Minimal-Intervention KV Retention via Set-Conditioned Diversity

Anonymous authors
Paper under double-blind review

Abstract

KV-cache compression at small budgets is a crowded design space spanning cache representation, head-wise routing, compression cadence, decoding behavior, and within-budget scoring. We study seven mechanisms across these five families on long-form mathematical reasoning (MATH-500 (Hendrycks et al., 2021)) at budgets $b \in \{64, 128\}$, under an evaluation standard that tightened over the study and converged on matched mean cache with $n \geq 200$ on two distilled-reasoning models (Qwen-7B and Llama-8B variants of DeepSeek-R1-Distill (DeepSeek-AI, 2025)). All seven were rejected as catalogue directions, one on screening-grade evidence. We then propose α, a one-function modification to the TriAttention (Mao et al., 2026) retention scorer that replaces argmax-top-k with greedy facility-location-inspired selection under a V-space redundancy penalty controlled by a single weight λ. A pre-registered protocol tunes λ on a frozen development split and confirms on a disjoint held-out split; with $\lambda = 0.5$, α clears Bonferroni on two of the four (model, budget) cells (Qwen b=128 and Llama b=64), no cell is significantly negative, and the pre-registered Branch A triggers. The finding is asymmetric: the surviving mechanism was among the smallest tested, but minimality alone did not predict survival — two comparably small scoring modifications were also rejected — so what distinguished α was its set-conditioned selection rule, in which each retention decision depends on the already-retained set, rather than its size. The combined matched-memory, sympy-graded, held-out confirmation protocol is the evidence standard that made the asymmetry visible.

1 Introduction

The KV cache is the dominant memory cost of long-context language-model inference. At decode time, cached key and value tensors occupy memory linear in sequence length, layers, heads, and head dimension, and grow monotonically with generation. KV-cache compression—selecting a budget-bounded subset of cached tokens at each compression event (Li et al., 2024; Zhang et al., 2023; Liu et al., 2023; Xiao et al., 2024)—trades a small amount of accuracy for substantial memory savings. Below a threshold, however, compression begins to degrade reasoning accuracy visibly: at small budgets ($b \in \{64, 128\}$) on long-form reasoning workloads, published mechanisms cluster within a few accuracy points of one another, and each proposes a different intervention to push the frontier.

This paper asks a narrower question. Holding the model, budget, routing policy, compression cadence, and decoding loop fixed, *which axis of intervention actually moves accuracy at matched memory?* We test seven mechanisms across five families: **state** (which K/V representations are stored), **routing** (which heads or layers see which subset of the cache), **cadence** (when to compress and with what budget across decode steps), **decoding** (how the model emits text given that the cache is being compressed), and **scoring** (which existing tokens are kept under a fixed budget). The seven mechanisms are organized into a retrospective design-space catalogue (Section 3). They were gated sequentially, under a methodological standard that itself tightened over the course of the study and converged on the one later used for α: matched mean cache instrumented during decode, sympy-based grading on MATH-500 (Hendrycks et al., 2021), and $n \geq 200$ evaluation. Earlier gates were decided under that standard as it then existed; Section 3 flags which kills rest on screening-grade (n=50 or single-model) evidence. We then propose α, a small, set-conditioned scoring-axis modification, and

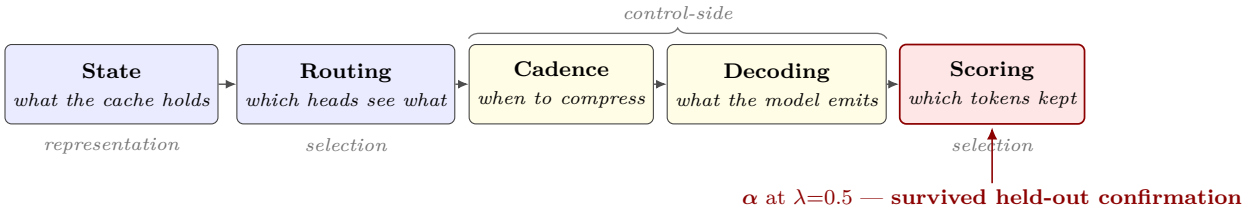

Figure 1: KV-retention pipeline at decode time: the five intervention surfaces examined in this study. State, Routing, and Scoring modify representation and selection over the cache; Cadence and Decoding are control-side interventions sharing a super-bracket. Seven mechanisms across these surfaces were tested and all rejected; the set-conditioned scoring intervention α at λ=0.5 (marked) is the only survivor. Family-by-family verdicts are in Table 1 (Section 3).

evaluate it under a pre-registered protocol with a frozen development split for hyperparameter selection and a disjoint held-out split for confirmation. The seven catalogue mechanisms were rejected; α survived. Figure 1 sketches the five-axis pipeline and marks where α sits.

Contributions. The design-space study (Section 3) catalogues seven mechanisms across the five families above, gives a falsified-mechanism verdict for each, and extracts five failure modes that recur across families. α (Section 4) is a one-function modification to the TriAttention (Mao et al., 2026) retention scorer that replaces argmax-top-k with a greedy facility-location-inspired selector under a V-space redundancy penalty controlled by a single weight λ; at $\lambda = 0$ the selector reduces bitwise to top-k. The pre-registered protocol (Section 4.2) is designed against these five failure modes: a frozen MD5-bucketed dev/confirm split, a λ-probe on the development split alone, and a Bonferroni-corrected joint test on the held-out confirmation split, with the decision rule and branch table committed in code before the confirmation experiment runs. The held-out result (Section 5) is that on MATH-500 with Qwen-7B and Llama-8B variants of DeepSeek-R1-Distill (DeepSeek-AI, 2025) at $b \in \{64, 128\}$, α clears Bonferroni on two of the four (model, budget) cells with no significantly negative cells and triggers the pre-registered Branch A; a post-confirmation head-to-head against a SnapKV-style windowed-attention baseline (Li et al., 2024) reproduces the same shape. Beyond the result, the positioning claim (Section 6) is that in this regime the scoring axis was the only intervention family to yield a survivor; minimality alone did not predict survival, since two comparably small scoring modifications were also rejected, and what set α apart was its set-conditioned selection rule — a property of the design space distinct from the accuracy headline itself.

Scope. The result is intentionally narrow. Our evaluation covers long-form mathematical reasoning with two distilled-reasoning models in the 7–8B-parameter range. We do not claim transfer to other workloads (LongBench (Bai et al., 2024), code generation, multi-document QA), to larger models, or to non-reasoning models without explicit re-evaluation. The pre-registered protocol covers the method only; the design-space study (Section 3) is retrospective and is presented as such throughout.

