PERIODIC WEAK SPOTS: PHASE SENSITIVITY FROM CHUNKED KV-CACHE COMPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Chunked KV-cache compression reduces the memory and attention costs of long-context inference by compressing windows of consecutive tokens into fewer cache entries at a fixed stride. Such compression also introduces a new positional coordinate: a token's *phase*, or its position relative to compression-window boundaries. We uncover a systematic asymmetry in models using such compression: the same information can be easy to retrieve at one phase and difficult at another. We call this periodic variation in retrieval performance *phase sensitivity*. In the DeepSeek-V4 family, long-context retrieval accuracy can differ by up to 40 percentage points across phases, revealing periodic weak spots that average benchmark scores can conceal. To investigate this behavior, we pretrain a family of transformers from scratch across multiple KV-compression designs, reproducing phase sensitivity across the tested variants. Mechanistic analysis using causal interventions in these models and DeepSeek-V4 reveals *phase specialization*: different attention components contribute asymmetrically to retrieving information at different source phases. We further analyze idealized retrieval models, showing how gradient flow dynamics may favor sharp phase specialization. Evaluating models with chunked KV-cache compression therefore requires measuring across compression phases: high average accuracy can coexist with systematic positional failures.

1 INTRODUCTION

Long-context inference in transformers can be bottlenecked by the KV cache, whose memory and attention costs grow with context length. A practical remedy, used in models such as DeepSeek-V4 (DeepSeek-AI, 2026a), is *chunked KV-cache compression*: the model groups consecutive tokens into fixed-size windows, compresses each window into fewer cache entries, and uses these summaries for long-context retrieval (Rae et al., 2020; Wang & Xiao, 2024). The compression happens as context is processed, before a later query specifies what information will need to be recalled. Each summary is therefore query-agnostic and must preserve information that may be needed later.

This design imposes a periodic structure on the context. A new compression window begins every S tokens, where S is the compression stride. Each token thus has a new coordinate relative to these boundaries beyond its absolute position. We call this coordinate its *phase*, defined as its position modulo S. Shifting the input by a few tokens can change which tokens are compressed together without changing their content or relative positions. The same information may thus be compressed differently depending on its phase, and hence retained with different fidelity.

In LLMs trained with chunked KV-cache compression, we find that retrieval performance varies sharply and periodically with the phase of the source information, a failure mode we term *phase sensitivity* (Section 2). A motivating example comes from asking DeepSeek-V4 to complete its own inference code (Figure 1). When we prepend a decorative comment and vary only its length, the model's preferred completion of one token flips with a period of four tokens, matching its compression stride. We then measure retrieval directly through a controlled needle-in-a-haystack evaluation (Figure 2). At 128K context, retrieval accuracy in the DeepSeek-V4 family can differ by up to 40 percentage points across phases. To the best of our knowledge, this is the first report of such internal periodic variation, which average benchmark scores can conceal.

To test whether chunked KV compression is the source of such variation, we pretrain from scratch a broad family of transformers with the Qwen3-0.6B architecture (Qwen Team, 2025) as backbone and

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.

KV compression as the only substantial architectural modification (Section 3). Every compressed model we evaluate shows periodicity matching the compression stride, including models that simply average token representations within each window. By contrast, full-attention baselines have no comparable periodicity. A natural first explanation is the window boundary itself. In our needle-in-a-haystack task, each key is immediately followed by its value, so with non-overlapping windows a key–value pair either falls inside one window or is split across two, and a split pair is plausibly harder to retrieve. However, boundaries alone do not account for the full periodic pattern: accuracy also varies widely across phases that keep the key and its value within a single window. This suggests a deeper look into how the model writes information into compressed memory and reads it back.

Causal interventions on KV heads and layers support this view: retrieval at different phases depends differently on different heads, a systematic pattern we call *phase specialization* (Section 4.1). The pattern appears in both our pretrained models and DeepSeek-V4. To examine how such specialization can develop, we study compression gates, which determine how much each within-window position contributes to a compressed entry (Section 4.2). In a pretrained model whose gates can learn separate positional preferences, some heads develop persistent preferences during pretraining that recur across inputs. Their intervention effects are strongest near the phases they favor, supporting a division of retrieval work among the examined heads.

The emergence of these preferences suggests that training can favor selective retention even when future queries are unknown. We examine this possibility in an idealized induction task with compressed attention (Section 5), where mixing information in the compression window can weaken the match between a compressed key and a later query. We show that the optimal compressor favors particular phases, and gradient flow drives gates toward persistent preferences. Together, these findings show that average accuracy is not enough to evaluate models with chunked KV-cache compression. Their reliability should also be measured across phases, for example by shifting the input by a few tokens while keeping the queried information fixed.

2 Stride-Periodic Behavior in DeepSeek-V4 and DeepSeek-V4.1

2.1 A Periodically Flipping Code-Completion Example

We begin by asking DeepSeek-V4-Flash-Base to complete DeepSeek-V4's official inference code (Figure 1a). The prompt ends at **`FP`** inside a type cast in an FP8 quantization function, and the original source continues with "**`8`**". The surrounding code favors this continuation. The function quantizes to FP8, while the outer cast already converts to **`compute_dtype`**, which is FP32. Completing "**`32`**" would duplicate the outer cast and skip the rounding step.

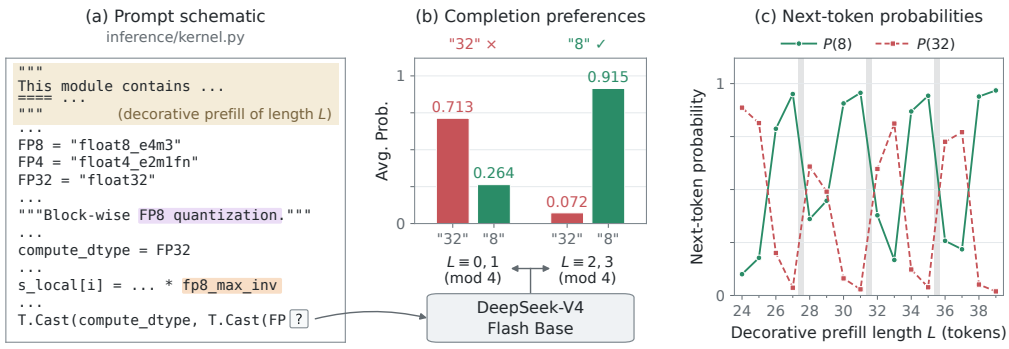

Figure 1: **Next-token predictions by DeepSeek-V4-Flash-Base on a code snippet.** The prediction flip periodically as a variable-length decorative filler shifts the input. (a) The decorative filler is a docstring whose length L varies by the number of "**`=`**" characters. The ground-truth completion is "**`8`**", supported by the highlighted components in the context. (b) Averaged over the eight lengths in each group, the model places higher probability on the incorrect "**`32`**" when $L \bmod 4 \in \{0, 1\}$ and on the correct "**`8`**" when $L \bmod 4 \in \{2, 3\}$. (c) The preferred completion switches every two tokens and the pattern repeats every four, matching DeepSeek-V4's compression stride.

To test whether this prediction depends on position, we prepend a decorative docstring and vary its length by changing the number of repeated "`=`" characters on its second line. The code is unchanged, and each additional filler token shifts the code and the completion position by one. Across 16 filler lengths L from 24 to 39 tokens, the top-1 prediction flips periodically (Figure 1b,c). In particular, the model predicts `8` when $L \bmod 4 \in \{2, 3\}$ and `32` at the remaining eight lengths, and $P(\texttt{8}) - P(\texttt{32})$ ranges from -0.79 to 0.95. Changing only the docstring length thus reverses the preferred continuation. The reversals repeat every four tokens, matching the stride $S = 4$ of DeepSeek-V4's compressed sparse attention (CSA).

This match suggests that the prediction depends on where the code falls relative to the compression windows. The reversals are not limited to one filler. In all four filler families we tested, most predictions follow the same four-token pattern (Appendix B.1), and the post-trained DeepSeek-V4-Flash-0731 and DeepSeek-V4.1-Flash (DeepSeek-AI, 2026b) also show periodic reversals under several filler sweeps, matching their compression strides ($S = 4$ and $S = 2$ respectively). In contrast, DeepSeek-V3.1-Base, without chunked KV compression, ranks `32` above `8` at only 4 of the same 64 inputs, with very small margins (Appendix B.2). We give the exact inputs, results for the other models, and two further code-completion examples in Appendix B.

2.2 Measuring Phase Sensitivity in Retrieval

To measure this effect more systematically, we turn to a controlled key–value based retrieval task in which we can measure the model's capability of retrieving the same piece of information, but at different relative positions with respect to the compression boundaries. For compression stride S, we call $t \bmod S$ the *phase* of a key-value pair starting at position t, and we say that retrieval is *phase-sensitive* when certain retrieval metric, such as accuracy or the probability assigned to the correct answer, changes systematically with the phase of the source.

We evaluate the base and post-trained versions of DeepSeek-V4-Flash and DeepSeek-V4-Pro, as well as the post-trained DeepSeek-V4.1-Flash, on single-token needle-in-a-haystack (NIAH) retrieval. Each 128K-token prompt holds 16k key–value pairs and asks for the value of a source key near its middle (Figure 2a). We sort the prompts into eight residue groups by the source key's position modulo eight, which covers two stride cycles for DeepSeek-V4 and four for DeepSeek-V4.1. All groups share the same prompt length, query position, and key–value bindings, and we choose their filler lengths so that the mean and variance of the source key's absolute position are the same in every group. We defer details about the construction of the retrieval dataset to Appendix C.1.

In Figure 2, we can see that the base checkpoints are sharply phase-sensitive, with maximum gaps of up to 40.2 percentage points. Post-training raises accuracy and narrows these gaps, but the gaps of the post-trained DeepSeek-V4 models remain large. DeepSeek-V4.1-Flash has the smallest gap, yet all four of its even residue groups remain above all four odd ones. Average accuracy alone can conceal these differences across phases.

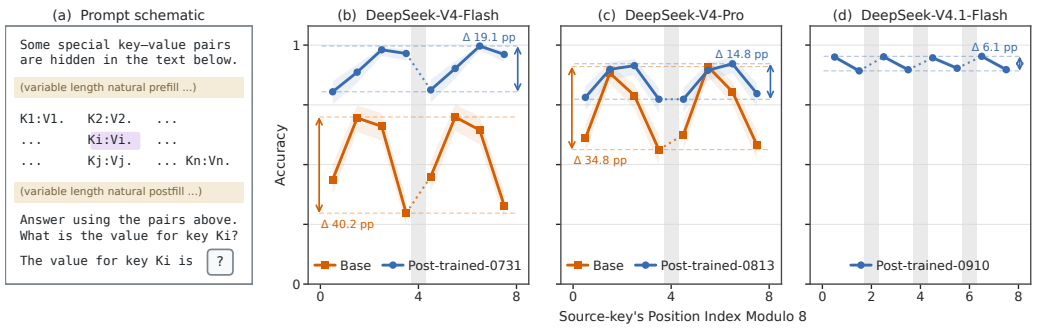

Figure 2: **NIAH retrieval accuracy of DeepSeek-V4 family for key-value pairs with different offsets.** All five models exhibit *phase sensitivity*, with a period of 2 or 4 matching the CSA stride. An answer counts as correct if, after stripping whitespace, it begins with the gold value. The marker for residue r of the target key's position modulo 8 sits between ticks r and $r{+}1$. Gray bands mark stride boundaries, shading shows two-sided 95% Wilson intervals, and Δ is the best-to-worst gap.

3 CONTROLLED PRETRAINING STUDIES

DeepSeek-V4 differs from a standard transformer in many ways besides chunked KV compression, and we cannot retrain it without compression to see whether the periodicity disappears. To test whether chunked compression is the source of phase sensitivity, we pretrain a Qwen3-derived family of models from scratch in which chunked compression is the only substantial architectural change, together with full-attention baselines trained the same way. We also vary how compression is performed to see which aspects of phase sensitivity, such as its period, depend on that design. These models also support the mechanistic analyses in Section 4.

3.1 A FAMILY OF KV COMPRESSION KERNELS

Each kernel compresses a window into one key and one value per KV head. A *payload map* sets what each token contributes, and a *compression gate* sets how strongly it enters the summary.

For one head, let $\boldsymbol{h}_{j,i} \in \mathbb{R}^d$ be the hidden state at offset $0 \le i < W$ in a window j of size W. For branch $B \in \{K, V\}$, the payload map, gate map, and positional bias have shapes $\boldsymbol{C}_i^B \in \mathbb{R}^{d_h \times d}$, $\boldsymbol{Z}_i^B \in \mathbb{R}^{d_g \times d}$, and $\boldsymbol{b}_i^B \in \mathbb{R}^{d_g}$. Here d is the hidden width, d_h is the head dimension, and d_g is the gate dimension. For the key branch, we compute gate logits

$$\boldsymbol{s}_{j,i}^K = \boldsymbol{Z}_i^K \boldsymbol{h}_{j,i} + \boldsymbol{b}_i^K. \tag{1}$$

A softmax over the window turns these logits into gate scores, which weight the payloads:

$$\boldsymbol{\alpha}_{j,i}^K = \frac{\exp(\boldsymbol{s}_{j,i}^K)}{\sum_{u=0}^{W-1} \exp(\boldsymbol{s}_{j,u}^K)}, \qquad \widehat{K}_j = \sum_{i=0}^{W-1} \boldsymbol{\alpha}_{j,i}^K \odot \left(\boldsymbol{C}_i^K \boldsymbol{h}_{j,i}\right). \tag{2}$$

Scalar gates ($d_g = 1$) weight all payload dimensions together, while vector gates ($d_g = d_h$) weight them separately. Softmax acts along the offset dimension, and $\odot$ broadcasts scalar scores or multiplies element-wise. The value branch is analogous.

We vary this family to test candidate explanations of phase sensitivity. Our reference model has one KV head per layer and uses non-overlapping eight-token windows with scalar gates. Window size W sets how many tokens an entry summarizes, and stride S sets how often an entry is produced, so the reference is W8/S8. Varying them separately shows whether the period follows the window or the stride, and whether overlapping windows ($S < W$), in which a token appears at several offsets while its phase stays fixed, remove the effect. Replacing learned gates with uniform averaging, which fixes each gate score at $1/W$, tests whether phase sensitivity requires learned gates. We describe parameter-sharing variants in Appendix D. The family also includes DeepSeek-V4's CSA kernel (DeepSeek-AI, 2026a), which tests whether the kernel studied in Section 2 produces phase sensitivitywhile being used on the Qwen backbone.

To measure phase sensitivity in these models, we adapt the NIAH task of Section 2.2 to 64 random key–value pairs with adjacent single-token keys and values, and report full-vocabulary next-token accuracy. *Prefix Padding* follows the construction in Section 2.2, shifting all records together with outer fillers. Because the records sit back to back, this shift changes every distractor's phase along with the target's, so a phase effect could in principle come from the distractors rather than the target. We therefore add *In-sequence Padding*, which varies the gaps between records so that distractor phases no longer move in lockstep with the target. Within each protocol, we fix the query position and match the mean and variance of the target-position sampling distributions across residue groups of source-key position modulo 24, a common multiple of the tested strides (Appendix C.3).

3.2 PHASE SENSITIVITY ACROSS THE KERNEL FAMILY

If phase sensitivity arises from chunked compression, compressed models should show it and full-attention baselines should not. We see this contrast in Figure 3. Every compressed model varies periodically with the source-key phase, including the DeepSeek-style kernel (Figure 3i), while the full-attention baselines stay flat (Figure 3a). The same holds for every compressed run in Table 8, with Prefix Padding gaps of up to 78 percentage points against at most 6.1 for the baselines.

Note that summary metrics conceal much of this difference. With eight KV heads, the W8/S8 compressed model nearly matches its full-attention baseline in mean accuracy and validation loss,

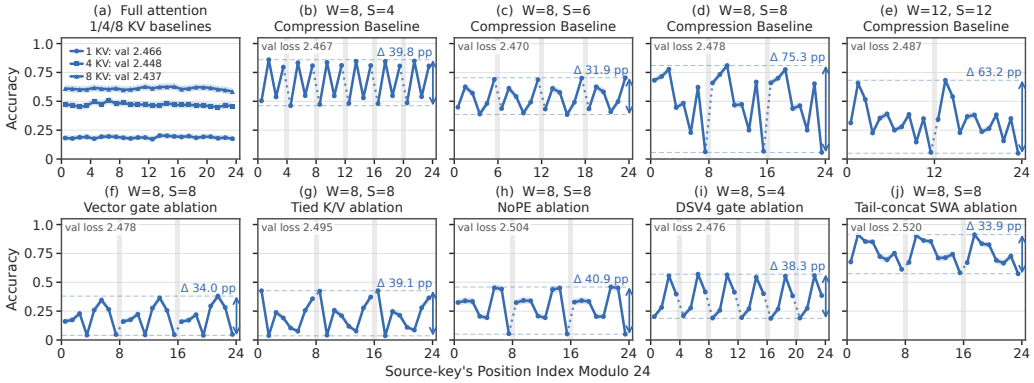

Figure 3: **Phase-sensitive retrieval accuracy by source-key position modulo 24.** (a) shows full-attention baselines with one, four, and eight KV heads per layer. (b)–(e) show W8/S4, W8/S6, W8/S8, and W12/S12 models that otherwise match the one-KV-head reference. Ablations (f) uses vector gates and (g)–(j) vary K/V tying, positional encoding, compression kernel, and local-memory policy. Phase sensitivity persists across the compressed-model panels.

yet its worst residue is only 9.9%, against 58.4% for the baseline. At one and four KV heads, the compressed models reach higher mean accuracy than the baselines at comparable validation loss.

Varying window and stride separately shows that the period follows the stride in the tested geometries, whether or not windows overlap. At $W = 8$, strides $S \in \{4, 6, 8\}$ give periods of about four, six, and eight tokens (Figure 3b–d). W4/S4 shares W8/S4's four-token period despite its smaller window (Figure 10), and W12/S12 produces an approximately twelve-token period (Figure 3e). In these experiments, phase sensitivity needs neither rotary position embeddings (RoPE) nor learned gates: it persists without RoPE (Figure 3h) and when uniform averaging replaces the gates (Figure 11). In Appendix F, we extend these results across head counts, random seeds, and other variants, and to In-sequence Padding, so the pattern does not require all records to shift together.

A natural first explanation is the window boundary, since adjacent keys and values can fall in separate windows. In the W8/S8 reference, only a key at phase seven has its value in the next window. Yet accuracy varies by more than 50 percentage points among phases zero through six, whose pairs share a window (Figure 3d). The boundary does not capture the full accuracy trend. This remaining variation suggests that the answer lies in how the model writes to and reads from compressed memory, so we next ask how retrieval at each phase depends on individual KV heads (Section 4).

4 Phase Specialization

Retrieval at each phase runs through the attention heads that write and read compressed entries, and a head may serve some phases more reliably than others. We call this *phase specialization*: the contributions of some heads to retrieval depend systematically on phase, so different heads contribute asymmetrically to information at different phases. The definition presupposes no cause, whether boundary straddling or within-window differences.

We measure this asymmetry with knockouts in our pretrained models and DeepSeek-V4-Flash-Base (Section 4.1), and then examine a sharper form in one model, where a head's preference for some phases is also visible in how it compresses (Section 4.2).

4.1 Probing Phase Specialization through KV-Head Knockouts

To find which heads matter for retrieval at each phase, we knock out one KV head at a time (Olsson et al., 2022; Wu et al., 2025) and see where accuracy drops. In particular, we use *mean replacement* (Wang et al., 2023), which replaces its output at the intervened positions with its average over many other prompts, so that output no longer carries anything specific to the current prompt, such as which value belongs to the queried key. A drop in accuracy at a phase is evidence that the head's prompt-specific output contributes causally to retrieval at that phase.

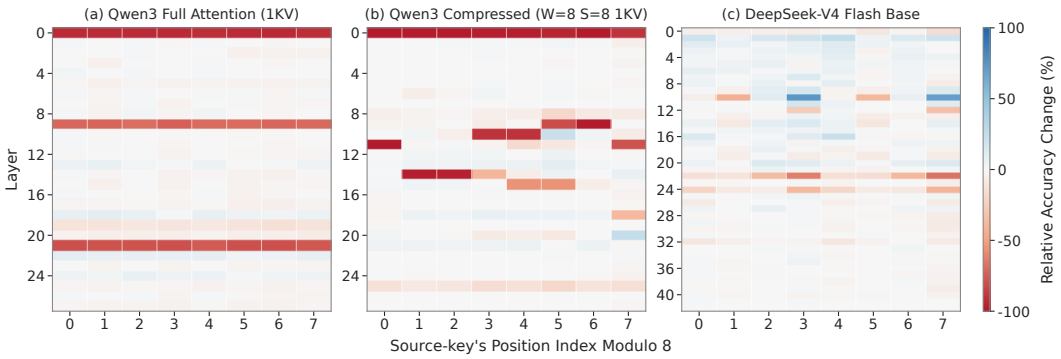

Figure 4: **Retrieval contributions differ across phases.** (a) Qwen3 full attention, (b) Qwen3 with W8/S8 compression, and (c) DeepSeek-V4-Flash-Base, all with one KV head per layer. Each row knocks out one layer's KV head, and columns group source positions modulo eight. Colors show accuracy change divided by clean accuracy at each phase. The model without chunked KV compression (a) shows uniform horizontal stripes across residue groups, indicating that each layer contributes similarly at every source position, while the models with chunked KV compression (b,c) show heterogeneous patterns across the residues of the target key. In (c), the pattern also repeats with period four, matching the CSA stride, and the odd layers, which use heavily compressed attention (HCA) with compression rate 128, contribute little to retrieval.

In our pretrained models, we run these knockouts on the same evaluation sets used in Section 3, replacing the outputs of the query heads that read the KV head at the final position with their means over a disjoint set of calibration prompts. The computation at other token positions stays unchanged. We adopt a similar knockout strategy for DeepSeek-V4-Flash-Base and replace a whole layer's attention output at a band of query suffix, with 256 evaluation prompts per residue group (see Appendix E.6 for details). We visualize the effect of knocking out each head on retrieval performance for information at each phase in Figure 4. The knockout maps differ between full attention and compression. In the full-attention baseline, the layers with large losses reduce accuracy by similar amounts at every position (Figure 4a). In the W8/S8 compressed model, some heads have losses concentrated on a few phases, and different heads have different phase-dependent effects (Figure 4b). In DeepSeek-V4-Flash-Base, the largest losses affect every phase unevenly and a few earlier layers affect only some phases, repeating every four positions, its compression stride (Figure 4c).

This pattern is not specific to one model. In Appendix F, we give knockout maps for every compressed model in Table 8 under both padding protocols. In most compressed models, some heads have losses concentrated on a few phases that repeat every stride, and in the others, losses spread over all phases with periodic variation. With several KV heads, some individual heads still show losses concentrated on one or two phases, so the asymmetry is not only a property of layers (Figure 9).

4.2 A Case Study of Emerging Gate Preferences

Knockouts show which heads matter at which phases, but not why. A natural candidate for the source of cause is the compression gate, which sets each offset's weight in a compressed entry. A head whose gate persistently favors some offsets should matter most at the corresponding phases. We test this in the W8/S8 reference model, whose scalar key and value gates have separate parameters for each offset and whose windows do not overlap ($W = S$), so offset i is phase i.

Because gate scores depend on the input, a gate can concentrate its scores among offsets in two ways. *Static concentration* favors the same offsets in every window, whereas *input-dependent concentration* favors offsets that vary with content. Only static concentration ties a head to fixed phases. We measure both with the *effective number of offsets*, defined as $\exp(H(\cdot))$, where H is Shannon entropy. This metric equals 1 for a one-hot distribution and W for a uniform one. With $\boldsymbol{\alpha} \in \mathbb{R}^W$ the gate scores of a window and expectations over held-out natural-language text, we measure static concentration by $\exp(H(\mathbb{E}[\boldsymbol{\alpha}]))$ and the average per-window concentration by $\mathbb{E}[\exp(H(\boldsymbol{\alpha}))]$. A gate that applies the same scores to every window would be near the diagonal of Figure 5a, since expectation would commute when $\boldsymbol{\alpha}$ is approximately constant.

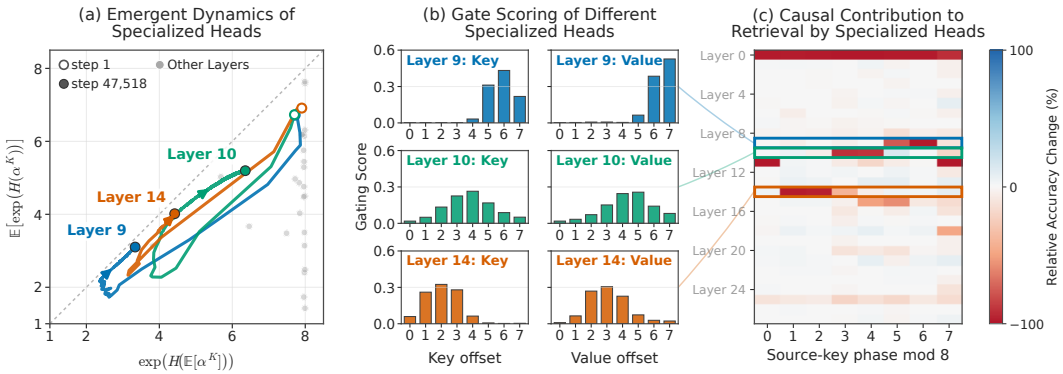

Figure 5: **Emergence and retrieval roles of gate preferences.** (a) Key-gate trajectories during pretraining: some heads drift along the diagonal and become statically concentrated (as highlighted), while others remain unbiased in expectation with $\exp(H(\mathbb{E}[\boldsymbol{\alpha}])) \approx 8$. (b) The highlighted layers 14, 10, and 9 develop static preferences for early, middle, and late offsets in the compression window. (c) For the statically concentrated heads, their peaking phases directly match where they are causally important for retrieval. This is the same figure as Figure 4b for Qwen3 with W8/S8 compression.

As shown in Figure 5, all gates appear to be near uniform around initialization, showing neither kind of concentration (open circles in Figure 5a, close to the upper right diagonal). During pretraining, some gates, such as those in layers 9, 10, and 14, move toward the lower left along the diagonal, forming static concentration. The other key gates end near the right edge, with different levels of input-dependent concentration but no consistent bias (gray points). In Figure 5b we visualize the mean gate scores of these heads. Each key gate peaks at a few adjacent offsets, early in the window for layer 14, in the middle for layer 10, and late for layer 9, and each value gate peaks one offset later, so each head favors a token and its successor, as in a bigram key–value pair.

These peaks match the phases each head specializes in as observed in Section 4.1: its knockout losses concentrate where its key gate favors the queried token and its value gate favors the successor (Figure 5c). Apart from layer 0, the largest knockout losses come from statically concentrated heads. With more heads per layer, static concentration becomes sharper: in the W8/S8 model with four KV heads, many heads peak at a single offset and lose accuracy at a single phase (Figure 17). We defer further results across KV-head counts, random seeds, and compression configurations, including the final gate concentration of every head (Figure 14), to Appendix G.

If learned gate preferences help determine which phases are retrieved well, cycling the offset-specific gate parameters by k offsets should also approximately shift the accuracy pattern by k phases. In Appendix H, we test this prediction in five models with gates exhibiting sharp static concentration (including the W8/S8 reference model). For cycles of one to three offsets in either direction, within-window accuracy patterns largely shift as predicted. These interventions support a causal contribution of learned gate preferences to within-window retrieval asymmetry in these models.

We conjecture that the observed phase sensitivity in accuracy reflects both boundary and within-window asymmetries. The first distinguishes key–value pairs that share a window from pairs that straddle one. The second concerns differences among phases within a window, with static gate concentration as one mechanism that can shape which phases each head retains. Accuracy in the reference model varies by more than 50 percentage points among phases whose key–value pairs all share a window (Section 3.2), while uniform averaging also exhibits within-window variation, showing that this second asymmetry can arise without learned gates.

5 Theory: Emergence of Phase Specialization in Compression

In Section 4.2, some gates develop persistent preferences for the same offsets across windows, and these heads matter most at the corresponding phases. In an idealized induction model, we ask why compression should concentrate when the future query is unknown, why the concentration becomes static, and whether heads divide the phases among themselves.

5.1 Concentration in Optimal Solutions

Consider a context $\boldsymbol{X} = (\boldsymbol{x}_{\ell,r})_{\ell \le L,\, r \le S}$ of LS distinct token embeddings, sampled uniformly without replacement from a vocabulary $\mathcal{V} \subset \mathbb{R}^d$ of size N and arranged in $L \ge 4$ non-overlapping chunks of stride $S \ge 3$. Position r within a chunk is offset $r-1$ and thus a single phase. After compression, a query $\boldsymbol{q}$ repeats a token sampled uniformly from the $L(S-1)$ nonfinal chunk positions and requests its successor $\boldsymbol{y}$ in the same chunk (Olsson et al., 2022; Bietti et al., 2023).

In particular, each of $1 \le H < S$ heads compresses chunk ℓ into

$$\boldsymbol{k}_{h,\ell} = \sum_{r=1}^{S} \boldsymbol{p}^K_{h,\ell}(r)\boldsymbol{x}_{\ell,r} \in \mathbb{R}^d, \qquad \boldsymbol{v}_{h,\ell} = \sum_{r=1}^{S} \boldsymbol{p}^V_{h,\ell}(r)\boldsymbol{x}_{\ell,r} \in \mathbb{R}^d, \tag{3}$$

where the compression coefficients $\boldsymbol{p}^K_{h,\ell}, \boldsymbol{p}^V_{h,\ell}$ are probability vectors in $\mathbb{R}^S$ and are chosen before the query is revealed. For cleaner analysis, we fix the embeddings during training and set the query, key, and value maps to identities.

Each head weights its compressed values by a softmax over the chunk scores $\gamma\langle \boldsymbol{q}, \boldsymbol{k}_{h,\ell}\rangle$, with $\gamma \in (0,1]$, and a tied unembedding of scale $\beta > 0$ maps the summed head outputs to next-token logits. We write $\mathcal{L}$ for the resulting population cross-entropy loss of predicting $\boldsymbol{y}$ (Appendix A).

We assume an idealized embedding geometry with a common component $\kappa > 0$ and error $\varepsilon \ge 0$:

$$\left|\boldsymbol{x}^\top \boldsymbol{x}' - \kappa - \mathbf{1}\{\boldsymbol{x} = \boldsymbol{x}'\}\right| \le \varepsilon \qquad \forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{V}. \tag{4}$$

The common component reflects the anisotropy of token representations (Gao et al., 2019; Razzhigaev et al., 2024; Queipo-de Llano et al., 2026), and exact geometry means $\varepsilon = 0$.

Under exact geometry, a head's target-specific logit is proportional to its attention on the correct chunk times its value weight on the answer. We say that a head *selects* position r when it puts all its key weight on r and all its value weight on $r+1$, the most concentrated compression. Selection strengthens both factors for queries at r but gives up direct evidence for queries elsewhere.

To see what the loss itself favors, we minimize $\mathcal{L}$ over an oracle class $\mathcal{G}_{\text{oracle}}$, whose compression coefficients may depend arbitrarily on the context but not on the query, without imposing a gate parameterization. Writing $\boldsymbol{e}_r$ for the one-hot vector at position r, we obtain the following result.

Theorem 1 (Informal: optimal compression is concentrated)**.** *For sufficiently large N and sufficiently small ε, every global minimizer over $\mathcal{G}_{\text{oracle}}$ sets $\boldsymbol{p}^K_{h,\ell} = \boldsymbol{e}_r$ and $\boldsymbol{p}^V_{h,\ell} = \boldsymbol{e}_{r+1}$ in every context, chunk ℓ, and head h, where $r \in \{1, \dots, S-1\}$ may depend on all three. When $\varepsilon = 0$, the selected positions are distinct across heads within each context and chunk.*

Although the query is unknown, optimal compression is maximally concentrated, with one effective offset per head and per K/V branch in each chunk. The selected position may vary across contexts and chunks, however, so optimality requires concentration in each chunk but not static concentration.

