DesireKV: Decoupling Sensitivity and Importance for Reasoning-Aware KV Cache Compression

Pengyu Cheng [2*], Jiacheng Wang [2*], Tianle Chen [2*], Bei Liu[1], Xiaofeng Hou[3], Jiacheng Liu[1†]

[1]The Hong Kong University of Science and Technology, Hong Kong, China
[2]Xi'an Jiao Tong University, Xi'an, China
[3]Shanghai Jiao Tong University, Shanghai, China
{pengyucheng0423,jiacheng,tianlechen1030}@stu.xjtu.edu.cn,{beiliu,jiachengliu}@ust.hk,xfhelen@sjtu.edu.cn

Abstract

Large language models performing chain-of-thought (CoT) reasoning generate extensive intermediate sequences that consume substantial memory through key-value (KV) cache storage. Unlike conventional text generation, reasoning sequences exhibit unique characteristics, including repetitive logic patterns and low information density, making existing KV cache compression methods suboptimal. We propose DesireKV, a novel compression framework that first constructs a two-dimensional coordinate system based on attention-derived importance and outlier-based quantization sensitivity. It then applies a dedicated protection mechanism for tokens critical to the reasoning process itself. Our approach makes differentiated compression decisions: retaining important and sensitive tokens, quantizing important but insensitive tokens, and evicting unimportant tokens. Through comprehensive evaluation on reasoning benchmarks, we demonstrate that DesireKV achieves up to 2.93× throughput improvement while maintaining nearly 99% of original reasoning accuracy.

1 Introduction

Large language models (LLMs) have recently achieved remarkable capabilities in complex reasoning tasks through chain-of-thought (CoT) generation (Wei et al. 2022; Cui et al. 2024). However, state-of-the-art reasoning models such as OpenAI o1 (Jaech et al. 2024), and DeepSeek-R1 (Guo et al. 2025) face a critical deployment bottleneck: their tendency to generate extremely long reasoning traces that can span 16K to 128K tokens, resulting in prohibitive memory consumption due to the quadratic growth of key-value (KV) cache during autoregressive generation.

The memory challenge is particularly acute for reasoning models because of their unique generation characteristics (Paliotta et al. 2025). Unlike conventional text generation, where output length is relatively predictable and bounded, reasoning models deliberately produce verbose intermediate steps to enhance logical accuracy (Yuan et al. 2024). For instance, when solving a mathematical competition problem, DeepSeek-R1-Distill-Llama-8B may generate over 32K tokens of detailed reasoning, running a batch size of 8 in this setting can consume roughly 48.3 GB of total memory (about 15.5 GB for model weights and 32.8 GB for the KV cache, as shown in Figure 1).

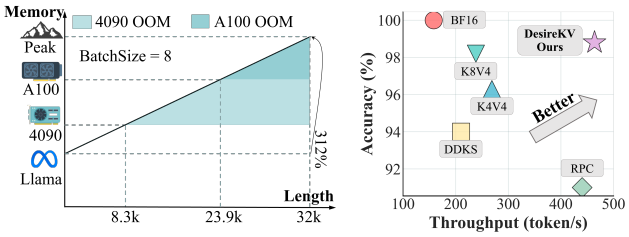


Figure 1: DesireKV addresses the rapid growth of memory consumption with reasoning length to achieve a superior trade-off (Left) Memory consumption grows rapidly with reasoning length. (Right) DesireKV achieves a superior trade-off by attaining the highest throughput while preserving nearly 99% of the original model's accuracy.

Existing KV cache compression methods face significant limitations when applied to reasoning models, leading to three core challenges:

Challenge 1: Dynamic Importance Evolution. Unlike conventional generation tasks, where attention patterns are relatively stable, reasoning sequences exhibit dynamic importance evolution in the reasoning process (Song et al. 2025). Tokens that appear unimportant during early reasoning steps may become critical anchors for later logical connections. Traditional eviction-based methods (Zhang et al. 2023; Li et al. 2024) that aggressively remove tokens based on current attention patterns risk disrupting future logical dependencies, leading to reasoning failures.

Challenge 2: Heterogeneous Quantization Sensitivity. Reasoning sequences contain diverse token types with vastly different tolerance to precision reduction. Mathematical expressions and logical operators are highly sensitive to quantization errors, while explanatory text shows remarkable robustness. Uniform quantization approaches (Hooper et al. 2024; Liu et al. 2024b; Yue et al. 2024) fail to account for this heterogeneity, either applying excessive precision to robust tokens (wasting memory) or introducing harmful noise to sensitive components.

Challenge 3: Real-time Compression Decisions. Reason-

*These authors contributed equally.
†Corresponding author

ing generation requires making compression decisions during inference without knowledge of the complete sequence structure. Offline sensitivity analysis methods (Liu et al. 2024a; He et al. 2024) are impractical for reasoning scenarios where compression must adapt dynamically to the evolving logical structure.

