Head-Aware KV Cache Compression for Efficient Visual Autoregressive Modeling

Ziran Qin[1], Youru Lv[1], Mingbao Lin[2*], Hang Guo[3], Zeren Zhang[4], Danping Zou[1], Weiyao Lin[1*]

[1] Shanghai Jiao Tong University, China
[2] Rakuten, Singapore
[3] Tsinghua University, China
[4] Peking University, China
{qinziran,mishall0914,dpzou,wylin}@sjtu.edu.cn, linmb001@outlook.com, cshguo@gmail.com, eric_zhang@stu.pku.edu.cn

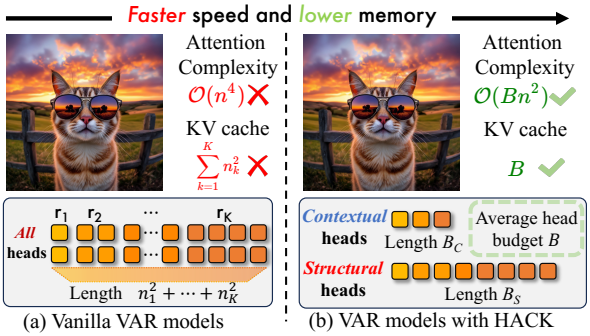
Abstract

Visual Autoregressive (VAR) models adopt a next-scale prediction paradigm, offering high-quality content generation with substantially fewer decoding steps. However, existing VAR models suffer from significant attention complexity and severe memory overhead due to the accumulation of key-value (KV) caches across scales. In this paper, we tackle this challenge by introducing KV cache compression into the next-scale generation paradigm. We begin with a crucial observation: attention heads in VAR models can be divided into two functionally distinct categories: Contextual Heads focus on maintaining semantic consistency, while Structural Heads are responsible for preserving spatial coherence. This structural divergence causes existing one-size-fits-all compression methods to perform poorly on VAR models. To address this, we propose HACK, a training-free Head-Aware KV cache Compression frameworK. HACK utilizes an offline classification scheme to separate head types, enabling it to apply pattern-specific compression strategies with asymmetric cache budgets for each category. By doing so, HACK effectively constrains the average KV cache length within a fixed budget B, reducing the theoretical attention complexity from $\mathcal{O}(n^4)$ to $\mathcal{O}(Bn^2)$. Extensive experiments on multiple VAR models across text-to-image and class-conditional tasks validate the effectiveness and generalizability of HACK. It achieves up to 70% KV cache compression without degrading output quality, resulting in memory savings and faster inference. For example, HACK provides a $1.75\times$ memory reduction and a $1.57\times$ speedup on Infinity-8B.

Code — https://github.com/Zr2223/HACK

Introduction

Autoregressive (AR) models (He et al. 2024a; Sun et al. 2024; Li et al. 2024b) have demonstrated promising performance in visual generation, rivaling diffusion models (Ho, Jain, and Abbeel 2020; Podell et al. 2023; Peebles and Xie 2023; Esser et al. 2024; Chen et al. 2024b) in both quality and flexibility. However, conventional AR models follow a "next-token" prediction paradigm, requiring long decoding steps that suffer from substantial inference latency. Recently, Visual AutoRegressive (VAR) modeling (Tian et al. 2024; Han et al. 2024) has emerged, replacing the "next-token" prediction with a "next-scale" prediction paradigm. By enabling the parallel generation of multiple tokens within a single step, VAR models generate high-quality content in a multi-scale, coarse-to-fine manner within a few steps, effectively balancing generation quality and efficiency.



(a) Vanilla VAR models (b) VAR models with HACK

Figure 1: (a) Vanilla VAR models cache all KV pairs across different scales. (b) HACK only preserves proper KV pairs selected by head-aware strategies, effectively hacking down both attention complexity and KV cache length.

However, VAR models suffer from memory and computational bottlenecks during inference. As shown in Figure 1(a), vanilla VAR indiscriminately caches all key-value (KV) pairs from all previous scales, leading to high attention complexity $\mathcal{O}(n^4)$ and even greater KV cache accumulation than conventional AR models. This problem is severe for large-scale content. For example, generating 1024×1024 images with Infinity (Han et al. 2024) requires processing over 10k tokens across multiple scales, imposing substantial burdens on both attention computation and KV cache storage. This poses a major challenge for practical deployment.

Inspired by the KV cache optimization in large language models (LLMs) (Achiam et al. 2023; Anthropic 2024; Dubey et al. 2024; Liu et al. 2024a), we address these challenges in VAR by compressing unimportant KV pairs before its attention computation. However, existing LLMs-oriented KV compression methods (Zhang et al. 2024b; Li et al. 2024c; Qin et al. 2025; Wan et al. 2024b) fail to generalize to VAR models due to fundamental differences in their at-

*Corresponding authors.

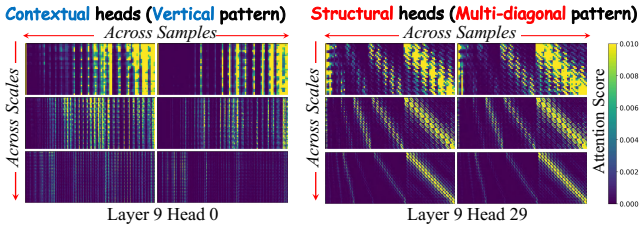


Figure 2: **Attention Patterns of Contextual and Structural Heads.** Both Contextual and Structural heads exhibit consistent vertical and multi-diagonal patterns, respectively, across different samples and scales.

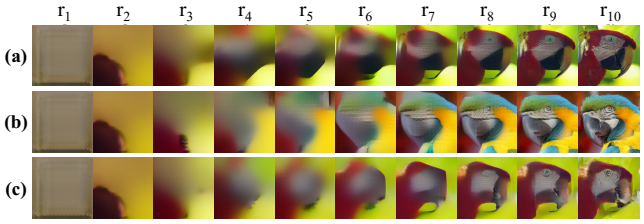


Figure 3: **Impact of selective head masking** (10% of total heads for each type). Compared with the original generation (a), masking *contextual* heads (b) leads to semantic divergence while maintaining high visual fidelity; in contrast, masking *structural* heads (c) preserves global content direction but results in severe structural degradation.

tention mechanisms. In particular, we observe that attention heads in VAR (shown in Figure 2) fall into two categories with distinct attention patterns and functional roles. ***Contextual heads*** focus on a small set of key tokens to maintain global semantic consistency, forming vertical attention patterns. ***Structural heads*** attend to spatially adjacent tokens across scales through multi-diagonal attention patterns, preserving the geometric structure of contents. We further validate their functional roles by selectively masking each type during generation. In Figure 3, masking contextual heads causes the generated content to semantically diverge, while still maintaining structural integrity and high visual fidelity, whereas masking structural heads preserves the global content direction but causes severe spatial distortions. Given the distinct attention characteristics and functional roles of them, existing one-size-fits-all compression strategies are ill-suited for VAR models. This calls for a head-aware KV cache compression approach considering the distinct properties and sensitivities of contextual and structural heads.

