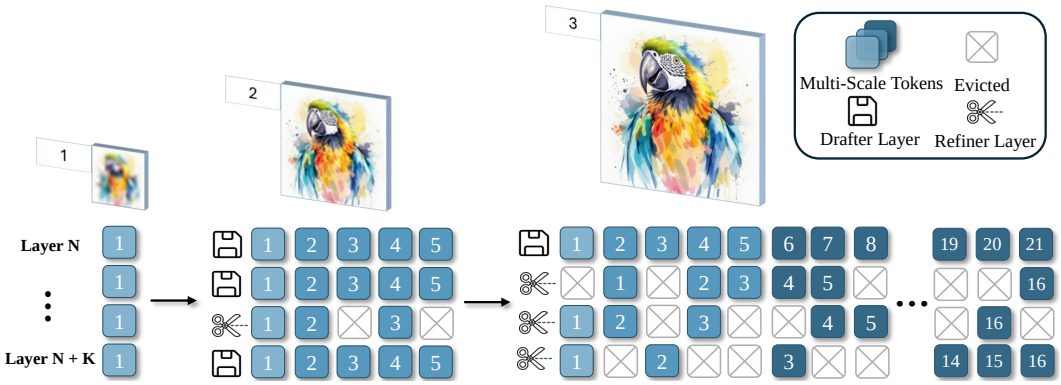



Figure 4: Overview of ScaleKV. Our method categorizes transformer layers into drafters (require extensive cache for global context) or refiners (process local details with minimal cache). This scale-wise identification enables adaptive cache allocation based on each layer's computational demands.

Intuitively, a high $ASI^{(l,k)}$ value indicates that the layer either focuses strongly on the current map or exhibits high selectivity toward specific history tokens, or both. This suggests the layer is functioning as a refiner. Conversely, a low $ASI^{(l,k)}$ value indicates the layer distributes attention more broadly across the prefix context, characteristic of drafter behavior.

Since raw $ASI^{(l,k)}$ values vary significantly across scales due to differences in token counts and attention patterns, we normalize these values within each scale using Z-scores. Let $\mathcal{S} = \{(l,k) \mid 1 \le l \le L, 1 \le k \le K\}$ be the set of all layer-scale pairs. We rank these pairs by their Z-scores and define the set of drafters $\mathcal{D}$ as the $N_d$ pairs with the lowest Z-scores:

$$\mathcal{D} = \{(l,k) \in \mathcal{S} \mid Z^{(l,k)} \le Z_{(N_d)}\}, \qquad Z^{(l,k)} = \frac{ASI^{(l,k)} - \mu_k}{\sigma_k + \epsilon}, \tag{3}$$

where $Z_{(N_d)}$ represents the $N_d$-th smallest Z-score. The remaining constitute the refiners $\mathcal{R} = \mathcal{S} \setminus \mathcal{D}$.

The identification process occurs prior to inference using minimal calibration data. Our experiments demonstrate that a set of 10 prompts is sufficient to accurately determine the drafters and refiners.

**Cache Budget Allocation.** After identifying drafters and refiners, we establish an efficient budget allocation strategy that satisfies the same total memory consumption as uniform budget allocation $B_{\text{uniform}}$ while implementing a scale-dependent reduction for refiners:

$$\sum_{k=1}^{K} (N_d^k \cdot B_d(k) + N_r^k \cdot B_r(k)) = B_{\text{uniform}} \cdot L \cdot K, \qquad B_r(k) = B_r(0) - \delta \cdot k. \tag{4}$$

Here, $N_d^k$ and $N_r^k$ represent drafter/refiner layer counts at scale $k$, while $B_d(k)$ and $B_r(k)$ denote their respective cache budgets. The parameter $\delta$ controls the refiner budget decay rate. By leveraging the second observation that refiner attention exhibits increasing concentration at higher scales, refiner cache budgets are linearly reduced from the initial refiner budget $B_r(0)$ as scale $k$ increases. The saved memory is subsequently reallocated to drafters, ensuring $B_d(k) \gg B_r(k)$ to align with the scale-specific computational demands of each layer.

**KV Cache Selection.** After establishing cache budgets for drafter and refiner layers, we implement an efficient token selection strategy to determine which specific KV states should be preserved.

For each token map $r_k$, we first partition the map into $N$ patches and select the centroid token from each patch to form an observation window $\mathcal{W}$. This sampling approach ensures spatial coverage across the token map while maintaining a minimal memory footprint. We then evaluate the relative importance of the remaining tokens based on their attention interactions with the observation window.

Similar to [30], for each attention head $h$, we compute an importance score $s_i^h$ for each token $i$ by measuring the cumulative attention it receives from the observation window tokens:

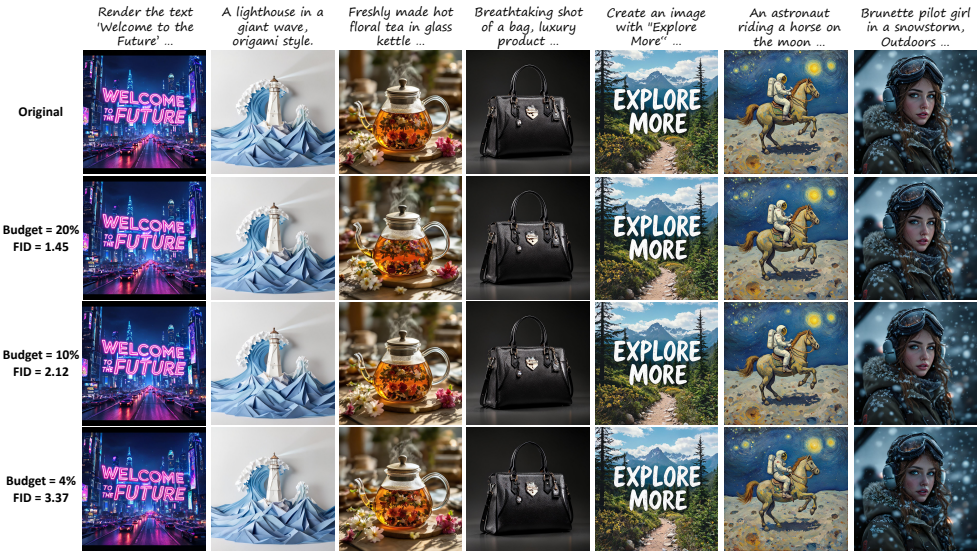



Figure 5: Qualitative comparison between the original Infinity-8B model and our proposed ScaleKV.

$$s_i^h = \sum_{j \in \mathcal{W}} A_{ji}^h, \qquad A^h = \text{softmax}(Q^h \cdot (K^h)^\top / \sqrt{d_k}). \tag{5}$$

Here, $A_{ji}^h$ represents the normalized attention weight from query token $j$ in the observation window to key token $i$, and $d_k$ denotes the dimension of the key vectors. This formulation effectively quantifies each token's contribution to the contextual representation of the observation window.