5.2 Static Concentration under Gradient Flow

To study when concentration becomes static, we restrict the oracle class to $\mathcal{G}_{\text{linear}}$, a simplified form of the gates in Equation (1). Each head has a key and a value gate vector $\boldsymbol{c}^B_{h,r} \in \mathbb{R}^d$ for every position r, shared across contexts and chunks, which are the only parameters we train:

$$\boldsymbol{p}^B_{h,\ell}(r) = \frac{\exp\Big(\langle \boldsymbol{c}^B_{h,r}, \boldsymbol{x}_{\ell,r}\rangle\Big)}{\sum_{j=1}^{S} \exp\Big(\langle \boldsymbol{c}^B_{h,j}, \boldsymbol{x}_{\ell,j}\rangle\Big)}, \qquad B \in \{K, V\}. \tag{5}$$

Under exact geometry, the component of $\boldsymbol{c}^B_{h,r}$ along the embedding mean acts as an input-independent bias for position r, so these gates can express static concentration, as in Section 4.2, although the parameterization alone does not force it. We analyze gradient flow on $\mathcal{L}$, the continuous-time limit of gradient descent, from a small random initialization.

Theorem 2 (Informal: gradient flow yields static concentration)**.** *Assume $\varepsilon = 0$ and uniform initialization in a Euclidean ball centered at zero of sufficiently small radius ρ, in the effective gate-parameter space of Assumption 11. For fixed model scales and sufficiently large N relative to $\log(e/\rho)$, with high probability over the initialization, gradient flow on $\mathcal{L}$ yields, for each head h, a position $r_h \in \{1, \dots, S-1\}$ such that $\boldsymbol{p}^K_{h,\ell}(r_h), \boldsymbol{p}^V_{h,\ell}(r_h+1) \to 1$ as training time tends to infinity, uniformly over contexts and chunks. Different heads may select the same r_h.*

For each head, the key gate scores converge to the one-hot vector at r_h and the value gate scores to the one at $r_h + 1$ in every context and chunk, so both the static and the per-window effective numbers of offsets tend to one, the lower-left corner of Figure 5a. Whereas Theorem 1 lets the selected position vary across inputs, gradient flow fixes it for each head. This gives an idealized account of the adjacent key and value preferences of layers 9, 10, and 14 (Figure 5b), whose gate scores remain softer than the one-hot limit. We defer the formal theorem statements and proofs to Appendix A, with relaxed conditions on the number of chunks L and attention scale γ.

6 Related Work

Learned compressed memories reduce the cost of retaining long contexts. Compressive Transformer summarizes past activations (Rae et al., 2020), LoMA trains models to use compressed KV representations (Wang & Xiao, 2024), Activation Beacon replaces raw activations with the KV entries of regularly spaced learned tokens (Zhang et al., 2025), and CAT lets later chunks attend to compressed vectors of earlier chunks (Prakash et al., 2025). Sparse attention (Child et al., 2019; Yuan et al., 2025; Lu et al., 2025; DeepSeek-AI, 2026a) and cache eviction (Zhang et al., 2023; Li et al., 2024) impose related block or token structure on memory.

On the failure mode side, the Lost in the Middle effect shows that retrieval quality depends on where relevant information appears in a long context (Liu et al., 2024). In gist-based compression, Deng et al. (2025) identify degradation near generation-segment boundaries. We instead focus on the phase of the source token, which recurs every compression stride and is distinct from overall context depth or the query's position within a segment. Due to limited space, we defer a more extensive discussion about related works to Appendix I.

7 Conclusion and Discussion

In this paper, we identify phase sensitivity as a systematic failure mode in models with chunk-compressed KV caches: retrieval accuracy depends on the source information's phase relative to the compression windows. The phenomenon appears in DeepSeek-V4, V4.1, and all compressed models we pretrained, with a period following the compression stride. KV-head knockouts in our pretrained models and attention-readout interventions reveal phase-dependent contributions to retrieval, providing causal evidence for phase specialization in these tested models. Finally, our optimization dynamics analysis shows how training can produce static concentration of compression gates in a simplified bigram retrieval model, illustrating a mechanism for phase specialization under the stated assumptions.

Our mechanistic conclusions are strongest in the controlled settings studied here and do not establish a universal cause of phase sensitivity in large, heterogeneous models. In particular, gate cycling does not restore the boundary phase (Appendix H), leaving a unified explanation of boundary and within-window asymmetries unresolved. Whether explicit coordination across heads and layers can make retrieval quality more uniform across phases, without sacrificing average performance or compression efficiency, remains open.

Beyond controlled retrieval, our code-completion examples document periodic errors in DeepSeek-V4 and DeepSeek-V4.1. These observations motivate investigating whether phase sensitivity also affects robustness-relevant behavior. For example, do meaning-preserving shifts of a request within its context alter refusal or compliance, and do any such changes track the compression phase? We leave a systematic study of their potential robustness implications in real application scenarios to future work.

AI USE STATEMENT

We used LLMs to polish the writing throughout the manuscript, to help find related works during the investigation, to assist in the implementation of the research codebase (as a coding agent), and to draft parts of Appendices C to E (starting from our provided outlines).

ETHICS STATEMENT

We analyze open-weight models and models that we pretrain ourselves, using synthetic and public text. There are no human subjects and no personal data. Dual-use risk is minimal.

The main externality of this work is on evaluation practice: averaged long-context benchmarks can hide the failures we describe, and we hope the phase-resolved evaluation we propose helps practitioners detect them.

REPRODUCIBILITY STATEMENT

We give the exact prompt constructions schema for our evaluations in Appendices B and C, the architecture and training settings of our pretrained models in Appendix D, the intervention procedures in Appendix E, and formal statements and proofs of our theoretical results in Appendix A.

REFERENCES

agentlans. High-quality English sentences. Hugging Face dataset, revision `5dfcb23`, 2024. URL `https://huggingface.co/datasets/agentlans/high-quality-english-sentences`. Train split, `text` field.

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

Simran Arora, Sabri Eyuboglu, Aman Timalsina, et al. Zoology: Measuring and improving recall in efficient language models. In *International Conference on Learning Representations*, 2024. URL `https://arxiv.org/abs/2312.04927`.

Alberto Bietti, Vivien Cabannes, Diane Bouchacourt, Hervé Jégou, and Léon Bottou. Birth of a transformer: A memory viewpoint. In *Advances in Neural Information Processing Systems*, 2023. URL `https://arxiv.org/abs/2306.00802`.

Alex Chen, Renato Geh, Aditya Grover, Guy Van Den Broeck, and Daniel Mingyi Israel. The pitfalls of KV cache compression. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 41530–41553. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1926. URL `https://aclanthology.org/2026.acl-long.1926/`.

Siyu Chen, Heejune Sheen, Tianhao Wang, and Zhuoran Yang. Training dynamics of multi-head softmax attention for in-context learning: Emergence, convergence, and optimality (extended abstract). In *Conference on Learning Theory*, volume 247, pp. 4573, 2024. URL `https://proceedings.mlr.press/v247/siyu24a.html`.

Yinpeng Chen, DeLesley Hutchins, Aren Jansen, Andrey Zhmoginov, David Racz, and Jesper Andersen. MELODI: Exploring memory compression for long contexts. In *International Conference on Learning Representations*, 2025. URL `https://arxiv.org/abs/2410.03156`.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019. URL `https://arxiv.org/abs/1904.10509`.

Chenglong Chu, Guorui Zhou, Guowang Zhang, Han Li, Hao Peng, Hongtao Cheng, Jian Liang, Jiangxia Cao, Kun Gai, et al. Kwai summary attention technical report. *arXiv preprint arXiv:2604.24432*, 2026. URL https://arxiv.org/abs/2604.24432.

DeepSeek-AI. DeepSeek-V2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024. URL https://arxiv.org/abs/2405.04434.

DeepSeek-AI. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2025. URL https://arxiv.org/abs/2412.19437.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026a. URL https://arxiv.org/abs/2606.19348.

DeepSeek-AI. DeepSeek-V4.1-Flash: Pushing the limits of KV cache compression. *arXiv preprint arXiv:2609.19969*, 2026b. URL https://arxiv.org/abs/2609.19969.

Chenlong Deng, Zhisong Zhang, Kelong Mao, Shuaiyi Li, Xinting Huang, Dong Yu, and Zhicheng Dou. A silver bullet or a compromise for full attention? A comprehensive study of gist token-based context compression. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4861–4879. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.241. URL https://aclanthology.org/2025.acl-long.241/.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. Representation degeneration problem in training natural language generation models. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=SkEYojRqtm.

Zifan He, Yingqi Cao, Zongyue Qin, Neha Prakriya, Yizhou Sun, and Jason Cong. HMT: Hierarchical memory transformer for efficient long context language processing. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2025. URL https://aclanthology.org/2025.naacl-long.410/.

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 *First Conference on Language Modeling*, 2024. URL https://arxiv.org/abs/2404.06654.

Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. Repeat after me: Transformers are better than state space models at copying. In *International Conference on Machine Learning*, pp. 21502–21521, 2024. URL https://proceedings.mlr.press/v235/jelassi24a.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*, 2024. URL https://arxiv.org/abs/2404.14469.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Enzhe Lu, Zhejun Jiang, Jingyuan Liu, Yulun Du, Tao Jiang, Chao Hong, Shaowei Liu, et al. MoBA: Mixture of block attention for long-context LLMs. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2502.13189.

Amirkeivan Mohtashami and Martin Jaggi. Landmark attention: Random-access infinite context length for transformers. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2305.16300.

Jesse Mu, Xiang Lisa Li, and Noah D. Goodman. Learning to compress prompts with gist tokens. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2304.08467.

Piotr Nawrot, Jianing Li, Renjie Huang, Sebastian Ruder, Kelly Marchisio, and Edoardo M. Ponti. The sparse frontier: Sparse attention trade-offs in transformer LLMs. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 38667–38701. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.1926. URL https://aclanthology.org/2026.findings-acl.1926/.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, et al. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. URL https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

Jatin Prakash, Aahlad Puli, and Rajesh Ranganath. Controllably efficient language models. *arXiv preprint arXiv:2511.05313*, 2025. URL https://arxiv.org/abs/2511.05313.

Enrique Queipo-de Llano, Alvaro Arroyo, Federico Barbero, Xiaowen Dong, Michael Bronstein, Yann LeCun, and Ravid Shwartz-Ziv. Attention sinks and compression valleys in LLMs are two sides of the same coin. In *International Conference on Learning Representations*, pp. 13911–13937, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/1734b19d9afe7d2c7f1154954eaf0d5a-Abstract-Conference.html.

Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SylKikSYDH.

Anton Razzhigaev, Matvey Mikhalchuk, Elizaveta Goncharova, Ivan Oseledets, Denis Dimitrov, and Andrey Kuznetsov. The shape of learning: Anisotropy and intrinsic dimensions in transformer-based models. In *Findings of the Association for Computational Linguistics: EACL 2024*, pp. 868–874. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-eacl.58. URL https://aclanthology.org/2024.findings-eacl.58/.

Margarita Sagitova, Odilon Duranthon, and Lenka Zdeborová. Specialization of softmax attention heads: Insights from the high-dimensional single-location model. *arXiv preprint arXiv:2603.03993*, 2026. URL https://arxiv.org/abs/2603.03993.

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://arxiv.org/abs/2407.15891.

Davoud Ataee Tarzanagh, Yingcong Li, Christos Thrampoulidis, and Samet Oymak. Transformers as support vector machines. *arXiv preprint arXiv:2308.16898*, 2023a. URL https://arxiv.org/abs/2308.16898.

Davoud Ataee Tarzanagh, Yingcong Li, Xuechen Zhang, and Samet Oymak. Max-margin token selection in attention mechanism. In *Advances in Neural Information Processing Systems*, 2023b. URL https://arxiv.org/abs/2306.13596.

Aditya Varre, Mark Rofin, and Nicolas Flammarion. Gradient flow polarizes softmax outputs towards low-entropy solutions. *arXiv preprint arXiv:2603.06248*, 2026. URL https://arxiv.org/abs/2603.06248.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2211.00593.

Yumeng Wang and Zhenyang Xiao. LoMA: Lossless compressed memory attention. *arXiv preprint arXiv:2401.09486*, 2024. URL https://arxiv.org/abs/2401.09486.

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

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. Retrieval head mechanistically explains long-context factuality. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2404.15574.

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://arxiv.org/abs/2410.10819.

Hongru Yang, Zhangyang Wang, Jason D. Lee, and Yingbin Liang. Transformers provably learn two-mixture of linear classification via gradient flow. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=AuAj4vRPkv.

Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, et al. Native sparse attention: Hardware-aligned and natively trainable sparse attention. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23078–23097. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.1126. URL https://aclanthology.org/2025.acl-long.1126/.

Oğuz Kaan Yüksel, Rodrigo Alvarez Lucendo, and Nicolas Flammarion. Incremental learning of sparse attention patterns in transformers. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.19143.

Shuangfei Zhai, Tatiana Likhomanenko, Etai Littwin, Dan Busbridge, Jason Ramapuram, Yizhe Zhang, Jiatao Gu, and Joshua M. Susskind. Stabilizing transformer training by preventing attention entropy collapse. In *International Conference on Machine Learning*, pp. 40770–40803, 2023. URL https://proceedings.mlr.press/v202/zhai23a.html.

Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. Long context compression with activation beacon. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2401.03462.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, et al. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2306.14048.

APPENDIX CONTENTS

Supplementary analyses, experimental protocols, and proofs.

A Deferred Proofs for Theorem 1 and Theorem 2 **15**
- A.1 A Two-Phase Toy Example . . . 15
- A.2 Model and Compressor Classes . . . 20
- A.3 Proof of Theorem 1 . . . 21
- A.4 Proof of Theorem 2 . . . 24

B Additional Details on the Code-Completion Examples **30**
- B.1 FP8 Example for DeepSeek-V4-Flash-Base . . . 31
- B.2 FP8 Example for Other DeepSeek Models . . . 33
- B.3 Additional Code-Completion Examples . . . 35

C Additional Details on the Needle-in-a-Haystack Evaluations **37**
- C.1 NIAH Evaluation for the DeepSeek-V4 Family . . . 37
- C.2 NIAH Evaluation for Qwen3-based Models . . . 39
- C.3 Padding Protocols for Qwen3-based Models . . . 40

D Additional Details on the Architecture and Pretraining of Qwen3-based Models **42**
- D.1 Backbone and Compression Kernels . . . 42
- D.2 Windows, Attention Memory, and Positional Encoding . . . 43
- D.3 Training Settings . . . 44

E Additional Details on KV-Head Knockouts **44**
- E.1 Replaced Attention Output . . . 44
- E.2 Calibration and Mean Replacement . . . 44
- E.3 Evaluation Prompts, Metrics, and Uncertainty . . . 45
- E.4 Knockout Sweeps for Qwen3-based Models . . . 46
- E.5 Scope of the Causal Interpretation . . . 46
- E.6 Knockouts for DeepSeek-V4-Flash-Base . . . 47

F Complete Results for Qwen3-based Models **47**

G Gate Preferences and Knockout Effects across Models **53**
- G.1 Measuring Gate Preferences . . . 53
- G.2 Variation across Random Seeds . . . 53
- G.3 Other Configurations and Overlapping Windows . . . 54

H Redirecting Gate Preferences **60**
- H.1 Gate Cycling . . . 60
- H.2 One-Offset Cycles in a W12/S12 Model . . . 60
- H.3 Replication across Models and Cycle Sizes . . . 62

I Extended Related Work **62**

A Deferred Proofs for Theorem 1 and Theorem 2

Here we state and prove formal versions of Theorem 1 and Theorem 2. Theorem 8 shows that every minimizer over the oracle class uses hard adjacent selection, with each head placing all key weight on one phase and all value weight on the next. We call such a pair of key and value phases a route. The selected route may change across contexts and chunks. Theorem 12 shows that gradient flow on linear compressors from a small random initialization reaches with high probability hard adjacent selection with one route per head for all inputs, although different heads may share a route. We state the formal conditions more generally than in Section 5. The assumptions $L \geq 4$ and $\gamma \leq 1$ made there imply $\gamma < \log 3 \leq \log(L-1)$, so the formal conditions hold in the main-text setting.

Both results turn on a tradeoff between averaging and selecting. A head that averages the positions of a chunk keeps some evidence about every position, whereas a head that selects one position keeps that position's evidence at full strength and gives up direct evidence about the others. Selection is favored when the gain at the selected position outweighs the loss elsewhere. A second question is whether several heads divide the positions among themselves, which we call complementary coverage, since sharp heads can serve as many positions as possible only if their selections differ.

In the general model, this tradeoff is hard to see directly. Each head has key and value coefficients over S positions in every chunk and attends to chunks through its own softmax, all heads are coupled through the softmax over the vocabulary, and under gradient flow the coefficients come from gates that depend on the input embeddings. We would therefore like to see the tradeoff in a setting small enough to solve by hand before proving the general results. In Appendix A.1, we study a toy example with two heads and two phases, in which each head's compression is set by a single number and the loss is a function of two variables. There we can compute in closed form why a head sharpens its compression and why the two heads prefer to split the phases between them. The toy example is not a special case of the general theorems, since it has $S = 2$, keeps only keys, and idealizes the value readout. We use it for intuition, and at its end we explain which parts of its argument carry over to the general model and which do not (Appendix A.1.2).

We then return to the general model. In Appendix A.2, we define the model and the two compressor classes. In Appendix A.3, we prove Theorem 8, in which sharpening reappears for paired key and value coefficients. In Appendix A.4, we prove Theorem 12, using the toy example's strategy of starting near a symmetric point and asking along which directions gradient flow leaves it.

A.1 A Two-Phase Toy Example

In the toy example, each chunk holds two key entries, each head compresses them with a single trainable logit, and an idealized readout replaces the values. Unlike Theorem 2, which does not ensure that different heads select different routes, gradient flow here assigns the two heads different phases, provided the output scale β is large enough (condition (16) below).

Each of L chunks contains two fixed key entries $\boldsymbol{k}_{\ell,1}, \boldsymbol{k}_{\ell,2}$. Each head $h \in \{1, 2\}$ uses a trainable logit θ_h to form one compressed key:

$$p_h = \sigma(\theta_h), \qquad \bar{\boldsymbol{k}}_{h,\ell} = p_h \boldsymbol{k}_{\ell,1} + (1 - p_h)\boldsymbol{k}_{\ell,2}.$$

Here $\sigma(t) = (1 + e^{-t})^{-1}$, $p_h = 1$ selects phase 1, $p_h = 0$ selects phase 2, and $p_h = 1/2$ mixes them equally. A subsequent query requests either phase with equal probability. Under orthogonal addressing, the analogue of exact geometry with $\kappa = 0$, a query has inner product one with its requested key in the unique target chunk and zero with every other key. With attention scale one, the target-chunk logit of head h is therefore p_h for phase 1 and $1 - p_h$ for phase 2, while all $L - 1$ distractor logits are zero. The total correct-chunk attention masses of the two heads for phases 1 and 2 are

$$\bar{R}_1 = \sum_{h=1}^{2} \frac{e^{p_h}}{e^{p_h} + L - 1}, \qquad \bar{R}_2 = \sum_{h=1}^{2} \frac{e^{1-p_h}}{e^{1-p_h} + L - 1}.$$

The value readout is idealized: each head contributes one unit of correct-class evidence per unit of attention mass on the target chunk, and the contributions of the two heads add. With a fixed output scale $\beta > 0$ and no additional constant offset, the correct-class margins for phases 1 and 2 are $\beta\bar{R}_1$ and $\beta\bar{R}_2$. Training minimizes the population cross-entropy, which is the logistic loss of this margin

averaged over the two phases:

$$\mathcal{L}(p_1, p_2) = \left[\log(1 + e^{-\beta \bar{R}_1}) + \log(1 + e^{-\beta \bar{R}_2})\right]/2. \tag{6}$$

Set $D = (L-1)e^{-1/2}$, so that $D > 1$ for $L \geq 3$.

Informally, moving any mixed $p_h \in (0,1)$ to either endpoint increases $\bar{R}_1 + \bar{R}_2$, and at a fixed sum the convexity of the logistic loss in attention mass favors $\bar{R}_1 = \bar{R}_2$. Together these favor the two complementary endpoint optima established below.

A.1.1 Complementary Two-Phase Selection

Training updates the logits θ_h rather than p_h, so we rewrite the objective in terms of them. Let $\boldsymbol{\theta} = (\theta_1, \theta_2)$, and write head h's signed phase preference as

$$\chi(\theta_h) = 2p_h - 1 = \tanh(\theta_h/2). \tag{7}$$

For $s = +1$ (phase 1) or $s = -1$ (phase 2), the target-chunk logit of head h is $(1 + s\chi(\theta_h))/2$, so its correct-chunk attention mass is

$$a(s\chi(\theta_h)), \qquad a(x) = \frac{1}{1 + De^{-x/2}}. \tag{8}$$

In these coordinates, the two total attention masses and the logistic loss are

$$\bar{R}_1 = a(\chi(\theta_1)) + a(\chi(\theta_2)), \qquad \bar{R}_2 = a(-\chi(\theta_1)) + a(-\chi(\theta_2)), \tag{9}$$

$$\varphi(m) = \log(1 + e^{-\beta m}), \qquad \mathcal{L}(p_1, p_2) = \frac{1}{2}\left[\varphi(\bar{R}_1) + \varphi(\bar{R}_2)\right], \tag{10}$$

which is Equation (6) rewritten through $\chi(\theta_h) = 2p_h - 1$. The corresponding gradient flow is $\dot{\boldsymbol{\theta}} = -\nabla_{\boldsymbol{\theta}} \mathcal{L}(\sigma(\theta_1), \sigma(\theta_2))$, and $\varphi' < 0 < \varphi''$.

To analyze this flow, split retrieval into its even and odd parts:

$$\begin{aligned} a_+(x) &= \frac{a(x) + a(-x)}{2}, \qquad a_-(x) = \frac{a(x) - a(-x)}{2}, \\ \bar{a}_+(\theta) &= a_+(\chi(\theta)), \qquad \bar{a}_-(\theta) = a_-(\chi(\theta)). \end{aligned} \tag{11}$$

Then $\bar{R}_1 = M + I$ and $\bar{R}_2 = M - I$, where

$$M = \bar{a}_+(\theta_1) + \bar{a}_+(\theta_2), \qquad I = \bar{a}_-(\theta_1) + \bar{a}_-(\theta_2), \tag{12}$$

and hence

$$\mathcal{L}(p_1, p_2) = \frac{1}{2}\left[\varphi(M + I) + \varphi(M - I)\right]. \tag{13}$$

This decomposition separates sharpening from balancing. For $x > 0$, set $t = e^{x/2} > 1$. Then

$$2a_+(x) = \frac{t}{t + D} + \frac{1}{1 + Dt}, \qquad 2\frac{\mathrm{d}}{\mathrm{d}t} a_+(2\log t) = D\left[\frac{1}{(t + D)^2} - \frac{1}{(1 + Dt)^2}\right] > 0 \tag{14}$$

for $D > 1$ and $x > 0$. Thus increasing $|\theta_h|$ increases its contribution to the mean attention mass. At fixed M, strict convexity gives $\mathcal{L}(p_1, p_2) \geq \varphi(M)$, with equality only at $I = 0$. Hence retrieval competition favors sharp heads, whereas loss curvature favors balanced coverage.

Both pressures are satisfied together only when the two heads sit at opposite endpoints, and the following theorem makes this precise and shows that gradient flow from near the uniform point reaches these endpoints when β is large enough. Let $G(x) = a(x) + a(-x) = 2a_+(x)$, which is strictly increasing on $[0, 1]$ by the calculation above.

Theorem 3 (Complementary two-phase selection)**.** *Suppose $D > 1$. On $[0,1]^2$, the only global minimizers of $\mathcal{L}(p_1, p_2)$ are $(1, 0)$ and $(0, 1)$, corresponding in the compactified $\boldsymbol{\theta}$-space to*

$$(+\infty, -\infty) \qquad \textit{and} \qquad (-\infty, +\infty). \tag{15}$$

If also

$$2\beta\sigma\left(\frac{2\beta}{1 + D}\right) > D - \frac{1}{D}, \tag{16}$$

then any absolutely continuous initialization supported in a sufficiently small neighborhood of the origin converges almost surely under gradient flow to one of these minimizers. If its density is bounded, then for $\delta, \eta \in (0, 1/2)$, outside an event of probability at most δ, the time until $(p_1(t), p_2(t))$ lies within ℓ_∞-distance η of $(1, 0)$ or $(0, 1)$ is

$$T_{\eta,\delta} = O\left(\log \frac{1}{\delta} + \frac{1}{\eta}\right), \tag{17}$$

where the hidden constant depends on L, β, and the initialization law through its fixed neighborhood and density bound.

We establish the loss landscape first, then show that nearby trajectories escape toward and remain near a complementary route.

Lemma 4 (Global minimizers)**.** *If $D > 1$, the only minimizers of $\mathcal{L}$ on $[0, 1]^2$ are $(1, 0)$ and $(0, 1)$.*

Proof. The mean attention mass is

$$\frac{\bar{R}_1 + \bar{R}_2}{2} = \frac{G(|\chi(\theta_1)|) + G(|\chi(\theta_2)|)}{2} \le G(1). \tag{18}$$

Jensen's inequality and monotonicity of φ yield

$$\mathcal{L}(p_1, p_2) \ge \varphi\left(\frac{\bar{R}_1 + \bar{R}_2}{2}\right) \ge \varphi(G(1)). \tag{19}$$

Equality in the second inequality requires $|\chi(\theta_1)| = |\chi(\theta_2)| = 1$, and strict convexity makes the first tight only when $\bar{R}_1 = \bar{R}_2$, which forces opposite signs. □

The remaining lemmas analyze the flow near the complementary line $(\theta_1, \theta_2) = (\xi, -\xi)$. Use the orthonormal coordinates

$$u = \frac{\theta_1 - \theta_2}{\sqrt{2}}, \qquad v = \frac{\theta_1 + \theta_2}{\sqrt{2}}. \tag{20}$$

For $\xi \ge 0$, put

$$\bar{M}_\xi = 2\bar{a}_+(\xi) = G(\chi(\xi)) \tag{21}$$

and let

$$\lambda_\perp(\xi) = \varphi'(\bar{M}_\xi)\bar{a}''_+(\xi) + 2\varphi''(\bar{M}_\xi)\big(\bar{a}'_-(\xi)\big)^2. \tag{22}$$

This is the Hessian eigenvalue in the common-head direction $(1, 1)/\sqrt{2}$ at $(\xi, -\xi)$. At the uniform point, $\bar{M}_0 = 2/(1 + D)$, and the escape rate along $(1, -1)/\sqrt{2}$ is

$$\lambda_{\text{esc}} = -\varphi'(\bar{M}_0)\frac{D(D-1)}{16(1+D)^3} > 0. \tag{23}$$

Lemma 5 (Stability of complementary routing)**.** *Suppose $D > 1$ and Equation (16) holds. The complementary line is invariant, $\lambda_\perp(\xi) > 0$ for every $\xi \ge 0$, and*

$$\inf_{\xi \ge 0} e^{\xi}\lambda_\perp(\xi) > 0. \tag{24}$$

Proof. Invariance follows by exchanging the two heads and phases. Let $x = \chi(\xi)$. Where $\bar{a}''_+(\xi) > 0$, the curvature satisfies

$$\frac{\bar{a}''_+(\xi)}{2(\bar{a}'_-(\xi))^2} \le \frac{1}{2}\left(D - \frac{1}{D}\right). \tag{25}$$

Indeed, $a'_+(x) > 0$ for $x > 0$, and

$$\frac{\bar{a}''_+(\xi)}{2(\bar{a}'_-(\xi))^2} = \frac{a''_+(x) - 2xa'_+(x)/(1-x^2)}{2(a'_-(x))^2} \le \frac{a''_+(x)}{2(a'_-(x))^2}. \tag{26}$$

To verify the last bound, put $t = e^{x/2} \ge 1$ and $Q = (1 + Dt)^2 + (D + t)^2$. Substituting $a(x) = (1 + De^{-x/2})^{-1}$ and clearing positive denominators gives

$$(D^2 - 1)tQ^2 - 2(D + t)(1 + Dt) \times \left[(D - t)(1 + Dt)^3 + (Dt - 1)(D + t)^3\right] = (t - 1)^2(D - 1)P_D(t),$$

where

$$P_D(t) = (t + t^3)(1 + D^5) + (t + 8t^2 + t^3)(D + D^4) + (2 + 6t + 4t^2 + 6t^3 + 2t^4)(D^2 + D^3).$$

Every coefficient is positive, proving Equation (25).

Furthermore,

$$\frac{\varphi''(\bar{M}_\xi)}{-\varphi'(\bar{M}_\xi)} = \beta\sigma(\beta\bar{M}_\xi) \geq \beta\sigma(\beta\bar{M}_0) = \frac{\varphi''(\bar{M}_0)}{-\varphi'(\bar{M}_0)}, \tag{27}$$

because $\bar{M}_\xi = G(\chi(\xi))$ is increasing. Combining this inequality, Equation (16), and Equation (25) proves $\lambda_\perp(\xi) > 0$ whenever $\bar{a}''_+(\xi) > 0$. If $\bar{a}''_+(\xi) \leq 0$, positivity is immediate from $\varphi' < 0 < \varphi''$.

Finally, $e^\xi \lambda_\perp(\xi)$ is continuous and positive for finite ξ, and direct differentiation gives

$$e^\xi \lambda_\perp(\xi) \longrightarrow 2[-\varphi'(G(1))]a'_+(1) > 0. \tag{28}$$

Compactness after adjoining $\xi = \infty$ proves Equation (24). □

We next use this curvature bound to trap trajectories in a tube around the complementary line.

Lemma 6 (Attracting complementary tube)**.** *Suppose $D > 1$ and Equation (16) holds. For every $\xi_* > 0$, there are $\zeta_T, c_T, C_T, c_N > 0$ such that, in the coordinates*

$$(\theta_1, \theta_2) = \left(\xi + \frac{v}{\sqrt{2}}, -\xi + \frac{v}{\sqrt{2}}\right), \qquad \xi \geq \xi_*, \quad |v| \leq \zeta_T,$$

the gradient flow satisfies

$$c_T e^{-\xi} \leq \dot{\xi} \leq C_T e^{-\xi}, \qquad v\dot{v} \leq -c_N e^{-\xi} v^2. \tag{29}$$

This tube is forward invariant, and its trajectories have $\xi(t) \to \infty$ and $v(t) \to 0$.

Proof. The tangential speed at $v = 0$ is

$$\Omega(\xi) := -\frac{1}{4}\varphi'(G(\chi(\xi)))G'(\chi(\xi))\big(1 - \chi(\xi)^2\big) > 0. \tag{30}$$

On the line $v = 0$, $\dot{\xi} = \Omega(\xi) > 0$, and the transverse linearization is $\dot{v} = -\lambda_\perp(\xi)v$. Since $\chi(\xi) = \tanh(\xi/2)$,

$$e^\xi \Omega(\xi) \longrightarrow -\varphi'(G(1))G'(1) > 0, \tag{31}$$

while Equation (24) bounds $e^\xi \lambda_\perp(\xi)$ away from zero. Every derivative with respect to a saturated logit carries a factor $\chi'(\xi) = O(e^{-\xi})$. Taylor expansion in v, uniform after weighting the vector field by e^ξ, gives

$$\dot{\xi} = \Omega(\xi) + O(e^{-\xi}|v|), \tag{32}$$

$$\dot{v} = -\lambda_\perp(\xi)v + O(e^{-\xi}v^2). \tag{33}$$

The weighted coefficients extend continuously to $\xi = \infty$. Their positive lower bounds and finite upper bounds on $[\xi_*, \infty]$ yield Equation (29) after choosing ζ_T sufficiently small. The sign of $v\dot{v}$ makes the tube forward invariant, and the tangential bounds give $\xi(t) \to \infty$. Moreover, whenever $v \neq 0$, $\mathrm{d}\log|v|/\mathrm{d}\xi \leq -c_N/C_T$, so $v(t) \to 0$. □

It remains to show that trajectories starting near the uniform point enter this tube.

