AdaKVQ: Adaptive Mixed-Precision KV Cache Quantization For Efficient Reasoning Models

Tao Zhang[1] **Yun Dai**[1] **Ziqian Zeng**[1]*
[1]South China University of Technology
zqzeng@scut.edu.cn

Abstract

Large Reasoning Models (LRMs) rely on extensive Chain-of-Thought generation but suffer from memory bottlenecks due to linear Key-Value (KV) cache growth. While low-bit quantization offers a potential solution, we identify that current static quantization strategies suffer from severe degradation on complex reasoning tasks at extreme bit-widths. Notably, this accuracy drop is accompanied by increased generation length, which ironically limits the intended efficiency gains. We attribute this performance collapse to the fact that static quantization methods are insufficiently flexible to meet dynamic demands of precise long-range retrieval. Utilizing Average Attention Distance (AAD) to quantify retrieval patterns, we find that the attention patterns in shallow layers exhibit strong locality, whereas those in deeper layers exhibit intensive long-range dependencies. Furthermore, deep layers have fluctuating AAD values across token positions. Based on these findings, we propose **AdaKVQ**, an adaptive mixed-precision quantization framework that modulates the bit-width allocation guided by the dynamic demands of long-range context retrieval in each layer. Extensive experiments on complex reasoning benchmarks with Llama-3.1 and Qwen2.5 demonstrate that AdaKVQ consistently outperforms state-of-the-art baselines, achieving performance comparable to full-precision models. The source code is available at `https://github.com/ZeroNLP/AdaKVQ`.

1 Introduction

Recent Large Reasoning Models (LRMs), such as OpenAI-o1 [OpenAI, 2024], DeepSeek-R1 [Guo et al., 2025], and Gemini 2.5 Pro [Comanici et al., 2025], have achieved remarkable reasoning performance by employing extended Chain-of-Thought (CoT) generation. This paradigm shift requires models to produce tens of thousands of intermediate tokens to address complex problems. However, this extended generation length exacerbates the inherent computational burdens of autoregressive inference. To generate each successive token, the model must access the Key-Value (KV) states of all preceding tokens to compute the attention output. While storing these states in a KV cache avoids redundant re-computation and accelerates inference, it introduces a severe memory scaling challenge. The cache footprint expands linearly with the sequence length. For instance, a 32B Qwen2.5 model [Qwen et al., 2025] processing a 33k token context with a batch size of 32 necessitates approximately 130 GB for the KV cache alone. Beyond mere capacity constraints, inference throughput is throttled by data movement. Every generation step requires transferring the entire historical KV cache from High Bandwidth Memory (HBM) to on-chip Static Random-Access Memory (SRAM). Notably, the memory capacity of SRAM is limited, hence the KV Cache should be moved to SRAM blocks by blocks. The inference throughput is bottlenecked by the bandwidth [Yu et al., 2022, Yuan et al., 2024].

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

To mitigate these bottlenecks, KV cache quantization has emerged as a promising technique, which compresses memory footprint through the conversion of KV cache from high-precision floating-point to low-bit integer representations. In particular, static quantization strategies [Liu et al., 2024, Hooper et al., 2024, Liu et al., 2025b] have gained widespread adoption due to the training-free nature and ability to maintain competitive accuracy. Specifically, these methods typically maintain recent KV states in original precision while compressing all past KV cache into low-bit representations. However, as illustrated in Figure 1, these methods suffer from severe degradation on complex reasoning tasks when KV states are compressed to extreme bit-widths such as 2-bit. Crucially, we observe that accuracy degradation is accompanied by increased generation length, which ironically limits the intended efficiency gains [Liu et al., 2025a]. For example, as shown in Figure 1, for R1-Qwen-14B [Qwen et al., 2025] under 2-bit quantization, we observed not only a sharp 11.67 percentage point decline in accuracy but also an 8.31% increase in average generation length.

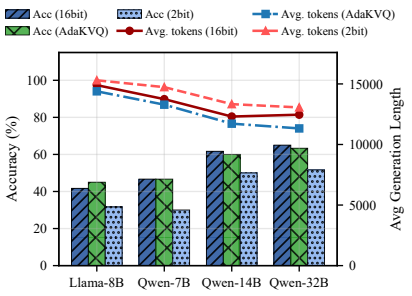


Figure 1: Accuracy and average generation length of different KV cache quantization methods on AIME 2024-2025.

We attribute this performance collapse to the fact that static quantization methods (e.g., KIVI [Liu et al., 2024], KVQuant [Hooper et al., 2024]) are insufficiently flexible to meet dynamic demands of precise long-range retrieval. While general language tasks are less sensitive to quantization errors on historical contexts, complex reasoning tasks demand the high-fidelity preservation of historical context to support multi-step reasoning processes. For instance, while a quantization error in a general language task might only result in a minor lexical substitution, the same error in a reasoning task can corrupt an intermediate calculation, leading to a complete failure of the final output. However, static quantization strategies typically maintain high precision only for recent tokens and apply aggressive compression to all past tokens, which fail to precisely retrieve long-range context that is essential for complex reasoning.

We utilize the *Average Attention Distance* (AAD) metric [Li et al., 2026] to quantify the degree to which attention focuses on short-range versus long-range dependencies. AAD is defined as the weighted average of the positional distances between a query token and the historical keys it attends to, where attention scores serve as the weights. The intuition behind this metric is that when a model performs long-range retrieval, it assigns higher weights to distant tokens, resulting in a larger AAD. We observe distinct attention patterns across layers. Shallow layers exhibit strong temporal locality, primarily attending to the immediate context. In contrast, deep layers exhibit significantly larger average attention distances. Furthermore, these layers display highly variable AAD values across token positions. These observations indicate that context retrieval in complex reasoning tasks is highly non-uniform across both layer depths and token positions.

Based on these findings, we propose **AdaKVQ**, an adaptive mixed-precision quantization framework. AdaKVQ dynamically modulates the bit-width allocation guided by the demands of long-range retrieval in each layer. By allocating larger memory budgets to layers requiring long-range context retrieval while aggressively compressing those layers mainly focusing on local contexts, AdaKVQ effectively preserves the fidelity of long-range dependencies within a constrained memory budget. To evaluate the effectiveness of our method, we conduct experiments on various models including Llama-3.1 [Grattafiori et al., 2024], and Qwen2.5 [Qwen et al., 2025] family. Compared with the state-of-the-art quantization method, our approach can achieve superior performance in complex reasoning tasks [Mathematical Association of America, 2024, 2025, Lightman et al., 2024, Rein et al., 2024]. Our contributions are summarized as follows:

- We provide a quantitative analysis of retrieval patterns using the Average Attention Distance metric. Experimental results reveal significant layer-wise heterogeneity in demands of long-range retrieval, demonstrating that context retrieval is highly non-uniform across both layer depths and token positions.
- Building on these findings, we propose **AdaKVQ**, a novel adaptive mixed-precision quantization method for KV cache quantization. AdaKVQ allocates larger memory budgets to layers requiring long-range context retrieval while aggressively compressing those layers focusing on local contexts.

- Experiments on complex reasoning benchmarks demonstrate that AdaKVQ consistently outperforms existing methods, achieving performance on par with the full-precision baseline.

2 Related Work

In long-sequence inference, the primary bottlenecks regarding memory capacity and inference latency stem from the storage and loading of the KV cache [Yuan et al., 2024]. An effective strategy to mitigate these overheads is to reduce the memory footprint of the KV cache through quantization. The core principle involves compressing full-precision KV cache into compact representations, such as 8-bit, 4-bit, or even 2-bit formats. Unlike weight quantization, which is static and performed once, KV cache quantization requires dynamic execution at runtime, presenting significantly greater challenges.

Per-token vs. Per-channel Quantization. The Key or Value cache for a specific attention head can be formalized as a matrix of shape (L, D), where L denotes the sequence length and D represents the head hidden dimension. Quantization can be applied along two distinct dimensions: the token dimension (L) or the channel dimension (D). Recent studies show that per-channel quantization is essential for Key cache to accommodate channel-specific outlier distributions [Liu et al., 2024, Hooper et al., 2024]. However, per-channel quantization is challenging for growing sequences. Since channel statistics change with new tokens, historical key states must be re-quantized, which results in high computational overhead. To address this, a fixed-size full-precision buffer is typically used to cache recent tokens. In practice, most recent methods [Liu et al., 2024, Duanmu et al., 2024, Hooper et al., 2024, Su et al., 2025a, Tao et al., 2025] generally adopt per-channel quantization for Keys and per-token quantization for Values, complemented by a full-precision buffer to manage streaming updates.

Mixed-Precision and Rotation-based Quantization. To mitigate the performance degradation caused by static low-bit quantization, researchers have developed various strategies. Mixed-precision schemes, such as KVTuner [Li et al., 2025], retain sensitive layers in higher precision. In parallel, approaches like QuaRot [Ashkboos et al., 2024] and RotateKV [Su et al., 2025b] utilize orthogonal rotations to suppress outliers and smoothen feature distribution. However, these methods primarily target general language tasks. They often overlook the heightened risks in reasoning tasks, where long Chain-of-Thought sequences [Wei et al., 2022] are highly susceptible to error accumulation.

