KEYS DECIDE ATTENTION, VALUES DECIDE CONTRIBUTION: VALUE-GATED KV CACHE EVICTION AT 20× COMPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Long documents are queried many times, so serving systems keep a document's key–value (KV) cache and compress it once, before any question arrives. Existing query-agnostic eviction methods rank cache entries by their keys alone and collapse under severe compression: with 5% of the cache retained, the best of them scores 34 on LongBench passage retrieval, against 100 for the full cache. Key-only scores ignore half of attention. Keys decide whether an entry is attended; values decide what an attended entry contributes, and a distinctive key with a small projected value contributes almost nothing. We propose VG-Evict (Value-Gated Eviction), a training-free rule that multiplies KeyDiff's normalized key-geometry score by the normalized magnitude of each entry's output-projected value, with a small floor and a positional reserve, at cost linear in context length. At 5%/7.5% retention, VG-Evict scores 50.5/70.5 on Qwen2.5-7B, 16.5/11.5 points above Compactor and 7.5/7.0 above CriticalKV applied to KeyDiff, and gains 14.0/17.0 points over Compactor on Mistral-7B; every paired bootstrap interval excludes zero. Controlled ablations attribute the gain to token-aligned value information: gating adds 18.0/23.5 points over KeyDiff at a matched reserve, shuffling the gates among tokens erases the gain, and a single ranking stage matches the best of CriticalKV's two-stage configurations without a stage-ratio hyperparameter. One compressed cache also serves a whole session: on 100 RULER documents at 10% retention it answers 97.5% of 400 independent questions with a tenth of the KV memory. The gains concentrate on passage identification and sparse lookup; for free-form answer extraction, leverage-score selection remains preferable. Code: https://anonymous.4open.science/status/VG-Evict-Anonymous-Code-706D.

1 INTRODUCTION

Reading is expensive and questions are cheap. Once a language model has processed a long document, everything it needs from that document sits in the document's key–value (KV) cache, and each later question is answered from the cache without reading the document again. The cache is the price of this reuse. It grows linearly with document length and, for long inputs, it is the dominant memory cost of serving, so it has to be compressed.

The right moment to compress is just after reading and before any question arrives, so that one compact cache serves every request that follows. That timing rules out the signal most eviction methods rely on. Accumulated or windowed attention scores (Zhang et al., 2023; Li et al., 2024; Oren et al., 2024; Feng et al., 2024) identify the entries a query attends to, but the query does not exist yet. Query-agnostic methods therefore score entries from the context alone: KeyDiff by how far a key points from the mean key (Park et al., 2025), Compactor by approximate leverage scores (Chari & Van Durme, 2025), Expected Attention by the attention a typical future query would pay (Devoto et al., 2025), and KVzip by how well the context can be reconstructed (Kim et al., 2025). They hold up at moderate budgets but not at severe ones. Retaining 5% of the cache on LongBench passage retrieval, where the model must name which of 30 paragraphs matches a description, KeyDiff

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.

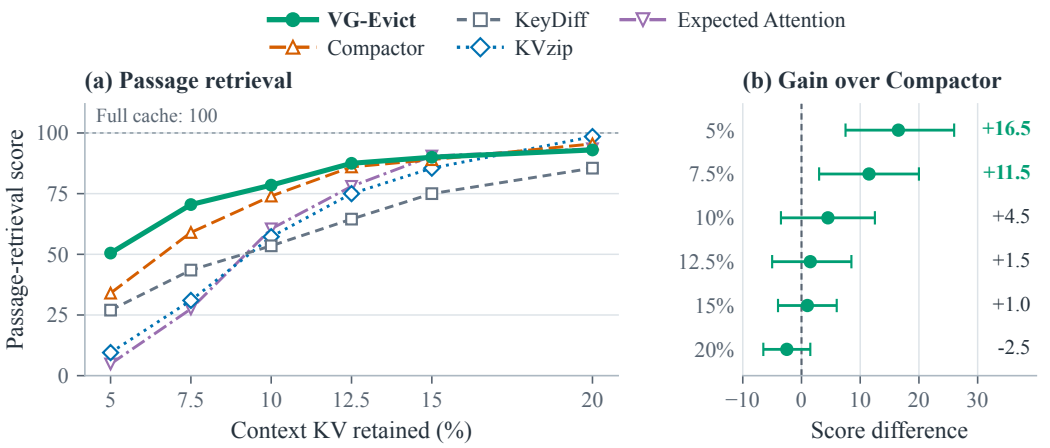

Figure 1: LongBench passage retrieval with Qwen2.5-7B (200 examples) across six budgets. Right: VG-EVICT minus Compactor with paired 95% intervals; rows indicate KV retention. Full cache scores 100. Table 1 adds CriticalKV(KeyDiff) at the two lowest budgets.

scores 27.0, Compactor 34.0, KVzip 9.5, and Expected Attention 4.8, against 100 for the full cache. Twenty-fold compression is where the memory savings are largest, and it is where current methods break.

We argue that they break because they score only half of what an attention head does. Keys set the attention weights; values are what those weights combine (Eq. 1). A cache entry can have a maximally distinctive key and still contribute almost nothing to the output if its projected value is small, while an entry with an unremarkable key can carry much of the output through a large value. A key-only score cannot tell the two apart, and when 19 of every 20 entries must go, that difference decides whether the entries that carry the document survive. The value half is available before any question arrives. Reweighting an attention score by value magnitude is known to help when the query is available (Guo et al., 2024; Feng et al., 2026); whether value magnitude can stand in for the attention signal when no query exists, and at budgets where key-only selection collapses, has not been studied.

We therefore make value magnitude a gate on key-based selection. VG-EVICT (Value-Gated Eviction)[1] computes, for every entry, KeyDiff's geometric key score and the L_1 norm of the entry's output-projected value, normalizes both within the layer and KV head, and ranks entries by the product of the key score and the value score plus a small floor. Four initial and 64 recent positions are reserved within the budget. The rule needs no training, calibration data, or reconstruction pass, and costs time linear in context length.

We test the rule where the gap is widest, at 5% and 7.5% retention, then ask where its gain comes from and how far it reaches.

- **Large gains where other methods fail.** On 200 LongBench passage-retrieval examples with Qwen2.5-7B, VG-EVICT scores 50.5 at 5% retention and 70.5 at 7.5%, 16.5 and 11.5 points above Compactor and 23.5 and 27.0 above KeyDiff (Figure 1). Against CriticalKV's value weighting applied to KeyDiff, the gains are 7.5 and 7.0 points. On Mistral-7B the gains over Compactor are 14.0 and 17.0 points. All paired bootstrap intervals exclude zero (Section 3.2).
- **The gain comes from token-aligned value information.** With the positional reserve held fixed, value gating adds 18.0/23.5 points over KeyDiff at 5%/7.5%; without the reserve it adds 16.5/21.5. Randomly permuting the value gates among tokens returns accuracy to the KeyDiff level, and raw value norms already recover most of the gain before output projection (Section 3.3).