We propose **HACK**, a novel, training-free, head-aware KV cache compression framework for efficient VAR models. HACK first exploits the inherent differences between contextual and structural heads through a robust offline classification based on attention variance. Then, as shown in Figure 1(b), HACK allocates asymmetric KV budgets and applies pattern-specific compression strategies tailored to each head type. This reduces the attention complexity from $\mathcal{O}(n^4)$ to $\mathcal{O}(Bn^2)$, where B is the average cache length per head. Finally, HACK achieves substantial memory and latency reductions without degrading generation quality. To the best of our knowledge, HACK is the first study to address KV cache compression in VAR models.

Our contributions include: (1) We provide an in-depth analysis that identifies and characterizes contextual and structural heads in VAR models, revealing their consistent functional roles and attention patterns, which offers crucial insights for optimizing these models. (2) We introduce HACK, a head-aware, training-free framework for VAR models that features asymmetric cache budget allocation and pattern-specific KV cache compression strategies suited for both contextual and structural heads. (3) We prove HACK's efficacy on six different VAR models across multiple tasks, showing substantial memory and latency reductions while maintaining or even improving generation quality.

Related Work

Autoregressive Visual Generation. AR models (Achiam et al. 2023; Anthropic 2024; Dubey et al. 2024; Liu et al. 2024a), originally successful in LLMs, have extended into the visual domain. Traditional visual AR approaches (He et al. 2024a; Sun et al. 2024; Li et al. 2024b) rely on next-token prediction, where images are first quantized into discrete tokens (*e.g.*, VQVAE (Van Den Oord, Vinyals et al. 2017), VQGAN (Esser, Rombach, and Ommer 2021)) and then decoded autoregressively. While effective, they incur high computation and quantization errors, resulting in lower efficiency. Recent VAR models (Tian et al. 2024; Han et al. 2024; Tang et al. 2024; Chen et al. 2024a; Ren et al. 2024) adopt a next-scale prediction paradigm, enabling multi-scale parallel decoding that improves both quality and speed. They have shown strong performance across text-to-image (Han et al. 2024; Voronov et al. 2024), 3D content (Chen et al. 2025; Gao et al. 2025), and unified vision-language tasks (Zhuang et al. 2025). However, their hierarchical design leads to growing attention complexity and KV cache accumulation across scales, resulting in both memory and compute bottlenecks. We address these challenges via VAR-specific KV cache compression that reduces resource cost while preserving generation quality.

KV Cache Compression. Existing efforts mainly focus on quantization, eviction, and merging to alleviate KV cache overhead in LLMs. Quantization (Liu et al. 2024d; Yue et al. 2024; Kang et al. 2024; He et al. 2024b) reduces cache size by lowering bit precision. Eviction methods discard less critical tokens based on attention metrics (Zhang et al. 2024b; Liu et al. 2024c; Oren et al. 2024; Ren and Zhu 2024; Li et al. 2024c), optimizing cache allocation under fixed budgets (Ge et al. 2023; Yang et al. 2024; Feng et al. 2024; Fu et al. 2024; Qin et al. 2025). Merging strategies (Zhang et al. 2024a; Liu et al. 2024b; Wan et al. 2024a,b) alleviate memory constraints by combining redundant KV pairs while preserving context. However, these methods fall short of VAR models, whose hierarchical attention demands careful preservation of structural and contextual dependencies. We introduce head-aware compression tailored to VAR's heterogeneous attention patterns, enabling substantial memory and compute reductions without degrading visual quality.

Preliminaries

Visual AutoRegressive Modeling. VAR extends autoregressive modeling from "next-token" to "next-scale" prediction. Given an image feature map $\mathbf{f} \in \mathbb{R}^{h\times w\times c}$, VAR quantizes it into K multi-scale token maps $\mathbf{R} = (\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_K)$. Each map $\mathbf{r}_k \in [V]^{h_k\times w_k}$ contains $h_k \times w_k$ tokens and the resolutions $\{(h_k, w_k)\}_{k=1}^K$ are predefined such that $h_k w_k = a^{2(k-1)}$, where $a > 1$ is a constant scaling factor. The autoregressive likelihood is:

$$p(\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_K) = \prod_{k=1}^{K} p(\mathbf{r}_k|\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_{k-1}), \quad (1)$$

where the generation of each scale k is conditioned on the tokens $\{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_{k-1}\}$ from all preceding scales, which serve as a contextual prefix. During inference, the entire content can be synthesized coarse-to-fine in just K steps, with all tokens for each scale predicted in parallel.

KV Cache in VAR. Akin to AR models, VAR improves efficiency by caching KV pairs across generation steps. In a multi-head attention module, each head h has weight matrices $\mathbf{W}_Q^{(h)}, \mathbf{W}_K^{(h)}, \mathbf{W}_V^{(h)} \in \mathbb{R}^{D\times D_h}$, projecting input tokens $\mathbf{X}_k \in \mathbb{R}^{N_k\times D}$ at step k into queries, keys, and values:

$$\mathbf{Q}_k^{(h)} = \mathbf{X}_k\mathbf{W}_Q^{(h)},\ \mathbf{K}_k^{(h)} = \mathbf{X}_k\mathbf{W}_K^{(h)},\ \mathbf{V}_k^{(h)} = \mathbf{X}_k\mathbf{W}_V^{(h)}, \quad (2)$$

where $N_k = h_k w_k$ denotes the number of tokens at scale k. The KV caches are updated independently per head:

$$\mathbf{K}^{(h)} = \text{Concat}(\mathbf{K}^{(h)}, \mathbf{K}_k^{(h)}), \quad \mathbf{V}^{(h)} = \text{Concat}(\mathbf{V}^{(h)}, \mathbf{V}_k^{(h)}), \quad (3)$$

with the cache length at step k, denoted as $T_k = \sum_{i=1}^{k} w_i h_i$, growing cumulatively. Attention is then computed as:

$$\mathbf{A}^{(h)} = \text{Softmax}(\mathbf{Q}^{(h)}{\mathbf{K}^{(h)}}^T), \quad \mathbf{O}^{(h)} = \mathbf{A}^{(h)}\mathbf{V}^{(h)}. \quad (4)$$

While caching avoids redundant computations across scales, the accumulation of tokens leads to escalating memory consumption. Also, the attention cost increases as the sequence length grows, causing complexity of $\mathcal{O}(n^4)$, where $n = a^{k-1}$. These dual challenges make efficient KV cache management a vital prerequisite for scalable inference.