After calculating cumulative attention, we aggregate these scores through average pooling to produce a unified importance metric. For each layer $l$, we then select the top-$k^l$ tokens with the highest aggregated importance scores, where $k^l$ corresponds to the layer-specific cache budget determined by our allocation strategy. Only the KV states of these selected tokens, along with the observation window tokens, are preserved in the cache for subsequent steps, while all others are efficiently pruned.

The observation window is compact (typically comprising only 16 tokens), enabling efficient attention score computation between this window and the remaining tokens with negligible computational overhead. Rather than hindering performance, our experimental results show this approach yields notable speedups of up to 1.25× through significantly reduced KV cache memory requirements.

# 4 Experiments

## 4.1 Experimental Setup

**Base Models**. We evaluated ScaleKV on two VAR-based text-to-image models of different capacities: Infinity-2B and Infinity-8B [16], to validate our method's generalizability across model scales. To represent practical deployment scenarios with varying resource constraints, we analyzed performance under three memory budget constraints: 4%, 10%, and 20% of the original KV cache size.

**Dataset and Metrics**. We assessed output consistency with the original models using the MS-COCO 2017 [32] validation set, which comprises 5,000 images and captions. Consistency was measured by the Fréchet Inception Distance (FID) [18], Learned Perceptual Image Patch Similarity (LPIPS) [75], and Peak Signal-to-Noise Ratio (PSNR). We also used two established image generation benchmarks GenEval [13] and DPG [20] for perceptual quality and semantic alignment with input prompts. For memory efficiency, we report the KV cache memory usage measured with a batch size of 8. We use GPT-4o [21] to generate 10 prompts for calibrating drafters and refiners. Our drafter/refiner identification method is effective across different calibration prompt sources, with results converging after analyzing only a small sample of prompts, as detailed in the appendix.

Table 1: Quantitative comparisons of output consistency on MS-COCO 2017 dataset.

| Method | Budget | Infinity-2B | | | | Infinity-8B | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | KV Cache | FID ↓ | LPIPS ↓ | PSNR ↑ | KV Cache | FID ↓ | LPIPS ↓ | PSNR ↑ |
| Full Cache | 100% | 38550 MB | - | - | - | 84328 MB | - | - | - |
| Sliding Window [1] | 20% | 7800 MB | 5.63 | 0.17 | 20.71 | 17062 MB | 4.82 | 0.14 | 20.99 |
| StreamingLLM [62] | 20% | 7800 MB | 3.85 | 0.12 | 22.00 | 17062 MB | 3.94 | 0.14 | 21.65 |
| SnapKV [30] | 20% | 7800 MB | 3.25 | 0.12 | 22.51 | 17062 MB | 3.10 | 0.10 | 22.65 |
| PyramidKV [3] | 20% | 7800 MB | 3.23 | 0.11 | 22.62 | 17062 MB | 3.03 | 0.10 | 22.76 |
| **ScaleKV** | 20% | 7800 MB | **1.82** | **0.08** | **24.84** | 17062 MB | **1.45** | **0.06** | **25.60** |
| Sliding Window [1] | 10% | 3900 MB | 8.58 | 0.24 | 18.99 | 8531 MB | 8.71 | 0.20 | 19.02 |
| StreamingLLM [62] | 10% | 3900 MB | 5.49 | 0.19 | 19.79 | 8531 MB | 6.29 | 0.17 | 19.97 |
| SnapKV [30] | 10% | 3900 MB | 4.66 | 0.16 | 20.83 | 8531 MB | 4.68 | 0.15 | 20.60 |
| PyramidKV [3] | 10% | 3900 MB | 4.52 | 0.16 | 20.92 | 8531 MB | 4.69 | 0.14 | 20.79 |
| **ScaleKV** | 10% | 3900 MB | **2.53** | **0.11** | **22.64** | 8531 MB | **2.12** | **0.09** | **23.25** |
| Sliding Window [1] | 4% | 1590 MB | 16.68 | 0.30 | 17.49 | 3478 MB | 19.23 | 0.27 | 17.50 |
| StreamingLLM [62] | 4% | 1590 MB | 8.71 | 0.25 | 18.31 | 3478 MB | 8.54 | 0.22 | 18.63 |
| SnapKV [30] | 4% | 1590 MB | 5.10 | 0.24 | 18.23 | 3478 MB | 6.68 | 0.19 | 19.15 |
| PyramidKV [3] | 4% | 1590 MB | 5.51 | 0.23 | 18.65 | 3478 MB | 6.55 | 0.19 | 19.26 |
| **ScaleKV** | 4% | 1590 MB | **3.51** | **0.16** | **20.82** | 3478 MB | **3.37** | **0.12** | **21.41** |

Table 2: Quantitative comparisons of perceptual quality on GenEval and DPG Benchmarks.

| Methods | # Params | GenEval | | | | DPG | | |
|---|---|---|---|---|---|---|---|---|
| | | Two Obj. | Position | Color Attri. | Overall ↑ | Global | Relation | Overall ↑ |
| SDXL [43] | 2.6B | 0.74 | 0.15 | 0.23 | 0.55 | 83.27 | 86.76 | 74.65 |
| LlamaGen [51] | 0.8B | 0.34 | 0.07 | 0.04 | 0.32 | - | - | 65.16 |
| Show-o [63] | 1.3B | 0.80 | 0.31 | 0.50 | 0.68 | - | - | 67.48 |
| PixArt-Sigma [5] | 0.6B | 0.62 | 0.14 | 0.27 | 0.55 | 86.89 | 86.59 | 80.54 |
| HART [52] | 0.7B | 0.62 | 0.13 | 0.18 | 0.51 | - | - | 80.89 |
| DALL-E 3 [2] | - | - | - | - | 0.67 | 90.97 | 90.58 | 83.50 |
| Emu3 [58] | 8.5B | 0.81 | 0.49 | 0.45 | 0.66 | - | - | 81.60 |
| Infinity-2B [16] | 2.0B | 0.84 | 0.43 | 0.57 | **0.725** | 89.01 | 90.03 | **83.06** |
| **+ ScaleKV (10%)** | 2.0B | 0.84 | 0.44 | 0.55 | **0.730** | 82.45 | 90.48 | **83.01** |
| Infinity-8B [16] | 8.0B | 0.89 | 0.61 | 0.68 | **0.792** | 89.51 | 93.08 | **86.61** |
| **+ ScaleKV (10%)** | 8.0B | 0.89 | 0.61 | 0.67 | **0.790** | 92.49 | 88.92 | **86.49** |

**Compression Baselines.** We assessed ScaleKV's performance mainly against four representative KV cache compression baselines: Sliding Window Attention [1] (retains local window of most recent tokens), StreamingLLM [62] (keeps attention sinks (initial tokens) and recent tokens), SnapKV [30] (clusters tokens based on attention scores) and PyramidKV [3] (employs a fixed pyramid-shaped allocation strategy across transformer layers).

### 4.2 Main Results

**Comparison with Compression Baselines**. Table 1 presents our evaluation on the MS-COCO 2017 validation set [32], focusing on pixel-level consistency with the original outputs. ScaleKV consistently outperforms all baselines across different memory budgets, with significant improvements in FID, LPIPS, and PSNR metrics.