[1] https://anonymous.4open.science/status/VG-Evict-Anonymous-Code-706D

- **A single stage beats two.** CriticalKV first protects entries by key score, then weights the rest by value. Its accuracy falls as more budget goes to the key-only stage, from 51.0 to 41.5 at 5%. VG-EVICT sits at the top of that range with no stage ratio, and its floor and centering choices move accuracy by at most 1.5 points (Section 3.4).
- **A compressed cache serves a session.** Compressing 100 RULER documents once to 10% and asking four independent questions each, VG-EVICT answers 97.5% correctly and completes 90 of 100 sessions, against 77 for KeyDiff and 86 for Compactor, with 87 MiB of KV instead of 873 MiB (Section 3.6).

Section 4 maps the boundary of these gains. Value gating excels when a few retained entries must identify a passage or hold a sparse record; for extracting free-form answers, Compactor's leverage scores retain more answer tokens and lead.

2 VALUE-GATED EVICTION

2.1 COMPRESSING BEFORE QUESTIONS ARRIVE

We process the n context tokens once to build the document KV cache (prefill). We retain $B = \lfloor bn \rfloor$ entries per layer and KV head, where b is the retained fraction; $b = 0.05$ keeps one entry in twenty. Throughout the paper, budget refers to this retained fraction. Questions are appended after compression and cannot influence the selection.

The compressed cache stays in memory between questions. After each answer we remove the states added for that question and its answer, so the next question starts from the same compressed cache (Section 3.6).

2.2 KEYS DECIDE ATTENTION; VALUES DECIDE CONTRIBUTION

For one attention head, let k_i and v_i denote the cached key and value at position i. Given a query vector q, the head contributes

$$o(q) = \sum_{i=1}^{n} \alpha_i(q) W_o v_i, \qquad \alpha_i(q) = \frac{\exp(q^\top k_i/\sqrt{d_h})}{\sum_j \exp(q^\top k_j/\sqrt{d_h})}, \tag{1}$$

where d_h is the head dimension and W_o is the corresponding block of the output projection. The keys determine the attention weights $\alpha_i(q)$; the projected values are what those weights combine. An entry with a distinctive key and a near-zero projected value can win attention and still add nothing to $o(q)$. Before the question arrives $\alpha_i(q)$ cannot be computed, but which entries would matter if attended can be. We therefore rank entries by key distinctiveness and gate that ranking by value magnitude; Figure 2 shows how the gate reorders a key-only ranking.

2.3 THE EVICTION SCORE

Both signals are computed within a layer and KV head. The key score follows KeyDiff (Park et al., 2025): normalize each key to unit length, average to obtain a reference direction μ, and score keys by how far they point away from it:

$$\mu = \frac{1}{n} \sum_{j=1}^{n} \frac{k_j}{\|k_j\|_2}, \qquad g_i = -\cos(k_i, \mu). \tag{2}$$

A larger g_i means a more distinctive key.

The value score is the magnitude of each value after output projection. Under grouped-query attention, several attention heads share one KV head, so we average the projected L_1 norms over the corresponding output-projection blocks:

$$w_i = \frac{1}{|\mathcal{H}|} \sum_{h \in \mathcal{H}} \left\| W_o^{(h)} v_i \right\|_1, \tag{3}$$

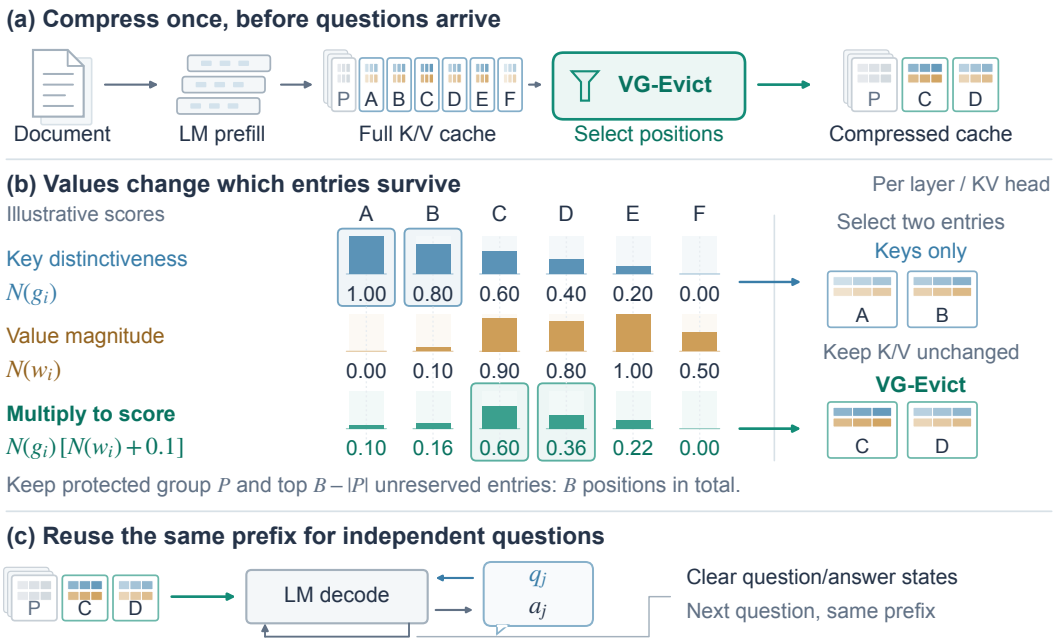

Figure 2: Overview of VG-EVICT. (a) Compress the document before questions arrive. (b) Value gating changes which entries survive a key-based ranking; the selected K/V pairs are stored unchanged. (c) Reuse the compressed prefix for independent questions. Scores are illustrative.

where $\mathcal{H}$ is the set of attention heads sharing the KV head. This is the quantity CriticalKV uses for value weighting (Feng et al., 2026); Section 3.3 shows that the raw norm $\|v_i\|_1$ already captures most of the benefit.

To combine the two signals we min–max normalize each across the context positions of the KV head and multiply:

$$N(z_i) = \frac{z_i - \min_j z_j}{\max_j z_j - \min_j z_j + 10^{-6}}, \qquad s_i = N(g_i)\big(N(w_i) + 0.1\big). \tag{4}$$

The value term gates the key score: a large value raises an entry's priority but cannot rescue a key score of zero. The gate changes only the selection; the stored K/V vectors are untouched.

Why a product. Evicting a set E of entries changes the head output in Eq. 1 by $\Delta(q) = \sum_{i\in E} \alpha_i(q) W_o v_i$ before the softmax renormalizes, so

$$\|\Delta(q)\|_1 \le \sum_{i\in E} \alpha_i(q)\ \|W_o v_i\|_1 . \tag{5}$$

CriticalKV derives a bound of this form and minimizes it with the attention weights of an observed query (Feng et al., 2026). Before the query exists, $\alpha_i(q)$ is unavailable, but the structure of the bound survives: each entry's worst-case contribution is a product of an attention factor and a value factor, so for a fixed budget the bound is minimized by discarding the entries with the smallest products, not the smallest sums and not the smallest of either factor alone. VG-EVICT instantiates this with a query-agnostic surrogate for the attention factor. KeyDiff's observation that distinctive keys receive high attention makes $N(g_i)$ a nonnegative, monotone proxy for α_i; the min–max map is the simplest such proxy, and because a positive per-head scale never changes a top-k selection, only its zero point matters. Under grouped-query attention the heads sharing a KV head share the key and hence the surrogate, and summing Eq. 5 over them gives $|\mathcal{H}|\, N(g_i)\, w_i$, which is why w_i averages the projected norms over the group. The floor contributes $0.1\, N(g_i)$, a key-only tie-breaker among entries whose projected values are near zero; the bound predicts it should matter little, and Section 3.4 confirms it.

