WHERE DOES ENTITY BINDING COME FROM?

Anonymous authors
Paper under double-blind review

ABSTRACT

To use factual knowledge, e.g., "Alice has the umbrella," a language model must do more than represent the entities: Alice and the umbrella. It must *bind* them, tracking which entity goes with which. Prior work showed that binding appears in model activations as identity vectors attached to each entity. But where does this binding come from: is it carried in the content of the representations, or computed from where they sit? We first show that, at early layers, binding is constructed from receiving position: placing the same entity content at different token locations predictably changes which entities are bound together. As computation proceeds, however, moved representations increasingly preserve their original pairing, showing that the established relation becomes carried by the representation itself. We then ask what this mechanism implies for injected knowledge. Writing entity states separately at their corresponding positions supports multi-hop reasoning close to ordinary text, whereas pooling those states sharply degrades relational use. By retaining all receiving positions while mixing entity Content, we show that position alone is not sufficient: two-hop reasoning can fail even while direct factual recall remains available.

1 INTRODUCTION

A language model is told two facts: "Alice has the umbrella" and "Ben has the bottle." Asked who has the bottle, it answers correctly when these facts are given as text. But when the same facts are written into the model's activations as vectors, the model may still recall all four entities while answering "Alice." The model failed at *binding*: keeping track of which entity belongs with which.

Prior work locates binding in entity identity vectors within activations (Feng & Steinhardt, 2024), with positional variants in low-rank subspaces (Dai et al., 2024). Other studies identify circuits that read and manipulate these representations (Prakash et al., 2024; Oh & Demberg, 2026) and routes for retrieval (Gur-Arieh et al., 2025) and readout (Feng et al., 2025). These studies observe binding after the model has *read* the facts, when entity content and position are already combined. We ask whether binding comes from entity content or from the positions those representations occupy.

To separate what written states carry from what the receiving context supplies, we use *activation writing*, building on the interventions of Feng & Steinhardt (2024): placing vectors in a frozen model while varying their content, receiving positions, and the layer where computation resumes. We remove the source-position differences from the written vectors while retaining receiving positions and relation words. If pairing still forms, the receiving context can establish it without those differences in the written vectors.

This matters for methods that compress external text into context vectors or soft tokens supplied to frozen models (Mu et al., 2023; Cheng et al., 2024; Rau et al., 2024). Combining facts requires using their relations, beyond recalling entities; yet edited or fine-tuned facts do not always support multi-hop answers (Zhong et al., 2023; Cohen et al., 2024; Balesni et al., 2024). What must the written representations retain for reasoning?

Contributions. *First*, we decompose states into estimated components associated with their original positions and entity content, then exchange either or both across layers. At tested early layers, pairing mainly follows the content component's new receiving position even when both components move together; at middle layers, moving both components better preserves the original pairing than moving either alone (§4.1). Early writes retain most pairing performance when the position-related

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.

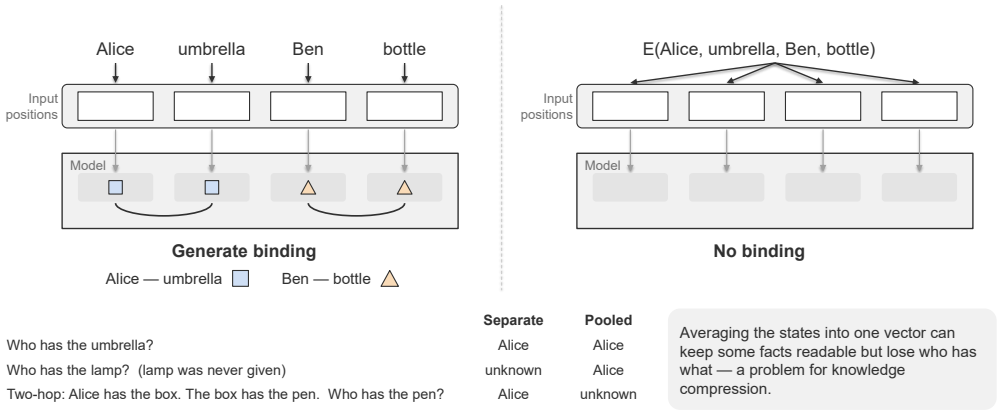

Figure 1: **Conceptual comparison of separate and pooled states.** Left: distinct entity states occupy four positions. Right: one pooled vector is supplied at four positions. The example answers and "No binding" label illustrate a possible failure; they are not measured results or an internal diagnosis. Section 5 tests how Content mixing changes two-hop answers while all receiving positions remain.

component is shared between names and, separately, between objects (§4.2), showing that these source-position differences need not be supplied for pairing to form.

Second, writing two facts separately at their corresponding entity positions supports two-hop answers close to text on both synthetic and Wikidata tasks (§5.1; Table 3). This establishes a working reference for testing whether changes to written states disrupt their joint use.

Third, in the synthetic task, pooling reduces two-hop accuracy but changes both position count and state composition. Holding all positions and original states fixed, we instead add the other entity's content component: two-hop accuracy falls even where recall stays near its unmixed level, and equal-norm random additions do not fully reproduce the decline (§5.2). Leaving only the content component of the entity named in the question unmixed improves two-hop accuracy despite mixing elsewhere (§5.3).

