EXPLORING A LAYER-WISE DESIGN SPACE FOR KV CACHE EVICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

KV cache eviction methods typically use a single retention-rule family throughout a model, making eviction-method identity a model-level design choice. Yet Transformer layers differ substantially in their attention behavior, representations, and sensitivity to compression, suggesting that a uniform rule may overlook useful layer-wise structure. This raises a basic question: should eviction methods themselves vary across layers? We investigate this question by composing existing eviction methods across Transformer layers and systematically exploring the resulting layer-wise design space. Using simple offline profiles, we construct fixed routes and study how their quality varies with method placement and cache budget. On LongBench, heterogeneous routing improves performance on a majority of tasks over homogeneous policies at the same cache budget. Even when method counts are held fixed, the profile-guided placement ranks second among 100 evaluated assignments, demonstrating that routing quality depends strongly on where methods are placed. Moreover, the same fixed route outperforms the best of nine standalone baselines across all five tested cache budgets. Together, these results establish layer-wise method composition as an exploitable, placement-sensitive design dimension for KV cache compression. Code is available here.

1 INTRODUCTION

Long-context inference powers increasingly capable LLM applications, including retrieval-augmented generation and language agents (Lewis et al., 2020; Yao et al., 2023). Yet its KV cache grows with context length, consuming substantial GPU memory and limiting the number of requests that can be served in parallel (Zhang et al., 2023; Kwon et al., 2023). KV cache eviction addresses this bottleneck by retaining only the historical states most likely to remain useful (Zhang et al., 2023; Liu et al., 2023; Xiao et al., 2024; Li et al., 2024). Existing methods encode distinct retention biases, but representative evaluations typically select one retention-rule family for the entire model. Although layer-local scores can lead the same rule to retain different positions at different layers, the identity of the rule itself remains a model-level choice.

Recent work relaxes uniform cache management through layer- or head-wise capacity adaptation, head specialization, and layer-wise routing of compression parameters (Ge et al., 2024; Cai et al., 2025; Feng et al., 2025; Zhou et al., 2025; Shen et al., 2025; Tang et al., 2025; Xiao et al., 2025; Fu et al., 2025; Sun et al., 2026). Yet a complementary question remains largely unexplored: whether different standalone retention-rule families should be assigned to different layers. Transformer layers differ markedly in attention patterns, representations, contextual roles, and pruning sensitivity (Tenney et al., 2019; Jawahar et al., 2019; Clark et al., 2019; Voita et al., 2019; Michel et al., 2019). Figure 1 shows sharp layer-wise variation in attention-support size and location, suggesting potential value in varying cache capacities and retention biases across layers. However, this variation does not establish layer-specific method preferences, since a sufficiently adaptive shared rule may already capture it. Model-wide evaluations therefore leave a question unanswered: *Does every layer need to use the same KV eviction method?*

To answer this question, we study layer-wise KV cache eviction design along two axes: *method routing*, which assigns a retention-rule family to each layer within the active phase or phases, and *capacity allocation*, which distributes a fixed aggregate nominal retained-token capacity across layers within each active phase. Our factorial comparisons separate routing, allocation, and their inter-

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.

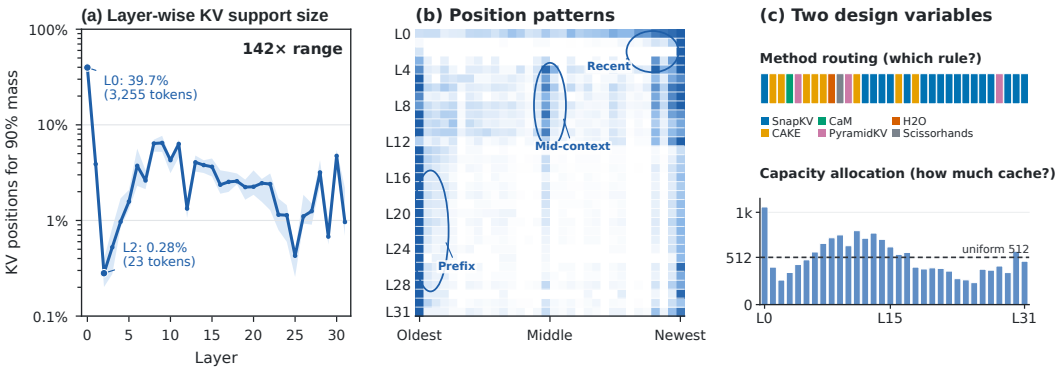

Figure 1: Layer-wise attention heterogeneity in LLaMA-3.1-8B-Instruct. Across 50 LongBench inputs with 8K-token prompts, we identify at the first decoding step the smallest KV-position set covering 90% of head-averaged attention. (a) Median set size varies by $142\times$ across layers; shading shows stratified 95% bootstrap intervals. (b) Row-normalized 32-bin support density for a representative input; outlines mark recent-token, mid-context, and prefix-focused patterns. (c) An evaluated RULER plan separates routing from allocation: upper-strip colors encode methods, lower bars encode entropy-derived capacities, and the dashed line marks uniform 512-token capacity.

action under matched prefill/decode scope and nominal retained-token capacity. Within routing, a count-matched control isolates placement by fixing method counts and varying only layer locations.

Within the layer-wise routing space, which grows exponentially with model depth, we study whether retention-rule identity constitutes an exploitable layer-wise design dimension. Identifying a globally optimal route lies outside this study's scope. We use PolyKV to represent each candidate as a layer- and phase-specific plan over compatible eviction methods and cache capacities. To construct tractable candidates, we profile existing methods on a small held-out set, using FullKV-preservation diagnostics for routing and layer-level proxies for allocation. These signals separately instantiate method-routing and capacity-allocation candidates. Once constructed, each plan remains fixed at inference and requires no retraining or online search. This process yields reproducible candidates for the controlled comparisons that follow.

Our results provide complementary positive evidence for layer-wise method routing. On LongBench, matched within-model routing contrasts are $+0.26$ points on LLaMA-3.1-8B-Instruct and $+0.27$ points in a retrospective exact-capacity Qwen3-8B check. Non-uniform allocation yields smaller and mixed changes, identifying routing as the more strongly supported factor. The count-matched placement control further tests whether layer locations matter after method counts are fixed. The guided route scores 0.31 points above the mean of 99 random count-matched placements and ranks second among the 100 evaluated assignments. One route held fixed from 128 to 1024 average tokens per layer improves over the budget-wise best standalone eviction baseline at all five operating points by 0.46 to 2.22 points. The largest gains occur under stronger compression. A second workload-specific construction on RULER improves over matched SnapKV by 15.98 points and remains ahead of that anchor at all four context lengths. The gains vary sharply across task families and include a regression on word extraction. The overall score remains slightly below standalone TOVA. Taken together, these results show that layer-wise routing of independently developed standalone retention rules can add value beyond selecting a single strong homogeneous method. Thus, even when the constituent rules are competitive on their own, method routing creates an independent design axis, with particularly strong evidence that routing quality depends on layer placement.

Our contributions are threefold.

