🚑 ER-KV: EVICTION RISK-AWARE KV CACHE COMPRESSION FOR EFFICIENT LONG REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reasoning language models generate long chain-of-thought sequences, causing KV-cache memory to grow substantially during decoding. KV-cache eviction mitigates this cost, but deciding which tokens to retain requires accounting for both future queries and the attention-output error caused by token removal. Existing methods address either future attention allocation or eviction-induced attention-output error, but do not jointly account for both when making eviction decisions. We propose **Eviction Risk-Aware KV Cache Compression (ER-KV)**, which ranks cached tokens by their expected eviction-induced attention-output error over future queries. ER-KV models the future-query distribution with a Gaussian fitted to observed pre-RoPE query statistics. To estimate eviction risk, which depends nonlinearly on each query, it samples virtual queries to represent this distribution, applies RoPE at multiple future positions, and aggregates the resulting eviction errors. Cholesky factorization reduces sampling overhead while yielding the same Gaussian sampling distribution as eigendecomposition. Under repeated pruning, ER-KV reduces attention-output MSE by 24.4% and output-distribution KL divergence by 48.6% relative to TriAttention. Across AIME24, AIME25, and MATH-500 with three reasoning models, ER-KV matches or outperforms TriAttention in eight of nine model–benchmark settings, improving Qwen3-8B accuracy on AIME24 from 42.1% to 49.2%.

1 INTRODUCTION

Reasoning language models generate chain-of-thought sequences spanning tens of thousands of tokens to solve challenging problems in mathematics, coding, and scientific reasoning (DeepSeek-AI, 2025; Yang et al., 2025). During decoding, the key-value (KV) cache stores past keys and values to avoid recomputation, but its memory footprint grows linearly with sequence length and can exceed that of model weights during long reasoning (Wang et al., 2025). KV-cache eviction mitigates this cost by retaining tokens important for preserving reasoning accuracy under a fixed memory budget. Many existing methods estimate this importance using historical attention patterns, assuming that tokens receiving greater attention in the past will remain important to future queries (Zhang et al., 2023; Liu et al., 2023; Li et al., 2024).

Although historical attention provides a useful estimate, a cached token's importance ultimately depends on future queries, whose information needs may change as reasoning progresses (Hu et al., 2025; Zhang et al., 2026). For example, when a model abandons an unsuccessful solution approach, tokens from that attempt may remain highly ranked by accumulated attention despite having little relevance to subsequent steps, causing historical attention-based methods to retain them at the expense of tokens needed later. Recent methods address this limitation by modeling future query behavior to estimate the attention that cached tokens will receive during subsequent decoding. Expected Attention fits a Gaussian approximation to the future-query distribution using recent query statistics and applies the Gaussian moment-generating function to compute expected unnormalized attention scores; these scores are then normalized and combined with value magnitude to estimate token importance (Devoto et al., 2025). LookaheadKV instead uses lightweight learned lookahead modules to predict future attention-based importance without generating the future continuation (Ahn et al., 2026). TriAttention uses offline-calibrated pre-RoPE query statistics to estimate position-dependent KV importance across multiple future RoPE offsets (Mao et al., 2026).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

These approaches anticipate future attention allocation, but do not directly evaluate the attention-output error that removing a token would cause under future queries. This distinction matters because a token's attention weight alone does not determine the consequences of eviction: the resulting error also depends on its value vector and the renormalization of attention over the remaining tokens. Although Expected Attention incorporates value magnitude, its score estimates a token's contribution rather than the output change induced by removal. CAOTE directly measures eviction-induced attention-output error using the current query during decoding (Goel et al., 2025). Its eviction decisions therefore rely on observed attention rather than estimating eviction error over future queries. This leaves a gap between predicting future attention allocation and evaluating the consequences of eviction, motivating a criterion that measures expected eviction-induced attention-output error over future queries.

To address this gap, we propose Eviction Risk-Aware KV Cache Compression (ER-KV), which ranks cached tokens by their eviction risk, defined as the expected attention-output error caused by their removal over future queries. This criterion accounts jointly for how future queries attend to cached tokens and how evicting those tokens changes the attention output, prioritizing tokens whose removal would cause greater output distortion. ER-KV models the future-query distribution with a Gaussian fitted to observed pre-RoPE queries. However, converting this distribution into an eviction-risk score requires evaluating eviction error, which depends nonlinearly on each query. We therefore sample virtual queries to represent the modeled future-query distribution, apply RoPE at multiple future positions, and average the resulting eviction errors. However, generating these queries requires the principal components and their corresponding variances to be extracted from the empirical covariance matrix, necessitating an eigendecomposition that introduces substantial overhead. To reduce this cost, we use Cholesky factorization, which we find yields the same Gaussian sampling distribution with substantially less factorization time. Our experimental results demonstrate that ER-KV outperforms all existing baselines, including TriAttention, improving Qwen3-8B accuracy on AIME24 from 42.1% to 49.2%.