KV Cache Compression for VAR. To address the above memory and attention compute challenges, we introduce a KV cache compression strategy for VAR models. Without compression, the total cache size can become prohibitively large. We quantify this as $B_{\text{total}} = T_K \cdot H \cdot L$, where T_K is the total number of generated tokens, H and L are the number of attention heads and layers, respectively. Our strategy is to enforce a fixed per-head budget B. Whenever the cache length T_k exceeds B at any generation step k, compression is applied to the KV cache before the attention computation:

$$\hat{\mathbf{K}}^{(h)}, \hat{\mathbf{V}}^{(h)} = f(\mathbf{K}^{(h)}, \mathbf{V}^{(h)}, B), \quad (5)$$

where $\hat{\mathbf{K}}^{(h)}, \hat{\mathbf{V}}^{(h)} \in \mathbb{R}^{B\times D_h}$ are the compressed KV pairs, and $f(\cdot)$ is any compression strategy.

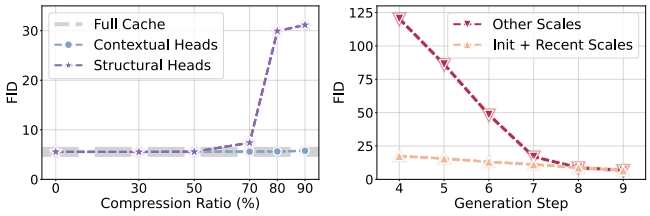


Figure 4: **Empirical analysis on VAR-d30.** (Left) Compression sensitivity of contextual vs. structural heads. (Right) Comparison of two scale-preserving strategies. Experiments are conducted on 8K ImageNet samples.

Methodology

Motivation

To guide the design of KV compression for VAR models, we first conduct a series of empirical analyses using VAR-d30. Our observations are summarized as follows:

Observation 1: Stability of attention heads across samples and scales. As shown in Figure 2, despite variations in input samples or generation scales, both contextual and structural heads consistently exhibit their respective vertical and multi-diagonal attention patterns. This consistency suggests that attention heads in VAR models play stable, functional roles that are intrinsic to the model and can thus be reliably classified. We provide more attention visualization results in the *supplementary material*.

Observation 2: Divergent compression sensitivity across head types. The two head types show markedly different sensitivities to compression. Contextual heads, which focus attention on key tokens to summarize global semantics, are resistant to pruning. In contrast, structural heads, which maintain geometric structure on each scale, are highly sensitive, as excessive compression can disrupt critical spatial information. As shown in Figure 4 (left), contextual heads maintain high image quality even with 90% compression ratio, while the performance of structural heads degrades significantly when compression exceeds 50%. This difference in compression sensitivity motivates our strategy of allocating asymmetric budgets to use resources more effectively.

Observation 3: Initial and recent scales are more important in generation. Drawing inspiration from LLMs, where initial and recent tokens receive higher attention and are crucial for coherence (Xiao et al. 2023), we hypothesized that initial and recent scales play a similar role in VAR models. To verify this, we compared two token preservation strategies: (1) retaining tokens only from the initial two and the most recent scales, versus (2) retaining an equal number of tokens from intermediate scales. As shown in Figure 4 (right), strategy (1) yields dramatically better generation quality, with FID scores up to $10\times$ lower in early generation steps. This confirms the greater importance of tokens from the initial and recent scales in VAR's generation.

Offline Head Classification via Attention Variance

Motivated by the distinct and consistent functional behaviors of contextual and structural heads, we propose an of-

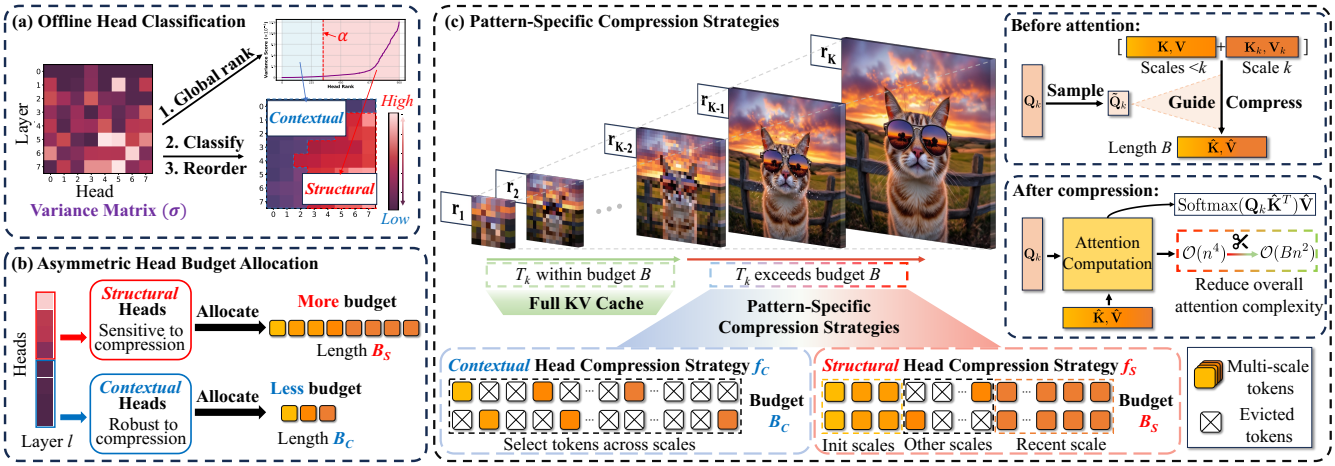


Figure 5: **Overview of our proposed HACK framework.** It consists of (a) offline head classification via attention variance, (b) asymmetric head budget allocation based on compression sensitivity, and (c) pattern-specific KV cache compression strategies.