Relation to VATP and CriticalKV. Both multiply an attention score by a value norm, $\|v_i\|_1$ in VATP and $\|W_o v_i\|_1$ in CriticalKV (Guo et al., 2024; Feng et al., 2026), and both therefore presup-

pose that attention weights from the query, or from an observation window, are available when the cache is compressed. Our proposition is different: when no query exists, value magnitude has to substitute for the attention signal rather than refine it. Eq. 5 says this is legitimate once the attention factor is replaced by a query-agnostic surrogate, and the experiments measure how much of the low-budget loss it recovers. CriticalKV's kvpress wrapper can also be run on a query-agnostic base scorer (NVIDIA, 2024), in two stages: a fraction r of the budget by the base score alone, then base score times w_i. With KeyDiff as the base scorer this is the closest existing rule to ours (Appendix A.1). Under Eq. 5, every entry admitted by its key alone is chosen by the wrong objective, and Section 3.4 shows that CriticalKV improves as that stage shrinks, with VG-EVICT matching its best setting without the hyperparameter.

2.4 SELECTING WITHIN A FIXED BUDGET

We protect the first four and last 64 context positions, following the initial-token retention of attention-sink and value-aware methods (Xiao et al., 2024; Guo et al., 2024). Let P contain these reserved positions, clipped to the context length. The remaining slots go to the highest-scoring unreserved entries:

$$S = P \cup \mathrm{Top}_{B-|P|}\{s_i : i \notin P\}. \tag{6}$$

Reserved positions count toward B, and all tested budgets satisfy $B \geq |P|$. Selected entries keep their K/V vectors and original rotary positions. The reserve sizes and the value floor are fixed in every experiment except the ablations that name them.

Scoring costs one output projection per entry and one mean over keys: linear in context length, with no attention matrix, no learned scoring parameters, no calibration data, and no reconstruction pass.

3 EXPERIMENTS

3.1 SETUP

The primary benchmark is LongBench English passage retrieval (Bai et al., 2024). Each example contains 30 paragraphs and an abstract describing one of them; the model must return that paragraph's number. We evaluate Qwen2.5-7B-Instruct on all 200 examples with the benchmark prompt and metric, compressing the document before the question in every run. The budget sweep covers 5%, 7.5%, 10%, 12.5%, 15%, and 20% retention.

Baselines are KeyDiff, Compactor, KVzip, and Expected Attention through their kvpress implementations (Park et al., 2025; Chari & Van Durme, 2025; Kim et al., 2025; Devoto et al., 2025; NVIDIA, 2024), plus CriticalKV applied to KeyDiff, written CKV(KD), which we evaluate at the two lowest budgets together with matched-reserve controls (Table 1). Mistral-7B-Instruct-v0.3 provides a second model family at the same budgets (Table 3). All runs use greedy decoding, BF16, FlashAttention-2, and fixed scoring parameters (Appendix A).

Differences are paired by example and reported with 95% percentile bootstrap intervals from 20,000 resamples. For shared-cache evaluations the resampling unit is the document, which preserves dependence among its questions. Appendix B lists every score and interval.

3.2 PASSAGE RETRIEVAL AT 20× COMPRESSION

Figure 1 shows the six-budget sweep. At 5% retention VG-EVICT scores 50.5, against 34.0 for Compactor, 27.0 for KeyDiff, 9.5 for KVzip, and 4.8 for Expected Attention. The 16.5-point gain over Compactor has a paired interval of [7.5, 26.0]. At 7.5% the scores are 70.5 versus 59.0, a gain of 11.5 [3.0, 20.0]. Over KeyDiff, whose key score VG-EVICT inherits, the gains are 23.5 [16.0, 31.0] at 5%, 27.0 [19.5, 34.5] at 7.5%, and 25.0 [18.0, 32.0] at 10%.

The advantage is largest exactly where the budget is tightest. VG-EVICT leads at every budget up to 12.5%; from 15% upward VG-EVICT, Compactor, KVzip, and Expected Attention converge within a few points, and KVzip, which spends a reconstruction pass, leads at 20%. Value gating matters in the regime where most of the cache must go.

Table 1: LongBench passage retrieval at the two lowest budgets (Qwen-7B; 200 examples). All methods retain $B = \lfloor bn \rfloor$ entries per layer and KV head; P reserves the first four and last 64 positions within B. CKV(KD) is CriticalKV applied to KeyDiff with $r = 0.5$. Gains are in points with paired 95% bootstrap intervals.

Method	5% KV	7.5% KV
KeyDiff	27.00	43.50
Compactor (original configuration)	34.00	59.00
CKV(KD)	43.00	63.50
KeyDiff with P	32.50	47.00
CKV(KD) with P	46.50	67.00
VG-EVICT (original, with P)	50.50	70.50
VG-EVICT minus CKV(KD)	7.50 [0.50, 14.50]	7.00 [1.00, 13.00]
VG-EVICT minus CKV(KD) with P	4.00 [−2.00, 10.00]	3.50 [−2.00, 9.00]
VG-EVICT minus KeyDiff with P	18.00 [11.00, 25.00]	23.50 [16.50, 30.50]

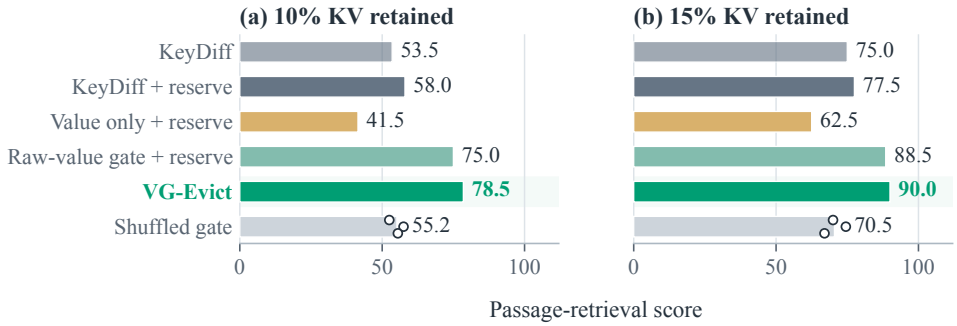

Figure 3: Selection-rule ablations at 10%/15% retention on 200 Qwen-7B examples. All variants except plain KeyDiff use the same reserve. Shuffled gates show the three-seed mean with individual seeds as circles. Table 8 adds 5%/7.5%.