Why this regime is worth isolating. Most published KV-compression methods report accuracy curves at budgets where the baseline retention pipeline already handles the workload well. The pressure point is the smallest budgets that remain coherent at all—where every byte of cache is contested and small intervention surfaces compete for the same accuracy budget. Long-form mathematical reasoning is particularly demanding there because chain-of-thought generation (Wei et al., 2022) produces many decode-time steps, so compression triggers fire repeatedly and per-trigger errors accumulate; and because its closed-form answers admit sympy-based grading, which removes a formatting-noise floor that would otherwise dominate small-budget evaluation (Section 2 details the grading protocol).

2 Related Work

We organize prior work along the five axes introduced in Section 1 and conclude with the workload and evaluation methodology that distinguish our protocol from the published norm.

Scoring-family compression. The dominant published line of KV-cache compression operates on the scoring axis: given a per-token retention score, retain the top-k. H2O (Zhang et al., 2023) introduces the heavy-hitter hypothesis — a small set of attention-heavy tokens carries most of the model's future predictions and should be preserved. SnapKV (Li et al., 2024) uses end-of-prefill windowed attention to identify which tokens future queries are likely to attend to and then retains those tokens through generation. Scissorhands (Liu et al., 2023) reframes retention selection as a persistence-of-importance test, keeping tokens that score high across multiple recent attention windows. StreamingLLM (Xiao et al., 2024) keeps a small set of initial "attention sink" tokens alongside a recent window, exploiting the empirical observation that early tokens receive disproportionate attention mass independent of semantic centrality. Quest (Tang et al., 2024) introduces query-aware sparsity, tracking per-page min/max keys and estimating page criticality from the current query rather than precomputed statistics. TriAttention (Mao et al., 2026), our internal baseline scorer ("1d" throughout), uses precomputed Q/K frequency statistics as a proxy for which positions future queries will need. These methods all modify *which* tokens are kept under a fixed budget; α (Section 4) inherits this scope, adding a single V-space diversity term and replacing the modular top-k argmax with a greedy facility-location-inspired selector that conditions each pick on the already-selected set. Closely related, R-KV (Cai et al., 2025) identifies redundancy in K-space for reasoning models; we share the redundancy-as-signal framing but operate in V-space and through the existing retention scorer rather than as a separate stage. The broader family of diverse subset selection via determinantal point processes (Kulesza & Taskar, 2012) offers an alternative formalisation; we use a lighter facility-location heuristic to stay one function away from the existing scorer.

Budget-allocation compression. A second line of work modifies how a global cache budget is split across the model's structure. PyramidKV (Cai et al., 2024) allocates KV-cache budget per layer in a pyramidal shape, motivated by the empirical observation that lower layers attend more broadly than upper layers. Ada-KV (Feng et al., 2025) allocates budget per attention head adaptively, moving budget toward heads with higher retention need, and is intended to compose with an underlying attention-based scorer (e.g., SnapKV) rather than replace it. These methods are budget allocators, not scoring functions: they operate on the same retention/allocation/scoring axis as α but at a different lever, and are in principle stackable with α as the within-budget scorer; Section 6 discusses such compositions.

Quantization-based compression. Orthogonal to selection-based compression, several methods compress the K and V tensors themselves via quantization: KIVI (Liu et al., 2024) (asymmetric per-channel K and per-token V quantization at 2 bits), KVQuant (Hooper et al., 2024) (non-uniform datatypes with per-channel scales), KVTuner (Li et al., 2025) (mixed-precision), MiniKV (Sharma et al., 2024) (rank-based), QAQ (Dong et al., 2024), and ASVD (Yuan et al., 2024). Brandon et al. (2024) reduce KV-cache memory by sharing the cache across layers. These directions are complementary to scoring-axis interventions: quantization modifies how each retained slot is represented, while selection modifies which slots are retained. Our negative gate `subspace_moekv` (Section 3) is in this family in spirit — per-head V is compressed via a low-rank PCA basis — and was killed under our matched-memory protocol.

Workload and evaluation methodology. We evaluate on long-form mathematical reasoning generated by distilled-reasoning models (DeepSeek-AI, 2025), which distill from the Llama 3 (Dubey et al., 2024) and Qwen base families and trace their math capability to math-specialized lines such as DeepSeek-Math (Shao et al., 2024); chain-of-thought prompting (Wei et al., 2022) unlocks multi-step reasoning at the cost of long decode horizons, the regime where KV-cache compression matters most. The MATH benchmark (Hendrycks et al., 2021) and GSM8K (Cobbe et al., 2021) provide closed-form answers that admit sympy-based grading, which normalizes equivalent answer forms (`\frac` vs. `\dfrac`, simplification, sign convention) and removes the 5–10 pp formatting-noise swings that exact-string graders show at small n; at $b = 64$, 44–62 % of incorrect runs produce no parseable boxed answer at all, which an exact-string grader would conflate with reasoning

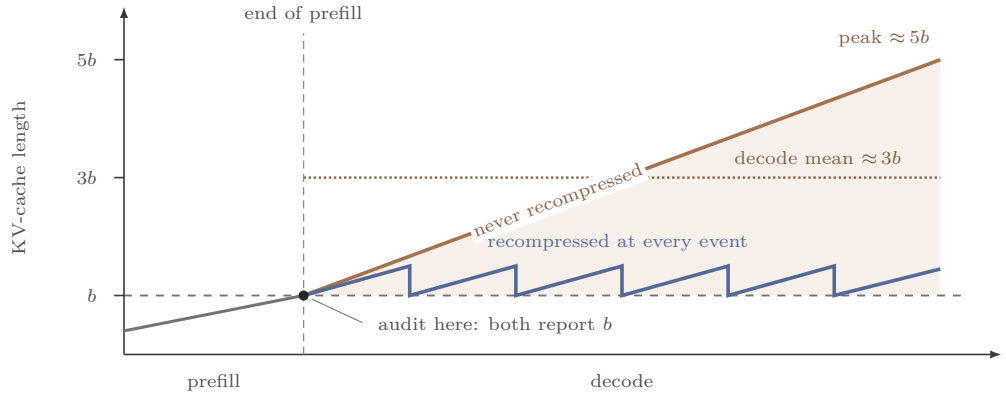

Figure 2: Why end-of-prefill memory matching is insufficient (schematic; not measured cache trajectories). Two methods identical at the end of prefill diverge sharply during decode, so the protocol matches *mean* cache over the full generation, not end-of-prefill cache (Section 3.2).

failure. Most published KV-compression methods report on LongBench (Bai et al., 2024), RULER (Hsieh et al., 2024), or similar retrieval benchmarks rather than long-form generation accuracy. Two protocol choices distinguish ours. End-of-prefill memory matching is insufficient when the decode horizon is long — methods that do not recompress during decode use multiples of the nominal budget (Figure 2) — so we match *mean* cache instrumented over the full generation; several negative gates were green on end-of-prefill matching and red under it. Separately, reporting the same dataset for tuning and headline inflates apparent performance (Bouthillier et al., 2021), so we separate the two via a frozen dev/confirm split with the decision rule committed in code before confirmation, and argue pre-registration should be standard here.