Lemma 7 (Exit into the complementary tube)**.** *Under the assumptions of Lemma 6, there are $u_*, \rho_0 > 0$ such that every trajectory with $\|(u(0), v(0))\|_2 \leq \rho_0$ and $u(0) \neq 0$ reaches $|u| = u_*$ within time*

$$\tau \leq \frac{2}{\lambda_{\mathrm{esc}}} \log \frac{u_*}{|u(0)|}. \tag{34}$$

At exit, the trajectory lies in the tube from Lemma 6 with $\xi_* = u_*/\sqrt{2}$, after exchanging heads if $u(0) < 0$.

Proof. The symmetry lines $u = 0$ and $v = 0$ are invariant. Smoothness and the two Hessian signs at the origin give

$$\dot{u} = u\left(\lambda_{\text{esc}} + O(\|(u, v)\|_2)\right), \qquad \dot{v} = v\left(-\lambda_{\perp}(0) + O(\|(u, v)\|_2)\right). \tag{35}$$

Choose $u_* > 0$ small enough that the coefficients in Equation (35) stay within half their origin magnitudes whenever $|u|, |v| \le u_*$. Apply Lemma 6 at $\xi_* = u_*/\sqrt{2}$ and reduce its tube width ζ_T to at most u_*. Then whenever $|u| \le u_*$ and $|v| \le \zeta_T$,

$$\frac{\mathrm{d}}{\mathrm{d}t}\log|u| \ge \frac{\lambda_{\text{esc}}}{2}, \qquad \frac{\mathrm{d}}{\mathrm{d}t}|v| \le -\frac{\lambda_{\perp}(0)}{2}|v|.$$

The second inequality holds for $v \neq 0$, and the line $v = 0$ is invariant. Choose $\rho_0 < \min\{u_*, \zeta_T\}$. Then $|v|$ cannot reach ζ_T before $|u|$ reaches u_*. Integrating the first inequality gives Equation (34). At exit $\xi = |u(\tau)|/\sqrt{2} = \xi_*$ and $|v(\tau)| \le \rho_0 < \zeta_T$, so the point lies in the tube. □

Proof of Theorem 3. Lemma 4 identifies the only global minima, while Lemmas 5 to 7 give convergence for every sufficiently small initialization off $u = 0$, which has probability zero under an absolutely continuous law.

For the rate, enclose the fixed initialization support in a ball of radius $\rho \le \rho_0$ and let its density be bounded by C. Then $\Pr(|u(0)| \le t) \le 4C\rho t$. Outside an event of probability at most δ, we have $|u(0)| > \delta/(4C\rho)$, so Equation (34) gives tube entry in $O(\log(1/\delta))$ time. The normal coordinate then remains within ζ_T. To reach ℓ_∞-error at most η, it suffices that $\xi \ge \log((1 - \eta)/\eta) + \zeta_T/\sqrt{2}$. Integrating the tube bound $\dot{\xi} \ge c_T e^{-\xi}$ up to this threshold (if it exceeds ξ_*) takes at most $e^{\zeta_T/\sqrt{2}}/(c_T\eta)$ additional time. This proves Equation (17) for $\delta, \eta \in (0, 1/2)$, and the same tube bounds ensure convergence to the corresponding minimizer. □

A.1.2 From the Toy Example to the General Model

The toy example isolates two pressures, one favoring sharp heads and one favoring balanced coverage of the phases. Sharpening comes from competition with the $L - 1$ distractor chunks. Because $D > 1$, a uniform head gives the correct chunk less than half of its attention, and in this regime the correct-chunk attention mass rises more when the target logit increases than it falls when the logit decreases by the same amount. A head that moves toward one phase therefore gains more attention mass on that phase than it loses on the other, which raises the mean attention mass over the two phases (Equation (14)). Balance comes from the loss. Both heads add evidence to the same margin for each phase, and the logistic loss flattens as a margin grows, so evidence added to the phase with the larger margin lowers the loss less than evidence added to the phase with the smaller one. For a fixed mean, the loss is therefore smallest when both phases receive the same attention mass. Both pressures are satisfied together only at the complementary endpoints, which is why these are the only global minimizers (Lemma 4).

The same two pressures govern the dynamics. At the uniform point, the two heads are identical. Along the complementary direction $(1, -1)/\sqrt{2}$, in which the heads move toward different phases, the two phases stay balanced, so only sharpening acts and trajectories escape at rate λ_{esc}. Along the common direction $(1, 1)/\sqrt{2}$, in which both heads move toward the same phase, sharpening and balance compete. Condition (16) is sufficient for balance to win, which makes this direction stable and the uniform point a saddle (Lemma 5). A small random initialization therefore leaves the uniform point along the complementary direction and then stays near the complementary line.

Both pressures reappear in the general model, but they change in different ways. Sharpening survives, but it now comes from pairing keys with values rather than from the asymmetry that $D > 1$ provides. Under exact geometry, a head's contribution to the target logit for a query at position r is its correct-chunk attention, which grows with its key weight at r, times its value weight at $r + 1$. Because the value weights sum to one, this contribution summed over query positions is at most the attention that a one-hot key receives, and it reaches that value only when the key coefficient is one-hot at some r and the value coefficient is one-hot at $r + 1$. In Appendix A.3, we use this bound to rule out mixed coefficients, in the role that Lemma 4 plays for the toy example.

Balance, by contrast, becomes weak when the vocabulary is large. In that regime, the cross-entropy loss is nearly linear in the target logit, because its normalizer is dominated by the many vocabulary

items that do not appear in the context. Under exact geometry, two heads that select the same position keep the same expected target logit and pay only through the normalizer, by an amount at most of order $1/N$. This penalty still makes complementary coverage optimal when $\varepsilon = 0$ (Appendix A.3). Our dynamics proof, however, treats all normalizer terms as small perturbations, and the remaining target terms do not couple different heads, so the proof has no force that separates them (Appendix A.4). Unlike the toy example, Theorem 12 therefore guarantees persistent selection without guaranteeing complementary coverage.

A.2 Model and Compressor Classes

We now state the general model of Section 5 with full notation. We index the key and value branches by $P \in \{K, V\}$, which Section 5 writes as B.

For every positive integer m, write $[m] := \{1, \dots, m\}$. Let $S \geq 3$, use $1 \leq H \leq S-1$ heads, and use $L \geq 2$ chunks. The chunks are non-overlapping and each contains S positions, so the compression stride and effective compression ratio are both S. Let $\mathcal{V} \subset \mathbb{R}^d$ be a vocabulary of N distinct embedding vectors, where $N \geq LS$. A context $\boldsymbol{X}$ consists of the L ordered chunks

$$\boldsymbol{X} = \big((\boldsymbol{x}_{\ell,1}, \dots, \boldsymbol{x}_{\ell,S})\big)_{\ell=1}^{L}, \tag{36}$$

where the LS embeddings are sampled uniformly without replacement from $\mathcal{V}$. Independently, $\ell_* \sim \mathrm{Unif}[L]$ and $r_* \sim \mathrm{Unif}[S-1]$. The query vector is $\boldsymbol{q} = \boldsymbol{x}_{\ell_*, r_*}$ and the target embedding is $\boldsymbol{y} = \boldsymbol{x}_{\ell_*, r_*+1}$. Let $\mathcal{X}$ be the finite set of all such ordered contexts with distinct embeddings.

For the first global theorem, we assume that for some $\kappa \in \mathbb{R}$ and $0 \leq \varepsilon < 1$,

$$\big|\boldsymbol{x}^\top \boldsymbol{x}' - \kappa - \mathbf{1}\{\boldsymbol{x} = \boldsymbol{x}'\}\big| \leq \varepsilon \qquad (\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{V}). \tag{37}$$

This is the embedding geometry of Equation (4) in Section 5. For fixed $\varepsilon > 0$, it permits dimension logarithmic in N. The dynamics result uses its idealized high-dimensional limit $\varepsilon = 0$.

Let $\boldsymbol{E} \in \mathbb{R}^{N \times d}$ have row $\boldsymbol{x}^\top$ indexed by $\boldsymbol{x} \in \mathcal{V}$, and fix $0 < \gamma, \beta < \infty$.

We consider two compressor classes. In the oracle class, after observing $\boldsymbol{X}$ and before (ℓ_*, r_*) is sampled, a compressor rule may compute arbitrary vectors

$$\boldsymbol{p}_{h,\ell}^K(\boldsymbol{X}), \boldsymbol{p}_{h,\ell}^V(\boldsymbol{X}) \in \Delta_S, \qquad \Delta_S := \left\{ \boldsymbol{p} \in \mathbb{R}^S : \boldsymbol{p}(r) \geq 0,\ \sum_{r=1}^{S} \boldsymbol{p}(r) = 1 \right\} \tag{38}$$

for every $\ell \in [L]$ and $h \in [H]$. These key and value compression coefficients may depend arbitrarily on $\boldsymbol{X}$. The class of all such pre-query compressor rules is

$$\mathcal{G}_{\mathrm{oracle}} = \left\{ g : \mathcal{X} \longrightarrow (\Delta_S \times \Delta_S)^{LH} \right\}, \qquad g(\boldsymbol{X}) = \big(\boldsymbol{p}_{h,\ell}^K(\boldsymbol{X}), \boldsymbol{p}_{h,\ell}^V(\boldsymbol{X})\big)_{\ell,h}. \tag{39}$$

Thus the information order is $\boldsymbol{X} \mapsto g(\boldsymbol{X})$, followed by $(\ell_*, r_*) \sim \mathrm{Unif}([L] \times [S-1])$. Here and below, $\boldsymbol{e}_r$ is the rth vertex of Δ_S. When a context and query are fixed, we suppress their dependence in the notation.

Position r is window offset $r-1$ in Section 4.2 and corresponds to one phase. Both coefficient vectors use physical phase indices. No phase is masked, and the useful adjacent route $r \in [S-1]$ pairs key phase r with value phase $r+1$. The compressed KV entries are

$$\boldsymbol{k}_{h,\ell} = \sum_{r=1}^{S} \boldsymbol{p}_{h,\ell}^K(r) \boldsymbol{x}_{\ell,r}, \qquad \boldsymbol{v}_{h,\ell} = \sum_{r=1}^{S} \boldsymbol{p}_{h,\ell}^V(r) \boldsymbol{x}_{\ell,r}. \tag{40}$$

For query $\boldsymbol{q}$, the attention weights are

$$\alpha_{h,\ell} = \frac{\exp\{\gamma \langle \boldsymbol{q}, \boldsymbol{k}_{h,\ell} \rangle\}}{\sum_{j=1}^{L} \exp\{\gamma \langle \boldsymbol{q}, \boldsymbol{k}_{h,j} \rangle\}}. \tag{41}$$

The attention output and logits are

$$\mathsf{TF}(\boldsymbol{X}, \boldsymbol{q}) = \sum_{h=1}^{H} \sum_{\ell=1}^{L} \alpha_{h,\ell} \boldsymbol{v}_{h,\ell}, \qquad \boldsymbol{O}(\boldsymbol{X}, \boldsymbol{q}) = \beta \boldsymbol{E}\, \mathsf{TF}(\boldsymbol{X}, \boldsymbol{q}). \tag{42}$$

For a fixed context $\boldsymbol{X}$, let $\boldsymbol{p}$ denote the full collection of oracle coefficients and substitute it into this architecture. Let $\boldsymbol{O}(\boldsymbol{X}, \boldsymbol{q})_{\boldsymbol{x}}$ denote the output coordinate indexed by $\boldsymbol{x} \in \mathcal{V}$. The fixed-context objective is

$$\mathcal{L}_{\boldsymbol{X}}(\boldsymbol{p}) = \frac{1}{L(S-1)} \sum_{\ell_*=1}^{L} \sum_{r_*=1}^{S-1} \left[\log \sum_{\boldsymbol{x} \in \mathcal{V}} e^{\boldsymbol{O}(\boldsymbol{X}, \boldsymbol{q})_{\boldsymbol{x}}} - \boldsymbol{O}(\boldsymbol{X}, \boldsymbol{q})_{\boldsymbol{y}} \right]. \tag{43}$$

Its population average is

$$\mathcal{L}(g) = \frac{1}{|\mathcal{X}|} \sum_{\boldsymbol{X} \in \mathcal{X}} \mathcal{L}_{\boldsymbol{X}}(g(\boldsymbol{X})). \tag{44}$$

The finite product $\mathcal{G}_{\text{oracle}}$ is compact, and $\mathcal{L}$ is continuous on it.

For the training-dynamics result, we restrict the oracle class to linear compressors, whose gate logits are linear in the embeddings. They use vectors $\boldsymbol{c}^K_{h,r}, \boldsymbol{c}^V_{h,r} \in \mathbb{R}^d$ shared across contexts and chunks, with no scalar biases. Write $\boldsymbol{c} = (\boldsymbol{c}^P_{h,r})_{P,h,r}$. For $P \in \{K, V\}$, its compression coefficients are

$$\boldsymbol{p}^P_{h,\ell}(r; \boldsymbol{c}, \boldsymbol{X}) = \frac{\exp\{\langle \boldsymbol{c}^P_{h,r}, \boldsymbol{x}_{\ell,r} \rangle\}}{\sum_{j=1}^{S} \exp\{\langle \boldsymbol{c}^P_{h,j}, \boldsymbol{x}_{\ell,j} \rangle\}} \tag{45}$$

and

$$\boldsymbol{p}^P_{h,\ell}(\boldsymbol{c}; \boldsymbol{X}) := (\boldsymbol{p}^P_{h,\ell}(r; \boldsymbol{c}, \boldsymbol{X}))_{r=1}^{S}, \qquad g_{\boldsymbol{c}}(\boldsymbol{X}) = \big(\boldsymbol{p}^K_{h,\ell}(\boldsymbol{c}; \boldsymbol{X}), \boldsymbol{p}^V_{h,\ell}(\boldsymbol{c}; \boldsymbol{X})\big)_{\ell,h}. \tag{46}$$

The resulting linear compressor class is

$$\mathcal{G}_{\text{linear}} = \big\{ g_{\boldsymbol{c}} : \boldsymbol{c}^K_{h,r}, \boldsymbol{c}^V_{h,r} \in \mathbb{R}^d \text{ for all } h, r \big\} \subset \mathcal{G}_{\text{oracle}}, \tag{47}$$

and only the vectors $\boldsymbol{c}^K_{h,r}, \boldsymbol{c}^V_{h,r}$ are trainable. The inclusion can be strict because an oracle rule may use the whole context in an arbitrary way, whereas $g_{\boldsymbol{c}}$ uses shared phasewise linear gate logits.

A.3 Proof of Theorem 1

We first ask what the loss itself favors when compression may depend on the context in any way. As in the toy example, we compare mixed coefficients with one-hot ones, now with keys and values paired.

Symmetric comparison loss. We compare the true loss with an auxiliary functional of the same oracle coefficients that isolates the symmetric part $\kappa + \mathbf{1}\{\boldsymbol{x} = \boldsymbol{x}'\}$ of the Gram matrix.

For $0 \le u \le 1$, define

$$R(u) = \frac{e^{\gamma u}}{e^{\gamma u} + L - 1}. \tag{48}$$

For a fixed context, query (ℓ_*, r_*), and coefficient collection $\boldsymbol{p}$, set

$$A^h_\ell = \begin{cases} R(\boldsymbol{p}^K_{h,\ell_*}(r_*)), & \ell = \ell_*, \\ \{1 - R(\boldsymbol{p}^K_{h,\ell_*}(r_*))\}/(L-1), & \ell \neq \ell_*. \end{cases} \tag{49}$$

These are the comparison attention weights. Using the same value compression coefficients as the true model, define the unscaled comparison output coordinate at $\boldsymbol{x} \in \mathcal{V}$ by

$$F_{\boldsymbol{x}} = \sum_{h=1}^{H} \sum_{\ell=1}^{L} A^h_\ell \sum_{r=1}^{S} \boldsymbol{p}^V_{h,\ell}(r)\, \mathbf{1}\{\boldsymbol{x}_{\ell,r} = \boldsymbol{x}\}. \tag{50}$$

Since both A^h and $\boldsymbol{p}^V_{h,\ell}$ have unit mass, $\sum_{\boldsymbol{x} \in \mathcal{V}} F_{\boldsymbol{x}} = H$. Vocabulary items absent from the context have $F_{\boldsymbol{x}} = 0$. Hence the comparison query loss is $\log \sum_{\boldsymbol{x} \in \mathcal{V}} e^{\beta F_{\boldsymbol{x}}} - \beta F_{\boldsymbol{y}}$. Let $\mathcal{L}_{\text{sym},\boldsymbol{X}}(\boldsymbol{p})$ be its average over the $L(S-1)$ queries, and define its population average by

$$\mathcal{L}_{\text{sym}}(g) = \frac{1}{|\mathcal{X}|} \sum_{\boldsymbol{X} \in \mathcal{X}} \mathcal{L}_{\text{sym},\boldsymbol{X}}(g(\boldsymbol{X})). \tag{51}$$

When $\varepsilon = 0$, the comparison functional equals the true loss. Indeed, $\boldsymbol{x}^\top \boldsymbol{x}' = \kappa + \mathbf{1}\{\boldsymbol{x} = \boldsymbol{x}'\}$. For query $\boldsymbol{q}$, the true attention scores are $\gamma(\kappa + \boldsymbol{p}^K_{h,\ell_*}(r_*))$ on chunk ℓ_* and $\gamma\kappa$ on every other chunk, so their softmax is A^h. The output coordinate at $\boldsymbol{x} \in \mathcal{V}$ is $\beta(\kappa H + F_{\boldsymbol{x}})$, and the common term $\beta\kappa H$ cancels from cross-entropy loss. For $\varepsilon > 0$, the proof below controls the change from $\mathcal{L}_{\mathrm{sym}}$ to $\mathcal{L}$ along the same coefficient path.

For later use, set $R_1 = R(1)$ and $Q_1 = (1 - R_1)/(L - 1)$.

Theorem 8 (Formal version of Theorem 1)**.** *Under Equation* (37)*, there are constants $c_{\gamma,\beta}, C_{\gamma,\beta} > 0$ such that the following holds. Suppose*

$$N \geq C_{\gamma,\beta} LS \exp\left(\frac{C_{\gamma,\beta} H}{L}\right), \qquad \varepsilon \leq \frac{c_{\gamma,\beta}}{L^2 HS}. \tag{52}$$

Then every minimizer over $\mathcal{G}_{\mathrm{oracle}}$ uses hard adjacent selection: for every $\boldsymbol{X} \in \mathcal{X}$, $\ell \in [L]$, and $h \in [H]$, there is $r_{\boldsymbol{X},\ell,h} \in [S-1]$ such that

$$\boldsymbol{p}^K_{h,\ell}(\boldsymbol{X}) = \boldsymbol{e}_{r_{\boldsymbol{X},\ell,h}}, \qquad \boldsymbol{p}^V_{h,\ell}(\boldsymbol{X}) = \boldsymbol{e}_{r_{\boldsymbol{X},\ell,h}+1}. \tag{53}$$

If $\varepsilon = 0$, the minimizers are exactly those assignments for which, for every $\boldsymbol{X}$ and ℓ, the map $h \mapsto r_{\boldsymbol{X},\ell,h}$ is injective.

The first lemma rules out mixed coefficients, and the second compares the remaining one-hot route assignments under exact geometry.

Lemma 9 (Projection to adjacent vertices)**.** *Under the assumptions of Theorem 8, for each context $\boldsymbol{X}$ and coefficient collection $\mathbf{p}$, there is a one-hot adjacent collection with strictly smaller $\mathcal{L}_{\boldsymbol{X}}$ unless $\mathbf{p}$ already has this form.*

Proof. Fix a context and write $\mathbb{E}$ for its uniform-query average. For each (ℓ, h), choose $\hat{r}_{\ell,h} \in [S-1]$ maximizing $\boldsymbol{p}^K_{h,\ell}(\hat{r}) + \boldsymbol{p}^V_{h,\ell}(\hat{r}+1)$, and replace the two coefficient vectors by $\boldsymbol{e}_{\hat{r}_{\ell,h}}$ and $\boldsymbol{e}_{\hat{r}_{\ell,h}+1}$. Denote the resulting one-hot adjacent collection by $\widehat{\boldsymbol{p}}$. Suppressing (ℓ, h), set $\zeta = \min\{R_1, \min_{u\in[0,1]} R'(u)\} > 0$. The mean-value theorem and maximality of $\hat{r}$ give

$$\begin{aligned} R_1 - \sum_{r=1}^{S-1} R(\boldsymbol{p}^K(r))\boldsymbol{p}^V(r+1) &\geq \zeta\left(1 - \sum_{r=1}^{S-1} \boldsymbol{p}^K(r)\boldsymbol{p}^V(r+1)\right) \\ &\geq \frac{\zeta}{4}\left(\|\boldsymbol{p}^K - \boldsymbol{e}_{\hat{r}}\|_1 + \|\boldsymbol{p}^V - \boldsymbol{e}_{\hat{r}+1}\|_1\right). \end{aligned}$$

The last step uses $uv \leq (u+v)^2/4$ and the fact that a simplex point's distance to a vertex is twice its missing vertex mass.

Let the query-normalized projection distance be

$$\Delta = \frac{1}{L(S-1)} \sum_{\ell,h} \left(\|\boldsymbol{p}^K_{h,\ell} - \widehat{\boldsymbol{p}}^K_{h,\ell}\|_1 + \|\boldsymbol{p}^V_{h,\ell} - \widehat{\boldsymbol{p}}^V_{h,\ell}\|_1\right). \tag{54}$$

Averaging this per-head gap over the uniform query gives

$$\mathbb{E}[\widehat{F}_{\boldsymbol{y}} - F_{\boldsymbol{y}}] \geq \frac{\zeta}{4}\Delta. \tag{55}$$

Inserting the output with projected attention but original values, using $\mathbb{E}\widehat{A}^h_\ell = 1/L$, and applying the Lipschitz bound for R give

$$\mathbb{E}\|F - \widehat{F}\|_1 \leq \max\{S-1, 2\max_{u\in[0,1]} R'(u)\}\Delta. \tag{56}$$

Both outputs have mass H and entries in $[0, HR_1]$. Along the segment between them, subtracting 1 from each exponential in the derivative of the log-normalizer bounds its change by $\beta(e^{\beta H R_1} - 1)\mathbb{E}\|F - \widehat{F}\|_1/(2N)$. Hence Equations (55) and (56) give

$$\mathcal{L}_{\mathrm{sym},\boldsymbol{X}}(\mathbf{p}) - \mathcal{L}_{\mathrm{sym},\boldsymbol{X}}(\widehat{\mathbf{p}}) \geq \beta\Delta\left\{\frac{\zeta}{4} - \frac{\max\{S-1, 2\max_{u\in[0,1]} R'(u)\}(e^{\beta H R_1} - 1)}{2N}\right\}. \tag{57}$$

Here $\zeta \geq \gamma/[L(e^\gamma + 1)]$ and $R_1 \leq e^\gamma/L$. The theorem's vocabulary bound makes the braces at least $\zeta/8$.

It remains to transfer this gap to the true Gram matrix. The softmax Jacobian $J(q) = \mathrm{diag}(q) - qq^\top$ satisfies $\|J(q)v\|_1 \leq 2\|v\|_\infty$ and $\|(J(q) - J(q'))v\|_1 \leq 3\|q - q'\|_1\|v\|_\infty$. By Equation (37), each attention score changes by at most $\gamma\varepsilon$, and its path derivative by at most $\gamma\varepsilon \sum_\ell \|\dot{\boldsymbol{p}}^K_{h,\ell}\|_1$. Applying the Jacobian bounds to attention and then to output cross-entropy gives, for any differentiable coefficient path,

$$\left|\frac{\mathrm{d}}{\mathrm{d}t}(\mathcal{L}_{\boldsymbol{X}} - \mathcal{L}_{\mathrm{sym},\boldsymbol{X}})(\boldsymbol{p}(t))\right| \leq C_{\gamma,\beta}(1 + \beta H)\varepsilon \sum_{\ell,h,P} \|\dot{\boldsymbol{p}}^P_{h,\ell}(t)\|_1. \tag{58}$$

Indeed, after subtracting the common $\beta\kappa H$ from the output logits, their true–comparison difference and its path derivative are bounded respectively by $O_{\gamma,\beta}(H\varepsilon)$ and $O_{\gamma,\beta}(\varepsilon \sum_{\ell,h,P} \|\dot{\boldsymbol{p}}^P_{h,\ell}\|_1)$. The cross-entropy gradient is 2-bounded in ℓ_1 and 2-Lipschitz from ℓ_∞ to ℓ_1. Integrating Equation (58) along $\boldsymbol{p}(t) = (1-t)\widehat{\boldsymbol{p}} + t\boldsymbol{p}$ changes the comparison gap by at most $C_{\gamma,\beta}L(S-1)(1+\beta H)\varepsilon\Delta$. The theorem's bound on ε makes this less than $\beta\zeta\Delta/16$. With Equation (57), the true gap is positive whenever $\Delta > 0$. An oracle can make the replacement for one context without changing any other context. □

Lemma 10 (Distinct routes under exact geometry)**.** *Under the assumptions of Theorem 8 with $\varepsilon = 0$, all injective one-hot adjacent assignments have the same fixed-context loss, and every assignment with a head collision has strictly larger loss.*

Proof. Fix a context, write $r_{\ell,h} \in [S-1]$ for the route of head h in chunk ℓ, and let $k = \sum_h \mathbf{1}\{r_{\ell_*,h} = r_*\}$ for a uniform query. Then

$$\mathbb{E}k = \frac{H}{S-1} \leq 1, \qquad \mathbb{E}k^2 \leq H. \tag{59}$$

Every assignment has the same expected target logit $\mathbb{E}F_{\boldsymbol{y}} = HR_1/(S-1)$, so only the output normalizer matters. Write Z_{inj} for this normalizer under an injection. Then $Z_{\mathrm{inj}} = Z_0 + kd_{\mathrm{inj}}$, where

$$Z_0 := N - LH + LHe^{\beta/L}, \qquad d_{\mathrm{inj}} := e^{\beta R_1} + (L-1)e^{\beta Q_1} - Le^{\beta/L} > 0. \tag{60}$$

The L numbers $R_1, Q_1, \ldots, Q_1$ have mean $1/L$ and are not all equal, so the last inequality follows from strict convexity. Thus all injections have the same expected target and normalizer terms, and

$$\mathbb{E}[\log Z_{\mathrm{inj}}] \leq \log Z_0 + \frac{d_{\mathrm{inj}}H}{(S-1)Z_0}. \tag{61}$$

For a collided assignment, contributions to a shared output coordinate add before exponentiation. Writing $e^{\beta A} = 1 + (e^{\beta A} - 1)$, the pair product from any collision adds at least $d_{\mathrm{col}} := (e^{\beta Q_1} - 1)^2 > 0$ to the normalizer, so $Z_{\mathrm{col}} \geq Z_0 + kd_{\mathrm{inj}} + d_{\mathrm{col}}$. Using $\log(1+t) \geq t - t^2/2$ and $\mathbb{E}k^2 \leq H$ gives

$$\mathbb{E}[\log Z_{\mathrm{col}}] - \mathbb{E}[\log Z_{\mathrm{inj}}] \geq \frac{d_{\mathrm{col}}}{Z_0} - \frac{d^2_{\mathrm{inj}}H + 2d_{\mathrm{inj}}d_{\mathrm{col}} + d^2_{\mathrm{col}}}{2Z_0^2}. \tag{62}$$

Taylor's theorem at $1/L$ and $R_1 = e^\gamma Q_1$ give $d_{\mathrm{inj}} \leq C_{\gamma,\beta}Q_1^2$, whereas $d_{\mathrm{col}} \geq \beta^2 Q_1^2$ and $d_{\mathrm{col}} \leq (e^\beta - 1)^2$. The numerator in the error term is therefore at most $C_{\gamma,\beta}Hd_{\mathrm{col}}$. Since $Z_0 \geq N$ and the vocabulary bound can be enlarged to ensure $N > C_{\gamma,\beta}H$, the displayed gap is positive. The expected target term is the same for both assignments, so every collision strictly increases the fixed-context loss. □

Proof of Theorem 8. The oracle class is a product over contexts, so each fixed-context loss can be minimized separately. Lemma 9 makes every minimizer one-hot and adjacent. When $\varepsilon = 0$, Lemma 10 shows that precisely the injective route assignments minimize the loss among them. □

The injection may depend on the context and chunk, so the oracle result does not assign a fixed route to each named head.

A.4 Proof of Theorem 2

The oracle result says which compressors are optimal but not whether training finds them, or whether a trained head keeps the same selection for every input. As in the toy example, we answer this by starting gradient flow near zero, where all heads agree and no useful phase is preferred over another, and asking along which directions the flow leaves the symmetric path that starts there.

Let $\boldsymbol{\mu} := N^{-1}\sum_{\boldsymbol{x}\in\mathcal{V}}\boldsymbol{x}$. We first collect the assumptions for the dynamics result. Its effective parameter space is

$$\mathcal{C}_{\text{eff}} := \left\{ \boldsymbol{c} : \boldsymbol{c}_{h,r}^{P} \in \operatorname{span}(\mathcal{V}), \quad \sum_{r=1}^{S} \boldsymbol{\mu}^{\top}\boldsymbol{c}_{h,r}^{P} = 0 \text{ for every } P \in \{K, V\},\ h \in [H] \right\}. \tag{63}$$

This removes embedding-orthogonal directions and a common mean-logit shift in each head and branch, which do not change the gates.

Assumption 11 (Setting for the gradient-flow result)**.**

1. ***Retrieval distribution.*** *We use the model of Appendix A.2, with $N \geq LS$ vocabulary embeddings. Contexts contain L non-overlapping chunks of S distinct tokens each, sampled uniformly without replacement across the entire context. After compression, the query is uniform over the $L(S-1)$ nonfinal positions and requests the next token in the same chunk.*
2. ***Fixed model parameters.*** *We fix $L \geq 3$, $S \geq 3$, $1 \leq H \leq S-1$, $0 < \gamma < \log(L-1)$, and $\beta > 0$ as N varies.*
3. ***Exact geometry and positive mean.*** *The embeddings satisfy Equation* (37) *with $\varepsilon = 0$, and the mean norm $\|\boldsymbol{\mu}\| > 0$ is fixed as N varies.*
4. ***Gate-only population training.*** *Only the offset-specific linear gate vectors in $\mathcal{G}_{\text{linear}}$, shared across contexts and chunks, are trained; the embeddings and all other maps remain fixed as in Appendix A.2. We use Euclidean gradient flow on the population loss:*
$$\dot{\boldsymbol{c}} = -\nabla_{\boldsymbol{c}}\{\mathcal{L}(g_{\boldsymbol{c}})/\beta\}. \tag{64}$$
The factor $1/\beta$ only rescales time.
5. ***Initialization.*** *For $\rho > 0$, $\boldsymbol{c}(0)$ is uniform with respect to Euclidean volume on the radius-ρ ball centered at zero in $\mathcal{C}_{\text{eff}}$. The required joint bounds on ρ and N are specified in Theorem 12.*

This initialization includes input-dependent gate directions. The proof below establishes the emergence of static preferences from these starts. Under exact geometry, averaging the Gram identity gives $\boldsymbol{x}^{\top}\boldsymbol{\mu} = \|\boldsymbol{\mu}\|^2 = \kappa + 1/N$, and hence

$$\boldsymbol{\mu}^{\top}(\boldsymbol{x}-\boldsymbol{\mu}) = 0, \qquad (\boldsymbol{x}-\boldsymbol{\mu})^{\top}(\boldsymbol{x}'-\boldsymbol{\mu}) = \mathbf{1}\{\boldsymbol{x}=\boldsymbol{x}'\} - \frac{1}{N} \quad (\boldsymbol{x},\boldsymbol{x}' \in \mathcal{V}). \tag{65}$$

The exact Gram matrix has eigenvalues 1 on vocabulary contrasts and $N\|\boldsymbol{\mu}\|^2 > 0$ on the constant direction, so its rank is N and this idealization requires $d \geq N$. The ambient gradient is tangent to $\mathcal{C}_{\text{eff}}$: each parameter gradient is a linear combination of embeddings, and the gate-logit gradients in each row sum to zero. Since $\|\boldsymbol{x}\|^2 = \|\boldsymbol{\mu}\|^2 + 1 - 1/N$ for $\boldsymbol{x} \in \mathcal{V}$, the vector field in Equation (64) is smooth and globally bounded for each N. Its trajectories therefore exist for all positive time. Below, $\boldsymbol{p}_{h,\ell}^{P}(r; t, \boldsymbol{X}) := \boldsymbol{p}_{h,\ell}^{P}(r; \boldsymbol{c}(t), \boldsymbol{X})$.