Other KV Cache Compression Techniques. Beyond quantization, researchers have explored various alternative compression paradigms. KV Cache Eviction methods enhance computational efficiency by identifying and retaining only critical KV pairs while discarding non-essential tokens [Xiao et al., 2024, Zhang et al., 2023, Han et al., 2024, Cai et al., 2024, Wan et al., 2025, Feng et al., 2025, Tian et al., 2025, Qin et al., 2025]. Another direction focuses on system-level optimizations to improve memory utilization. These approaches include prefilling-decoding disaggregation [Qin et al., 2024], low-rank decomposition [Chang et al., 2024, Zhang et al., 2024, Saxena et al., 2024, Chang et al., 2025], offloading strategies [Lee et al., 2024, Ren et al., 2025], prefetching [Yüzügüler et al., 2025, Dong et al., 2025], and retrieval-based strategies [Tang et al., 2025, Di et al., 2025].

3 Preliminaries

3.1 Transformer and the KV Cache

The core of a Transformer layer is the self-attention mechanism. The output $\boldsymbol{O}_i^l$ for a query tensor $\boldsymbol{Q}_i^l$ at layer l is computed by attending to a sequence of key and value vectors:

$$\boldsymbol{A}_i^l = \text{softmax}\left(\frac{\boldsymbol{Q}_i^l(\boldsymbol{K}^l)^\top}{\sqrt{H}}\right), \quad \boldsymbol{O}_i^l = \boldsymbol{A}_i^l \boldsymbol{V}^l, \tag{1}$$

where H is the model hidden size, $\boldsymbol{A}_i^l$ denotes the attention weight tensor for the i-th token at layer l. During autoregressive generation, the matrices $\boldsymbol{K}^l$ and $\boldsymbol{V}^l$ contain the key and value tensor from all previous timesteps. To avoid costly recomputation, these matrices are stored in a **KV cache**, which grows linearly with the sequence length.

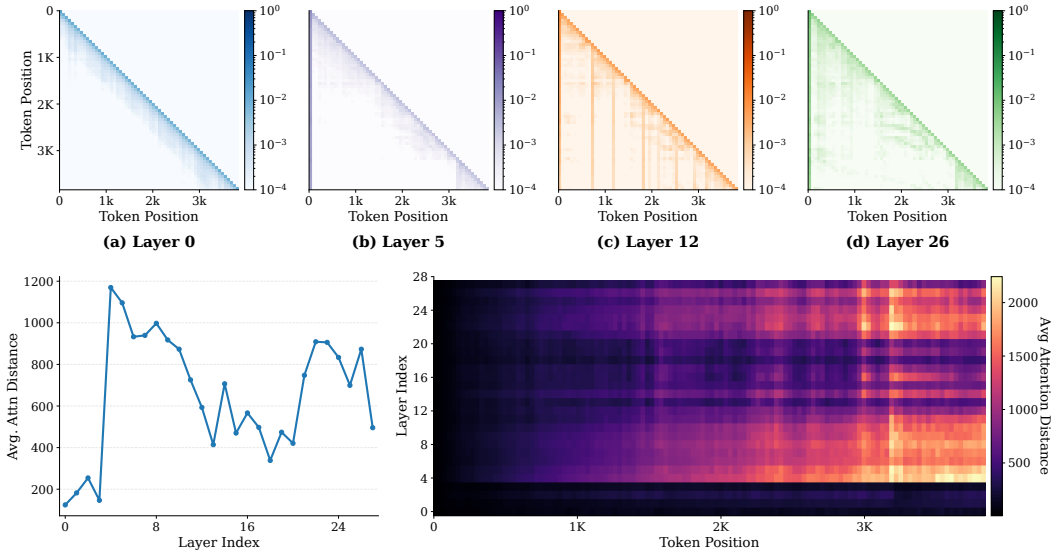


Figure 2: Layer-wise analysis of attention patterns in R1-Qwen2.5-7B. The top row (a-d) visualizes the attention weight matrices for shallow (Layer 0), middle (Layer 5, 12), and deep (Layer 26) layers, highlighting the structural transition from local diagonal patterns to global attention. The bottom row illustrates quantitative metrics: (e) shows the average attention distance per layer, revealing which layers capture long-range dependencies; (f) depicts the token-wise evolution of attention distance for different layers, demonstrating how the context retrieval range expands dynamically cross token positions and layer depths.

3.2 Asymmetric Quantization

We employ the B-bit asymmetric quantization scheme, which approximates a tensor $\boldsymbol{X}$ with its dequantized representation $\tilde{\boldsymbol{X}}$:

$$Q(\boldsymbol{X}) = \text{round}\left(\frac{\boldsymbol{X} - \boldsymbol{z}}{\boldsymbol{s}}\right), \tag{2}$$

$$\tilde{\boldsymbol{X}} = Q(\boldsymbol{X}) \cdot \boldsymbol{s} + \boldsymbol{z}, \tag{3}$$

where $\boldsymbol{z} = \min(\boldsymbol{X})$ is a zero-point and $\boldsymbol{s} = (\max(\boldsymbol{X}) - \min(\boldsymbol{X}))/(2^B - 1)$ is a quantization scaling factor.

4 Method

In this section, we first analyze the layer-wise retrieval patterns of LRMs through the computation of average attention distance, revealing a distinct heterogeneity in long-range retrieval demands in each layer (Sec. 4.1). We show that current quantization methods fail to accommodate the dynamic long-range retrieval demands in reasoning tasks (Sec. 4.2). Consequently, we propose **AdaKVQ**, an adaptive mixed-precision KV cache quantization framework that allocates dynamic cache budgets to meet the different retrieval demands of individual layers (Sec. 4.3).

4.1 Observation: Layer-wise Heterogeneity

To understand how LRMs utilize context during complex reasoning tasks, we analyze the attention patterns of R1-Qwen2.5-7B [Qwen et al., 2025] on the AIME dataset [Mathematical Association of America, 2025]. We compute attention maps through a single forward pass over the concatenated prompt-response sequence. As shown in the top row of Figure 2 (a-d), we observe a clear distinction in attention behaviors across network depths:

- **Shallow Layers (Local Attention):** In early layers (e.g., Layer 0), the attention weights are concentrated heavily along the diagonal. This pattern indicates that shallow layers primarily attend to recent tokens. This suggests that these layers are likely responsible for capturing local syntactic structures and immediate semantic cues.
- **Deep Layers (Global Aggregation):** In contrast, deep layers (e.g., Layers 5, 12, 26) exhibit a more dispersed attention distribution. In these layers, queries interact with distant tokens, indicating that the model is performing long-range context retrieval to aggregate previous information necessary for complex reasoning.

We utilize the **Average Attention Distance** metric [Li et al., 2026] to quantify the degree to which attention focuses on short-range versus long-range context. The intuition behind this metric is that when the model performs long-range retrieval, it assigns significant attention weights to distant tokens, thereby increasing the product of the attention score and relative distance. Formally, let $\mathbf{A}^l \in \mathbb{R}^{N\times N}$ denote the causal attention map for layer l, where N is the sequence length. The average attention distance d^l over the generated response is defined as:

$$d^l = \frac{1}{N}\sum_{t=1}^{N}\sum_{s=1}^{t}\mathbf{A}^l_{t,s}(t-s), \tag{4}$$

Here, $(t-s)$ represents the relative position distance between the query token at position t and the attended key token at position s. Equation 4 calculates the weighted average distance between the query token and the keys it attends to. A low value of d^l indicates that the layer primarily focuses on local context, whereas a high value implies a reliance on distant context.

The bottom row of Figure 2 illustrates the evolution of d^l across layer-wise and token-wise dimensions. The statistics in Figure 2 (e) reveal a distinct disparity in magnitude: shallow layers have a small average distance, whereas deep layers exhibit a significantly larger average distance. The heatmap in Figure 2 (f) reveals the fluctuations of AAD over token positions. Unlike the rigid locality observed in shallow layers, deep layers display a fluctuating attention pattern across tokens. This suggests that the model dynamically adjusts its retrieval needs: expanding to access long-range dependencies when triggered by specific semantic content, and contracting to the immediate context for local generation. These observations underscore that **context retrieval is highly non-uniform across both layer depths and token positions.**

4.2 Motivation: The Limitations of Current Quantization

In contrast to the heterogeneity highlighted in Section 4.1, current KV cache quantization methods, such as KIVI and KVQuant [Liu et al., 2024, Hooper et al., 2024], employ a static strategy. They typically maintain a fixed-size buffer of recent tokens at full precision, and apply aggressive quantization to the rest. While this design suffices for general language tasks, it struggles with complex reasoning. Complex reasoning tasks demand precise error control and the preservation of long-range dependencies, which are often compromised by static approaches.

We demonstrate this limitation using an example from the AIME dataset, as shown in Table 1. In this case, the quantization error from aggressive 2-bit quantization compromises attention computation fidelity, causing the model to retrieve incorrect historical states from the context. This retrieval failure results in a calculation error (80×89) at Step 490. As the model weights are preserved in original precision, its ability to execute arithmetic operations remains uncompromised. We consider that this calculation error is thus a direct consequence of performing operations on incorrect historical states retrieved from the quantized cache. Furthermore, this ambiguity triggers the self-correction mechanism, which leads to increased generation length yet fails to fix the underlying error.

