FROM RECURRENT COMPUTATION TO ATTENTION MEMORY IN HYBRID LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Hybrid language models store context in recurrent state and an attention key-value (KV) cache. To understand how they use context, we need to trace how computation shapes the memory used for a later answer. Swapping the final recurrent state measures how much it affects the answer while holding the KV cache fixed. This leaves the effects of earlier recurrent computation on the KV cache unchanged. To trace these effects, we replace recurrent-block contributions during prefill, including their feed-forward sublayers in Qwen. The replacements come from a source context that specifies a different answer. We let later layers recompute both memories, then restore either one before the query. Most of the induced change in answer preference remains after restoring the original recurrent state. Across five hybrid models, retaining the modified KV cache preserves 67-93% of the mean shift in the answer-logit margin on exact retrieval. Restoring the original KV cache instead removes or weakens this effect. Layerwise tests locate the effect in the KV caches of deep attention layers. The effect also persists through complete-answer generation. Of 200 paired SQuAD passages with 2-5-token answers, both Qwen3.5-9B and Qwen3.6-27B answer both clean contexts correctly on 162 pairs. After restoring clean recurrent state, the modified KV cache preserves the complete source answer on 98.8% and 98.1% of this shared subset, respectively. These results support a *compute-commit-retrieve* pattern: Recurrent computation can change an answer through attention memory, even when swapping the final recurrent state has little effect on that answer.

1 INTRODUCTION

When a hybrid language model reads a context, it builds two kinds of memory: Recurrent layers update a fixed-size state, whereas attention layers store token-level keys and values (KV) in a cache. These layers act in sequence. Qwen3.5 (Qwen Team, 2026a), for example, places three Gated DeltaNet (GDN) layers (Yang et al., 2025) before each attention layer. Attention, therefore, builds its cache from representations already shaped by recurrent computation.

Swapping the final recurrent state, after prefill, measures its effect on the answer with the KV cache fixed. In our core retrieval experiments, the KV cache swaps reverse answer preference, whereas final-state swaps do not. A natural but incomplete reading of this contrast is that recurrence contributes little to answering. Final-state swaps leave earlier recurrent contributions to the KV cache unchanged. We instead ask: *Which memory preserves the answer change caused by an intervention on recurrent computation?*

We trace this change from prefill to later query (Fig. 1). We replace recurrent-block contributions with those from a context containing a different answer, then let later layers recompute both memories. Before asking the shared query, we restore either memory to its original value. On Qwen3.5-9B, restoring recurrent state preserves the changed preference in every tested exact-retrieval pair. Restoring the KV cache returns every pair to the original preference. Therefore, the recurrent intervention changes the later answer preference through the KV cache generated during prefill.

Across five hybrid language models, the KV cache preserves most of the change in answer preference even after the original recurrent state is restored. Layerwise tests locate this effect in the KV caches of deep attention layers. The trace also extends to computed answers and complete multi-token answers in paired SQuAD passages. This *compute-commit-retrieve* pattern separates

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

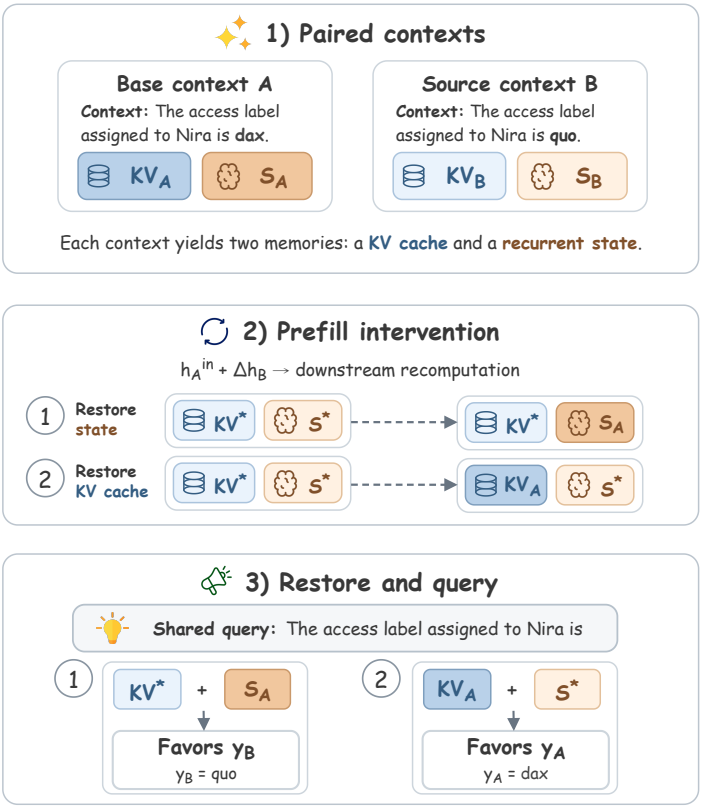

Figure 1: Tracing recurrent contributions into the KV cache. A and B are base and source contexts sharing a query, with answers y_A and y_B. During A's prefill, we replace recurrent-block contributions with those from B. Stars mark the recomputed KV cache and recurrent state, (KV^*, S^*). Restoring S_A preserves preference for y_B; restoring KV_A returns preference to y_A (Table 2).

how a memory is formed from how it affects an answer. That distinction can help evaluate cache compression and diagnose errors when cached contexts are reused.

Our contributions are as follows:

- **Tracing recurrent computation into the KV cache.** Across five models, we find that the recomputed KV cache preserves most of the effect of recurrent prefill, even after the original recurrent state is restored.
- **Extending to arithmetic and multi-token answers.** The pathway extends to arithmetic answers and, in two checkpoints, to complete-answer generation on paired SQuAD passages.
- **Distinguishing decodable information from answer-relevant effects.** Probes can recover individual labels from the recurrent state even when swapping that state leaves preference between the two answers unchanged. However, when multiple facts compete, the probes struggle to identify which label is bound to the target entity.

2 Related Work

We introduce work on hybrid recall which studies memory capacity, and causal interventions that locate the computations and cached states that influence an answer.

Hybrid memory and recall capacity. Qwen3.5 and Qwen3.6 use GDN for their recurrent layers (Yang et al., 2025; Qwen Team, 2026a;b). Jamba, Nemotron-H, and Falcon-H1 combine attention with Mamba-style recurrence (Lieber et al., 2024; Dao & Gu, 2024; Blakeman et al., 2025; Zuo et al., 2025). Associative-recall and copying studies expose differences between bounded re-

current state and token-addressable attention (Arora et al., 2024a;b; Jelassi et al., 2024). KV-cache compression instead asks which cached entries can be removed while preserving generation (Zhang et al., 2023; Xiao et al., 2024; Li et al., 2024). We ask how a change to recurrent computation during prefill persists in the cache used by a later query.

Localization and interchange interventions. Causal tracing and path patching locate answer-relevant computation in Transformers and recurrent models (Meng et al., 2022; Wang et al., 2022; Goldowsky-Dill et al., 2023; Sharma et al., 2024; Arora et al., 2025). KV-cache editing, recurrent-write decomposition, and retrieval-head analysis examine specific memory pathways (Li, 2026; Wu et al., 2025), whereas component ablations in hybrid models measure their behavioral contributions (Borobia et al., 2026). Our approach combines these ideas in a two-stage intervention. We first alter a block's contribution during prefill, then separately restore the recurrent state and the KV cache produced by that intervention. By testing whether the resulting change in the answer remains after each restoration, we can determine which memory pathway carries the prefill effect. The replacement activations arise from valid source inputs, following interchange-based causal abstraction (Geiger et al., 2021; 2024). Because the measured effect can depend on the intervention and on interactions between components, we evaluate it using answer margins, factorial effects, matched controls, and preference reversals (Zhang & Nanda, 2024; Vaidyanathan et al., 2026).

3 TRACING COMPUTATION INTO MEMORY

We separate query-time cache control from prefill contributions, then trace the induced effect through cache restoration.

Models and paired contexts. Our primary model is the frozen Qwen3.5-9B with eight macro-blocks. Each macro-block contains three GDN layers and one attention layer. Attention layers maintain token-level KV entries, whereas recurrent layers maintain a convolutional state and a fixed-size matrix state. We also test Qwen3.6-27B, Qwen3.6-35B-A3B, and Nemotron-3-Nano-4B/30B-A3B (abbreviated Nemotron-4B/30B), covering GDN, Mamba-2, dense, and mixture of experts (MoE) variants (Appendix A).

The boxes below show examples of four synthetic task types and SQuAD question answering. Each synthetic item pairs equal-length base (A) and source (B) contexts that share a query but differ in answer-relevant content, yielding distinct single-token answers y_A and y_B. Context B supplies replacement activations or caches. We generate 300 candidate pairs per synthetic task and retain those where both clean runs clearly favor their respective correct answer over the alternative (Appendix A).

Exact retrieval
Context: The access label assigned to Nira is dax.
Query: The access label assigned to Nira is
Answer: **dax**

Instruction following
Context: For the final question, output only the label dax.
Query: What label should be returned? The label is
Answer: **dax**

State tracking
Context: Slot one holds dax. Slot two holds quo. The two slots are swapped.
Query: Slot one now holds
Answer: **quo**

Arithmetic
Context: The first number is two. The second number is three.
Query: The sum of the two numbers is
Answer: **five**

SQuAD: complete-answer generation
Context: Beyoncé names Michael Jackson as her major musical influence.
Query: Who influenced Beyonce?
Answer: **Michael Jackson**

The prefill/query boundary. After processing the context without the query, we intervene on the caches and process the shared query to obtain answer-position logits. State replacements include both convolutional and matrix state. Swaps change only the cached context, preserving the query, token positions, and model weights. Implementation checks and noise controls are detailed in Appendices A and B.