Our main contributions are:

- We develop an eviction-risk criterion that estimates the attention-output error caused by token removal over future queries, using virtual queries sampled from observed pre-RoPE query statistics and evaluated at multiple future positions.
- We reduce the computational overhead of virtual-query generation through Cholesky factorization, which yields the same Gaussian sampling distribution as eigendecomposition with substantially less factorization time.
- We demonstrate that ER-KV reduces attention-output MSE by 24.4% and output-distribution KL divergence by 48.6% relative to TriAttention under repeated pruning. Across three reasoning models and three benchmarks, ER-KV matches or outperforms TriAttention in eight of nine settings, including an improvement in Qwen3-8B accuracy on AIME24 from 42.1% to 49.2%.

2 Related Work and Background

Attention-Based KV-Cache Eviction. Many KV-cache eviction methods estimate token importance from attention patterns observed during decoding. H2O (Zhang et al., 2023) retains heavy-hitter tokens that accumulate high attention scores, while Scissorhands (Liu et al., 2023) leverages the persistence of attention-based importance across decoding steps. SnapKV (Li et al., 2024) identifies important KV positions using attention patterns within a recent observation window and performs head-specific cache compression. More recently, R-KV (Cai et al., 2025) extends attention-based selection to long reasoning by additionally accounting for redundancy among cached tokens.

Future-Aware KV Importance Estimation. Expected Attention (Devoto et al., 2025) models future queries as Gaussian, motivated by the approximately Gaussian distribution observed in query activations. During decoding, it estimates the distribution statistics from a recent buffer of activations and uses the Gaussian moment-generating function (MGF) to compute the expected unnormalized attention to each cached key in closed form. The resulting attention estimate is combined with value magnitude for KV scoring. LookaheadKV (Ahn et al., 2026) instead takes a learned approach,

using lightweight lookahead modules to predict future attention-based importance without explicitly generating the future response. TriAttention (Mao et al., 2026) exploits the concentration of queries and keys in the pre-RoPE space. Using offline-calibrated, frequency-wise query statistics together with the cached keys, it estimates position-dependent KV importance across multiple future RoPE offsets.

Output-Aware and Eviction-Based KV Importance. Attention-based scores do not directly measure the change in the attention output caused by evicting a cached token. VATP (Guo et al., 2024) incorporates value-vector magnitude into attention-based importance to better account for each token's contribution to the attention output. CAOTE (Goel et al., 2025) measures the attention-output error caused by removing a cached token, accounting for the renormalization of the remaining attention weights. We build on this formulation and use the resulting attention-output error as our measure of *eviction damage*.

For a query q, let the attention output be

$$O(q) = \sum_j p_j(q) v_j, \tag{1}$$

where $p_j(q)$ is the attention weight assigned to token j and v_j is its value vector. If token i is removed, the remaining attention weights are renormalized, yielding

$$O_i'(q) = \frac{O(q) - p_i(q) v_i}{1 - p_i(q)}. \tag{2}$$

We define the eviction damage of token i as

$$D_i(q) = \frac{1}{d_v} \left\| O(q) - O_i'(q) \right\|_2^2 = \left(\frac{p_i(q)}{1 - p_i(q)} \right)^2 \frac{1}{d_v} \left\| v_i - O(q) \right\|_2^2, \tag{3}$$

where d_v is the dimension of the value and attention-output vectors.

This provides an eviction-based measure of token importance for an observed query. ER-KV combines the eviction-based and future-aware perspectives. It exploits the alignment between past and future pre-RoPE query geometry to construct virtual queries from the mean and covariance of observed queries. ER-KV then evaluates eviction damage for these virtual queries at multiple future positions and aggregates the resulting scores.

3 ER-KV

3.1 Evicition Risk and Pre-RoPE Query Geometry

3.1.1 Eviction Risk

A retained KV token can be reused by many subsequent queries. Therefore, tokens that would cause large eviction damage for future queries should be preserved. Let t denote the current decoding position, and let $\mathcal{F}_t = \{q_{t+1}, \ldots, q_T\}$ denote the remaining future queries until the end of generation. We define the eviction risk of token i as its expected eviction damage over these future queries:

$$R_i = \frac{1}{|\mathcal{F}_t|} \sum_{q \in \mathcal{F}_t} D_i(q). \tag{4}$$

At eviction time, however, the queries in $\mathcal{F}_t$ are unavailable, so Eq. 4 cannot be evaluated directly. We therefore ask whether the observed past queries contain geometric structure that remains relevant to future queries.