KV cache quantization is fundamentally a multi-objective optimization problem. **It demands a trade-off between minimizing the memory footprint and preserving attention computation fidelity.** However, as highlighted in Section 4.1, the model exhibits dynamic demands of long-range retrieval that vary significantly across layer depths and token positions. Existing static quantization methods are inherently unable to satisfy these demands. This motivates our proposed approach, which focuses on how to effectively allocate cache budgets across different layers to balance these conflicting objectives.

Table 1: Comparison of reasoning chains between BF16 and KIVI-2bit. indicates the correct answer, represents incorrect intermediate results or answers, and denotes information to be retrieved in subsequent steps. Low-bit KV cache quantization induces algebraic hallucinations in complex mathematical reasoning, leading to incorrect derivations.

Question: *Let $ABCD$ be a tetrahedron such that $AB = CD = \sqrt{41}$, $AC = BD = \sqrt{80}$, and $BC = AD = \sqrt{89}$. Find the sum $m + n + p$ for the inradius expressed as $\frac{m\sqrt{n}}{p}$.*	
BF16	[Step 41-43] ...$a^2 + b^2 = 41 \to a = 4$. Then, $b^2 = 25 \to b = 5$. Then $c^2 = 64 \to c = 8$. [Step 88] Thus, $V = \frac{1}{6} \times 320 = \frac{160}{3}$. [Step 154] Thus, $m + n + p = 20 + 21 + 63 =$ 104 .
KIVI-2bit	[Step 487] ... $= 41*($ $80 * 89$ $-64*64)-16*($ $16 * 89$ $-64*25)+25*(16*64-80*25)$. [Step 490] First term:$41 * ($ 7040 $- 4096) = 41 * (2944) = 120,704$. [Step 491] Second term:$-16 * ($ $1,344$ $- 1,600) = -16 * (-256) = +4,096$. [Step 494] So, total determinant: $120,704 + 4,096 - 24,400 = (120,704 + 4,096) =$ $100,400$. [Step 508] ... Therefore, $V = (1/6) * 20 * \sqrt{251} =$ $(10\sqrt{251})/3$. [Step 617-619] "Then, Area $\approx$ sqrt$(14,185)/4 \approx 29.775$ ... **Wait, that's a problem.**" [Step 636-642] "But Heron's formula gave ≈ 29.775, cross product gives ≈ 22.5 ... **Wait, this is conflicting. So, something is wrong.**" [Step 884] "Therefore, the answer is $5\sqrt{5271}/252$, so $m+n+p = 5+5271+252 =$ 5528 ."

4.3 AdaKVQ: Adaptive Key-Value Quantization

Based on the findings above, we propose **AdaKVQ**, an adaptive mixed-precision quantization framework that allocates cache budgets according to the demands of long-range context retrieval in each layer.

Hierarchical Storage Architecture. AdaKVQ partitions the KV cache of each layer into a three-tier hierarchy to optimize the trade-off between retrieval accuracy and memory efficiency. While prior studies suggest that 4-bit quantization is sufficient for general tasks [Liu et al., 2024, Lin et al., 2025, Liu et al., 2025b], we argue that complex reasoning requires a more fine-grained strategy to preserve long-range dependencies. Our design consists of the following components:

- **Full-Precision Cache** (M^l_{full}): To capture local context, this buffer retains the recent R Keys and Values cache in their original precision (FP16/BF16). This ensures that the immediate history is preserved without quantization error.
- **Adaptive High-Precision Cache** (M^l_{adapt}): To handle the fluctuating retrieval patterns in deep layers, this tier stores critical history in high precision. Specifically, we quantize **Keys to UINT8** and **Values to UINT4**. Crucially, the capacity of this cache, C^l, is dynamic. It expands to cover the effective retrieval distance at each step.
- **Compact Low-Precision Cache** (M^l_{comp}): For the remaining context, we apply aggressive compression to minimize memory usage. Specifically, both Keys and Values cache in this tier are quantized to **UINT2**.

AAD-Guided Capacity Modulation. Since the retrieval pattern varies across layers and time steps, a static cache allocation is suboptimal. To address this, we modulate the capacity of the adaptive cache, C^l, using the **Average Attention Distance (AAD)**, d^l (Equation 4). We employ d^l as a quantitative proxy for the demand of long-range context retrieval in layer l. However, as shown in Figure 2 (f), AAD in deep layers varies over token positions. Relying on d^l_t at a single position t is unstable. A temporary shift to local attention would shrink the Adaptive High-Precision Cache. KV states beyond the capacity will be moved to Compact Low-Precision Cache. If the model subsequently switches to long-range context retrieval, the fidelity of these states can not be recovered. To mitigate this issue, we apply an Exponential Moving Average (EMA) to smooth the distance estimation:

$$\bar{d}^l_t = \alpha \cdot d^l_t + (1 - \alpha) \cdot \bar{d}^l_{t-1}, \tag{5}$$

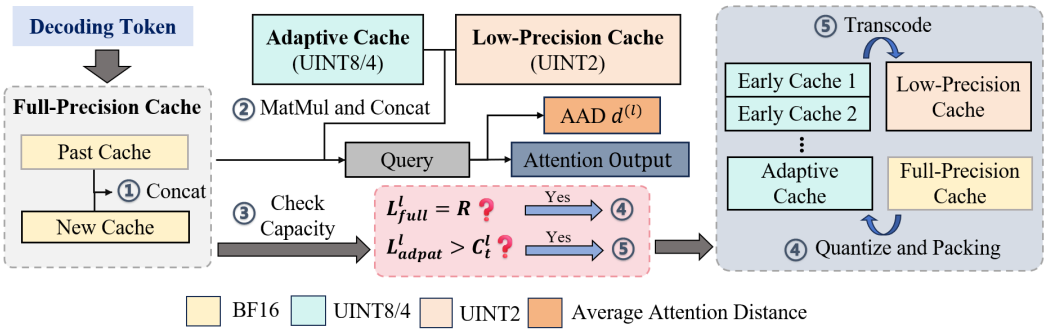


Figure 3: Workflow of AdaKVQ. At each BF16 buffer flush, capacity is checked before admission. If the effective hard limit is exceeded, eligible pages are selected by saliency; otherwise, FIFO is applied when the AAD-derived target is exceeded. Demotion switches to existing K2/V2 pages and recycles K8/V4 slots. A fused quantization pass constructs both representations for incoming pages. Protected prompt pages are excluded from demotion.

where $\alpha \in [0, 1]$ is a smoothing factor.

To determine the target capacity C_t^l at position t, we design a mapping strategy that converts the statistical estimate into a bounded cache budget. This strategy addresses three design objectives. First, since the weighted average d_t^l is inherently less than or equal to the maximum distance of tokens the query may need to attend to, we introduce an inflation coefficient β. This ensures the capacity is sufficient to accommodate these critical, long-range tokens. Second, despite the smoothing process, $\bar{d}_t^l$ may still fluctuate. As the sharp decline in $\bar{d}_t^l$ could shrink the Adaptive High-Precision Cache, KV states beyond the capacity will be moved to compact low-precision cache. We enforce a lower bound $C_{\min}$ to mitigate the risk of irreversible loss. Third, to adhere to memory budget constraints, we impose an upper limit $C_{\max}$. Consequently, the target capacity C_t^l is derived as:

$$C_t^l = \min\left(C_{\max}, \max\left(C_{\min}, \lfloor \beta \cdot \bar{d}_t^l \rfloor\right)\right). \tag{6}$$

Dynamic Cache Management. The overall workflow of AdaKVQ is illustrated in Figure 3. Incoming tokens are maintained in the full-precision buffer M_{full}^l until it reaches size R. The buffered states are then quantized into pages for admission to the adaptive cache M_{adapt}^l. Before admission, the capacity manager checks whether the resident and incoming pages together exceed the effective hard limit. If so, it selects the eligible pages with the lowest saliency scores for demotion. Otherwise, if the effective AAD-derived page limit is exceeded, it selects the oldest eligible pages in FIFO order. These branches are mutually exclusive at each admission step. Selected pages switch to UINT2 in the compact cache M_{comp}^l, and their high-precision slots are reused for incoming pages. The effective page limits and prompt-protection rules are detailed in Appendix B.

System Support. We implement PagedAttention [Kwon et al., 2023] using Triton [Tillet et al., 2019] to support non-contiguous page storage. The high- and low-precision paths each fuse dequantization with attention computation; their partial outputs and softmax statistics are then merged to produce the final attention output. AAD is accumulated alongside attention computation. Implementation details and AAD overhead analysis are provided in Appendices B and D, respectively.

5 Experiments

5.1 Experimental Setup

Models. We evaluate AdaKVQ using the DeepSeek-R1-Distill family [Guo et al., 2025], a series of reasoning-oriented models distilled from DeepSeek-R1. Specifically, we select DeepSeek-R1-Distill-Qwen (7B, 14B and 32B) [Qwen et al., 2025] and DeepSeek-R1-Distill-LLaMA-8B [Grattafiori et al., 2024] for our experiments. All evaluations are conducted on a single NVIDIA A800 GPU (80GB).