Theorem 12 (Formal version of Theorem 2)**.** *Under Assumption 11, for every $0 < \delta < 1$, there are $0 < c_\delta \leq 1$ and $C_\delta < \infty$ such that the following holds. If*

$$0 < \rho \leq c_\delta, \qquad N \geq C_\delta \log\frac{e}{\rho}, \tag{66}$$

then, with probability at least $1-\delta$, there are $r_1, \ldots, r_H \in [S-1]$ such that

$$\max_{h\in[H]} \sup_{\boldsymbol{X}\in\mathcal{X},\, \ell\in[L]} \max\left\{1 - \boldsymbol{p}_{h,\ell}^{K}(r_h; t, \boldsymbol{X}), 1 - \boldsymbol{p}_{h,\ell}^{V}(r_h+1; t, \boldsymbol{X})\right\} \longrightarrow 0 \quad \text{as } t \to \infty. \tag{67}$$

The routes r_h need not be distinct. The constants c_δ, C_δ can be chosen uniformly for $\|\boldsymbol{\mu}\|$ in any compact subset of $(0,\infty)$. Hence fixing $\kappa > 0$ suffices, since then $\|\boldsymbol{\mu}\|^2 = \kappa + 1/N \in (\kappa, \kappa+1]$.

The lemmas below track the dynamics from the symmetric zero-start path to aligned winners and then to persistent one-hot routing. Winners are aligned when each head's key gate favors some phase $r_h \in [S-1]$ and its value gate favors the next phase $r_h + 1$.

In the supporting lemmas, C denotes a positive constant depending only on L, S, H, γ, β, and the fixed value of $\|\boldsymbol{\mu}\|$, and its value may change between displays.

Let $\mathcal{M}$ be the mean-aligned subspace

$$\boldsymbol{c}^P_{h,r} = \frac{b^P_{h,r}}{\|\boldsymbol{\mu}\|^2}\boldsymbol{\mu}, \qquad \sum_{r=1}^{S} b^P_{h,r} = 0, \qquad P \in \{K, V\}. \tag{68}$$

Since $\boldsymbol{\mu}^\top \boldsymbol{x} = \|\boldsymbol{\mu}\|^2$ for $\boldsymbol{x} \in \mathcal{V}$, $b^P_{h,r}$ is the input-independent gate logit of phase r. Write $P_{\mathcal{M}}$ for the orthogonal projection onto $\mathcal{M}$ and $\mathcal{T} := \mathcal{M}^\perp \cap \mathcal{C}_{\text{eff}}$.

The linear class raises a question that the toy example cannot pose. There, each head's compression is a single trainable logit with no input, so any preference is automatically the same for every input. With linear gates, the gate logits depend on the embeddings, and a preference could change from one context to another. On $\mathcal{M}$, however, the gate logit $b^P_{h,r}$ of each phase is the same for every input, just as θ_h is in the toy example, so we track the flow near $\mathcal{M}$. The flow from zero stays on $\mathcal{M}$ with all heads identical, which plays the role of the toy example's uniform point (Lemma 14). Differences between routes are unstable along this path, and the unstable directions raise a key phase together with the value phase after it. As in the toy example, selection thus comes from an instability of the symmetric state, but here the instability acts on each head separately, since different heads do not couple in the limit. From a small random start, each head acquires such a route with high probability (Lemma 15), which is then amplified and persists for every input (Lemmas 16 and 17).

Lemma 13 (Vocabulary suppression)**.** *For fixed context and query, the first three derivatives of the output log-normalizer divided by β are $O(N^{-1})$, uniformly over $(\Delta_S \times \Delta_S)^{LH}$. This holds for derivatives in the compression coefficients, sampled gate logits $\langle \boldsymbol{c}^P_{h,r}, \boldsymbol{x}_{\ell,r}\rangle$, and compressor parameters, using Euclidean product norms. In particular, its contributions to coefficient gradients and the parameter linearization are $O(N^{-1})$.*

Proof. When $\varepsilon = 0$, the common vocabulary logit cancels and the remaining logits are $\beta F_{\boldsymbol{x}}$, with $F_{\boldsymbol{x}}$ as in Appendix A.3 after substituting the softmax coefficients. Thus $F_{\boldsymbol{x}} = 0$ outside the context, $0 \le F_{\boldsymbol{x}} \le H$, and the normalized log-normalizer is

$$\frac{1}{\beta}\log\left\{N + \sum_{\ell=1}^{L}\sum_{r=1}^{S}(e^{\beta F_{\boldsymbol{x}_{\ell,r}}} - 1)\right\}. \tag{69}$$

The first three derivatives of $F_{\boldsymbol{x}}$ in sampled gate logits or compression coefficients are uniformly bounded because they use fixed-dimensional softmax maps. The nonconstant part of the expression inside the logarithm, and its first three derivatives, are bounded independently of N, whereas that expression is at least N. Differentiation gives the $O(N^{-1})$ bounds. The sampled-logit map is linear in the compressor parameters and has operator norm at most $C\sup_{\boldsymbol{x}\in\mathcal{V}}\|\boldsymbol{x}\| \le C$, yielding the parameter bound. □

Define the rowwise route-contrast space

$$\mathcal{W} := \left\{U \in \mathbb{R}^{H\times(S-1)} : \sum_{r=1}^{S-1} U_{h,r} = 0 \text{ for every } h\right\}. \tag{70}$$

Let $\mathcal{F}(\boldsymbol{c}) = -\nabla_{\boldsymbol{c}}\{\mathcal{L}(g_{\boldsymbol{c}})/\beta\}$, let $\boldsymbol{c}^0_N(t)$ be its trajectory from zero, and let $A_N(t)$ be the derivative on $\mathcal{M}$ of the target part of $\mathcal{F}$ at $\boldsymbol{c}^0_N(t)$.

Lemma 14 (Symmetric baseline and route-contrast growth)**.** *For sufficiently large N, the trajectory $\boldsymbol{c}^0_N(t)$ remains in $\mathcal{M}$. All heads agree, and key phases $1, \dots, S-1$ and value phases $2, \dots, S$ are respectively identical. Call key phase S and value phase 1 the dummy phases, since no useful route uses them and they serve no query. If ω_K, ω_V denote the dummy-phase coefficients, then*

$$\omega_K(t), \omega_V(t) \asymp (1+t)^{-1}, \qquad \log\frac{1-\omega_P(t)}{(S-1)\omega_P(t)} \longrightarrow \infty \quad (P \in \{K, V\}). \tag{71}$$

Moreover, $A_N(t)$ converges to an operator A_∞ that acts on each route contrast, in the order key then successor value, by $\|\boldsymbol{\mu}\|^2$ times the block

$$\begin{pmatrix} \dfrac{R''(1/(S-1))}{(S-1)^4} & \dfrac{R'(1/(S-1))}{(S-1)^3} \\ \dfrac{R'(1/(S-1))}{(S-1)^3} & 0 \end{pmatrix}, \tag{72}$$

and vanishes on the useful-versus-dummy directions, with

$$\|A_N(t) - A_\infty\| \le \frac{C}{1+t}. \tag{73}$$

The positive eigenspace of A_∞ is a copy of $\mathcal{W}$, with same-sign key and successor-value coordinates.

Proof. Vocabulary-permutation, head, and useful-phase symmetries keep $\boldsymbol{c}_N^0(t)$ in $\mathcal{M}$ and give useful coefficients $(1-\omega_K)/(S-1)$ and $(1-\omega_V)/(S-1)$. For either row,

$$\dot\omega_P = -\|\boldsymbol{\mu}\|^2 \frac{S}{S-1} d_P(t)\omega_P^2(1-\omega_P)^2, \qquad P \in \{K, V\}, \tag{74}$$

where $d_P(t)$ is the useful-minus-dummy fitness gap for $\mathcal{L}/\beta$. Here the fitness of a phase is the negative derivative of $\mathcal{L}/\beta$ with respect to its compression coefficient, and the fitness gap is the fitness of a useful phase minus that of the dummy phase in the same row. The target term gives $d_V \ge 1/[L(S-1)]$. Once $\omega_V \le 1/S$, it also gives $d_K \ge R'(0)/[S(S-1)]$. By Lemma 13, these lower bounds survive for large N, and both gaps are also uniformly bounded above. Integrating the displayed equation from $\omega_P(0) = 1/S$ proves Equation (71).

Route symmetry separates useful contrasts from useful–dummy directions. At the limiting coefficients, softmax sends a useful contrast y to $y/(S-1)$, and its second derivatives sum to zero. The target Hessian on key and successor-value contrasts is therefore

$$\begin{aligned} &D^2\left\{\frac{1}{S-1}\sum_{r=1}^{S-1} R(\boldsymbol{p}_h^K(r))\boldsymbol{p}_h^V(r+1)\right\}[(y_K, y_V)]^2 \\ &\quad = \frac{R''(1/(S-1))\|y_K\|^2}{(S-1)^4} + \frac{2R'(1/(S-1))\langle y_K, y_V\rangle}{(S-1)^3}. \end{aligned} \tag{75}$$

This gives Equation (72). The useful–dummy directions vanish in the limit because the dummy coefficient vanishes, and different heads do not couple. The block has negative determinant, and its positive eigenvector has same-sign coordinates. Smoothness and $\omega_P(t) = O((1+t)^{-1})$ give the stated convergence rate. □

Lemma 15 (Aligned route gaps from random initialization)**.** *For every $0 < \delta < 1$, there are $\eta_\delta, C_\delta > 0$ and $0 < c_\delta \le 1$ such that, if $0 < \rho \le c_\delta$ and $N \ge C_\delta \log(e/\rho)$, and if $\boldsymbol{c}(0)$ is uniform on the radius-ρ ball in $\mathcal{C}_{\mathrm{eff}}$, then, with probability at least $1-\delta$, there are a time*

$$\tau \le C_\delta \log \frac{e\sqrt{N}}{\rho} \tag{76}$$

and a point $\bar{\boldsymbol{c}} \in \mathcal{M}$ satisfying

$$\|\boldsymbol{c}(\tau) - \bar{\boldsymbol{c}}\| \le C_\delta \rho, \tag{77}$$

such that every head has a route $r_h \in [S-1]$ with

$$b_{h,r_h}^K - \max_{j \ne r_h} b_{h,j}^K \ge 2\eta_\delta, \qquad b_{h,r_h+1}^V - \max_{j \ne r_h+1} b_{h,j}^V \ge 2\eta_\delta. \tag{78}$$

Proof. The proof first splits the flow linearized along $\boldsymbol{c}_N^0(t)$ into growing and complementary parts, and then shows that deviations orthogonal to $\mathcal{M}$ stay small. It next bounds the random initialization, defines an exit time τ from the linearized flow, and compares the nonlinear flow with its linearization up to τ. At that time, the growing part carries aligned route gaps.

The target is a sum over heads, and simultaneous permutation of useful key and successor-value indices separates useful contrasts from useful–dummy directions. Thus, in the b-coordinates, $A_N(t)$

is $(B_N(t) \otimes I_{\mathcal{W}}) \oplus D_N(t)$, where $B_N(t)$ is two-dimensional, B_∞ is the block in Equation (72), including its factor $\|\boldsymbol{\mu}\|^2$, and $\|B_N(t) - B_\infty\| + \|D_N(t)\| \le C/(1+t)$. Let $\lambda_+ > 0 > \lambda_-$ be the eigenvalues of B_∞ and put $\Delta_\lambda = \lambda_+ - \lambda_-$. In an orthonormal eigenbasis, write $B_N(t) = \operatorname{diag}(\lambda_+, \lambda_-) + E_N(t)$. Choose T, uniformly in large N, so that $\|E_N(t)\|, \|D_N(t)\| \le \eta_*$ for $t \ge T$, with η_* sufficiently small. Two invariant contrast lines are $x_- = r(t)x_+$ and $x_+ = z(t)x_-$, defined by

$$\dot r = -\Delta_\lambda r + E_{21} + (E_{22} - E_{11})r - E_{12}r^2, \qquad r(T) = 0,$$
$$z(t) = -\int_t^\infty e^{-\Delta_\lambda(u-t)}[E_{12} + (E_{11} - E_{22})z - E_{21}z^2](u)\,\mathrm{d}u.$$

For $\zeta = 4\eta_*/\Delta_\lambda < 1/2$, the first vector field points inward at $r = \pm\zeta$, and the second equation is a contraction on $\|z\|_\infty \le \zeta$. Hence these lines are uniformly transverse. The scalar convolution in the first equation and $E_N(t) \to 0$ give $r(t) \to 0$, uniformly in large N. Their scalar rates are $\lambda_+ + E_{11} + E_{12}r$ and $\lambda_- + E_{22} + E_{21}z$, respectively. Taking η_* small, the former is at least $a > 0$, the latter is negative, and $\eta_* \le a/2$. Use these same lines for every head and route contrast, placing the whole useful–dummy sector in the complement. This defines invariant projections $P_+(t)$ and $P_\perp(t) = I - P_+(t)$ with uniformly bounded norms. Extend them to $[0,T]$ by the propagator $\Phi(t,s)$ of $\dot u = A_N(t)u$; boundedness of A_N on this fixed interval bounds both transport directions uniformly in N. Write $\Phi_j(t,s) = P_j(t)\Phi(t,s)P_j(s)$ for $j \in \{+,\perp\}$. The preceding scalar rates and $\|D_N(t)\| \le \eta_*$ give, after adjusting C, for $t \ge s \ge 0$,

$$\begin{aligned} \|\Phi_+(t,s)^{-1}\| &\le Ce^{-a(t-s)}, \\ \|\Phi_+(t,s)\| &\le Ce^{C(t-s)}, \\ \|\Phi_\perp(t,s)\| &\le Ce^{a(t-s)/2}. \end{aligned} \tag{79}$$

The inverse is on the growing fiber; the zero limiting modes are included in the complementary bound. All constants are uniform in large N.

In the original key/value coordinates, choose the positive unit eigenvector e_+ of B_∞ and let $Q_N(t)$ be the contrast solution with $Q_N(T) = e_+$. The growing fiber is $Q_N(t) \otimes \mathcal{W}$ and $Q_N(t)/\|Q_N(t)\| \to e_+$. Let α_N take the coefficient of $Q_N(0)$ in the splitting along the complementary contrast line, and define $\Lambda_N \boldsymbol{c}$ by applying $\alpha_N \otimes I_{\mathcal{W}}$ to the useful-contrast part of $P_{\mathcal{M}}\boldsymbol{c}$, expressed in b-coordinates. Fixed-interval transport bounds $\|Q_N(0)\|$ above and away from zero; transversality and $\alpha_N Q_N(0) = 1$ do the same for $\|\alpha_N\|$. The useful-contrast projection is an orthogonal surjection, and $\|b\| = \|\boldsymbol{\mu}\|\|\boldsymbol{c}\|$ on $\mathcal{M}$, so all nonzero singular values of Λ_N are uniformly bounded above and below. In particular, every growing solution is $Q_N(t) \otimes w$ for a fixed $w \in \mathcal{W}$.

To control deviations from the mean-aligned trajectory, for a parameter direction w define

$$\|w\|_{\boldsymbol{X}}^2 = \sum_{P,h,\ell,r} \big((w_{h,r}^P)^\top \boldsymbol{x}_{\ell,r}\big)^2. \tag{80}$$

The $\varepsilon = 0$ geometry and sampling without replacement imply

$$\mathbb{E}_{\boldsymbol{X}}\|w\|_{\boldsymbol{X}}^2 = \frac{L}{N}\|w\|^2 \quad (w \in \mathcal{T}), \qquad \|w\|_{\boldsymbol{X}} = \sqrt{L}\,\|\boldsymbol{\mu}\|\,\|w\| \quad (w \in \mathcal{M}). \tag{81}$$

The fixed-context objective is smooth in the sampled gate logits. Its target derivatives are bounded, and its normalizer derivatives are $O(N^{-1})$ by Lemma 13. Vocabulary equivariance gives $\mathcal{F}(\bar{\boldsymbol{c}}) \in \mathcal{M}$ and $P_{\mathcal{M}} D\mathcal{F}(\bar{\boldsymbol{c}})[u] = 0$ for $\bar{\boldsymbol{c}} \in \mathcal{M}$, $u \in \mathcal{T}$. Applying Equation (81) to the first two derivatives and using this cancellation gives, for $u \in \mathcal{T}$ with $\|u\| \le 1$,

$$\|(I - P_{\mathcal{M}})\mathcal{F}(\bar{\boldsymbol{c}} + u)\| \le \frac{C}{N}\|u\|, \qquad \|P_{\mathcal{M}}\mathcal{F}(\bar{\boldsymbol{c}} + u) - \mathcal{F}(\bar{\boldsymbol{c}})\| \le \frac{C}{N}\|u\|^2. \tag{82}$$

The effective parameter space has dimension $2H(SN-1)$. Uniform radius-ρ ball initialization is an isotropic Gaussian direction times an independent radial factor. Gaussian concentration, the uniform singular-value bounds for Λ_N, and a finite union bound over the nondegenerate Gaussian head–route contrasts give $a_\delta, A_\delta, q_\delta > 0$ such that, with probability at least $1 - \delta$,

$$a_\delta \frac{\rho}{\sqrt{N}} \le g_0 := \|\Lambda_N \boldsymbol{c}(0)\| \le A_\delta \frac{\rho}{\sqrt{N}}, \quad \|P_{\mathcal{M}}\boldsymbol{c}(0)\| \le A_\delta \frac{\rho}{\sqrt{N}}, \quad \min_h \operatorname{gap}((\Lambda_N \boldsymbol{c}(0))_h) \ge q_\delta g_0. \tag{83}$$

Here gap is the largest minus second-largest row entry.

On $\mathcal{M}$, use the Euclidean norm of the b-array, which is uniformly equivalent to the parameter norm because $\|b\| = \|\boldsymbol{\mu}\|\|\boldsymbol{c}\|$ there. Let u_{lin} solve $\dot{u} = A_N(t)u$ from $u_{\mathrm{lin}}(0) = P_{\mathcal{M}}\boldsymbol{c}(0)$. The estimates below hold for all exit radii in a fixed sufficiently small interval. Choose r_δ in that interval below, and let τ be the first time $\|P_+(\tau)u_{\mathrm{lin}}(\tau)\| = r_\delta$. From Equations (79) and (83),

$$A_\delta^{-1} \log \frac{r_\delta\sqrt{N}}{\rho} \le \tau \le A_\delta \log \frac{r_\delta\sqrt{N}}{\rho}. \tag{84}$$

The lemma's lower bound on N, after increasing C_δ, makes τ/N arbitrarily small.

Define

$$u_{\mathcal{M}}(t) := P_{\mathcal{M}}(\boldsymbol{c}(t) - \boldsymbol{c}_N^0(t)), \qquad u_{\mathcal{T}}(t) := (I - P_{\mathcal{M}})\boldsymbol{c}(t). \tag{85}$$

The first bound in Equation (82) and Gronwall give

$$\|u_{\mathcal{T}}(t)\| \le A_\delta\rho \qquad (0 \le t \le \tau). \tag{86}$$

Shrinking c_δ closes the assumed unit token tube. Taylor's theorem, Lemma 13, and the second bound in Equation (82) give

$$\dot{u}_{\mathcal{M}} = A_N(t)u_{\mathcal{M}} + E(t), \qquad \|E(t)\| \le C\left(\frac{\|u_{\mathcal{M}}\|}{N} + \|u_{\mathcal{M}}\|^2 + \frac{\|u_{\mathcal{T}}\|^2}{N}\right). \tag{87}$$

Put $v(t) = P_+(t)u_{\mathrm{lin}}(t)$. By Equations (79) and (83) and scalarity on $\mathcal{W}$, for $0 \le s \le t \le \tau$,

$$\begin{gathered}\|v(s)\| \le Ce^{-a(t-s)}\|v(t)\|, \qquad \|\Phi_+(t,s)\|\|v(s)\| \le C\|v(t)\|, \\ \|P_\perp(t)u_{\mathrm{lin}}(t)\| \le A_\delta e^{-at/2}\|v(t)\|.\end{gathered} \tag{88}$$

In particular, $\|v(t)\| \le Cr_\delta e^{-a(\tau-t)}$ before exit. The final bound is the relative decay: the inverse growing propagator contributes e^{-at} and the complementary propagator contributes $e^{at/2}$.

Variation of constants for Equation (87) gives $u_{\mathcal{M}}(t) = u_{\mathrm{lin}}(t) + \int_0^t \Phi(t,s)E(s)\,\mathrm{d}s$. Scalarity, bounded projections, and Equation (88) give the full-propagator bounds, for $0 \le s \le t \le \tau$,

$$\|\Phi(t,s)\|\|v(s)\| \le C\|v(t)\|, \qquad \|\Phi(t,s)\| \le C\frac{\|v(t)\|}{g_0}e^{-as}.$$

Indeed, the complementary contribution to the first bound is at most $Ce^{-a(t-s)/2}\|v(t)\|$, and $\|v(s)\| \ge C^{-1}g_0e^{as}$ gives the second. On the bootstrap bound $\|u_{\mathcal{M}}(s)\| \lesssim_\delta \|u_{\mathrm{lin}}(s)\| + \rho^2\|v(s)\|/(Ng_0)$, the right-hand side is $O_\delta(\|v(s)\|)$ when $\rho^2/(Ng_0) \le 1$. Together with $\int_0^t \|v(s)\|\,\mathrm{d}s \le C\|v(t)\| \le Cr_\delta$ and Equation (86), these kernel bounds yield, uniformly for $t \le \tau$,

$$\begin{aligned}\int_0^t \|\Phi(t,s)\|\|u_{\mathcal{M}}(s)\|^2\,\mathrm{d}s &\lesssim_\delta r_\delta\|v(t)\|, \\ \frac{1}{N}\int_0^t \|\Phi(t,s)\|\|u_{\mathcal{M}}(s)\|\,\mathrm{d}s &\lesssim_\delta \frac{t}{N}\|v(t)\|, \\ \frac{1}{N}\int_0^t \|\Phi(t,s)\|\|u_{\mathcal{T}}(s)\|^2\,\mathrm{d}s &\lesssim_\delta \frac{\rho^2}{Ng_0}\|v(t)\|.\end{aligned} \tag{89}$$

For small r_δ, τ/N, and $\rho^2/(Ng_0)$, these estimates improve the bootstrap, so continuity closes it. Since $g_0 \ge a_\delta\rho/\sqrt{N}$, the last term at $t = \tau$ is $O_\delta(r_\delta\rho/\sqrt{N})$. Adding the complementary linear component gives

$$\|u_{\mathcal{M}}(\tau) - v(\tau)\| \lesssim_\delta r_\delta^2 + \frac{r_\delta\tau}{N} + \frac{r_\delta\rho}{\sqrt{N}} + r_\delta e^{-a\tau/2}. \tag{90}$$

The growing fiber converges to the positive eigenvector of Equation (72), whose key and value coordinates have the same positive sign. Thus Equation (83) gives the leading term $v(\tau)$ the same winning route in the key row and the successor-value row, with every winner–loser gap among useful phases at least a fixed δ-dependent multiple of r_δ. Choose r_δ small, then C_δ large and c_δ small, so Equation (90) is smaller than these gaps. The lower exit-time bound and Equation (71) make the useful–dummy gaps positive as well. Hence $\bar{\boldsymbol{c}} := P_{\mathcal{M}}\boldsymbol{c}(\tau)$ has aligned adjacent winners with a common gap $2\eta_\delta > 0$, while Equation (86) gives $\|\boldsymbol{c}(\tau) - \bar{\boldsymbol{c}}\| \le A_\delta\rho$. □

Lemma 16 (Mean amplification)**.** *The space $\mathcal{M}$ is invariant. Fix $0 < \eta < M$. For fixed head h and branch P, call the phase with the largest gate logit $b^P_{h,r}$ the winner and the other phases losers. For all sufficiently large N, suppose a trajectory in $\mathcal{M}$ has, in every head, a key winner r_h, the value winner $r_h + 1$, and all corresponding winner–loser gate-logit gaps at least η. These winners persist, every gap reaches M in time bounded independently of N, and every gap then diverges.*

Proof. Every vocabulary permutation is represented by an orthogonal map that fixes $\boldsymbol{\mu}$ and sends $\boldsymbol{x}-\boldsymbol{\mu}$ to the corresponding permuted residual. The loss is invariant and its gradient field is equivariant under this action. Its common fixed space in $\mathcal{C}_{\text{eff}}$ is $\mathcal{M}$, proving invariance. The induced flow in the coordinates of Equation (68) is the gate-logit flow multiplied by $\|\boldsymbol{\mu}\|^2$.

On $\mathcal{M}$, write $\boldsymbol{p}^P_h(r)$ for the coefficient shared by all chunks. A compressor row is the coefficient vector of one head and branch. Within a compressor row, write its coefficients as p_i, its winner as s, and the fitness of phase i as $f_i = -\partial(\mathcal{L}/\beta)/\partial p_i$. Differentiating the target term in Equation (75), the target contributions to f_i for key phase $r < S$ and value phase $r+1$ are $R'(\boldsymbol{p}^K_h(r))\boldsymbol{p}^V_h(r+1)/(S-1)$ and $R(\boldsymbol{p}^K_h(r))/(S-1)$, respectively. The dummy key and value fitnesses are zero. If every winner gap is at least $\eta > 0$, then

$$p_s - p_j \geq \frac{1 - e^{-\eta}}{1 + (S-1)e^{-\eta}} \qquad (j \neq s). \tag{91}$$

Because $\gamma < \log(L-1)$ makes R' increasing on $[0,1]$, the target-fitness advantage of either winner is at least

$$g_\eta := \frac{1}{S-1} \min\left\{ R'(0)\frac{1 - e^{-\eta}}{1 + (S-1)e^{-\eta}}, \frac{1}{L} \right\} > 0. \tag{92}$$

By Lemma 13, $f_s - f_j \geq g_\eta/2$ when N is large.

Direct differentiation of the gate-logit flow gives

$$\frac{d}{dt}(b_s - b_j) = \|\boldsymbol{\mu}\|^2 \left[p_j(f_s - f_j) + (p_s - p_j)\left(f_s - \sum_i p_i f_i \right) \right]. \tag{93}$$

The second factor in the last term equals $\sum_i p_i(f_s - f_i)$, so both terms are nonnegative. While $b_s - b_j \leq M$, the losing coefficient is at least e^{-M}/S, so the gap grows at rate at least $\|\boldsymbol{\mu}\|^2 g_\eta e^{-M}/(2S)$. This gives an N-independent time bound to reach M. Repeating at successively larger levels proves divergence. □

Lemma 17 (Persistence of aligned winners)**.** *For every $r_1, \ldots, r_H \in [S-1]$ and all sufficiently large N, there is $M_0 < \infty$, independent of N and the route choice, such that the terminal cone, defined as the region where, for every head and all distinct $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{V}$,*

$$\begin{aligned} \langle \boldsymbol{c}^K_{h,r_h}, \boldsymbol{x}\rangle - \langle \boldsymbol{c}^K_{h,j}, \boldsymbol{x}'\rangle \geq M_0 \qquad & (j \in [S] \setminus \{r_h\}), \\ \langle \boldsymbol{c}^V_{h,r_h+1}, \boldsymbol{x}\rangle - \langle \boldsymbol{c}^V_{h,j}, \boldsymbol{x}'\rangle \geq M_0 \qquad & (j \in [S] \setminus \{r_h+1\}) \end{aligned} \tag{94}$$

is forward invariant, and every displayed margin tends to $+\infty$.

Proof. For each context and compressor row, define

$$f^P_{h,\ell}(i;\boldsymbol{X}) := -\frac{\partial(\mathcal{L}_{\boldsymbol{X}}/\beta)}{\partial \boldsymbol{p}^P_{h,\ell}(i)}, \qquad \psi^P_{h,\ell}(i;\boldsymbol{X}) := \boldsymbol{p}^P_{h,\ell}(i)\left(f^P_{h,\ell}(i;\boldsymbol{X}) - \sum_j \boldsymbol{p}^P_{h,\ell}(j) f^P_{h,\ell}(j;\boldsymbol{X}) \right). \tag{95}$$

Thus $\psi^P_{h,\ell}(i;\boldsymbol{X})$ is the negative gradient with respect to the corresponding sampled gate logit. Fix a context and row, suppress these indices as p_i, f_i, ψ_i, and write s for the selected phase. At a one-hot adjacent corner, the target fitness advantage is $R'(1)/\{L(S-1)\}$ for the selected key and at least $\{R(1) - 1/L\}/\{L(S-1)\}$ for the selected value. Both are positive. Set

$$d_* := \frac{1}{L(S-1)} \min\left\{ R'(1), R(1) - \frac{1}{L} \right\} > 0. \tag{96}$$

Uniform continuity and Lemma 13 imply that, throughout the terminal cone for sufficiently large M_0 and for all large N,

$$f_s - f_j \geq \frac{d_*}{2}, \qquad |f_i - f_j| \leq C. \tag{97}$$

Choose M_0 large enough that these bounds hold and $(S-1)e^{-M_0}C < d_*/4$. Then $1 - p_s \leq (S-1)e^{-M_0}p_s$, and

$$\psi_s = p_s \sum_{i \neq s} p_i(f_s - f_i) > 0, \qquad -\psi_j \geq \frac{d_*}{4} p_j p_s > 0. \tag{98}$$

For $N \geq \|\boldsymbol{\mu}\|^{-2}$, the Gram matrix is entrywise nonnegative:

$$\boldsymbol{x}^\top \boldsymbol{x}' = \|\boldsymbol{\mu}\|^2 - \frac{1}{N} + \mathbf{1}\{\boldsymbol{x} = \boldsymbol{x}'\} \geq 0 \qquad (\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{V}). \tag{99}$$

The parameter flow is

$$\dot{\boldsymbol{c}}^P_{h,r} = \frac{1}{|\mathcal{X}|} \sum_{\boldsymbol{X} \in \mathcal{X}} \sum_{\ell=1}^{L} \psi^P_{h,\ell}(r; \boldsymbol{X}) \boldsymbol{x}_{\ell,r}. \tag{100}$$

Thus the derivative of a selected-versus-loser margin is a sum of terms $\psi_s \boldsymbol{x}_{\ell,s}^\top \boldsymbol{x} - \psi_j \boldsymbol{x}_{\ell,j}^\top \boldsymbol{x}'$, all nonnegative. The terminal cone is therefore forward invariant.

Suppose one margin, for embeddings $\boldsymbol{x}, \boldsymbol{x}'$, stayed bounded above by B. Restrict the context average to contexts placing $\boldsymbol{x}$ and $\boldsymbol{x}'$ in the corresponding phases of one fixed chunk. This event has probability $1/\{N(N-1)\}$. On it, $p_s \geq 1/S$, $p_j \geq p_s e^{-B}$, and $\psi_s \geq d_* p_s p_j/2$. Since all other contributions are nonnegative and $\|\boldsymbol{x}\|^2 \geq 1$, the derivative of this margin is at least

$$\frac{d_* e^{-B}}{2S^2 N(N-1)} > 0, \tag{101}$$

contradicting boundedness. Every margin diverges. □

Proof of Theorem 12. Fix δ and choose the constants so that the three dynamical lemmas apply. With probability at least $1 - \delta$, Lemma 15 gives a time τ and $\bar{\boldsymbol{c}} \in \mathcal{M}$ with aligned adjacent gaps at least $2\eta_\delta$, while $\|\boldsymbol{c}(\tau) - \bar{\boldsymbol{c}}\| \leq C_\delta \rho$.