- We formulate two axes of layer-wise KV eviction: method routing and capacity allocation.
- We develop PolyKV for profile-guided construction and controlled evaluation of static layer-wise plans under matched nominal retained-token capacities.
- We show that method routing contains exploitable placement-sensitive structure, while the tested non-uniform allocations yield weaker and inconsistent effects.

2 Related Work

2.1 Retention Rules as Model-Wide Alternatives

KV cache eviction reduces long-context inference memory by retaining or consolidating only a subset of historical key and value states. Existing methods embody different assumptions about which states will remain useful. H2O preserves heavy hitters using accumulated attention, SnapKV scores prompt positions through a recent observation window, Scissorhands tracks persistent importance across decoding steps, and TOVA uses attention from the current query (Zhang et al., 2023; Li et al., 2024; Liu et al., 2023; Oren et al., 2024). StreamingLLM follows a positional rule that retains attention sinks and recent tokens (Xiao et al., 2024). CaM consolidates cached states through merging, while D2O combines cumulative-attention eviction with recall and merge operations (Zhang et al., 2024; Wan et al., 2025). These methods provide retention-rule families with meaningfully different biases toward historical importance, recency, position, and consolidation.

Applying one rule family throughout a model does not imply that every layer retains the same positions. Layer-local attention signals can cause a shared rule to select different states at different layers. The relevant convention is that the identity of the rule used to make those selections is commonly fixed across layers. Representative formulations and evaluations therefore treat retention-rule families as standalone model-wide alternatives.

2.2 Component-Specific KV Cache Management

Adaptive allocation relaxes a different uniformity assumption by changing how much cache each component receives. PyramidKV uses a pyramidal layer-budget schedule, while CAKE derives layer preferences from spatial and temporal attention dynamics (Cai et al., 2025; Qin et al., 2025). D2O also incorporates layer-dependent capacity within its own eviction framework (Wan et al., 2025). Ada-KV reallocates capacity across heads and can augment different base eviction policies, while HeadKV assigns head-specific budgets using retrieval and reasoning importance (Feng et al., 2025; Fu et al., 2025). DynamicKV updates layer-wise retained-token counts, and LAVa derives a unified information-loss-based eviction metric together with dynamic head and layer budgets (Zhou et al., 2025; Shen et al., 2025). These methods show that layers and heads need not receive uniform capacity. Even when scoring and allocation are designed jointly, however, the retention decisions generally remain governed by a shared method-specific rule family.

A related line of work makes cache behavior heterogeneous across attention heads. FastGen profiles heads and selects head-level cache behaviors according to their attention patterns (Ge et al., 2024). RazorAttention and DuoAttention distinguish retrieval heads from local or streaming heads, preserving richer history for retrieval heads while compressing the remaining heads more aggressively (Tang et al., 2025; Xiao et al., 2025). These approaches establish the value of component-specific policy behavior. Their choice spaces are defined by framework-specific categories for cache behavior at the attention-head level within the corresponding eviction frameworks.

MoE-nD brings heterogeneous assignment to the layer level through multi-axis compression routing (Sun et al., 2026). It assigns each layer a tuple consisting of an eviction ratio, key precision, and value precision under a global budget. The routing scheme therefore defines the routed object as a compression configuration in its assignment space. This separates compression-parameter variation from variation in the retention algorithm.

Together, these studies establish the value of component-specific KV cache management through heterogeneous capacity allocation, head-level cache-behavior specialization, and layer-wise routing of compression parameters. This work studies a complementary axis, namely whether independently developed standalone eviction-rule families can be routed across Transformer layers. We use a common experimental interface to represent method identity and cache capacity as separate layer-wise factors, construct static layer- and phase-specific candidate plans, and evaluate method routing and capacity allocation under matched nominal retained-token capacity. Prefill and decoding define the deployment scope of each comparison.

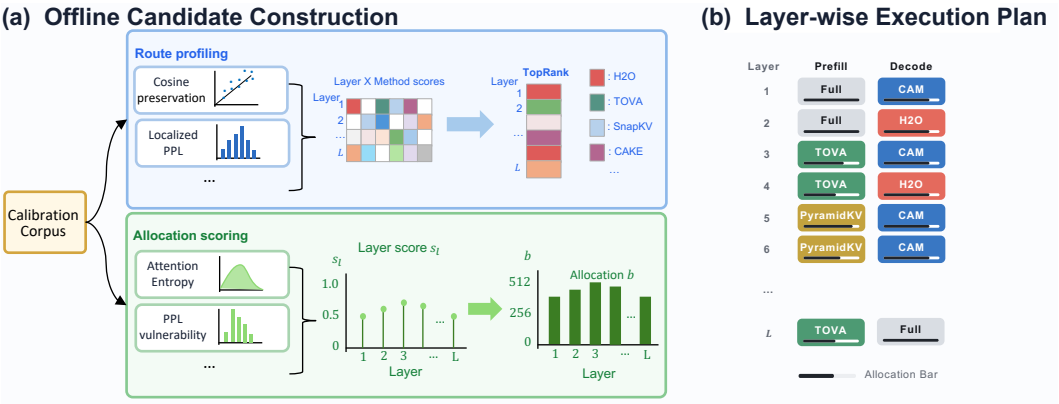

Figure 2: Overview of the controlled-plan construction. (a) Offline profiles from a calibration corpus produce layer-wise method rankings and allocation scores. (b) The resulting static execution plan assigns an operator and nominal capacity to each layer and inference phase; FULL denotes no additional eviction in that phase.

3 LAYER-WISE METHOD ROUTING AND CAPACITY ALLOCATION

We ask whether eviction-method identity is a meaningful layer-wise design variable for KV cache compression. Heterogeneous plans may differ in routing, allocation, deployment phase, or total budget; routes may also differ in method counts or layer placement. We therefore factorize the space and compare matched configurations. Offline profiles construct reproducible candidates in the exponentially large routing space, as summarized in Figure 2. The plans are frozen before inference, and POLYKV executes them through a common interface and runtime; it introduces neither a new retention rule nor an online route selector.

3.1 FACTORIZED DESIGN SPACE AND CONTROLLED CONTRASTS

Consider an autoregressive Transformer with L layers and prefill and decoding phases $\Phi = \{\mathrm{pre}, \mathrm{dec}\}$. Let $\mathcal{R}_\phi$ be the operators compatible with phase ϕ. Some compress the prompt cache after prefill; others update or sparsely access it during decoding. A layer-wise execution plan is

$$\Pi = \{(\{r_{l,\phi}\}_{\phi\in\Phi}, \{b_{l,\phi}\}_{\phi\in\Phi})\}_{l=1}^{L}, \tag{1}$$

where $r_{l,\phi} \in \mathcal{R}_\phi \cup \{\mathrm{FULL}\}$ is the operator at layer l in phase ϕ. For a capacity-controlled operator, $b_{l,\phi}$ is its nominal retained-token capacity; otherwise, it is undefined. An inactive phase uses FULL and performs no additional phase-specific eviction. Figure 2(b) shows this plan.