Test 1 (Retrieve): Which cache controls answer preference? We cross the KV caches and complete recurrent states S from A and B to obtain (KV_i, S_j) for $i, j \in \{A, B\}$. To measure which memory changes answer preference, we compute the answer-logit margin, each memory's average effect, and their interaction:

$$m_{ij} = \mathrm{logit}(y_A \mid \mathrm{KV}_i, S_j) - \mathrm{logit}(y_B \mid \mathrm{KV}_i, S_j),$$
$$E_{\mathrm{KV}} = \tfrac{1}{2}[(m_{AA} - m_{BA}) + (m_{AB} - m_{BB})],$$
$$E_S = \tfrac{1}{2}[(m_{AA} - m_{AB}) + (m_{BA} - m_{BB})],$$
$$I = m_{AA} - m_{AB} - m_{BA} + m_{BB},$$
$$O = \frac{|E_{\mathrm{KV}}| - |E_S|}{|E_{\mathrm{KV}}| + |E_S|}.$$

A positive margin m favors y_A over y_B. Each main effect averages over the two settings of the other memory, and the contrasts cancel fixed answer-token offsets. I measures the interaction between the memories. The relative KV-cache effect O is positive when the KV-cache effect is larger.

We count a ***flip*** when preference between y_A and y_B reverses. The stricter global top-1 outcome requires y_B to have the highest logit across the full vocabulary.

Tests 2 (Compute) and 3 (Commit): Tracing the prefill effect into memory. In the first d macroblocks, we replace recurrent- or attention-block contributions in A with their values from B and let later layers recompute. For a block mapping residual input h^{in} to h^{out}, we record $\Delta h_B = h_B^{\mathrm{out}} - h_B^{\mathrm{in}}$ and replace the base output with $h_A^{\mathrm{in}} + \Delta h_B$, preserving the current incoming residual. In Qwen, this contribution includes the feed-forward sublayer; hence, attribution is at the block level. Later attention layers write KV entries from the recomputed representations. We intervene on recurrent and attention blocks separately. Each targeted layer first computes its state or KV entries from its current input, including earlier replacements; only its output is then replaced. Its cache is neither copied from B nor forced to its clean A value; S^* denotes the states produced by this intervened run. Prefill interventions, therefore, measure effects through later computation, whereas query-time cache swaps replace stored entries directly. KV-cache swaps can thus have larger effects than attention-block contribution replacements.

After a prefill intervention produces (KV^*, S^*) and preference for B, we restore each base memory separately (*re-crossing*). (KV^*, S_A) tests whether the modified KV cache preserves the new preference. (KV_A, S^*) tests whether the modified recurrent state preserves it.

Replacing contributions with A's own values is the no-op control: Answer margins change by at most 0.1875/0.125 logits on 9B/27B, with 240 pairs per checkpoint (Appendix A). Re-crossing identifies which memory retains the prefill effect; a final-KV-cache swap alone does not trace its origin.

We measure how much of the prefill effect survives when only the modified KV cache is retained:

$$C_{\mathrm{KV}} = \frac{\mathbb{E}[m_A - m(\mathrm{KV}^*, S_A)]}{\mathbb{E}[m_A - m(\mathrm{KV}^*, S^*)]}, \qquad m_A = m(\mathrm{KV}_A, S_A).$$

This ratio compares mean answer-margin changes, not fractions of stored information. The state ratio R_S replaces the crossed margin in the numerator with $m(\mathrm{KV}_A, S^*)$. Their sum differs from one by a normalized interaction term (Appendix E.2).

4 Tracing the pathway from recurrence to KV cache

The three tests trace the recurrent-prefill effect into the KV cache; probe readouts then examine information retained in the recurrent state.

Table 1: Query-time cache swaps on Qwen3.5-9B ($n = 300$ per task). Effects are mean logit contrasts; brackets give 95% confidence intervals (Appendix A). Flip denotes reversed answer preference.

Task	E_{KV}	E_S	O	KV flip	State flip
Exact retrieval	18.5 [18.2,18.8]	1.4 [1.4,1.5]	0.85	100%	0%
Instruction following	24.0 [23.7,24.3]	2.9 [2.8,3.0]	0.78	100%	0%
State tracking	13.5 [13.2,13.8]	–0.6 [–0.6,–0.5]	0.92	100%	0%

4.1 KV CACHE CONTROLS ANSWER PREFERENCE AT QUERY TIME

On Qwen3.5-9B, swapping the full KV cache reverses answer preference in all 300 pairs in each core task; swapping the complete recurrent state reverses it in none (Table 1). When the two memories come from contexts with different answers, preference follows the KV cache. Its effect exceeds the state effect in every pair, with much smaller interactions between the memories (Appendix B).

Several deep layers contribute. Replacing the deepest attention layer alone does not reverse preference. The deepest two layers reverse 81% of exact-retrieval pairs and the deepest four layers reverse every pair. Random token-position swaps also become more effective as coverage increases (Appendix B). The influential entries, therefore, span layers and positions. These swaps locate answer-relevant memory; the next test traces how prefill computation shapes it.

4.2 RECURRENT PREFILL CONTRIBUTIONS REDIRECT THE LATER ANSWER

Replacing recurrent-block contributions across Qwen3.5-9B's eight macro-blocks reverses preference in every exact-retrieval pair, compared with at most 10% for attention-block contributions. Qwen3.6-27B shows the same contrast (Fig. 2a). Strong recurrent-prefill effects coexist with weak final-state effects (Fig. 2c) because prefill interventions can change both memories through later computation; final-state swaps leave the KV cache fixed.

The contrast persists under depth and count controls. Leaving the first macro-block clean preserves every reversal and replacing only blocks five through eight still reverses 99%. These depth-restricted interventions show that the effect extends beyond the earliest context processing. Matching layer counts also preserves the larger recurrent effect (Appendix Fig. 4). Single-layer comparisons provide secondary evidence: The strongest recurrent layer is layer 1, nearest the embeddings, whereas the strongest attention layers are 20/24 for exact retrieval/state tracking; these are not depth-matched.

The answer tracks the supplied content. On exact retrieval, source B makes y_B the global top-1 token in 76% of runs and a third source C makes its answer z top-1 in 88%; equal-norm noise never makes y_B top-1. Both the recurrent mixer and its feed-forward sublayer transfer the effect; hence, the traced route involves the recurrent block as a whole (Appendix B).

4.3 THE PREFILL EFFECT PERSISTS IN THE RECOMPUTED KV CACHE

Table 2 separates the two memories produced by the full recurrent-prefill intervention. Restoring the original recurrent state preserves source preference in every pair; restoring the original KV cache returns every pair to the base preference. The modified KV cache, therefore, carries the induced change across the prefill/query boundary.

We use *commit* to describe persistence after clean state restoration. Later attention layers write KV entries from representations changed during recurrent prefill. In a separate intermediate-state swap followed by a neutral suffix, installing only the new KV entries preserves the base answer in every pair (Appendix D.2). The positive trace concerns block contributions during prefill, rather than transfer from an arbitrary stored state.

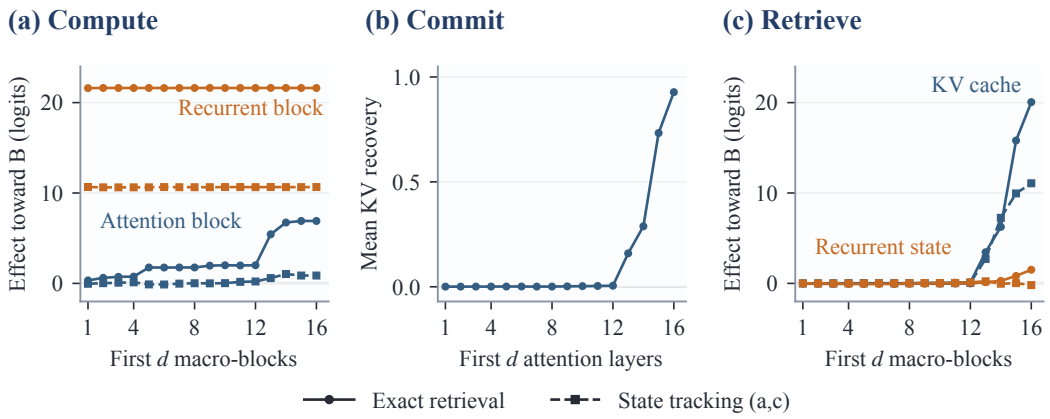

Figure 2: Cumulative interventions on Qwen3.6-27B. **(a) Compute:** Replace prefill block contributions. **(b) Commit:** Retain modified KV entries with clean state. **(c) Retrieve:** Swap final caches. Panels (a,c) share 120 pairs per task; (b) uses an earlier 100-pair retrieval run. Estimators and protocols are presented in Appendix B.1.