fline classification method (illustrated in Figure 5(a)) to distinguish them before deployment. As we analyzed, contextual heads exhibit stable attention patterns across queries, selectively focusing on or disregarding semantic cues, thereby yielding low column-wise variance. In contrast, structural heads attend in a position-sensitive manner, dynamically shifting attention based on spatial configurations, which results in higher variance across queries. Therefore, we capture their divergent behaviors by measuring the attention variance. Specifically, we analyze the attention matrix $\mathbf{A}_K^{(l,h)} \in \mathbb{R}^{N_K \times T_K}$ at the final generation step K, which encapsulates interactions from all prior scales. For each head h in layer l, we compute the sum of column-wise variances:

$$\sigma_{l,h} = \sum_{j=1}^{T_K} \mathrm{Var}(\mathbf{A}_K^{(l,h)}[:, j]). \tag{6}$$

Averaging these values over a small validation set yields a variance matrix $\boldsymbol{\sigma} \in \mathbb{R}^{L \times H}$. We then globally rank all heads according to their variance scores. Notably, the resulting distribution exhibits a long-tailed pattern, with a clear boundary separating low-variance and high-variance regions, corresponding to contextual and structural heads, respectively. To assign head types, we define a global contextual head ratio α, which can be tuned based on the empirical variance distribution to suit different models.

Following classification, we obtain per-layer index sets $\mathbb{H}_C$ and $\mathbb{H}_S$ for contextual and structural heads, respectively, satisfying $\mathbb{H}_C \cup \mathbb{H}_S = \{1, \ldots, H\}$ and $\mathbb{H}_C \cap \mathbb{H}_S = \emptyset$. For clarity, we annotate head-specific variables with superscripts (C) and (S); for instance, $\mathbf{K}^{(C)}$ and $\mathbf{V}^{(C)}$ denote the key and value states of contextual heads, while $\mathbf{K}^{(S)}$ and $\mathbf{V}^{(S)}$ correspond to those of structural heads. To facilitate efficient inference, we conceptually reorder the heads within each layer by grouping them according to type.

Asymmetric Head Budget Allocation

Given an average target budget B per head and the contextual head ratio $0 < \alpha < 1$, we allocate memory asymmetrically to contextual and structural heads:

$$B = \alpha B_C + (1 - \alpha) B_S, \tag{7}$$

where B_C and B_S denote the cache budget assigned to each contextual and structural head, respectively. In practice, we assign substantially smaller budgets to contextual heads (*i.e.*, $B_C \ll B_S$), preserving more capacity for structural heads due to their higher sensitivity to compression. It is worth noting that although α defines the global proportion of contextual heads, the actual contextual–structural head ratio varies across layers. This naturally induces a layer-adaptive compression effect, where cache budgets are allocated according to each layer's specific head composition.

When the cache length T_k for a head group exceeds its assigned budget at any step k, we invoke dedicated compression strategies *prior to attention computation* to limit the size of stored KV pairs. This ensures that memory and compute costs remain bounded before executing the attention module. The compressed KV pairs are computed as:

$$\hat{\mathbf{K}}^{(p)}, \hat{\mathbf{V}}^{(p)} = f_p\left(\{\mathbf{K}^{(h)} | h \in \mathbb{H}_p\}, \{\mathbf{V}^{(h)} | h \in \mathbb{H}_p\}, B_p\right), \tag{8}$$

where $p \in \{C, S\}$ and f_p denotes the specific compression strategy for contextual or structural heads.

Pattern-Specific Compression Strategies

We design compression strategies, as shown in Figure 5(c), that are tailored to the distinct attention patterns of contextual and structural heads, enabling fine-grained and head-aware KV cache compression.

Importance Estimation via Efficient Subset Attention. To make compression efficient, we propose to approximate full attention behavior by observing a small subset of queries. For each head type $p \in \{C, S\}$, we uniformly sample a small subset of queries $\tilde{\mathbf{Q}}_k^{(p)} \in \mathbb{R}^{N_{\text{obs}} \times D_h}$ from full

$\mathbf{Q}_k^{(p)} \in \mathbb{R}^{N_k \times D_h}$ and compute attention scores over uncompressed key matrix $\mathbf{K}^{(p)} \in \mathbb{R}^{T_k \times D_h}$:

$$\tilde{\mathbf{A}}_k^{(p)} = \text{Softmax}(\tilde{\mathbf{Q}}_k^{(p)}(\mathbf{K}^{(p)})^T). \tag{9}$$

These approximated scores act as a computationally cheap proxy for full attention, enabling token selection and compression for both contextual and structural heads without incurring the cost of full attention computation.

Contextual Head Compression Strategy f_C. Contextual heads typically focus on a small number of semantically salient tokens, resulting in vertically concentrated attention patterns. Based on this property, we directly select KV pairs $\{\hat{\mathbf{K}}^{(C)}, \hat{\mathbf{V}}^{(C)}\}$ with the highest cumulative attention scores:

$$\begin{aligned} \mathbf{I}^{(C)} &= \text{TOPK}\left(\sum_{i=1}^{N_{\text{obs}}} \tilde{\mathbf{A}}_k^{(C)}[i,:], B_C\right), \\ \hat{\mathbf{K}}^{(C)} &= \mathbf{K}^{(C)}[\mathbf{I}^{(C)},:], \quad \hat{\mathbf{V}}^{(C)} = \mathbf{V}^{(C)}[\mathbf{I}^{(C)},:], \end{aligned} \tag{10}$$

where TOPK selects the indices of the top-K tokens with highest scores. To mitigate potential semantic loss from aggressive pruning at the final generation step, we apply a merging operation following LOOK-M (Wan et al. 2024b). Discarded KV tokens in the last generation step are softly integrated into their nearest retained counterparts through similarity-weighted averaging. Merging is performed only once at the final scale to balance efficiency and performance.

Structural Head Compression Strategy f_S. Unlike contextual heads, structural heads exhibit position- and scale-dependent patterns, making them sensitive to spatial structures and positional continuity. To preserve structural coherence, our scale-aware compression strategy prioritizes tokens from the crucial initial and recent scales. Specifically, we retain N_{init} tokens from initial generation stages and N_k tokens from the latest scale, using the remaining budget M to select intermediate-scale tokens with the highest cumulative attention scores via subset attention. The conserved KV caches $\{\hat{\mathbf{K}}^{(S)}, \hat{\mathbf{V}}^{(S)}\}$ in structural heads are given by:

$$\begin{aligned} \mathbf{I}^{(S)} &= \text{TOPK}\left(\sum_{i=1}^{N_{\text{obs}}} \tilde{\mathbf{A}}_k^{(S)}[i,:],\, M\right), \quad M = B_S - N_{\text{init}} - N_k, \\ \hat{\mathbf{K}}^{(S)} &= \text{Concat}\left(\mathbf{K}^{(S)}[:N_{\text{init}},:],\, \mathbf{K}^{(S)}[\mathbf{I}^{(S)},:],\, \mathbf{K}^{(S)}[-N_k:,:]\right), \\ \hat{\mathbf{V}}^{(S)} &= \text{Concat}\left(\mathbf{V}^{(S)}[:N_{\text{init}},:],\, \mathbf{V}^{(S)}[\mathbf{I}^{(S)},:],\, \mathbf{V}^{(S)}[-N_k:,:]\right). \end{aligned} \tag{11}$$

This preserves long-range dependencies and recent local context, while adaptively selecting intermediate tokens to maintain spatial continuity with minimal redundancy.