At the most constrained budget (4%), ScaleKV achieves FID reductions of 31.2% and 48.5% compared to the next best baseline for Infinity-2B and Infinity-8B. The performance gap widens at higher budgets, with ScaleKV achieving FID scores of 1.82 and 1.45 at 20% budget, representing substantial improvements over all competitors. The LPIPS results further validate these findings, with ScaleKV achieving scores of 0.08 and 0.06 at 20% budget for the two models, compared to PyramidKV's 0.11 and 0.10, indicating better perceptual similarity to the original outputs.

Each baseline exhibits specific limitations in the VAR context: Sliding Window Attention and StreamingLLM employ static policies that lose information carried by middle tokens, resulting in poor performance at low budgets. SnapKV's clustering strategy helps preserve some image coherence but cannot effectively prioritize critical tokens across different scales. Notably, PyramidKV's fixed allocation pattern offers limited improvement over SnapKV and sometimes produces worse results

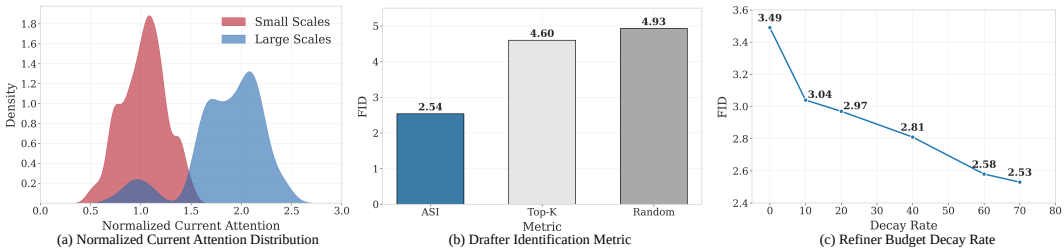



Figure 6: (a) Kernel Density Estimation of normalized current attention scores at small scales $(r_2, r_3, r_4)$ and large scales $(r_{10}, r_{11}, r_{12})$. (b) Ablation experiments on using different drafter identification metrics. (c) Ablation experiments on the impact of refiner budget decay rate.

(e.g., 4% budget on Infinity-2B with FID 5.51 vs. SnapKV's 5.10), confirming that predetermined allocation strategies do not generalize well to VAR's scale-dependent attention behaviors.

**Results on GenEval and DPG**. To further validate the perceptual quality and semantic understanding capabilities of our compressed models, we conducted evaluations on two established benchmarks: GenEval [13] and DPG [20]. Table 2 presents these results, comparing our ScaleKV-compressed models against state-of-the-art image generation models, including diffusion models [43, 5, 2] and autoregressive models [51, 63, 58, 52]. The results demonstrate that ScaleKV preserves semantic understanding remarkably well despite substantial KV cache reduction. For Infinity-2B, our method delivers exceptional performance that matches the full model (83.01 vs. 83.06 on DPG), with the GenEval score even showing a slight improvement from 0.725 to 0.730, while using only 10% of the original KV cache. Similarly, for Infinity-8B, ScaleKV maintains nearly identical performance (0.790 vs. 0.792 on GenEval, and 86.49 vs. 86.61 on DPG). This minimal performance degradation is particularly noteworthy given that Infinity models already outperform most existing approaches on these benchmarks, including larger models like DALL-E 3 and Emu3-8.5B. ScaleKV-compressed Infinity-8B maintains this superior performance while requiring only 8.5 GB of KV cache memory, a dramatic reduction from the original 85 GB.

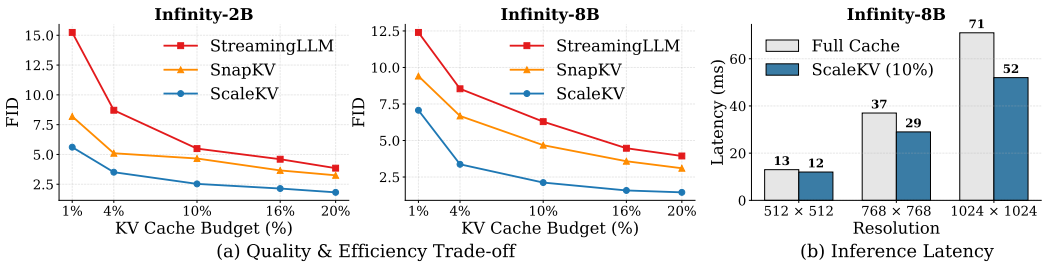
**Qualitative Results**. We provide an extensive qualitative comparison between the Infinity-8B model with full KV cache and our proposed ScaleKV, with varying budgets of 4%, 10%, and 20%. As illustrated in Figure 5, our approach achieves significant memory optimization, with only minimal quality degradation that is nearly imperceptible to the human eye. Even at a compression rate of 25 times, the generated images maintain exceptionally high quality and accurate semantic information.

### 4.3 Analytical Experiments

**Attention Distributions Across Scales.** We quantify attention pattern variations throughout the multi-scale generation using *normalized current scale attention*, defined as the average attention per token within the current scale to the average attention across the entire sequence, to capture contribution of the cached tokens to generation. Figure 6(a) demonstrates the kernel density estimation (KDE) of normalized current attention, revealing distinct distributions across small scales $(r_2, r_3, r_4)$ and large scales $(r_{10}, r_{11}, r_{12})$. Small scales exhibit an approximately uniform distribution, indicating broad context utilization without strong selectivity. In contrast, large scales concentrate around higher attention values, suggesting focused information selection. This progression reflects an evolution from global information aggregation in early scales to selective attention refinement in later scales.

**Impact of Drafter Identification Metric.** Figure 6(b) demonstrates the comparative effectiveness of different metrics for drafter identification. Our proposed Attention Selectivity Index (ASI) (Equation 2) combines the attention score ratio in the current token map and the Top-K ratio in history sequence. When evaluated on Infinity-2B with a 10% cache budget constraint, ASI achieves an FID score of 2.53, representing a substantial 42.5% improvement over using only the Top-K ratio (4.60). This significant performance gap validates that our comprehensive attention pattern analysis effectively identifies layers requiring larger cache allocation, enabling optimal resource distribution.

**Impact of Refiner Budget Decay Rate.** Figure 6(c) shows the effectiveness of refiner budget decay strategy (Equation 4) under a 10% budget constraint (650 tokens per head/layer). With an initial refiner budget of 600 tokens, we observe a consistent improvement in FID from 3.49 to 2.53 as decay




(a) Quality & Efficiency Trade-off (b) Inference Latency

Figure 7: (a) FID under different KV cache budgets. (b) Inference latency for different resolutions.

Table 3: Memory usage comparison across different batch sizes

| Method | Memory Consumption↓ | | | |
|---|---|---|---|---|
| | Running | KV Cache | Params | Total |
| Infinity (bs=1) | 1.1GB | 10.5GB | 19.5GB | 31.4GB |
| **+ScaleKV (10%)** | 0.8GB | **1.1GB** | 19.5GB | 21.4GB |
| Infinity (bs=8) | - | 85GB | 19.5GB | OOM (>100GB) |
| **+ScaleKV (10%)** | 21.1GB | **8.5GB** | 19.5GB | 49.2GB |
| Infinity (bs=16) | - | 170GB | 19.5GB | OOM (>100GB) |
| **+ScaleKV (10%)** | 42.4GB | **17.1GB** | 19.5GB | 78.8GB |

rate increases from 0 to 70, confirming our observation that refiner attention becomes increasingly focused at higher scales, requiring fewer resources. The monotonic improvement also validates our drafter identification method, as reallocating cache capacity from refiners to drafters consistently enhances generation quality, indicating accurate identification of layers with divergent cache demands.