Tasks. We assess the performance of AdaKVQ across four diverse reasoning and coding benchmarks: (1) AIME 2024-2025 [Mathematical Association of America, 2024, 2025], comprising 60

Table 2: Performance comparison of BF16, KVQuant, KIVI, KVTuner, RotateKV and AdaKVQ on AIME 2024-2025, MATH 500, GPQA-Diamond, and LiveCodeBench datasets. Bold values indicate the best results among quantized methods.

Model	Method	Bit-width	Accuracy (pass@1)				Avg.
			AIME 2024-2025	MATH 500	GPQA-Diamond	LiveCodeBench	
DeepSeek-R1-Distill-Llama-8B	FullKV	BF16	41.67	89.80	48.99	34.62	53.77
	KIVI	KV4	35.00	86.60	46.46	28.57	49.16
		KV2	31.67	85.80	37.37	17.58	43.11
	KVQuant	KV4	36.67	87.40	44.95	27.47	49.12
		KV2	0	4.20	21.72	12.64	9.64
	RotateKV	KV4	36.67	88.00	42.93	30.22	49.46
	KVTuner-C3.25		30.00	87.80	46.96	31.32	49.02
	AdaKVQ		**45.00**	**88.40**	**48.48**	**32.42**	**53.58**
DeepSeek-R1-Distill-Qwen-7B	FullKV	BF16	46.67	89.00	51.51	31.32	54.63
	KIVI	KV4	36.67	87.00	44.94	20.88	47.37
		KV2	30.00	84.60	35.35	16.48	41.61
	KVQuant	KV4	33.33	87.40	41.41	19.44	45.40
		KV2	1.67	37.80	25.76	5.49	17.68
	RotateKV	KV4	33.33	86.40	42.93	18.33	45.25
	KVTuner-C3.92		31.67	86.00	39.90	19.23	44.20
	AdaKVQ		**46.67**	**89.80**	**53.03**	**30.22**	**54.93**
DeepSeek-R1-Distill-Qwen-14B	FullKV	BF16	61.67	93.60	58.59	45.05	64.73
	KIVI	KV4	55.00	92.20	56.06	42.31	61.39
		KV2	50.00	87.80	52.53	31.32	55.41
	KVQuant	KV4	53.33	90.40	54.58	41.76	60.02
		KV2	0	11.00	22.22	14.29	11.88
	RotateKV	KV4	56.67	92.40	55.56	42.31	61.74
	KVTuner-C2.90		41.67	89.80	50.00	40.11	55.40
	AdaKVQ		**60.00**	**93.20**	**57.58**	**43.96**	**63.69**
DeepSeek-R1-Distill-Qwen-32B	FullKV	BF16	65.00	94.80	62.63	47.25	67.42
	KIVI	KV4	58.33	93.40	59.09	44.51	63.83
		KV2	51.67	89.80	54.54	39.56	58.89
	KVQuant	KV4	60.00	92.60	58.08	43.96	63.66
		KV2	0	42.80	30.81	20.33	23.49
	RotateKV	KV4	61.67	93.40	60.10	42.86	64.51
	KVTuner-C2.91		51.67	91.40	59.60	41.76	61.11
	AdaKVQ		**63.33**	**93.60**	**63.13**	**46.15**	**66.55**

problems from the American Invitational Mathematics Examination to evaluate mathematical reasoning stability; (2) MATH 500 [Lightman et al., 2024], a comprehensive benchmark featuring diverse mathematical problems, tests general reasoning capabilities; (3) GPQA-Diamond [Rein et al., 2024], a graduate-level question-answering benchmark, assesses the capability to generate rigorous proofs and reasoning steps; (4) LiveCodeBench [Jain et al., 2025], a collection of competition-level coding problems. We specifically utilize the subset of problems released between January 1, 2025, and April 6, 2025, to evaluate code generation performance on unseen data.

Baseline. We compare our method against the BF16 baseline and several state-of-the-art quantization methods, including KVQuant [Hooper et al., 2024], KIVI [Liu et al., 2024], KVTuner [Li et al., 2025], and RotateKV [Su et al., 2025b]. Detailed introduction of baselines is shown in Appendix C. For all reasoning evaluations, we set the sampling temperature to 0.6 and top-p to 0.95. To ensure a fair comparison, we standardize the quantization group size at $G = 32$ and the full-precision window size at $R = 128$ across all methods. For AdaKVQ, we configure the hyperparameters as follows: $C_{\text{max}} = 4096$, $C_{\text{min}} = 512$, and the inflation coefficient $\beta = 2$.

5.2 Main Results

Table 2 summarizes the performance of AdaKVQ on the AIME 2024-2025, MATH 500, GPQA-Diamond, and LiveCodeBench datasets. The results demonstrate that AdaKVQ consistently outperforms existing methods across all evaluated models. For instance, on the Qwen-14B model, AdaKVQ achieves an average accuracy of 63.69%, closely matching the BF16 baseline of 64.73%. In comparison, the 4-bit baseline RotateKV attains 61.74% (a drop of 2.99 percentage points), while the 2-bit baseline KIVI degrades to 55.41% (a significant drop of 9.32 percentage points). Furthermore,

as illustrated in Figure 1, experiments with R1-Qwen-14B indicate that AdaKVQ reduces the average generation length by 4.78% and 12.08% compared to the BF16 and KIVI-2bit baselines, respectively. Additionally, by calculating the number of tokens at different precision cache, the effective bit-width of the KV cache for R1-Qwen-14B is estimated to be 2.7 bits. These results highlight the high compression efficiency of AdaKVQ, which effectively mitigates quantization errors while maintaining a low effective bit-width. On eight NIAH tasks at 32K context, AdaKVQ also outperforms KIVI-2bit in retrieval accuracy; detailed results are provided in Appendix G.1.

5.3 Discussion and Ablation.

Impact of AdaKVQ Components. Table 3 ablates the effect of each AdaKVQ component on accuracy, throughput and latency, using R1-Qwen-14B on AIME 2024–2025. For a fair comparison, we use an iso-batch setting with batch size 16. KV quantization with K8V4 operates at an average precision of 6 bits and maintains accuracy comparable to FullKV. As the quantization bit-width decreases, throughput improves, but accuracy decreases substantially. When a uniformly large high-precision cache budget is assigned to all layers, the accuracy loss is small, but the average bit-width remains high and thus limits throughput. AdaKVQ uses AAD as a per-layer dynamic modulation signal for the high-precision cache, which enables strong compression with only a small accuracy loss. Specifically, AdaKVQ expands the high-precision cache in global-attention layers to preserve important KV entries that consistently receive attention, while maintaining a smaller high-precision cache in local-attention layers to protect recent KV entries.

Table 3: Impact of AdaKVQ components.

Method	Avg. Precision	Accuracy	Norm. Throughput	Norm. Latency
FullKV	–	61.67	1.0×	1.0×
K8V4 Quant	6	61.67	1.27×	0.88×
K4V4 Quant	4	55.00	1.35×	0.79×
K4V2 Quant	3	53.33	1.42×	0.75×
K2V4 Quant	3	48.33	1.42×	0.75×
K2V2 Quant	2	50.00	1.79×	0.64×
Multi-tier cache ($C = 4096$)	3.7	60.00	1.33×	0.84×
Multi-tier cache ($C = 2048$)	2.9	55.00	1.48×	0.74×
Multi-tier cache ($C = 1024$)	2.4	53.33	1.65×	0.69×
AdaKVQ	**2.7**	**60.00**	**1.52**×	**0.71**×

Quantization Precision. Table 3 shows that the K8V4 configuration achieves near-lossless performance. Reducing Key precision to 4 bits leads to significant accuracy degradation. Based on this sensitivity, AdaKVQ uses UINT8 for Keys and UINT4 for Values in high-precision layers. We also evaluated more aggressive reductions in the high-precision tier, such as reducing Key precision from 8 to 2 bits, but the resulting accuracy loss outweighs the memory savings. For the low-precision tier, we uniformly apply UINT2 to both Keys and Values. We also considered lower-precision configurations, such as UINT1, but they introduce non-negligible accuracy degradation.

Efficiency Comparison. We evaluate the runtime efficiency of AdaKVQ using workloads synthesized from the ShareGPT dataset [2], following the evaluation protocol of vLLM [Kwon et al., 2023] with $l_{\text{prompt}} = 513$ and $l_{\text{gen}} = 3584$. We increase the batch size to the limit of memory saturation and compare the throughput of AdaKVQ against the BF16 baseline on the R1-Qwen2.5-7B model. As illustrated in Figure 4, under comparable peak memory constraints, AdaKVQ achieves up to $\mathbf{4.38}\times$ higher throughput. We expect these performance gains to scale further with extended generation lengths and larger batch sizes.

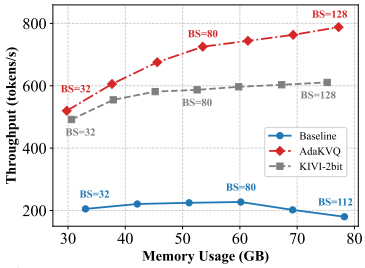


Figure 4: Memory usage and throughput comparison.

6 Conclusion

In this paper, we propose AdaKVQ, a plug-and-play layer-adaptive quantization framework without the need for any tuning. AdaKVQ allocates larger memory budgets to layers requiring long-range retrieval while aggressively compressing those focused on local contexts. AdaKVQ effectively preserves the fidelity of long-range dependencies within a constrained memory budget. The evaluation shows that AdaKVQ achieves optimal performance under extreme low-bit widths in complex reasoning tasks.