3.1.2 Past–Future Pre-RoPE Geometry

We analyze queries before rotary positional embedding (RoPE) (Su et al., 2024) to separate query-space geometry from position-dependent rotations. Let (μ_P, Σ_P) and (μ_F, Σ_F) denote the empirical

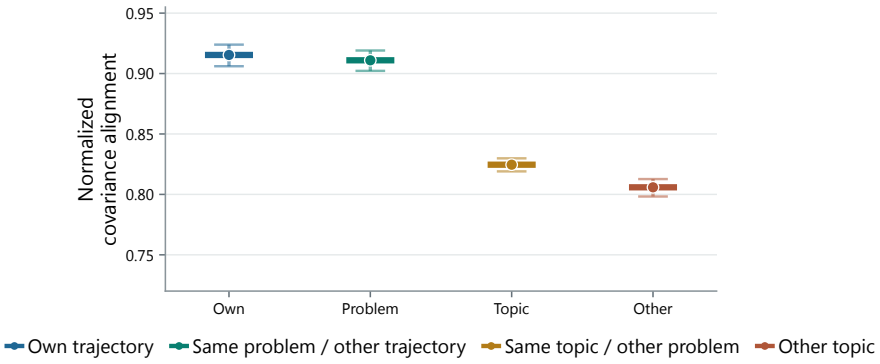

Figure 1: **Past–Future pre-RoPE covariance alignment.** At 50% decoding progress, we compute normalized covariance alignment between the observed past pre-RoPE queries and the remaining future pre-RoPE queries. Past queries from the same trajectory show the highest alignment, with lower alignment for queries from different problems or topics.

mean and covariance of the past and future pre-RoPE queries, respectively. Each covariance is computed after centering its corresponding queries by their own empirical mean.

We eigendecompose the two covariance matrices as

$$\Sigma_P = U_P \Lambda_P U_P^\top, \qquad \Sigma_F = U_F \Lambda_F U_F^\top, \tag{5}$$

where λ_i^P and λ_j^F denote the corresponding eigenvalues, and u_i^P and u_j^F denote the eigenvectors. We normalize the eigenvalues by the Frobenius norm of each covariance:

$$\alpha_i = \frac{\lambda_i^P}{\|\Sigma_P\|_F}, \qquad \beta_j = \frac{\lambda_j^F}{\|\Sigma_F\|_F}. \tag{6}$$

We then measure Past–Future similarity using normalized covariance alignment:

$$A_{\text{cov}} = \sum_{i,j} \alpha_i \beta_j \left({u_i^P}^\top u_j^F \right)^2. \tag{7}$$

The term $({u_i^P}^\top u_j^F)^2$ measures the directional alignment between the i-th past and j-th future covariance directions, while $\alpha_i \beta_j$ assigns greater weight to directions carrying larger variance. Thus, A_{cov} measures variance-weighted directional alignment between the centered past and future query geometries.

We evaluate the covariance alignment on Qwen3-8B over AIME24. As shown in Figure 1, the observed past queries from the same trajectory have the highest alignment with the remaining future queries, and this pattern remains consistent across decoding progress (Appendix A). The observed high alignment indicates that the dominant directions of query variation in the past remain relevant to future queries. This motivates using the observed past query geometry to construct virtual queries for estimating eviction risk over future queries.

3.2 Eviction-Risk Estimation

ER-KV estimates the eviction risk of cached KV tokens using the geometry of observed past queries. As illustrated in Figure 2, it constructs virtual pre-RoPE queries from the observed past query geometry, evaluates them at multiple future RoPE offsets, and aggregates their eviction damage for KV-cache pruning.