3 Design-Space Study: Why Most Mechanisms Failed

This section is a retrospective design-space study, not a pre-registered sweep. We tested seven mechanisms across the KV-retention pipeline; every structural intervention failed under matched-memory evaluation, and the surviving design is a small scoring fix (Section 4). The negatives are what make the survivor interpretable: the contribution is not that one mechanism worked, but that the lone survivor is a scoring-axis mechanism of a particular form (Section 3.3) and that the failures fall along mechanism-class boundaries rather than implementation-detail boundaries.

At each decode step the model holds a key–value cache of length up to some budget B. A KV-retention pipeline must answer five questions, in roughly this order. *State* asks what the cache holds: per-token K/V tensors by default, with alternatives that compress or replace tensors via PCA bases, span-mean "synthetic" summaries, or low-rank projections. *Routing* asks which heads or layers see which subset of the cache: a single global policy by default, with alternatives that admit per-head or per-layer choices among multiple scoring strategies. *Cadence* asks when compression is triggered and with what budget across decode steps: a fixed budget at every trigger by default, with alternatives that modulate budget by step index. *Decoding* asks what the model emits while the cache is being compressed: unmodified generation by default, with alternatives that prompt for dense anchor tokens the eviction scorer can pin. *Scoring* asks which tokens are kept under fixed budget, routing, cadence, and decoding behavior: this is where attention-based heuristics (TriAttention (Mao et al., 2026), SnapKV (Li et al., 2024), H2O (Zhang et al., 2023)) and our α regularizer live. The five families fall along three axes: state modifies the *representation* of cached entries, routing and scoring modify the *selection* over cached entries, and cadence and decoding are control-side — they modify when to compress and what the model emits without changing what the cache holds at any given moment. Figure 1 (page 2) shows the five surfaces with cadence and decoding under a shared "control-side" super-bracket; in the verdict table below they are tracked separately because they were independent hypotheses with independent kills.

3.1 Verdict Table

Table 1 summarizes every mechanism we tested, organized by family. Each of the seven negative gates corresponds to a single entry in the negative-results catalog distributed with this paper; the shaded bottom row is the surviving mechanism α. The synthetic-memory state family explored four sub-attempts before we closed it at family level; for table compactness those four are reported as a single row with a footnote pointer.

Family	Gate	Hypothesis tested	Verdict	Failure mode
State	`subspace`	Per-head V is low-rank; a rank-32 PCA basis holds quality at $\sim$1.6$\times$ memory saving.	Killed	Proxy-to-runtime drift
	`syn_mem`[†]	Span-mean "synthetic" slots plus a $\log(N)$ mass bias beat per-token retention at extreme budgets.	Killed	Family-level bundling
Routing	`static_route`	An offline-calibrated per-head choice between uniform and attention scoring beats either alone.	Killed	Proxy-to-runtime drift
Cadence	`schedule`	Round-indexed budget schedules favor high-risk reasoning stretches at matched mean cache.	Killed	Iso-cache underperformance
Decoding	`state_emit`	Prompted compact "State:" anchor lines give the eviction scorer dense pinnable tokens.	Killed	Iso-cache underperformance
Scoring	`variant_b`	A mean-V cosine prior ($+\lambda\cos(V, \text{dir})$) adds signal beyond attention-only scoring.	Killed	Proxy-to-runtime drift
	`Q_stats`	Recalibrating TriAttention's Q/K statistics on math traces improves `1d` at aggressive budgets.	Killed[‡]	n=50 instability
	α (λ=0.5)	Set-conditioned greedy selection with a cosine-against-retained V-diversity penalty.	**Survived**	n/a

Table 1: Verdict summary across mechanism families. The bottom row, shaded, is the surviving mechanism examined in detail in Sections 4–5. [†] The synthetic-memory state family closed at family level after four consecutive sub-attempts (span-mean prefill states, bounded variant, and two static/dynamic `sel_hybrid` formulations). All four failed iso-cache evaluation; we report them as a single family-level kill in the table and itemize them in the catalog. [‡] The `Q_stats` gate was decided on an n=50, single-model (Qwen-7B) screening evaluation that predates the matched-memory, $n\geq$200, two-model standard later applied to the other catalog gates and to α; we report it as a screening-grade kill (Section 3.2).

The coverage is asymmetric, and we flag it. Across the four structural families we ran five catalog mechanisms (and roughly ten sub-attempts once the synthetic-memory family is itemized), against three scoring-family mechanisms (variant-B V-direction prior, reasoning-statistics recalibration, and α). Scoring received the same or more design effort per mechanism; the family is narrower because, once budget, routing, cadence, and decoding are fixed, the remaining design surface is the score function itself, which admits fewer qualitatively distinct interventions than the structural surfaces it sits on top of. Section 3.3 makes this argument precise.

3.2 Shared Failure Modes

The kill verdicts in Table 1 are not independent. Five failure modes recur across the mechanism families. Four of them carry the kills in Table 1 — its **Failure mode** column names, for each gate, the mode its kill rests on — and the fifth, cross-model transfer failure, is a validity threat that no single gate's kill rests on but that the protocol of Section 4.2 is built to defend against. One further effect, b=64 extraction collapse, is an evaluation-regime caveat rather than a mechanism-failure mode; we describe it last.

The first mode, *iso-cache underperformance*, has the widest reach. Once mean cache length is matched at decode time — not merely at the end of prefill — structural mechanisms sit below the linear interpolation between fixed-budget reference points. End-of-prefill matching is not matched memory: a mechanism that does not recompress during decode can silently use 5$\times$ more cache than its reported budget would suggest (Figure 2), so we require per-problem peak/mean cache instrumentation as a gate-passing condition. Two gates fail this way through gate-specific symptoms. The cadence schedules (`schedule`) sat $\sim$6 pp below

the iso-cache interpolation whether budget was front-loaded or back-loaded, and a budget-step inversion test confirmed the shortfall is phase-independent; the decoding gate (`state_emit`) instead saw its compact "State:" anchor lines crowd reasoning content out of the cache at b=128. The local symptom differs, but the mode is the same.

Two further modes share both a cause and a remedy: each is evidence read off a shortcut that a proper test does not bear out, and each is therefore treated as a screening signal rather than a verdict. Under n=50 *instability*, headline deltas at n=50 on MATH-500 routinely shrank substantially at n=200: an early α-line measurement at n=50 on Qwen-7B reported a mean lift of +5.6 pp at aggressive budgets, yet the same mechanism at n=200 on two models shrank to between +0.2 and +0.5 pp, and the reasoning-statistics gate was even sign-inconsistent across budgets at fixed n=50 (+6 pp at b=128, −2 pp at b=64). We therefore treat any n=50 decision as screening only and require $n \geq 200$ to confirm; because the `Q_stats` kill rests on exactly this n=50, single-model evidence and was not re-run at the $n \geq 200$ two-model standard, Table 1 marks it as a screening-grade kill rather than a fully protocol-matched one. *Proxy-to-runtime drift* is the same hazard in metric form: pre-RoPE Q/K attention surrogates, oracle-keep capture rates, and mean-V cosine similarities all failed to predict matched-memory task accuracy, and the variant-B, subspace-moeKV, and static-route gates were each green on a proxy and red at runtime. In both cases the decision rule is runtime accuracy on the matched-memory protocol; a proxy or a small-n delta only screens.