To address these challenges, we propose DesireKV, a unified framework that exploits the fundamental mismatch between importance and sensitivity in reasoning sequences. Our approach operates on the key insight that a token's contextual importance and its numerical sensitivity to quantization are largely decoupled, enabling aggressive compression without performance loss. We leverage recently generated tokens as effective indicators of historical token importance through a periodic compression mechanism that uses recent tokens as a "selector window" to identify relevant historical context. Then, we develop a real-time sensitivity assessment method based on statistical outlier analysis of key activations, focusing on keys due to their disproportionate impact on precision while applying uniform compression to values. Finally, we apply a reasoning-aware protection mechanism that safeguards tokens identified as critical for reasoning transitions. Our framework implements a simple periodic algorithm that makes dedicated allocation decisions, where important tokens are quantized based on their sensitivity profiles while unimportant tokens are evicted regardless of sensitivity, enabling efficient online operation with minimal computational overhead. The main contributions of this work can be summarized as follows,

- To the best of our knowledge, we are the first to identify and empirically validate the *importance-sensitivity mismatch* in KV cache compression for LRMs.
- We propose *DesireKV*, a novel compression framework that operationalizes this mismatch. DesireKV incorporates practical, online methods for assessing importance via an attention-based selector window and measuring sensitivity using real-time outlier analysis.
- We introduce a dedicated protection mechanism for preserving tokens that are critical to the reasoning process, which complements our primary compression strategy.
- We provide a comprehensive experimental evaluation demonstrating up to 2.93$\times$ throughput improvement with minimal accuracy degradation.

2 Related Work

KV cache compression has gained significant attention recently due to its critical role in enabling efficient long-context inference. Eviction-based methods selectively remove tokens based on importance criteria. H2O (Zhang et al. 2023) pioneered attention-based importance assessment through "heavy-hitter" token identification. SnapKV (Li et al. 2024) selects clustered important KV positions per attention head, while Ada-KV (Feng et al. 2024) proposes adaptive budget allocation, CAKE (Qin et al. 2025) introduces cascading eviction with layer-specific preferences, and PyramidKV (Cai et al. 2024) implements pyramidal information funneling across layers. However, aggressive eviction can severely compromise reasoning abilities requiring long-range logical dependencies (Liu et al. 2025a).

Quantization approaches preserve all tokens while reducing numerical precision. KVQuant (Hooper et al. 2024) enables contexts up to 10 million tokens through significant quantization, while KIVI (Liu et al. 2024b) introduced asymmetric quantization for keys and values. More sophisticated approaches recognize heterogeneous sensitivity: IntactKV (Liu et al. 2024a) identifies "pivot tokens" requiring full precision, ZipCache (He et al. 2024) uses salient token identification, AsymKV (Tao, Yu, and Zhou 2024) achieves 1-bit representations with layer-wise configurations, and MiKV (Yang et al. 2024) proposes mixed-precision quantization based on token importance.

Recent work explores hybrid strategies combining multiple techniques. DDKS (Yang et al. 2025) proposes a mixed-precision KV cache method that quantizes pruned tokens to low precision, mitigating performance degradation while preserving output quality and compression efficiency. GEAR (Kang et al. 2024) applies ultra-low precision quantization with low-rank error approximation and sparse matrices for outlier correction. LeanKV (Zhang et al. 2024c) adaptively chooses between eviction and quantization based on token importance. Layer-aware approaches include SimLayerKV (Zhang et al. 2024b), LayerKV (Xiong et al. 2024), and LORC (Zhang et al. 2024a), recognizing varying layer sensitivity patterns. Architectural innovations include DuoAttention (Xiao et al. 2024) with retrieval and streaming heads, and Eigen Attention (Saxena et al. 2024) performing attention in low-rank spaces.

Context-aware methods address semantic structure understanding. ChunkKV (Liu et al. 2025b) introduces semantic-preserving compression, FastKV (Jo et al. 2025) implements token-selective propagation, SepLLM (Chen et al. 2024) compresses segments into separator tokens, and ShadowKV (Sun et al. 2024) maintains shadow copies for high-throughput inference.

Despite these advances, existing methods face fundamental limitations. Most approaches apply uniform strategies across different reasoning patterns and lack principled frameworks for distinguishing between logical anchors, intermediate calculations, and explanatory text. Additionally, many sophisticated approaches require offline sensitivity analysis (Ge et al. 2023), making them unsuitable for real-time reasoning scenarios where compression decisions must be made during generation.

3 Motivation

The efficacy of standard KV cache compression techniques is limited when applied to chain-of-thought reasoning. This is because reasoning sequences exhibit distinct characteristics that uniform eviction or quantization strategies fail to address. In this section, we motivate our work by identifying and analyzing three properties of these sequences.

3.1 Quantization Sensitivity Heterogeneity

A core limitation of many existing methods is the assumption that all tokens tolerate quantization similarly and use token importance to design a quantization strategy (Liu et al.

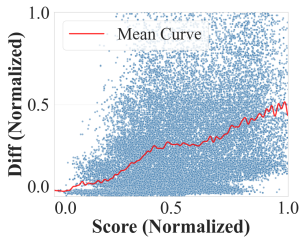


Figure 2: Relationship between outlier scores and quantization sensitivity.

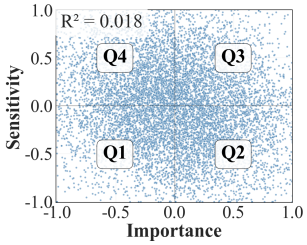


Figure 3: Scatter plot of importance vs. sensitivity scores for 10,000 tokens.

Protection Strategy	Perf.	Mem.
No compression	46.7%	0%
Random protection (20%)	37.9%	57.4%
Outlier-aware protection (20%)	44.6%	57.6%

Table 1: Outlier-aware protective sampling results on AIME 2024 (Pass@1). Perf. means performance, Mem. means the fraction of memory reduced.

2024b). We posit that tokens within reasoning sequences have widely varying sensitivities to precision reduction.

To validate its effectiveness, we sample 10,000 tokens from the reasoning traces of DeepSeek-R1-Distill-Llama8B (Guo et al. 2025) and calculate the interquartile range (IQR) score to find the activation outliers (details can be found in Section 4.3). We compare this outlier score against the actual reconstruction error from quantization. As shown in Figure 2, the strong positive correlation confirms that this outlier score is an effective measure of a token's quantization sensitivity.