3.2.1 Window-Based Pruning and Online Query Statistics

Our eviction score measures the effect of removing a single KV token. Recomputing the scores after every individual eviction, however, would incur substantial overhead. We therefore perform batched

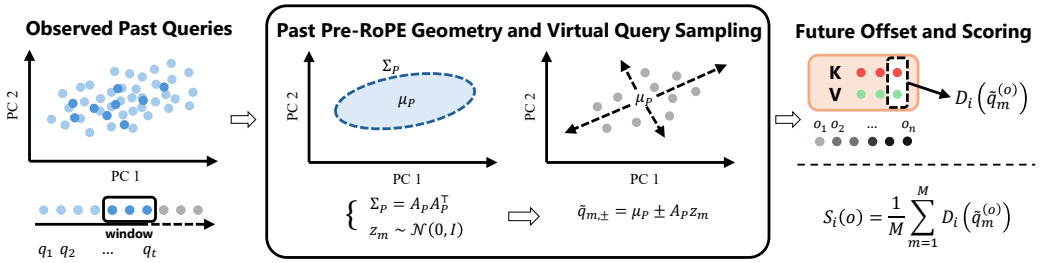

Figure 2: **Overview of ER-KV.** ER-KV summarizes the geometry of observed past pre-RoPE queries and uses it to construct virtual queries. The virtual queries are evaluated at multiple future RoPE offsets to estimate the eviction damage of cached KV tokens. The resulting scores are aggregated across query heads and future offsets and used for window-based KV-cache pruning.

pruning once every $\beta = 128$ generated tokens. At each pruning point, if the cache exceeds the target budget B, we compute the scores for the current cache and retain the top-B tokens for each KV head, evicting the remaining tokens in a single pruning step. Following R-KV (Cai et al., 2025), we refer to each β-token interval as a pruning window. This window-based procedure reduces the frequency of scoring and eviction.

During decoding, ER-KV maintains the cumulative mean and covariance of the observed pre-RoPE queries for each query head. The queries newly observed in each pruning window are merged into the current statistics using a centered pairwise covariance update (Chan et al., 1983).

3.2.2 Virtual Queries from Past Geometry

Motivated by the alignment between past and future query geometry, we construct virtual pre-RoPE queries from the empirical mean and covariance of the observed queries. Let μ_P and Σ_P denote these statistics. We draw $M/2$ standard Gaussian vectors and construct M virtual queries using antithetic pairs:

$$\widetilde{q}_{m,\pm} = \mu_P \pm A_P z_m, \qquad z_m \sim \mathcal{N}(0, I), \qquad A_P A_P^\top = \Sigma_P. \tag{8}$$

The antithetic construction makes the empirical mean of the virtual queries exactly μ_P.

Since our analysis characterizes query geometry through its principal directions and associated variances, eigendecomposition provides a direct construction of the covariance factor:

$$\Sigma_P = U_P \Lambda_P U_P^\top, \qquad A_P^{\mathrm{EVD}} = U_P \Lambda_P^{1/2}. \tag{9}$$

This explicitly constructs virtual queries along the principal directions of query variation with scales determined by their corresponding eigenvalues. Computing a full eigendecomposition at every pruning step, however, introduces substantial overhead.

ER-KV therefore uses a Cholesky-based factorization in practice:

$$\Sigma_P = L_P L_P^\top, \qquad A_P^{\mathrm{Chol}} = L_P. \tag{10}$$

This avoids explicitly computing the principal directions while providing a covariance factor for the virtual-query construction. The relationship between the EVD- and Cholesky-based constructions is derived in Appendix B.

We denote the resulting M virtual queries by $\{\widetilde{q}_m\}_{m=1}^{M}$ and use them to estimate eviction risk.

3.2.3 Position-Aware Eviction Scoring

A cached KV token may be attended to by queries at different future positions, so its eviction damage depends on the position of the future query. We therefore apply RoPE to each virtual pre-RoPE query at multiple future offsets. At the current decoding position t, the m-th virtual query at future offset o is given by

$$\widetilde{q}_m^{(o)} = \mathcal{R}_{t+o} \widetilde{q}_m, \tag{11}$$

where $\mathcal{R}_{t+o}$ denotes the RoPE rotation at position $t + o$.

For each query head, we estimate the eviction risk of cached KV token i at future offset o by averaging over the M virtual queries:

$$S_i(o) = \frac{1}{M} \sum_{m=1}^{M} D_i\left(\widetilde{q}_m^{(o)}\right), \tag{12}$$

where $D_i(\cdot)$ is the eviction damage defined in Eq. 3.

We evaluate Eq. 12 at eight logarithmically spaced future offsets, $\mathcal{O} = \{4^r\}_{r=0}^{7}$, covering both near- and long-range future positions. This yields an offset-specific eviction score for each cached KV token.

3.2.4 Grouped-Query Attention

Under GQA (Ainslie et al., 2023), each KV head is shared by multiple query heads. Let $\mathcal{H}_g$ denote the set of query heads sharing KV head g, and let $S_{h,i}(o)$ denote the score in Eq. 12 computed for query head h.