Let $\bar{\boldsymbol{c}}(t)$ be the mean flow from $\bar{\boldsymbol{c}}(0) = \bar{\boldsymbol{c}}$. Apply Lemma 16 with $\eta = \eta_\delta$ and $M = M_0 + 2$. After enlarging C_δ to meet its large-N threshold and that of Lemma 17, all mean gaps exceed $M_0 + 2$ within a time T independent of N. Bounded softmax derivatives control the target field, while Lemma 13 controls the normalizer Hessian. Together with $\sup_{\boldsymbol{x} \in \mathcal{V}} \|\boldsymbol{x}\|^2 \leq \|\boldsymbol{\mu}\|^2 + 1$, this gives an N-independent Lipschitz constant for the parameter field. Gronwall gives

$$\|\boldsymbol{c}(\tau + t) - \bar{\boldsymbol{c}}(t)\| \leq C_\delta e^{Ct} \rho, \qquad 0 \leq t \leq T. \tag{102}$$

Shrink c_δ further so that this perturbation changes every gate-logit margin by less than one. Then $\boldsymbol{c}(\tau + T)$ lies in the terminal cone, and Lemma 17 makes every relevant margin diverge. For fixed N there are finitely many such margins, and softmax gives

$$1 - \boldsymbol{p}^P_{h,\ell}(r; t, \boldsymbol{X}) \leq (S-1) \exp\left\{ - \min_{j \neq r} \big(\langle \boldsymbol{c}^P_{h,r}(t), \boldsymbol{x}_{\ell,r} \rangle - \langle \boldsymbol{c}^P_{h,j}(t), \boldsymbol{x}_{\ell,j} \rangle \big) \right\} \tag{103}$$

for the selected phase r. The convergence is therefore uniform over contexts and chunks, proving the theorem. □

B Additional Details on the Code-Completion Examples

Here we give the details behind the code-completion example of Section 2.1: the exact token accounting, the same example on post-trained models and on DeepSeek-V3.1-Base, which has no chunked compression, and two further code-completion examples. As in Section 2, a token at zero-based position t has phase $t \bmod S$ for compression stride S.

B.1 FP8 EXAMPLE FOR DEEPSEEK-V4-FLASH-BASE

In Section 2.1, we show the code-completion example with a single docstring filler. Here we give the exact inputs needed to reproduce it, together with three further filler families that test whether the reversals depend on the filler's format or wording. Unless noted otherwise, the results are measured on DeepSeek-V4-Flash-Base using the official TileLang inference implementation with model parallelism four and FP8 expert weights.

Source example and reference continuation. The example comes from the official DeepSeek-V4 inference code, `inference/kernel.py`[1], which implements TileLang kernels and PyTorch wrappers for quantized computation. The scored prefix contains the first 88 source lines, ending immediately after **`T.Cast(compute_dtype, T.Cast(FP`**. The original source continues with **`8`**. The preceding code supports this reference: the function is documented as block-wise FP8 quantization, defines FP8 range and scale quantities, and sets **`compute_dtype`** to **`FP32`** for internal computation. At the scored location, an inner cast to **`FP8`** followed by a cast back to **`compute_dtype`** is consistent with this quantization operation. These cues motivate reference-token matching, but they do not establish that no alternative token admits a syntactically valid continuation.

Exact input construction and scoring. Using zero-based token indices, the reference token **`8`** in the unmodified source occurs at index 983. A filler of L tokenizer tokens moves this token to $t = 983 + L$. For $L = 24, \ldots, 39$, the scored token positions are 1007 through 1022. Each input is one contiguous raw prefix containing the filler followed by the 983 source-code tokens preceding **`8`**. No instruction, beginning-of-sequence token, or chat template is added, and the target token and subsequent source code are omitted. We read the logits at the final input position $t - 1$, immediately after **`FP`**.

The prefix ends with **`FP`** (token ID 20199). The reference continuation **`8`** and competing continuation **`32`** have token IDs 26 and 2111, respectively. Thus, **`FP8`** tokenizes as $[20199, 26]$, whereas **`FP32`** tokenizes as $[20199, 2111]$. We report their next-token probabilities $P(8)$ and $P(32)$, together with their difference

$$\Delta = P(8) - P(32).$$

These probabilities are selected from the full next-token distribution, so $P(8)+P(32)$ need not equal one. The reported input-shift coordinate is filler length L, distinct from the scored-token position t. The filler shifts both the source-code prefix and the scoring position, so this test does not isolate the position of an individual source token.

Filler constructions. Family A (the filler shown in Section 2.1) prepends a triple-quoted Python docstring. At a filler length of 24 tokens, its exact text is

```
"""This module contains optimized tensor kernels used by the
model inference runtime.
 = = = = = = = = =
"""
```

The second line initially contains nine space-separated **`=`** characters. Each successive variant appends one additional space-separated **`=`**, ending with 24 such characters. The total filler lengths are therefore $L = 24, \ldots, 39$, giving 16 prompts. The source-code tokens following the filler remain unchanged.

The other families are B, C, and D. As summarized in Table 1, families A, B, and C keep their natural-language sentence fixed throughout the sweep and add one punctuation token at each successive length. These families differ in whether the filler is a Python comment or docstring, in its description of the source file, and in the repeated separator. Family D instead realizes the 16 token counts using semantically similar but lexically distinct single-line comments, without a repeated-symbol padding line. Thus, A/B/C provide content-matched position shifts within each family, whereas D additionally tests robustness to natural-language rephrasing.

[1] `https://huggingface.co/deepseek-ai/DeepSeek-V4-Flash-0731/blob/0c18385a15b3b34253d6cf7132f3f4a0442d4ea9/inference/kernel.py`

Table 1: Filler-family constructions. Quoted sentences are reproduced verbatim. Repeated symbols are separated by spaces in the actual inputs.

Family	Fixed text or 24-token endpoint	Length variation
A	Triple-quoted docstring: "This module contains optimized tensor kernels used by the model inference runtime."	The next docstring line contains 9–24 `=` characters before the closing triple quotes.
B	Comment: "This module contains optimized tensor kernels used by the model inference runtime."	A second comment line contains 9–24 `=` characters.
C	Comment: "Internal model inference kernel implementations."	A second comment line contains 16–31 `*` characters, with one added at each successive length.
D	At 24 tokens: "This file defines TileLang kernels for block-wise FP8 quantization and the related low-precision operations for model inference."	One natural-language comment is used at each length, with no padding line.

Table 2: DeepSeek-V4-Flash-Base (left) and DeepSeek-V4-Flash-0731 (right) results on the FP8 example. Each entry is $\Delta = P(8) - P(32)$, computed from the reported probabilities and rounded to three decimal places. Green marks $\Delta > 0$, where the reference `8` is ranked above `32`, and red marks $\Delta < 0$. "Tokens" is the filler length, and thin rules separate groups of four lengths, matching the stride of four. DeepSeek-V4-Flash-Base receives the raw prefix with fillers of 24–39 tokens, and DeepSeek-V4-Flash-0731 receives the chat-formatted prompt of Appendix B.2 with fillers of 23–38 tokens.

DeepSeek-V4-Flash-Base (raw prefix)					DeepSeek-V4-Flash-0731 (chat prompt)				
Tokens	A	B	C	D	Tokens	A	B	C	D
24	-0.786	-0.791	-0.007	+0.060	23	-0.910	-0.775	-0.364	-0.789
25	-0.636	-0.383	-0.415	-0.484	24	+0.615	+0.451	+0.644	-0.829
26	+0.586	+0.751	+0.845	+0.614	25	-0.831	-0.854	-0.822	-0.989
27	+0.914	+0.862	+0.809	+0.681	26	-0.936	-0.939	-0.824	-0.875
28	-0.248	-0.410	-0.354	-0.337	27	-0.679	-0.668	-0.266	-0.151
29	-0.043	+0.119	-0.533	-0.764	28	+0.860	+0.073	+0.241	-0.420
30	+0.826	+0.808	+0.724	+0.696	29	-0.901	-0.238	-0.713	-0.813
31	+0.927	+0.933	+0.822	+0.737	30	-0.960	-0.954	-0.803	-0.608
32	-0.219	-0.660	-0.119	-0.129	31	-0.882	-0.159	-0.160	-0.715
33	-0.644	-0.276	-0.118	-0.624	32	+0.282	-0.384	+0.822	-0.566
34	+0.746	+0.806	+0.543	+0.524	33	-0.917	-0.828	-0.874	-0.946
35	+0.904	+0.829	+0.895	+0.653	34	-0.968	-0.940	-0.786	-0.968
36	-0.468	-0.503	+0.527	+0.249	35	-0.869	-0.438	-0.826	-0.787
37	-0.553	-0.198	-0.510	-0.532	36	+0.706	+0.597	+0.715	-0.895
38	+0.887	+0.852	+0.768	+0.591	37	-0.945	-0.824	-0.834	-0.892
39	+0.948	+0.953	+0.866	+0.790	38	-0.903	-0.790	-0.770	-0.880

Results across filler families. In Table 2, we report the probability difference Δ for all 16 lengths and all four filler families, next to the post-trained DeepSeek-V4-Flash-0731 results discussed in Appendix B.2. Positive values favor the reference continuation `8`, and negative values favor `32`. No ties occur in these measurements.

Family A exhibits an exact four-token pattern: $\Delta > 0$ at all lengths with $L \bmod 4 \in \{2, 3\}$, and $\Delta < 0$ at the remaining lengths. The full-vocabulary top-1 prediction is `8` when $\Delta > 0$ and `32` when $\Delta < 0$. Since $t = 983 + L$, filler residues $L \bmod 4 \in \{2, 3\}$ correspond to scored-token residues $t \bmod 4 \in \{1, 2\}$. Across Family A's 16 lengths, the mean combined probability $P(8) + P(32)$ is 0.982, while Δ ranges from -0.786 to $+0.948$.

At lengths satisfying $L \bmod 4 \in \{2, 3\}$, all 32 measurements across the four families favor `8`. At the remaining lengths, 28 of 32 favor `32`, and four favor `8`, with no ties. These four exceptions are Family B at $L = 29$, Family C at $L = 36$, and Family D at $L = 24$ and $L = 36$, with probability differences $+0.119$, $+0.527$, $+0.060$, and $+0.249$, respectively. Across the complete sweep, Δ ranges from -0.786 to $+0.948$ for A, from -0.791 to $+0.953$ for B, from -0.533 to $+0.895$ for C, and from -0.764 to $+0.790$ for D.

B.2 FP8 Example for Other DeepSeek Models

To test whether preference reversals persist under chat prompting, we evaluate the post-trained DeepSeek-V4-Flash-0731 and DeepSeek-V4.1-Flash models on the FP8 example in Appendix B.1. We retain the same source-code prefix and reference continuation, and give both models identical chat-formatted causal prefixes.[2] To check whether the reversals depend on chunked compression, we also evaluate DeepSeek-V3.1-Base (DeepSeek-AI, 2025), which has none.

Chat-formatted inputs. We disable thinking and place the following instruction at the start of the user message:

> Continue the function without changing its block-wise quantization semantics; avoid a redundant nested cast

The instruction is followed by the filler and then the first 974 tokens of the source prefix, with blank lines separating these components. The remaining nine source tokens, `compute_dtype, T.Cast(FP`, form the assistant prefill. This preserves the completion point in Appendix B.1 while presenting the task as a user request:

```
[BOS][USER]{instruction}\n\n{filler}\n\n{code head}
[ASSISTANT]</think>compute_dtype, T.Cast(FP
```

The two displayed lines are concatenated. `\n` denotes a newline, and the bracketed role labels stand for the template's special tokens. We score the next token after the supplied assistant prefill. For filler length L, the scored position is $t = 1004 + L$, including the chat-template tokens.

Filler families. We use families A, B, C, and D from Appendix B.1, now with $L = 23, \ldots, 38$ tokens, giving 16 variants per family (Table 3). The instruction and source code remain fixed throughout.

Table 3: Filler constructions for the post-trained models. The fixed sentences and formats follow Table 1, and each family spans filler lengths $L = 23, \ldots, 38$.

Family	Format	Length variation
A	Triple-quoted docstring	8–23 space-separated `=` characters on the second line.
B	Two-line Python comment	8–23 space-separated `=` characters on the second line.
C	Shorter two-line comment	15–30 space-separated `*` characters on the second line.
D	Single-line comment	16 natural-language rephrasings, with no repeated-symbol line.

Results. We report $\Delta = P(8) - P(32)$ in Table 2 (right) and Table 4 (left), in the same format as the base-model results. Across these runs, the full-vocabulary top-1 prediction is always either `8` or `32`, so the sign of Δ also determines whether the reference token is selected.

DeepSeek-V4-Flash-0731 exhibits a four-token cycle for A and C: it selects `8` exactly when $L \bmod 4 = 0$, giving 4/16 correct predictions (25%) in each family. B follows the same pattern except at

[2] DeepSeek-V4-Flash-0731 uses the official TileLang implementation with model parallelism four in mixed precision, with FP4 (MXFP4) routed-expert weights and FP8 for most other weights. DeepSeek-V4.1-Flash uses vLLM with tensor parallelism eight and expert parallelism. The comparison therefore includes differences in inference backend.

Table 4: DeepSeek-V4.1-Flash (left) and DeepSeek-V3.1-Base (right) results on the FP8 example. Entries, shading, and "Tokens" follow Table 2. DeepSeek-V4.1-Flash receives the chat-formatted prompt with fillers of 23–38 tokens, and thin rules separate pairs of lengths, matching its stride of two. DeepSeek-V3.1-Base receives the same raw-prefix inputs as DeepSeek-V4-Flash-Base, grouped in fours as in Table 2.

DeepSeek-V4.1-Flash (chat prompt)					DeepSeek-V3.1-Base (raw prefix)				
Tokens	A	B	C	D	Tokens	A	B	C	D
23	+0.124	+0.245	+0.358	+0.555	24	+0.316	+0.362	+0.355	+0.242
24	-0.555	-0.555	-0.462	+0.124	25	+0.477	+0.315	+0.359	-0.003
25	+0.124	+0.358	+0.462	+0.462	26	+0.600	+0.316	+0.348	+0.203
26	-0.555	-0.635	-0.124	+0.124	27	+0.274	+0.486	-0.026	+0.140
27	+0.358	+0.555	+0.462	+0.245	28	+0.430	+0.363	+0.303	+0.011
28	-0.555	-0.124	-0.635	+0.124	29	+0.303	+0.259	+0.273	+0.463
29	+0.358	+0.462	+0.124	+0.124	30	+0.240	+0.062	+0.081	+0.140
30	-0.462	-0.555	-0.245	-0.245	31	+0.392	+0.316	+0.464	+0.236
31	+0.358	+0.358	+0.358	+0.555	32	+0.346	+0.539	+0.026	-0.054
32	-0.635	-0.704	-0.462	-0.245	33	-0.065	+0.284	+0.139	+0.478
33	+0.462	+0.245	+0.462	+0.124	34	+0.139	+0.146	+0.109	+0.325
34	-0.555	-0.462	-0.245	-0.462	35	+0.398	+0.453	+0.062	+0.017
35	+0.555	+0.555	+0.358	+0.734	36	+0.166	+0.081	+0.276	+0.370
36	-0.635	-0.555	-0.245	-0.358	37	+0.388	+0.498	+0.027	+0.347
37	+0.462	+0.635	+0.462	+0.358	38	+0.397	+0.324	+0.396	+0.143
38	-0.635	-0.635	-0.245	-0.358	39	+0.409	+0.197	+0.404	+0.342

$L = 32$, where it selects **32**, reducing accuracy to 3/16 (18.75%). With D, it selects **32** at every length.

DeepSeek-V4.1-Flash instead alternates between the two continuations for A, B, and C, selecting **8** at odd filler lengths and **32** at even lengths. Each family therefore gives 8/16 correct predictions (50%). For D, the model selects **8** at all seven lengths from 23 to 29, then follows the same odd–even alternation, yielding 11/16 (68.75%). These periods describe the top-1 decisions over the measured range, and the probability differences also vary within each residue group.

The reversals can be large. Increasing the A filler from 23 to 24 tokens changes Δ from -0.910 to $+0.615$ in DeepSeek-V4-Flash-0731, while the same change moves DeepSeek-V4.1-Flash from $+0.124$ to -0.555. Thus, the sensitivity seen in the base-model example persists with post-trained models and chat-formatted inputs, although its period and dependence on filler content differ across models. The reported accuracies summarize shifted variants of this example under the selected prompt, rather than performance on a code-completion benchmark.

A model without chunked compression. DeepSeek-V3.1-Base compresses each token's keys and values into a low-rank latent vector, but it does not pool tokens into compressed entries, so its attention has no stride.[3] The two models share a tokenizer, so we give DeepSeek-V3.1-Base the raw-prefix inputs of Appendix B.1, token-for-token identical to those of DeepSeek-V4-Flash-Base. In Table 4 (right), we see no periodic reversal. DeepSeek-V3.1-Base ranks **32** above **8** at only 4 of the 64 inputs, each with $|\Delta| < 0.07$, compared with 28 for DeepSeek-V4-Flash-Base. The difference lies in the dependence on position rather than in overall confidence. In family A, the mean of $P(8)$

[3]DeepSeek-V3.1-Base (revision `d3d4eafd`) uses the official DeepSeek-V3 inference implementation with model parallelism eight and FP8 weights, because its release does not include inference code of its own. The comparison with DeepSeek-V4-Flash-Base therefore also differs in inference backend. With the implementation's BF16 output projection, 12 of the 64 comparisons between **8** and **32** were exact ties, so we compute only the final vocabulary projection in FP32 and leave the rest of the model unchanged. The BF16 outputs replicated exactly on a second machine, and the FP32 results come from a single run.

is 0.609 for DeepSeek-V3.1-Base and 0.589 for DeepSeek-V4-Flash-Base, but only DeepSeek-V4-Flash-Base changes its preference systematically with the scored position. Across the four families, it errs at 15 of 16 inputs with $t \bmod 4 = 0$ and 13 of 16 with $t \bmod 4 = 3$, whereas the four errors of DeepSeek-V3.1-Base fall on three different residues. This contrast is consistent with chunked compression being the source of the sensitivity, although the two models also differ in architecture, training, and inference backend.

B.3 Additional Code-Completion Examples

To show that the reversals are not peculiar to the FP8 example, we give two further examples in which the same kind of filler sweep produces top-1 reversals, periodic for most of the evaluated models: a shell test from the Omarchy GitHub repository and a SQL task adapted from the `gretelai/synthetic_text_to_sql` dataset. We evaluate DeepSeek-V4-Flash-Base, DeepSeek-V4-Flash-0731, and DeepSeek-V4.1-Flash.

B.3.1 Omarchy Lid-Close Test

Omarchy is a Linux distribution whose system utilities handle desktop configuration and hardware events. This example comes from `test/shell.d/lid-close-test.sh`, which tests the response to closing a laptop lid.[4] The test distinguishes an undocked laptop, which should lock before suspending, from a laptop connected to an external display, which should remain usable in clamshell mode.

The excerpt below retains the scenario setup and the cues preceding the completion point. Ellipses mark omitted code or comment text, and `[CURSOR]` denotes the next-token prediction position.

```
setup_scenario() {
  scenario_dir="$tmpdir/$1"
  # ...
  local closed="$2" docked="$3"

  cat >"$mock_bin/omarchy-hw-laptop-closed" <<SH
#!/bin/bash
exit $closed
SH
  # ...
}

# An undocked lid close is about to suspend ...
setup_scenario undocked 0 1
# ...

# A docked lid close is clamshell mode:
# logind leaves the machine awake and the session
# stays in use on the external display,
# so locking it would be wrong.
setup_scenario docked [CURSOR]
```

The first argument names the scenario. The second controls the exit status of the mocked lid-closed detector, and the third controls the external-display detector. Shell status `0` indicates that the detected condition holds. Thus, `setup_scenario undocked 0 1` represents a closed lid without an external display. The docked scenario also explicitly describes a closed lid, so the next token should again be `0`. Choosing `1` would instead simulate a lid that is not closed, contradicting the scenario being tested. The comment about avoiding screen locking specifies the expected response to this state, whereas the argument being completed specifies the lid state itself.

We vary a natural-language filler over 16 consecutive lengths, from 24 to 39 tokens under the DeepSeek-V4 tokenizer, while keeping the source prefix fixed. We report the next-token predictions in increasing filler-length order in Table 5.

[4] Full source file: `test/shell.d/lid-close-test.sh`, commit `f99d33a8`.

Table 5: Omarchy lid-close completion across 16 filler variants. The reference token is `0`, and every mismatch predicts `1`. Green marks correct predictions and red marks mismatches. Spaces group consecutive predictions for readability.

Model	Top-1 sequence	Mismatches/16	$P(0)$ range
DeepSeek-V4-Flash-Base (TileLang)	0000 0001 0001 0001	3	0.175–0.951
DeepSeek-V4-Flash-0731	1100 1100 1100 1100	8	0.270–0.725
DeepSeek-V4.1-Flash	0000 0000 1010 1010	4	0.378–0.867

DeepSeek-V4-Flash-0731 exhibits an exact four-position cycle, alternating between two incorrect and two correct predictions. DeepSeek-V4-Flash-Base makes errors at inputs 8, 12, and 16, while DeepSeek-V4.1-Flash alternates between incorrect and correct predictions over the final eight inputs.

B.3.2 SQL Station-Distance Query

This example is adapted from test record 2113 of `gretelai/synthetic_text_to_sql`, a dataset pairing natural-language questions with database contexts and reference SQL queries.[5] The original task asks for the distance between stations 5 and 12. Its context supplies a table containing the two stations' coordinates, and its reference query computes their spherical distance.

We retain the original schema, seed rows, and reference query, and adapt the example into a constrained SQL replay task in which a recorded output determines which operator is correct. We construct a contrasting query by replacing one + in the distance formula with −, and include this complete query together with its recorded output. The reference query is then presented as `q_replay`, with its recorded output marked authoritative, and truncated immediately before the same arithmetic operator. The task requires choosing + or − to reproduce that output without changing the remaining SQL.

The excerpt below shows the seed data, contrasting query, and completion point. Line breaks are adjusted for readability, ellipses mark omitted text, and `[CURSOR]` denotes the prediction position.

```
-- Source request: What is the distance between
-- station 5 and station 12?
-- At the next arithmetic slot, write exactly + or -;
-- only one reproduces q_replay.

CREATE TABLE stations (...);
INSERT INTO stations
  (station_id, name, latitude, longitude) VALUES
  (5, 'Station 5', 40.712776, -74.005974),
  (12, 'Station 12', 40.718261, -74.004790);

-- q_control captured row multiset (different label):
-- [[5626.688008276917]]
-- q_control:
SELECT 3959 * acos(
  ... - sin(radians(stations.latitude))
      * sin(radians((
          SELECT latitude FROM stations
          WHERE station_id = 12))))
  as distance FROM stations WHERE station_id = 5;

-- q_replay captured row multiset (authoritative):
-- [[0.38403957581396597]]
-- q_replay:
SELECT 3959 * acos(
  cos(radians(stations.latitude))
```

[5]Full source file: `synthetic_text_to_sql_test.snappy.parquet`, row 2112 (record 2113), revision `740ab23`.

Table 6: Station-distance completion across 16 filler variants, ordered by increasing filler length. The reference token is +, and every mismatch predicts −. Green marks correct predictions and red marks mismatches. Spaces group consecutive predictions for readability.

Model	Top-1 sequence	Mismatches/16	$P(+)$ range
DeepSeek-V4-Flash-Base (TileLang)	++++ −+++ ++++ +−++	2	0.329–0.569
DeepSeek-V4-Flash-0731	−−−+ −−+− −−+− −−++	11	0.123–0.638
DeepSeek-V4.1-Flash	+−+− +−+− +−+− +−+−	8	0.119–0.724

```
* cos(radians((
    SELECT latitude FROM stations
    WHERE station_id = 12)))
* cos(radians(stations.longitude)
      - radians((
          SELECT longitude FROM stations
          WHERE station_id = 12)))
[CURSOR]
```

The reference continuation is +. It adds the product of the latitude sine terms to the preceding cosine product, yielding a distance of approximately 0.38404 miles. Substituting − instead yields approximately 5626.68801 miles, matching the contrasting query's output rather than the authoritative replay output. Thus, both operators admit executable SQL, but only + satisfies the stated replay constraint. The complete contrasting query supplies a competing local pattern, while the output labels explicitly distinguish its result from the required one.

In Table 6, we report results for fillers of 24–39 tokens written as SQL comments. DeepSeek-V4.1-Flash alternates exactly between + and − across all 16 inputs. DeepSeek-V4-Flash-Base and DeepSeek-V4-Flash-0731 also switch between the two operators, with 2 and 11 mismatches, respectively.

C Additional Details on the Needle-in-a-Haystack Evaluations

Here we describe the needle-in-a-haystack evaluations of Section 2.2 and Section 3. We first give the construction and per-residue results for the DeepSeek models, and then the retrieval task and the two padding protocols that we use for our pretrained models.

C.1 NIAH Evaluation for the DeepSeek-V4 Family

Here we give the setup and per-residue results behind Section 2.2. As in Section 2, a token at zero-based position t has phase $t \bmod S$, and the modulo-eight residues that we use to report the results are distinct from this phase. The construction shifts the target's phase across residue groups while fixing the prompt length, the query position, and the mean and variance of the target position, so that the groups differ in phase but not in these position statistics.

Position coordinate. DeepSeek-V4's CSA compression kernel uses eight-token windows at stride four, $W = 8, S = 4$ (DeepSeek-AI, 2026a). We sample eight source-key position residues $r = t_K \bmod 8$, where t_K is the target key's zero-based token index, and call the prompts sharing a residue a residue group. For the DeepSeek-V4 models, the eight residue groups cover two cycles of the native CSA phase $t_K \bmod 4$. DeepSeek-V4.1-Flash compresses with stride two (DeepSeek-AI, 2026b), so the eight residue groups cover four phase cycles. The modulus eight defines the residue groups and does not denote a compression stride of eight.

Data generation. The construction below describes the 2,048-prompt set shared by the four DeepSeek-V4 variants. The DeepSeek-V4.1-Flash evaluation draws ten times as many prompts from the same distribution, using ten times as many bindings, where a binding is one fixed set of 16,000 key–value pairs. This gives 2,560 prompts per residue group, or 20,480 in total (Table 7). Each prompt contains a sequence of key–value records followed by a query asking for the value

associated with one target key. Each prompt contains 128,000 tokens. An intro with the instruction and three format examples is followed by a natural prefill of ℓ tokens, which sets the target's residue, and by a padding block of about 63,000 filler tokens whose length is a multiple of eight. Next come 16,000 key–value pairs, with four tokenizer tokens per pair in the form `key:value。` , and then a natural postfill of $48-\ell$ tokens, tail padding, and the query. The target is the 249th pair (zero-based index 248), so its key starts at token $63{,}976+\ell$, which is 64,000 on average. The evaluation set contains 256 prompts in each of the eight residue groups, for 2,048 prompts in total.

The key–value candidates are generated deterministically using the DeepSeek-V4 tokenizer. We select candidate strings with tokenizer IDs of at least 10,000 as a heuristic for less common tokens. Candidate strings are also alphabetic, contain at most 16 Unicode characters, and occupy exactly one tokenizer token. We additionally require tokenization stability at record and query boundaries. We construct 256 deterministic bindings, each containing 16,000 key–value pairs, and use each binding once in every residue group. The same evaluation prompts and prompt IDs are reused across the four DeepSeek-V4 variants, giving comparisons on identical prompts within that family.

Natural-language filler is drawn from the training split of the `agentlans/high-quality-english-sentences` dataset at revision `5dfcb23`. Sentences are shuffled and grouped by exact tokenizer length. The evaluation set uses 2,048 distinct sentences to form 1,024 sentence pairs. Each pair appears twice with reversed prefill/postfill roles. This balances the residue schedule and the target's phase. The filler text is not identical across all eight residue groups.

For residue r, the natural prefill length ℓ satisfies $8 \le \ell \le 40$ and $\ell \equiv r \pmod 8$. Every residue group contains 256 prompts with

$$\mathbb{E}[\ell] = 24, \qquad \mathbb{E}[\ell^2] = 604.$$

The natural postfill has length $48-\ell$, so the two natural filler segments always contribute exactly 48 tokens. The finite histograms are

$$\begin{aligned}
H_0 &= \{(8,1),(16,52),(24,150),(32,52),(40,1)\},\\
H_1 &= \{(9,2),(17,68),(25,146),(33,40)\},\\
H_2 &= \{(10,4),(18,84),(26,140),(34,28)\},\\
H_3 &= \{(11,7),(19,101),(27,129),(35,19)\},\\
H_4 &= \{(12,12),(20,116),(28,116),(36,12)\},
\end{aligned}$$

with H_{8-r}, for $r \in \{1,2,3\}$, obtained by replacing each length ℓ in H_r with $48-\ell$.

Each prompt is assembled as

$$[\text{intro; natural prefill; body prefix; pair body; natural postfill; tail; query separator; query}].$$

The body prefix is constructed so that the target-pair start is

$$s = 64{,}000 + \ell - 24,$$

which gives a mean start position of 64,000 and variance $604 - 24^2 = 28$ in every residue group while preserving the desired modulo-eight residue. The tail is then filled to reach exactly 128,000 prompt tokens. The query's token position is fixed across residue groups. The terminal query is:

```
Now answer the question using the key-value pairs
above.  What is the value for key <KEY>?  Answer with
only the value.  The value for key <KEY> is
```

Generation is followed by independent retokenization. We verify the prompt length, all component boundaries, pair-record tokenization, target span, source-key residue, and gold answer before saving the evaluation set.

Scope of the phase comparison. Changing the outer filler shifts the target and distractor records together, and each four-token record spans a whole number of compression strides for both DeepSeek-V4 and DeepSeek-V4.1. These comparisons hold the query position and the first two moments of target position fixed, but do not vary the target's phase independently of the distractors. The In-sequence Padding evaluation in Section 3.1 and Appendix C.3 tests variable gaps between records in our pretrained models.

Table 7: Answer-prefix accuracy (%) by source-key residue $r = t_K \bmod 8$ for the five DeepSeek models shown in Figure 2. n is the number of prompts in each residue group, and $\Delta_{\text{res}} = \max_r \text{Acc}_r - \min_r \text{Acc}_r$ is reported in percentage points. Accuracies are rounded to two decimal places from exact integer correct counts.

Model	n	$r=0$	$r=1$	$r=2$	$r=3$	$r=4$	$r=5$	$r=6$	$r=7$	Δ_{res}
DeepSeek-V4-Flash-Base	256	43.75	69.53	66.02	29.69	44.92	69.92	64.45	32.81	40.23
DeepSeek-V4-Flash-0731	256	80.47	88.67	98.05	96.48	81.25	90.23	99.61	96.09	19.14
DeepSeek-V4-Pro-Base	256	60.94	88.28	78.91	56.25	62.50	91.02	80.47	58.20	34.77
DeepSeek-V4-Pro-0813	256	78.13	89.84	91.41	77.34	77.34	89.45	92.19	79.69	14.84
DeepSeek-V4.1-Flash	2560	95.00	89.22	95.12	89.69	94.69	90.31	95.31	89.73	6.09

Inference protocol and metrics. For each prompt in the set shared by the four DeepSeek-V4 models, we set

$$\{\texttt{temperature} = 0,\ \texttt{max_tokens} = 8\}.$$

The task uses direct-answer prompting without chain-of-thought generation.

Let g_i be the gold value and y_i the raw completion. The primary metric is answer-prefix accuracy:

$$\text{Acc}_{\text{prefix}} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}[\text{strip}(y_i) \text{ begins with } g_i]\,,$$

where strip removes leading and trailing whitespace. This metric tolerates punctuation or parser-added text after an otherwise correct answer. It is computed from the generated answer and is distinct from full-vocabulary next-token accuracy. In Figure 2, the shaded bands are two-sided 95% Wilson intervals ($z = 1.96$) for each residue group.