**Quality-Efficiency Trade-off.** We evaluated the quality-efficiency trade-off of ScaleKV against established compression methods across different cache budgets, as shown in Figure 7(a). For both Infinity-2B and Infinity-8B models, ScaleKV consistently achieves the lowest FID at all budget constraints, with performance gap widening at more restrictive memory allocations. These results demonstrate that ScaleKV effectively preserves generation quality even under severe memory constraints, making it adaptable to diverse deployment scenarios with varying computational resources.

**Memory Efficiency and Time Cost Analysis.** In Table 3, we present a comprehensive analysis of memory consumption during the Infinity-8B model inference process. The KV cache of the Infinity model is the largest memory consumer, requiring approximately 10 times the memory needed for the model's decoding operation due to the significantly extended sequence length. Our proposed ScaleKV drastically reduces the KV cache memory requirements, compressing it to 10% of the original model. Moreover, as batch size increases, the memory savings with ScaleKV become even more pronounced. We are able to generate images with a large batch size of 16 using less than 80GB total memory, whereas the KV cache alone of the original model requires 170GB during inference.

While primarily developed to improve memory efficiency, ScaleKV also delivers notable inference acceleration by reducing tensor access and transfer operations. Figure 7(b) illustrates how inference latency increases substantially with image resolution due to exponential growth in the token sequence. Our method achieves up to 1.25× speedup on a single NVIDIA H20 GPU, with performance gains becoming more pronounced as resolution increases. These results demonstrate ScaleKV's potential for deployment in resource-constrained environments and scaling VAR models to ultra-high resolutions such as 4K, which would otherwise be limited by memory bottlenecks and inference latency.

## 5 Conclusion

This work introduces ScaleKV, a novel KV cache compression framework for Visual Autoregressive modeling that effectively addresses the memory bottlenecks in high-resolution image generation. By implementing scale-aware layer budget allocation, ScaleKV enables adaptive cache management tailored to the specific demands of each layer across scales. Through extensive experimentation, our method demonstrates a superior efficiency-quality trade-off, enabling efficient deployment in resource-constrained environments and facilitating the scaling of VAR models to ultra-high resolutions.

## References


[1] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[2] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

[3] Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Baobao Chang, Junjie Hu, and Xiao Wen. Pyramidkv: Dynamic kv cache compression based on pyramidal information funneling. *arXiv preprint arXiv:2406.02069*, 2024.

[4] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11315–11325, 2022.

[5] Junsong Chen, Chongjian Ge, Enze Xie, Yue Wu, Lewei Yao, Xiaozhe Ren, Zhongdao Wang, Ping Luo, Huchuan Lu, and Zhenguo Li. PixArt-Sigma: Weak-to-strong training of diffusion transformer for 4k text-to-image generation. In *ECCV*, pages 74–91. Springer, 2024.

[6] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International conference on machine learning*, pages 1691–1703. PMLR, 2020.

[7] Zigeng Chen, Xinyin Ma, Gongfan Fang, and Xinchao Wang. Collaborative decoding makes visual auto-regressive modeling efficient. *arXiv preprint arXiv:2411.17787*, 2024.

[8] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021.

[9] Gongfan Fang, Kunjun Li, Xinyin Ma, and Xinchao Wang. Tinyfusion: Diffusion transformers learned shallow. *arXiv preprint arXiv:2412.01199*, 2024.

[10] Yuan Feng, Junlin Lv, Yukun Cao, Xike Xie, and S Kevin Zhou. Ada-kv: Optimizing kv cache eviction by adaptive budget allocation for efficient llm inference. *arXiv preprint arXiv:2407.11550*, 2024.

[11] Yu Fu, Zefan Cai, Abedelkadir Asi, Wayne Xiong, Yue Dong, and Wen Xiao. Not all heads matter: A head-level kv cache compression method with integrated retrieval and reasoning. *arXiv preprint arXiv:2410.19258*, 2024.

[12] Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model tells you what to discard: Adaptive kv cache compression for llms. *arXiv preprint arXiv:2310.01801*, 2023.

[13] Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 52132–52152. Curran Associates, Inc., 2023.

[14] Jiatao Gu, Yuyang Wang, Yizhe Zhang, Qihang Zhang, Dinghuai Zhang, Navdeep Jaitly, Josh Susskind, and Shuangfei Zhai. Dart: Denoising autoregressive transformer for scalable text-to-image generation. *arXiv preprint arXiv:2410.08159*, 2024.

[15] Hang Guo, Yawei Li, Taolin Zhang, Jiangshan Wang, Tao Dai, Shu-Tao Xia, and Luca Benini. Fastvar: Linear visual autoregressive modeling via cached token pruning. *arXiv preprint arXiv:2503.23367*, 2025.

[16] Jian Han, Jinlai Liu, Yi Jiang, Bin Yan, Yuqi Zhang, Zehuan Yuan, Bingyue Peng, and Xiaobing Liu. Infinity: Scaling bitwise autoregressive modeling for high-resolution image synthesis, 2024.

[17] Yefei He, Luoming Zhang, Weijia Wu, Jing Liu, Hong Zhou, and Bohan Zhuang. Zipcache: Accurate and efficient kv cache quantization with salient token identification. *arXiv preprint arXiv:2405.14256*, 2024.

[18] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[19] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

[20] Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. Ella: Equip diffusion models with llm for enhanced semantic alignment. *arXiv preprint arXiv:2403.05135*, 2024.

[21] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[22] Hao Kang, Qingru Zhang, Souvik Kundu, Geonhwa Jeong, Zaoxing Liu, Tushar Krishna, and Tuo Zhao. Gear: An efficient kv cache compression recipe for near-lossless generative inference of llm. *arXiv preprint arXiv:2403.05527*, 2024.

[23] Haopeng Li, Jinyue Yang, Kexin Wang, Xuerui Qiu, Yuhong Chou, Xin Li, and Guoqi Li. Scalable autoregressive image generation with mamba. *arXiv preprint arXiv:2408.12245*, 2024.

[24] Muyang Li*, Yujun Lin*, Zhekai Zhang*, Tianle Cai, Xiuyu Li, Junxian Guo, Enze Xie, Chenlin Meng, Jun-Yan Zhu, and Song Han. Svdquant: Absorbing outliers by low-rank components for 4-bit diffusion models. *arXiv preprint arXiv:2411.05007*, 2024.

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

[26] Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *arXiv preprint arXiv:2406.11838*, 2024.

[27] Xiang Li, Kai Qiu, Hao Chen, Jason Kuen, Zhe Lin, Rita Singh, and Bhiksha Raj. Controlvar: Exploring controllable visual autoregressive modeling. *arXiv preprint arXiv:2406.09750*, 2024.