Deployment scope matters because prefill determines which prompt KV states reach decoding; decoding then accesses and updates them during generation. We denote a route's active phases by $\delta \in \{\mathrm{prefill}, \mathrm{decode}, \mathrm{both}\}$. Although Equation 1 permits different operators across phases, we evaluate prefill-only, decoding-only, and restricted both-phase routes. In a both-phase route, each layer uses the same routed operator in both phases, chosen from operators compatible with both.

For comparisons with nominal capacities at all active assignments, matched configurations fix the total $B_\phi = \sum_{l=1}^{L} b_{l,\phi}$ and average $\bar{b}_\phi = B_\phi/L$ within each active phase. The match on configured retained KV positions excludes measured memory, auxiliary state, latency, and runtime.

Under a fixed model, workload, deployment scope, and total nominal capacity, we compare four families. Homogeneous–uniform uses one operator and capacity across layers; allocation-only redistributes capacity without changing operator assignments; route-only varies operators under uniform capacity; and joint adds non-uniform allocation to the same heterogeneous route. Thus, routing is compared at fixed allocation and allocation at fixed operator assignments; the factorial comparison measures their interaction. The homogeneous–route contrast changes both method mixture and layer assignment, so it cannot alone identify a placement effect.

To isolate placement, define a completed route and its operator count as

$$\rho_\delta = \{\rho_{l,\delta}\}_{l=1}^{L}, \qquad N_r(\rho_\delta) = |\{l : \rho_{l,\delta} = r\}|.$$

A count-matched placement preserves every $N_r(\rho_\delta)$ and changes only which layers receive each operator. We compare the guided assignment with 99 unique random assignments from this space, excluding the guided assignment, while holding the model, workload, scope, allocation, and nominal budget fixed. The guided assignment is fixed before sampling and scoring. This control evaluates placement information beyond the method multiset within a bounded placement comparison.

For the budget sweep, we hold a completed route's operator identities and layer assignments fixed while varying $\bar{b}_\phi$, without reselecting the route. This traces one construction's quality–capacity behavior across budgets under a fixed route.

3.2 Profile-Based Route Construction

For scope δ, let $\mathcal{R}_\delta$ contain the operators compatible with every active phase; this space contains $|\mathcal{R}_\delta|^L$ assignments, making exhaustive evaluation infeasible. We use a heterogeneous calibration corpus $\mathcal{D}_{\text{cal}}$ comprising 20 sequences, each at most 2,048 tokens, from WikiText, C4, PG-19, and CodeParrot. FullKV supplies the reference behavior, and candidate operators are profiled at the common calibration capacity $b_{\text{cal}} = 256$. The resulting statistic $g_{l,\delta,r}$ ranks compatible operators at each layer. The deterministic profile-to-route mapping does not use scores from the placement controls, and the designated route is frozen before those controls are run.

The upper branch of Figure 2(a) shows two route profiles: representation similarity and perplexity degradation. For representation similarity, operator r is applied to every layer within scope δ. During free-running continuation generation, we pool each layer's attention-module outputs over the generated trajectory and calibration corpus. The cosine similarity between the resulting FullKV and candidate summaries is

$$C_{l,\delta,r} = \cos\left(\bar{h}_l^{\text{full}}, \bar{h}_l^{(\delta,r)}\right),$$

where larger values indicate closer pooled representations. Because r acts scope-wide and deeper layers may inherit upstream compression effects, this layer-indexed diagnostic provides a descriptive, intervention-level characterization of the resulting behavior.

The localized PPL profile covers the fixed set $\mathcal{R}_{\text{profile}}$: CAKE, CAM, D2O, H2O, PYRAMIDKV, SCISSORHANDS, SNAPKV, and TOVA. For each operator r, only layer l uses it in both prefill and decoding; all other layers retain FullKV. On teacher-forced continuation suffixes, we define

$$\Delta\text{PPL}_{l,r} = \text{PPL}_{l,r} - \text{PPL}_{\text{full}},$$

where smaller values indicate closer FullKV predictions. For a requested scope, we restrict this phase-shared profile to operators compatible with that scope; it does not estimate separate prefill and decoding preferences. Each profile independently yields one route.

For profile g, let $\mathcal{R}_\delta^g \subseteq \mathcal{R}_\delta$ denote the compatible operators covered by g. The constructor selects

$$\rho_{l,\delta} = \text{TopRank}_g\left(\{(r, g_{l,\delta,r}) \mid r \in \mathcal{R}_\delta^g\}\right), \tag{2}$$

where higher is better for cosine preservation and lower for PPL degradation. The selected operator is installed in every active phase, while inactive phases use FULL, yielding a prefill-only, decoding-only, or tied both-phase candidate. This independent layer-wise rule ignores cross-assignment interactions. We use each profile as a separate heuristic to construct a reproducible route, without searching for which profile best supports route selection.

3.3 Capacity Allocation as a Separate Factor

After fixing the route, the route-only plan assigns $b_{l,\phi} = \bar{b}_\phi$ to every capacity-controlled layer in each active phase. Allocation refinement keeps all operator assignments fixed and redistributes the same total using layer score s_l.

The lower branch of Figure 2(a) shows two allocation scores: attention entropy and compression-induced PPL degradation. Let $A_{h,q,k}^{(l)}(x)$ denote the FullKV attention probability from query q to

key k in head h at layer l on calibration sequence x, and let $\mathcal{H}$ denote Shannon entropy. We define

$$H_l = \mathbb{E}_{x,h,q}\Big[\mathcal{H}\Big(A^{(l)}_{h,q,:}(x)\Big)\Big], \qquad S^{\mathrm{ppl}}_l = \max_{r\in\mathcal{R}_{\mathrm{profile}}} \Delta\mathrm{PPL}_{l,r}.$$

Higher entropy indicates diffuse attention, so the heuristic assigns such layers more capacity; downstream evaluation tests whether this improves quality. Larger S^{ppl}_l values indicate layers more affected by the profiled interventions, so the allocator gives them more capacity. Because every operator is profiled at one capacity, this score measures compression vulnerability at the profiled capacity. Both scores are model-specific, phase-shared, and fixed before evaluation.

Given a score s_l, a fixed transformation T produces non-negative weights. Entropy allocation uses the identity, whereas PPL-derived allocation caps the score at the across-layer 95th percentile:

$$T(H_l) = H_l, \qquad T(S^{\mathrm{ppl}}_l) = \min\Big\{S^{\mathrm{ppl}}_l, Q_{0.95}(S^{\mathrm{ppl}})\Big\}.$$

The cap prevents one outlier from dominating the total. If the transformed minimum is non-positive, all values are shifted by $-\min_l T(s_l) + 10^{-6}$ before normalization. The provisional allocation is

$$w_l = \frac{T(s_l)}{\sum_{j=1}^{L} T(s_j)}, \qquad \tilde{b}_{l,\phi} = B_\phi w_l. \tag{3}$$

The allocator iteratively fixes layers below $b_{\min}$ at that minimum and redistributes the remainder among other layers in proportion to their weights. We use $b_{\min} = 32$ for entropy allocation and $b_{\min} = 48$ for PPL-derived allocation; the latter keeps SnapKV's capacity above its 32-token observation window. After nearest-integer rounding, residual units are assigned by fractional remainder, so $b_{l,\phi} \geq b_{\min}$ and $\sum_{l=1}^{L} b_{l,\phi} = B_\phi$. Applied to a homogeneous or heterogeneous route, the procedure yields an allocation-only or joint configuration without changing operator assignments. For the non-uniform-allocation budget sweep, we scale the completed 512-token vector to each target total and apply largest-remainder rounding; neither the score profile nor the route is recomputed.