Table 1 adds the value-aware comparator. CKV(KD) in its published configuration, with stage ratio $r = 0.5$ and no reserve, scores 43.0/63.5 at 5%/7.5%; VG-EVICT gains 7.5 [0.5, 14.5] and 7.0 [1.0, 13.0] points over it. Giving CKV(KD) the same reserve P raises it to 46.5/67.0, still 4.0 and 3.5 points below VG-EVICT.

3.3 WHERE THE GAIN COMES FROM

The gate, with and without the reserve. With reserve P held fixed, KeyDiff scores 32.5/47.0 at 5%/7.5% and VG-EVICT scores 50.5/70.5, a gate gain of 18.0 [11.0, 25.0] and 23.5 [16.5, 30.5]. Removing the reserve from both rules leaves the gate gain almost unchanged at 16.5 [9.5, 23.5] and 21.5 [14.5, 28.5]. The reserve itself contributes 5.5 and 7.0 points at 5% and 3.5 and 5.5 at 7.5% (Appendix A.2, Table 5). The gate and the reserve make separate contributions, and the gate's is several times larger. At 10%/15%, the gate adds 20.5 [14.0, 27.5] and 12.5 [7.5, 18.0] over reserved KeyDiff (Figure 3).

Shuffled gates. To test whether the gain comes from the association between each token and its own value magnitude, rather than from the gate's distribution, we randomly permute the scalar value scores among unreserved positions within each layer and KV head. This keeps the candidate set, the stored K/V vectors, and the distribution of gate values, and breaks only the token–gate alignment. Averaged over three seeds, shuffled gates score 30.2/46.0 at 5%/7.5%, no better than reserved KeyDiff (32.5/47.0) and 20.3 [13.5, 27.2] and 24.5 [17.8, 31.0] points below the aligned gate. At 10%/15% the shuffled scores are 55.2/70.5 against 78.5/90.0. The entire gain depends on each token keeping its own value score.

Table 2: Stage-ratio sweep and single-stage variants on the same 200 LongBench examples (Qwen-7B), all with reserve P within budget. c_i is the shifted product of Eq. 8; D_w is the value range plus 10^{-6}. CKV denotes CriticalKV(KeyDiff). The right columns give VG-EVICT minus each rule, in points with paired 95% bootstrap intervals; — marks contrasts not resampled.

	Score		VG-EVICT minus rule	
Selection rule	5% KV	7.5% KV	5% KV	7.5% KV
KeyDiff	32.50	47.00	18.00 [11.00, 25.00]	23.50 [16.50, 30.50]
CKV, $r = 0.75$	41.50	60.50	—	—
CKV, $r = 0.5$	46.50	67.00	4.00 [−2.00, 10.00]	3.50 [−2.00, 9.00]
CKV, $r = 0.25$	49.00	70.00	—	—
Single-stage product c_i ($r = 0$)	51.00	69.50	−0.50 [−4.50, 3.50]	1.00 [−2.50, 4.50]
No floor: $N(g_i)N(w_i)$	50.00	71.00	0.50 [−3.50, 4.50]	−0.50 [−4.00, 3.00]
No centering: $N(g_i)(w_i/D_w + 0.1)$	49.50	69.00	1.00 [−2.50, 4.50]	1.50 [−2.00, 5.00]
VG-EVICT	50.50	70.50		

Which value signal. Value magnitude alone is not a good selector: projected magnitude with the reserve scores 41.5 at 10%, below reserved KeyDiff at 58.0. It is the combination that works. Gating the key score by the raw norm $\|v_i\|_1$ already reaches 75.0 at 10% and 88.5 at 15%, 17.0 [10.5, 23.5] and 11.0 [6.0, 16.0] points over reserved KeyDiff. Output projection adds a further 3.5 and 1.5 points (Appendix C).

3.4 SINGLE-STAGE SELECTION VERSUS CRITICALKV'S TWO STAGES

CriticalKV protects a fraction r of the budget by key score alone before value weighting. Table 2 sweeps r with the reserve fixed. Accuracy at 5% rises monotonically as the key-only stage shrinks: 41.5 at $r = 0.75$, 46.5 at $r = 0.5$, 49.0 at $r = 0.25$, and 51.0 when the stage is removed entirely ($r = 0$, a single-stage product of the shifted key score and w_i). At 7.5% the trend is the same, from 60.5 at $r = 0.75$ to 70.0 and 69.5 at $r = 0.25$ and $r = 0$. Every entry admitted on its key alone displaces an entry that a value-aware ranking would have preferred, and the cost grows with the number of such entries. The default $r = 0.5$, which the two-stage design must tune, costs 4.5 [−1.0, 10.0] and 2.5 [−2.5, 7.5] points relative to $r = 0$, while still improving over reserved KeyDiff by 14.0 [7.5, 20.5] and 20.0 [13.0, 27.0].

VG-EVICT scores 50.5/70.5, matching the best two-stage setting and the single-stage product within a point at both budgets, with no stage ratio to tune. The single-stage variants behave as Eq. 5 predicts. The shifted product c_i, the no-floor rule $N(g_i)N(w_i)$, and the no-centering rule differ from VG-EVICT only in the zero point and floor of the attention surrogate, and they score 51.0/69.5, 50.0/71.0, and 49.5/69.0: every difference from VG-EVICT lies within ± 1.5 points with intervals that include zero. What separates these product rules from KeyDiff is 16 to 24 points; what separates them from each other is within noise. The design principle that carries the gain is to rank every candidate by the product of a key-based attention surrogate and value magnitude, and VG-EVICT is its simplest form.

3.5 A SECOND MODEL FAMILY

On Mistral-7B-Instruct-v0.3 at 5%/7.5% retention, VG-EVICT scores 36.5/52.0, against 22.5/35.0 for Compactor, 18.0/31.5 for KeyDiff, 11.0/40.5 for KVzip, and 29.0/43.5 for CKV(KD). The gains over Compactor are 14.0 [7.0, 21.0] and 17.0 [9.0, 25.0], over CKV(KD) 7.5 [1.5, 14.0] and 8.5 [2.0, 15.0], and all eight paired intervals against the four baselines exclude zero (Table 5). The full cache scores 98.5 on Mistral, so the task remains far from solved at these budgets for every method, but the ordering of methods and the size of the gate's advantage carry over from Qwen.

At the higher budgets of 15%/20%, VG-EVICT also exceeds KeyDiff and Compactor on Qwen2.5-14B and Mistral-7B (Table 3). KVzip leads there on both models, as on Qwen-7B at 20%; the advantage of value gating is a low-budget one.

Table 3: Passage retrieval on additional models, 200 examples per cell. CKV(KD) is CriticalKV applied to KeyDiff with $r = 0.5$ and no reserve. Table 5 gives the Mistral-7B paired intervals at 5%/7.5%; — marks cells not measured.