Complexity Analysis

Compared to vanilla VAR, which incurs a total attention complexity of $\mathcal{O}(n^4)$ due to cumulative KV cache growth (see *supplementary material* for derivation), HACK reduces this cost to $\mathcal{O}(Bn^2)$ by enforcing an average per-head cache budget B. At generation step k, the attention complexity is upper-bounded by $N_k \cdot \min(T_k, B)$, where N_k is the number of tokens at step k and T_k is the cumulative cache length up to step k. Summing over all scales yields:

$$\sum_{k=1}^{K} N_k \cdot \min(T_k, B) \le B \sum_{k=1}^{K} a^{2(k-1)} \sim \mathcal{O}(Bn^2). \tag{12}$$

The additional overhead in HACK arises from KV cache compression, triggered only when $T_k > B$. Let k_s denote the first step where compression is applied. The main cost comes from subset attention, where $N_{\text{obs}} \ll N_k$ sampled queries estimate token importance:

$$\sum_{k=k_s}^{K} N_{\text{obs}} \cdot (B + N_k) = N_{\text{obs}} \sum_{k=k_s}^{K} (B + a^{2(k-1)}) \sim \mathcal{O}(N_{\text{obs}} n^2). \tag{13}$$

Experimentation

Experiment Settings

Evaluation Models. We evaluate multiple next-scale generation models across diverse generative tasks: Infinity-2B, Infinity-8B (Han et al. 2024) and Hart (Tang et al. 2024) for text-to-image generation; VAR-d30 (Tian et al. 2024) for class-conditional generation.

Evaluation Benchmarks and Metrics. For text-to-image generation, we adopt four benchmarks: GenEval (Ghosh, Hajishirzi, and Schmidt 2023) for assessing high-level semantic alignment; ImageReward (IR) (Xu et al. 2023), HPSv2.1 (HPS) (Wu et al. 2023), and MJHQ30k (evaluated using FID and CLIP) (Li et al. 2024a) for perceptual quality. For class-conditional generation, we report FID, Inception Score (IS), Precision, and Recall evaluated on ImageNet-1K (Deng et al. 2009) using 50 samples per class.

Baseline Methods. We compare against two categories of KV cache compression methods: (1) *Eviction-based*, including StreamingLLM (Xiao et al. 2023) (position-based), and H2O (Zhang et al. 2024b), SnapKV (Li et al. 2024c), and CAKE (Qin et al. 2025) (attention-based); (2) *Merging-based*, including LOOK-M (Wan et al. 2024b) and Meda (Wan et al. 2025), which merge evicted KV pairs.

Implementation Details. The compression ratio for evaluation is defined by the average per-head budget B as $\rho = 1 - \frac{B}{T_K}$. Additional HACK implementation details for different models are provided in the *supplementary material*.

Main Results

Text-to-Image Generation. *Quantitative Results:* Table 1 compares HACK with six baseline methods on Infinity (2B/8B) and Hart models at compression ratio $\rho = 70\%$. Compared to baseline methods, which suffer performance degradation in perceptual quality, HACK achieves lossless performance across all evaluation metrics and models, demonstrating strong effectiveness and generalizability. Notably, HACK even surpasses the full KV cache in some cases, underscoring significant redundancy in both attention computation and KV cache in vanilla next-scale generation.

Influence of Compression Ratios: Figure 6 shows HACK's qualitative results at varying compression ratios on Infinity-2B/8B. While baselines struggle to maintain perceptual

Method	GenEval ↑	HPS↑	IR↑	FID↓	CLIP↑
Infinity-2B	0.68	30.49	0.946	10.34	27.52
+Streaming	0.68	29.76	0.901	11.20	27.54
+H2O	0.68	29.60	0.910	10.68	27.57
+SnapKV	0.68	29.60	0.904	10.60	27.56
+LOOK-M	0.67	28.73	0.864	11.14	27.59
+CAKE	0.68	29.46	0.906	10.59	27.56
+MEDA	0.67	28.70	0.867	11.14	27.59
+HACK	**0.68**	**30.18**	**0.933**	**10.56**	**27.62**
Infinity-8B	0.81	30.99	1.049	8.75	28.73
+Streaming	0.81	30.52	1.016	8.98	29.05
+H2O	0.81	30.53	1.020	9.04	29.02
+SnapKV	0.81	30.21	1.015	9.45	29.06
+LOOK-M	0.81	29.99	0.994	10.84	29.04
+CAKE	0.81	30.04	1.002	9.80	29.05
+MEDA	0.79	29.57	0.954	11.56	29.01
+HACK	**0.82**	**30.69**	**1.043**	**8.62**	**29.08**
Hart	0.50	28.75	0.658	10.70	27.69
+Streaming	0.48	25.57	0.458	11.16	27.38
+H2O	0.48	25.69	0.475	11.13	27.32
+SnapKV	0.47	25.62	0.469	11.33	27.33
+LOOK-M	0.37	20.83	0.140	25.64	25.85
+CAKE	0.46	25.89	0.458	12.88	27.21
+MEDA	0.36	20.57	0.117	28.37	25.71
+HACK	**0.50**	**28.36**	**0.658**	**8.58**	**27.76**

Table 1: Quantitative comparison of text-to-image generation on different models with compression ratio $\rho = 70\%$.