Because eviction scores can have different scales across query heads, we normalize the scores within each head before aggregation. Let $\mu_h(o)$ and $\sigma_h(o)$ denote the mean and standard deviation of $\{S_{h,i}(o)\}_i$ over cached KV tokens. We define

$$\widehat{S}_{h,i}(o) = \frac{S_{h,i}(o) - \mu_h(o)}{\sigma_h(o)}. \tag{13}$$

For each KV head, we then aggregate the normalized scores using

$$S_{g,i}(o) = \max_{h \in \mathcal{H}_g} \widehat{S}_{h,i}(o). \tag{14}$$

This prevents differences in score scale across query heads from dominating the aggregation, while assigning a high score to tokens that are important to any associated query head.

We finally average across future offsets:

$$S_{g,i} = \frac{1}{|\mathcal{O}|} \sum_{o \in \mathcal{O}} S_{g,i}(o). \tag{15}$$

4 Experiments

4.1 Experimental Setup

Models and Benchmarks. We evaluate ER-KV on three reasoning-capable language models: Qwen3-8B (Yang et al., 2025), DeepSeek-R1-Distill-Qwen-7B (DeepSeek-AI, 2025), and DeepSeek-R1-Distill-Llama-8B (DeepSeek-AI, 2025). These models span different model families and backbone architectures, allowing us to evaluate ER-KV across a range of reasoning models.

We evaluate mathematical reasoning on AIME 2024 (Mathematical Association of America, 2024) (30 problems), AIME 2025 (Mathematical Association of America, 2025) (30 problems), and MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024) (500 problems). AIME consists of challenging competition-level problems that often require extended multi-step reasoning, providing a natural setting for evaluating KV-cache compression during long generations. MATH-500 covers a broader range of mathematical topics and difficulty levels, complementing AIME with a larger and more diverse set of reasoning problems.

Baselines. We compare ER-KV against Full Attention and representative KV-cache compression methods, including SnapKV (Li et al., 2024), R-KV (Cai et al., 2025), Expected Attention (Devoto et al., 2025), and TriAttention (Mao et al., 2026).

Table 1: Reasoning performance on AIME24 and AIME25. Results for Full Attention, SnapKV, R-KV, and TriAttention are reported from prior work. For each model, all methods are compared under the same KV-cache budget. Among KV-cache compression methods, best results are in **bold**, and second-best results are underlined.

Method	AIME24				AIME25			
	Qwen3-8B	DS-Llama	DS-Qwen	**Avg.**	Qwen3-8B	DS-Llama	DS-Qwen	**Avg.**
Full Attention	57.1	50.4	43.8	50.4	40.8	31.4	34.2	35.5
SnapKV	34.6	5.0	34.6	24.7	20.0	6.7	25.0	17.2
R-KV	25.4	25.8	34.6	28.6	17.5	11.2	23.3	17.3
Expected Attention	23.75	4.2	35.4	21.1	22.92	5.0	26.2	18.0
TriAttention	42.1	33.8	**42.5**	39.5	32.9	19.6	30.0	27.5
ER-KV (Ours)	**49.2**	**38.8**	**42.5**	**43.5**	**38.8**	**22.1**	**31.3**	**30.7**

Table 2: Reasoning performance on MATH-500 with a KV-cache budget of 512. Results for Full Attention, SnapKV, R-KV, and TriAttention are reported from prior work. Among KV-cache compression methods, best results are in **bold**, and second-best results are underlined.

Method	Qwen3-8B	DS-Llama	DS-Qwen	Avg.
Full Attention	69.6	82.4	87.0	79.7
SnapKV	49.2	65.5	66.4	60.4
R-KV	46.4	76.9	71.6	65.0
Expected Attention	51.2	75.0	64.6	63.6
TriAttention	**56.0**	80.6	79.6	72.1
ER-KV (Ours)	55.0	**80.8**	**82.8**	**72.9**

Evaluation and Implementation Details. We conduct experiments on NVIDIA A100 40GB GPUs using bfloat16 precision and FlashAttention-2 (Dao, 2024). We use a maximum generation length of 32,768 tokens, with temperature 0.6 and top-p 0.95. For AIME 2024 and AIME 2025, we sample each problem eight times and report the average pass rate, whereas MATH-500 is evaluated with a single generation per problem. KV-cache pruning is performed every 128 generated tokens, reducing the cache to the specified budget at each pruning step. We use a default KV budget of 2,048 tokens and a budget of 512 tokens for DeepSeek-R1-Distill-Llama-8B and MATH-500, where generations are generally shorter. Unless otherwise specified, ER-KV uses Cholesky factorization with $M = 128$ virtual queries. Expected Attention uses adaptive head-wise KV budget allocation. Although its original implementation prunes every 512 tokens, we use a 128-token pruning interval to match the average number of cached tokens during decoding across methods. The total KV budget of each layer remains identical across methods, while Expected Attention is allowed to distribute this budget adaptively across heads.