3.4 Executing Controlled Plans with PolyKV

To compare methods developed as standalone alternatives, PolyKV preserves their scoring and retention logic through static configuration, a common operator interface, and phase- and layer-specific dispatch. Each adapter declares compatible phases and implements the handlers required by the original method for attention-kernel dispatch, prompt-cache compression, and incremental cache updates. Capacity-controlled adapters also expose $b_{l,\phi}$. These declarations define $\mathcal{R}_\phi$ and the intersection for tied both-phase candidates. Outside declared handlers, execution follows the standard attention path, and inactive phases use Full.

Each completed plan is serialized and loaded once. The runtime tracks the prefill-to-decoding transition and dispatches each layer according to Equation 1. Calibration is offline, and routing and allocation remain static during evaluation. Execution requires no model-weight changes, online profiling, route search, or policy adaptation.

4 Evaluation

The evaluation addresses three questions. First, under a matched nominal KV budget, can evidence for method routing be separated from evidence for non-uniform capacity allocation? Second, after fixing the method mixture, does the offline profile provide an informative ranking of sampled layer placements? Third, does favorable construction-level behavior recur across capacity constraints and on a second model, and what workload-specific boundaries appear? Our aim is not to identify a global optimum or an automatic route selector. We therefore distinguish primary controlled evidence on LongBench, supporting descriptive consistency across budgets and a second model, and a workload-specific RULER boundary. This organization keeps the conclusion narrow: the tested routing and placement space contains exploitable profile-guided structure, whereas the tested allocation refinements and reliable selection on unseen workloads remain unresolved.

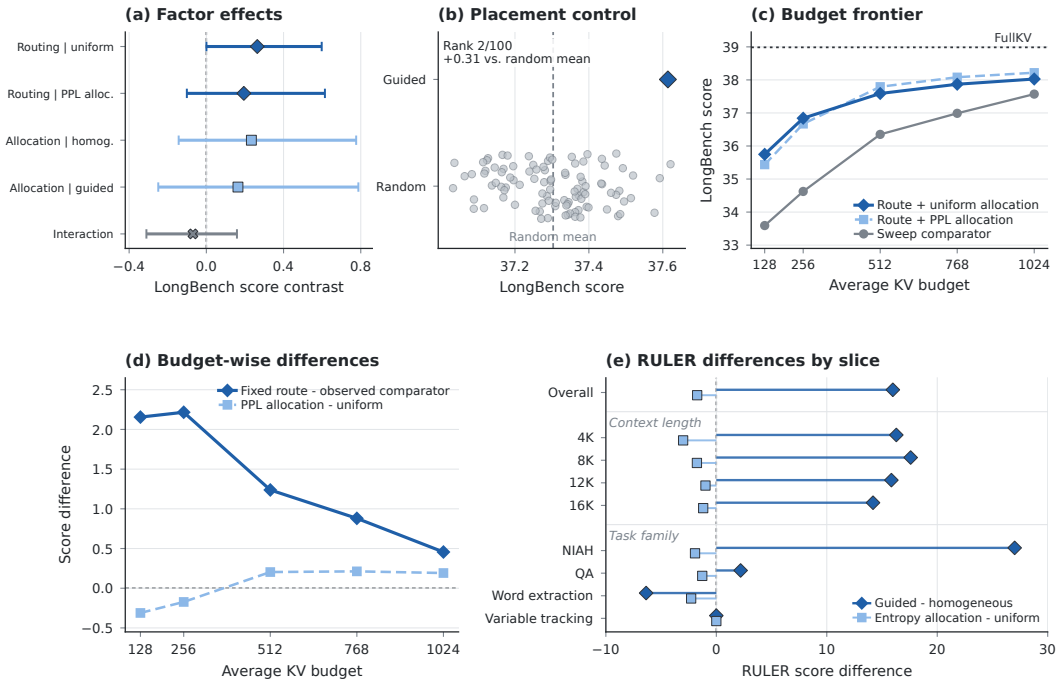

Figure 3: Five views of profile-guided method routing and capacity allocation. (a) Task-paired routing effects (guided cosine route minus homogeneous TOVA) under uniform and PPL-derived allocation, allocation effects (PPL-derived minus uniform) under homogeneous and guided routing, and their interaction on LongBench; bars are 95% bootstrap intervals. (b) LongBench scores of the frozen guided placement and 99 unique random placements with the same counts of three D2O and 29 TOVA layers. (c) LongBench scores across five average KV budgets for one fixed route with uniform or PPL-derived allocation, the strongest observed standalone result among nine baselines at each budget, and FullKV. (d) The uniform-allocation fixed route minus the budget-wise standalone reference, and PPL-derived minus uniform allocation within the fixed route; the reference is TOVA at 128 tokens and CAKE at 256–1024 tokens. (e) RULER differences for the guided uniform policy minus homogeneous SnapKV under uniform allocation, and for entropy-derived minus uniform allocation within the guided route, reported overall and by context length and task family. Only (a) reports bootstrap uncertainty; panels (b)–(e) are descriptive.

4.1 Experimental Setup and Evidence Roles

The primary controlled study uses LLaMA-3.1-8B-Instruct (Grattafiori et al., 2024) on LongBench (Bai et al., 2024) and reports the macro average over 21 tasks. A retrospective Qwen3-8B (Yang et al., 2025) check uses the same 21-task LongBench macro protocol to assess a uniquely closest exact-capacity pair. The check is a targeted comparison with a narrower scope than a second full evaluation. The RULER (Hsieh et al., 2024) case uses LLaMA-3.1-8B-Instruct and reports the official average over 13 tasks, four context lengths, and 52 task–length cells.

The framework integrates nine representative KV eviction methods. They are H2O, TOVA, CaM, PyramidKV, SnapKV, D2O, CAKE, Scissorhands, and StreamingLLM. FullKV provides the uncompressed reference. Every compressed comparison matches the nominal average retained-token budget within its protocol. This criterion controls configured retained KV positions and does not imply matched memory, auxiliary state, latency, or throughput. Appendix E reports measured steady KV-related memory and prefill/decode throughput for one fixed H2O/TOVA composition.

The LongBench factorial and placement studies use a cosine decode route frozen before the controlled runs, while the original-cohort sweep evaluates it across budgets. These studies provide the primary controlled evidence and the within-route capacity check, respectively. The Qwen comparison provides a retrospective second-model check, whereas RULER uses a separately constructed

workload-specific prefill policy to expose a workload boundary. Appendix Table 1 summarizes settings and evidence roles. Selection-free route choice on unseen workloads remains unresolved.