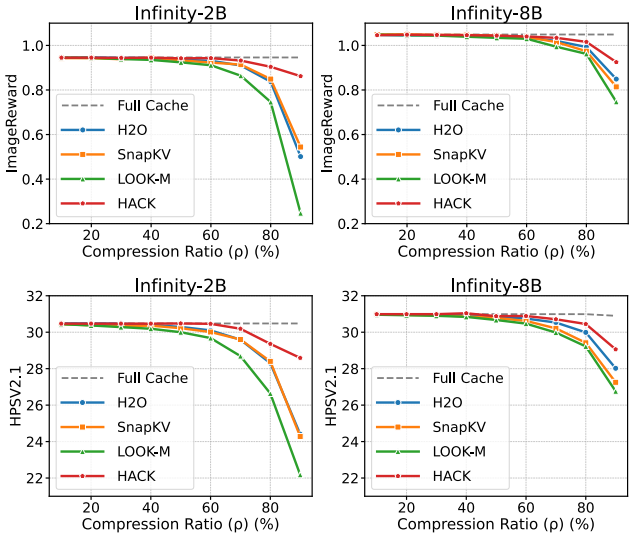


Figure 6: Performance comparison of different compression methods across varying compression ratios measured by ImageReward and HPSv2.1 on Infinity-2B and Infinity-8B.

quality, resulting in substantial performance drops at extreme compression levels ($\rho > 70\%$), HACK consistently preserves robust image quality.

Qualitative Results: We further present qualitative examples from HACK on Infinity-8B in Figure 7. HACK effectively maintains pixel-level fidelity, scene layout, object structure, and semantic coherence with original outputs. Even at extreme compression ($\rho = 90\%$), generated images remain visually faithful, confirming HACK's resilience.

Figure 7: Qualitative results of text-to-image generation by HACK on Infinity-8B. HACK can preserve visual fidelity even under an extreme compression ratio.

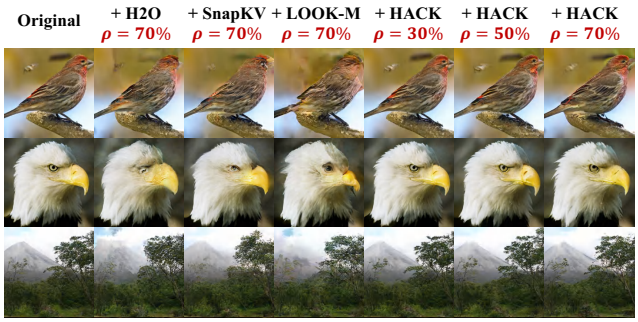


Figure 8: Qualitative va of class-conditional image generation on VAR-d30.

Method	ρ	FID↓	IS↑	Precision↑	Recall↑
VAR-d30	0%	1.96	302.23	0.81	0.60
+Streaming	30%	2.04	297.09	0.80	0.61
+H2O	30%	2.10	295.01	0.81	0.60
+SnapKV	30%	2.13	294.85	0.81	0.60
+LOOK-M	30%	3.32	255.36	0.76	0.61
+HACK	30%	**1.97**	**299.98**	0.81	0.61
+Streaming	50%	2.36	281.30	0.79	0.61
+H2O	50%	3.04	262.68	0.77	0.62
+SnapKV	50%	3.09	261.63	0.77	0.62
+LOOK-M	50%	6.89	203.47	0.70	0.62
+HACK	50%	**2.06**	**293.60**	0.80	0.61
+Streaming	70%	4.84	228.43	0.74	0.62
+H2O	70%	8.81	182.84	0.68	0.62
+SnapKV	70%	7.31	196.62	0.70	0.62
+LOOK-M	70%	18.88	116.70	0.58	0.63
+HACK	70%	**2.78**	**268.69**	0.78	0.62

Table 2: Quantitative comparison of class-conditional image generation on VAR-d30 using the ImageNet dataset.

Class-Conditional Image Generation. *Quantitative Results:* HACK exhibits exceptional performance in class-conditional generation tasks, low resolution of which causes sensitivity to compression. In Table 2, baseline methods experience great performance deterioration (*e.g.*, LOOK-M's

Model	Memory	Com. O	Latency	Speedup
Infinity-2B	28.96GB	–	3.85s	1.00×
+HACK (70%)	14.64GB	0.16s	2.61s	1.48×
Infinity-8B	60.42GB	–	8.14s	1.00×
+HACK (70%)	34.44GB	0.44s	5.17s	1.57×
Hart	35.47GB	–	2.43s	1.00×
+HACK (70%)	23.82GB	0.15s	1.38s	1.76×

Table 3: Efficiency analysis on Infinity and Hart models, evaluated on NVIDIA A100 GPUs with a batch size of 1.

FID score increases by 18.88 at 70% compression). In contrast, HACK consistently demonstrates superior robustness and outperforms all baselines across different compression ratios, aligning with results from text-to-image generation.

Qualitative Results: The qualitative results in Figure 8 align with our quantitative findings. Images generated by baselines exhibit severe distortions, whereas HACK maintains high visual fidelity, highlighting our effectiveness.

Efficiency Analysis

We evaluate HACK's efficiency by comparing memory consumption and inference latency on Infinity and Hart models under standard attention implementations. Table 3 summarizes the results, including additional compression overhead (denoted as Com. O) introduced by HACK. Compared to vanilla VAR models, which suffer from intensive attention computations and cumulative KV cache memory usage, HACK significantly reduces both computational cost and KV cache length, resulting in substantial improvements (*e.g.*, a 1.75× memory reduction and 1.57× speedup on Infinity-8B). Crucially, HACK incurs negligible compression overhead (around $6-8\%$) compared to overall latency.

HACK supports efficient scaling to higher resolutions. As shown in Figure 9, the latency of full attention grows exponentially with increasing resolution, while HACK constrains this growth to be nearly linear, achieving a remarkable 5.8× speedup at 1024×1024 resolution ($\rho = 90\%$). HACK is also compatible with frameworks like FlashAttention (Dao 2023), yielding further memory savings and speedups.

Ablation Study

Impact of Key Components. Table 4 shows that both asymmetric budget allocation and pattern-specific compression are crucial to HACK's effectiveness. Removing either reduces performance, while combining both yields the best results, highlighting their complementary roles.

Ablation Study on Compression Strategies. Table 5 highlights the importance of pattern-specific compression. Removing merging in f_C or deprioritizing key scales in f_S reduces performance, while swapping strategies performs worst, confirming the need for head-type-specific designs. The full method performs best across all metrics.

Impact of Query Subset Selection. Table 6 evaluates sampling strategies for efficient subset attention in importance estimation. We compare four query selection methods with a fixed subset size ($N_{\text{obs}} = 32$): Random sampling, Initial-N (first N tokens), Recent-N (most recent N tokens), and our uniform sampling. Among them, uniform sampling achieves the closest performance to full attention, offering an effective trade-off between efficiency and generation quality, thereby justifying its use in our framework.