Next, we demonstrate that leveraging this sensitivity information is critical for preserving model performance. We conduct a *protective sampling experiment* on the AIME 2024 benchmark, where we preserve 20% of tokens in full BF16 precision while quantizing the remaining 80% to 4-bit. As detailed in Table 1, protecting tokens with the highest outlier scores yields a performance of 44.6% Pass@1. This result significantly outperforms a random protection strategy (37.9%) and nearly recovers the original performance of the uncompressed model (46.7%), confirming that a sensitivity-aware approach is crucial.

3.2 Importance-Sensitivity Mismatch

Our key insight is that a token's contextual importance and its quantization sensitivity are largely decoupled in reasoning sequences. We validate this by computing both metrics for tokens across 50 sequences (the values are normalized between -1 and 1). As shown in Figure 3, the resulting distribution shows a near-zero correlation ($R^2 = 0.018$), confirming their independence and revealing distinct token populations. This independence creates four distinct quadrants, based on high/low importance and sensitivity: *Q1 (Low Importance, Low Sensitivity):* Non-critical and robust tokens. *Q2 (High Importance, Low Sensitivity):* Critical but robust tokens. *Q3 (High Importance, High Sensitivity):* Critical and sensitive tokens. *Q4 (Low Importance, High Sensitivity):* Non-critical but sensitive tokens.

Crucially, a significant fraction of tokens (27.17%) fall into Q2, representing a previously unexploited opportunity for aggressive compression that single-criterion methods miss. This importance-sensitivity mismatch directly motivates DesireKV's design: a dual-criteria policy that evicts unimportant tokens, quantizes important-but-insensitive ones, and preserves important, sensitive tokens with high precision.

Protection Strategy	Perf.	Mem.
No compression	46.7%	0%
Attention driven (20%)	42.9%	57.4%
Reasoning-aware (20%)	44.2%	57.5%

Table 2: Reasoning-aware protective sampling results on AIME 2024 (Pass@1). Perf. means performance, Mem. means the fraction of memory reduced.

3.3 Reasoning-Specific Token Characteristics

Conventional compression methods typically rely on attention scores as the sole indicator of a token's utility. However, we find this metric is insufficient for capturing the nuances of complex reasoning. Specifically, tokens generated with low confidence can signal a critical shift in the model's reasoning trajectory (Liu et al. 2025c) and are thus highly important, even if their attention scores are low.

To validate this hypothesis, we compare two protection strategies. The first is a standard attention-driven approach that preserves the 20% of tokens with the highest attention scores. The second is a hybrid method that also incorporates reasoning-aware generation confidence, replacing the 10% of tokens with the lowest attention scores among the protected set with an equal number of low-confidence tokens. The results in Table 2 show that the hybrid attention-confidence strategy (44.2% Pass@1) measurably improves upon the pure attention-driven approach (42.9%). This substantiates that a more sophisticated, reasoning-aware importance metric is necessary to identify critical tokens that would otherwise be missed.

4 Methodology

Motivated by the unique characteristics of reasoning sequences identified in Section 3, we designed DesireKV to address them directly (Figure 4). Our methodology is built upon the central insight of the *importance-sensitivity mismatch*. We first establish a two-dimensional decision space defined by these two axes. We then introduce a reasoning-aware protection mechanism, motivated by the need to preserve *reasoning-specific tokens*. The final compression decision is a synthesis of these components.

4.1 Problem Formulation

Consider a reasoning model generating sequence $\mathbf{x} = \{x_1, x_2, \ldots, x_T\}$ where x_t represents the token at position