4.2 Controlled LongBench Evidence for Routing and Placement

We first ask whether a frozen profile-guided route improves over a homogeneous route under matched allocation rules, and whether non-uniform allocation provides an independent benefit. All four cells use decode-only compression and an average budget of 512 tokens per active layer. The routing factor compares homogeneous TOVA with the guided cosine route. The allocation factor compares uniform capacity with a PPL-derived allocation under the same total nominal capacity. Figure 3a reports the task-paired contrasts. Appendix Table 2 gives the complete cells and intervals.

Only the routing contrast under uniform allocation has an interval that excludes zero: the guided route gains 0.2638 points with a task-bootstrap interval of $[+0.0005, +0.5976]$, which only narrowly excludes zero. For this contrast, guided-route scores are higher on 11 tasks, unchanged on 2, and lower on 8. Despite this heterogeneity, all 21 leave-one-task-out macro contrasts remain positive, ranging from $+0.1285$ to $+0.3150$. Together, task-level variation and leave-one-out stability provide modest evidence that the routing profile is informative at the macro-average level relative to the homogeneous anchor across the evaluated task set. Other factorial intervals include zero, leaving the tested allocation refinement unresolved. Appendix Table 3 lists all paired differences.

The routing contrast changes method mixture and placement together. We therefore fix method counts at three D2O and 29 TOVA layers, changing only the D2O locations. The model, workload, decode-only scope, uniform allocation, and 512-token average budget remain fixed. Figure 3b compares the guided assignment with 99 unique random placements from the count-matched space.

Because the guided assignment was fixed before sampling, its upper-tail position yields a corrected one-sided Monte Carlo value of $p_{\mathrm{MC}} = 0.020$. Appendix Table 4 summarizes the sampled placement distribution. Within the finite sample of count-matched alternatives, the guided assignment's relative position provides finite-sample evidence that the placement signal contains information about relative placement quality across the evaluated assignment space.

Profile validation. Across the same 99 random placements, the frozen offline profile score is positively associated with downstream LongBench performance (tie-aware Spearman $\rho = 0.459$; corrected one-sided permutation $p = 0.0001$). Excluding the guided assignment, this association shows that the profile ranks performance within the sampled count-matched placement space, beyond one strong route. Appendix Figure 4 shows this relationship.

Together, the matched routing gain, upper-tail fixed placement, and profile–performance association across sampled count-matched placements provide the primary evidence for exploitable routing and placement structure. They do not identify a global optimum or imply task-universal improvement.

4.3 Supporting Consistency Across Capacity Constraints and a Second Model

We next test capacity consistency without route reselection by holding the cosine decode route and its layer assignments fixed from 128 to 1024 average tokens per layer. At every budget, the original sweep cohort evaluates the same nine standalone eviction baselines, so their per-budget maximum provides a clearly delimited strongest observed reference within this fixed cohort for that budget. Figures 3c and 3d report scores and differences.

Under uniform allocation, the fixed route exceeds the strongest observed standalone score at all five matched budgets, with margins ranging from $+0.46$ to $+2.22$ points. At 128, 256, and 512 tokens, it also matches or exceeds the strongest standalone reference at twice the nominal retained capacity. This comparison does not measure memory, latency, or throughput. Because the layer assignments remain fixed throughout the five tested constraints, the observed behavior supports construction-level consistency across multiple operating points within the evaluation. By contrast, the changing direction of the PPL-minus-uniform effect across budgets does not provide the tested allocation heuristic with the same consistent support as routing. Appendix Table 5 gives the full result matrix.

Second-model descriptive consistency. The retrospective audit rule in Appendix C.2 identifies a Qwen3-8B pair with FullKV prefill, decode-only compression, and identical entropy-derived layer

capacities with mean 512. The homogeneous H2O anchor scores 46.9619, while the routed variant, which changes two decode assignments from H2O to TOVA, scores 47.2333, a $+0.2714$ difference. The routed plan improves 13 tasks, ties on 3, and declines on 5. The median difference is $+0.1400$, and the descriptive paired-bootstrap interval is $[+0.0148, +0.5553]$. Because the audit rule was formalized retrospectively, this similar direction and magnitude support a strictly descriptive, model-specific second-model consistency check based on a within-model LongBench contrast reported here. The same exact pair changes the RULER average by only -0.0712; this observed near-zero difference does not support cross-workload transfer.

4.4 A Workload-Specific Construction and Its Boundary on RULER

RULER evaluates whether a separately profiled, workload-specific construction can reach a high-performing range under its own protocol, yielding a workload-specific performance characterization for this construction. It uses prefill-only compression at a 512-token average budget. Because SnapKV occupies 18 of the 32 layers in the frozen route, the factorial comparison uses homogeneous SnapKV as the dominant-method ablation anchor. Standalone TOVA is reported separately as the strongest homogeneous performance reference. Figure 3e reports overall, length, and task-family differences for the frozen protocol-matched comparisons. Appendix Table 7 reports all four cells and adds one newly completed homogeneous entropy cell; effects involving that cell are interpreted descriptively.

The large matched-anchor gain does not make the guided uniform construction the best observed policy: it is about 16 points above homogeneous SnapKV but 0.10 points below standalone TOVA overall. NIAH accounts for 32 of the 52 task–length cells and drives most of the aggregate improvement. Variable tracking remains at the floor under both the guided-uniform and homogeneous-SnapKV policies, so its zero contrast does not indicate robust parity. Together with the cross-family directions in Figure 3e, this concentration characterizes the matched-anchor advantage as specific to particular task families. The non-positive entropy-minus-uniform differences in the non-floor slices provide no supporting evidence for the tested allocation signal in this protocol.

Taken together, the primary LongBench controls support exploitable structure in method routing and placement, including an association between the frozen offline profile and performance within the sampled count-matched placement space. The fixed-route budget sweep and retrospective Qwen check add supporting construction-level consistency, but neither establishes automatic route selection or cross-workload transfer. The near-zero Qwen RULER difference and the workload-specific RULER case delimit the claim, while the tested allocation refinements remain unresolved. These experiments neither identify a global optimum nor establish universal superiority.

5 Conclusion

This work asks whether KV eviction-method identity should remain a model-wide choice. Our results indicate that method routing forms a meaningful layer-wise design dimension in the tested settings, with the clearest evidence coming from placement-sensitive structure.

The primary matched LongBench comparison provides modest macro-average support for routing over a homogeneous anchor, while the tested non-uniform allocation refinements remain unresolved. With method counts fixed, the guided assignment ranks second among 100 count-matched placements, and the frozen offline profile is positively associated with LongBench performance across that sampled space. A fixed route supports construction-level consistency by exceeding the strongest observed score among the same nine standalone baselines at all five matched nominal budgets. A retrospective Qwen comparison is a separate descriptive second-model check with a similar direction on LongBench. The near-zero Qwen RULER difference and the task-family concentration of a separately constructed RULER route limit cross-workload and task-universal claims.

These results do not identify a globally optimal route or establish reliable selection on unseen workloads. The fixed-composition measurements of steady KV-related memory, prefill throughput, and decode throughput reported by the separate system characterization in Appendix E leave end-to-end gains across routes or workloads unestablished. The evidence shows KV eviction methods should be studied as both model-wide alternatives and components whose identity and placement can vary by layer.