4.2 Main Reasoning Results

Tables 1 and 2 report reasoning performance under constrained KV-cache budgets. ER-KV matches or outperforms TriAttention in eight of the nine model–benchmark settings. Averaged across the three models, ER-KV improves accuracy over TriAttention from 39.5% to 43.5% on AIME24 and from 27.5% to 30.7% on AIME25. On MATH-500, the two methods achieve similar average performance (72.9% vs. 72.1%).

Figure 3 further compares ER-KV and TriAttention across KV-cache budgets on DS-Qwen. The difference is largest at the tightest budget, $K = 512$, where ER-KV improves accuracy from 19.2% to 24.6% on AIME24 and from 11.7% to 19.6% on AIME25.

4.3 Do Virtual Queries Improve Eviction Risk Estimation?

We analyze ER-KV's virtual-query-based KV selection under repeated pruning using 11 Qwen3-8B AIME24 trajectories with at least 16K continuation tokens. For all methods, we replay the same

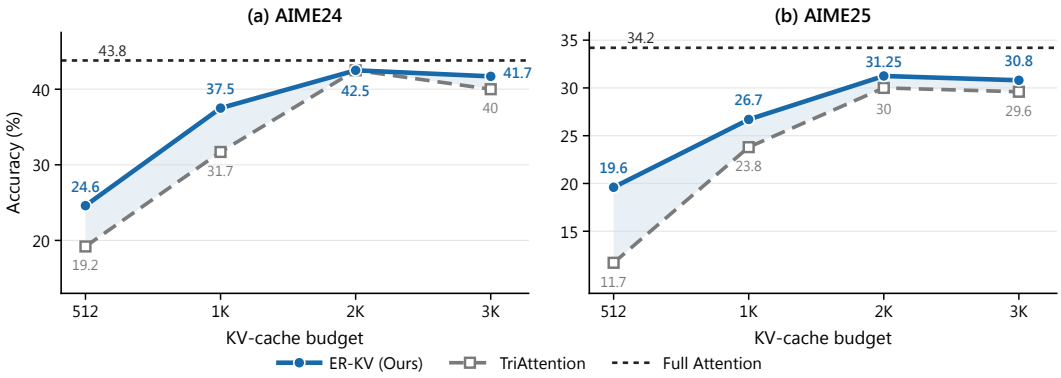

Figure 3: **Reasoning performance across KV-cache budgets.** Accuracy of DS-Qwen on AIME24 and AIME25 as the KV-cache budget varies from 512 to 3K tokens. ER-KV matches or outperforms TriAttention at every evaluated budget on both benchmarks.

continuation tokens generated by the dense model, while each compressed branch independently maintains and updates its own hidden states and KV cache.

We use two complementary metrics. First, *attention-output MSE* measures the quality of the retained KV set. For each method, we apply its retained KV positions to the corresponding dense queries, keys, and values and measure the resulting attention-output error. Evaluating all methods with the same dense activations isolates differences in KV selection from state drift in the compressed branches. Second, we measure $\mathrm{KL}(p_{\mathrm{Dense}} \| p_{\mathrm{Compressed}})$ using the actual compressed branches to quantify how closely the compressed output distribution follows that of the dense model under repeated pruning.

Figures 4(a) and 4(b) compare ER-KV with controlled variants and existing eviction methods. Mean-only evaluates eviction damage at the empirical query mean. Identity Cov. constructs multiple virtual queries around the same mean using an identity covariance. Gaussian MGF uses an analytic Gaussian expectation to obtain a distribution-level score without explicitly evaluating individual virtual queries. ER-KV instead constructs virtual queries from the empirical mean and covariance and averages the resulting token-wise eviction-damage scores.

For attention-output MSE, Cholesky MC128 achieves 10.5% lower error than Mean-only, 8.6% lower error than Identity Cov., and 16.5% lower error than Gaussian MGF. It also achieves 24.4% lower MSE than TriAttention and 83.0% lower MSE than Expected Attention. In this evaluation, the full ER-KV construction therefore produces lower attention-output error than the controlled alternatives considered here.

A similar pattern is observed for output-distribution fidelity. Cholesky MC128 achieves 33.2% lower KL than Mean-only, 27.9% lower KL than Identity Cov., and 36.4% lower KL than Gaussian MGF. It also achieves 48.6% lower KL than TriAttention and 70.4% lower KL than Expected Attention. Together, these results indicate that ER-KV maintains lower attention-output error and output-distribution divergence under repeated pruning in this setting.