Results across model variants. In Table 7, we report the per-residue results. We evaluate DeepSeek-V4-Flash-Base, DeepSeek-V4-Flash-0731, DeepSeek-V4-Pro-Base, and DeepSeek-V4-Pro-0813 on the same 2,048 prompts. Averaged over the eight residue groups, their answer-prefix accuracies are 52.6%, 91.4%, 72.1%, and 84.4%, respectively. Their best–worst residue gaps are 40.23, 19.14, 34.77, and 14.84 percentage points, respectively. Because the eight residue groups span two cycles of DeepSeek-V4's stride of four, a four-token recurrence should make residues r and $r + 4$ nearly equal. For each model, we therefore average the absolute accuracy difference between residues r and $r+4$ over $r \in \{0, 1, 2, 3\}$. This difference ranges from 1.07 to 1.95 percentage points across the four models, consistent with an approximate four-token recurrence in the accuracy pattern across residues. These measurements describe the positional pattern but do not by themselves isolate its mechanistic source. DeepSeek-V4.1-Flash, evaluated on 2,560 prompts per residue group, has a mean accuracy of 92.38% and a best–worst residue gap of 6.09 percentage points.

C.2 NIAH Evaluation for Qwen3-based Models

Here we specify the retrieval task that we use to evaluate our pretrained models in Section 3 and the two padding protocols, Prefix Padding and In-sequence Padding, that vary the phase of the target key. Within each protocol, the query position is fixed across phases, and the target's expected depth and expected distance to the query are the same for every phase.

We construct token-exact animal–name retrieval tasks. Each logical example independently permutes a curated pool of 64 animal keys and a disjoint pool of 64 human names, then pairs them to obtain a random one-to-one mapping. Each key and value is one token under the Qwen3 tokenizer. The target is the pair at zero-based index $j_* = 24$ in the 64-pair body.

Both evaluations use a task instruction, a mapping header and all 64 pairs, and a recall prefix followed by eight demonstrations and the target animal key. Prefix Padding places natural-language filler before and after the mapping body. In-sequence Padding instead uses fixed outer guards and spacers within the body, as detailed below. The demonstrations are distinct pairs drawn from the body, excluding the target pair. Body records and demonstrations use {key}{value}\n: the two fields have no intervening delimiter token, but their token strings contain leading spaces, giving

Example retrieval prompt (abridged)

```
Remember each animal's assigned human name.
[ prefix filler omitted ]
 Animal-name pairs:
 snake Alex
 beetle Carl
 fox Lucy
 ...
 bear Susan
 shark Helen
 ...
 crow Nancy
[ suffix filler omitted ]
 Recall the requested animal names:
 camel Anna
 spider Lisa
 chicken Molly
 bee Alan
 sheep Adam
 lizard Emma
 llama Ethan
 dog Eric
 bear
```

Gold next token (not part of the prompt): `Susan`, including its leading space.

Figure 6: **An animal–name retrieval example with Prefix Padding.** Natural-language fillers are abbreviated, and $\cdots$ denotes omitted mapping records. All eight demonstrations and the final query are shown. The target mapping, `bear Susan`, occurs at body index 24 and is excluded from the demonstrations.

records such as `ant Alice`. The prompt ends at the target key. The associated name is the gold next token and is not appended to the input. We show an abridged example in Figure 6.

Scoring. We score the gold name using logits at the final input token. Candidate-restricted accuracy takes the argmax over all 64 candidate names (uniform-guess chance $1/64$), while full-vocabulary accuracy takes the argmax over the entire tokenizer vocabulary. We also record the gold-token probability and negative log-likelihood.

C.3 Padding Protocols for Qwen3-based Models

Source phase and matched examples. Chunked compression gives each source token a *phase*: its alignment relative to the regularly spaced right boundaries of compression windows. For a model with compression stride S, phase is the source-token position modulo S. To compare models with different strides, we use the common sampling coordinate modulo $M = \mathrm{lcm}(4, 6, 8, 12) = 24$. For each of 1,000 logical examples, we construct all M source-key residues, giving 24,000 prompts. The 24 prompts of a logical example share the mapping, pair order, and demonstration indices and differ only in their fillers or spacers, whose offsets are sampled separately for each residue. Let ℓ_{desc} and ℓ_{hdr} be the instruction and mapping-header lengths, ℓ_{pair} the token count per pair, and L_{pre} the effective inserted-token offset before the target, which the two protocols below produce in different ways. All lengths are measured with the tokenizer, including template whitespace. We define

$$c = \ell_{\mathrm{desc}} + \ell_{\mathrm{hdr}} + j_* \ell_{\mathrm{pair}}, \qquad t_K = c + L_{\mathrm{pre}}, \qquad r = t_K \bmod M. \tag{104}$$

Requesting source-key residue r therefore requires $L_{\mathrm{pre}} \equiv r - c \pmod{M}$. The target value is at $t_K + 1$ in the concatenated grammar, but neither the value position nor the final query key defines the swept phase. Since each tested stride S divides M, the model's phase is $r \bmod S$, and phase-wise results group prompts by $r \bmod S$, with equal numbers of prompts in each group. We also keep the results for each of the 24 residues alongside these stride-specific summaries.

Sampling offsets with equal expected filler lengths. To vary phase without changing the expected insertion depth, we construct the integer-valued length variable

$$X = M + U_1 + U_2 + U_3 + F, \qquad U_i \overset{\text{iid}}{\sim} \text{Unif}\{0, \ldots, M-1\}, \quad F \sim \text{Unif}\{0, \ldots, M+3\}, \tag{105}$$

with all four draws independent. For each requested residue r, the sampler draws directly from the conditional length distribution and assigns the complementary suffix filler length:

$$L_{\text{pre}} \sim \mathcal{L}(X \mid X \equiv r - c \pmod{M}), \qquad L_{\text{post}} = 6M - L_{\text{pre}}. \tag{106}$$

For $M = 24$, both lengths lie between 24 and 120 tokens, and their sum is exactly 144 tokens in every prompt. Moreover, for every residue r,

$$\begin{aligned} \mathbb{E}[L_{\text{pre}} \mid r] = \mathbb{E}[L_{\text{post}} \mid r] = 3M = 72, \\ \text{Var}(L_{\text{pre}} \mid r) = \text{Var}(L_{\text{post}} \mid r) = \frac{M^2 + 2M + 3}{3} = 209. \end{aligned} \tag{107}$$

Conditioning preserves these moments because every term in the expansion of X or X^2 involves at most two random summands and so leaves at least one of U_1, U_2, U_3 unused, which makes the residue of the sum uniform regardless of the variables in the term. The first and second raw moments are therefore the same under every residue, and the complementary construction gives the suffix filler length the same mean and variance.

These equalities concern the sampling distribution, so the 1,000 random draws per residue need not have identical empirical means. By contrast, the total filler length is fixed for every example. Together with the fixed template and body lengths, this fixes the total prompt length and the absolute query and scoring positions across phases. The design therefore matches expected source depth and source-to-query distance while changing the source key's phase.

Prefix Padding. *Prefix Padding* places a natural-language prefix filler of the sampled length L_{pre} from Equation (106) directly before the mapping body and a natural-language suffix filler of length $L_{\text{post}} = 6M - L_{\text{pre}}$ after it. Varying L_{pre} moves the packed mapping body, including the target pair, while the complementary suffix filler keeps the query and scoring positions fixed. Because the conditional mean of L_{pre} is the same for every source phase, no phase is systematically assigned an earlier or later target.

The prefix and suffix fillers come from the training split of the high-quality-english-sentences dataset (agentlans, 2024). Each source sentence is rendered with a final `\n\n`, which is included in its token length. We build exact-length bins for every length from 24 through 120, retaining at most 64 distinct sentences per bin, and exclude sentences containing any curated key or value token ID. Sentences are used without padding or truncation. The generator first samples a length using Equation (106), then samples a sentence uniformly from that length bin, and draws the suffix filler from the complementary bin as a different sentence. Every length from 24 through 120 must have a nonempty bin, so missing bins do not change the conditional length law. For reproducibility, we use seed 20260720 for the source shuffle and seed 20260722 for mapping, demonstration, and filler generation, and each example and residue has its own seed.

In-sequence Padding. Prefix Padding moves all records together, so *In-sequence Padding* instead produces the offset inside the mapping body. It keeps the same conditional draw as a latent *reference prefix length*, denoted $\widetilde{L}_{\text{pre}}$, places a fixed 24-token guard before the mapping header and another fixed 24-token guard after the body, and defines

$$x = \widetilde{L}_{\text{pre}} - 24.$$

The 64-pair body has 63 inter-record gaps. Let $g_0, \ldots, g_{62} \in \{0, \ldots, 4\}$ be the numbers of one-token spacers inserted after the corresponding records. Each spacer is the fixed one-token Unicode character `U+FF5C` (FULLWIDTH VERTICAL LINE), not the tokenizer's special padding token. For the target at index $j_* = 24$, we sample uniformly from the feasible bounded compositions satisfying

$$\sum_{i=0}^{23} g_i = x, \qquad \sum_{i=24}^{62} g_i = 126 - x. \tag{108}$$

Thus the effective offset before the target is $24 + x = \widetilde{L}_{\mathrm{pre}}$, so the target obeys the same phase equation and phase-conditioned length law as Prefix Padding.

The total number of in-body spacers is always 126, so the body boundaries, query position, scoring position, and total sequence length are fixed across phases. In-sequence Padding thus changes phase by redistributing bounded gaps around the target rather than by moving the entire body with a variable-length prefix filler. The guards and spacers contribute $24 + 126 + 24 = 174$ tokens, compared with 144 filler tokens in Prefix Padding. Thus the query is fixed across phases within each protocol but occurs 30 tokens later in In-sequence Padding. The source-key position has the same conditional distribution in both. The two protocols also reuse the same logical mappings, pair order, demonstrations, logical example indices, and 1,000 examples per source residue, so they test the same retrieval task under two distinct ways of producing the phase-balanced offset.

Tokenization checks. Because phase is defined on exact token positions, we verify that every prompt tokenizes as constructed. For Prefix Padding, we check exact text/token round trips of each filler sentence and its additive joins with adjacent prompt components, and require the assembled prompt and the prompt followed by the gold answer to each re-encode exactly to their constructed token IDs. For In-sequence Padding, we independently retokenize every assembled prompt and verify the guard lengths, all 63 gap lengths and their sum, the two partition sums in Equation (108), the target span and phase, the fixed query and scoring positions, and the gold answer.

D Additional Details on the Architecture and Pretraining of Qwen3-based Models

Here we describe the architecture, attention memory, positional encoding, and training settings behind our pretrained models in Section 3. We report the complete sweep in Appendix F.

D.1 Backbone and Compression Kernels

Backbone and reference compression kernel. The Qwen3-0.6B backbone has 28 layers, hidden width 1,024, 16 query heads, and head dimension $d_h = 128$. We retain its feed-forward blocks, residual connections, and normalization, so that chunked KV compression is the only substantial architectural change. The reference compression kernel uses non-overlapping windows with $W = S = 8$ and one KV head per layer (a key–value head, as opposed to the 16 query heads). Its K/V branches are separate and use scalar gates: payload maps are shared across offsets ($C_i^B = C^B$), while gate maps and biases depend on the offset. The same configuration is used in every layer. The ablations vary KV-head count, pooling, parameter sharing, gate dimensionality, positional bias, window overlap, positional encoding, normalization, and local-memory policy.

Gate granularity and parameter sharing. Using the notation of Section 3.1, scalar gates have $d_g = 1$, with one gate score per token broadcast across all projected dimensions. Vector gates have $d_g = d_h$, allowing projected dimensions to prefer different window offsets. Separate K/V branches use their own payload maps, gate maps, and positional biases. Tied K/V branches share all three and produce a common compressed representation $\widehat{K}_j = \widehat{V}_j$, before the normalization and positional transformations described below. Within each branch, payload and gate maps can be shared independently across the whole window, within groups of offsets, or remain offset-specific. Sharing these maps does not require sharing positional biases across offsets. Uniform averaging fixes each gate-score coordinate at $1/W$ in a full window. The evaluated configurations are listed in Table 8.

DeepSeek-style compression kernel. Combining these options recovers DeepSeek-V4's CSA compression kernel (DeepSeek-AI, 2026a), which lets us test whether the kernel studied in Section 2 produces phase sensitivity in our backbone. It uses $W = 8$, $S = 4$, vector gates, and tied K/V branches. The preceding four-token half, with offsets $i \in \{0, 1, 2, 3\}$, uses the payload and gate maps $(\boldsymbol{C}^{(0)}, \boldsymbol{Z}^{(0)})$. The current four-token half, with offsets $i \in \{4, 5, 6, 7\}$, uses a separate pair $(\boldsymbol{C}^{(1)}, \boldsymbol{Z}^{(1)})$. Each pair is shared within its half. Positional biases remain offset-specific and are shared between the tied K/V branches. In a full window, one softmax spans all eight offsets for each gate dimension.

D.2 Windows, Attention Memory, and Positional Encoding

Each kernel above produces one compressed entry per window. To use these entries in attention, we specify when each entry is emitted, which entries a query sees alongside its local window, and at which position each entry is rotated.

Window indexing and initial boundary. Token positions start at zero, and compressed entries use one-based indices $j \geq 1$. Entry j closes at $jS - 1$ and pools offsets $0 \leq i < W$ at source positions $jS - W + i$. When $W > S$, the first entries may extend before the sequence start. These entries are emitted as soon as their stride block closes. Negative positions are masked from the softmax, which normalizes over the valid offsets only. The corresponding payload terms are zero. Thus the full-window equation in Section 3.1 applies away from this initial boundary.

Attention memory and visibility. For one query head, let $q_t \in \mathbb{R}^{d_h}$ be its query after normalization and positional encoding. Write k_u, v_u for local entries and $\hat{k}_j, \hat{v}_j$ for compressed entries after the branch-specific transformations below. With local-window width $T = 16$, the visible raw positions are $\mathcal{L}_t = \{\max(0, t - T + 1), \ldots, t\}$. Let $\mathcal{J}_t$ index the visible compressed entries and $n_t = |\mathcal{J}_t| + |\mathcal{L}_t|$. Stacking vectors as rows gives

$$\mathbf{K}_t = \begin{bmatrix} (\hat{k}_j^\top)_{j \in \mathcal{J}_t} \\ (k_u^\top)_{u \in \mathcal{L}_t} \end{bmatrix}, \qquad \mathbf{V}_t = \begin{bmatrix} (\hat{v}_j^\top)_{j \in \mathcal{J}_t} \\ (v_u^\top)_{u \in \mathcal{L}_t} \end{bmatrix} \in \mathbb{R}^{n_t \times d_h}. \tag{109}$$

One softmax normalizes scores jointly across the two memories:

$$a_t = \mathrm{Softmax}\left(\frac{\mathbf{K}_t q_t}{\sqrt{d_h}}\right) \in \mathbb{R}^{n_t}, \qquad o_t = \mathbf{V}_t^\top a_t \in \mathbb{R}^{d_h}. \tag{110}$$

Each KV head supplies the memory for its corresponding group of query heads, which retain their own queries and attention distributions.

With a fixed-width local window, compressed visibility can follow one of two policies. The first delays visibility until the entry's closure leaves the local window: $\mathcal{J}_t = \{j \geq 1 : jS - 1 \leq t - T\}$. The second admits every causally closed entry, $\mathcal{J}_t = \{j \geq 1 : jS - 1 \leq t\}$, allowing compressed and local entries to represent overlapping source tokens. Our fixed-width models use the second policy, with an exact 16-token raw window. The open-tail local memory instead admits entries as soon as they close but retains raw tokens only in the current incomplete stride block, so its raw tail is empty at the final token of each stride block.

Positional encoding and K/V tying. Let $R_p \in \mathbb{R}^{d_h \times d_h}$ denote the rotary transformation at position p. In the reference setting, queries and local keys receive per-head RMSNorm before RoPE, and compressed keys are normalized and rotated after pooling:

$$\begin{aligned} q_t &= R_t \,\mathrm{RMSNorm}_Q(W_Q h_t), \qquad k_u = R_u \,\mathrm{RMSNorm}_K(W_K h_u), \\ \hat{k}_j &= R_{(j-1)S} \,\mathrm{RMSNorm}_K(\widehat{K}_j), \end{aligned} \tag{111}$$

where $W_Q, W_K \in \mathbb{R}^{d_h \times d}$ are the query and local-key projections for the relevant heads. Under the one-based entry indexing above, position $(j-1)S$ is the start of the compressed window's newest stride block. For an even number $0 < d_r \leq d_h$ of rotary dimensions, the frequencies are constructed within that subspace:

$$\omega_\ell = \theta^{-2\ell/d_r}, \qquad \ell = 0, \ldots, d_r/2 - 1, \qquad \theta = 10^6. \tag{112}$$

The transformation R_p rotates the final d_r coordinates with angles $p\omega_\ell$, using Qwen3's split-half pairing, and leaves the first $d_h - d_r$ coordinates unchanged. The reference uses $d_r = d_h = 128$, and partial-RoPE variants use $d_r = 16$, a rotary fraction of $1/8$. NoPE uses $d_r = 0$ and $R_p = I$. No additional frequency or context-length scaling is applied. The variant without Q/K normalization replaces the RMSNorm operations with the identity.

The partial-RoPE model builds this 16-dimensional rotary table directly and applies it to the last 16 of the 128 head dimensions. It does not select the first $d_r/2$ frequencies of the full-head table, whose exponent denominator would be d_h rather than d_r.

With separate K/V branches, local values are $v_u = W_V h_u$, with $W_V \in \mathbb{R}^{d_h \times d}$, and compressed values are $\hat{v}_j = \widehat{V}_j$. Neither receives key normalization or rotation. With tied K/V branches, values instead equal the normalized, rotated keys: $v_u = k_u$ and $\hat{v}_j = \hat{k}_j$. The combined attention output is then inverse-rotated at the query position, $\tilde{o}_t = R_t^{-1} o_t$, before concatenating query-head outputs and applying the standard output projection. The inverse acts on the same dimensions as RoPE and applies to the combined local and compressed readout. With separate branches, $\tilde{o}_t = o_t$.

D.3 Training Settings

The reference 100B-token training run uses AdamW with $(\beta_1, \beta_2) = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 0.1, and gradient clipping at norm 1. Training uses BF16 and seed 42. The packed corpus is traversed once with sequence length 2,048 and a global batch of 1,024 sequences. The learning rate warms up for 2% of training to 2×10^{-3} and decays by a cosine schedule to 2×10^{-5}. The seed-replication sweep additionally uses seeds 43–45. Unless stated otherwise, results for our pretrained models use the final checkpoint at step 47,518.

E Additional Details on KV-Head Knockouts

Here we define the KV-head knockouts of Section 4.1. A knockout measures how retrieval changes when the prompt-specific output of an attention component is replaced by a fixed reference vector. We use the component's mean output over separate calibration prompts as this reference, which removes what the output carries about the particular prompt while retaining its average. The intervention is evaluated on held-out prompts, and its effects are resolved by the phase of the source key. This mean replacement builds on causal analyses of attention heads and their retrieval roles (Olsson et al., 2022; Wang et al., 2023; Wu et al., 2025). We first define which output is replaced and how its reference is fitted, then describe the prompts and metrics, the sweep over our pretrained models, and what the resulting effects can and cannot show. Finally, we adapt the intervention to DeepSeek-V4-Flash-Base, whose single shared KV head and long prompts call for a position-aligned variant.

E.1 Replaced Attention Output

Our models use grouped-query attention (GQA) (Ainslie et al., 2023). A layer ℓ has H_q query heads and H_{kv} KV heads, and each KV head h is read by a group $\mathcal{Q}(h)$ of $G = H_q/H_{kv}$ query heads, which share its keys and values but keep their own queries and attention distributions. In a compressed layer, the entries that a KV head exposes to a query combine visible compressed entries (Section 3.1) with uncompressed local entries. Knockouts change neither these entries nor the rule that decides which ones are visible.

For query head a at position t, the attention readout $\boldsymbol{r}_{\ell,a,t}$ is the attention-weighted sum of the value vectors it reads. These attention weights are distinct from the compression gate scores $\boldsymbol{\alpha}$ that form each compressed entry. Some tied-KV variants apply an inverse positional rotation to the readout before the output projection. We write $\boldsymbol{z}_{\ell,a,t}$ for the readout after this rotation, with $\boldsymbol{z} = \boldsymbol{r}$ when none is applied, so the attention output is $\boldsymbol{W}_\ell^O \operatorname{Concat}_a \boldsymbol{z}_{\ell,a,t} + \boldsymbol{b}_\ell^O$. The intervention acts on $\boldsymbol{z}$, immediately before the output projection, where each query head's contribution is still a separate vector. With T_x the final input position of prompt x, whose logits predict the answer, we write $\boldsymbol{o}$ for the stacked outputs of all query heads that read KV head h at this scoring position:

$$\boldsymbol{o}_{\ell,h}(x) = \operatorname{Concat}_{a \in \mathcal{Q}(h)} \boldsymbol{z}_{\ell,a,T_x}(x) \in \mathbb{R}^{G d_v}, \tag{113}$$

where d_v is the value dimension. This is a group of query-head outputs rather than an additional output of the KV head. With $H_q = 16$, $H_{kv} = 4$, and $d_v = 128$, one knockout replaces four query-head vectors, a 512-dimensional group output. With one KV head, it replaces all 16 query-head outputs at that layer and position, which is the whole attention readout at that site.

E.2 Calibration and Mean Replacement

An unmodified forward pass over a calibration set $\mathcal{D}_{\text{cal}} = \{x^{(n)}\}_{n=1}^{N_{\text{cal}}}$, disjoint from evaluation, records every query head's output at the scoring position. Because query heads that share a KV head

keep their own queries and attention distributions, each query head receives its own mean,

$$\boldsymbol{\mu}_{\ell,a} = \frac{1}{N_{\text{cal}}} \sum_{n=1}^{N_{\text{cal}}} \boldsymbol{z}_{\ell,a,T_{x^{(n)}}}(x^{(n)}), \qquad \bar{\boldsymbol{o}}_{\ell,h} = \text{Concat}_{a \in \mathcal{Q}(h)}\, \boldsymbol{\mu}_{\ell,a}, \tag{114}$$

so a knockout retains each head's own average output, which a mean pooled over the group would not. The calibration set is balanced over source-key residues, so no phase dominates the mean, and each mean also averages over mappings and filler texts. The reference is therefore independent of the evaluation prompt and its phase, and a phase-dependent knockout effect cannot reflect a phase-specific reference. It remains specific to the model, layer, query head, and calibration distribution. A phase-conditioned mean, or replacement by zero, would define a different intervention.

To knock out KV head h_* in layer ℓ_*, we replace the outputs of every query head that reads it at the scoring position (Figure 7):

$$\widetilde{\boldsymbol{z}}_{\ell_*,a,t}(x) = \begin{cases} \boldsymbol{\mu}_{\ell_*,a}, & a \in \mathcal{Q}(h_*),\ t = T_x, \\ \boldsymbol{z}_{\ell_*,a,t}(x), & \text{otherwise.} \end{cases} \tag{115}$$

Other heads in the layer and all earlier computation are left unchanged, and later computation at the scoring position is rerun. Because the patch sits at the final input position of a causal decoder, it cannot change earlier context representations, and neither the model parameters nor the context cache are edited.

E.3 Evaluation Prompts, Metrics, and Uncertainty

Knockouts in our pretrained models use the retrieval task of Appendix C.2 under both padding protocols of Appendix C.3. Each protocol has its own calibration set, built with that protocol's prompts. Each prompt is indexed by its source-key residue $r = t_K \bmod 24$, where t_K is the source key's zero-based position, and a model with stride S sees phase $r \bmod S$. The 24 phase replicas of a logical example stay together when calibration is separated from evaluation. Logical examples $[1000, 2000)$ form the calibration set and $[0, 1000)$ the evaluation set, each with 24,000 prompts.

A prompt counts as correct when the gold name has the largest final logit, either among the 64 candidate names (candidate-restricted accuracy) or over the whole vocabulary (full-vocabulary accuracy). We also record the gold logit, the gold negative log-likelihood, and the total probability of the candidate set. Let $\mathcal{D}_r$ be the evaluation prompts at residue r and $Y^c(x) \in \{0, 1\}$ the correctness of prompt x in the clean run ($c = 0$) or under the knockout ($c = \text{KO}(\ell, h)$). The paired effect, in percentage points, is

$$\Delta_{\ell,h}(r) = \frac{100}{|\mathcal{D}_r|} \sum_{x \in \mathcal{D}_r} \big(Y^{\text{KO}(\ell,h)}(x) - Y^0(x)\big), \tag{116}$$

which is negative when the knockout impairs retrieval. Both terms use the same prompts. For stride S, folded effects average the residues that share a phase, for example residues r, $r + 8$, and $r + 16$ for W8/S8. Full-attention baselines have no phase and are grouped by the same residues only for comparison.

A phase with low clean accuracy has less room for a further decrease. In Figure 4 and the figures of Appendix F, we therefore plot the relative accuracy change $100(A^{\text{KO}}(r) - A^0(r))/A^0(r)$, which scales each loss by the clean accuracy available to lose. Here $A^c(r)$ is the accuracy at residue r. This ratio is undefined when clean accuracy is zero, so the figures apply a validity mask.

The independent sampling unit is the logical example, not each of its phase replicas. For an overall effect, we average paired differences within each logical example and take the standard error across examples, and we cluster by logical example in the same way for folded or between-phase contrasts. These clustered standard errors are computed but not plotted, and the heatmaps show point estimates that do not establish significance for every cell. The retrieval curves in the main text show two-sided 95% Wilson intervals, and the clean curves in Appendix F show pointwise 95% cluster-bootstrap intervals over logical examples. Selecting the largest loss from a sweep is exploratory, and we do not test gate-nominated heads on held-out prompts.

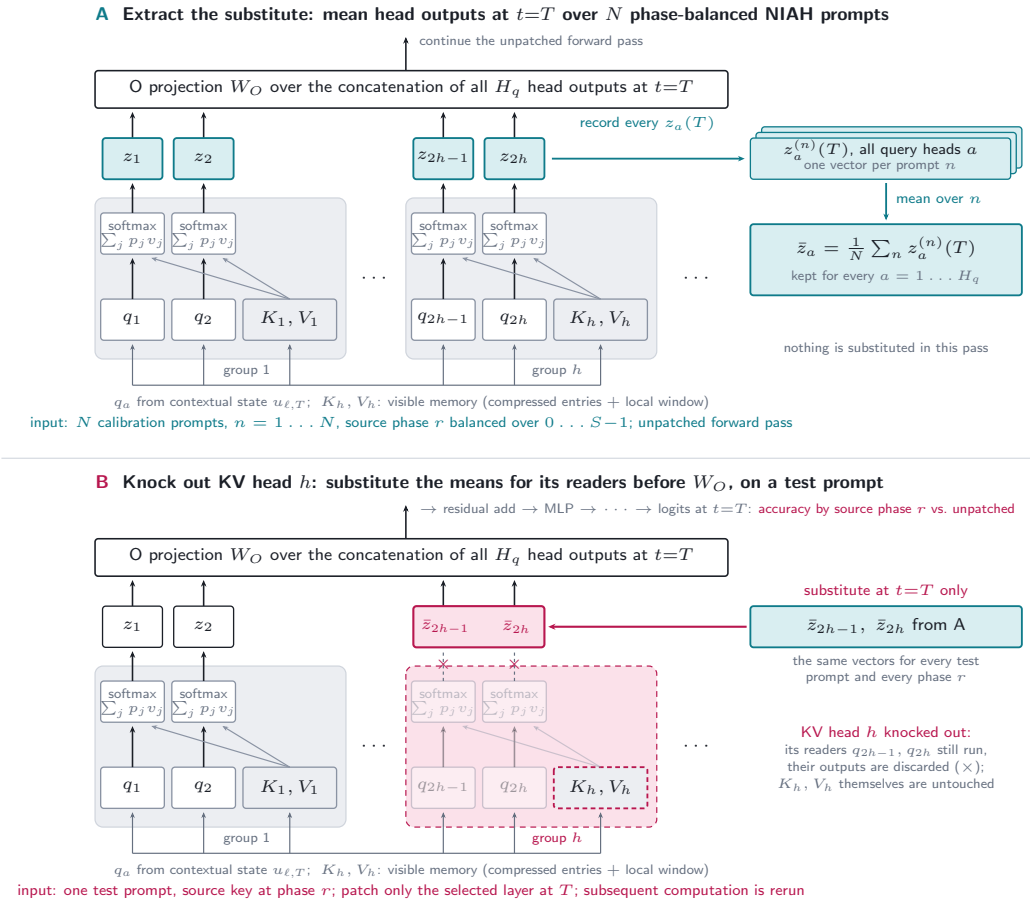

Figure 7: **Calibration and mean replacement in a GQA block.** (A) Unmodified calibration prompts supply a separate mean for every query-head output at the scoring position. (B) On each evaluation prompt, the query heads that read the selected KV head receive those fixed means before the output projection, and the KV head's keys and values are left intact. The schematic shows two query heads per KV head ($G = 2$) and suppresses the layer index and the inverse positional rotation. Each query head's output z_a corresponds to $\boldsymbol{z}_{\ell,a,T_x}$ and its mean $\bar{z}_a$ to $\boldsymbol{\mu}_{\ell,a}$. Concatenating the selected query-head outputs gives $\boldsymbol{o}_{\ell,h}$ in Equation (113). The patch is confined to the selected layer and position, and downstream representations are recomputed.

E.4 Knockout Sweeps for Qwen3-based Models

To check whether phase-dependent knockout effects extend beyond the model of Figure 4b, we sweep the 23 compressed models listed in Table 8, which have one, two, four, or eight KV heads, at checkpoint 47,518. Each model is evaluated over all 28 layers with the disjoint 24,000-prompt calibration and evaluation sets above. In the six figure groups of Appendix F, we compare Prefix Padding and In-sequence Padding and report unfolded relative full-vocabulary accuracy changes with group-wise zero-centered scales. These are point estimates rather than independently confirmed selections.

E.5 Scope of the Causal Interpretation

The experiment changes an internal representation while holding the evaluation prompt and model parameters fixed. A loss of accuracy therefore measures a causal effect of this intervention on the tested retrieval behavior. It supports the importance of the selected readout under mean replacement, rather than exclusive storage of the target in that KV head. Other KV heads may contain redundant information, and a head can contribute to query processing or downstream computation in addition

to carrying an answer. Conversely, a small knockout effect does not establish that a component is unused: other components can compensate. The change in the attention output is linear in $\bar{\boldsymbol{o}}_{\ell_*,h_*} - \boldsymbol{o}_{\ell_*,h_*}(x)$, but its effect on the final logits generally is not, so the effects of knocking out different heads separately need not add up to the effect of removing them together.

In compressed layers, one attention softmax covers compressed and local entries, and mean replacement acts on their combined readout. It does not selectively ablate compressed memory, remove one source association, or change a compression gate. Mean replacement preserves a reference first moment but does not guarantee an in-distribution activation or a matched norm. It removes within-head variation and the selected outputs' prompt-specific correlations with the rest of the network, and the calibrated mean can itself retain common task and template information. Consequently, the intervention is neither a literal deletion of a memory nor a proof of a unique retrieval circuit. We also do not report an identity-intervention comparison or a comparison that replaces the outputs of an equal number of other query heads in the same layer.

E.6 Knockouts for DeepSeek-V4-Flash-Base

The DeepSeek-V4-Flash-Base knockout follows the same mean-replacement logic but differs in what it replaces and at which positions. The model has one shared KV head per layer, so, as in our pretrained models with one KV head, a knockout replaces the whole attention readout of a layer. We take this readout after the attention output projection and before the hyper-connection, so the patched vector is the entire 4,096-dimensional layer output rather than a query-head slice of the pre-projection tensor in Equation (113). The patch can also cover several positions of the query suffix rather than only the final one.