Model	Budget	KeyDiff	Compactor	KVzip	CKV(KD)	VG-EVICT
Mistral-7B	5%	18.00	22.50	11.00	29.00	36.50
	7.5%	31.50	35.00	40.50	43.50	52.00
Qwen-14B	20%	77.50	89.58	98.50	—	91.83
	15%	65.75	85.33	95.50	—	87.42
Mistral-7B	20%	59.00	63.00	97.00	—	71.00
	15%	53.50	45.00	94.00	—	62.00

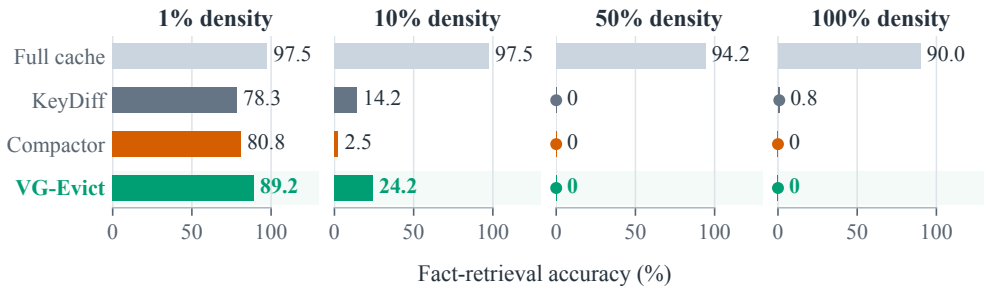

Figure 4: Fact retrieval at four nominal record densities with Qwen-7B and 10% KV retention (30 documents per setting, four questions each). Density is the construction target before tokenization and truncation.

3.6 SPARSE LOOKUP AND CACHE REUSE

Record density. We embed key–number records in background prose at four densities, with 30 documents per setting, four questions per document, and 10% retention of a prefix capped at 16K tokens (Appendix D.1). At 1% density, VG-EVICT answers 89.2% of questions, against 80.8% for Compactor, 78.3% for KeyDiff, and 97.5% for the full cache (Figure 4). At 10% density it answers 24.2%, against 2.5% and 14.2%. At the two highest densities every compressed method falls to zero while the full cache stays above 90%: no 10% cache can hold hundreds of independent answers. Value gating retains sparse records that the other rules lose; it cannot enlarge the budget.

Table 4: Shared-cache lookup on 100 RULER documents, four questions each, at 10% KV retention (Qwen-7B, 16K). Complete sessions answer all four questions correctly.

Method	Accuracy (%)	Complete sessions	KV payload (MiB)
Full cache	99.75	99/100	872.7
KeyDiff	92.75	77/100	87.2
Compactor	96.50	86/100	87.2
VG-EVICT	**97.50**	**90/100**	87.2

One cache, four questions. We compress each of the first 100 documents of the 16K RULER multi-query subset once (Hsieh et al., 2024) and ask four independent key–number questions from the same retained prefix, removing each question's states after its answer. At 10% retention VG-EVICT answers 390 of 400 questions (97.5%, versus 99.75% for the full cache) and completes all four questions on 90 of 100 documents, against 77 for KeyDiff and 86 for Compactor (Table 4). Per-question accuracy improves over KeyDiff by 4.75 points [2.25, 7.50] and the complete-session rate by 13 points [5, 21]. The resident cache holds 87.2 MiB of BF16 keys and values instead of 872.7 MiB (Eq. 7, Appendix A). The comparison covers the three methods that physically evict entries; KVzip's implementation masks entries while retaining full-length tensors.

4 WHERE VALUE GATING HELPS, AND WHERE IT DOES NOT

The gains above share a structure: a small set of entries carries the answer, whether the paragraph that matches a description or the record that holds a number. Natural-answer extraction is different. We build 20 SQuAD-derived documents of about 16K tokens, each embedding eight source paragraphs in background prose, and ask the eight original questions independently from one compressed cache (Rajpurkar et al., 2016). Every compressed method loses most of the full-cache score (80.1 F1); Compactor leads at all three budgets, scoring 27.8 F1 at 7.5% retention against 12.6 for VG-EVICT, a paired difference of 15.2 [8.0, 22.3] (Appendix D.2).

A retention diagnostic explains the reversal. At 7.5% retention, VG-EVICT keeps 7.1% of the annotated answer-span tokens, averaged over layers and KV heads, the same fraction as uniform selection with the same reserve; Compactor keeps 11.7%. The value gate favors entries with large projected values, which identifies a passage and preserves sparse records, but it does not preferentially keep the tokens of a free-form answer; leverage scores do. Compactor also leads on scientific-document and code probes, and KVzip leads a 13-task RULER average on which VG-EVICT trails Compactor by about two points (Appendix E). At severe budgets, then, value gating suits workloads that identify passages or look up sparse facts, and leverage-score selection suits free-form answer extraction.

5 RELATED WORK

Attention-based eviction (Zhang et al., 2023; Oren et al., 2024; Li et al., 2024; Feng et al., 2024; Cai et al., 2024) needs the query and cannot compress a document before its questions are known. Query-agnostic methods score entries from the context alone: KeyDiff by key geometry (Park et al., 2025), key-norm strategies by key magnitude (Devoto et al., 2024), Compactor by approximate leverage scores over keys (Chari & Van Durme, 2025), Expected Attention by an estimate of future attention (Devoto et al., 2025), and KVzip, KVzap, and Fast KVzip by context reconstruction or scores learned from it (Kim et al., 2025; Jégou & Jeblick, 2026; Kim et al., 2026). All of them rank entries without their values.

Value information enters eviction through VATP and CriticalKV (Guo et al., 2024; Feng et al., 2026), both of which multiply an attention score by a value norm and therefore compress with the query in hand. Neither asks whether value magnitude can replace the attention signal when the query is absent, which is the question here (Section 2). CriticalKV's kvpress wrapper can be applied to a query-agnostic base scorer in two stages (NVIDIA, 2024), so CriticalKV over KeyDiff is the closest existing method and our main comparator; Section 3.4 shows that removing its key-only stage is what recovers the low-budget accuracy. Protecting initial tokens follows the attention-sink observation (Xiao et al., 2024; Sun et al., 2024).

6 CONCLUSION

Query-agnostic KV eviction has ranked cache entries by their keys alone, ignoring the values that decide what an attended entry contributes. VG-EVICT gates a key-geometry score by projected value magnitude in one normalized product. At 5% and 7.5% retention it raises LongBench passage retrieval by 16.5 and 11.5 points over Compactor on Qwen2.5-7B and by 14.0 and 17.0 on Mistral-7B, and controlled ablations attribute the gain to each token's own value magnitude rather than to the reserve or the exact form of the rule. One compressed cache serves a session of independent questions at a tenth of the memory. The gains concentrate where a few entries must carry the answer; extending them to free-form answer extraction is the natural next step.

LIMITATIONS

The low-budget comparisons cover two 7B models on one benchmark, and the attribution controls cover Qwen-7B only. Free-form answer extraction at these budgets favors Compactor (Section 4), and the reuse protocol assumes a fixed document with independent questions.

AI USE STATEMENT

Generative AI tools were used in four ways. (1) Writing: to aid and polish the prose; all scientific claims, numbers, and interpretations were written or verified by the authors. (2) Literature retrieval and discovery, such as finding related work; every cited paper was read and checked by the authors. (3) Implementation: parts of the evaluation and analysis code, including evaluation runners, bootstrap scripts, and figure scripts, were drafted with AI assistance and then reviewed, tested, and corrected by the authors; the core selection rule and the CriticalKV integration were implemented and verified by the authors. (4) Data organization: AI assistance was used to collate and format experimental records, such as aggregating saved predictions into tables; all reported results come from the authors' runs, and every number in the paper was checked against the source records by the authors.