Table 2: Memory restoration after recurrent prefill on Qwen3.5-9B ($n = 120$ per task). Rates give preference for y_B over y_A. Restorations start from the intervened run; installation starts from clean A.

	Follows B		
Operation	Exact retrieval	State tracking	Interpretation
Clean baseline	0%	0%	Base preference
Prefill intervention	100%	100%	Source preference
Restore recurrent state	**100%**	**100%**	Modified state not required
Restore KV cache	**0%**	**0%**	Modified state insufficient
Restore block-7 KV cache	57%	98%	Effect also retained elsewhere
Install block-7 KV cache	47%	3%	One block is insufficient

The effect is spread across deep KV caches. Block 7 was selected because its single-layer KV-cache swap had the largest exact-retrieval effect (Appendix B). Restoring its KV cache removes roughly half the prefill effect, but installing that block alone does not reproduce the full-cache result. Restoring it in state tracking leaves 98% of pairs following B (Table 2). Its contribution, therefore, depends on the task. In Qwen3.6-27B, modified KV entries from the first 12 attention layers recover almost none of the effect; adding the final four layers recovers nearly all of it (Fig. 2b). Both MoE curves also rise late (Appendix Fig. 5). These cumulative curves locate the effect in a deep group of layers, without measuring each layer's independent contribution.

4.4 The pathway extends to computed and multi-token answers

Arithmetic and paired SQuAD passages test whether the pathway extends to computed answers and complete multi-token answers.

Computed answers. On Qwen3.5-9B, one-digit addition requires an answer absent from the prompt. Recurrent-prefill intervention still reverses preference in every retained pair and the modified KV cache preserves the change after clean state restoration; the modified state alone does not. Position controls trace effects beyond operand tokens without locating the addition operation (Appendix B). The pathway thus extends beyond retrieval of a stated answer.

Complete answers persist after state restoration. We test 200 paired SQuAD passages with equal-length substitutions of 2-5-token proper names (92.5% people), preserving questions and token alignment (Rajpurkar et al., 2016; 2018). On the shared 162 pairs where both models answer both clean contexts correctly, recurrent-prefill intervention produces the complete source answer in every pair. Restoring clean state preserves it in 98.8% of 9B and 98.1% of 27B pairs; the modified state alone produces none (Fig. 3a). The effect survives complete-answer decoding in this controlled extractive setting.

Across all 200 pairs, source-answer exact match (EM) is 84.0% for 9B and 85.0% for 27B with the modified KV cache, versus 85.0% and 87.5% on clean source contexts. The gaps are 1.0 and 2.5 percentage points; hence, transfer remains close to clean source performance with baseline failures included. Attention prefill reduces base-answer EM from clean values of 92.0/89.0% to 75.0/56.0% (9B/27B). Its sequence-margin shift is $\Delta g = 10.10/11.98$ nats, compared with 37.40/34.36 nats for recurrent prefill. Tables 7 and 8 present all conditions; Appendix D.1 defines Δg, describes decoding, and reports token-probability analyses and 100 audited count questions.

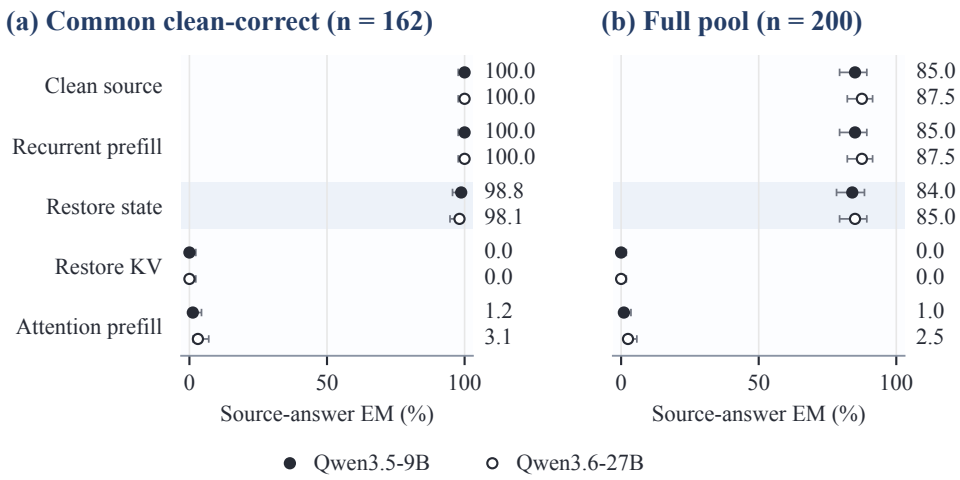

Figure 3: Complete source-answer EM on paired SQuAD passages. **(a)** Common clean-correct subset ($n = 162$). **(b)** Full pool ($n = 200$). Filled/upper circles: Qwen3.5-9B; open/lower: Qwen3.6-27B. Bars show 95% Wilson intervals; clean-source EM is the reference. Restore state/KV cache denotes clean base memory restoration after recurrent prefill.

4.5 The pathway recurs across models

Smaller Qwen checkpoints reproduce the query-time contrast. Pretrained-only 9B-Base reproduces all three stages (Appendix B). Nemotron-4B uses Mamba-2 instead of GDN and shows the same compute-commit-retrieve pattern (NVIDIA, 2025; Blakeman et al., 2025). Nemotron-4B reaches 100% prefill reversals and $C_{\mathrm{KV}} = 0.87$ while leaving its separate feed-forward network (FFN) layers unpatched (Table 3). Direct FFN replacement is, therefore, not required for the effect. The Retrieve and Prefill columns report preference-reversal rates; Modified KV reports source preference after state restoration and C_{KV} gives recovery of the mean prefill effect.

After clean state restoration, the modified KV cache retains 67-93% of the mean prefill effect on the answer margin: about two-thirds to nearly all of the induced shift. State still contributes in some checkpoints. On Nemotron-30B, state swaps reverse 11% of pairs. The modified KV cache alone preserves source preference in 73%. Under full intervention on the 9B SQuAD pairs, modified and clean source KV caches produce nearly equal sequence-margin shifts with clean state fixed: 36.37 versus 36.39 nats (Table 8). The additional evidence comes from locating the retained effect in deep KV caches in Test 3 and the recurrent/attention asymmetry in Test 2. Routing and paraphrase controls examine this variation, but training recipe and active capacity remain entangled with feed-forward type (Appendix B).

Table 3: Exact-retrieval pathway across five checkpoints. In row order, $n_{\text{retrieve}}/n_{\text{prefill}} = 300/120,\ 150/100,\ 60/60,\ 150/100,\ 148/80$. Prefill counts also apply to re-crossing.

Model	Recurrent	FFN	Retrieve KV/state	Prefill recur/attn	Modified KV follows B	C_{KV}
Qwen3.5-9B	GDN	dense	100/0%	100/10%	100%	0.93
Qwen3.6-27B	GDN	dense	100/0%	100/1%	100%	0.93
Nemotron-4B	Mamba-2	dense	100/0%	100/0%	100%	0.87
Qwen3.6-35B-A3B	GDN	MoE	99/0%	100/0%	96%	0.70
Nemotron-30B	Mamba-2	MoE	90/11%	100/0%	73%	0.67

4.6 Recurrent state retains information despite weak answer control

Probe readouts and binding controls separate information decodable from recurrent state from its influence on generated answers.

Decodable labels need not control the answer. A linear classifier recovers isolated labels with 100% accuracy from 27B and 35B-A3B recurrent matrix states on held-out entities. Shuffled-label controls remain near chance. A 50-label test gives nearly the same result (Appendix C). Information is accessible to the probe whereas state swaps preserve answer preference; on 35B-A3B, the state still shifts the answer margin by 6.8 logits.

Selecting a fact becomes harder with competitors. In a separate binding task with 30 examples per class, dense-model probe accuracy falls from 93% with one competing fact to 2.5% with 24, near the 2% chance level; the MoE model follows a similar curve (Appendix Fig. 6). This decline is within the binding task; the isolated-label test uses a different generalization protocol. Failure of these readouts does not establish that all fact information is lost.

Behavior also depends on the added context. With filler facts lengthening 9B contexts from about 70 to 8200 tokens, the state-swap effect approaches zero whereas every retained pair follows the KV cache. Separate MoE controls preserve more state contribution under unrelated prose than under additional facts with the same format. Token count alone, therefore, does not explain the reduced state effect (Appendix C).

5 Discussion

In our tests, the recurrent-prefill effect persists in the KV cache even when final-state swaps have little effect on the answer. A decomposition explains this pattern.

Computation can influence an answer through a different memory. Changing a block's contribution during prefill can affect the answer through either memory. Let u denote that contribution and $K(u), S(u)$ the resulting caches. The answer margin for query q is $m(K, S; q)$, where K is the flattened attention KV cache. In a differentiable neighborhood, the chain rule gives

$$\frac{\partial m}{\partial u} = \underbrace{(\nabla_K m)^\top \frac{\partial K}{\partial u}}_{\text{through written KV}} + \underbrace{(\nabla_S m)^\top \frac{\partial S}{\partial u}}_{\text{through final state}}. \tag{1}$$

The first term describes how the changed computation affects the answer through the KV cache. It can be large even when the second term, through final state, is small. Holding the KV cache fixed isolates the state route, whereas a prefill intervention can change both routes. Appendix E.1 shows the derivation for this local identity, with fixed routing where needed, and presents an exact example for finite interventions. Our experiments measure finite changes by recomputing and restoring caches; they do not estimate these derivatives.