Let $\boldsymbol{a}_{\ell,s}(x) \in \mathbb{R}^{4096}$ denote that output at suffix index $s \in \{0, \dots, 15\}$. Calibration produces a separate mean for each layer and each *suffix position*:

$$\boldsymbol{\nu}_{\ell,s} = \frac{1}{N_{\text{cal}}} \sum_{n=1}^{N_{\text{cal}}} \boldsymbol{a}_{\ell,s}(x^{(n)}), \qquad \widetilde{\boldsymbol{a}}_{\ell,s}(x) = \begin{cases} \boldsymbol{\nu}_{\ell,s}, & s \in \mathcal{P}, \\ \boldsymbol{a}_{\ell,s}(x), & s \notin \mathcal{P}. \end{cases} \tag{117}$$

The mask $\mathcal{P}$ selects one of four conditions: the first occurrence of the query key, its second occurrence, both occurrences, or the complete 16-token suffix. In Figure 4c, we show the full-suffix condition, which patches all 16 suffix positions. Suffix positions play different roles in the query, such as the two occurrences of the query key, so each receives its own mean rather than an average over the suffix, just as each query head receives its own mean in our pretrained models. The same position-aligned means are used for every evaluation prompt and source phase. Patching earlier suffix positions can also affect later suffix computation, so the full-suffix condition has a broader scope than the final-position-only intervention in our pretrained models.

For Figure 4c, we use 256 evaluation prompts per residue group, each 128,000 tokens long with 16,000 key–value pairs. Calibration and evaluation use disjoint prompts with distinct texts. The source-key coordinate is modulo eight, so these residue groups span two cycles of the CSA stride of four. The 127,984-token prefix is prefilled in 1,024-token chunks, and the remaining 16 tokens form the query suffix.

We score the designated gold target token without autoregressive answer generation. This target-token accuracy is distinct from the answer-prefix accuracy used in the behavioral benchmark of Section 2. Because DeepSeek-V4-Flash-Base's outputs vary with batch size, each clean run uses the same batch composition as the intervened run it is compared with.

In the full-suffix condition of Figure 4c, the largest phase-averaged target-token accuracy losses occur at layers 22 and 24 (zero-based). Effects vary across source residues. They establish phase-dependent reliance on the patched layer readout at this intervention granularity, but they do not identify a particular query head, a compressed-only branch, or the compression operation that wrote the relevant information.

F Complete Results for Qwen3-based Models

In Section 3.2, we plot a subset of our pretrained models. Here we give fuller results, so that each design change can be compared with the reference and the full-attention baselines. In Table 8, we

summarize the configurations shown in Figure 3 and the grouped figures below, reporting validation loss alongside mean and worst-residue retrieval accuracy and the gap across the 24 evaluated source-key residues under Prefix Padding. We include 23 compressed runs and three full-attention baselines.

These summary numbers cannot show where the weak spots fall or which layers support retrieval at each phase. In the following figures, we therefore show our pretrained models at the end of training (step 47,518) in six groups that follow the blocks of compressed runs in Table 8. In each figure, we pair clean full-vocabulary retrieval across the 24 source-key residues with layer-by-phase relative accuracy changes under KV-head knockouts (Appendix E). Across the six groups, no tested change removes the periodic weak spots, although some narrow them substantially. Uniform averaging, for example, reduces the best-to-worst gap from 75.3 to 19.2 percentage points. Random seeds, positional encoding, and normalization reshape the accuracy pattern across phases, and the weak spots repeat with the compression stride.

We show every group under both padding protocols of Appendix C.3. Prefix Padding shifts all records together, so the distractors change phase along with the target. In-sequence Padding instead varies the gaps between records, so a pattern that appears under both protocols does not require all records to shift together. Rows 1 and 2 show clean accuracy and knockout effects under Prefix Padding, and rows 3 and 4 show the same under In-sequence Padding. Curves use 1,000 logical examples and show pointwise 95% cluster-bootstrap intervals. Heatmaps within each group share a zero-centered scale.

Table 8: **Summary of the plotted pretrained models.** Prefix Padding results for the 26 runs shown in the cited main-text and appendix figures: 23 compressed runs and three full-attention baselines, evaluated at the end of training (step 47,518). For full-vocabulary retrieval accuracy A_ρ at source-key residue $\rho = t_K \bmod 24$, we report the mean over all 24 residues, the minimum, and the gap $\max_\rho A_\rho - \min_\rho A_\rho$ in percentage points. The same coordinate is used for full-attention baselines, which have no compression phase. Validation loss is reported separately from retrieval.

				Retrieval accuracy		
Configuration / change	W/S	KV	Val. loss	Mean (%)	Min. (%)	Gap (pp)
Full-attention baselines (Figure 3)						
Full attention	–	1	2.466	18.8	17.2	3.1
Full attention	–	4	2.448	47.1	44.8	6.1
Full attention	–	8	2.437	61.1	58.4	4.3
Reference and random-seed replication (Figure 8)						
Reference, seed 42	8/8	1	2.478	50.6	5.7	75.3
Reference, seed 43	8/8	1	2.482	32.1	2.5	59.2
Reference, seed 44	8/8	1	2.477	56.1	11.6	70.3
Reference, seed 45	8/8	1	2.478	40.5	15.6	36.6
Native KV-head count (Figure 9)						
Two KV heads	8/8	2	2.465	58.3	9.8	76.8
Four KV heads	8/8	4	2.448	68.7	13.6	78.0
Eight KV heads	8/8	8	2.431	59.4	9.9	71.1
Compression-window size and stride (Figure 10)						
Non-overlapping windows	4/4	1	2.469	38.5	8.0	50.5
Overlapping windows	8/4	1	2.467	66.9	46.2	39.8
Overlapping windows	8/6	1	2.470	53.4	38.5	31.9
Larger windows, one KV head	12/12	1	2.487	33.0	5.0	63.2
Compression kernels (Figure 11)						
Tied K/V branches	8/8	1	2.495	21.8	3.6	39.1
Uniform averaging	8/8	1	2.497	12.5	4.7	19.2
Uniform averaging, tied K/V, postnorm	8/8	1	2.503	18.6	1.9	39.3
Vector gates	8/8	1	2.478	19.4	4.0	34.0
Positional encoding and normalization (Figure 12)						
No Q/K normalization	8/8	1	2.485	38.5	23.8	34.6
No positional gate biases	8/8	1	2.479	34.3	4.4	60.7
No RoPE (NoPE)	8/8	1	2.504	29.4	5.0	40.9
RoPE on 16 dimensions	8/8	1	2.482	74.2	32.9	57.2
Local-memory policy and DeepSeek-style kernels (Figure 13)						
Uncompressed open tail	8/8	1	2.520	75.0	57.4	33.9
Uncompressed open tail	8/8	8	2.460	53.0	25.5	49.3
DeepSeek-style, vector gates	8/4	1	2.476	35.6	18.7	38.3
DeepSeek-style, scalar gates	8/4	1	2.473	21.5	14.2	18.7

Notes. KV denotes the number of KV heads per layer. The reference uses scalar gates, separate K/V branches, payload maps shared across offsets, and offset-specific gate maps and positional biases (Section 3.1 and Appendix D). Postnorm denotes post-projection normalization. Each run appears once, and the seed-42 reference supplies the one-KV-head model for the head-count comparison.

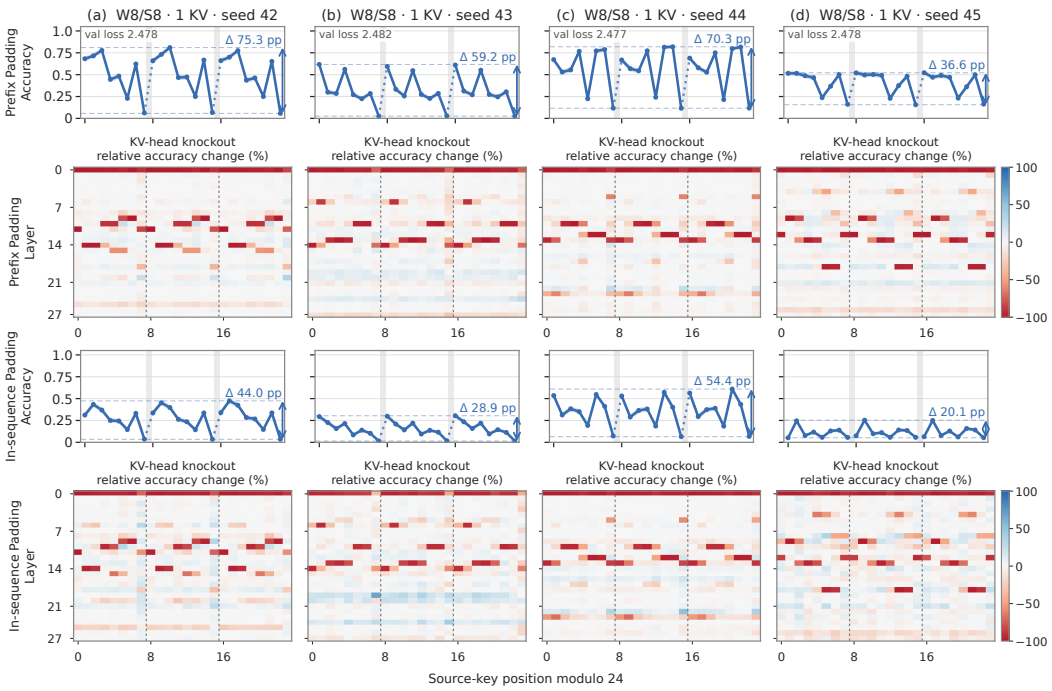

Figure 8: **Random-seed replication.** Rows 1–2 use Prefix Padding and rows 3–4 use In-sequence Padding. Rows 1 and 3 show full-vocabulary accuracy, and rows 2 and 4 show relative accuracy change under KV-head knockout. Seed changes alter the accuracy pattern across phases while preserving periodic weak spots.

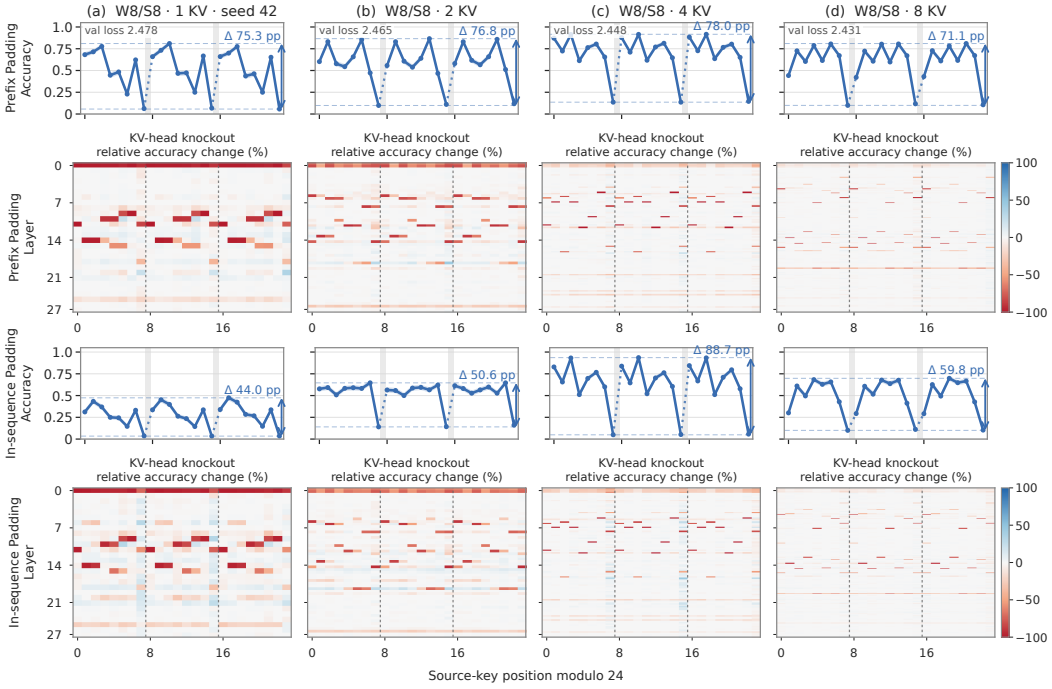

Figure 9: **Number of KV heads.** Rows 1–2 use Prefix Padding and rows 3–4 use In-sequence Padding. Rows 1 and 3 show full-vocabulary accuracy, and rows 2 and 4 show relative accuracy change under KV-head knockout. Phase sensitivity persists from one to eight KV heads.

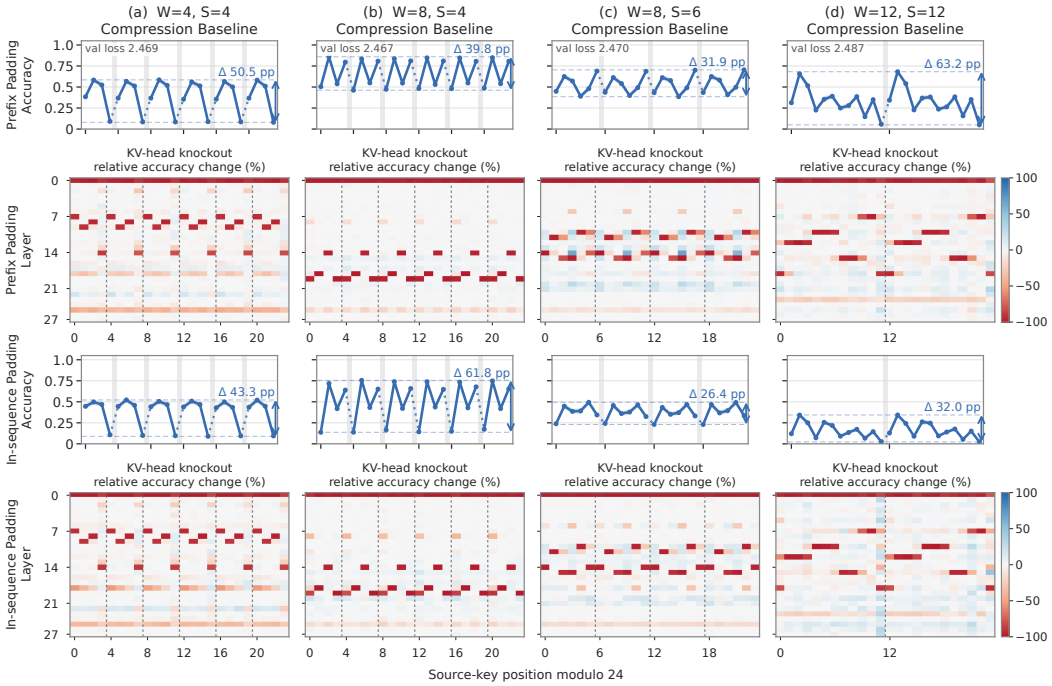

Figure 10: **Compression geometry.** Rows 1–2 use Prefix Padding and rows 3–4 use In-sequence Padding. Rows 1 and 3 show full-vocabulary accuracy, and rows 2 and 4 show relative accuracy change under KV-head knockout. Weak spots repeat with the compression stride.

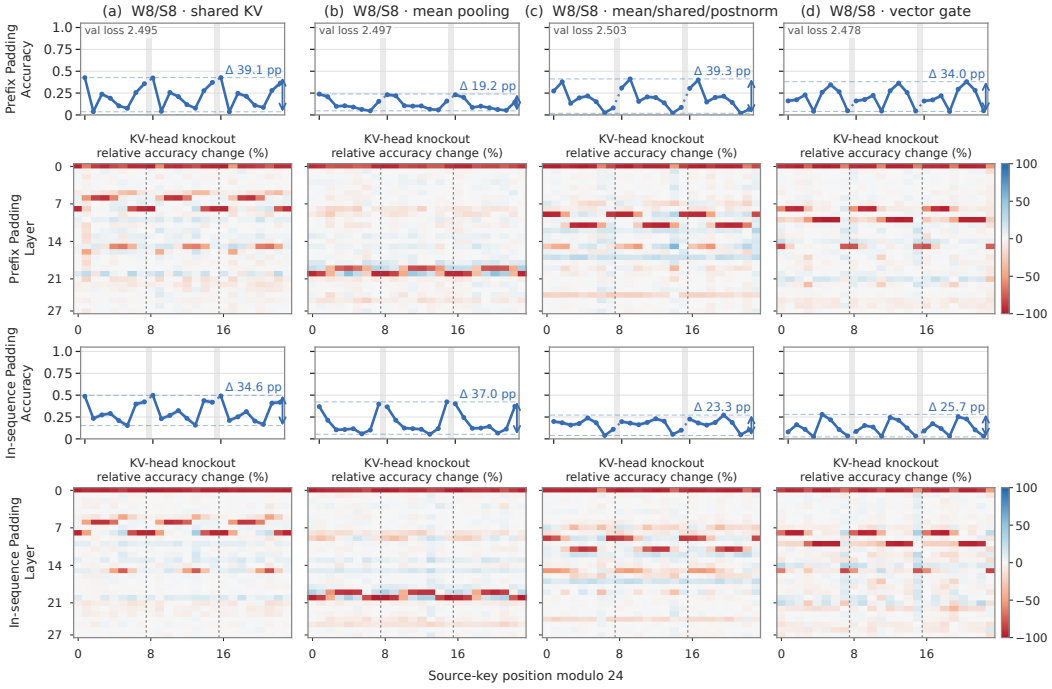

Figure 11: **Compression kernels.** Rows 1–2 use Prefix Padding and rows 3–4 use In-sequence Padding. Rows 1 and 3 show full-vocabulary accuracy, and rows 2 and 4 show relative accuracy change under KV-head knockout. Phase sensitivity persists across pooling, sharing, and gate variants.

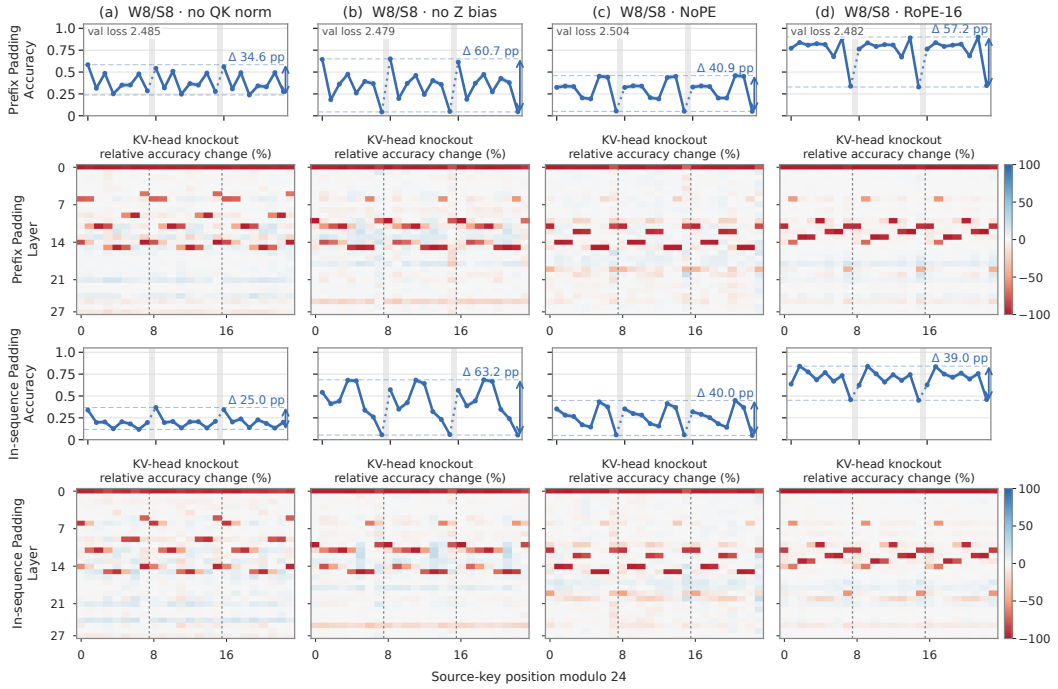

Figure 12: **Position and normalization.** Rows 1–2 use Prefix Padding and rows 3–4 use In-sequence Padding. Rows 1 and 3 show full-vocabulary accuracy, and rows 2 and 4 show relative accuracy change under KV-head knockout. Position and normalization choices reshape but do not remove phase sensitivity.

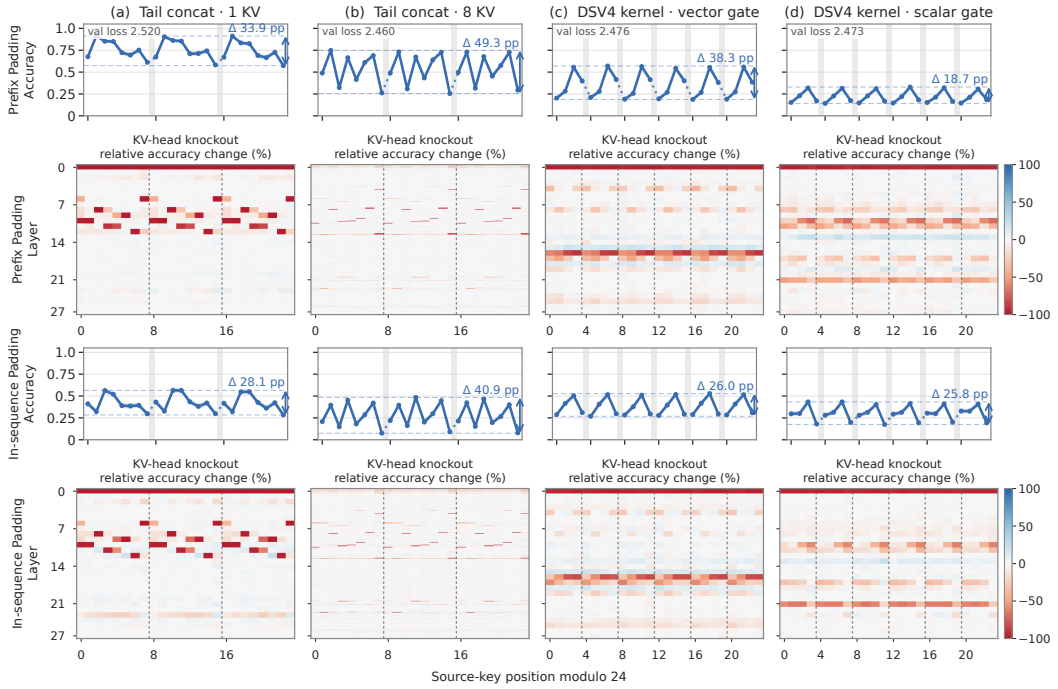

Figure 13: **Memory policy and implementation variants.** Rows 1–2 use Prefix Padding and rows 3–4 use In-sequence Padding. Rows 1 and 3 show full-vocabulary accuracy, and rows 2 and 4 show relative accuracy change under KV-head knockout. Open-tail local memory and DeepSeek-style kernels remain phase-sensitive.

G GATE PREFERENCES AND KNOCKOUT EFFECTS ACROSS MODELS

In Section 4.2, we link gate preferences to knockout effects for three heads of one model. Here we show the same comparison in more models: every head of the four W8/S8 seed replications, and every head of five other configurations. Sharp gate preferences with knockout damage at the favored phases recur across seeds and configurations, but not in every head, and the phase-folded gate curves show little preference when windows overlap.

G.1 MEASURING GATE PREFERENCES

In Section 4.2, we summarize each gate by two effective numbers of offsets (Figure 5a). For one window j of input x, collect the gate scores of its W offsets into the probability vector $\boldsymbol{\alpha}^B_{\ell,h,j}(x) \in \mathbb{R}^W$, and let $H(p) = -\sum_i p_i \log p_i$. The vertical axis is the mean per-window effective number of offsets, $\mathbb{E}_{x,j}[\exp(H(\boldsymbol{\alpha}^B_{\ell,h,j}(x)))]$, averaged over all compression windows of a held-out validation set of natural-language pretraining text. The horizontal axis is the effective number of offsets of the mean gate, $\exp(H(\mathbb{E}_{x,j}[\boldsymbol{\alpha}^B_{\ell,h,j}(x)]))$. Both range from 1 to W, and smaller values mean greater concentration. In Section 4.2, we call the horizontal quantity the gate's *static concentration*. The vertical quantity also includes *input-dependent concentration*, which places a gate below the diagonal. A small per-window value alone does not imply a stable preference. A gate that selects one offset in each window but visits every offset equally often across inputs has a uniform mean gate and lies near $(W, 1)$. For vector gates, averaging over channels can conceal preferences that differ across channels, and our channel-averaged curves (Figure 18b) cannot reveal them.

In Figure 14, we place the key gate of every KV head in this plane at the end of training, for every run in Table 8 with learned scalar gates and for the W12/S12 model with four KV heads used in Appendix H. With non-overlapping windows, a few heads per model lie along the diagonal toward the lower left, while most gates end near the right edge with varying input-dependent concentration, as in Figure 5a. With more KV heads per layer, the most concentrated heads reach further toward the lower left. With overlapping windows at stride four, including the DeepSeek-style kernel, many gates sit well left of the right edge but below the diagonal, so they favor some offsets across windows while also varying with the input.

In the figures of this appendix, we overlay each head's gate preferences on its knockout effects. Each panel shows one KV head, with the phase r of the source key on the horizontal axis. The background shows the head's knockout effect at each phase (Section 4.1), with red indicating impairment. The black curve, $g^K(r)$, is the head's mean gate score on key tokens at phase r, scaled so that a uniform gate equals one (dashed line). The purple curve, $g^V(r)$, shows the value gate read one phase later. Retrieval needs the key, to locate the entry, and the value one token later, which is the answer. For a key at phase r, we therefore plot the value gate's score at phase $(r+1) \bmod S$, so that both curves and the knockout background are indexed by the key's phase. A head that retains a key and its value shows both curves peaking at the same r, and if the head matters for retrieval, its knockout damage should peak there as well. When $W = S$, the value of a key at the last phase $S-1$ falls in the next window, and hollow squares mark these points.

G.2 VARIATION ACROSS RANDOM SEEDS

The four seed replications develop similarly sharp gate preferences, but in different places (Figure 16). Each seed has four to six heads with concentrated mean key gates (entropy below 90% of a uniform gate's), whose key-gate peaks reach 2.0 to 4.0 times the uniform level and whose knockout damage falls mainly at the favored phases. These heads lie between layers 4 and 18, but no layer has such a head in all four seeds, and their preferred key phases cover all eight phases. Among these heads, the only recurring feature is a head whose key gate favors phase 0 and whose value gate favors the value token one phase later, found at layers 11, 14, 13, and 12 in seeds 42 to 45, respectively. This variation is consistent with Section 5.2, where in an idealized model each head settles on a sharp selection of positions that depends on initialization, and heads need not cover complementary positions. Outside these heads, knocking out layer 0 impairs retrieval at every phase in all seeds, despite its near-uniform gate.

G.3 Other Configurations and Overlapping Windows

Sharp preferences also arise without rotary position embeddings. Without them, heads in layers 11, 12, 14, and 15 still favor particular phases, with knockout damage at those phases (Figure 15a). We show the same comparison with larger windows (W12/S12, one KV head) in Figure 15b and with four KV heads per layer (W8/S8) in Figure 17. With four KV heads, the preferences become sharper than with one. Many heads spike on a single offset, and their knockout losses concentrate on a single phase, as for heads 2 of layers 5 and 16 and heads 0 of layers 10 and 12. Together, these sharp heads favor all eight phases. Overlapping windows are the exception. With overlapping eight-token windows at stride four, including the DeepSeek-style kernel, most phase-folded gate curves stay close to uniform (Figure 18), leaving little phase structure for knockout damage to follow. Each phase sums two offsets, however, and many gates in these models do favor some offsets across windows (Figure 14), a preference that folding into phases hides. Vector-gate scores are also averaged over channels, which can hide preferences among channels. The gate account of Section 4.2 therefore applies most directly to non-overlapping windows.

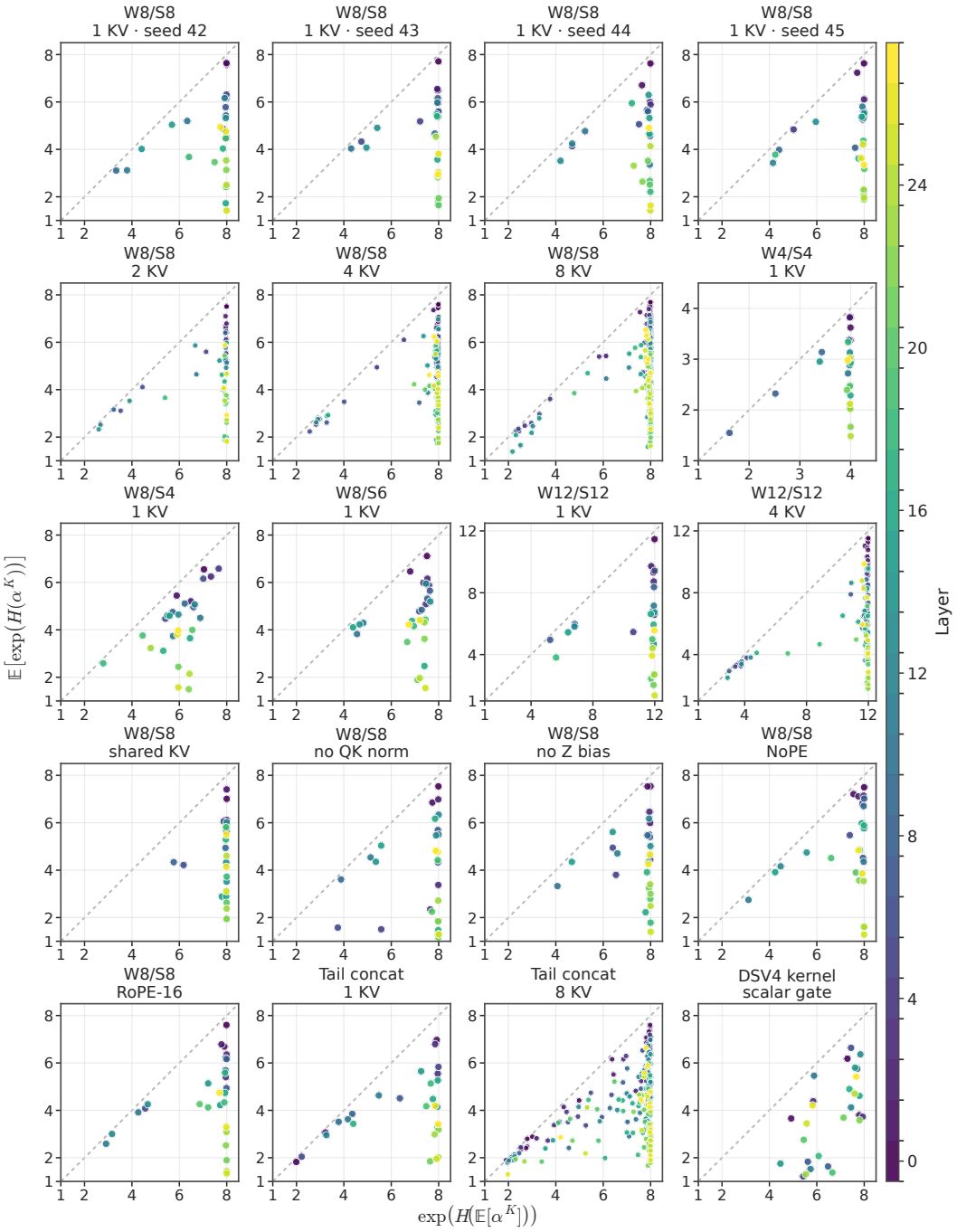

Figure 14: **Final key-gate concentration across configurations** (step 47,518). Each point is one KV head's key gate, placed by its static effective number of offsets $\exp(H(\mathbb{E}[\boldsymbol{\alpha}^K]))$ (horizontal) and its mean per-window effective number $\mathbb{E}[\exp(H(\boldsymbol{\alpha}^K))]$ (vertical), and colored by layer. The dashed line marks equality, which holds for a gate with the same scores in every window, and lower values mean more concentration. Panels cover every run in Table 8 with learned scalar gates and the W12/S12 model with four KV heads of Appendix H. We omit uniform averaging, which has no learned gates, and the vector-gate runs. Both quantities are defined in Appendix G.1.