REPRODUCIBILITY STATEMENT

Section 2 defines the score and selection rule, and Appendix A gives the fixed configuration, prompt handling, and bootstrap procedure. Appendix A.1 specifies the CriticalKV integration and every control, and Appendix D describes the density, natural-answer, and additional-task probes. The anonymous repository at https://anonymous.4open.science/status/VG-Evict-Anonymous-Code-706D contains the implementation, evaluation runners, probe inputs, aggregate records, saved predictions, and the figure and bootstrap scripts.

REFERENCES

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 (ACL)*, 2024.

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. In *arXiv preprint arXiv:2406.02069*, 2024.

Vivek Chari and Benjamin Van Durme. Compactor: Calibrated Query-Agnostic KV Cache Compression with Approximate Leverage Scores. *arXiv preprint arXiv:2507.08143*, 2025.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of NAACL-HLT*, 2021.

Alessio Devoto, Yu Zhao, Simone Scardapane, and Pasquale Minervini. A Simple and Effective L_2 Norm-Based Strategy for KV Cache Compression. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

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.

Yuan Feng, Junlin Lv, Yukun Cao, Xike Xie, and S. Kevin Zhou. Ada-KV: Optimizing KV Cache Eviction by Adaptive Budget Allocation for Efficient LLM Inference. In *arXiv preprint arXiv:2407.11550*, 2024.

Yuan Feng, Junlin Lv, Haoyu Guo, Yukun Cao, S. Kevin Zhou, and Xike Xie. CriticalKV: Optimizing KV Cache Eviction from an Output Perturbation Perspective. In *International Conference on Machine Learning (ICML)*, 2026.

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. *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the Real Context Size of Your Long-Context Language Models? In *Conference on Language Modeling (COLM)*, 2024.

Simon Jégou and Maximilian Jeblick. KVzap: Fast, Adaptive, and Faithful KV Cache Pruning. *arXiv preprint arXiv:2601.07891*, 2026.

Jang-Hyun Kim, Jinuk Kim, Sangwoo Kwon, Jae W. Lee, Sangdoo Yun, and Hyun Oh Song. KVzip: Query-Agnostic KV Cache Compression with Context Reconstruction. *arXiv preprint arXiv:2505.23416*, 2025.

Jang-Hyun Kim, Dongyoon Han, and Sangdoo Yun. Fast KVzip: Efficient and Accurate LLM Inference with Gated KV Eviction. *arXiv preprint arXiv:2601.17668*, 2026.

Yucheng Li, Huiqiang Jiang, Qianhui Wu, Xufang Luo, Surin Ahn, Chengruidong Zhang, Amir H. Abdi, Dongsheng Li, Jianfeng Gao, Yuqing Yang, and Lili Qiu. SCBench: A KV Cache-Centric Analysis of Long-Context Methods. In *International Conference on Learning Representations (ICLR)*, 2025.

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 (NeurIPS)*, 2024.

Jiawei Liu, Jia Le Tian, Vijay Daita, Yuxiang Wei, Yifeng Ding, Yuhan Katherine Wang, Jun Yang, and Lingming Zhang. RepoQA: Evaluating long context code understanding. *arXiv preprint arXiv:2406.06025*, 2024.

NVIDIA. kvpress: KV Cache Compression Methods Implemented in the Hugging Face Transformers Ecosystem. https://github.com/NVIDIA/kvpress, 2024.

Matanel Oren, Michael Hassid, Nir Yarden, Yossi Adi, and Roy Schwartz. Transformers are Multi-State RNNs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Junyoung Park, Dalton Jones, Matthew J. Morse, Raghavv Goel, Mingu Lee, and Chris Lott. KeyDiff: Key Similarity-Based KV Cache Eviction for Long-Context LLM Inference in Resource-Constrained Environments. *arXiv preprint arXiv:2504.15364*, 2025.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, 2016. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264/.

Mingjie Sun, Xinlei Chen, J. Zico Kolter, and Zhuang Liu. Massive Activations in Large Language Models. In *Conference on Language Modeling (COLM)*, 2024.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient Streaming Language Models with Attention Sinks. In *International Conference on Learning Representations (ICLR)*, 2024.

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. H_2O: Heavy-Hitter Oracle for Efficient Generative Inference of Large Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

A IMPLEMENTATION AND STATISTICAL DETAILS

The standard VG-EVICT configuration uses a 10^{-6} normalization stabilizer, a value floor of 0.1, and a reserve of four initial and 64 recent tokens; named ablations change only their stated component. Reserved entries count toward the retained budget. Scores and selected positions are computed separately for each layer and KV head, with projected value magnitude averaged over the attention heads that share a KV head. The same indices select K and V, and original rotary positions are preserved.

We split LongBench prompts at the context field, compress the document together with the fixed instruction prefix, and then append the question. The context cap is 32,768 tokens and passage-retrieval generation is capped at 32 tokens. RULER uses its released context, query, reference, and answer-prefix fields. Baselines use their kvpress implementations; Compactor uses increased factorization precision and extended regularization retries, and Expected Attention uses its AdaKV wrapper with epsilon 10^{-2}. Scoring parameters are fixed across tasks, and the CriticalKV integration is specified in Appendix A.1.

Shared-cache evaluation creates a fresh `DynamicCache` for each document and method. After decoding each question we crop the appended states back to the retained prefix length, and question positions start at the original uncompressed prefix length. For L layers, H_{KV} KV heads, head dimension d_h, and B retained entries per head, the resident BF16 payload is

$$M_{\mathrm{KV}} = 4LH_{\mathrm{KV}}d_hB \quad \text{bytes}, \tag{7}$$

where the factor four accounts for K and V at two bytes each; the payload excludes temporary question and answer states, weights, and allocator overhead. KVzip evicts logically through attention masks while retaining full-length tensors, so the physical-cache reuse comparison covers KeyDiff, Compactor, and VG-EVICT.

Intervals are percentile bootstraps from 20,000 draws (NumPy seed 42), pointwise and without multiple-comparison adjustment. Passage-retrieval intervals resample the 200 paired examples; reuse intervals resample documents and keep their questions together. Shuffled-gate seeds are interventions on the same examples, so we average the three seed scores within each example before resampling; the interval is conditional on those seeds. The Expected Attention export at 5% and the density and 15%/20% cross-model runs lack complete per-example predictions, so those cells are reported without paired intervals.

A.1 CRITICALKV(KEYDIFF) INTEGRATION AND CONTROLS

Integration. We wrap the KeyDiff scorer (`KeyDiffPress`) in the CriticalKV wrapper (`CriticalKVPress`) with a first-stage ratio of 0.5 and $\epsilon_C = 10^{-4}$. The public wrapper first protects entries by the base score, then fills the budget by projected-value weighting. Because KeyDiff scores are signed, multiplying them directly by w_i would penalize large values on negative-scored entries, so our patch shifts the second-stage score to be nonnegative:

$$t_i = g_i - \min_j g_j, \qquad c_i = (t_i + \epsilon_C)w_i, \tag{8}$$

with the minimum taken over the full prefix within each layer and KV head. The shift leaves the first-stage ranking unchanged. A separate compatibility patch sets a missing `config.head_dim` to hidden size divided by the number of attention heads and does not alter scores. Projected norms and GQA averaging use the same utility as VG-EVICT.