[28] Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-diffusion: Quantizing diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17535–17545, 2023.

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

[30] Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation. *arXiv preprint arXiv:2404.14469*, 2024.

[31] Zongming Li, Tianheng Cheng, Shoufa Chen, Peize Sun, Haocheng Shen, Longjin Ran, Xiaoxin Chen, Wenyu Liu, and Xinggang Wang. Controlar: Controllable image generation with autoregressive models. *arXiv preprint arXiv:2410.02705*, 2024.

[32] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, pages 740–755. Springer, 2014.

[33] Akide Liu, Jing Liu, Zizheng Pan, Yefei He, Gholamreza Haffari, and Bohan Zhuang. Minicache: Kv cache compression in depth dimension for large language models. *arXiv preprint arXiv:2405.14366*, 2024.

[34] Dongyang Liu, Shitian Zhao, Le Zhuo, Weifeng Lin, Yu Qiao, Hongsheng Li, and Peng Gao. Lumina-mgpt: Illuminate flexible photorealistic text-to-image generation with multimodal generative pretraining. *arXiv preprint arXiv:2408.02657*, 2024.

[35] Zichang Liu, Aditya Desai, Fangshuo Liao, Weitao Wang, Victor Xie, Zhaozhuo Xu, Anastasios Kyrillidis, and Anshumali Shrivastava. Scissorhands: Exploiting the persistence of importance hypothesis for llm kv cache compression at test time. *Advances in Neural Information Processing Systems*, 36, 2024.

[36] Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. *arXiv preprint arXiv:2402.02750*, 2024.

[37] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023.

[38] Zhuoyan Luo, Fengyuan Shi, Yixiao Ge, Yujiu Yang, Limin Wang, and Ying Shan. Open-magvit2: An open-source project toward democratizing auto-regressive visual generation. *arXiv preprint arXiv:2409.04410*, 2024.

[39] Zhaoyang Lyu, Xudong Xu, Ceyuan Yang, Dahua Lin, and Bo Dai. Accelerating diffusion models via early stop of the diffusion process. *arXiv preprint arXiv:2205.12524*, 2022.

[40] Xiaoxiao Ma, Mohan Zhou, Tao Liang, Yalong Bai, Tiejun Zhao, Huaian Chen, and Yi Jin. Star: Scale-wise text-to-image generation via auto-regressive representations. *arXiv preprint arXiv:2406.10797*, 2024.

[41] Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free. *arXiv preprint arXiv:2312.00858*, 2023.

[42] Matanel Oren, Michael Hassid, Yossi Adi, and Roy Schwartz. Transformers are multi-state rnns. *arXiv preprint arXiv:2401.06104*, 2024.

[43] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[44] Ziran Qin, Youru Lv, Mingbao Lin, Zeren Zhang, Danping Zou, and Weiyao Lin. Head-aware kv cache compression for efficient visual autoregressive modeling. *arXiv preprint arXiv:2504.09261*, 2025.

[45] Kai Qiu, Xiang Li, Hao Chen, Jie Sun, Jinglu Wang, Zhe Lin, Marios Savvides, and Bhiksha Raj. Efficient autoregressive audio modeling via next-scale prediction. *arXiv preprint arXiv:2408.09027*, 2024.

[46] Siyu Ren and Kenny Q Zhu. On the efficacy of eviction policy for key-value constrained generative language model inference. *arXiv preprint arXiv:2402.06262*, 2024.

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

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

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

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

[51] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024.

[52] Haotian Tang, Yecheng Wu, Shang Yang, Enze Xie, Junsong Chen, Junyu Chen, Zhuoyang Zhang, Han Cai, Yao Lu, and Song Han. HART: Efficient visual generation with hybrid autoregressive transformer. *arXiv preprint arXiv:2410.10812*, 2024.

[53] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in neural information processing systems*, 37:84839–84865, 2024.

[54] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016.

[55] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[56] Zhongwei Wan, Xinjian Wu, Yu Zhang, Yi Xin, Chaofan Tao, Zhihong Zhu, Xin Wang, Siqi Luo, Jing Xiong, and Mi Zhang. D2o: Dynamic discriminative operations for efficient generative inference of large language models. *arXiv preprint arXiv:2406.13035*, 2024.

[57] Zhongwei Wan, Ziang Wu, Che Liu, Jinfa Huang, Zhihong Zhu, Peng Jin, Longyue Wang, and Li Yuan. Look-m: Look-once optimization in kv cache for efficient multimodal long-context inference. *arXiv preprint arXiv:2406.18139*, 2024.

[58] Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024.

[59] Felix Wimbauer, Bichen Wu, Edgar Schoenfeld, Xiaoliang Dai, Ji Hou, Zijian He, Artsiom Sanakoyeu, Peizhao Zhang, Sam Tsai, Jonas Kohler, et al. Cache me if you can: Accelerating diffusion models through block caching. *arXiv preprint arXiv:2312.03209*, 2023.

[60] Chengyue Wu, Xiaokang Chen, Zhiyu Wu, Yiyang Ma, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, Chong Ruan, et al. Janus: Decoupling visual encoding for unified multimodal understanding and generation. *arXiv preprint arXiv:2410.13848*, 2024.

[61] Yecheng Wu, Zhuoyang Zhang, Junyu Chen, Haotian Tang, Dacheng Li, Yunhao Fang, Ligeng Zhu, Enze Xie, Hongxu Yin, Li Yi, et al. VILA-U: A unified foundation model integrating visual understanding and generation. *arXiv preprint arXiv:2409.04429*, 2024.

[62] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv*, 2023.

[63] Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. Show-o: One single transformer to unify multimodal understanding and generation. *arXiv preprint arXiv:2408.12528*, 2024.

[64] Rui Xie, Tianchen Zhao, Zhihang Yuan, Rui Wan, Wenxi Gao, Zhenhua Zhu, Xuefei Ning, and Yu Wang. Litevar: Compressing visual autoregressive modelling with efficient attention and quantization. *arXiv preprint arXiv:2411.17178*, 2024.

[65] Dongjie Yang, XiaoDong Han, Yan Gao, Yao Hu, Shilin Zhang, and Hai Zhao. Pyramidinfer: Pyramid kv cache compression for high-throughput llm inference. *arXiv preprint arXiv:2405.12532*, 2024.

[66] Xingyi Yang and Xinchao Wang. Hash3d: Training-free acceleration for 3d generation. *arXiv preprint arXiv:2404.06091*, 2024.

[67] Xingyi Yang, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Diffusion probabilistic model made slim. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22552–22562, 2023.

[68] Ziyu Yao, Jialin Li, Yifeng Zhou, Yong Liu, Xi Jiang, Chengjie Wang, Feng Zheng, Yuexian Zou, and Lei Li. Car: Controllable autoregressive modeling for visual generation. *arXiv preprint arXiv:2410.04671*, 2024.

[69] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. *arXiv preprint arXiv:2311.18828*, 2023.

[70] Yuxuan Yue, Zhihang Yuan, Haojie Duanmu, Sifan Zhou, Jianlong Wu, and Liqiang Nie. Wkvquant: Quantizing weight and key/value cache for large language models gains more. *arXiv preprint arXiv:2402.12065*, 2024.