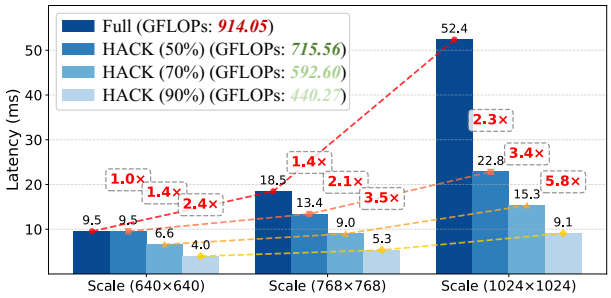


Figure 9: Efficiency Profiling of average latency on attention module for different scales.

Asym. Budget	Pattern Comp.	Infinity-2B IR ↑	Infinity-2B HPS↑	VAR-d30 FID↓	VAR-d30 IS ↑
✗	✗	0.910	29.60	3.04	262.68
✓	✗	0.914	29.58	2.17	288.18
✗	✓	0.923	30.08	2.19	289.24
✓	✓	**0.933**	**30.18**	**2.06**	**293.60**

Table 4: Component ablation of HACK on Infinity-2B ($\rho = 70\%$) and VAR-d30 ($\rho = 50\%$).

Strategy	Infinity-2B IR ↑	Infinity-2B HPS↑	VAR-d30 FID↓	VAR-d30 IS ↑
f_C w/o merge	0.931	30.11	2.09	291.79
f_S w/o init+rec. scales	0.908	29.79	2.17	287.73
$f_C \leftrightarrow f_S$ (Swapped)	0.859	28.63	2.69	271.04
Full f_C+f_S	**0.933**	**30.18**	**2.06**	**293.60**

Table 5: Compression strategy ablation on Infinity-2B ($\rho = 70\%$) and VAR-d30 ($\rho = 50\%$).

Model	Random	Init	Recent	Uniform	Full
Infinity-2B	0.927	0.923	0.927	**0.933**	0.944
Infinity-8B	1.038	1.028	1.027	**1.043**	1.045

Table 6: Query-subset selection strategy ablation on Infinity models evaluated by ImageReward (↑) with $\rho = 70\%$.

Conclusion

We propose HACK, a training-free KV cache compression framework for VAR models. By distinguishing between contextual and structural attention heads, HACK allocates asymmetric cache budgets and applies tailored compression strategies, achieving high compression, reduced KV cache size, and faster inference while maintaining generation quality. Extensive experiments confirm its effectiveness.

Acknowledgements

The paper is supported in part by the National Natural Science Foundation of China (No. U21B2013 , 62325109), and in part by the Shanghai 'The Belt and Road' Young Scholar Exchange Grant (24510742000).

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Anthropic. 2024. The Claude 3 Model Family: Opus, Sonnet, Haiku.

Chen, H.; Su, H.; Sun, P.; and Zhu, J. 2024a. Toward Guidance-Free AR Visual Generation via Condition Contrastive Alignment. *arXiv preprint arXiv:2410.09347*.

Chen, J.; Ge, C.; Xie, E.; Wu, Y.; Yao, L.; Ren, X.; Wang, Z.; Luo, P.; Lu, H.; and Li, Z. 2024b. Pixart-σ: Weak-to-strong training of diffusion transformer for 4k text-to-image generation. In *European Conference on Computer Vision*, 74–91.

Chen, Y.; Lan, Y.; Zhou, S.; Wang, T.; and Pan, X. 2025. SAR3D: Autoregressive 3D Object Generation and Understanding via Multi-scale 3D VQVAE. In *CVPR*.

Dao, T. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition*, 248–255.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Esser, P.; Kulal, S.; Blattmann, A.; Entezari, R.; Müller, J.; Saini, H.; Levi, Y.; Lorenz, D.; Sauer, A.; Boesel, F.; et al. 2024. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*.

Esser, P.; Rombach, R.; and Ommer, B. 2021. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 12873–12883.

Feng, Y.; Lv, J.; Cao, Y.; Xie, X.; and Zhou, S. K. 2024. Ada-kv: Optimizing kv cache eviction by adaptive budget allocation for efficient llm inference. *arXiv preprint arXiv:2407.11550*.

Fu, Y.; Cai, Z.; Asi, A.; Xiong, W.; Dong, Y.; and Xiao, W. 2024. Not all heads matter: A head-level KV cache compression method with integrated retrieval and reasoning. *arXiv preprint arXiv:2410.19258*.

Gao, J.; Liu, W.; Sun, W.; Wang, S.; Song, X.; Shang, T.; Chen, S.; Li, H.; Yang, X.; Yan, Y.; et al. 2025. MARS: Mesh AutoRegressive Model for 3D Shape Detailization. *arXiv preprint arXiv:2502.11390*.

Ge, S.; Zhang, Y.; Liu, L.; Zhang, M.; Han, J.; and Gao, J. 2023. Model tells you what to discard: Adaptive kv cache compression for llms. *arXiv preprint arXiv:2310.01801*.

Ghosh, D.; Hajishirzi, H.; and Schmidt, L. 2023. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 52132–52152.

Han, J.; Liu, J.; Jiang, Y.; Yan, B.; Zhang, Y.; Yuan, Z.; Peng, B.; and Liu, X. 2024. Infinity: Scaling bitwise autoregressive modeling for high-resolution image synthesis. *arXiv preprint arXiv:2412.04431*.

He, W.; Fu, S.; Liu, M.; Wang, X.; Xiao, W.; Shu, F.; Wang, Y.; Zhang, L.; Yu, Z.; Li, H.; et al. 2024a. Mars: Mixture of auto-regressive models for fine-grained text-to-image synthesis. *arXiv preprint arXiv:2407.07614*.

He, Y.; Zhang, L.; Wu, W.; Liu, J.; Zhou, H.; and Zhuang, B. 2024b. ZipCache: Accurate and Efficient KV Cache Quantization with Salient Token Identification. *arXiv preprint arXiv:2405.14256*.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising diffusion probabilistic models. In *Advances in neural information processing systems*, 6840–6851.

Kang, H.; Zhang, Q.; Kundu, S.; Jeong, G.; Liu, Z.; Krishna, T.; and Zhao, T. 2024. Gear: An efficient kv cache compression recipefor near-lossless generative inference of llm. *arXiv preprint arXiv:2403.05527*.