AI USE STATEMENT

Generative AI tools were used to assist with software-code implementation and with editing the manuscript for clarity and readability. AI-assisted code was reviewed and tested by the authors, and all AI-assisted manuscript changes were reviewed by the authors. The authors take responsibility for the correctness of the implementation and for the final content of this work, including any text, code, or other artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Sections 3.1–3.4 define the factorized plan, profiling signals, fixed-total allocator, and common execution interface. Section 4.2 and Appendices A–D document the evaluation settings, controlled comparisons, task-level results, placement analysis, and the Qwen and RULER boundary checks.

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 (Volume 1: Long Papers)*, pp. 3119–3137, 2024. doi: 10.18653/v1/2024.acl-long.172. URL https://aclanthology.org/2024.acl-long.172/.

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. In *Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=ayi7qezU87.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 276–286, 2019. doi: 10.18653/v1/W19-4828. URL https://aclanthology.org/W19-4828/.

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*, volume 38, pp. 113152–113188, 2025. doi: 10.52202/085713-3773. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/a40ff56daab9f4808b1e18350c8a11ce-Abstract-Conference.html.

Yu Fu, Zefan Cai, Abedelkadir Asi, Wayne Xiong, Yue Dong, and Wen Xiao. Not all heads matter: A head-level KV cache compression method with integrated retrieval and reasoning. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/f649556471416b35e60ae0de7c1e3619-Abstract-Conference.html.

Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model tells you what to discard: Adaptive KV cache compression for LLMs. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=uNrFpDPMyo.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? In *Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=kIoBbc76Sy.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3651–3657, 2019. doi: 10.18653/v1/P19-1356. URL https://aclanthology.org/P19-1356/.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 22947–22970, 2024. doi: 10.52202/079017-0722. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/28ab418242603e0f7323e54185d19bde-Abstract-Conference.html.

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, pp. 52342–52364, 2023. doi: 10.52202/075280-2279. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/a452a7c6c463e4ae8fbdc614c6e983e6-Abstract-Conference.html.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/hash/2c601ad9d2ff9bc8b282670cdd54f69f-Abstract.html.

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*, pp. 18724–18741, 2024. doi: 10.18653/v1/2024.emnlp-main.1043. URL https://aclanthology.org/2024.emnlp-main.1043/.

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

Yiqun Shen, Song Yuan, Zhengze Zhang, Xiaoliang Wang, Daxin Jiang, and Cam-Tu Nguyen. LAVa: Layer-wise KV cache eviction with dynamic budget allocation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 13672–13692, 2025. doi: 10.18653/v1/2025.findings-emnlp.737. URL https://aclanthology.org/2025.findings-emnlp.737/.

Libo Sun, Peixiong He, Po-Wei Harn, and Xiao Qin. MoE-nD: Per-layer mixture-of-experts routing for multi-axis KV cache compression. *arXiv preprint arXiv:2604.17695*, 2026. URL https://arxiv.org/abs/2604.17695.

Hanlin Tang, Yang Lin, Jing Lin, Qingsen Han, Danning Ke, Shikuan Hong, Yiwu Yao, and Gongyi Wang. RazorAttention: Efficient KV cache compression through retrieval heads. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tkiZQlL04w.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4593–4601, 2019. doi: 10.18653/v1/P19-1452. URL https://aclanthology.org/P19-1452/.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5797–5808, 2019. doi: 10.18653/v1/P19-1580. URL https://aclanthology.org/P19-1580/.

Zhongwei Wan, Xinjian Wu, Yu Zhang, Yi Xin, Chaofan Tao, Zhihong Zhu, Xin Wang, Siqi Luo, Jing Xiong, Longyue Wang, and Mi Zhang. D2O: Dynamic discriminative operations for efficient long-context inference of large language models. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=HzBfoUdjHt.

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. URL https://openreview.net/forum?id=NG7sS51zVF.

Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, Junxian Guo, Shang Yang, Haotian Tang, Yao Fu, and Song Han. DuoAttention: Efficient long-context LLM inference with retrieval and streaming heads. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=cFu7ze7xUm.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Yuxin Zhang, Yuxuan Du, Gen Luo, Yunshan Zhong, Zhenyu Zhang, Shiwei Liu, and Rongrong Ji. CaM: Cache merging for memory-efficient LLMs inference. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 58840–58850, 2024. URL https://proceedings.mlr.press/v235/zhang24n.html.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 34661–34710, 2023. doi: 10.52202/075280-1506. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/6ceefa7b15572587b78ecfcebb2827f8-Abstract-Conference.html.

Xiabin Zhou, Wenbin Wang, Minyan Zeng, Jiaxian Guo, Xuebo Liu, Li Shen, Min Zhang, and Liang Ding. DynamicKV: Task-aware adaptive KV cache compression for long context LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 8042–8057, 2025. doi: 10.18653/v1/2025.findings-emnlp.426. URL https://aclanthology.org/2025.findings-emnlp.426/.

A Experimental Setup and Statistical Procedures

This appendix follows the evidence roles in the main paper. We first summarize the evaluation settings and statistical procedures, then report the primary LongBench controls, supporting consistency across cache budgets and a second model, the RULER boundary analysis, and a system-level characterization of one fixed composition. Every inactive inference phase uses FullKV.

Table 1: Summary of the evaluation settings and evidence roles. All compressed comparisons match total nominal retained-token capacity within their stated protocol.

Study	Evaluation setting	Route or anchor	Allocation and mean capacity	Evidence role
Matched factorial	LLaMA-3.1-8B-Instruct / LongBench; decode	Guided cosine versus homogeneous TOVA	Uniform or PPL-derived; mean 512	Separates routing, allocation, and their interaction
Count-matched placements	LLaMA-3.1-8B-Instruct / LongBench; decode	Three D2O and 29 TOVA assignments	Uniform; mean 512	Isolates layer placement at fixed method counts
Retrospective cross-model check	Qwen3-8B / LongBench; decode	Two-layer H2O/TOVA route versus homogeneous H2O	Same entropy-derived vector; mean 512	Tests within-model consistency under an exact capacity match
Fixed-route sweep	LLaMA-3.1-8B-Instruct / LongBench; decode	One frozen cosine route	Uniform or scaled PPL-derived; mean 128–1024	Tests one construction across capacity constraints
Matched RULER case	LLaMA-3.1-8B-Instruct / RULER; prefill	Guided PPL route versus homogeneous SnapKV	Uniform or entropy-derived; mean 512	Tests a second construction and localizes its task-family boundary

Statistical procedures. All evaluated plans are constructed offline and remain fixed during inference; no route search or capacity adaptation uses downstream evaluation scores. For the LongBench placement control, the guided assignment is fixed before evaluating 99 unique random count-matched assignments from the remaining placement space. For profile validation, each placement receives the frozen score $S(\rho) = \sum_{\ell \in D(\rho)} (C_{\ell,\mathrm{D2O}} - C_{\ell,\mathrm{TOVA}})$. Here, $D(\rho)$ denotes the layers assigned D2O by placement ρ, and $C_{\ell,r}$ is the frozen cosine-preservation score for method r at layer ℓ. The primary statistic is a tie-aware Spearman rank correlation between $S(\rho)$ and the LongBench macro score over the 99 random placements. Its corrected one-sided permutation value is computed by shuffling the 99 downstream scores 10,000 times. The guided assignment is excluded from the primary calculation and included only in the secondary descriptive analysis. For the LongBench factorial, confidence intervals use 10,000 paired bootstrap replicates over the 21 LongBench tasks.