[71] Zongsheng Yue, Jianyi Wang, and Chen Change Loy. Resshift: Efficient diffusion model for image super-resolution by residual shifting. *Advances in Neural Information Processing Systems*, 36, 2024.

[72] Dingkun Zhang, Sijia Li, Chen Chen, Qingsong Xie, and Haonan Lu. Laptop-diff: Layer pruning and normalized distillation for compressing diffusion models. *arXiv preprint arXiv:2404.11098*, 2024.

[73] Jinzhi Zhang, Feng Xiong, and Mu Xu. G3pt: Unleash the power of autoregressive modeling in 3d generation via cross-scale querying transformer. *arXiv preprint arXiv:2409.06322*, 2024.

[74] Qian Zhang, Xiangzi Dai, Ninghua Yang, Xiang An, Ziyong Feng, and Xingyu Ren. Var-clip: Text-to-image generator with visual auto-regressive modeling. *arXiv preprint arXiv:2408.01181*, 2024.

[75] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[76] Wentian Zhang, Haozhe Liu, Jinheng Xie, Francesco Faccio, Mike Zheng Shou, and Jürgen Schmidhuber. Cross-attention makes inference cumbersome in text-to-image diffusion models. *arXiv preprint arXiv:2404.02747*, 2024.

[77] Yuxin Zhang, Yuxuan Du, Gen Luo, Yunshan Zhong, Zhenyu Zhang, Shiwei Liu, and Rongrong Ji. Cam: Cache merging for memory-efficient llms inference. In *Forty-first International Conference on Machine Learning*, 2024.

[78] Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710, 2023.

[79] Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-time video generation with pyramid attention broadcast. *arXiv preprint arXiv:2408.12588*, 2024.

[80] Yang Zhao, Yanwu Xu, Zhisheng Xiao, and Tingbo Hou. Mobilediffusion: Subsecond text-to-image generation on mobile devices. *arXiv preprint arXiv:2311.16567*, 2023.

[81] Chunting Zhou, Lili Yu, Arun Babu, Kushal Tirumala, Michihiro Yasunaga, Leonid Shamis, Jacob Kahn, Xuezhe Ma, Luke Zettlemoyer, and Omer Levy. Transfusion: Predict the next token and diffuse images with one multi-modal model. *arXiv preprint arXiv:2408.11039*, 2024.

## NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The main claims made in our abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss the limitation of our work in the Appendix.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide a detailed description of our method along with extensive experimental results.

   Guidelines:

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

5. **Open access to data and code**

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

Answer: [Yes]

Justification: We offer the full code along with relevant instructions.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We provide all the details about the experiment in our paper.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We provide the details about initialization and dataset split.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide the details about the computation resources we used in the experiments.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We strictly adhere to the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We discuss it in Appendix.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: The creators or original owners of assets (e.g., code, data, models) used in the paper are properly credited, and the license and term of use are explicitly mentioned and properly adhered to.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: New assets introduced in the paper are well documented, and the documentation is provided alongside the assets.

    Guidelines:

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

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

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

    Answer: [NA]

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

    Guidelines:

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

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

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

    Answer: [NA]

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

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [NA]

    Justification: We do not use LLMs as core methods in this research.

    Guidelines:

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

## Technical Appendices and Supplementary Material

In this document, we provide supplementary material that we could not fit into the main manuscript due to the page limit. It includes detailed explanations, visualization results, and quantitative experiments.

## A Robustness to Calibration Data

To evaluate the stability of our drafter/refiner identification method across varying calibration set sizes, we conducted an ablation study using prompts sampled from the LAION-Art dataset. We systematically evaluated ScaleKV's performance using calibration sets ranging from a single prompt to 128 prompts, measuring the resulting FID scores on the MS-COCO validation set. As demonstrated in Figure 8, the FID score remains stable at 2.53 with zero standard deviation across the entire range of calibration set sizes. This exceptional consistency indicates that the attention patterns distinguishing drafters from refiners represent fundamental architectural properties of VAR models rather than dataset-specific characteristics. The immediate convergence with even a single calibration sample demonstrates that our Attention Selectivity Index effectively captures the intrinsic scale-dependent attention behaviors—dispersed attention for drafters and concentrated attention for refiners—without requiring extensive statistical sampling. These findings validate that ScaleKV's layer categorization is both theoretically sound and practically robust.

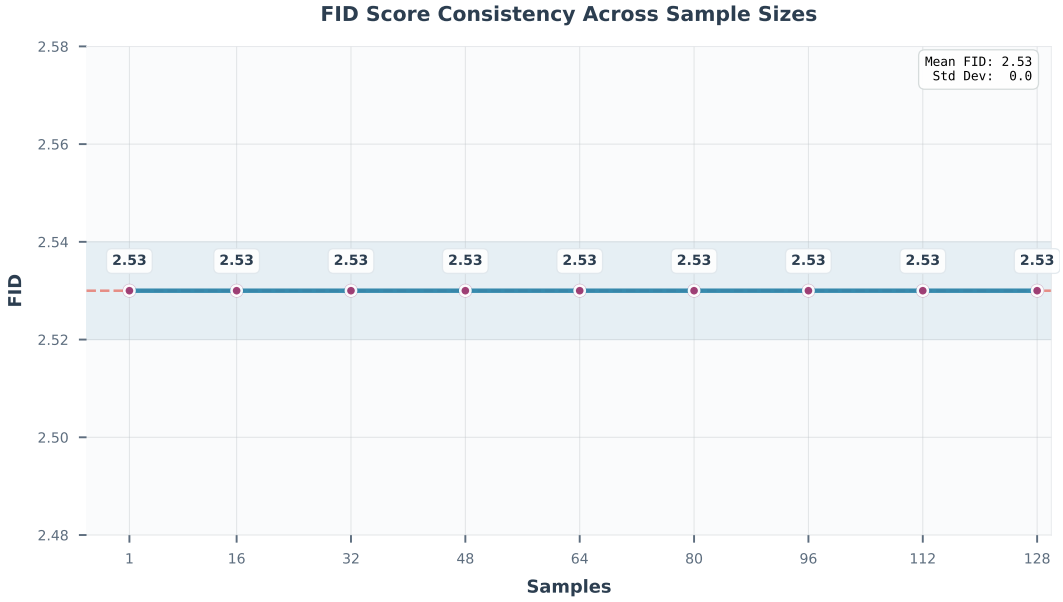



Figure 8: FID score consistency across calibration set sizes. ScaleKV maintains stable performance (FID = 2.53, $\sigma$ = 0.0) from 1 to 128 calibration samples, demonstrating the robustness of our drafter/refiner identification method.

## B Attention Map Visualization

Figures 9 and 10 present representative attention maps from transformer layers identified as drafters and refiners at scale 7, providing empirical validation for our layer categorization framework. The drafter layer (Block 23) exhibits distinctly dispersed attention patterns across multiple attention heads, with weights distributed broadly across the spatial dimensions to capture global contextual information from preceding scales. This dispersed mechanism enables comprehensive integration of hierarchical features, justifying the larger cache allocation for such layers. In contrast, the refiner layer (Block 29) demonstrates highly concentrated attention patterns, with attention heads focusing predominantly on localized spatial regions within the current token map. These contrasting attention behaviors provide strong empirical evidence for our differentiated cache management strategy: drafters require

substantial cache capacity to maintain broad contextual access while refiners can operate effectively with significantly reduced cache allocation due to their localized processing nature.