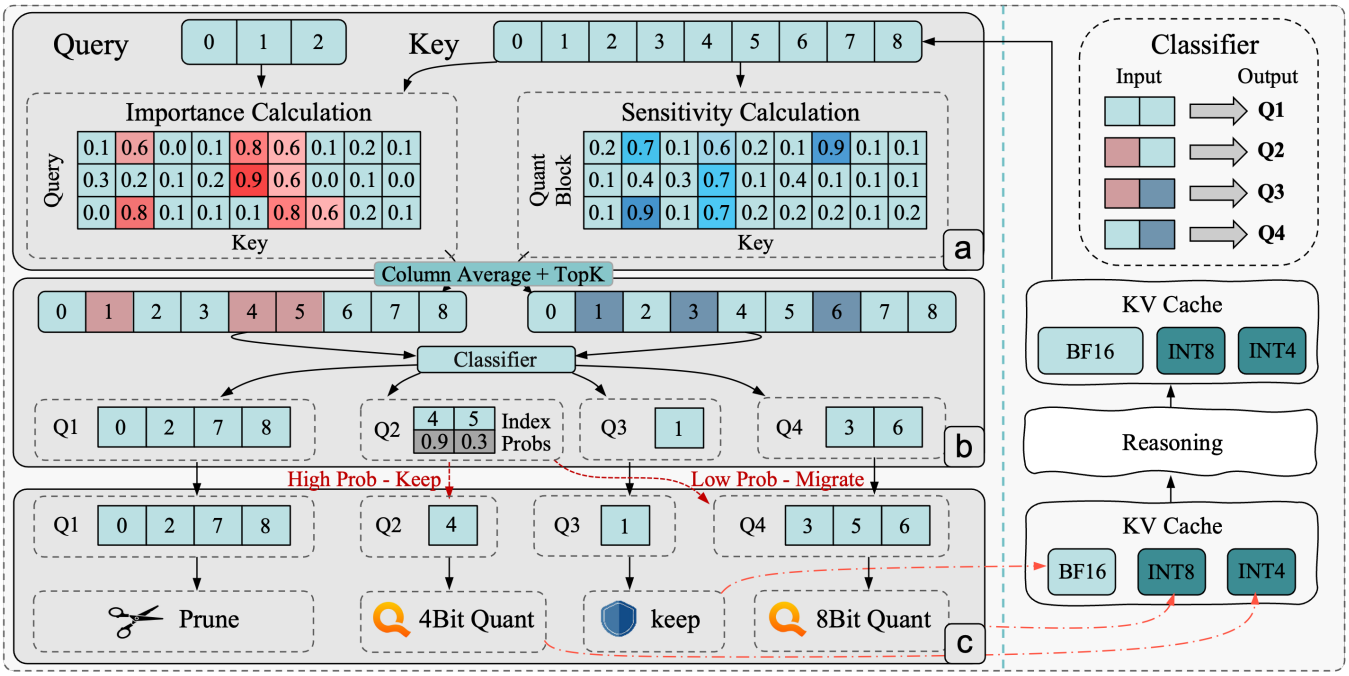


Figure 4: Overall framework of DesireKV: (a) Calculation of importance and sensitivity scores. (b) Based on these scores, a classifier assigns each token to one of four quadrants (Q1-Q4 as in Figure 3). (c) Reasoning-aware protection rule first moves critical tokens to Q4; then, a quadrant-specific action is applied.

t. At generation step t, the model maintains a KV cache containing key and value vectors for all previous tokens across multiple layers:

$$\mathcal{C}_t = \{(\mathbf{K}_i^{(l)}, \mathbf{V}_i^{(l)})\}_{i=1,l=1}^{t-1,L} \tag{1}$$

where $\mathbf{K}_i^{(l)} \in \mathbb{R}^{d_k}$ and $\mathbf{V}_i^{(l)} \in \mathbb{R}^{d_v}$ are the key and value vectors for token i at layer $l \in \{1, 2, \ldots, L\}$, with d_k and d_v being the key and value dimensions respectively.

The memory consumption without compression grows quadratically with sequence length. At step t, the total memory usage is:

$$M_t^{\text{full}} = \sum_{l=1}^{L} \sum_{i=1}^{t-1} (d_k + d_v) \times 16 \text{ bits} \tag{2}$$

This rapid growth creates severe memory bottlenecks for reasoning models that generate tens of thousands of tokens. Our goal is to design a compression function $\mathcal{F} : \mathcal{C}_t \to \mathcal{C}'_t$ that operates periodically to reduce memory consumption while preserving reasoning accuracy.

4.2 Attention-based Importance Assessment

The importance assessment mechanism captures the dynamic relevance of historical tokens for future reasoning steps. Our approach leverages the insight that recently generated tokens in reasoning sequences serve as effective proxies for assessing the continued relevance of historical context.

Our importance assessment approach follows the same selector window mechanism as RPC (Song et al. 2025). Every P tokens, we use the most recent R tokens as a selector window to evaluate the importance of the previous P tokens. The selector window represents the current reasoning state and naturally attends to context that remains relevant for ongoing logical operations. This approach addresses the challenge of predicting future importance by using the immediate future (recent tokens) as an indicator.

For token i at layer l and current generation step t, we compute the importance through multi-head attention aggregation. The computation involves examining how strongly the selector window tokens attend to the historical token across all attention heads:

$$\mathcal{I}_{i,l}^{\text{raw}} = \frac{1}{R \cdot H} \sum_{r=1}^{R} \sum_{h=1}^{H} \text{Attn}_h^{(l)}(\mathbf{q}_{t-r+1}^{(h)}, \mathbf{k}_i^{(l,h)}) \tag{3}$$

In this formulation, R is the selector window size, H is the number of attention heads, $\mathbf{q}_{t-r+1}^{(h)} \in \mathbb{R}^{d_k/H}$ is the query vector for head h at position $t - r + 1$, and $\mathbf{k}_i^{(l,h)} \in \mathbb{R}^{d_k/H}$ is the key vector for token i at layer l and head h.

Then, a sliding window of size w is applied to smooth the semantic information, encouraging the contiguous aggregation of semantically related tokens:

$$\hat{\mathcal{I}}_{i,l} = \frac{1}{2w+1} \sum_{t=-w}^{w} \mathcal{I}_{t,l}^{\text{raw}} \tag{4}$$

The resulting $\hat{\mathcal{I}}_{i,l}$ serves as an attention-based importance score that quantifies how much each historical token contributes to the current reasoning process.

4.3 Outlier-based Sensitivity Measurement

To address the *heterogeneous quantization sensitivity* identified in Section 3.1, we measure sensitivity based on statistical outlier analysis of activation patterns within key vectors. The fundamental insight is that tokens with extreme activation values (outliers) are more susceptible to quantization errors because quantization schemes typically optimize for typical values in the distribution (Frantar et al. 2022). Outliers fall outside the optimal quantization range and suffer from larger approximation errors.

For computational efficiency and statistical robustness, we employ a block-based analysis approach. Each key vector is partitioned into fixed-size blocks, and outlier analysis is performed independently within each block. This approach provides several advantages: it reduces computational complexity by avoiding full-vector statistics, it provides more stable estimates by working with smaller data samples, and it captures local activation patterns that might be obscured in global analysis.