Li, D.; Kamko, A.; Akhgari, E.; Sabet, A.; Xu, L.; and Doshi, S. 2024a. Playground v2. 5: Three insights towards enhancing aesthetic quality in text-to-image generation. *arXiv preprint arXiv:2402.17245*.

Li, T.; Tian, Y.; Li, H.; Deng, M.; and He, K. 2024b. Autoregressive image generation without vector quantization. *Advances in Neural Information Processing Systems*, 37: 56424–56445.

Li, Y.; Huang, Y.; Yang, B.; Venkitesh, B.; Locatelli, A.; Ye, H.; Cai, T.; Lewis, P.; and Chen, D. 2024c. Snapkv: Llm knows what you are looking for before generation. *arXiv preprint arXiv:2404.14469*.

Liu, A.; Feng, B.; Xue, B.; Wang, B.; Wu, B.; Lu, C.; Zhao, C.; Deng, C.; Zhang, C.; Ruan, C.; et al. 2024a. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Liu, A.; Liu, J.; Pan, Z.; He, Y.; Haffari, G.; and Zhuang, B. 2024b. MiniCache: KV Cache Compression in Depth Dimension for Large Language Models. *arXiv preprint arXiv:2405.14366*.

Liu, Z.; Desai, A.; Liao, F.; Wang, W.; Xie, V.; Xu, Z.; Kyrillidis, A.; and Shrivastava, A. 2024c. Scissorhands: Exploiting the persistence of importance hypothesis for llm kv cache compression at test time. *Advances in Neural Information Processing Systems*, 36.

Liu, Z.; Yuan, J.; Jin, H.; Zhong, S.; Xu, Z.; Braverman, V.; Chen, B.; and Hu, X. 2024d. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. *arXiv preprint arXiv:2402.02750*.

Oren, M.; Hassid, M.; Adi, Y.; and Schwartz, R. 2024. Transformers are multi-state rnns. *arXiv preprint arXiv:2401.06104*.

Peebles, W.; and Xie, S. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, 4195–4205.

Podell, D.; English, Z.; Lacey, K.; Blattmann, A.; Dockhorn, T.; Müller, J.; Penna, J.; and Rombach, R. 2023. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*.

Qin, Z.; Cao, Y.; Lin, M.; Hu, W.; Fan, S.; Cheng, K.; Lin, W.; and Li, J. 2025. Cake: Cascading and adaptive kv cache eviction with layer preferences. *arXiv preprint arXiv:2503.12491*.

Ren, S.; Yu, Y.; Ruiz, N.; Wang, F.; Yuille, A.; and Xie, C. 2024. M-VAR: Decoupled Scale-wise Autoregressive Modeling for High-Quality Image Generation. *arXiv preprint arXiv:2411.10433*.

Ren, S.; and Zhu, K. Q. 2024. On the efficacy of eviction policy for key-value constrained generative language model inference. *arXiv preprint arXiv:2402.06262*.

Sun, P.; Jiang, Y.; Chen, S.; Zhang, S.; Peng, B.; Luo, P.; and Yuan, Z. 2024. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*.

Tang, H.; Wu, Y.; Yang, S.; Xie, E.; Chen, J.; Chen, J.; Zhang, Z.; Cai, H.; Lu, Y.; and Han, S. 2024. Hart: Efficient visual generation with hybrid autoregressive transformer. *arXiv preprint arXiv:2410.10812*.

Tian, K.; Jiang, Y.; Yuan, Z.; Peng, B.; and Wang, L. 2024. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in neural information processing systems*, 37: 84839–84865.

Van Den Oord, A.; Vinyals, O.; et al. 2017. Neural discrete representation learning. *Advances in neural information processing systems*, 30.

Voronov, A.; Kuznedelev, D.; Khoroshikh, M.; Khrulkov, V.; and Baranchuk, D. 2024. Switti: Designing Scale-Wise Transformers for Text-to-Image Synthesis.

Wan, Z.; Shen, H.; Wang, X.; Liu, C.; Mai, Z.; and Zhang, M. 2025. Meda: Dynamic kv cache allocation for efficient multimodal long-context inference. *arXiv preprint arXiv:2502.17599*.

Wan, Z.; Wu, X.; Zhang, Y.; Xin, Y.; Tao, C.; Zhu, Z.; Wang, X.; Luo, S.; Xiong, J.; and Zhang, M. 2024a. D2O: Dynamic Discriminative Operations for Efficient Generative Inference of Large Language Models. *arXiv preprint arXiv:2406.13035*.

Wan, Z.; Wu, Z.; Liu, C.; Huang, J.; Zhu, Z.; Jin, P.; Wang, L.; and Yuan, L. 2024b. Look-m: Look-once optimization in kv cache for efficient multimodal long-context inference. *arXiv preprint arXiv:2406.18139*.

Wu, X.; Hao, Y.; Sun, K.; Chen, Y.; Zhu, F.; Zhao, R.; and Li, H. 2023. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. *arXiv preprint arXiv:2306.09341*.

Xiao, G.; Tian, Y.; Chen, B.; Han, S.; and Lewis, M. 2023. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*.

Xu, J.; Liu, X.; Wu, Y.; Tong, Y.; Li, Q.; Ding, M.; Tang, J.; and Dong, Y. 2023. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 15903–15935.

Yang, D.; Han, X.; Gao, Y.; Hu, Y.; Zhang, S.; and Zhao, H. 2024. PyramidInfer: Pyramid KV Cache Compression for High-throughput LLM Inference. *arXiv preprint arXiv:2405.12532*.

Yue, Y.; Yuan, Z.; Duanmu, H.; Zhou, S.; Wu, J.; and Nie, L. 2024. Wkvquant: Quantizing weight and key/value cache for large language models gains more. *arXiv preprint arXiv:2402.12065*.

Zhang, Y.; Du, Y.; Luo, G.; Zhong, Y.; Zhang, Z.; Liu, S.; and Ji, R. 2024a. Cam: Cache merging for memory-efficient llms inference. In *Forty-first International Conference on Machine Learning*.

Zhang, Z.; Sheng, Y.; Zhou, T.; Chen, T.; Zheng, L.; Cai, R.; Song, Z.; Tian, Y.; Ré, C.; Barrett, C.; et al. 2024b. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36.

Zhuang, X.; Xie, Y.; Deng, Y.; Yang, D.; Liang, L.; Ru, J.; Yin, Y.; and Zou, Y. 2025. Vargpt-v1. 1: Improve visual autoregressive large unified model via iterative instruction tuning and reinforcement learning. *arXiv preprint arXiv:2504.02949*.