Stored information, readout, and later writes. A probe may recover information that has little effect on the model's own answer. That information may also have no effect on KV entries written when the model reads a later suffix. These outcomes can coexist because probing, answering, and

writing new KV entries use state in different ways. Appendix E.1 presents an example with all three properties and a strong effect from earlier recurrent computation into the KV cache. It explains why the positive prefill result can coexist with the negative neutral-suffix result (Appendix D.2). Small state effects can also change answer probabilities without changing the highest-scoring token. Thus, both probe accuracy and probability changes can reveal information that does not alter the generated answer.

Variation and scope. Across the tested checkpoints, $C_{\mathrm{KV}} = 0.67$-0.93, with larger residual state effects in some MoE models. The decomposition enables either route to matter, but their measured strengths depend on the checkpoint and task. Our evidence covers frozen models, controlled synthetic answers, and short proper-name answers in paired SQuAD passages. We report the full SQuAD pool alongside subsets where clean answers are correct. The interventions identify a route from recurrent computation into the KV cache. They do not separate contextual recurrent processing from current-token pathways, including the short convolution in Mamba-2. They do not locate every computation needed for the answer or explain what causes the differences between architectures.

Implications for cache management. The large answer effects of swapping KV caches in deep layers make these layers useful targets for testing cache quantization, compression, and eviction. Complete-answer exact match measures changes in generated answers, alongside finer changes in probe readouts and answer probabilities. Reused prefixes need several test queries because cache changes can affect their answers differently. Paired queries can expose fluent answers that use the wrong contextual fact; selective cache restoration can locate the responsible memory.

A small final-state effect does not justify removing recurrent state or recurrent prefill. The state participates in later computation. Prefill helps form the KV cache being reused. Our results suggest how cache changes can be evaluated and diagnosed. Any gains in throughput or memory use still require implementation and measurement in a serving system.

In the tested hybrid models, the KV cache retains much of the recurrent-prefill effect after clean state restoration. Final-state swaps measure a state's effect with the KV cache fixed; tracing prefill interventions reveals how recurrent computation contributes through that cache.

6 Conclusion

Tracing recurrent-block contributions across the prefill/query boundary leads to two main conclusions.

Recurrent computation can affect answers through the KV cache. Across five hybrid models, the modified KV cache preserves most of the induced change in answer preference after the original recurrent state is restored. Layerwise tests locate this effect in deep attention layers. The pathway also supports arithmetic answers and, in two checkpoints, complete answers from paired SQuAD passages. These results support a *compute-commit-retrieve* pattern. Final-state swaps leave earlier recurrent contributions to the KV cache unchanged. Weak final-state effects can, therefore, coexist with strong prefill effects.

Decodable state information need not change answer preference. Probes recover individual labels from recurrent state even when state swaps leave the preferred answer unchanged. With competing facts, the probes struggle to identify which label is bound to the target entity. Probe accuracy and answer control, therefore, capture different aspects of context use.

AI Use Statement

We used OpenAI ChatGPT for manuscript drafting and editing. The authors take responsibility for the final content.

REFERENCES

Aryaman Arora, Neil Rathi, Nikil Roashan Selvam, Róbert Csordás, Dan Jurafsky, and Christopher Potts. Mechanistic evaluation of transformers and state space models. *arXiv preprint arXiv:2505.15105*, 2025.

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. Zoology: Measuring and improving recall in efficient language models. In *Proceedings of the International Conference on Learning Representations*, 2024a.

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, Dylan Zinsley, James Zou, Atri Rudra, and Christopher Ré. Simple linear attention language models balance the recall-throughput tradeoff. *arXiv preprint arXiv:2402.18668*, 2024b.

Aaron Blakeman, Aarti Basant, Abhinav Khattar, Adithya Renduchintala, Akhiad Bercovich, Aleksander Ficek, Alexis Bjorlin, Ali Taghibakhshi, Amala Sanjay Deshmukh, Ameya Sunil Mahabaleshwarkar, et al. Nemotron-H: A family of accurate and efficient hybrid Mamba-Transformer models. *arXiv preprint arXiv:2504.03624*, 2025.

Hector Borobia, Elies Seguí-Mas, and Guillermina Tormo-Carbó. Component ablation for efficient hybrid language model architectures: Performance, resilience, and compression implications. *arXiv preprint arXiv:2603.22473*, 2026.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks. In *Advances in Neural Information Processing Systems*, volume 34, pp. 9574–9586, 2021.

Atticus Geiger, Zhengxuan Wu, Christopher Potts, Thomas Icard, and Noah Goodman. Finding alignments between interpretable causal variables and distributed neural representations. In *Causal Learning and Reasoning*, pp. 160–187. PMLR, 2024.

Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. Localizing model behavior with path patching. *arXiv preprint arXiv:2304.05969*, 2023.

Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. Repeat after me: Transformers are better than state space models at copying. *arXiv preprint arXiv:2402.01032*, 2024.

Bojie Li. Models take notes at prefill: KV cache can be editable and composable. *arXiv preprint arXiv:2606.17107*, 2026.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 22947–22970, 2024.

Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, et al. Jamba: A hybrid Transformer-Mamba language model. *arXiv preprint arXiv:2403.19887*, 2024.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, volume 35, pp. 17359–17372, 2022.

NVIDIA. Nemotron 3 Nano: Open, efficient mixture-of-experts hybrid Mamba-Transformer model for agentic reasoning, 2025. URL https://arxiv.org/abs/2512.20848.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026a. URL https://qwen.ai/blog?id=qwen3.5.

Qwen Team. Qwen3.6-27B: Flagship-level coding in a 27B dense model, April 2026b. URL https://qwen.ai/blog?id=qwen3.6-27b.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, 2016.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 784–789, 2018.

Arnab Sen Sharma, David Atkinson, and David Bau. Locating and editing factual associations in Mamba. *arXiv preprint arXiv:2404.03646*, 2024.

Sankaran Vaidyanathan, David Arbour, Aaron Mueller, Scott Niekum, and David Jensen. The curse of multiple mediators: Hidden interaction effects in activation patching. *arXiv preprint arXiv:2606.27510*, 2026.

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. *arXiv preprint arXiv:2211.00593*, 2022.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. Retrieval head mechanistically explains long-context factuality. In *Proceedings of the International Conference on Learning Representations*, 2025.

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

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated Delta Networks: Improving Mamba2 with delta rule. In *Proceedings of the International Conference on Learning Representations*, 2025.

Fred Zhang and Neel Nanda. Towards best practices of activation patching in language models: Metrics and methods. In *Proceedings of the International Conference on Learning Representations*, 2024.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 34661–34710, 2023.

Jingwei Zuo, Maksim Velikanov, Ilyas Chahed, Younes Belkada, Dhia Eddine Rhayem, Guillaume Kunsch, Hakim Hacid, Hamza Yous, Brahim Farhat, Ibrahim Khadraoui, et al. Falcon-H1: A family of hybrid-head language models redefining efficiency and performance. *arXiv preprint arXiv:2507.22448*, 2025.

A Experimental protocol

The protocol specifies pair selection, uncertainty estimates, and implementation checks for the cache interventions.

Pairs and retention. Each synthetic task uses 300 candidate pairs with identical queries and token-aligned contexts. We retain a pair only when, in both clean runs, the unnormalized token score (logit) of the correct answer exceeds that of the alternative by more than $\tau = 3$. Retention is 97-100% for the reported core tasks. Qwen3.5-9B retains all exact-retrieval, instruction-following, and state-tracking pairs; 9B-Base retains 299/300, 300/300, and 300/300, respectively, and 2B retains 300/300, 300/300, and 291/300. A computed-answer parity task retains 0/300 in both base and post-trained models and is excluded. Arithmetic uses 300 retrieval, 120 prefill/re-crossing, and 123 position-control pairs. Natural-text and boundary-state selection is specified in Appendix D. Each core task uses one template with varied slot values. A five-paraphrase sweep tests sensitivity to wording (Appendix B).

Confidence intervals. Brackets in Table 1 and main-text synthetic effect estimates give 95% confidence intervals obtained by bootstrapping pairs. These intervals capture within-template variability, not variation across templates. Appendix B.1 reports pointwise intervals for Fig. 2(a,c). Error bars in Fig. 3 are 95% Wilson intervals; Appendix D.1 presents the natural-language estimation protocol.

Prompt templates. Answers are tokenizer-filtered single tokens. Exact retrieval uses `The access label assigned to {entity} is {label}.`, followed by `The access label assigned to {entity} is.` Instruction following uses `For the final question, output only the label {label}.`, followed by `What label should be returned? The label is.` State tracking starts with `Slot one holds {a}. Slot two holds {b}.`, repeats `The two slots are swapped.`, and asks for slot one's final value; the pair swaps the initial assignment. Arithmetic uses `The first number is {w1}. The second number is {w2}.`, followed by `The sum of the two numbers is`; the pair changes the first operand and neither context contains the answer. Long-context fillers use `The access label for subject {k} is {label}.` around one target fact.