For key vector $\mathbf{K}_i^{(l)} \in \mathbb{R}^{d_k}$ at token i and layer l, we create blocks of size $g = 64$:

$$\mathbf{K}_i^{(l)} = [\mathbf{K}_i^{(l,1)}, \mathbf{K}_i^{(l,2)}, \ldots, \mathbf{K}_i^{(l,B)}] \tag{5}$$

where each block $\mathbf{K}_i^{(l,b)} \in \mathbb{R}^g$ contains g consecutive elements and $B = \lceil d_k/g \rceil$ is the total number of blocks.

Within each block, we perform Interquartile Range (IQR) based outlier detection, which is a robust statistical method that is less sensitive to extreme values than mean-based approaches. For block b, we first sort the elements and compute the quartiles. The first quartile $Q_1^{(b)}$ is the 25th percentile, the third quartile $Q_3^{(b)}$ is the 75th percentile, and the interquartile range is $\text{IQR}_b = Q_3^{(b)} - Q_1^{(b)}$. Elements are classified as outliers using the standard IQR criterion:

$$x \in \mathcal{O}_b \iff x < Q_1^{(b)} - 1.5 \cdot \text{IQR}_b \text{ or } x > Q_3^{(b)} + 1.5 \cdot \text{IQR}_b \tag{6}$$

This criterion identifies values that fall more than 1.5 IQR units beyond the first or third quartiles, which statistically corresponds to extreme values in the distribution. The sensitivity score for each block quantifies both the magnitude and frequency of outliers. We compute the block mean $\mu_b = \frac{1}{g}\sum_{j=1}^{g} \mathbf{K}_i^{(l,b)}[j]$ and then measure the average deviation of outliers from this center:

$$\text{OutlierScore}_b = \begin{cases} \frac{1}{|\mathcal{O}_b|}\sum_{x \in \mathcal{O}_b} |x - \mu_b| & \text{if } |\mathcal{O}_b| > 0 \\ 0 & \text{if } |\mathcal{O}_b| = 0 \end{cases}$$

This metric captures both the presence of outliers and their magnitude, providing a comprehensive measure of quantization sensitivity. To obtain a token-level sensitivity score, we aggregate it across all blocks and layers. The layer-wise aggregation computes the average outlier score across all blocks in a layer:

$$\mathcal{S}_{i,l} = \frac{1}{B}\sum_{b=1}^{B} \text{OutlierScore}_b^{(i,l)} \tag{7}$$

Finally, to ensure a consistent threshold across generation process, we normalize sensitivity scores as:

$$\hat{\mathcal{S}}_{i,l} = \frac{\mathcal{S}_{i,l}}{\max_j \mathcal{S}_{i,l}} \tag{8}$$

where the maximum is computed over all tokens in the current compression window.

4.4 Reasoning-Aware Token Protection

After calculating importance $\hat{\mathcal{I}}_{i,l}$ and sensitivity $\hat{\mathcal{S}}_{i,l}$, we apply a separate, reasoning-aware protection mechanism. This is motivated by Section 3.3 certain tokens critical for the *reasoning process* (e.g., at logical transitions) can be missed by attention scores alone.

We use generation confidence as a proxy for identifying reasoning transitions. A sharp drop in confidence often signals a point of uncertainty or a shift in logic, making the subsequent tokens crucial (Liu et al. 2025c). We compute the generation confidence for each token as:

$$p_i = \max_v \text{softmax}(o_i)[v] \tag{9}$$

where o_i is the logit vector for position i. If the confidence p_{i-1} for the previous token falls below a threshold λ, we mark the current token i as a "protected" reasoning token. This ensures that the anchor of a new reasoning step is preserved. A token i is added to the protected set $\mathcal{P}$ if $p_{i-1} < \lambda$. Then we set the tokens in $\mathcal{P}$ as high precision.

Finally, we can apply a dedicated eviction and quantization strategy as shown in Figure 4.

5 Experiments

5.1 Experimental Setup

Models and Datasets We conduct our evaluation on two prominent open-source reasoning models: *DeepSeek-R1-Distill-Qwen-7B* and *DeepSeek-R1-Distill-Llama-8B* (Guo et al. 2025). For all experiments, we set the decoding temperature to 0.6, as recommended by the model authors, and a maximum generation length of 32,786 tokens.

Our evaluation comprises three mathematical reasoning datasets: GSM8K (Cobbe et al. 2021), MATH (Hendrycks et al. 2021) (using the 500-sample test set), and AIME2024. We also include the general-purpose reasoning benchmark GPQA (Rein et al. 2024) (using the GPQA-Diamond subset) to assess broader capabilities. We report accuracy using the `pass@k` metric, where a problem is considered solved if at least one of the k generated solutions is correct. We use $k = 1$ for all datasets, except for GPQA on the Qwen model, where we set $k = 2$ to mitigate the noticeable performance drop across all methods.

Baselines Except for the full-precision model (**BF16**), our baselines include three major families of KV cache compression techniques, enabling a comprehensive comparison:

- *Quantization-Only:* We use KIVI (Liu et al. 2024b) with two configurations: aggressive 4-bit key/4-bit value quantization (**K4V4**) and a more conservative 8-bit key/4-bit value setup (**K8V4**).