[2] https://sharegpt.com/

Acknowledgments and Disclosure of Funding

This work was supported by National Natural Science Foundation of China (62406114), Guangdong Basic and Applied Basic Research Foundation (2025A1515011413), and High Performance Computing Platform of South China University of Technology.

References

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. QuaRot: Outlier-free 4-bit inference in rotated LLMs. In *Advances in Neural Information Processing Systems*, volume 37, pages 100213–100240, 2024.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3119–3137, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.172. URL https://aclanthology.org/2024.acl-long.172/.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Baobao Chang, Junjie Hu, and Wen Xiao. PyramidKV: Dynamic KV cache compression based on pyramidal information funneling. arXiv preprint arXiv:2406.02069, 2024. URL https://arxiv.org/abs/2406.02069.

Zefan Cai, Wen Xiao, Hanshi Sun, cheng Luo, Yikai Zhang, Ke Wan, Yucheng Li, Yeyang Zhou, Li-Wen Chang, Jiuxiang Gu, Zhen Dong, Animashree Anandkumar, Abedelkadir Asi, and Junjie Hu. R-KV: Redundancy-aware KV cache compression for reasoning models. In D. Belgrave, C. Zhang, H. Lin, L. Montoya, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen, editors, *Advances in Neural Information Processing Systems*, volume 38, pages 60980–61005. Curran Associates, Inc., 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/57e0358aed314784cad14a26e1eba642-Paper-Conference.pdf.

Chi-Chih Chang, Wei-Cheng Lin, Chien-Yu Lin, Chong-Yan Chen, Yu-Fang Hu, Pei-Shuo Wang, Ning-Chi Huang, Luis Ceze, Mohamed S. Abdelfattah, and Kai-Chiang Wu. Palu: Compressing KV-cache with low-rank projection. arXiv preprint arXiv:2407.21118, 2024. URL https://arxiv.org/abs/2407.21118.

Chi-Chih Chang, Chien-Yu Lin, Yash Akhauri, Wei-Cheng Lin, Kai-Chiang Wu, Luis Ceze, and Mohamed S. Abdelfattah. xKV: Cross-layer SVD for KV-cache compression. arXiv preprint arXiv:2503.18893, 2025. URL https://arxiv.org/abs/2503.18893.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. arXiv preprint arXiv:2507.06261, 2025. URL https://arxiv.org/abs/2507.06261.

Shangzhe Di, Zhelun Yu, Guanghao Zhang, Haoyuan Li, Tao Zhong, Hao Cheng, Bolin Li, Wanggui He, Fangxun Shu, and Hao Jiang. Streaming video question-answering with in-context video KV-cache retrieval. In *The Thirteenth International Conference on Learning Representations*, 2025.

Yanhao Dong, Yubo Miao, Weinan Li, Xiao Zheng, Chao Wang, Jiesheng Wu, and Feng Lyu. Accelerating LLM inference throughput via asynchronous KV cache prefetching. arXiv preprint arXiv:2504.06319, 2025. URL https://arxiv.org/abs/2504.06319.

Haojie Duanmu, Zhihang Yuan, Xiuhong Li, Jiangfei Duan, Xingcheng ZHANG, and Dahua Lin. SKVQ: Sliding-window key and value cache quantization for large language models. In *First Conference on Language Modeling*, 2024.

Yuan Feng, Junlin Lv, Yukun Cao, Xike Xie, and S Kevin Zhou. Identify critical KV cache in LLM inference from an output perturbation perspective. arXiv preprint arXiv:2502.03805, 2025. URL https://arxiv.org/abs/2502.03805.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, et al. The Llama 3 herd of models. arXiv preprint arXiv:2407.21783, 2024. URL https://arxiv.org/abs/2407.21783v3.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, September 2025. doi: 10.1038/s41586-025-09422-z. URL https://doi.org/10.1038/s41586-025-09422-z.

Chi Han, Qifan Wang, Hao Peng, Wenhan Xiong, Yu Chen, Heng Ji, and Sinong Wang. LM-Infinite: Zero-shot extreme length generalization for large language models. arXiv preprint arXiv:2308.16137, 2024. URL https://arxiv.org/abs/2308.16137.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. KVQuant: Towards 10 million context length LLM inference with KV cache quantization. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 1270–1303. Curran Associates, Inc., 2024.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations*, 2025.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, pages 611–626, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400702297. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Wonbeom Lee, Jungi Lee, Junghwan Seo, and Jaewoong Sim. InfiniGen: Efficient generative inference of large language models with dynamic KV cache management. In *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)*, pages 155–172, Santa Clara, CA, July 2024. USENIX Association. ISBN 978-1-939133-40-3. URL https://www.usenix.org/conference/osdi24/presentation/lee.

Xing Li, Zeyu XING, Yiming Li, Linping Qu, Hui-Ling Zhen, Yiwu Yao, Wulong Liu, Sinno Jialin Pan, and Mingxuan Yuan. KVTuner: Sensitivity-aware layer-wise mixed-precision KV cache quantization for efficient and nearly lossless LLM inference. In *Forty-second International Conference on Machine Learning*, 2025.

Yang Li, Zhichen Dong, Yuhan Sun, Weixun Wang, Shaopan Xiong, Yijia Luo, Jiashun Liu, Han Lu, Jiamang Wang, Wenbo Su, Bo Zheng, and Junchi Yan. Attention illuminates LLM reasoning: The uncovered preplan-and-anchor rhythm enables fine-grained policy optimization. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=ld40RIJnpb.

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. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 22947–22970. Curran Associates, Inc., 2024. doi: 10.52202/079017-0722. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/28ab418242603e0f7323e54185d19bde-Paper-Conference.pdf.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun, editors, *International Conference on Learning Representations*, volume 2024, pages 39578–39601, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/aca97732e30bcf1303bc22ac3924fd16-Paper-Conference.pdf.

Yujun Lin, Haotian Tang, Shang Yang, Zhekai Zhang, Guangxuan Xiao, Chuang Gan, and Song Han. QServe: W4A8KV4 quantization and system co-design for efficient LLM serving. In *Eighth Conference on Machine Learning and Systems*, 2025.

Minghui Liu, Aadi Palnitkar, Tahseen Rabbani, Hyunwoo Jae, Kyle Rui Sang, Dixi Yao, Shayan Shabihi, Fuheng Zhao, Tian Li, Ce Zhang, Furong Huang, and Kunpeng Zhang. Hold onto that thought: Assessing KV cache compression on reasoning. arXiv preprint arXiv:2512.12008, 2025a. URL https://arxiv.org/abs/2512.12008.

Ruikang Liu, Yuxuan Sun, Manyi Zhang, Haoli Bai, Xianzhi Yu, Tiezheng YU, Chun Yuan, and Lu Hou. Quantization hurts reasoning? an empirical study on quantized reasoning models. In *Second Conference on Language Modeling*, 2025b.

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. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32332–32344. PMLR, 21–27 Jul 2024.

Mathematical Association of America. American invitational mathematics examination (AIME) 2024, 2024. URL https://maa.org/maa-invitational-competitions/.

Mathematical Association of America. American invitational mathematics examination (AIME) 2025, 2025. URL https://maa.org/maa-invitational-competitions/.

OpenAI. Introducing OpenAI o1, September 2024. URL https://openai.com/o1/.

Ruoyu Qin, Zheming Li, Weiran He, Mingxing Zhang, Yongwei Wu, Weimin Zheng, and Xinran Xu. Mooncake: A KVCache-centric disaggregated architecture for LLM serving. arXiv preprint arXiv:2407.00079, 2024. URL https://arxiv.org/abs/2407.00079.

Ziran Qin, Yuchen Cao, Mingbao Lin, Wen Hu, Shixuan Fan, Ke Cheng, Weiyao Lin, and Jianguo Li. CAKE: Cascading and adaptive KV cache eviction with layer preferences. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=EQgEMAD4kv.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, et al. Qwen2.5 technical report. arXiv preprint arXiv:2412.15115, 2025. URL https://arxiv.org/abs/2412.15115.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. In *First Conference on Language Modeling*, 2024.

Zebin Ren, Krijn Doekemeijer, Tiziano De Matteis, Christian Pinto, Radu Stoica, and Animesh Trivedi. An I/O characterizing study of offloading LLM models and KV caches to NVMe SSD. In *Proceedings of the 5th Workshop on Challenges and Opportunities of Efficient and Performant Storage Systems*, pages 23–33, 2025.

Utkarsh Saxena, Gobinda Saha, Sakshi Choudhary, and Kaushik Roy. Eigen attention: Attention in low-rank space for KV cache compression. arXiv preprint arXiv:2408.05646, 2024. URL https://arxiv.org/abs/2408.05646.

Yi Su, Yuechi Zhou, Quantong Qiu, Juntao Li, Qingrong Xia, Ping Li, Xinyu Duan, Zhefeng Wang, and Min Zhang. Accurate KV cache quantization with outlier tokens tracing. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12895–12915, Vienna, Austria, July 2025a. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.631. URL https://aclanthology.org/2025.acl-long.631/.