A fourth mode, *family-level bundling*, is a discipline on negative inference: a single sub-attempt's failure is not a class kill. The synthetic-memory state family was rejected only after four consecutive sub-attempts — span-mean prefill, a bounded re-gate, and static and dynamic `sel_hybrid` — had all failed iso-cache. That family-level kill is load-bearing: it is what licenses the claim "state-side rewrites do not survive matched-memory evaluation" rather than the weaker "this particular state-side rewrite did not."

The fifth mode is the one no gate's kill rests on. *Cross-model transfer failure* is a validity threat the protocol must defend against regardless: interventions calibrated on Qwen-7B did not always survive transfer to Llama-8B at $n \geq 200$. The α-only mechanism at λ=1 — an earlier checkpoint on the path to the Branch-A result — produced a Qwen-7B lift around +1.7 pp at $p \approx 0.05$ on the b=128 cell while leaving the Llama-8B b=64 cell statistically null ($p = 0.92$); only after re-tuning λ to 0.5 on a held-out dev split did Llama recover a Bonferroni-significant lift. Several other gates evaluated on Qwen alone — the reasoning-statistics recalibration among them — provide no Llama-side evidence either way, which is itself a transfer-failure mode: a single-model gate is not gate-passing. We therefore require both Qwen-7B and Llama-8B in any gate-passing evaluation, and flag single-model wins as such.

One further effect is a caveat rather than a failure mode, and it bounds how the b=64 cells should be read. At budgets near 64 on long-form math reasoning, 44–62 % of incorrect runs fail with `extracted_answer = None` — the model produces no parseable boxed answer at all, regardless of reasoning quality. This b=64 extraction collapse afflicts every method, the baseline included, so it kills no gate; but it does mean that aggressive-budget evaluation at b=64 partly tests structured-output fluency rather than retention quality, so we report b=64 alongside b=128 to keep both signals visible and flag the affected cells where they arise (Section 5.2).

These five failure modes and the b=64 caveat are the methodological backbone of the evaluation in Sections 4–5. The protocol was designed around these risks: a frozen dev/confirm split (against n=50 instability and threshold chasing); evaluation on both Qwen-7B and Llama-8B (against cross-model transfer failure); per-problem mean cache instrumented during decode (against iso-cache underperformance); runtime accuracy as the decision variable rather than any proxy metric (against proxy-to-runtime drift); and a Bonferroni-corrected joint test across the four (model, budget) cells (against per-cell multiple-comparison artifacts).

3.3 What Distinguished the Survivor: Preview, with Caveats

The pattern in Table 1 is the design-space claim: the sole survivor is on the scoring axis. Structural mechanisms change what the cache contains, what routes to it, when it is compressed, or what gets generated — each family adds degrees of freedom that have to be paid for in retention quality at matched memory, and the payments were not recovered. The α selector (Section 4) operates on the same K/V tensors, routing, budget

schedule, and generated text as the baseline, adding a single diversity term to the existing $1d$ score; at $\lambda = 0$ it is a bitwise no-op against the baseline.

A small intervention surface is, on its own, not what separated the survivor from the failures. The `variant_b` gate is also a one-term, one-weight scoring modification — it adds $\lambda \cos(V, \text{dir})$ to the score — and it was rejected; so was `Q_stats`, a recalibration that adds no term at all. Of the three scoring-family mechanisms we tested, two failed, and the intervention surface does not separate them: `variant_b`'s is if anything smaller than α's, since α additionally replaces top-k with a greedy selector. That extra machinery is not a free design choice — it is *forced* by what does distinguish α: its selection rule is *set-conditioned*. The redundancy penalty in Equation 2 is evaluated against the already-retained set, so each pick depends on the picks before it, which an independent per-token argmax cannot express. `variant_b`'s cosine-to-a-fixed-direction prior and `Q_stats`'s recalibrated statistics both rescore tokens *modularly* — token by token, with no dependence on the retained set — leaving top-k intact. The design-space reading is therefore two-part: the scoring axis was the only family to yield a survivor, and the feature separating that survivor from the two scoring-family failures was set-conditioned redundancy control, not minimality as such. This is a single-survivor observation rather than a controlled comparison; Section 6 states it as a design lesson with that caveat, and Section 5 provides the matched-memory evidence under the pre-registered protocol.

Two caveats bound how Table 1 should be read. The seven negative gates were sequential hypothesis tests with shifting motivations, not a pre-designed factorial sweep across the family taxonomy: the taxonomy is the contribution of this section, the chronology is a fact about how we arrived at it, and the catalog distributed with this paper records the full timeline. Separately, the pre-registered protocol applies to α (Sections 4–5), not to the negative gates that pre-date it: Section 3 is retrospective, and the strength of the design-space claim rests on the confirmation in Section 5, not on the negatives alone.

4 α: Method and Protocol

This section defines α and the pre-registered protocol used to evaluate it. α is a one-function modification to the TriAttention $1d$ retention scorer (Mao et al., 2026) that replaces argmax-top-k with greedy facility-location-inspired selection under a V-space redundancy penalty. The companion protocol (Section 4.2) is designed against the failure modes catalogued in Section 3: a frozen 201-problem development split for λ tuning and a disjoint 299-problem held-out split for confirmation, with three seeds, cluster bootstrap, and a Bonferroni-corrected joint test across four (model, budget) cells. Section 5 reports Phase 1 and Phase 2 results and reads the verdict against a pre-committed branch table.

4.1 Method: A V-Space Diversity Penalty

Let $s \in \mathbb{R}^n$ be the per-token retention scores produced by the TriAttention $1d$ scorer at a compression event, where n is the current cache length and $k \le n$ is the post-compression budget. The baseline retention rule is modular top-k: keep the indices in $\arg\max_k s$. We replace this with a greedy facility-location-style selector under a V-space redundancy penalty.

For each token i, define a V-signature $v_i \in \mathbb{R}^d$ by averaging the value tensor across layers and across attention heads and unit-normalizing:

$$v_i = \frac{1}{\|u_i\|_2 + \varepsilon}\, u_i, \quad u_i = \frac{1}{LH} \sum_{\ell=1}^{L} \sum_{h=1}^{H} V_i^{(\ell,h)}, \tag{1}$$

where $V_i^{(\ell,h)} \in \mathbb{R}^{d_{\text{head}}}$ is the value vector for token i at layer ℓ, head h, and L, H are the layer and head counts. Cosine similarity between two unit-normalized signatures reduces to a dot product: $\cos(v_i, v_j) = v_i^\top v_j$.