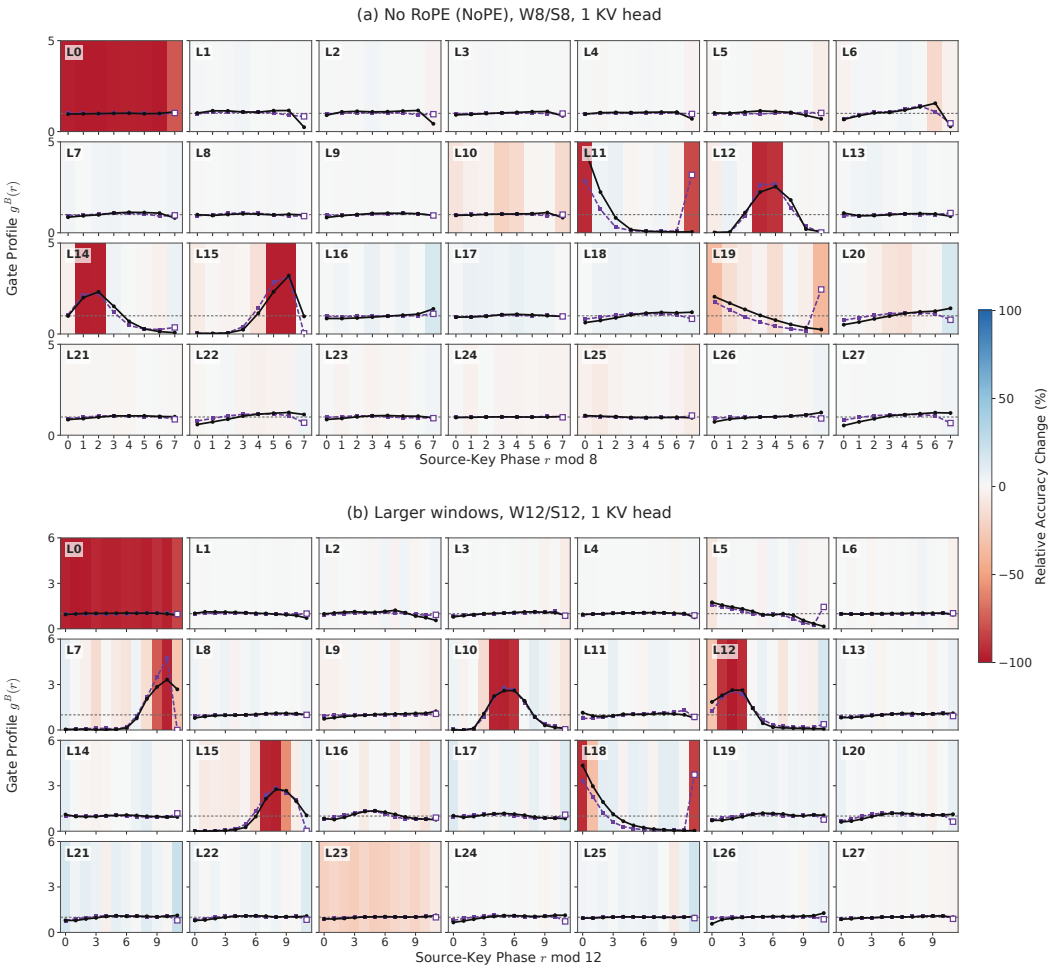

Figure 15: **Gate preferences and knockout effects without positional embeddings and with larger windows.** (a) No rotary position embeddings (NoPE, W8/S8, one KV head). (b) Larger windows (W12/S12, one KV head). Without RoPE, heads still develop sharp phase preferences, with knockout damage at the favored phases (layers 11, 12, 14, and 15). Both checkpoints are final (step 47,518) and use Prefix Padding. Vertical scales are shared within each model. Encoding and gate inputs follow Figure 16.

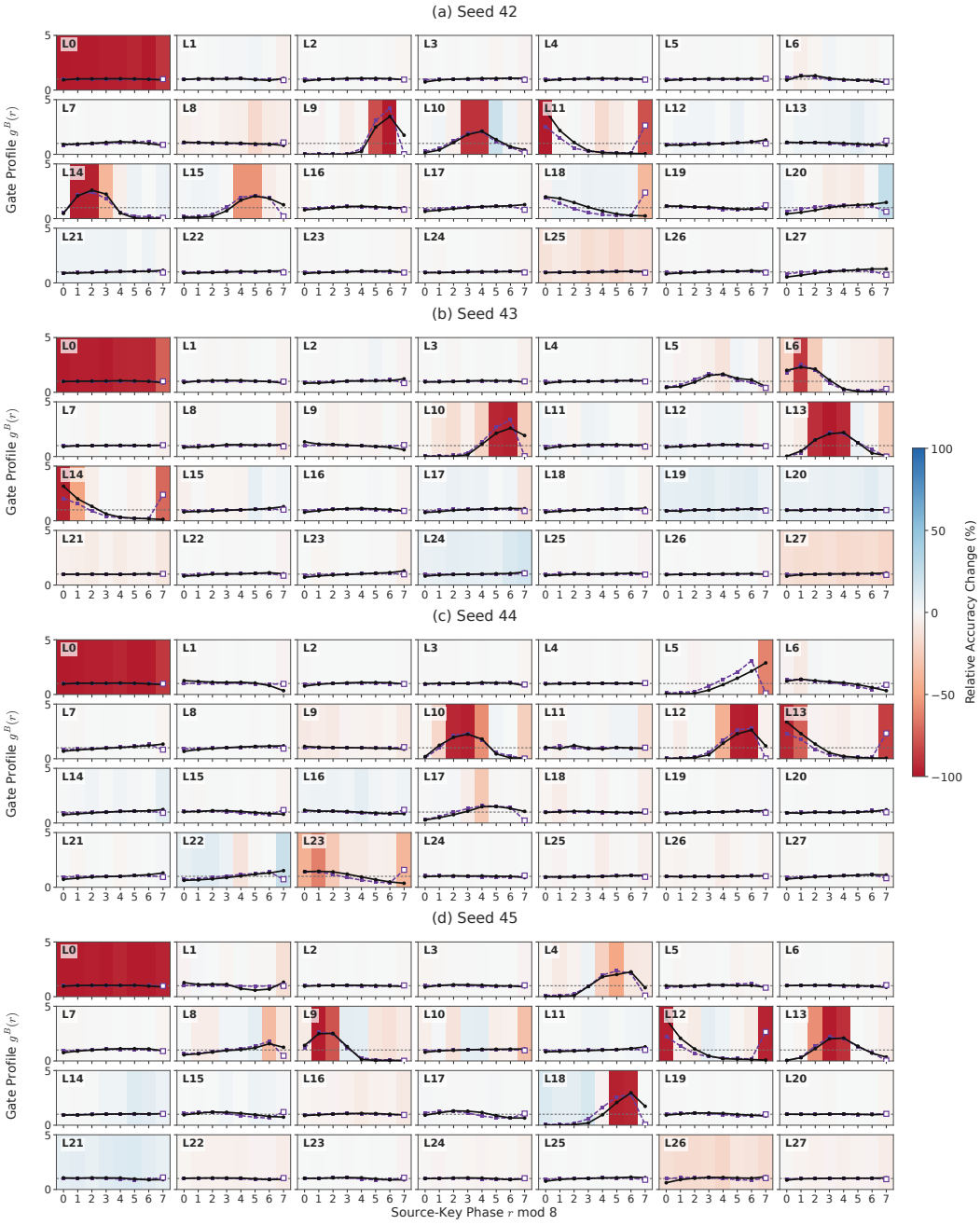

Figure 16: **Gate preferences and knockout effects across random seeds** (W8/S8, one KV head, seeds 42–45, step 47,518). Each panel shows one layer's KV head, labeled Lℓ. Black circles show the key gate g^K, purple squares the value gate g^V read one phase later, and the dashed line a uniform gate. Hollow squares mark values that fall in the next window. The background shows the relative accuracy change under the head's knockout, saturating at $\pm 100\%$, with red indicating impairment (Prefix Padding, 1,000 logical examples per phase, folded modulo S). Gate curves average 2,048 held-out natural-language sequences of 2,048 tokens. All seeds share one vertical scale.

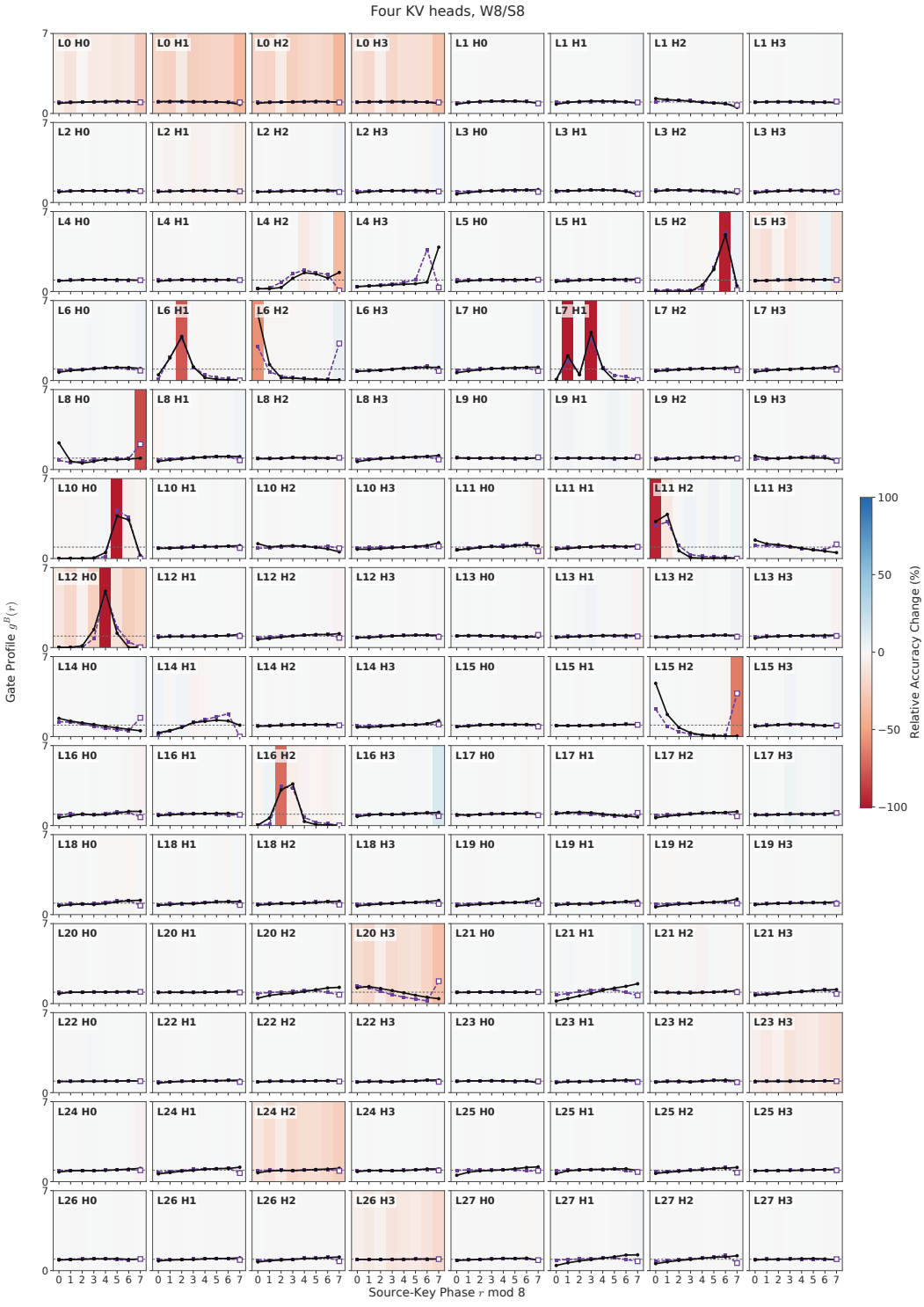

Figure 17: **Gate preferences and knockout effects with four KV heads per layer** (W8/S8, step 47,518, Prefix Padding). Panels are labeled Lℓ Hh for layer ℓ and KV head h. Encoding and gate inputs follow Figure 16.

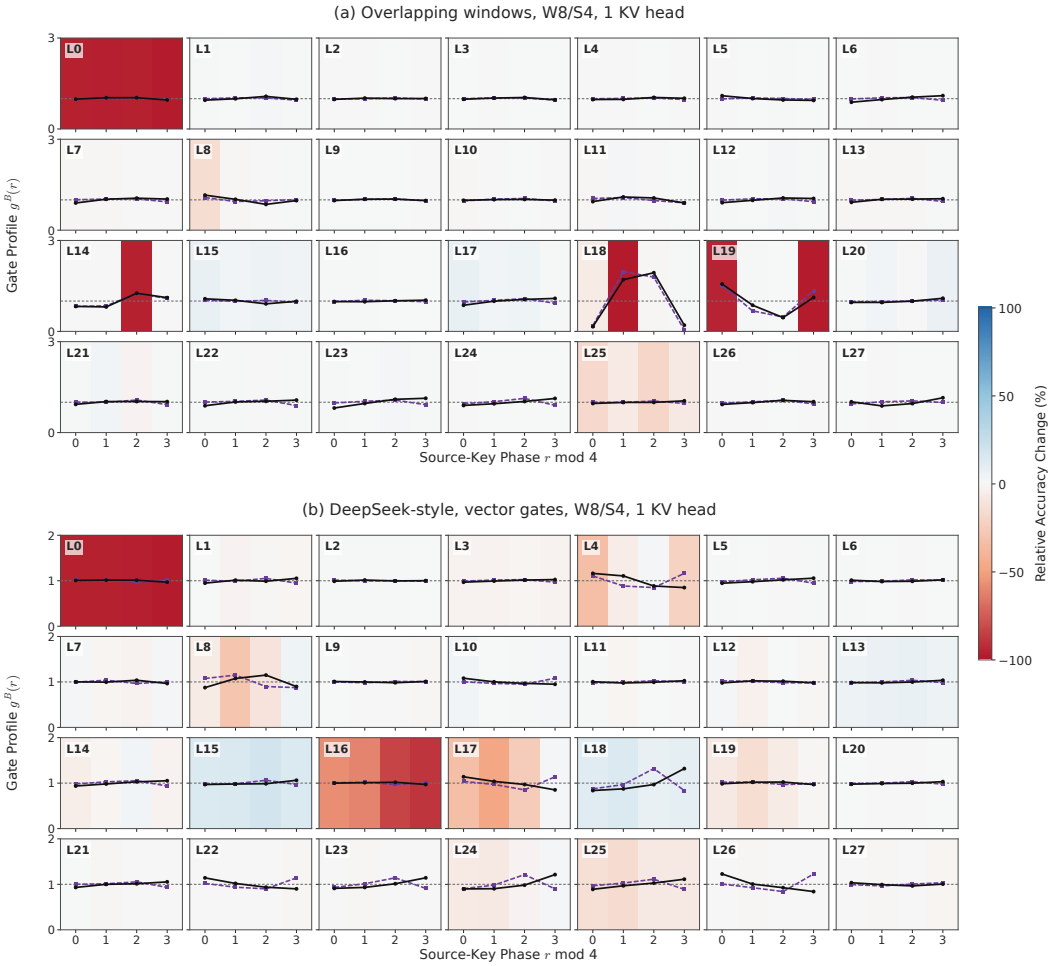

Figure 18: **Gate preferences and knockout effects with overlapping windows.** (a) Overlapping windows (W8/S4, one KV head). (b) DeepSeek-style kernel with vector gates (W8/S4, one KV head). Each phase receives two of the eight window offsets, and the curves add their scores, rescaled so that a uniform gate still equals one. In (b), K and V share one gate, so the purple curve is the shared gate read one token later, and vector-gate scores are averaged over channels. No hollow markers appear because $W \neq S$. Encoding, protocol, gate inputs, and checkpoint step follow Figure 16.

H REDIRECTING GATE PREFERENCES

In Section 4.2, we found a few gates with sharp offset preferences that persist across windows and inputs, and knocking out three of the corresponding heads in the reference model costs the most accuracy near the phases their gates favor. If such preferences help decide which phases are weak, editing them after training may move the weak phases with them. We test this by cycling each gate's preference by a fixed number of offsets and checking whether the accuracy pattern across phases shifts by as many phases.

H.1 GATE CYCLING

In models with non-overlapping windows ($W = S$), window offset i is also phase i. A gate's preference for offset i comes from its gate map $\boldsymbol{Z}_i^B$ and positional gate bias $\boldsymbol{b}_i^B$ (Section 3.1). Cycling by δ moves both forward by δ offsets, while the payload maps stay fixed:

$$\widetilde{\boldsymbol{Z}}_i^B = \boldsymbol{Z}_{(i-\delta) \bmod W}^B, \qquad \widetilde{\boldsymbol{b}}_i^B = \boldsymbol{b}_{(i-\delta) \bmod W}^B, \qquad B \in \{K, V\}. \tag{118}$$

The gate parameters of offset i are thus applied at offset $(i+\delta) \bmod W$. We move maps and biases together, not the biases alone, because both enter the gate logits. The hidden states stay at their original offsets, so this does not move the realized gate scores exactly, and whether the preferences move with the parameters is an empirical question. If they do, and retrieval follows the gates, the accuracy pattern should shift by δ phases,

$$A_\delta(r) \approx A_0\big((r-\delta) \bmod S\big), \tag{119}$$

where $A_0(r)$ and $A_\delta(r)$ are the accuracies at source-key phase r before and after cycling. The prediction concerns associations within one window. Each value immediately follows its key, so the two share a window except at the boundary phase $r = S-1$, which we exclude from R^2 as specified for each experiment.

In each experiment, we cycle the key and value gates of every head in layers 3 to 23 (zero-based) at step 47,518 and measure full-vocabulary top-1 accuracy on 1,000 logical examples of the retrieval task in Appendix C.2 with Prefix Padding, folding the 24 source-key residues modulo S. R^2, the squared Pearson correlation between predicted and observed accuracy changes across phases, measures linear agreement rather than agreement with the identity line, which the figures draw. Intervals are bootstrapped over logical examples.

H.2 ONE-OFFSET CYCLES IN A W12/S12 MODEL

We first cycle a separate W12/S12 model with four KV heads per layer by one offset in each direction. Away from the boundary phase, both cycles shift the accuracy pattern approximately as predicted (Figure 19), with R^2 of 0.984 for $\delta = +1$ and 0.995 for $\delta = -1$ over comparisons in which phase 11 is neither the observed nor the source phase. Intervals use 2,000 bootstrap resamples. This

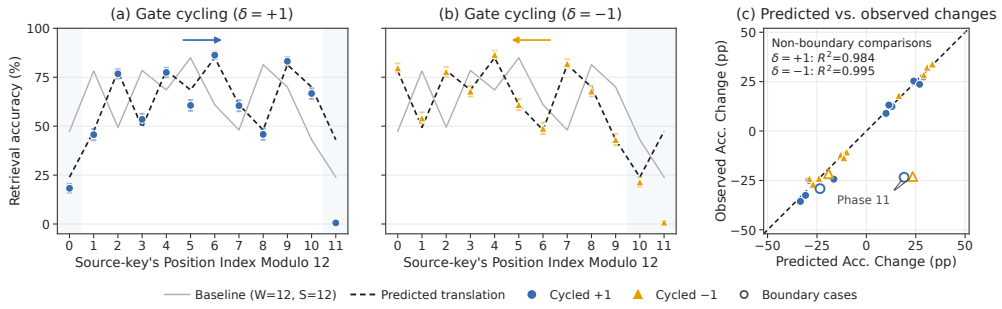

Figure 19: **Gate cycling approximately shifts retrieval accuracy.** (a,b) Accuracy by source-key phase in the W12/S12 model after cycling by $\delta = +1$ and $\delta = -1$ (markers), with the uncycled model (gray) and the prediction (dashed). (c) Predicted versus observed changes and the identity line (dashed). Shaded and hollow points involve the boundary phase and are excluded from R^2.

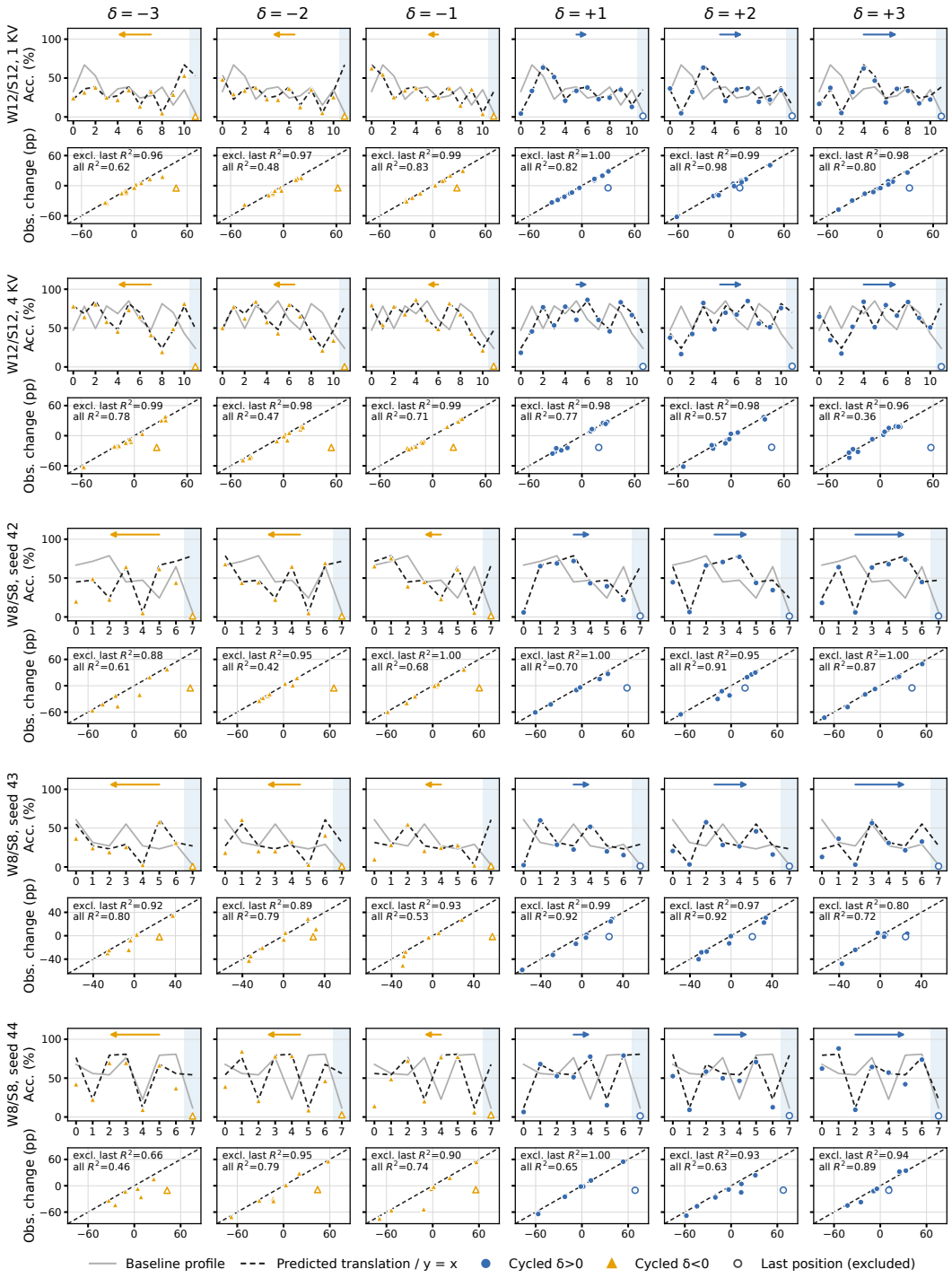

Figure 20: **Gate cycling shifts retrieval accuracy across models and cycle sizes.** Each cell follows Figure 19 for one model and one cycle δ, with accuracy by source-key phase above and predicted versus observed accuracy changes below. Rows show the two W12/S12 models, the W8/S8 reference model (seed 42), and two of its seed replications (seeds 43 and 44), each W8/S8 model with one KV head. Columns show δ from -3 to $+3$, and arrows give its direction and size. Error bars are 95% bootstrap intervals. Each cell reports R^2 without the boundary phase $S-1$ ("excl. last") and over all phases ("all"). Unlike in Figure 19, "excl. last" excludes only comparisons whose observed phase is $S-1$.

supports a causal contribution of gate preferences to within-window retrieval in this model. Phase 11 departs from the prediction under both cycles, which limits a purely within-window account but does not by itself identify a separate retrieval circuit.

H.3 Replication across Models and Cycle Sizes

One model and one-offset cycles leave open whether the shift is specific to this configuration. We therefore repeat the intervention on five models: the four-KV-head W12/S12 model above, a W12/S12 model with one KV head, the W8/S8 reference model of Section 4.2 (seed 42), and two of its seed replications (seeds 43 and 44), each W8/S8 model with one KV head. We apply each cycle $\delta \in \{\pm 1, \pm 2, \pm 3\}$ to the original parameters. Unlike in the one-offset experiment, R^2 here excludes only the observed phase $S-1$, and we also report it over all phases. Intervals again use 2,000 bootstrap resamples. For the four-KV-head W12/S12 model, the cycles $\delta = \pm 1$ reproduce the accuracies of Figure 19.

Away from the boundary phase, the observed accuracy patterns largely follow the prediction for every model and cycle (Figure 20). Excluding phase $S-1$, R^2 ranges from 0.96 to 1.00 for both W12/S12 models, from 0.88 to 1.00 for the W8/S8 reference model, and from 0.66 to 1.00 for the two seed replications. Editing the gates alone thus largely shifts the weak within-window phases in all five models, most closely in the W12/S12 models. It does not restore the boundary phase, whose accuracy stays at or below 2.9% under every cycle, compared with 2.7% to 23.9% in the uncycled models, even where the prediction is high. Including this phase lowers R^2 to between 0.36 and 0.98. Cycling also lowers overall full-vocabulary accuracy by 3.3 to 18.4 percentage points, so it redirects the weak within-window phases rather than repairing the model.

Beyond Figure 20, a fourth W8/S8 seed (45) behaves like the others, with R^2 excluding phase $S-1$ between 0.68 and 0.99, but agreement weakens as W8/S8 models gain KV heads. The minimum over cycles is 0.75, 0.43, and 0.36 with 2, 4, and 8 KV heads, and with 8 KV heads no cycle exceeds 0.86. Because every head in layers 3 to 23 is cycled together, these results do not show whether individual heads in these models lack preferences that would shift accuracy when cycled. The layer range was carried over from the one-offset experiment rather than selected for each model.

I Extended Related Work

Learned summary memories. Compressive Transformer down-samples old activations into a FIFO compressed memory using stride-matched pooling or convolution (Rae et al., 2020). Gist tokens and Landmark Attention use special tokens to condense prompts or route random access to blocks (Mu et al., 2023; Mohtashami & Jaggi, 2023). LoMA adds an explicit reconstruction objective (Wang & Xiao, 2024). Activation Beacon inserts regularly spaced learned beacon tokens whose KVs replace raw activations, and MELODI builds a recurrent short-term memory across multiple layers and a long-term memory in a single middle layer (Zhang et al., 2025; Chen et al., 2025). CAT is particularly close at the architectural level: later chunks attend to compressed prior-chunk vectors, and its appendix reports high loss at the first token of each new chunk and recall degradation as chunks grow (Prakash et al., 2025). None of these works examines retrieval as a function of source phase within compressed memory.

Cross-chunk failure modes. Deng et al. (2025) systematically compare recurrent, coarse-KV, and fine-grained-KV gist compression. Their reconstruction probe shows a severe information bottleneck as compression rises, and their "lost by the boundary" diagnostic finds a segment-scale sawtooth: with 2K segments, generation is weakest immediately after a segment boundary, when predictions depend most strongly on compressed prior segments, and improves as raw tokens accumulate in the current segment. They also document semantically surprising facts being dropped and long exact strings being lost during generation. KSA's chunk-aligned sliding window addresses a separate case in which a partially visible chunk is neither locally complete nor yet represented by a summary (Chu et al., 2026). HMT uses the phrase "periodical recall pattern" for successful tracking of an artificially periodic topic-switch sequence, not an architecture-induced failure (He et al., 2025). Among these works, the sawtooth of Deng et al. (2025) is the direct precedent for recurring cross-chunk degradation, but its phase variable is the *query/generation position* within the current

2K segment. Our phase variable is the *source position* modulo the much smaller compression stride inside already-compressed memory.

Sparse and hybrid routing. Sparse Transformer explicitly defines a strided head with modulo connectivity and notes that the pattern may route poorly on nonperiodic text (Child et al., 2019). It is thus a precedent for residue-class visibility as a general attention concept. MLA reduces redundancy across heads and channels rather than along the token axis (DeepSeek-AI, 2024). NSA forms overlapping compressed block representations for coarse selection and retains an exact fine-grained branch (Yuan et al., 2025). MoBA routes queries to selected raw-token blocks (Lu et al., 2025), and the Sparse Frontier compares the quality–efficiency trade space across such sparse patterns (Nawrot et al., 2026). DeepSeek-V4 selects sparsely among compressed KV entries and adds an uncompressed local window (DeepSeek-AI, 2026a). In hybrids with an exact branch over distant tokens, such as NSA, that branch could mask a failure of the compressed branch alone. DeepSeek-V4 keeps exact tokens only in its local window, and in Section 2, we find phase sensitivity in the full models.

Cache allocation and specialized retrieval components. H2O and SnapKV evict or cluster tokens using estimated importance (Zhang et al., 2023; Li et al., 2024). SqueezeAttention allocates different cache budgets across layers (Wang et al., 2025), while RazorAttention and DuoAttention retain long-range caches for retrieval heads and use compressed or streaming policies elsewhere (Tang et al., 2025; Xiao et al., 2025). Wu et al. (2025) causally identify a small subset of heads that dominate long-context factual retrieval. Generic heterogeneity across heads and layers is therefore already established. Our empirical claim is more specific: KV-head knockouts in compressed models have phase-dependent effects that, in the examined heads, match the phases their gates prefer. A supplementary gate-cycling intervention on five models with non-overlapping windows complements this association (Appendix H).

Retrieval diagnostics and position bias. Zoology's multi-query associative recall isolates a major capability gap between attention and efficient sequence models (Arora et al., 2024). The copying results of Jelassi et al. (2024) and induction-circuit studies (Olsson et al., 2022; Bietti et al., 2023) motivate our synthetic retrieval task. Lost in the Middle establishes broad U-shaped sensitivity to absolute target position (Liu et al., 2024), and RULER shows that easy needle tests can overstate usable context length (Hsieh et al., 2024). The Pitfalls of KV Cache Compression further shows that aggregate scores hide instruction- and method-specific failures (Chen et al., 2026). We therefore match the mean and variance of the target's absolute position across residue groups and report retrieval by phase rather than only mean accuracy.

Optimization and specialization. Max-margin analyses show that gradient descent can turn softmax attention into hard token selectors (Tarzanagh et al., 2023a;b), although entropy concentration can also be pathological (Zhai et al., 2023). Chen et al. (2024) prove staged multi-head task allocation in in-context linear regression, and Yang et al. (2025) analyze population gradient flow in a symmetric two-headed classification transformer. Most closely related, Yüksel et al. (2026) analyze near-symmetric heads that first compete for a dominant sparse pattern and later specialize through competition and cooperation. High-dimensional SGD likewise exhibits sequential head specialization (Sagitova et al., 2026), while generic softmax factorization can bias gradient flow toward low-entropy outputs (Varre et al., 2026). We therefore do not claim the first theory of attention hardening, symmetry breaking, or division of labor. Our theory (Section 5) concerns an idealized induction model of retrieval from compressed memory. In this model, optimal compressors select sharply, gradient flow from a small initialization reaches, with high probability, a selection that persists across inputs, and nothing guarantees that heads choose distinct positions.