Zunhai Su, Hanyu Wei, Zhe Chen, Wang Shen, Linge Li, Huangqi Yu, and Kehong Yuan. RotateKV: accurate and robust 2-bit KV cache quantization for LLMs via outlier-aware adaptive rotations. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, IJCAI '25, pages 6200–6208, 2025b. ISBN 978-1-956792-06-5. doi: 10.24963/ijcai.2025/690. URL https://doi.org/10.24963/ijcai.2025/690.

Hanlin Tang, Yang Lin, Jing Lin, Qingsen Han, Danning Ke, Shikuan Hong, Yiwu Yao, and Gongyi Wang. RazorAttention: Efficient KV cache compression through retrieval heads. In *The Thirteenth International Conference on Learning Representations*, 2025.

Qian Tao, Wenyuan Yu, and Jingren Zhou. AsymKV: Enabling 1-bit quantization of KV cache with layer-wise asymmetric quantization configurations. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2316–2328, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics. URL https://aclanthology.org/2025.coling-main.158/.

Yuxuan Tian, Zihan Wang, Yebo Peng, Aomufei Yuan, Zhiming Wang, Bairen Yi, Xin Liu, Yong Cui, and Tong Yang. KeepKV: Eliminating output perturbation in KV cache compression for efficient LLMs inference. arXiv preprint arXiv:2504.09936, 2025. URL https://arxiv.org/abs/2504.09936.

Philippe Tillet, H. T. Kung, and David Cox. Triton: an intermediate language and compiler for tiled neural network computations. In *Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages*, MAPL 2019, pages 10–19, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450367196. doi: 10.1145/3315508.3329973. URL https://doi.org/10.1145/3315508.3329973.

Zhongwei Wan, Xinjian Wu, Yu Zhang, Yi Xin, Chaofan Tao, Zhihong Zhu, Xin Wang, Siqi Luo, Jing Xiong, Longyue Wang, and Mi Zhang. D2O: Dynamic discriminative operations for efficient long-context inference of large language models. arXiv preprint arXiv:2406.13035, 2025. URL https://arxiv.org/abs/2406.13035.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc., 2022.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NG7sS51zVF.

Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. Orca: A distributed serving system for Transformer-Based generative models. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, pages 521–538, Carlsbad, CA, July 2022. USENIX Association. ISBN 978-1-939133-28-1. URL https://www.usenix.org/conference/osdi22/presentation/yu.

Zhihang Yuan, Yuzhang Shang, Yang Zhou, Zhen Dong, Zhe Zhou, Chenhao Xue, Bingzhe Wu, Zhikai Li, Qingyi Gu, Yong Jae Lee, Yan Yan, Beidi Chen, Guangyu Sun, and Kurt Keutzer. LLM inference unveiled: Survey and roofline model insights. arXiv preprint arXiv:2402.16363, 2024. URL https://arxiv.org/abs/2402.16363.

Ahmet Caner Yüzügüler, Jiawei Zhuang, and Lukas Cavigelli. PRESERVE: Prefetching model weights and KV-cache in distributed LLM serving. arXiv preprint arXiv:2501.08192, 2025. URL https://arxiv.org/abs/2501.08192.

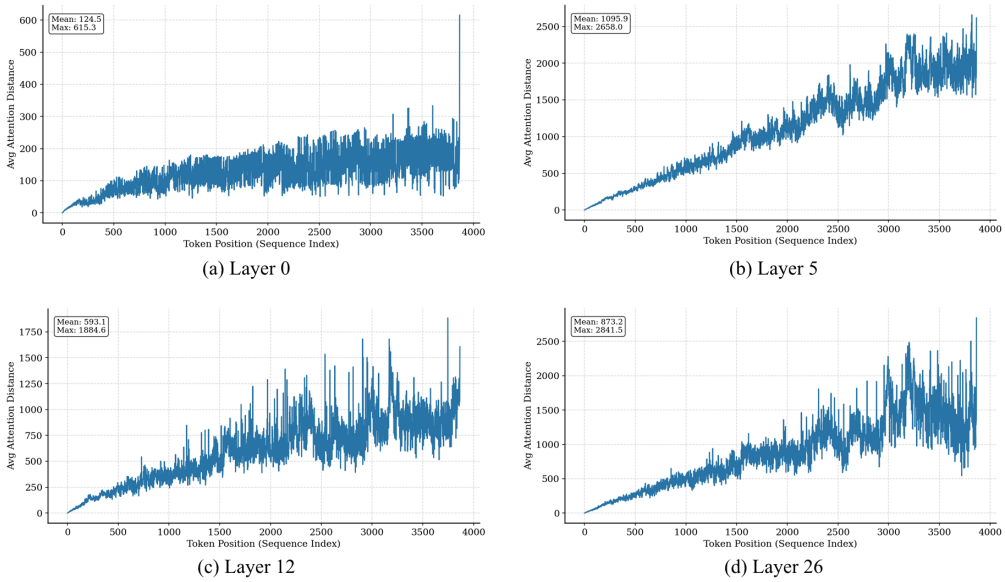
Rongzhi Zhang, Kuang Wang, Liyuan Liu, Shuohang Wang, Hao Cheng, Chao Zhang, and Yelong Shen. LoRC: Low-rank compression for LLMs KV cache with a progressive compression strategy. arXiv preprint arXiv:2410.03111, 2024. URL https://arxiv.org/abs/2410.03111.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang "Atlas" Wang, and Beidi Chen. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 34661–34710. Curran Associates, Inc., 2023.

A Additional Attention Analysis

Figure 5 presents the token-wise average attention distance across representative layers. The fluctuation in deep layers suggests that long-range context retrieval is demand-driven. During the CoT process, the model does not maintain a uniform global focus. Instead, it selectively retrieves the historical KV cache when the current reasoning step requires the prior results.

This provides a strong motivation for AdaKVQ (Sec. 4.3): since the demand for precise long-range retrieval varies across layer depth and time step, the static quantization bit-width would introduce unacceptable quantization errors during critical long-range retrieval.



(a) Layer 0 (b) Layer 5

(c) Layer 12 (d) Layer 26

Figure 5: Layer-wise analysis of attention patterns in R1-Qwen2.5-7B.

B Implementation Details

This section describes the released vLLM implementation of the three-tier cache in Section 4.3. It uses 32-token pages and a 128-token BF16 buffer, with GPU kernels for page construction, capacity management, and attention computation.

B.1 Prefill Phase: Chunk-and-Quantize

During prefill, the input sequence of length L is divided into $\lfloor L/P \rfloor$ complete pages, with $P = 32$. A fused kernel constructs both K8/V4 and K2/V2 representations directly from the BF16 states of each complete page. Keys are quantized per channel within each page, and Values are quantized per token. The K8/V4 representation is initially active for attention, while the K2/V2 representation is retained as its compact backing storage. The remaining $L \bmod P$ tokens enter the full-precision buffer M^l_{full}.

B.2 Decoding Phase: Dynamic Capacity & On-the-Fly Modulation

During decoding, new KV states enter the BF16 buffer. At each 128-token flush, the capacity manager reserves four high-precision page slots for the incoming states and selects resident pages for demotion when required. The attention kernel uses K8/V4 for pages retained in M^l_{adapt} and K2/V2 for demoted pages in M^l_{comp}.

B.2.1 Fused Attention and Metric Computation

To drive the adaptive capacity C^l_t without incurring the cost of a separate inspection kernel, we embed the *Average Attention Distance (AAD)* calculation directly into the decoding attention kernel. The

reduction loop for the query output O simultaneously accumulates the attention-weighted distance:

$$\text{AAD} = \frac{1}{\sum_j P_{ij}} \sum_j P_{ij} \cdot (t - j), \tag{7}$$

where P_{ij} represents the unnormalized attention logits after the online Softmax rescaling. By keeping this reduction in the GPU registers, we obtain the layer-wise instantaneous attention distance with no additional high-bandwidth memory (HBM) read operations. This raw metric is then smoothed via an Exponential Moving Average (EMA) (Equation 5) to determine the target capacity for the adaptive cache. For AdaKVQ, we set smoothing factor $\alpha = 0.5$ in Equation 5.

B.2.2 Dynamic Cache Management

FIFO and saliency share one capacity manager, with the hard-limit branch taking priority. Let n be the number of resident high-precision pages before a flush and n_{prot} the number of protected resident pages. Pages containing prompt tokens are protected, including the page that combines a partial prompt with subsequent generated tokens. The four incoming pages are admitted together, giving a prospective occupancy of $n + 4$. Using the AAD-derived target C_t^l from Equation 6, the effective page limits are

$$H_t^l = \max\left(\frac{C_{\max}}{P}, n_{\text{prot}} + 4\right), \qquad S_t^l = \max\left(\left\lfloor \frac{C_t^l}{P} \right\rfloor, n_{\text{prot}} + 4\right). \tag{8}$$

The protection floor ensures space for the protected history and the incoming batch; long prompts can therefore raise the resident capacity above the nominal $C_{\max}$.

- **Hard-limit branch (saliency).** If $n+4 > H_t^l$, the manager demotes the $n+4-H_t^l$ eligible pages with the lowest saliency scores. This branch completes the selection for the current flush.
- **Soft-limit branch (FIFO).** Otherwise, if $n + 4 > S_t^l$, the manager demotes the $n + 4 - S_t^l$ oldest eligible pages. Age is determined by the logical token position, so reusing a physical slot preserves chronological FIFO ordering.