Budget and reserve controls. All compared methods retain the same integer count $B = \lfloor bn \rfloor$ in every layer and KV head. Let R be either empty (the published CriticalKV configuration) or P (the matched-reserve control), $U = \{1, \ldots, n\} \setminus R$, and $M = B - |R|$. For stage ratio r,

$$A_r = \mathrm{Top}_m\{g_i : i \in U\}, \qquad S_C = R \cup A_r \cup \mathrm{Top}_{M-m}\{c_i : i \in U \setminus A_r\}, \tag{9}$$

where $m = \lfloor rM \rfloor$ for nonzero r and $m = 0$ at $r = 0$. Without a reserve, $r = 0.5$ agrees with the archived first-stage count $\lfloor bn/2 \rfloor$. Reserved and first-stage entries are counted once, and stage two fills exactly the remaining slots. Score statistics are computed over the full prefix before excluding reserved candidates. The $r = 0$ diagnostic bypasses stage one explicitly rather than relying on a zero-ratio code path in the wrapper.

Single-stage variants. Comparing the single-stage product c_i with VG-EVICT tests the gate transformation as a whole; the no-floor variant uses $N(g_i)N(w_i)$, and the no-centering variant uses $N(g_i)(w_i/D_w + 0.1)$ with $D_w = \max_j w_j - \min_j w_j + 10^{-6}$, which preserves the scale of the floor. The key-normalization denominator is constant within a layer and head and cannot by itself change a top-k ranking, so we do not attribute a gain to each normalization step separately. All Qwen-7B controls use the same 200 examples, checkpoint, prompt split, truncation, generation, and scoring as the primary evaluation. Each contrast is a conditional comparison within the tested configuration, not an additive decomposition of the total gain.

A.2 GATE AND RESERVE ATTRIBUTION AT THE HEADLINE BUDGETS

The four-cell diagnostic crosses key-only scoring g_i against gated scoring s_i with $R \in \{\varnothing, P\}$. The no-reserve gate replaces P by the empty set in Eq. 6 and keeps normalization and floor. All cells use the same B and full-prefix statistics; within each R, candidate positions and deterministic tie handling agree. With $A_{K,R}$ and $A_{V,R}$ the mean scores, we define

$$\Delta_{G,R} = A_{V,R} - A_{K,R}, \qquad \Delta_{R,u} = A_{u,P} - A_{u,\varnothing}, \qquad I = \Delta_{G,P} - \Delta_{G,\varnothing}, \tag{10}$$

for $u \in \{K, V\}$. Table 8 reports the four cells and Table 5 the contrasts. The interaction I is 1.5 [-4.0, 7.0] at 5% and 2.0 [-3.0, 7.0] at 7.5%: the gate's gain is nearly the same with and without the reserve. Bootstrap draws for I resample the same example indices across all four cells. Three shuffled-gate seeds at each low budget retain P and permute gates only among unreserved entries.

Mistral-7B-Instruct-v0.3 uses the 200 LongBench passage-retrieval examples, the released metric, the existing Mistral prompt and tokenizer configuration, a 32,768-token context cap, and greedy decoding up to 32 tokens. Within the model, all methods receive identical tokenized prefixes and questions, and budgets are $B = \lfloor bn \rfloor$ per layer and KV head with original rotary positions. Token counts differ from Qwen because the tokenizers differ. KeyDiff, Compactor, KVzip, and VG-EVICT keep their existing configurations; CKV(KD) uses the patch above with $r = 0.5$ and no reserve.

Table 5: Paired contrasts at 5%/7.5% retention, in points with 95% bootstrap intervals ($n = 200$ examples). Δ and I are defined in Eq. 10. The shuffled-gate contrast averages three seeds within each example before resampling.

Mistral-7B full-cache score (100% retained): 98.50		
Contrast [95% interval]	5% KV	7.5% KV
Mistral-7B: additional-model validation		
VG-EVICT minus KeyDiff	18.50 [11.50, 25.50]	20.50 [13.00, 28.50]
VG-EVICT minus Compactor	14.00 [7.00, 21.00]	17.00 [9.00, 25.00]
VG-EVICT minus KVzip	25.50 [18.00, 33.00]	11.50 [4.00, 19.00]
VG-EVICT minus CriticalKV(KeyDiff)	7.50 [1.50, 14.00]	8.50 [2.00, 15.00]
Qwen-7B: gate and reserve attribution		
Gate with P, $\Delta_{G,P}$ (primary)	18.00 [11.00, 25.00]	23.50 [16.50, 30.50]
Gate without P, $\Delta_{G,\varnothing}$	16.50 [9.50, 23.50]	21.50 [14.50, 28.50]
Reserve with key-only scores, $\Delta_{R,K}$	5.50 [1.50, 9.50]	3.50 [-0.50, 7.50]
Reserve with the gate, $\Delta_{R,V}$	7.00 [2.00, 12.00]	5.50 [1.00, 10.00]
Reserve–gate interaction, I	1.50 [-4.00, 7.00]	2.00 [-3.00, 7.00]
VG-EVICT minus mean shuffled gate (P)	20.33 [13.50, 27.17]	24.50 [17.83, 31.00]

B COMPLETE PASSAGE-RETRIEVAL RESULTS

Tables 6 and 7 report the five-method, six-budget sweep on Qwen-7B. Table 1 adds CKV(KD) at 5% and 7.5%.

Table 6: LongBench passage retrieval across all six budgets, using Qwen-7B and 200 examples per cell. Columns give the fraction of context KV retained.

Method	5%	7.5%	10%	12.5%	15%	20%
KeyDiff	27.00	43.50	53.50	64.50	75.00	85.50
VG-Evict	50.50	70.50	78.50	87.50	90.00	93.00
Compactor	34.00	59.00	74.00	86.00	89.00	95.50
KVzip	9.50	31.00	57.25	75.00	85.50	98.50
Expected Attention	4.83	27.50	60.50	78.00	90.50	93.50

Table 7: Paired 95% bootstrap intervals for VG-Evict minus each baseline, in points. Expected Attention at 5% has no interval because its saved predictions are incomplete.

Budget	KeyDiff	Compactor	KVzip	Expected Attn.
5%	[16.0, 31.0]	[7.5, 26.0]	[33.0, 49.0]	—
7.5%	[19.5, 34.5]	[3.0, 20.0]	[31.0, 48.0]	[34.5, 51.5]
10%	[18.0, 32.0]	[-3.5, 12.5]	[13.0, 29.8]	[9.5, 26.5]
12.5%	[17.0, 29.5]	[-5.0, 8.5]	[5.5, 19.5]	[2.5, 16.5]
15%	[9.5, 21.0]	[-4.0, 6.0]	[-1.5, 10.5]	[-6.0, 5.0]
20%	[3.5, 12.0]	[-6.5, 1.5]	[-9.5, -1.5]	[-5.0, 4.0]

Table 8: Qwen-7B component and shuffled-gate scores on 200 passage-retrieval examples. Reserved variants keep the first four and last 64 positions within budget. Table 5 gives the paired contrasts at 5%/7.5%; — marks cells not measured.