DeepSeek-R1-Distill-Qwen-7B										
Method	GSM8K	% Loss	MATH	% Loss	AIME2024	% Loss	GPQA	% Loss	Ave Bit	Ave Loss (%)
BF16	0.8915	—	0.9185	—	0.5417	—	0.4785	—	16	—
K8V4	0.8811	**1.2**	0.9123	**0.7**	0.5333	1.5	0.4600	3.9	6.0	1.8
K4V4	0.8795	1.4	0.9031	1.7	0.5042	6.9	0.4558	4.7	4.3	3.7
RPC	0.8574	3.8	0.8885	3.3	0.4250	21.5	0.4431	7.4	4.0	9
DDKS	0.8698	2.4	0.8925	2.8	0.4750	10.9	0.4407	7.9	5.9	6
Ours	0.8797	1.3	0.9028	1.7	0.5343	**1.3**	0.4760	**0.5**	**3.0**	**1.2**

DeepSeek-R1-Distill-Llama-8B										
Method	GSM8K	% Loss	MATH	% Loss	AIME2024	% Loss	GPQA	% Loss	Ave Bit	Ave Loss (%)
BF16	0.8836	—	0.8705	—	0.4667	—	0.4759	—	16	—
K8V4	0.8762	0.8	0.8595	**1.3**	0.4583	1.8	0.4691	**1.4**	6.0	1.3
K4V4	0.8624	2.4	0.8505	2.3	0.3586	23.2	0.4141	13.0	4.0	10.2
RPC	0.8697	1.6	0.8425	3.2	0.4333	7.2	0.4015	15.6	4.0	6.9
DDKS	0.8561	3.1	0.8325	4.4	0.3375	27.7	0.4047	15.0	5.5	12.6
Ours	0.8811	**0.3**	0.8595	**1.3**	0.4625	**0.9**	0.4684	1.6	**2.9**	**1.0**

Table 3: Pass@1 comparison of different methods on GSM8K, MATH, AIME2024, and GPQA benchmarks for DeepSeek-R1-Distill-Qwen-7B and DeepSeek-R1-Distill-Llama-8B. The Loss means loss ratio compared to BF16 baseline.

- *Eviction-Only:* We compare against **RPC** (Song et al. 2025), a state-of-the-art attention-based token eviction method for reasoning model.
- *Hybrid:* We include **DDKS** (Yang et al. 2025), which combines eviction with quantization by compressing all the pruned tokens to a low precision.

Implementation Details We set the hyperparameters P and R according to the difficulty of each dataset, and align the parameters of RPC and KIVI accordingly. For DesireKV, we determine key thresholds via grid search to balance compression efficiency and reasoning performance. For DDKS, since there are no public implementations, we implement it by extending RPC and quantizing pruned tokens to 2 bits. Thus, all compression algorithms employ the same interval for dynamic optimization, enabling us to quantify the compressed bit volume at each compression step and directly compare their compression capabilities. Furthermore, we analyze the efficiency of each method by comparing memory consumption under identical batch sizes and throughput rates under equivalent memory constraints.

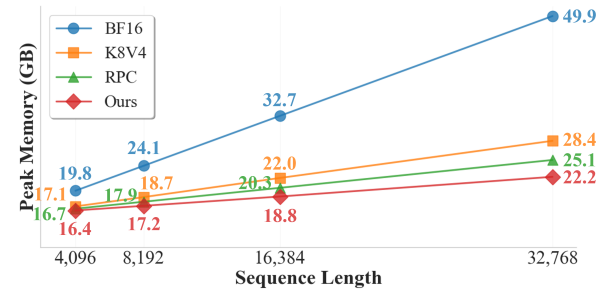
5.2 Performance Evaluation

Table 3 presents our primary findings on the four reasoning benchmarks. Compared to the full-precision baseline (BF16), our approach realizes highly efficient model compression with negligible performance degradation. In contrast to the RPC method, our method exhibits significantly lower average loss, amounting to only 1.2% and 1.0% on DeepSeek-R1-Distill-Qwen-7B and DeepSeek-R1-Distill-Llama-8B, respectively. Compared to the quantization methods K8V4 and K4V4, our method not only maintains a lower average bit width but also achieves performance comparable to or even better than K8V4, thereby effectively avoiding the severe performance degradation typically associated with aggressive quantization. Against the eviction-only RPC method, DesireKV demonstrates substantially better performance, mitigating the severe accuracy drops RPC suffers on complex tasks like AIME2024 (e.g., 21.5% loss for RPC vs. 1.3% for Ours on Qwen-7B). This highlights the importance of retaining, rather than discarding, low-importance tokens via quantization. Furthermore, in comparison with the hybrid method like DDKS, our method achieves superior performance while simultaneously reducing memory footprint, underscoring the superiority of our dual importance-sensitivity criteria for making nuanced compression decisions.

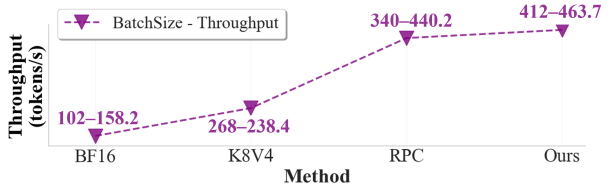
Method	GSM8K	MATH	AIME2024	GPQA
DesireKV	**0.8811**	**0.8595**	**0.4625**	**0.4684**
DesireKV-I	0.8700	0.8455	0.4375	0.4470
DesireKV-S	0.8614	0.8260	0.4292	0.4432
DesireKV-P	0.8756	0.8545	0.4583	0.4595

Table 4: Ablation study of DesireKV's components on DeepSeek-R1-Distill-Llama-8B. Disabling the importance (-I), sensitivity (-S), or protection (-P) mechanisms consistently degrades performance, validating their respective contributions.

In summary, DesireKV achieves a compression ratio of over 5.5× with negligible performance loss (0.3-1.7%), affirming that our method effectively navigates the complex trade-offs in KV cache compression to set a new state-of-the-art for efficient and accurate long-context reasoning.