B Complete Controlled LongBench Evidence

The factorial below separates the guided routing policy from PPL-derived capacity redistribution. The routing contrast changes both method counts and layer locations, whereas the subsequent placement control holds method counts fixed at three D2O and 29 TOVA layers and changes only their locations.

B.1 Routing and Allocation

Cell values are 21-task macro averages, and intervals use 10,000 paired bootstrap replicates over tasks. Routing contrasts are guided minus homogeneous, while allocation contrasts are PPL-derived minus uniform.

Table 2: Complete matched LongBench factorial for LLaMA-3.1-8B-Instruct at a 512-token average KV budget.

Routing policy	Uniform allocation	PPL allocation
Homogeneous TOVA (decode-only)	37.3500	37.5833
Guided cosine route	37.6138	37.7776

Contrast	Estimate	Task-bootstrap 95% CI
Routing under uniform	+0.2638	$[+0.0005, +0.5976]$
Routing under PPL	+0.1943	$[-0.1010, +0.6143]$
Allocation under homogeneous	+0.2333	$[-0.1433, +0.7762]$
Allocation under guided	+0.1638	$[-0.2490, +0.7876]$
Routing $\times$ allocation	-0.0695	$[-0.3105, +0.1586]$

Task-level robustness. For the uniform-allocation routing contrast, the 21 paired task differences have mean $+0.2638$ and median $+0.0400$; guided routing improves 11 tasks, is unchanged on 2, and declines on 8. All 21 leave-one-task-out macro contrasts remain positive, ranging from $+0.1285$ to $+0.3150$. The largest individual contribution is `passage_retrieval_zh` at $+2.97$, which accounts for 53.6% of the summed task-level contrast. Omitting this task reduces the macro improvement to $+0.1285$ but does not reverse its direction. Thus, the magnitude is concentrated, while no single task overturns the positive macro-average contrast.

Table 3: Complete task-level differences for routing under uniform allocation in the matched LongBench study. Δ_{route} is the guided cosine route minus homogeneous TOVA; positive values favor guided routing.

Task	Δ_{route}	Task	Δ_{route}
narrativeqa	$+0.72$	vcsum	$+0.23$
qasper	$+0.69$	trec	$+0.00$
multifieldqa_en	$+0.81$	triviaqa	$+0.44$
multifieldqa_zh	$+0.73$	samsum	$+0.04$
hotpotqa	$+0.32$	lsht	$+0.00$
2wikimqa	-0.02	passage_count	$+0.29$
musique	-0.40	passage_retrieval_en	-0.76
dureader	$+0.29$	passage_retrieval_zh	$+2.97$
gov_report	-0.15	lcc	-0.08
qmsum	-0.23	repobench-p	-0.31
multi_news	-0.04		

B.2 Count-Matched Placement and Profile Validation

Table 4: Summary of the count-matched placement control on LongBench. Every assignment contains three D2O layers and 29 TOVA layers under uniform allocation at a 512-token average budget. The control pool contains 99 unique random assignments and excludes the guided assignment.

Statistic	Value
Random controls	99
Guided score	37.6138
Random mean (sample SD)	37.3034 (0.1303)
Random range	[37.0324, 37.6210]
Guided minus random mean	$+0.3105$
Random $\geq$ guided	1
Guided rank	2/100

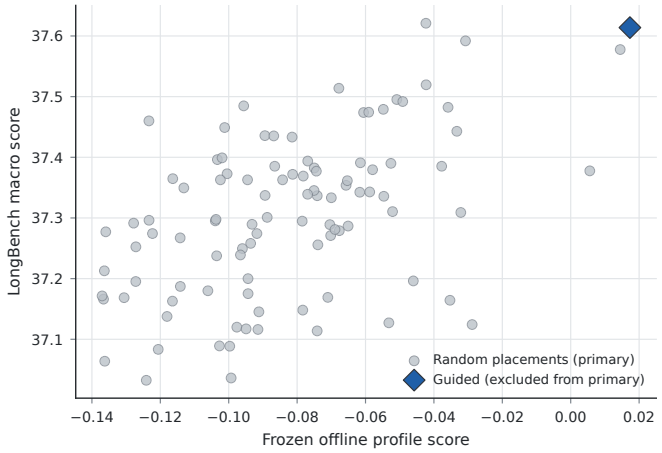

Figure 4: Frozen-profile score versus downstream quality for count-matched placements. Across the 99 random placements, the primary tie-aware Spearman coefficient is $\rho = 0.459$ with corrected one-sided permutation $p = 0.0001$. The guided placement is shown separately and excluded from that calculation; including it gives the secondary coefficient $\rho = 0.475$.

C Supporting Consistency Across Budgets and Models

C.1 Fixed-Route Robustness Across Cache Budgets

The complete matrix reports 21-task macro averages from raw LongBench outputs; higher is better. It holds the guided cosine route and its layer assignments fixed from 128 to 1024 average tokens per layer. The uniform variant changes only the common capacity, while the PPL-derived variant scales the fixed 512-token profile to the target total. FullKV is budget-free and shown once, while Quest is excluded because it is not an eviction method. The same nine standalone eviction baselines appear at every budget under their original sweep protocols; in particular, TOVA is active in both prefill and decode, whereas the matched factorial in Table 2 uses decode-only TOVA. Their per-budget maximum defines the baseline envelope used in the main-text comparison.

Table 5: Complete LLaMA-3.1-8B-Instruct LongBench budget sweep across five mean nominal capacities.

Method / configuration	128	256	512	768	1024
FullKV			38.9857		
Single baselines					
CAKE	32.1214	34.6248	36.3514	36.9890	37.5705
CaM	29.5862	31.4490	32.7248	33.9633	34.5600
D2O	11.9476	11.5695	11.6457	11.9238	11.8486
H2O	29.0733	31.3352	32.7910	33.9533	34.7167
PyramidKV	31.7910	34.1567	35.8090	36.6376	37.1414
Scissorhands	27.1529	29.2233	31.1881	32.0886	33.2905
SnapKV	32.2457	34.5805	35.9352	36.5167	36.9905
StreamingLLM	17.6043	16.8233	15.8124	15.5871	15.1319
TOVA	33.5924	34.4814	35.6962	36.4300	37.0571
Fixed layer-wise route variants					
Route-only	35.7476	36.8410	37.5890	37.8690	38.0267
Route + PPL allocation	35.4362	36.6681	37.7929	38.0814	38.2181

C.2 Retrospective Cross-Model Consistency on Qwen3-8B

We select the Qwen pair by identifiability rather than downstream rank. Among 43 Qwen LongBench runs, the audit retains 22 Qwen-family-verified eviction recipes and eight exact-capacity non-routed–routed pairs. The reported pair is the unique minimum-change comparison, differing in two decode assignments while keeping every layer capacity fixed. Because the rule was formalized after the completed suite was inspected, this remains a retrospective consistency check.