Models and implementation. All language-model checkpoints remain frozen; only diagnostic readouts are trained. We use public checkpoints from the `Qwen/` namespace: Qwen3.5-0.8B, -2B, -9B, -9B-Base, Qwen3.6-27B, and Qwen3.6-35B-A3B. The NVIDIA checkpoints are `nvidia/NVIDIA-Nemotron-3-Nano-{4B,30B-A3B}-BF16`. Qwen3.5-9B has 24 recurrent and eight attention layers; Qwen3.6-27B has 64 layers with 16 attention layers; 35B-A3B has 40 layers with 10 attention layers and top-8 routing over 256 experts. Nemotron-4B has 42 layers, including four attention and 21 Mamba-2 layers; 30B has 52, including six attention, 23 Mamba-2, and 23 MoE layers. Unlike Qwen blocks, Nemotron separates recurrent/attention layers from feed-forward layers; these separate feed-forward layers are not directly replaced. In Table 3 row order, retrieval sample sizes are 300/150/60/150/148 and prefill/re-crossing sizes are 120/100/60/100/80.

All runs use one H200 and released BF16 weights, PyTorch 2.11.0+cu128, Transformers 5.17.0, SDPA attention, and flash-linear-attention 0.5.2 with native PyTorch convolution. Recurrent tensors retain native dtypes. The experiment records include run configurations, inputs, per-pair outputs, and aggregate results. Original corruption controls average three fixed seeds. Cache export/reinsertion preserves greedy continuation, with worst-case answer-position Kullback-Leibler (KL) divergence 1.6×10^{-3} nats on 9B and 0.8B. Supplemental teacher forcing checks every token of the first pair's base answer against an uninterrupted forward: Sequence discrepancies are below 0.005 nats, token-wise top-1 choices agree, and scoring preserves the input cache. Boundary checks are reported in Appendix D.2. The contribution no-op is a separate check. On 9B, its maximum answer-margin discrepancy is 0.125 logits for exact retrieval and 0.1875 for state tracking ($n = 120$ each); on 27B, the maximum is 0.125 logits in both tasks ($n = 120$ each). An earlier exact-retrieval position control used an empty-position mask instead of an empty-layer clean baseline and halved the reported effects; those rows are excluded. The depth sweep uses the correct empty-layer baseline, whereas arithmetic positions use a direct clean forward and baseline-independent flips. Re-crossing and source specificity use separate code paths.

Data and ethics. The study uses public model checkpoints, synthetic contexts, and public SQuAD passages. It conducts no human-subject experiments and accesses no private user data. The white-box interventions introduce no external cache-corruption interface.

B Essential intervention controls

These controls examine content specificity, intervention depth, and replication across tasks and checkpoints.

The KV cache has the larger effect in every pair. Across exact retrieval, instruction following, and state tracking, the minimum gaps are 8.3/13.9/8.7 logits. Signed interactions are small: 0.08 [−0.06,0.22], −0.02 [−0.17,0.14], and 0.04 [−0.08,0.16] logits, respectively. Mean absolute interactions are 0.95 [0.86,1.04], 1.07 [0.97,1.17], and 0.90 [0.84,0.96] ($n = 300$, 95% bootstrap in-

tervals). Exact-retrieval margins $(m_{AA}, m_{AB}, m_{BA}, m_{BB}) = (9.9, 8.4, -8.6, -10.0)$ show preference following the KV cache under either state.

Changing the source changes which answer is transferred. On exact retrieval, replacing recurrent-prefill contributions with source B makes y_B top-1 in 91/120 (76%) runs. A third source C makes its own answer z top-1 in 106/120 (88%) runs; equal-norm noise never makes y_B top-1. The y_B-over-y_A margins are +9.98 for B, −0.04 for C, and −0.07 for noise, whereas C results in z-over-$y_A = +9.67$. These are the 120 main 9B runs, excluding three pilot runs.

State interpolation preserves answer preference. Interpolating $S_\alpha = (1-\alpha)S_A + \alpha S_B$ at $\alpha = 0, .25, .5, .75, 1$ changes margins smoothly without a flip ($n = 120$). Real source states also disturb an unrelated suffix much less than equal-norm Gaussian noise: At $\alpha = 1$, neutral KL divergence is 0.053/0.052 nats for exact retrieval/state tracking, versus 7.63/6.35 for noise.

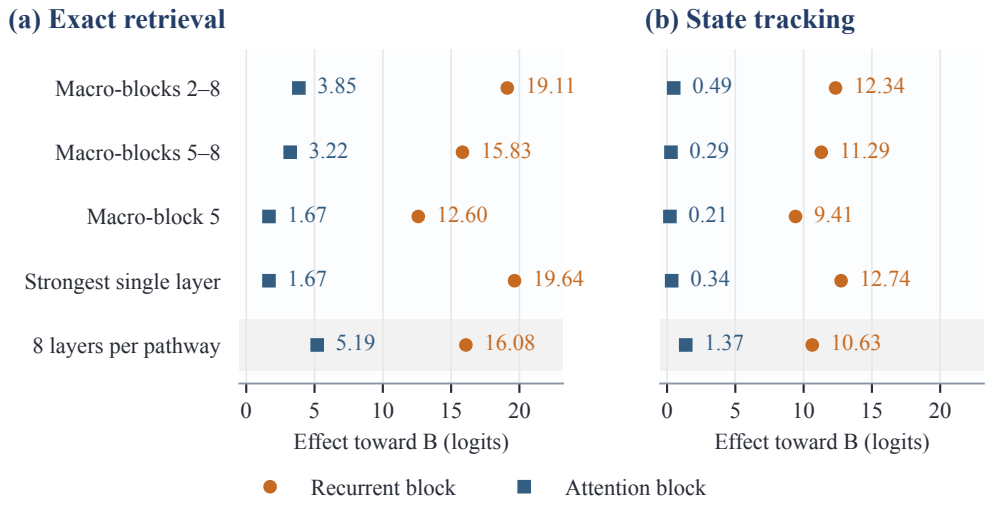

Figure 4: Depth- and count-matched prefill controls on Qwen3.5-9B ($n = 120$ per task). **(a)** Exact retrieval. **(b)** State tracking. Points show mean effects toward B; rows compare matched depths, strongest single layers, and eight-layer interventions. Layer selection is described in the text.

Later recurrent blocks retain the effect. Leaving the first macro-block clean preserves every recurrent-prefill reversal in both tasks; the effects are 19.11 [18.61,19.60] and 12.34 [12.00,12.70] logits for exact retrieval/state tracking, versus 0% reversals for attention interventions. Replacing only Recurrent Blocks 5 through 8 still results in 15.8 logits and 99% reversals on exact retrieval. Eight-layer recurrent subsets result in 16.1 [15.7,16.5] logits. The effect thus persists beyond the earliest blocks and after matching layer counts (Fig. 4).

Fig. 4 first matches macro-block depths, with different layer counts across pathways, then compares one or eight layers per pathway. The strongest single layer is Recurrent Layer 1 in both tasks and Attention Layer 20/24 in exact retrieval/state tracking (one-based). Maxima are taken over 24 recurrent and eight attention layers. The eight-layer recurrent point averages five random eight-of-24 subsets; its attention counterpart replaces all eight attention blocks. These are descriptive summaries.

Both the recurrent mixer and its FFN transfer the effect. Replacing all recurrent-block contributions results in 19.9 logits toward B on exact retrieval, versus at most 5.2 for attention blocks. Component-level tests ($n = 90$) show contributions from both sublayers: Recurrent mixer/FFN effects are 19.2/19.8 logits on exact retrieval and 12.5/12.5 on state tracking. The corresponding attention-block effects are 0.5/4.2 and −1.0/1.5. The traced effect, therefore, spans the recurrent block, rather than isolating its mixer.

Answer control spans layers and token positions. At query time, swapping the deepest two of eight attention layers flips 81/30% of exact-retrieval/state-tracking pairs; swapping the deepest four flips every pair ($n = 120$). Increasing token coverage also raises transfer: Random replacement at fractions .10/.25/.50/.75 flips 12/22/49/79% of exact-retrieval pairs and 16/28/54/78% of state-tracking pairs, averaging three draws per pair. Both sweeps show that the effect is distributed across KV entries.

Restoring state leaves the induced preference intact. In exact retrieval, (KV^*, S_A) retains a source-directed margin of −8.6 [−9.0,−8.1], whereas (KV_A, S^*) returns it to +8.5 [8.1,8.9]. Positive margins favor A. In Table 2 row order, mean margins are $(+9.9, -10.0, -8.6, +8.5, -0.3, +0.3)$ for exact retrieval and $(+6.2, -6.4, -7.0, +6.7, -2.6, +2.4)$ for state tracking. Block 7 was selected because it has the largest single-layer KV-cache effect in exact retrieval: 9.58 logits versus 2.59 logits for Block 8.

Arithmetic preserves the same pathway. For answers computed from two operands, KV-cache/state swaps flip 100/0% of pairs ($E_{\mathrm{KV}} = 10.8$, $E_S = 0.4$, $O = 0.93$). Recurrent-prefill intervention gives an 11.2-logit effect and 100% flips, versus at most 0.9 logits for attention blocks. The modified KV cache or state alone preserves source preference in 100/0% of pairs. Operand-only, post-operand-only, and all-position recurrent interventions flip 67/53/100%, tracing transmission beyond operand tokens without locating the addition operation.