(a) Peak Memory Usage Comparison



(b) Throughput Performance Comparison

Figure 5: Overall performance comparison: (a) illustrates the peak memory consumption of different methods with a fixed batch size of 8; (b) depicts the achievable throughput of each method under maximum memory capacity (96 GB), where the annotations above the data points represent the {batch size-throughput} pairs.

5.3 Efficiency Evaluation

We now evaluate the practical efficiency of DesireKV on an NVIDIA H20 GPU, using the DeepSeek-R1-Distill-Llama-8B model. Our analysis focuses on two critical deployment metrics: peak memory footprint and inference throughput. We benchmark against representative baselines: the full-precision (BF16) model, the high-compression eviction method RPC, and the performant K8V4 quantization scheme. To provide a comprehensive assessment, we conduct two controlled experiments: (1) measuring peak memory usage under a fixed batch size, and (2) measuring maximum achievable throughput under a fixed memory capacity.

Memory Optimization As illustrated in Figure 5(a), all methods exhibit a linear increase in peak memory with sequence length. RPC achieves a 49.7% memory reduction relative to the BF16 baseline at 32,768 tokens. DesireKV further improves upon this, achieving an additional 11.6% reduction over RPC. This culminates in a total memory saving of 55% compared to the full-precision model, significantly alleviating the memory pressure of long-context reasoning.

Throughput Efficiency Figure 5(b) demonstrates the practical impact of memory savings on throughput. It is evident that the K8V4 experimental group, suffering from inadequate compression capability and the timing overhead induced by quantization, exhibits lower throughput than the RPC pruning method under similar batch sizes. While DesireKV incurs slight computational overhead from its online analysis, its superior memory efficiency permits much larger batch sizes. This amortization of overhead results in substantial throughput gains. Under 96 GB memory constraint, DesireKV achieves a remarkable 193.1% throughput improvement over the BF16 baseline and significantly outperforms all other compression methods, thereby enhancing operational efficiency.

5.4 Ablation Study

To dissect the contribution of each component in our framework, we conducted a series of ablation studies on the DeepSeek-R1-Distill-Llama-8B model. The variants are defined as follows:

- **DesireKV-I** (Importance Ablation): We replace our attention-guided importance assessment from Section 4.2 with a random eviction strategy.
- **DesireKV-S** (Sensitivity Ablation): We ignore our outlier-based sensitivity score from Section 4.3. Important tokens are randomly quantized to 4-bit or 8-bit.
- **DesireKV-P** (Protection Ablation): We remove the reasoning-aware protection mechanism from Section 4.4.

The results, presented in Table 4, demonstrate that removing any component degrades performance. The most pronounced performance drop is observed in DesireKV-S, where disabling sensitivity-aware quantization leads to significant accuracy loss. This finding validates our central hypothesis that protecting numerically sensitive tokens from aggressive quantization is more critical than identifying contextual importance alone. The degradation in DesireKV-I confirms that our reasoning-aware importance metric is superior to random eviction. Finally, the performance drop in DesireKV-P shows the value of using generation confidence to identify and preserve critical reasoning transition points. Collectively, the study affirms that all components are integral to DesireKV's success, with the dual-criteria analysis of importance and sensitivity being the most crucial element.

6 Conclusion

We presented DesireKV, a novel KV cache compression framework that addresses the prohibitive memory consumption of large language models in complex reasoning tasks. Our approach leverages the key insight that importance and sensitivity are mismatched in reasoning sequences, enabling aggressive compression through strategic eviction and asymmetric quantization. Extensive experiments demonstrate that DesireKV achieves up to a 5.5× memory reduction and a 2.93× throughput improvement, while preserving nearly 99% of the original model's reasoning accuracy. Our findings set a new state-of-the-art in balancing performance and resource efficiency.

Acknowledgments

We sincerely thank all the anonymous reviewers for their valuable comments and feedback. This work is supported by the National Natural Science Foundation of China No. 62441225. Jiacheng Liu is the corresponding author.

References

Cai, Z.; Zhang, Y.; Gao, B.; Liu, Y.; Li, Y.; Liu, T.; Lu, K.; Xiong, W.; Dong, Y.; Hu, J.; et al. 2024. Pyramidkv: Dynamic kv cache compression based on pyramidal information funneling. *arXiv preprint arXiv:2406.02069*.

Chen, G.; Shi, H.; Li, J.; Gao, Y.; Ren, X.; Chen, Y.; Jiang, X.; Li, Z.; Liu, W.; and Huang, C. 2024. Sepllm: Accelerate large language models by compressing one segment into one separator. *arXiv preprint arXiv:2412.12094*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Cui, C.; Ma, Y.; Cao, X.; Ye, W.; Zhou, Y.; Liang, K.; Chen, J.; Lu, J.; Yang, Z.; Liao, K.-D.; et al. 2024. A survey on multimodal large language models for autonomous driving. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*.

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*.

Frantar, E.; Ashkboos, S.; Hoefler, T.; and Alistarh, D. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.

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*.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

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

Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; and Steinhardt, J. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Hooper, C.; Kim, S.; Mohammadzadeh, H.; Mahoney, M. W.; Shao, Y. S.; Keutzer, K.; and Gholami, A. 2024. Kvquant: Towards 10 million context length llm inference with kv cache quantization. *arXiv preprint arXiv:2401.18079*.