On LongBench, the routed plan improves the macro average by 0.2714 points, with 13 wins, 3 ties, and 5 losses across the 21 tasks. Its similar direction and magnitude to the matched LLaMA contrast provide descriptive construction-level consistency on a second model family, not confirmatory replication or transfer of a common policy. The same pair is essentially neutral on RULER, restricting the evidence to within-model LongBench consistency and providing no support for cross-workload transfer. In Table 6, arrows run from homogeneous to routed, and W/T/L counts routed wins, ties, and losses over 21 LongBench tasks or 52 RULER task–length cells. The LongBench interval resamples 21 tasks, while the RULER interval resamples 13 task clusters across four context lengths; both use 10,000 bootstrap replicates.

Table 6: Retrospective Qwen3-8B matched comparison and cross-workload boundary under identical layer-wise capacities.

Workload	Matched scores	Δ	Descriptive 95% CI	W/T/L
LongBench	$46.9619 \rightarrow 47.2333$	$+0.2714$	$[+0.0148, +0.5553]$	13/3/5
RULER	$24.3194 \rightarrow 24.2483$	-0.0712	$[-0.2315, +0.0812]$	14/21/17

D MATCHED RULER CASE STUDY AND TASK-FAMILY BOUNDARY

The RULER study uses a separately constructed guided PPL-prefill route. Table 7 places its matched SnapKV anchor in the context of FullKV and standalone TOVA, then reports the complete routing-by-allocation cells, length slices, and task-family slices. The factorial abbreviates homogeneous SnapKV as H, guided routing as G, uniform allocation as U, and entropy-derived allocation as E. In the effect panel, Δ_R is guided minus homogeneous, Δ_E is entropy-derived minus uniform, and $\Delta_{R\times E} = \Delta_{R|E} - \Delta_{R|U}$.

The aggregate routing gain is concentrated in NIAH. QA improves slightly, word extraction regresses, and variable tracking remains at the floor. Entropy allocation lowers the guided score in every non-floor family and at every reported context length. Three cells are frozen protocol-matched anchors; the homogeneous-entropy cell was completed separately, so effects involving that cell remain descriptive.

Table 7: Reference context and complete RULER factorial scores and effects.

Reference	Score
FullKV	83.01
Homogeneous TOVA	58.54

Slice	H–U	H–E	G–U	G–E
All cells	42.46	41.40	58.44	56.71
4K	48.57	46.54	64.86	61.86
8K	40.95	40.10	58.54	56.79
12K	41.31	40.28	57.16	56.18
16K	39.03	38.69	53.21	52.02
NIAH	44.76	43.25	71.77	69.84
QA	70.33	69.70	72.53	71.25
Word extraction	26.63	26.41	20.28	18.00
Variable tracking	0.00	0.00	0.00	0.00

Slice	$\Delta_{R\|U}$	$\Delta_{R\|E}$	$\Delta_{E\|H}$	$\Delta_{E\|G}$	$\Delta_{R\times E}$
All cells	+15.98	+15.31	-1.06	-1.73	-0.67
4K	+16.30	+15.32	-2.03	-3.01	-0.98
8K	+17.59	+16.69	-0.85	-1.76	-0.90
12K	+15.85	+15.90	-1.03	-0.98	+0.05
16K	+14.19	+13.33	-0.33	-1.19	-0.86
NIAH	+27.01	+26.59	-1.51	-1.93	-0.42
QA	+2.20	+1.55	-0.63	-1.28	-0.65
Word extraction	-6.36	-8.41	-0.22	-2.28	-2.05
Variable tracking	0.00	0.00	0.00	0.00	0.00

E System-Level Performance and Memory Characterization

To complement the controlled quality evidence, we characterize one fixed H2O/TOVA composition at the implementation level. The configuration uses LLaMA-3.1-8B-Instruct on a single NVIDIA H200 GPU with batch size 1, prompt lengths from 4K to 32K, and 128 generated tokens. H2O is assigned to layers 0–15 and TOVA to layers 16–31, with every layer retaining 512 tokens. H2O-512, TOVA-512, and PolyKV therefore retain 16,384 layer–token positions, corresponding to 64.00 MiB of exact KV state. Steady memory includes the retained KV state and persistent eviction state and excludes model weights. Table 8 reports measured prefill throughput, decode throughput, and steady KV-related memory.

Table 8: System-level measurements for the fixed H2O/TOVA composition on LLaMA-3.1-8B-Instruct.

Context	Method	Prefill tok/s ↑	Decode tok/s ↑	Steady MiB ↓
4K	FullKV	28,686.71	42.58	529.02
4K	H2O-512	10,113.81	25.59	65.14
4K	TOVA-512	17,130.04	29.58	65.14
4K	PolyKV	13,697.16	27.53	65.14
8K	FullKV	27,289.50	42.26	1,041.14
8K	H2O-512	6,165.08	26.67	65.27
8K	TOVA-512	16,006.73	30.46	65.27
8K	PolyKV	8,962.82	27.91	65.27
16K	FullKV	22,886.05	40.31	2,065.39
16K	H2O-512	3,193.75	26.69	65.52
16K	TOVA-512	10,212.05	30.62	65.52
16K	PolyKV	4,866.39	28.21	65.52
32K	FullKV	17,904.96	25.45	4,113.89
32K	H2O-512	1,705.67	26.22	66.02
32K	TOVA-512	6,054.22	30.08	66.02
32K	PolyKV	2,662.11	26.05	66.02

Across 4K–32K contexts, the fixed composition keeps measured steady memory at 65.14–66.02 MiB, whereas FullKV grows from 529.02 to 4,113.89 MiB. The FullKV-to-compressed steady-memory ratio therefore grows from $8.1\times$ at 4K to $62.3\times$ at 32K. Under the same 512-token-per-layer capacity, the same compose-consistent configuration achieves a LongBench macro average of 37.31 over 21 tasks, compared with 36.06 for both homogeneous H2O-512 and homogeneous TOVA-512. These three quality scores share evaluation snapshot `12493d2` and contextualize only the fixed configuration in Table 8. They are not pooled with later reruns.

The composition's runtime closely tracks the envelope of its constituent implementations rather than introducing a new dominant cost. At every measured context length, its prefill throughput is higher than homogeneous H2O and lower than homogeneous TOVA. Its decode throughput lies between the two through 16K and is within 0.7% of H2O at 32K. PolyKV prioritizes faithful composition through a common operator interface and layer- and phase-specific dispatch. The current implementation does not use kernels specialized for a composed route, leaving fused dispatch and method-specific operations as direct optimization opportunities. Thus, the current system-level benefit is higher LongBench quality at a bounded steady KV footprint, rather than universal latency acceleration.