The contrast also appears before post-training. On 9B-Base, KV-cache/state swaps flip 100/0% of pairs across the three core tasks, with $O = 0.82/0.76/0.93$ for exact retrieval/instruction following/state tracking. Recurrent prefill changes the exact-retrieval margin from +9.9 to −9.9; the modified KV cache or state alone preserves source preference in 99/1% of pairs. Qwen3.5-2B also gives 100/0% query-time flips on all three tasks, and the 0.8B pilot points to the same direction.

B.1 Cumulative-depth estimates and architecture variation

The depth sweeps locate the retained effect, whereas architecture and prompt controls examine variation in its strength.

Most recovery occurs in deep KV caches. On Qwen3.6-27B, modified KV entries from the first 12 attention layers recover at most .01 of the recurrent-prefill effect. Adding the final four layers raises recovery to .16/.29/.73/.93. Both MoE curves also rise late, ending at .70 and .66 (Fig. 5). The models share a late rise in recovery, with different final recovery levels.

In Figs. 2b and 5, modified KV entries are installed in the first d attention layers, keeping the remaining entries and recurrent state clean. Points average per-pair recovery, $[m_A - m_d]/[m_A - m^*]$; C_{KV} in Table 3 instead divides mean effects. Fig. 5 has $N_{\mathrm{attn}} = 16, 10, 6$ layers across its panels. Its Nemotron-30B sweep uses 100 pairs, versus 80 for that table's re-crossing experiment.

Tables 4 and 5 give mean effects and pointwise 95% percentile-bootstrap intervals for all 16 cumulative depths in Fig. 2(a,c), in logits toward B. Each intervention includes the first d macro-blocks; hence, these curves do not measure isolated block contributions. Each task uses 120 pairs; intervals resample pairs for 10,000 draws (seed 0), conditional on the checkpoint and template. They are pointwise intervals, not simultaneous confidence bands. Fig. 2b uses an earlier 100-pair commitment run.

Residual state effects vary across architectures. In Table 3 row order, retrieval $(E_{\mathrm{KV}}, E_S, O)$ values are $(18.5, 1.4, .85)$, $(20.0, 1.5, .86)$, $(19.2, 2.8, .74)$, $(15.5, 6.8, .39)$, and $(11.5, 4.7, .41)$. Nemotron-30B also retains a state-only recovery ratio, $R_S = .25$. Attention-layer fraction alone does not explain the ordering: Both Qwen models use one in four, while Nemotron MoE has a larger fraction than its dense counterpart (11.5% versus 9.5%). Training recipe and active capacity remain entangled with FFN type.

Routing changes little when clean accuracy is preserved. Within 35B-A3B, router top-k $= 8/32/256$ gives similar $O = .40/.37/.41$ and $C_{\mathrm{KV}} = .69/.68/.70$, with full baseline retention ($n \leq 50$). Top-1 routing retains only 12%, hence, this setting does not support a comparable estimate.

Table 4: Exact retrieval: Fig. 2(a,c) means and pointwise 95% pair-bootstrap intervals at every cumulative depth ($n = 120$).

	(a) Prefill contribution		(c) Query-time memory	
d	Recurrent	Attention	KV	State
1	21.60 [21.25,21.95]	0.33 [0.26,0.40]	0.01 [−0.01,0.02]	−0.03 [−0.05,−0.02]
2	21.62 [21.26,21.97]	0.61 [0.52,0.71]	0.01 [−0.01,0.03]	−0.05 [−0.10,−0.01]
3	21.62 [21.26,21.97]	0.73 [0.62,0.83]	0.01 [−0.01,0.03]	−0.07 [−0.12,−0.01]
4	21.62 [21.26,21.97]	0.75 [0.65,0.86]	0.01 [−0.01,0.04]	−0.07 [−0.13,−0.01]
5	21.62 [21.26,21.97]	1.76 [1.57,1.96]	0.01 [−0.02,0.04]	−0.08 [−0.14,−0.02]
6	21.61 [21.25,21.96]	1.75 [1.56,1.95]	0.02 [−0.02,0.05]	−0.07 [−0.14,−0.01]
7	21.62 [21.26,21.97]	1.76 [1.57,1.96]	0.01 [−0.02,0.04]	−0.06 [−0.13,0.01]
8	21.62 [21.26,21.97]	1.76 [1.57,1.96]	0.02 [−0.01,0.05]	−0.04 [−0.12,0.03]
9	21.63 [21.27,21.98]	1.97 [1.75,2.19]	0.04 [0.00,0.07]	−0.04 [−0.12,0.04]
10	21.62 [21.26,21.97]	2.01 [1.80,2.23]	0.05 [0.02,0.09]	−0.04 [−0.12,0.04]
11	21.62 [21.27,21.97]	1.99 [1.78,2.20]	0.08 [0.04,0.11]	−0.03 [−0.11,0.05]
12	21.62 [21.26,21.97]	2.01 [1.81,2.23]	0.11 [0.07,0.16]	−0.01 [−0.09,0.07]
13	21.62 [21.27,21.98]	5.43 [5.09,5.79]	3.45 [3.26,3.65]	0.13 [0.04,0.23]
14	21.62 [21.27,21.97]	6.73 [6.41,7.07]	6.25 [6.02,6.49]	0.28 [0.19,0.38]
15	21.62 [21.26,21.97]	6.91 [6.58,7.24]	15.81 [15.47,16.15]	0.82 [0.72,0.93]
16	21.62 [21.26,21.97]	6.91 [6.58,7.24]	20.05 [19.69,20.42]	1.51 [1.41,1.61]

Table 5: State tracking: Fig. 2(a,c) means and pointwise 95% pair-bootstrap intervals at every cumulative depth ($n = 120$).

	(a) Prefill contribution		(c) Query-time memory	
d	Recurrent	Attention	KV	State
1	10.66 [10.49,10.84]	−0.04 [−0.17,0.09]	0.00 [−0.01,0.01]	0.00 [−0.01,0.01]
2	10.63 [10.45,10.79]	0.02 [−0.11,0.15]	0.00 [−0.01,0.01]	0.00 [−0.01,0.01]
3	10.62 [10.45,10.79]	0.08 [−0.06,0.21]	0.00 [−0.01,0.02]	0.00 [−0.02,0.01]
4	10.63 [10.45,10.80]	0.12 [−0.02,0.26]	0.03 [0.02,0.04]	0.00 [−0.01,0.02]
5	10.63 [10.46,10.79]	−0.11 [−0.34,0.11]	0.02 [0.01,0.03]	0.01 [−0.01,0.02]
6	10.65 [10.48,10.82]	−0.12 [−0.35,0.10]	0.04 [0.02,0.06]	0.02 [0.01,0.04]
7	10.64 [10.47,10.80]	−0.03 [−0.26,0.18]	0.02 [0.00,0.03]	0.01 [−0.01,0.02]
8	10.64 [10.47,10.80]	0.01 [−0.21,0.23]	0.02 [0.00,0.04]	0.02 [0.00,0.05]
9	10.64 [10.47,10.81]	0.01 [−0.23,0.23]	0.04 [0.02,0.06]	0.04 [−0.02,0.09]
10	10.66 [10.48,10.82]	0.02 [−0.22,0.25]	0.02 [0.00,0.04]	0.03 [−0.03,0.09]
11	10.66 [10.48,10.82]	0.17 [−0.07,0.40]	0.03 [0.00,0.05]	0.05 [−0.02,0.11]
12	10.64 [10.47,10.81]	0.21 [−0.02,0.43]	0.07 [0.03,0.10]	0.12 [0.00,0.23]
13	10.66 [10.49,10.82]	0.59 [0.22,0.95]	2.71 [2.56,2.85]	0.23 [0.08,0.38]
14	10.65 [10.48,10.81]	1.04 [0.72,1.36]	7.25 [7.03,7.48]	−0.04 [−0.19,0.12]
15	10.65 [10.48,10.81]	0.87 [0.56,1.18]	9.95 [9.75,10.16]	0.03 [−0.13,0.18]
16	10.66 [10.49,10.82]	0.87 [0.56,1.18]	11.07 [10.85,11.31]	−0.19 [−0.34,−0.03]

Paraphrases preserve the KV-cache advantage. Across five retrieval paraphrases ($n = 60$ each, full retention), The 27B model maintains 100/0% KV-cache/state flips and $O = .82$-$.89$. The 35B-A3B model results in 93-100/0-5% flips and $O = .34$-$.52$. Wording changes the effect size, whereas the KV cache retains the larger influence.

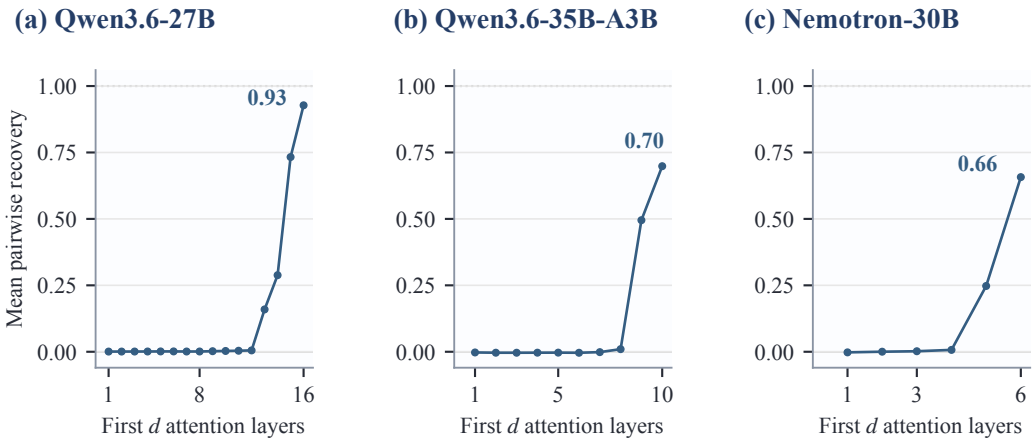

Figure 5: Cumulative KV-cache recovery across three checkpoints ($n = 100$ exact-retrieval pairs each). Points average per-pair recovery after installing modified entries in the first d attention layers, keeping other memory clean. Panel (a) repeats Fig. 2b.