Both branches exclude protected pages. If neither limit is exceeded, all resident pages are retained. The selected slots are then reused for the four incoming pages.

B.2.3 Block Saliency Score

The attention kernel updates saliency for every resident high-precision page at each decoding step. For page b and KV head h, let $u_{b,h,t}$ be the maximum scaled attention logit over the page's tokens and the query heads associated with h. The stored score follows

$$s_{b,h,t} = \max(s_{b,h,t-1} - 0.5,\ u_{b,h,t})\,. \tag{9}$$

This update retains recent attention peaks while decaying older scores. The capacity manager ranks pages by $\max_h s_{b,h,t}$, preserving a page's importance to any KV head. Protected pages have score $+\infty$; other newly admitted pages start at $-\infty$ and receive their first score in the following attention computation. Score updates and victim selection run on the GPU.

B.2.4 Physical Storage and Precision Switching

The implementation materializes the compact representation when a page first leaves BF16 storage. One fused quantization pass writes K8/V4 codes into a high-precision slot and K2/V2 codes into a vLLM-managed base page. K8 uses one byte per code, V4 packs two codes per byte, and K2/V2 pack four codes per byte. Both representations share the FP32 minimum and range parameters stored in the base page: per-channel parameters for Keys and per-token parameters for Values.

Each layer maintains a logical-to-physical map for high-precision pages and an owner table for reusable slots. Demotion clears the page's high-precision mapping, allowing its slot to be overwritten by an incoming page. Subsequent attention reads the existing K2/V2 base page; the high- and low-precision attention paths select disjoint page sets. Thus, demotion changes the active representation through page-table updates, with no additional requantization or payload copy at that point.

High-precision slots are preallocated separately for each layer and request slot, with capacity determined by $C_{\max}$ and the admitted prompt-length limit. vLLM manages the K2/V2 base-page allocation. The high-precision pool remains allocated as slots are recycled, and each retained high-precision page also has a compact backing copy. Physical memory accounting therefore includes the base pages, reserved high-precision slots, shared quantization parameters, and page-management state.

C Baseline Details

We compare our method against the BF16 baseline and four state-of-the-art quantization methods, categorized into three distinct groups based on their quantization paradigms:

- **Static Quantization**: KIVI [Liu et al., 2024] and KVQuant [Hooper et al., 2024] serve as the representative for static quantization. They apply per-channel quantization for keys and per-token quantization for values, maintaining a fixed low-bit precision for past KV cache while keeping a small streaming window in full precision.
- **Mixed-precision Quantization**: KVTuner [Li et al., 2025] adopts a granular approach by dynamically determining the optimal bit-width for different layers based on their quantization error.
- **Rotation-based Quantization**: RotateKV [Su et al., 2025b] mitigates the quantization error caused by outliers through coordinate rotation. By applying random orthogonal matrix rotations to the key and value matrices, it smooths the numerical distribution.

D Computational Overhead

D.1 AAD Computation

We evaluate the computational overhead of AdaKVQ using samples from the LiveCodeBench dataset [Jain et al., 2025]. Our breakdown of computational costs reveals that the calculation of the Average Attention Distance accounts for only 0.98% of the total inference time. In contrast, the attention mechanism and MLP operations consume 69.31% and 29.71%, respectively. These results demonstrate that the additional computational overhead introduced by the Average Attention Distance calculation is negligible compared to the backbone model operations.

D.2 Dynamic Management Overhead at 32K

Table 4 reports the fraction of end-to-end latency spent on dynamic cache management for Qwen3-8B at 32K context. In the profiled implementation, selected blocks are requantized and physically repacked into UINT2 storage. Transcoding accounts for 0.58% of latency and saliency-score computation for 1.07%, totaling 1.65%. The compact storage uses a global page pool shared across layers and a common free list, allowing layers to draw pages according to their demand.

Table 4: Dynamic management overhead for Qwen3-8B at 32K context using the implementation that repacks blocks during demotion.

Stage	End-to-end latency (%)
Transcode kernel	0.58
Saliency-score computation	1.07
Total dynamic management	1.65

The released implementation described in Appendix B constructs the K2/V2 backing copy together with K8/V4 and switches page mappings at demotion. The AAD computation profile is reported in Appendix D.1.

E Ablation and Sensitivity Analysis

E.1 Component Ablation

We isolate the contributions of saliency-based selection, EMA smoothing, and the minimum adaptive-cache capacity on Qwen3-8B using MATH500. Starting from the full configuration ($\beta = 2$, $\alpha = 0.5$, $C_{\min} = 512$), we remove one component at a time. FIFO-only replaces saliency-based selection at the hard limit with FIFO; the other variants set $\alpha = 1$ or $C_{\min} = 0$. Each removal reduces accuracy by 2.2–3.0 percentage points (Table 5), showing that all three components contribute to the full method.

Table 5: Component ablation on Qwen3-8B. MATH500 scores are reported as avg@1 (%).

Configuration	Removed component	MATH500
Full AdaKVQ	—	**90.8**
FIFO-only	Saliency-based selection	88.6
No EMA ($\alpha = 1$)	Temporal smoothing	87.8
No lower bound ($C_{\min} = 0$)	Minimum adaptive capacity	88.0

E.2 Sensitivity Analysis on Hyperparameters

We evaluate sensitivity to the full-precision buffer size R, adaptive-cache bounds $C_{\min}$ and $C_{\max}$, inflation coefficient β, and smoothing factor α on R1-Qwen-7B and AIME 2024–2025.

Impact of full-precision cache size R. We evaluate the impact of the full-precision cache size by varying $R \in \{64, 128, 256, 512\}$. As illustrated in Figure 6 (a), performance remains relatively stable once R reaches a threshold of 128. However, a smaller cache size (e.g., $R = 64$) results in a performance drop, suggesting that maintaining a sufficiently large full-precision cache for local tokens is critical for preserving accuracy.

Impact of inflation coefficient β. We vary β across $\{1, 1.5, 2, 3\}$. As shown in Figure 6 (b), once β reaches a threshold of 1.5, the performance is stable. When β is small, the Adaptive High-Precision Cache is unable to accommodate critical but distant tokens, leading to performance drops. However, when it reaches a threshold, the performance is not sensitive to β.

Impact of maximum capacity $C_{\max}$. We investigate the impact of $C_{\max}$ varying across $\{1024, 2048, 3072, 4096\}$. As illustrated in Figure 6 (c), expanding the maximum capacity $C_{\max}$ initially leads to significant performance gains. However, the performance tends to stabilize after $C_{\max}$ reaches 2048. This indicates that while higher capacity can improve performance, a moderate size is sufficient to achieve optimal results.

Impact of minimum capacity We examine $C_{\min}$ varying $\{128, 256, 384, 512\}$. As illustrated in Figure 6 (d), expanding the minimum capacity $C_{\min}$ tends to enhance model performance. While higher minimum capacity yields better results, $C_{\min}$ also impact the capacity of adaptive high-precision cache.

Impact of smoothing factor We investigate the impact of α varying across $\{0.1, 0.3, 0.5, 0.7\}$. As illustrated in Figure 6 (e), $\alpha = 0.5$ yields the best trade-off, and performance remains stable across a reasonable range.

E.3 Sensitivity across Models and Tasks

We extend the Qwen-7B/AIME sensitivity analysis in Appendix E.2 to DeepSeek-R1-Distill-Llama-8B on GPQA-Diamond. We vary $C_{\max}$ and β separately, keeping the remaining hyperparameters at their defaults. Table 6 shows that the default configuration ($C_{\max} = 4096$, $\beta = 2$) achieves 48.48% without retuning for this model and task.

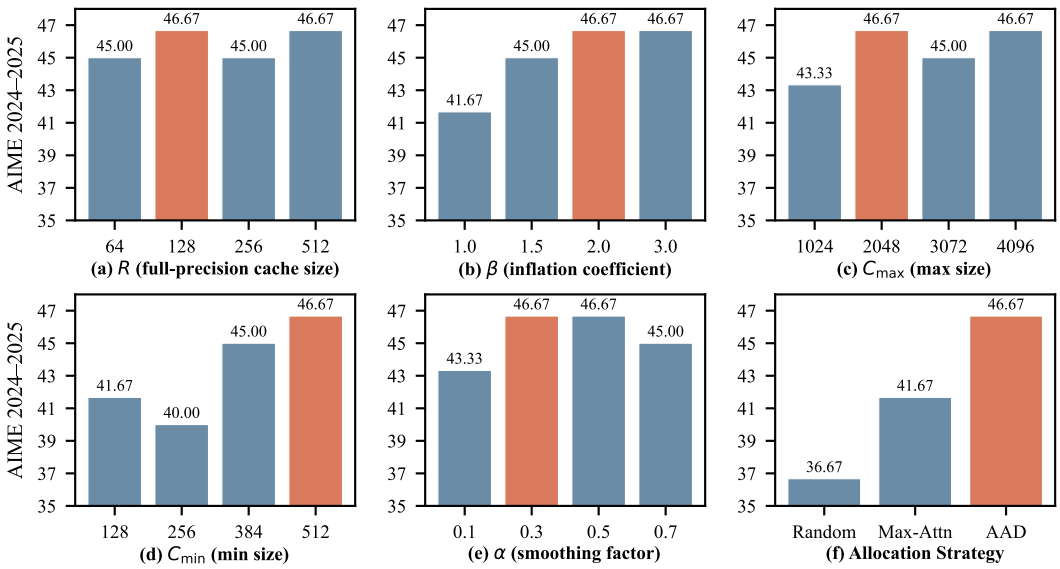


Figure 6: Analysis of hyperparameters and comparison of different allocation strategies for R1-Qwen-7B on AIME 2024–2025.