Given an already-selected subset S_t of size t, the marginal gain of adding candidate token $i \notin S_t$ is

$$\Delta(i \mid S_t) \;=\; s_i \;-\; \lambda \max\Big(0,\; \max_{j \in S_t} \cos(v_i, v_j)\Big). \tag{2}$$

The outer $\max(0, \cdot)$ floor reflects an implementation choice: candidates anti-aligned with the selected set should not receive a *negative* redundancy penalty (i.e., a bonus). A token whose V-signature is roughly

Algorithm 1 α retention selection (one compression event).

Require: base scores $s \in \mathbb{R}^n$, V-signatures $V \in \mathbb{R}^{n \times d}$ with rows unit-normalized, budget k, regularization $\lambda \geq 0$.

if $\lambda = 0$ **then**
  return indices of top-$k(s)$ *(top-k fast path)*
end if
Sims $\leftarrow VV^\top$ // $n \times n$ cosine matrix
$S \leftarrow \emptyset$; $m \leftarrow \mathbf{0}_n$ // $m_i = \max\big(0,\ \max_{j \in S} \cos(v_i, v_j)\big)$
$i_1 \leftarrow \arg\max_i s_i$; $S \leftarrow S \cup \{i_1\}$; $m \leftarrow \max(m, \text{Sims}[i_1, :])$
for $t = 1, \ldots, k-1$ **do**
  $a \leftarrow s - \lambda m$; $a[S] \leftarrow -\infty$
  $i \leftarrow \arg\max_j a_j$
  $S \leftarrow S \cup \{i\}$; $m \leftarrow \max(m, \text{Sims}[i, :])$
end for
return sort(S)

orthogonal to or anti-aligned with S_t is treated as having zero redundancy with it. The budget is filled greedily:

$$i_{t+1} \;=\; \arg\max_{i \notin S_t} \Delta(i \mid S_t), \qquad t = 1, \ldots, k-1. \tag{3}$$

The first pick $i_1 = \arg\max_i s_i$ is the modular argmax (matching the baseline). At $\lambda = 0$ the selector reduces exactly to top-k; this is enforced by an explicit fast path so that $\lambda = 0$ is a bitwise no-op against the TriAttention (Mao et al., 2026) baseline.

Algorithm 1 summarizes the selector.

Equation 3 is a facility-location-inspired greedy selector (Krause & Golovin, 2014): each pick balances the per-token base score against a redundancy term against the already-selected set. Compared to the modular $\arg\max_k s$ baseline, the per-step decision now depends on the current selection, breaking the indices-are-independent property of top-k. Our objective is not in general submodular, so we do not claim the $(1 - 1/e)$-style approximation guarantee for greedy submodular maximization (Nemhauser et al., 1978); the empirical value of the selector is established by the held-out evaluation in Section 5.1.

The modification is intentionally minimal: the baseline K and V tensors, routing, budget schedule, and decoding procedure are all unchanged, and only the k-element selection step inside the existing $1d$ scorer is modified. The hyperparameter λ controls the strength of the diversity penalty and is tuned in Phase 1. Algorithm 1 computes an $n \times n$ cosine matrix once per compression event and then runs $k - 1$ greedy picks with constant-time marginal-gain updates, an $O(n^2 d + nk)$ step over the candidate pool of size n; at $b \in \{64, 128\}$ this overhead is negligible against the rest of decode in our matched-memory harness, and we did not specifically optimize it.

4.2 Pre-registered Protocol

Section 3.2 listed five failure modes that an honest evaluation in this setting has to defend against; the protocol below addresses each, and was registered as executable code before Phase 2 ran. We hash each MATH-500 (Hendrycks et al., 2021) problem's `unique_id` with MD5, bucket by (md5 mod 5), and assign buckets $\{0, 1\}$ to a 201-problem development split and buckets $\{2, 3, 4\}$ to a 299-problem held-out confirmation split; the split is deterministic, is produced by `alpha_paper_split.py`, and was committed before Phase 1 rather than recomputed once Phase 1 results were observed. The Phase 1 λ-selection (`analyze_alpha_lambda_probe.py`) and the Phase 2 cluster bootstrap and Branch A/B/C decision (`analyze_alpha_paper_confirmation.py`, whose branch-decision function is the rule stated under *Decision rule* below) are committed code as well: the branch-decision code carries development-repository commit `42b8e75` of 2026-04-30, six days before the Phase 2 confirmation run of 2026-05-06, and was not modified afterwards. All three scripts and the frozen

Model	b	Δ (pp)	95 % CI	p	Bonferroni
qwen7b	64	+1.56	$[-0.89, +4.01]$	0.20	—
qwen7b	**128**	**+4.79**	**[+1.67, +7.92]**	**0.0022**	**PASS**
llama8b	**64**	**+2.23**	**[+0.56, +3.90]**	**0.0054**	**PASS**
llama8b	128	−0.22	$[-2.45, +2.01]$	0.86	—

Table 2: Phase 2 confirmation on the held-out split. Per-cell paired delta with cluster bootstrap on per-problem accuracy differences ($n_{\text{boot}} = 10{,}000$). Joint Bonferroni threshold $\alpha = 0.0125$. Two of four cells clear Bonferroni; no cell has a significantly negative effect.

split files ship as anonymized supplementary material with this submission — so the decision rule can be read and the split reproduced during review — and form part of the public release described in Section 6.

Phase 1 (λ-probe; dev split only). Phase 1 is a lightweight λ-selection screen, not a full factorial: we sweep $\lambda \in \{0.5, 1.0, 1.5\}$ across two models (DeepSeek-R1-Distill-Qwen-7B, DeepSeek-R1-Distill-Llama-8B) and two budgets ($b \in \{64, 128\}$) at a single seed, on the 201-problem dev split only. The $\lambda = 1.0$ point reuses an existing gate (Appendix A); the $\lambda \in \{0.5, 1.5\}$ points required eight new evaluations. The $\lambda = 1.5$ arm is incomplete on three of four cells (Appendix A and Table 3 report the exact per-cell n); we disclose the gap, verify the ranking is robust to it, and rely on Phase 2 to carry the central claim. The decision rule is fixed in advance: pick the λ that maximizes the mean of $\Delta(1d_{\text{div}} - 1d)$ across the four (model, budget) cells. Ties within ± 0.5 pp are broken toward the smaller λ (Occam).

Phase 2 (confirmation; confirm split only). The winning λ from Phase 1 is evaluated on the 299-problem confirm split with three seeds per cell, two models, two budgets — twelve cells total, all sympy-regraded. The 1d baseline is reused from the prior gate (subsetted to the confirm IDs; no new compute). We report per-cell paired deltas with cluster bootstrap on per-problem accuracy differences ($n_{\text{boot}} = 10{,}000$).