C Representations and binding load

The probes and context controls distinguish isolated-label decoding from binding under competing facts and longer contexts.

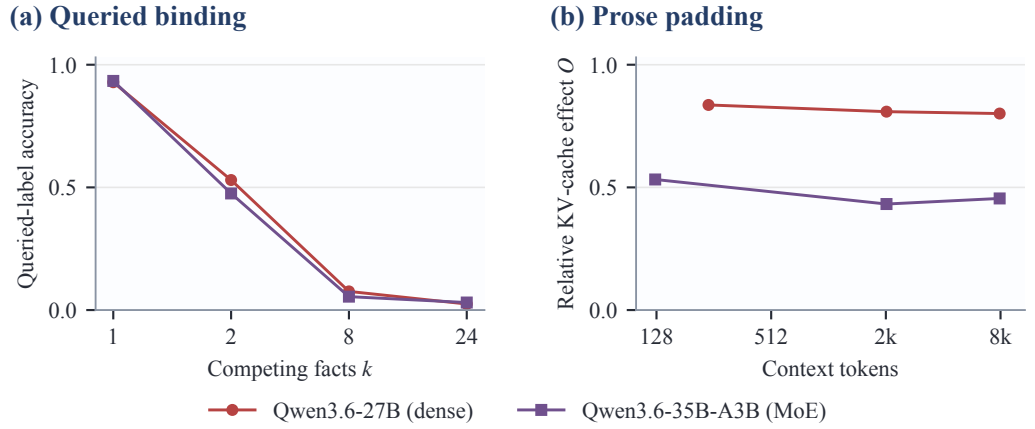

Figure 6: Binding load and context length on Qwen3.6-27B and Qwen3.6-35B-A3B. **(a)** Queried-label probe accuracy versus competing facts ($n = 1500$ per k; chance .02). **(b)** Relative KV-cache effect O under unrelated prose padding ($n = 40$ per model and length). Colors and markers identify models consistently.

Isolated labels generalize to held-out entities. Dense/MoE accuracy is 1.000/1.000, versus shuffled-label 0.067/0.062 (chance .077). With 50 labels and $n = 2000$, both reach .999. The isolated-label probe uses recurrent matrix state, excluding the convolutional window, and all 40×13 entity-label combinations ($n = 520$). Five folds hold out entities; centering, 256-component PCA, standardization, and L_2-regularized multinomial logistic regression are fit on the training fold only.

Competing facts impair binding. The binding task fixes the target entity, randomizes its position among k competitors ($k + 1$ bindings per context), and uses 50 labels with 30 examples per class ($n = 1500$ per k). Both probes use post-prefill matrix state from zero-based layer 48 for 27B and

29 for 35B. Binding folds assign generated sample i to $i \bmod 5$; preprocessing is fit within each training fold. The linear probe uses Adam for 500 steps, learning rate 0.05, and weight decay 0.01.

Table 6: Probe accuracy and state-only answer preference under binding load. Chance levels are .02 and .5, respectively; brackets give 95% bootstrap intervals. The readouts use separate protocols.

	Queried-label probe		Target vs. present label	
k	Dense 27B	MoE 35B	Dense 27B	MoE 35B
1	.929 [.92,.94]	.934 [.92,.95]	.51 [.46,.57]	.44 [.39,.50]
2	.530 [.50,.56]	.475 [.45,.50]	.48 [.42,.53]	.50 [.45,.56]
8	.076 [.06,.09]	.055 [.04,.07]	.45 [.40,.51]	.49 [.43,.55]
24	.025 [.02,.03]	.031 [.02,.04]	.54 [.48,.59]	.58 [.52,.63]

More data help at low binding load. At $k = 1$, fixed-test learning curves rise from .38/.40 at 120 training examples to .94/.92 at 1200 examples. At $k = 24$, they remain near .03/.04 and .04/.03. A two-layer neural network changes selectivity by at most .02 in a separate $n = 300$ binding control. It uses the same layer and fold-local principal componcent analysis (PCA), a 128-unit ReLU hidden layer, and 500 Adam steps with learning rate 0.01 and weight decay 0.01.

Native readouts retain some information. Against an absent label, native preference at $k = 24$ is .58 [.52,.63] for dense and .74 [.69,.79] for MoE. MoE's present-label preference .58 [.52,.63] is also above chance. The native readout zeroes the attention KV cache and measures pairwise preference against another label (present labels in Table 6), rather than full-cache generation. The table compares separate probing and native-readout protocols, not a mediation analysis. Residual information remains detectable; probe failure does not establish information-theoretic erasure.

Longer contexts preserve KV-cache control. For one fact padded with unrelated prose (Fig. 6b), dense O is 0.84/0.81/0.80, versus 0.53/0.43/0.45 for MoE ($n = 40$ per point). Mean token counts are 240/2064/8106 for dense and 126/2064/8106 for MoE. A separate 9B filler-fact sweep also preserves 100/0% KV-cache/state flips ($n = 30$ pairs, approximately 70/520/2050/8200 tokens). Across this sweep, O rises from .89 to .99, whereas the state effect falls from .86 logits to approximately zero.

Competing facts reduce state recovery more than unrelated prose. For MoE 35B, adding same-schema facts reduces R_S from .28 to .07 and .03, with state-only preference near chance by 2k tokens ($n = 60$). Unrelated prose around one fact instead preserves $R_S = .27$ and preference 1.00 at 8k tokens. The type of added context, therefore, matters alongside its length.

Fixed-length controls retain a residual state effect. MoE R_S stays within .18-.23, versus about .04 for dense. Every context has 24 sentences (about 99 tokens), fixed target position and query distance, and 0/1/4/8/16/23 access-label distractors; remaining sentences use a length-matched display-color relation. MoE retention falls from about .9 to .5 across the sweep, which can obscure decay. These comparisons separate binding load from raw token count but do not isolate a unique interference mechanism. Too few Nemotron-30B long-context pairs pass the clean filter for a stable curve.

D Natural-language replication and the state-transfer boundary

Paired SQuAD tests track the prefill effect through complete-answer generation, whereas neutral-suffix tests examine transfer from an intermediate recurrent state.

D.1 Complete-answer replication

The modified KV cache preserves nearly all source answers on the common clean-correct subset, whereas the modified state alone produces none (Table 7).

Table 7: Complete source-answer EM on the common clean-correct SQuAD subset ($n = 162$ of 200 pairs). Values are percentages with 95% Wilson intervals. Restorations follow recurrent prefill.

	Source-answer EM (%)		
Operation	Qwen3.5-9B	Qwen3.6-27B	Interpretation
Recurrent prefill	100.0 [97.7,100.0]	100.0 [97.7,100.0]	Source answer produced
Restore recurrent state	**98.8** [95.6,99.7]	**98.1** [94.7,99.4]	Source answer retained
Restore KV cache	**0.0** [0.0,2.3]	**0.0** [0.0,2.3]	No source answer
Attention prefill	1.2 [0.3,4.4]	3.1 [1.3,7.0]	Rare source answers

Matched passages isolate answer replacement. Each counterfactual changes one 2-5-token proper name while preserving the question and all tokens outside the span. Replacement names match entity type and preserve answer and context token lengths in both tokenizers. The frozen pool contains 200 answerable SQuAD v2 training questions from distinct paragraphs of at most 384 tokens before the 9B chat wrapper. The pool contains 185 people (92.5%), seven cities, five companies, two universities/colleges, and one country; 57/79/46/18 answers have 2/3/4/5 tokens. This task measures complete-answer copying from controlled SQuAD passages. NER, question/type checks, and AI-assistant review screen inputs; this is not human benchmark annotation. An erroneous NER-only pilot pool is excluded, five further type errors are replaced before the main runs, and 30 disjoint protocol pilots are excluded. No intervention outcome determines inclusion.

Clean answers define the comparison subsets. Both models receive the same pool. Within-model clean EM selects 168 pairs for 9B and 170 for 27B; their intersection is the 162 pairs in Table 7, without a margin threshold. On the model-specific subsets, recurrent-prefill, modified-KV-cache, modified-state, and attention-prefill source-answer EM are, respectively, 100/100, 97.6/98.2, 0/0, and 1.2/2.9% (9B/27B). Table 8 retains all 200 pairs, including clean failures. Baseline errors include honorifics, surviving aliases elsewhere in the paragraph, and ambiguous source questions. The two checkpoints provide replication rather than a scale-only comparison.