## C Additional Qualitative Results

Figures 11 and 12 present comprehensive galleries of images generated by ScaleKV-compressed Infinity-8B and Infinity-2B models, respectively, demonstrating the practical effectiveness of our compression framework across diverse visual content. These compressed models operate with merely 10% of the original KV cache memory requirement, yet maintain exceptional generation quality across various image categories including natural scenes, objects, portraits, and artistic compositions. These results demonstrate that ScaleKV achieves substantial memory reduction without compromising the generative capabilities of VAR models, making high-quality image synthesis feasible in memory-constrained deployment scenarios.

## D Limitations

In this analysis, we critically examine the constraints of our methodology.

First, while ScaleKV demonstrates robust compression performance across models of varying capacities, evaluation on larger VAR models would provide additional insights into our method's scalability. Due to the limited availability of large-scale models, our evaluation was restricted to Infinity-8B, currently the largest available VAR model. Testing on models with greater capacity, such as those with 20B parameters, would enable more comprehensive assessment of ScaleKV's scalability. Second, ScaleKV functions as a post-training KV cache compression solution that relies on pre-trained VAR models and mirrors the original model's outputs. Therefore, if the baseline quality of the original VAR models is unsatisfactory, achieving high-quality results with our method could be challenging.

## E Societal impacts

This work introduces a new KV cache compression framework for VAR models that addresses critical memory bottlenecks in high-resolution image generation. By reducing memory requirements to 10% of the original capacity while maintaining generation quality, our method enhances the accessibility of advanced image synthesis technologies and carries several important societal implications. First, it democratizes access to high-quality image generation by enabling deployment on consumer-grade hardware and edge devices, thereby benefiting creative industries and educational institutions that previously lacked the computational resources for such applications. Second, the reduced memory footprint results in lower energy consumption during inference, contributing to more sustainable AI deployment practices. Third, by enabling ultra-high resolution generation at scales up to 4K, our framework creates new opportunities for professional content creation, medical imaging, and scientific visualization applications.

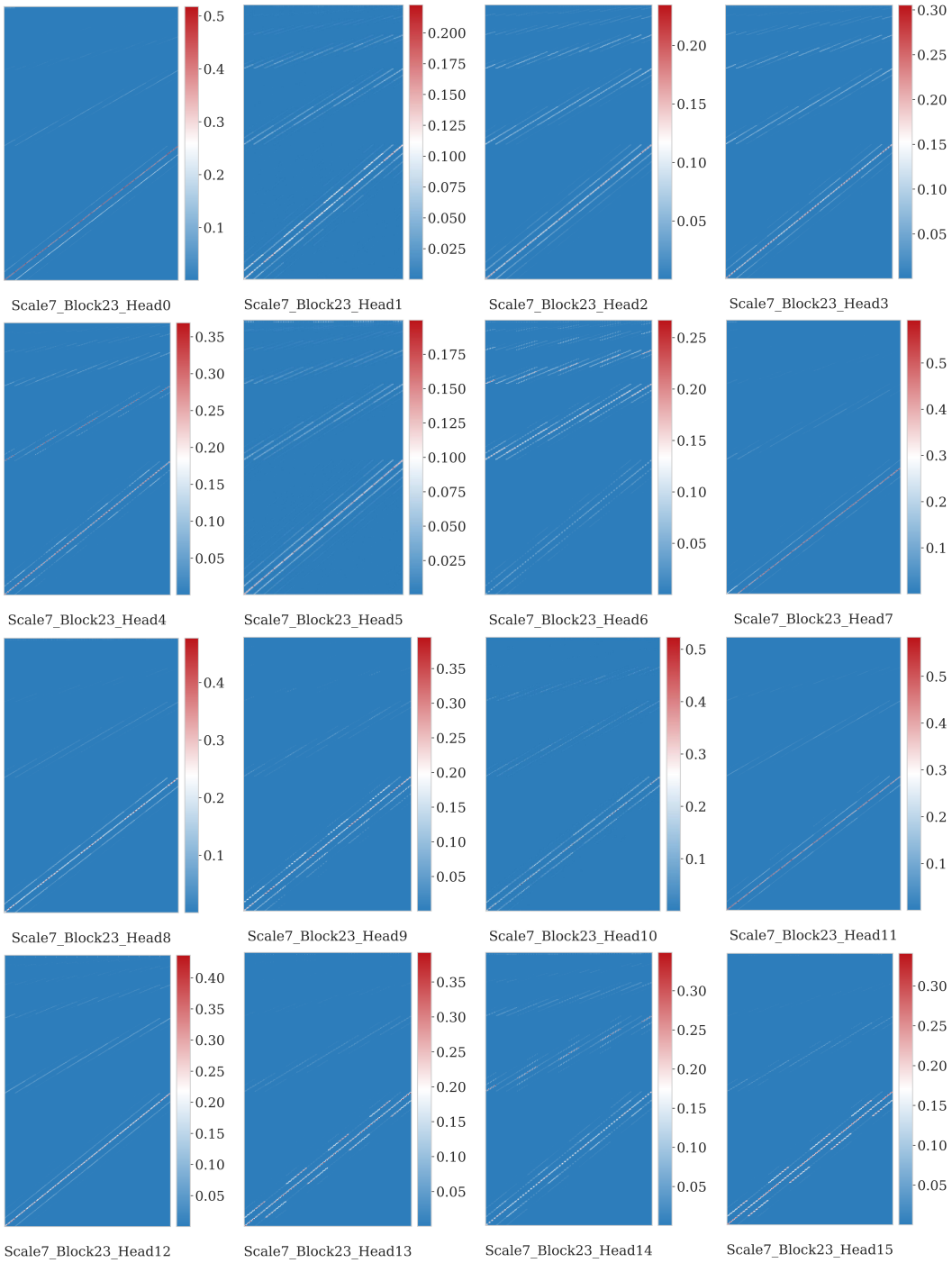



Figure 9: Visualization of Drafter Layer Attention Maps.