4.4 Quality–Throughput Trade-off

Figures 4(a,b) show the trade-off between fidelity and computational cost. At the same sampling budget, EVD generally achieves lower error than Cholesky, although the difference in output-distribution fidelity is relatively small. For example, at $M = 128$, the KL decreases from 0.05147 with Cholesky to 0.04889 with EVD.

This improvement comes with substantially higher computational cost. As shown in Figure 4(c), at $M = 128$, Cholesky takes 512 ms per pruning event, whereas EVD takes 3,175 ms, corresponding to a $6.2\times$ increase. The scoring costs are nearly identical (424.0 vs. 426.5 ms), while the factorization cost increases from 14.3 ms to 2,667.1 ms.

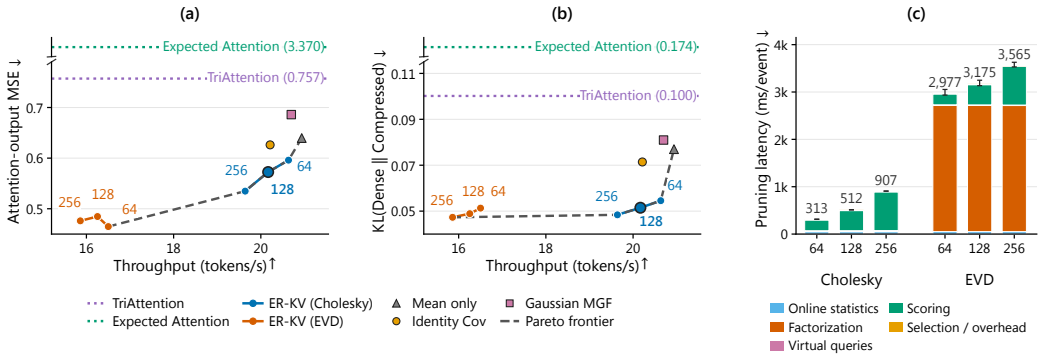

Figure 4: **Fidelity and efficiency under repeated pruning.** (a) Attention-output MSE and (b) $\mathrm{KL}(\mathrm{Dense}\|\mathrm{Compressed})$ versus decoding throughput at 16K. Mean-only, Identity Cov., and Gaussian MGF provide controlled comparisons for the virtual-query construction, while TriAttention and Expected Attention serve as eviction baselines. Numbers next to ER-KV points denote the number of virtual queries, and the dashed curve denotes the Pareto frontier. (c) Per-event pruning latency for Cholesky and EVD, decomposed into online statistics, factorization, virtual-query construction, scoring, and selection.

Table 3: Memory and throughput efficiency of Qwen3-8B at a 16K generation length on a single NVIDIA A100 40GB GPU. Throughput is measured at the maximum batch size supported by each setting.

Method	**KV Budget**	**Peak KV Saved**	**Max Batch**	**Throughput (tokens/s) ↑**	**Speedup ↑**
Full Attention	Full	0.0%	9	128.18	1.00×
ER-KV	2,048	86.8%	46	272.50	2.13×
ER-KV	1,024	93.0%	72	420.77	3.28×
ER-KV	512	96.1%	104	618.85	4.83×

We therefore use Cholesky with $M = 128$ virtual queries as the default ER-KV configuration. Increasing M to 256 further improves fidelity, but increases the per-event cost from 512 to 907 ms. We use $M = 128$ as the default to balance fidelity and pruning overhead.

4.5 Memory and Throughput Efficiency

We evaluate the memory and throughput efficiency of ER-KV on Qwen3-8B with a 16K generation length using a single NVIDIA A100 40GB GPU. At the default KV budget of 2,048, ER-KV achieves 38.8% accuracy on AIME25, within 2.0 percentage points of Full Attention (40.8%). In the 16K efficiency setting, the same budget reduces peak KV-cache memory by 86.8%, increases the maximum batch size from 9 to 46, and achieves a 2.13× throughput speedup over Full Attention. With a tighter 512-token budget, ER-KV reduces peak KV-cache memory by 96.1%, supports a maximum batch size of 104, and reaches 618.85 tokens/s, corresponding to a 4.83× speedup.