Scoring configuration	5% KV	7.5% KV	10% KV	15% KV
KeyDiff ($K, \varnothing$)	27.0	43.5	53.5	75.0
KeyDiff + reserve (K, P)	32.5	47.0	58.0	77.5
VG-EVICT without reserve ($V, \varnothing$)	43.5	65.0	—	—
VG-EVICT (V, P)	50.5	70.5	78.5	90.0
Shuffled gate, seed 1	30.5	45.0	52.5	70.0
Shuffled gate, seed 2	28.0	48.5	57.5	74.5
Shuffled gate, seed 3	32.0	44.5	55.5	67.0
Projected value only + reserve	—	—	41.5	62.5
Geometry $\times$ raw value + reserve	—	—	75.0	88.5

C COMPONENT AND SHUFFLED-GATE SCORES

The positional reserve and budget are fixed across the protected variants. Raw-value gating improves over protected KeyDiff by 17.0 points [10.5, 23.5] at 10% retention and 11.0 [6.0, 16.0] at 15%; output projection adds 3.5 [-0.5, 8.0] and 1.5 [-2.5, 5.5] on top. Shuffled gates average 30.17/46.00 at 5%/7.5% and 55.17/70.50 at 10%/15%, against 50.50/70.50 and 78.50/90.00 for aligned gates.

D SUPPLEMENTARY EVALUATION PROTOCOLS

D.1 RECORD-DENSITY CONSTRUCTION

We use 30 seeds per density and four questions per document, for 120 documents in total. The four settings contain 8, 78, 393, and 787 records within approximately 47,600 to 55,100 characters of context. Keys have three alphabetic components and answers are seven-digit numbers. The tokenized prefix is capped at 16,384 tokens, and queries target records in the first 80% of character positions to limit truncation losses. Density labels refer to construction targets before tokenization. Generation is capped at 48 tokens, and an answer is correct when it contains the reference string. Figure 4 reports aggregate accuracy including the full-cache controls.

D.2 NATURAL-ANSWER RETRIEVAL AND ANSWER-POSITION RETENTION

We construct 20 development documents from SQuAD 1.1 training articles (Rajpurkar et al., 2016), shuffling article IDs with seed 42. Each contains eight source paragraphs embedded in background prose and one original question per paragraph, for 160 questions with their references and annotated answer offsets. Prefill length, including the chat prefix, ranges from 16,377 to 16,681 tokens. Each method compresses a document once and answers all eight questions independently, generating at most 48 tokens per answer. We compute normalized exact match (EM) and token F1 from the saved predictions, average over questions within each document, then over documents. Compactor gains 17.2 [10.4, 24.6], 15.2 [8.0, 22.3], and 12.6 [5.9, 19.7] F1 points over VG-EVICT at 10%, 7.5%, and 5% retention, with intervals that resample the 20 paired documents. This is a development probe rather than the official SQuAD evaluation.

Table 9: Natural-answer retrieval on 20 SQuAD-derived documents, eight questions each (Qwen-7B, approximately 16K). Cells show document-averaged EM / token F1 (%).

Method	10% KV	7.5% KV	5% KV
KeyDiff	10.00 / 17.83	7.50 / 15.95	8.75 / 16.73
Compactor	**21.25 / 32.03**	**18.75 / 27.76**	**13.12 / 21.04**
VG-EVICT	8.75 / 14.81	8.12 / 12.55	5.00 / 8.41
Full cache	63.12 / 80.10 (100% retained)		

To measure answer-position retention we take the top-k indices that gather K and V, map the first gold answer's annotated occurrence and its source paragraph to token positions through tokenizer and chat-prefix offsets, and average the retained fraction of the annotated set over layers and KV heads, then over questions and documents. At 7.5% retention, answer-span and source-paragraph retention are 6.95%/5.69% for KeyDiff, 11.74%/7.21% for Compactor, and 7.07%/5.93% for VG-EVICT. A selector that keeps the same $|P| = 68$ reserved entries and samples the remaining positions uniformly retains an expected span fraction of $(B_d - |P|)/(N_d - |P|)$ for document d, since annotated spans lie outside the reserved ends; this reference averages 7.11%. The VG-EVICT difference from it is -0.04 points [-0.94, 1.00], and from Compactor -4.67 [-6.09, -3.07]. The diagnostic tracks annotated positions; contextualized entries elsewhere may also encode an answer.

E ADDITIONAL TASK COMPARISONS

E.1 SCIENTIFIC-DOCUMENT AND CODE RETRIEVAL

The QASPER probe contains 25 papers and 104 questions (Dasigi et al., 2021). We insert paragraph identifiers, cap context at 16,384 tokens, and generate up to 96 tokens containing evidence identifiers and an answer. Evidence F1 takes the maximum set F1 over reference evidence sets using at most ten predicted identifiers; answer F1 uses normalized token overlap. The adapted SCBench/RepoQA probe contains 25 code contexts and 96 target functions (Li et al., 2025; Liu et al., 2024), with 32,768-token contexts and at most 256 generated tokens; name match checks the target function name and code F1 measures multiset overlap of identifiers and non-whitespace symbols. Both probes reuse a compressed context for independent questions, and their subsets and metrics differ from the official protocols. Table 10 reports both budgets with full-cache controls.

Table 10: Scientific-document and code retrieval with Qwen-7B: QASPER evidence/answer F1 and adapted RepoQA name match/code F1.

Method	KV	Evidence F1	Answer F1	Name match	Code F1
Full cache	100%	34.51	25.21	65.62	63.12
KeyDiff	20%	7.04	14.83	34.38	41.85
	10%	4.07	12.84	23.96	38.69
Compactor	20%	12.71	24.85	51.04	51.11
	10%	10.48	20.95	36.46	48.48
VG-EVICT	20%	9.35	17.43	42.71	49.02
	10%	4.97	11.83	32.29	46.81

E.2 BROADER RULER EVALUATION

Table 11 averages 13 RULER tasks with 50 examples each at 10% retention. CKV(KD) uses the patches of Appendix A.1 with $r = 0.5$ and no reserve; VG-EVICT keeps its reserve. VG-EVICT exceeds KeyDiff by 1.1 and 3.0 points at 16K and 32K and matches CKV(KD) within 2.2 points; Compactor is 2.2 and 1.6 points higher, and KVzip leads by a wide margin at both lengths. These are aggregates without paired intervals.

Table 11: RULER macro-average at 10% KV retention (Qwen-7B; 50 examples per task). CKV(KD) is patched CriticalKV(KeyDiff), $r = 0.5$, without a reserve. EA denotes Expected Attention.

Context	KeyDiff	VG-EVICT	CKV(KD)	Compactor	KVzip	EA
16K	63.46	64.58	62.42	66.81	79.44	23.69
32K	57.93	60.91	60.94	62.54	69.57	—