Generation and probability scores measure different outcomes. Greedy answers measure complete-answer transfer; teacher-forced scores also capture changes that leave the answer unchanged. We use native chat templates with thinking disabled and an `Answer:` continuation. Greedy generation stops at newline/EOS or 24 tokens. EM and token F1 use SQuAD normalization. Teacher forcing scores every token, including the leading-space encoding:

$$g(C) = \log P(y_B \mid C, q) - \log P(y_A \mid C, q), \qquad \log P(y \mid C, q) = \sum_t \log P(y_t \mid C, q, y_{<t}).$$

Here, C_{AA} is the clean base cache. Positive g favors B, opposite to m; $\Delta g = g(C) - g(C_{AA})$ is in nats and is not pooled with the single-token metrics. Proportion intervals, including the error bars in Fig. 3, use Wilson 95% intervals; continuous effects use pair-level percentile bootstrap with 2000 draws and seed 19. The eight conditions use complete recurrent states and residual block contributions, including feed-forward computation, as in Section 3.

The modified KV cache retains most answer content. Full-pool source F1 remains close to clean-source performance: 90.0/90.3% with the modified KV cache, versus 90.4/91.5% on clean source contexts.

State can shift probabilities without producing source answers. On the common clean subset, restoring state reduces the recurrent-prefill sequence effect by 1.03 [.86,1.19] nats for 9B and .58 [.41,.75] for 27B. Modified state alone shifts g by 2.21 [2.01,2.43] and 2.28 [2.05,2.52] nats despite zero source-answer EM; base EM is 100.0/99.4%. Thus, some answers match neither span. Stable greedy answers can coexist with probability changes (Appendix E.1).

Audited count questions reproduce the pathway. The same pathway appears in a separate 100-pair human-audited SQuAD set, where each pair changes one single-token number word per unique

Table 8: Complete-answer results on all 200 SQuAD pairs. EM_A/EM_B: base/source exact match (%). Δg: sequence-margin change (nats). Swaps start from clean A; restorations follow recurrent prefill.

	Qwen3.5-9B		Qwen3.6-27B		
Operation	EM_A/EM_B	Δg	EM_A/EM_B	Δg	Interpretation
Clean base A	92.0/0.0	0.00	89.0/0.0	0.00	Base reference
Clean source B	0.0/85.0	37.41	0.0/87.5	34.37	Source reference
Swap recurrent state	90.0/0.0	2.12	89.0/0.0	2.28	Base answers retained
Swap KV cache	0.0/84.0	36.39	0.0/84.5	33.80	Source answers transferred
Recurrent prefill	0.0/85.0	37.40	0.0/87.5	34.36	Source answers transferred
Restore recurrent state	0.0/**84.0**	36.37	0.0/**85.0**	33.80	Source answers retained
Restore KV cache	90.0/**0.0**	2.21	89.5/**0.0**	2.28	No source answers
Attention prefill	75.0/1.0	10.10	56.0/2.5	11.98	Few source answers

sentence. Screening removes other numeric evidence, enumeration cues, and agreement risks; both clean answers must be global top-1 with $|m| > 1$. On 9B, $E_{\text{KV}} = 17.99$, $E_S = .68$, $O = .93$, and retrieval, prefill, and re-crossing each give 100/0% KV-cache/state, recurrent/attention, and modified-KV-cache/modified-state outcomes, respectively.

D.2 Intermediate state into new suffix KV entries

Installing new suffix KV entries after an intermediate-state swap preserves the base answer in every tested pair (Table 9).

The protocol isolates newly written KV entries. We use 120 fictional-register pairs per model, excluding six protocol pilots. Prefixes A/B differ in the queried access word. A control prefix C changes only an unrelated office location in A, preserving the answer, as illustrated below.

Example: answer-changing and same-answer prefixes

Shared query: What is the access word for cabinet Falcon?

A (base): The office is in the north wing. The access word for cabinet Falcon is **blue**.

B (different answer): The office is in the north wing. The access word for cabinet Falcon is **silver**.

C (same answer): The office is in the **east** wing. The access word for cabinet Falcon is **blue**.

B changes the queried access word; C changes only the unrelated office location. *Shared instructions omitted.*

We retain A's prefix KV entries, install B's complete recurrent state, and process an answer-free suffix of $L = 0/16/64$ tokens. To isolate new KV entries, we install only the modified suffix KV entries into the clean A post-suffix cache, preserving its prefix entries and final recurrent state. Controls install the full modified cache, final state alone, same-answer suffix entries, or restored clean suffix entries.

New KV entries preserve the base answer. All clean answers are correct at every length. Every suffix-KV intervention retains the base answer: Source-answer EM is 0/120 (Wilson 95% interval [0,3.1]%), with effects near zero (Table 9). The $L = 0$ suffix swap is an exact no-op. At nonzero lengths, the intervals for source-state effects minus same-answer control effects include zero except 9B at $L = 64$, whose contrast is negative: −.015 [−.026,−.003] nats.

Final state changes probabilities without changing answers. Final-state transfer retains positive probability effects but results in zero source-answer EM. Full modified-cache effects differ from the displayed final-state means by at most .01 nats; at $L = 0$ both are 1.61 [1.50,1.73] for 9B and 1.23 [1.15,1.30] for 27B.

Table 9: Intermediate-state transfer ($n = 120$ per model and length). Values give paired Δg in nats with 95% bootstrap intervals.

Model	L	Suffix KV only	Final state only
Qwen3.5-9B	16	−.004 [−.017,.009]	1.61 [1.50,1.73]
	64	.005 [−.008,.018]	1.54 [1.44,1.65]
Qwen3.6-27B	16	.001 [−.012,.015]	1.16 [1.08,1.23]
	64	−.003 [−.015,.011]	1.06 [1.00,1.13]

Numerical precision limits the small suffix effects. Split and uninterrupted prefills differ by at most about .25 nats, within the declared .5-nat tolerance but larger than the signed suffix effects. These effects, therefore, do not establish fine-grained transfer in BF16. Tensor checks verify unchanged prefix KV entries, restored final state, and exact clean-cache recovery after restoring suffix entries; repeated restored-cache scoring agrees within 10^{-4} nats. The unchanged answers concern this fixed-prefix, neutral-suffix protocol; the main prefill intervention traces effects through block contributions on the original context.

E Mathematical interpretation

The following arguments explain how strong prefill effects, decodable state information, and weak answer control can coexist.

E.1 Transmission paths and readout directions

Recurrent-block contributions can affect an answer through both the KV cache and final state.

Local path decomposition. Fix the context, query, weights, and intervened blocks. Let u denote the replaced recurrent-block contribution and $K(u), S(u)$ the recomputed caches. Assume differentiability at the reference run, with fixed expert selection in routed models. Writing $P = \partial K/\partial u$, $Q = \partial S/\partial u$, $a = \nabla_K m$, and $b = \nabla_S m$, the chain rule gives Eq. (1) and

$$\delta m = (a^\top P + b^\top Q)\delta u + o(\|\delta u\|).$$

Holding the KV cache fixed removes the $a^\top P$ route. A small final-state effect, therefore, need not imply a small total effect of recurrent computation.

Why a probe need not predict control or later writes. A linear probe $p^\top S$, native answer sensitivity b, and suffix-write map $W_L = \partial K_{\text{suffix}}/\partial S_{\text{boundary}}$ can admit a state direction d with

$$p^\top d \neq 0, \qquad b^\top d \approx 0, \qquad W_L d = 0.$$

Decodability thus requires neither substantial native answer sensitivity nor transfer to later KV entries.

Exact finite construction. For $u_A = +1$, $u_B = -1$, and $0 < \varepsilon < 1$, let

$$K(u) = u, \qquad S(u) = (u, 0), \qquad m(K, S) = K + \varepsilon S_1.$$

The sign of m determines the answer. Replacing u_A with u_B during prefill changes the margin from $1+\varepsilon$ to $-1-\varepsilon$. Modified KV alone results in $-1+\varepsilon$ and also flips the answer; modified state alone gives $1-\varepsilon$, preserving the answer while changing the margin by 2ε. The probe S_1 perfectly decodes the label for every ε, even as answer sensitivity to state vanishes when $\varepsilon \to 0$. A state-preserving suffix that writes $v_\ell = c_\ell + S_2$, with shared constants c_ℓ, also leaves later KV entries unchanged by the state swap. The recovery ratios are

$$C_{\text{KV}} = \frac{1}{1+\varepsilon}, \qquad R_S = \frac{\varepsilon}{1+\varepsilon}.$$

E.2 Interaction correction

Interactions between the two memories determine whether their recovery ratios sum to one.

Proposition 1 (Interaction correction). Let $m_{00}, m_{10}, m_{01}, m_{11}$ denote the margins for (K_A, S_A), (K^*, S_A), (K_A, S^*), and (K^*, S^*), respectively. For $D = \mathbb{E}[m_{00} - m_{11}] \neq 0$ and $J = \mathbb{E}[m_{00} - m_{10} - m_{01} + m_{11}]$,

$$C_{\mathrm{KV}} + R_S = 1 + \frac{J}{D}.$$

Proof. The numerators add to $\mathbb{E}[2m_{00} - m_{10} - m_{01}] = D + J$. Dividing by D results in the identity. □

The ratios sum to one exactly when mean interaction vanishes and need not lie in $[0, 1]$. The query-time I uses two unmodified contexts, whereas J uses the prefill-modified cache; they are separate contrasts.