Decision rule. The four (model, budget) cells form a Bonferroni family. Per-cell significance uses $\alpha = 0.05$; joint significance after Bonferroni correction uses $\alpha = 0.05/4 = 0.0125$. Branch decisions were pre-committed before unblinding, mutually exclusive. *Branch A* (primary-claim success) requires at least one Qwen cell with $\Delta > 0$ at per-cell $p < 0.05$, a positive Llama mean Δ across cells, and no significantly negative Llama cell. *Branch B* (restricted generalization) satisfies the Qwen condition but fails one of the two Llama conditions. *Branch C* (no central α claim) has no qualifying Qwen cell. One asymmetry in the rule as registered: the no-significantly-negative guard is evaluated over the Llama cells only, an oversight in the committed rule that we did not catch before unblinding. We report against the rule as registered, and note in Section 5.1 that the Phase 2 verdict is unchanged under the stricter symmetric guard — no significantly negative cell on *either* model — which the data also satisfy.

5 Confirmation Results

This section reports the held-out Phase 2 confirmation for α under the protocol of Section 4.2, reads the verdict against the pre-committed branch table, and accounts for the two non-significant cells.

Phase 1 swept $\lambda \in \{0.5, 1.0, 1.5\}$ on the 201-problem development split and applied the pre-registered rule — maximize the mean of $\Delta(1d_{\text{div}} - 1d)$ across the four (model, budget) cells, ties broken toward the smaller λ. The winner is $\lambda = 0.5$ (mean development-split $\Delta = +0.75$ pp, against +0.43 at λ=1.5 and −0.50 at λ=1.0); Appendix A reports the full per-cell development table. Phase 2 evaluates the winning λ only.

5.1 Phase 2: Held-Out Confirmation

Phase 2 evaluates $1d_{\text{div}}$ at $\lambda = 0.5$ against $1d$ on the held-out confirm split, with three seeds per cell. Table 2 and Figure 3 (a) report per-problem cluster-bootstrap estimates of $\Delta(1d_{\text{div}} - 1d)$ with two-sided p-values and 95 % CIs.

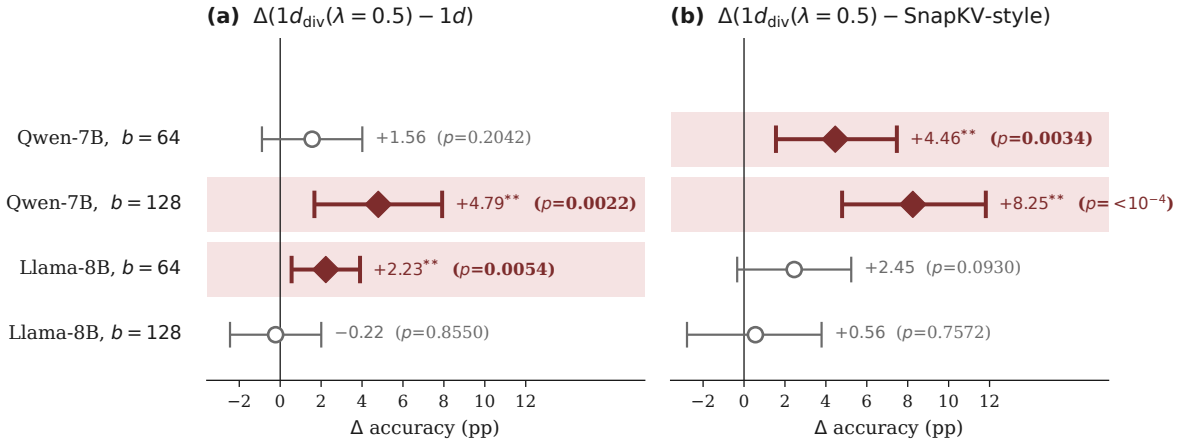

Figure 3: Side-by-side forest plots of two α-vs-baseline contrasts on the same held-out confirm split. Three seeds, two-sided cluster bootstrap ($n_{\text{boot}} = 10{,}000$), Bonferroni-corrected jointly over the four (model, budget) cells ($\alpha = 0.0125$, marked **). Panel **(a)**: $\Delta(1d_{\text{div}}(\lambda{=}0.5) - 1d)$ — the pre-registered Phase 2 confirmation. Panel **(b)**: $\Delta(1d_{\text{div}}(\lambda{=}0.5) - \text{SnapKV-style})$ — the post-confirmation head-to-head, reported in Appendix B. Filled brick diamonds with shaded row tints mark Bonferroni-pass cells; open gray circles mark non-significant ones; inline annotations show the Δ point estimate (pp) and two-sided p.

$\lambda = 0.5$ was selected on the disjoint development split by the pre-registered rule, not on held-out confirm; on the confirm split it strictly improves on the earlier $\lambda = 1.0$ checkpoint at all four cells, including flipping Llama b=64 from null to Bonferroni-pass (Appendix A). The split structure makes that Llama b=64 pass especially clean: the cell stood at $\Delta = +0.0$ pp on the development split, so the selection of $\lambda = 0.5$ drew no signal from it, and its +2.23 pp confirm result is free of dev-to-confirm selection bias.

The pre-committed branch decision reads directly from Table 2. Qwen has a Bonferroni-passing cell at b=128, the Llama mean Δ across cells is +1.00 pp (positive), and the Llama b=128 null ($p = 0.86$) is well within the null region rather than significantly negative. All three Branch-A conditions hold; **Branch A** triggers. The verdict does not depend on the Llama-only asymmetry in the registered guard (Section 4.2): under the stricter symmetric form — no significantly negative cell on either model — the two non-significant cells (Qwen b=64, $p = 0.20$; Llama b=128, $p = 0.86$) sit within their null regions, so no cell on either model is significantly negative and Branch A still triggers.

5.2 Honest Negatives

Two cells in Table 2 did not clear per-cell significance and we report them as such. Qwen b=64 ($\Delta = +1.56$ pp, $p = 0.20$) is positive in direction, like two of the other three cells, but its 95 % CI contains zero; the underlying baseline accuracy is only $\sim 8\%$ and roughly half of the failures at b=64 are extraction-collapse rather than reasoning quality (Section 3.2), so the effective signal is small. Llama b=128 ($\Delta = -0.22$ pp, $p = 0.86$) is the one robust null: at the highest baseline accuracy of the four cells, the diversity penalty produces no measurable change, and the cell was also null at the prior $\lambda = 1$ checkpoint on the same problems, so the null is consistent across two λ values rather than a one-shot fluctuation. We do not propose a mechanism for this null and the protocol does not support a post-hoc explanation. The honest framing is that α produces positive point estimates at three of four cells, of which two clear the joint Bonferroni threshold.

A post-confirmation head-to-head against a SnapKV-style windowed-attention baseline (Li et al., 2024), outside the pre-registered decision rule, reproduces the shape of the Phase 2 result: $1d_{\text{div}}$ is positive in point estimate on all four cells against the SnapKV-style comparator and clears the joint Bonferroni threshold on both Qwen cells (mean $\Delta = +3.93$ pp), with positive but underpowered Llama cells. An auxiliary contrast indicates the internal $1d$ baseline is itself competitive with the SnapKV-style implementation, so α's lift does not rest on an artificially weak floor. Appendix B reports the head-to-head in full.