Table 6: Parameter sensitivity on DeepSeek-R1-Distill-Llama-8B and GPQA-Diamond. Each sweep varies one parameter while holding the others at their defaults.

$C_{\max}$	1024	2048	3072	4096
Accuracy (%)	46.46	46.96	48.98	48.48
β	1.0	1.5	2.0	3.0
Accuracy (%)	47.47	48.48	48.48	49.49

E.4 Comparison of Allocation Strategies

To validate the effectiveness of the AAD-guided capacity modulation proposed in Section 4.3, we compare against two baselines: Max-Attention Score, which uses the distance of tokens with the highest attention scores, serving as an alternative online signal for retrieval importance; Random Allocate, which randomly assigns the high-precision budget. Figure 6 (f) indicates that AAD is the optimal signal for high-precision cache capacity modulation.

F Comparison with Eviction Methods

We conducted experiments comparing AdaKVQ against eviction methods (H2O [Zhang et al., 2023], SnapKV [Li et al., 2024], and R-KV [Cai et al., 2025]). We evaluated the R1-Qwen-14B model on the AIME dataset under a constrained 4096-token cache budget. Table 7 clearly shows that aggressive token eviction indeed causes severe performance degradation in complex reasoning tasks. However, AdaKVQ also outperforms all evaluated eviction baselines.

Table 7: Comparison with eviction methods.

Method	Budget	Accuracy
FullKV	-	61.67
H2O	4096	40.00
SnapKV	4096	46.67
R-KV	4096	55.00
AdaKVQ	4096	**60.00**

G Long-Context Retrieval and Generalization

G.1 Long-Context Retrieval at 32K

We evaluate Qwen3-8B on eight needle-in-a-haystack (NIAH) tasks at a 32K context length, with 500 samples per task. Table 8 compares BF16, KIVI-2bit, and AdaKVQ. AdaKVQ outperforms KIVI-2bit on every task and matches BF16 on Single 1, Single 2, and Multiquery.

Table 8: Retrieval accuracy (%) of Qwen3-8B at 32K context, with 500 samples per task.

Task	BF16	KIVI-2bit	AdaKVQ
Single 1	**100.0**	99.0	**100.0**
Single 2	**100.0**	97.4	**100.0**
Single 3	**100.0**	93.6	98.0
Multikey 1	99.0	97.6	**99.4**
Multikey 2	98.4	96.4	**98.6**
Multikey 3	**97.2**	88.0	94.8
Multiquery	**94.0**	91.2	**94.0**
Multivalue	**76.8**	76.0	76.4

G.2 Generalization to LongBench

To evaluate generalization across diverse task distributions, we tested Llama3.1-8B on the LongBench [Bai et al., 2024] dataset. This benchmark covers question answering (QA), summarization, and few-shot learning. As shown in Table 9, AdaKVQ achieves performance comparable to BF16, with an average accuracy degradation of less than 2%. This suggests that AdaKVQ generalizes well beyond step-by-step reasoning workloads.

Table 9: Performance comparison of BF16, KIVI and AdaKVQ on LongBench datasets. Bold values indicate the best results among quantized methods.

Method	**Bit-width**	**Single-document QA**		**Summarization**		**Few-shot Learning**			**Avg. ↑**
		Qasper	**MultiFieldQA**	**QMSum**	**MultiNews**	**TREC**	**TriviaQA**	**SAMSum**	
Llama3.1-8B									
FullKV	BF16	44.57	55.23	25.75	27.82	72.50	91.65	43.62	51.59
KIVI	KV4	41.19	52.10	23.90	**26.86**	72.50	91.42	43.41	50.20
	KV2	38.56	50.47	22.94	25.28	71.50	91.14	42.96	48.98
AdaKVQ		**42.83**	**53.96**	**24.37**	26.64	**72.50**	**92.09**	**43.52**	**50.84**

G.3 Generalization to MoE Models

We extended our evaluation to a Mixture-of-Experts (MoE) architecture, specifically Qwen3-30B-A3B, which represents a different model family and architectural paradigm from the DeepSeek-R1-Distill series. We report results on AIME 2024–2025 under both non-thinking and thinking modes. As shown in Table 10, AdaKVQ preserves performance under non-thinking mode and incurs only a minor degradation under thinking mode, while achieving significant memory savings.

Table 10: AIME accuracy comparison between BF16, KIVI-4bit, and AdaKVQ on Qwen3-30B-A3B.

Model & Mode	**Method**	**AIME Accuracy**
Qwen3-30B-A3B (Non-thinking)	BF16	26.67
	KIVI-4bit	23.33
	AdaKVQ	26.67
Qwen3-30B-A3B (Thinking)	BF16	68.33
	KIVI-4bit	56.67
	AdaKVQ	65.00

H Failure Case Analysis

Table 11 presents a case in which AdaKVQ correctly identified the methodology and solved the problem in the early steps. However, it failed during the final boundary check at Step 255. While the true root for the final interval is $y \approx 0.9854$ (yielding $x \approx 33.01$), the model's approximation of $y \approx 0.979$ led to $x \approx 32.57$. Consequently, the model incorrectly concluded that the intersection point fell outside the valid domain $[33, 35)$.

This failure highlights a critical challenge in KV cache quantization for complex reasoning tasks. While AdaKVQ helps preserve essential historical information, the compression of intermediate reasoning states (BF16 $\rightarrow$ UINT8/4) can introduce subtle numerical drifts. In rigorous mathematical contexts, even a minor loss of precision (e.g., 0.9854 $\rightarrow$ 0.979) is sufficient to cause a failure in the final output.

Table 11: Comparison of reasoning chains between BF16 and AdaKVQ. indicates the correct answer, and represents incorrect intermediate results or answers,

Question: *A piecewise linear periodic function is defined by* $f(x) = x$ *if* $x \in [-1, 1)$ *and* $2 - x$ *if* $x \in [1, 3)$*, with period 4. The parabola* $x = 34y^2$ *intersects the graph of* $f(x)$*. The sum of the y-coordinates is* $\frac{a+b\sqrt{c}}{d}$*. Find* $a + b + c + d$*.*	
BF16	[Step 809] ... $x = 34y^2 = 34 * ($ $[1 + 5 * sqrt(185)]/68$ $)^2$.
	[Step 824] ... Therefore, in the interval $[33, 35)$, only one solution: $x \approx 33.01$, $y \approx 0.985$.
	[Step 994] So, total sum: $2/68 + [-1 + 5\sqrt{185}]/68 =$ $(1 + 5\sqrt{185})/68$.
	[Step 996] Therefore, the final answer is $1 + 5 + 185 + 68 =$ 259.
AdaKVQ	[Step 255] Case2: $[33, 35)$. ... Then, $x = 34y^2 \approx 34 * ($ 0.979 $)^2 \approx 34 * 0.958 \approx$ 32.57, which is in $[33, 35)$? Wait, 32.57 is less than 33, so invalid.
	[Step 287] Therefore, total sum is $1/34$.
	[Step 629] Summing these gives $1 + 0 + 1 + 34 =$ 36.

I Limitations of AdaKVQ

While AdaKVQ significantly reduces the KV cache memory footprint through its hierarchical mixed-precision strategy, the dynamic management of these hierarchies introduces complexity. Additionally, our study does not encompass all attention mechanisms; specifically, we exclude Multi-Head Latent Attention, which differs significantly from the widely adopted Group-Query Attention. Furthermore, our latency analysis focuses primarily on memory-bound generation stages, leaving the computational bottlenecks in prompt processing phases insufficiently explored.

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: In Section 1

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: In Appendix I

   Guidelines:

   - The answer [N/A] 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: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer [N/A] 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: In section 5 and Appendix B

   Guidelines:

   - The answer [N/A] 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: The code is available at `https://github.com/ZeroNLP/AdaKVQ`.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: In section 5

   Guidelines:

   - The answer [N/A] 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: [No]

   Justification: Limited computational resources prevented us from performing multiple runs for all experiments.

   Guidelines:

   - The answer [N/A] 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: In section 5

   Guidelines:

   - The answer [N/A] 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: This paper does not involve human subjects, does not collect new datasets, and poses no foreseeable harmful consequences. All aspects of the work are consistent with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] 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: [N/A]

    Justification: There is no societal impact of the work performed.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] 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: We cite the original paper that produced the code package or dataset.

    Guidelines:

    - The answer [N/A] 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: We release the code implementation with a README describing the environment setup and license (MIT).

    Guidelines:

    - The answer [N/A] 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: [N/A]

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

    Guidelines:

    - The answer [N/A] 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: [N/A]

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

    Guidelines:

    - The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The LLM is used only for writing, editing, and formatting.

Guidelines:

- The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.