5 Conclusion

We studied KV-cache eviction for long reasoning as a future-dependent problem, where the importance of a cached token depends on queries that are unavailable at eviction time. We showed that past and future pre-RoPE queries share aligned directions of variation, motivating the use of observed past query geometry to construct virtual queries. ER-KV estimates the eviction risk of each cached token using these virtual queries at multiple future RoPE offsets and uses the resulting scores for fixed-budget KV-cache pruning. Across reasoning benchmarks and model families, ER-KV preserves reasoning performance under constrained cache budgets while improving over strong eviction baselines in most evaluated settings.

AI USE STATEMENT

We used generative AI tools to assist with methodology and experimental design, code implementation and debugging, result analysis, figure preparation, and manuscript drafting. All AI-assisted code, analyses, figures, and text were reviewed and verified by the authors. The authors take full responsibility for the methodology, experiments, results, claims, and final content of this work.

REPRODUCIBILITY STATEMENT

Sections 3–5 describe the method and evaluation protocol. Our implementation builds on the public TriAttention codebase. We plan to release our code and experiment configurations upon publication.

REFERENCES

Jinwoo Ahn, Ingyu Seong, Akhil Kedia, Junhan Kim, Hyemi Jang, Kangwook Lee, and Yongkweon Jeon. LookaheadKV: Fast and accurate KV cache eviction by glimpsing into the future without generation. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2603.10899.

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4895–4901. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.298. URL https://aclanthology.org/2023.emnlp-main.298/.

Zefan Cai, Wen Xiao, Hanshi Sun, Cheng Luo, Yikai Zhang, Ke Wan, Yucheng Li, Yeyang Zhou, Li-Wen Chang, Jiuxiang Gu, Zhen Dong, Anima Anandkumar, Abedelkadir Asi, and Junjie Hu. R-KV: Redundancy-aware KV cache compression for reasoning models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 60980–61005, 2025. doi: 10.52202/085713-2038. URL https://arxiv.org/abs/2505.24133.

Tony F. Chan, Gene H. Golub, and Randall J. LeVeque. Algorithms for computing the sample variance: Analysis and recommendations. *The American Statistician*, 37(3):242–247, 1983. doi: 10.1080/00031305.1983.10483115.

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2307.08691.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Alessio Devoto, Maximilian Jeblick, and Simon Jégou. Expected attention: KV cache compression by estimating attention from future queries distribution. *arXiv preprint arXiv:2510.00636*, 2025. URL https://arxiv.org/abs/2510.00636.

Raghavv Goel, Junyoung Park, Mukul Gagrani, Dalton Jones, Matthew Morse, Harper Langston, Mingu Lee, and Chris Lott. CAOTE: KV cache selection for LLMs via attention output error-based token eviction. *arXiv preprint arXiv:2504.14051*, 2025. URL https://arxiv.org/abs/2504.14051.

Zhiyu Guo, Hidetaka Kamigaito, and Taro Watanabe. Attention score is not all you need for token importance indicator in KV cache reduction: Value also matters. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 21158–21166. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.1178. URL https://aclanthology.org/2024.emnlp-main.1178/.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https:

//datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Junhao Hu, Wenrui Huang, Weidong Wang, Zhenwen Li, Tiancheng Hu, Zhixia Liu, Xusheng Chen, Tao Xie, and Yizhou Shan. RaaS: Reasoning-aware attention sparsity for efficient LLM reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 2577–2590. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.131. URL https://aclanthology.org/2025.findings-acl.131/.

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 *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-0722. URL https://arxiv.org/abs/2404.14469.

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 *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2305.20050.

Zichang Liu, Aditya Desai, Fangshuo Liao, Weitao Wang, Victor Xie, Zhaozhuo Xu, Anastasios Kyrillidis, and Anshumali Shrivastava. Scissorhands: Exploiting the persistence of importance hypothesis for LLM KV cache compression at test time. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-2279. URL https://arxiv.org/abs/2305.17118.

Weian Mao, Xi Lin, Wei Huang, Yuxin Xie, Tianfu Fu, Bohan Zhuang, Song Han, and Yukang Chen. TriAttention: Efficient long reasoning with trigonometric KV compression. In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=0tgzJK50Jz.

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

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

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. doi: 10.1016/j.neucom.2023.127063.

Zihao Wang, Bin Cui, and Shaoduo Gan. Squeezeattention: 2d management of KV-cache in LLM inference via layer-wise optimal budget. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2404.04793.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Haoyue Zhang, Hualei Zhang, Xiaosong Ma, Jie Zhang, and Song Guo. LazyEviction: Lagged KV eviction with attention pattern observation for efficient long reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 36335–36352. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1683. URL https://aclanthology.org/2026.acl-long.1683/.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-1506. URL https://arxiv.org/abs/2306.14048.