6 Design Lesson and Conclusion

In the small-budget KV-retention regime we study, the scoring axis was the only intervention family to yield a survivor, and the feature separating that survivor from the scoring-family failures was a set-conditioned selection rule rather than the size of its intervention surface. The surviving mechanism (α, Section 4) preserves the K and V tensors, the routing policy, the budget cadence, and the decoding behavior of the baseline, modifying only the k-element selection step inside the eviction scorer with a single V-space redundancy term and one tunable weight. The seven negative-gate mechanisms catalogued in Section 3 — five structural attempts across state ($\times 2$), routing, cadence, and decoding, together with two earlier scoring-family attempts (a V-direction prior and a reasoning-statistics recalibration) — each failed to recover their cost in retention quality at matched memory. The two scoring-family failures are what keep the lesson honest: both are modifications of a size comparable to α, and both rescore tokens modularly, so it is α's set-conditioning — each pick conditioned on the already-retained set (Section 3.3) — and not its minimality that the evidence singles out. α is the third scoring-family attempt and the only mechanism that survived.

At the small budgets and long-form math-reasoning workload we tested, the baseline retention pipeline already does most of the work, so a structural intervention must outperform a strong default rather than fill an empty slot — and the matched-memory, $n \geq 200$, two-model protocol is what made that bar, and the resulting asymmetry, visible.

Two methodological practices generalize beyond this specific finding, whether or not the finding itself transfers. The first is to match memory across the full decode horizon rather than at the end of prefill. The single largest source of false positives in our design-space study was end-of-prefill memory matching: several gates—most prominently the synthetic-memory family—were green on end-of-prefill cache equality and sat 5–12 pp below their fixed-budget reference once mean cache was instrumented over the full generation. We recommend per-problem mean (and ideally peak) cache instrumentation as a gate-passing condition for any KV-retention claim.

The second is to treat $n = 50$ as a screening signal and to require $n \geq 200$ for confirmation. Headline deltas at $n = 50$ on MATH-500 routinely shrank or sign-flipped at $n = 200$: the α direction itself looked like a +5.6 pp lift at $n = 50$ and shrank to +0.2–0.5 pp at $n = 200$, and the reasoning-statistics gate was sign-inconsistent across budgets at $n = 50$ (+6 pp at b=128, −2 pp at b=64). We treat $n = 50$ as screening only; confirmation requires $n \geq 200$ on both models.

The Branch-A confirmation is intentionally narrow, and we separate what the protocol entitles us to claim from what is open. The evaluation is long-form mathematical reasoning (MATH-500, DeepSeek-R1-Distill chain-of-thought) on two distilled-reasoning models in the 7–8B range. The matched-memory protocol carries to other workloads and model scales in principle, but the magnitude and Bonferroni pass-rate we report are not evidence about other workloads (LongBench, multi-document QA, code generation) or other models (non-reasoning, instruct-tuned, or larger, including 70B-class models), and the cell-level heterogeneity we observe (saturation at the larger budget on Llama, smaller-but-positive on Qwen b=64) is a within-this-pair observation; the method has not been re-tested outside MATH-500. The hyperparameter $\lambda = 0.5$ was selected on a coarse $\{0.5, 1.0, 1.5\}$ probe grid and we do not claim it is the global optimum — the lesson from our λ refinement is qualitative (light regularization beat heavy), not numerical.

Two extensions are natural and untested. First, α is a within-budget scorer whereas PyramidKV and Ada-KV are budget allocators, so a natural stack places our scorer inside Ada-KV's per-head or PyramidKV's per-layer budget — a composition our protocol can evaluate directly. Second, λ is held constant across decode steps and the V-signature is a flat unit-normalized average of V over all layers and heads; whether a λ schedule or a per-head, per-layer, or learned signature recovers additional lift is open, as is a finer λ grid. Tooling for these variants is present (`diversity-signature-mode`); the experiments were deferred to keep the pre-registration scope tight.

Code and data availability. A standalone reference implementation of the α selector and the V-signature aggregator (Algorithm 1, Eqs. 1–3), the pre-registered MD5-bucketed dev/confirm split, the raw per-problem outputs from every cell in Section 5 and Appendices A–B, and the analysis scripts behind the tables and

figures will be released in a public repository; the link is omitted here for double-blind review and added after de-anonymization. The pre-registered split and the Phase 1/2 protocol scripts are already provided as anonymized supplementary material with this submission.

We tested seven mechanism interventions across state, routing, cadence, decoding, and scoring at small budgets on long-form mathematical reasoning, under a matched-memory standard that tightened over the study. None survived; α — a one-function diversity penalty on the TriAttention $1d$ (Mao et al., 2026) scorer with a top-k fast path at $\lambda = 0$ — did, clearing Bonferroni on two of the four (model, budget) cells under a pre-registered dev/confirm split and triggering the pre-committed Branch A. Whether a set-conditioned scoring rule remains the productive intervention outside this regime — across workloads, models, and budgets — is open; we expect the protocol that surfaced the asymmetry to carry over more readily than the specific finding.

Broader Impact Statement

This paper studies memory-efficiency techniques for language-model inference; it introduces no new datasets or models and evaluates only public benchmarks and open-weight models. The intended benefit is a lower hardware barrier to long-context reasoning inference. The risk most specific to this line of work is that aggressive KV-cache compression can silently degrade reasoning reliability; the matched-memory, held-out protocol advocated here is intended to make such degradation measurable rather than hidden.

References

Yushi Bai, Xin Lv, Jiajie Zhang, et al. LongBench: A bilingual, multitask benchmark for long context understanding. In *Annual Meeting of the Association for Computational Linguistics*, 2024.

Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Assya Trofimov, Brennan Nichyporuk, Justin Szeto, Nazanin Mohammadi Sepah, Edward Raff, Kanika Madan, Vikram Voleti, Samira Ebrahimi Kahou, Vincent Michalski, Tal Arbel, Chris Pal, Gaël Varoquaux, and Pascal Vincent. Accounting for variance in machine learning benchmarks. In *Conference on Machine Learning and Systems*, 2021.

William Brandon, Mayank Mishra, Aniruddha Nrusimha, Rameswar Panda, and Jonathan Ragan Kelly. Reducing transformer key-value cache size with cross-layer attention, 2024.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Yucheng Li, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Junjie Hu, and Wen Xiao. PyramidKV: Dynamic KV cache compression based on pyramidal information funneling, 2024.

Zefan Cai, Wen Xiao, Hanshi Sun, et al. R-KV: Redundancy-aware KV cache compression for reasoning models. In *Advances in Neural Information Processing Systems*, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.

Shichen Dong, Wen Cheng, Jiayu Qin, and Wei Wang. QAQ: Quality adaptive quantization for LLM KV cache, 2024.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, et al. The Llama 3 herd of models, 2024.