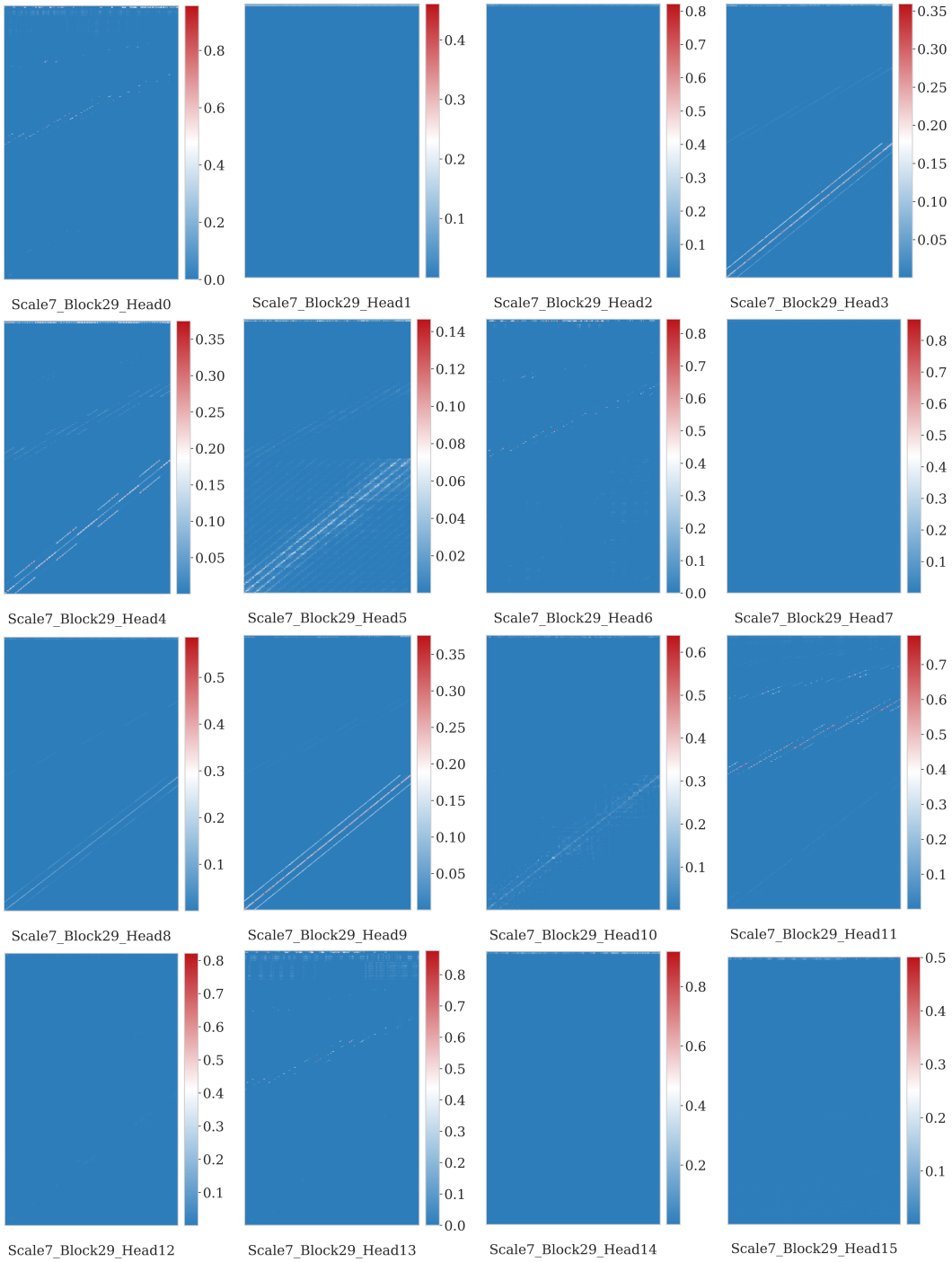



Figure 10: Visualization of Refiner Layer Attention Maps.


Figure 11: Generated Images from ScaleKV-Compressed Infinity-8B.

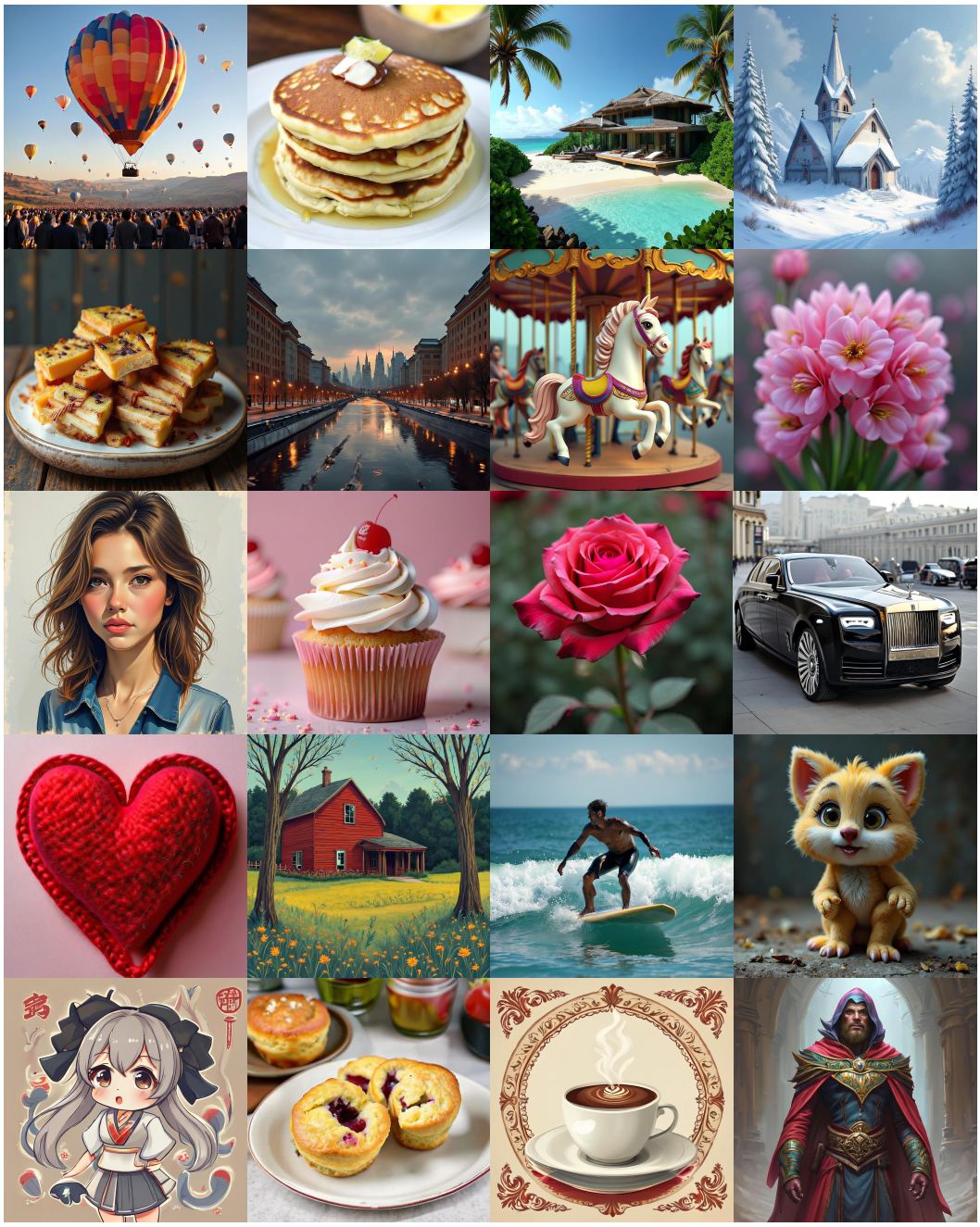


Figure 12: Generated Images from ScaleKV-Compressed Infinity-2B.