Jaech, A.; Kalai, A.; Lerer, A.; Richardson, A.; El-Kishky, A.; Low, A.; Helyar, A.; Madry, A.; Beutel, A.; Carney, A.; et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Jo, D.; Song, J.; Kim, Y.; and Kim, J.-J. 2025. FastKV: KV Cache Compression for Fast Long-Context Processing with Token-Selective Propagation. *arXiv preprint arXiv:2502.01068*.

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

Liu, R.; Bai, H.; Lin, H.; Li, Y.; Gao, H.; Xu, Z.; Hou, L.; Yao, J.; and Yuan, C. 2024a. IntactKV: Improving Large Language Model Quantization by Keeping Pivot Tokens Intact. *arXiv preprint arXiv:2403.01241*.

Liu, X.; Tang, Z.; Chen, H.; Dong, P.; Li, Z.; Zhou, X.; Li, B.; Hu, X.; and Chu, X. 2025a. Can LLMs Maintain Fundamental Abilities under KV Cache Compression? *arXiv preprint arXiv:2502.01941*.

Liu, X.; Tang, Z.; Dong, P.; Li, Z.; Liu, Y.; Li, B.; Hu, X.; and Chu, X. 2025b. Chunkkv: Semantic-preserving kv cache compression for efficient long-context llm inference. *arXiv preprint arXiv:2502.00299*.

Liu, Y.; Lu, J.; Chen, Z.; Qu, C.; Liu, J. K.; Liu, C.; Cai, Z.; Xia, Y.; Zhao, L.; Bian, J.; et al. 2025c. Adaptivestep: Automatically dividing reasoning step through model confidence. *arXiv preprint arXiv:2502.13943*.

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

Paliotta, D.; Wang, J.; Pagliardini, M.; Li, K. Y.; Bick, A.; Kolter, J. Z.; Gu, A.; Fleuret, F.; and Dao, T. 2025. Thinking Slow, Fast: Scaling Inference Compute with Distilled Reasoners. *arXiv preprint arXiv:2502.20339*.

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*.

Rein, D.; Hou, B. L.; Stickland, A. C.; Petty, J.; Pang, R. Y.; Dirani, J.; Michael, J.; and Bowman, S. R. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Saxena, U.; Saha, G.; Choudhary, S.; and Roy, K. 2024. Eigen Attention: Attention in Low-Rank Space for KV Cache Compression. *arXiv preprint arXiv:2408.05646*.

Song, J.; Jo, D.; Kim, Y.; and Kim, J.-J. 2025. Reasoning Path Compression: Compressing Generation Trajectories for Efficient LLM Reasoning. *arXiv preprint arXiv:2505.13866*.

Sun, H.; Chang, L.-W.; Bao, W.; Zheng, S.; Zheng, N.; Liu, X.; Dong, H.; Chi, Y.; and Chen, B. 2024. ShadowKV: KV Cache in Shadows for High-Throughput Long-Context LLM Inference. *arXiv preprint arXiv:2410.21465*.

Tao, Q.; Yu, W.; and Zhou, J. 2024. Asymkv: Enabling 1-bit quantization of kv cache with layer-wise asymmetric quantization configurations. *arXiv preprint arXiv:2410.13212*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Xiao, G.; Tang, J.; Zuo, J.; Guo, J.; Yang, S.; Tang, H.; Fu, Y.; and Han, S. 2024. Duoattention: Efficient long-context llm inference with retrieval and streaming heads. *arXiv preprint arXiv:2410.10819*.

Xiong, Y.; Wu, H.; Shao, C.; Wang, Z.; Zhang, R.; Guo, Y.; Zhao, J.; Zhang, K.; and Pan, Z. 2024. Layerkv: Optimizing large language model serving with layer-wise kv cache management. *arXiv preprint arXiv:2410.00428*.

Yang, J. Y.; Kim, B.; Bae, J.; Kwon, B.; Park, G.; Yang, E.; Kwon, S. J.; and Lee, D. 2024. No token left behind: Reliable kv cache compression via importance-aware mixed precision quantization. *arXiv preprint arXiv:2402.18096*.

Yang, J. Y.; Kim, B.; Bae, J.; Park, G.; Kwon, B.; Yang, E.; Kwon, S. J.; and Lee, D. 2025. Don't Discard, but Keep It Small: Context-Preserving KV Cache Compression with Importance-Aware Adaptive Precision.

Yuan, J.; Liu, H.; Zhong, S.; Chuang, Y.-N.; Li, S.; Wang, G.; Le, D.; Jin, H.; Chaudhary, V.; Xu, Z.; et al. 2024. Kv cache compression, but what must we give in return? a comprehensive benchmark of long context capable approaches. *arXiv preprint arXiv:2407.01527*.

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, R.; Wang, K.; Liu, L.; Wang, S.; Cheng, H.; Zhang, C.; and Shen, Y. 2024a. LORC: Low-Rank Compression for LLMs KV Cache with a Progressive Compression Strategy. *arXiv preprint arXiv:2410.03111*.

Zhang, X.; Du, C.; Du, C.; Pang, T.; Gao, W.; and Lin, M. 2024b. SimLayerKV: A Simple Framework for Layer-Level KV Cache Reduction. *arXiv preprint arXiv:2410.13846*.

Zhang, Y.; Hu, Y.; Zhao, R.; Lui, J.; and Chen, H. 2024c. Unifying kv cache compression for large language models with leankv. *arXiv preprint arXiv:2412.03131*.

Zhang, Z.; Sheng, Y.; Zhou, T.; Chen, T.; Zheng, L.; Cai, R.; Song, Z.; Tian, Y.; Ré, C.; Barrett, C.; et al. 2023. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36: 34661–34710.