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 *Advances in Neural Information Processing Systems*, 2025.

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 *NeurIPS Track on Datasets and Benchmarks*, 2021.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. KVQuant: Towards 10 million context length LLM inference with KV cache quantization. In *Advances in Neural Information Processing Systems*, 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*, 2024.

Andreas Krause and Daniel Golovin. Submodular function maximization. In *Tractability: Practical Approaches to Hard Problems*, pp. 71–104. Cambridge University Press, 2014.

Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 5(2–3):123–286, 2012.

Xing Li, Zeyu Xing, Yiming Li, Linping Qu, Hui-Ling Zhen, Wulong Liu, Yiwu Yao, Sinno Jialin Pan, and Mingxuan Yuan. KVTuner: Sensitivity-aware layer-wise mixed-precision KV cache quantization for efficient and nearly lossless LLM inference. In *International Conference on Machine Learning*, 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*, 2024.

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.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. KIVI: A tuning-free asymmetric 2-bit quantization for KV cache. In *International Conference on Machine Learning*, 2024.

Weian Mao, Xi Lin, Wei Huang, et al. Triattention: Efficient long reasoning with trigonometric KV compression, 2026.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14(1):265–294, 1978.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models, 2024.

Akshat Sharma, Hangliang Ding, Jianping Li, Neel Dani, and Minjia Zhang. MiniKV: Pushing the limits of LLM inference via 2-bit layer-discriminative KV cache, 2024.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-aware sparsity for efficient long-context LLM inference. In *International Conference on Machine Learning*, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, 2024.

Zhihang Yuan, Yuzhang Shang, Yue Zhou, Zhen Dong, Zhe Zhou, Chenhao Xue, Bingzhe Wu, Zhikai Li, Qingyi Gu, Yong Jae Lee, et al. ASVD: Activation-aware singular value decomposition for compressing large language models, 2024.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Appendix

A Phase 1: λ-Probe on the Development Split

This appendix gives the full Phase 1 development-split detail summarized in Section 5. Table 3 reports per-cell accuracy on the development split for each λ in the probe grid, against the $1d$ baseline. The $\lambda = 1.0$ column reuses a prior gate's outputs subsetted to the dev IDs; the $\lambda \in \{0.5, 1.5\}$ columns are new dev-split evaluations. The aggregate row averages $\Delta(1d_{\text{div}} - 1d)$ across the four cells.

Model	b	$1d$ (n)	$\Delta_{\lambda=0.5}$	$\Delta_{\lambda=1.0}$	$\Delta_{\lambda=1.5}$
Qwen-7B	64	8.0 % (201)	+0.5 (201)	−1.0 (201)	+2.5 (200)
Qwen-7B	128	15.4 % (201)	+4.0 (201)	+1.0 (201)	+1.0 (164)
Llama-8B	64	7.5 % (201)	+0.0 (201)	+0.0 (201)	−1.9 (196)
Llama-8B	128	16.9 % (201)	−1.5 (201)	−2.0 (201)	+0.0 (201)
Mean Δ across 4 cells			+0.75	−0.50	+0.43

Table 3: Phase 1 λ-probe on the development split. Cells report $\Delta(1d_{\text{div}} - 1d)$ in percentage points; parenthetical n is the number of development problems present at that cell (several λ=1.5 cells are short due to an incomplete sweep slot). $\lambda = 0.5$ wins by mean Δ and is also the smaller of $\{0.5, 1.5\}$ within the ± 0.5 pp tie tolerance, so the Occam tie-break selects the same value.

By the pre-registered rule, $\lambda = 0.5$ wins. The $\lambda = 1.5$ cells are short on three of four because the probe sweep manifest combined $\lambda \in \{0.5, 1.5\}$ outputs into a single file per (model, budget); a per-row λ filter recovers each arm cleanly, and the λ=0.5 > λ=1.5 > λ=1.0 ranking is unchanged if the short cells are excluded.

Why $\lambda = 0.5$ outperformed $\lambda = 1.0$. The precursor gate that motivated this paper evaluated $1d_{\text{div}}$ at $\lambda = 1.0$. Subsetted to the confirm IDs, that arm produced a per-cell Bonferroni-pass on Qwen b=128 (+3.23 pp, $p = 0.0076$) but a null on Llama b=64 (+0.67 pp, $p = 0.45$); at $\lambda = 0.5$ on the same IDs the Qwen cell strengthens and the Llama cell flips to Bonferroni-pass (Table 2). Smaller regularization is strictly better on both formerly-positive and formerly-null cells under the same problems and models. We read this as evidence that the prior intuition ("more diversity penalty = better retention") was incorrect at this magnitude: at $\lambda = 1$ the discount on attention-driven content exceeded the gain from de-duplicating the kept set, so the base attention score must dominate the ranking except where two attention-equivalent candidates are V-redundant.

B Head-to-Head: α vs SnapKV-Style Baseline

This appendix reports in full the post-confirmation head-to-head summarized at the end of Section 5. The pre-registered protocol of Section 4.2 compared α against the internal TriAttention $1d$ baseline. We run a post-confirmation head-to-head against a SnapKV-style windowed-attention retention family (Li et al., 2024) on the same confirm split, three seeds, and four cells. The comparator is implemented inside our matched-memory harness under the method tag `snapkv_with_window`; we do not claim exact parity with the official SnapKV implementation. This analysis is outside the branch decision rule and tests whether α's gains survive comparison to a same-lever published scoring family.

The main contrast, $\Delta(1d_{\text{div}} - \text{SnapKV-style})$, is reported per cell in Figure 3 (b) under the same protocol used for the internal contrast (Section 5.1). All four cells are positive in point estimate; both Qwen cells clear the joint Bonferroni threshold, and the mean Δ across cells is +3.93 pp. The Llama point estimates are positive but underpowered against the SnapKV-style baseline: b=64 at +2.45 pp with $p = 0.09$, and b=128 at +0.56 pp with $p = 0.76$.

That contrast's interpretation depends on whether the internal $1d$ baseline itself is competitive. We therefore report a second, auxiliary contrast, $\Delta(1d - \text{SnapKV-style})$, as supporting evidence, without Bonferroni

correction. At Qwen b=128, the internal $1d$ is +3.46 pp above the SnapKV-style baseline at uncorrected $p = 0.048$; at Qwen b=64 it is +2.90 pp with $p = 0.058$. On Llama the two methods are within ± 0.78 pp at $p > 0.6$. The auxiliary contrast is consistent with the internal $1d$ baseline being competitive with the SnapKV-style implementation in our protocol, so α's additional lift on top of $1d$ in Section 5.1 does not sit on top of an artificially weak floor. We do not draw a stronger conclusion from the auxiliary contrast given its uncorrected p values.

The Qwen-vs-Llama asymmetry observed in Phase 2 recurs against a baseline of a different mechanism class, so the heterogeneity is more plausibly a property of the (model, b) point itself than of α's relationship to the internal $1d$ baseline.