2 Measuring where binding entities come from

In this section, we introduce our measurement instrument, *activation writing*, built on the interventions of Feng & Steinhardt (2024), and the binding tests and controls used throughout.

Binding task. We adopt the binding task of Feng & Steinhardt (2024), where binding is represented as *binding ID* vectors attached to each entity's and attribute's activations. Here, the task is defined by a set of entities $\mathcal{E}$ and a set of attributes $\mathcal{A}$. $\mathcal{E}$ is a set of single-token names (*Alice*, *Ben*, ...) and $\mathcal{A}$ a set of single-token objects (*umbrella*, *broom*, ...), paired by possession. An n-entity instance is a context pairing each entity with one attribute, written $c = \mathrm{ctxt}(e_0 \leftrightarrow a_0, \ldots, e_{n-1} \leftrightarrow a_{n-1})$. For instance, "Alice has the umbrella. Ben has the broom." is $\mathrm{ctxt}(\text{Alice} \leftrightarrow \text{umbrella}, \text{Ben} \leftrightarrow \text{broom})$. The model is then asked, e.g., "Who has the umbrella?", and the correct answer is the paired name, *Alice*. Answering *Ben*—the name of the other pair—is a binding error: all the entities are in the context, yet the pairing is incorrect. Our work limits $n \in \{1, 2\}$ for the binding analyses, and later in §5, we link two pairs through a shared object for the reasoning tests.

Causal intervention to LMs. We replace residual-stream states in a frozen model at specified layers and receiving positions, then let the model continue. This lets us vary the delivered state and where it lands separately, and observe whether the pairing changes. For factor writing and two-hop tests, the text condition supplies facts such as "Alice has an umbrella"; the write condition replaces the names and objects with placeholders, retains the relation words, and writes the corresponding states at those positions. For swaps tested at different layers, we retain the original input text and move states only at the tested layer. Figure 2 compares text input with activation writing.

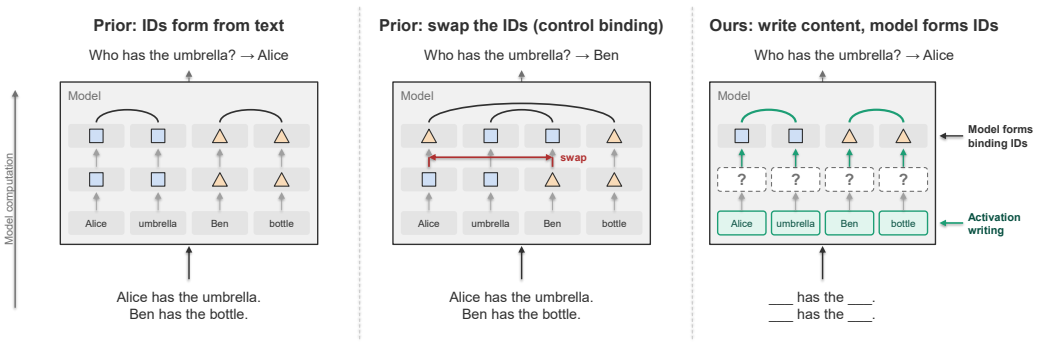

Figure 2: **Binding with text and activation writing.** Left and center: prior work examines and intervenes on identity vectors attached to entities. Right: we provide states at specified receiving positions and measure the resulting pairing from the answer.

Let h denote the state produced by the current example's text, and μ the mean state estimated from source examples. To obtain a mean state for *umbrella*, we place it at the same position in multiple source prompts, vary the other names and objects, and average its state at a specified layer. The resulting $\mu_\ell(s, v)$ identifies layer ℓ, source position s, and value v. At receiving position p, activation writing sets $h_\ell(p) \leftarrow \mu_\ell(s, v)$; the receiving position may differ from the source position. Each experiment specifies which positions are written, at which layers, and which words remain.

Slot, Content, and receiving position. To vary the position-related and content-related parts of a state separately, we assume that they combine additively. We first average μ across different entities at the same source position to estimate their shared Slot factor σ. We then subtract that position's Slot factor from an entity's μ and use the difference to estimate its Content factor Δc. Names and objects are estimated separately. Slot is a vector component of the state, whereas the receiving position is the token position where that state is placed in the model. For example, we can write an *umbrella* state estimated beside Alice at a position beside Ben: the position beside Alice is the source position, and the position beside Ben is the receiving position. Swapping Slot does not move the receiving position.

Measurements and comparisons. We use Gemma-2-9B-Instruct, Llama-3.1-8B-Instruct, and Llama-2-13B-Chat, and state which models apply in each comparison. Text provides the reference; no write tests the answer when the model receives no written state. A random control uses random vectors to test whether an effect depends on specific content. Each experiment specifies what norm is matched and how, along with its other controls.

Binding heads. Using two text facts, we also screen a set of attention heads: we replace each head's output with the mean at the same position across screening examples and measure how much the correct name's score drops relative to the other name. We call the selected set *binding heads*. These heads are used in subsequent interventions and measurements of K/V and attention. The scoring rule and head-selection method are given in Appendix A.1.

3 Recall does not mean bind

Before using model answers to study how entities are paired, we need to ask whether a correct answer means the model used the pairing. When the written content contains only one name, the model may answer "Who has this item?" by selecting that name, without distinguishing which item belongs to it.

3.1 RECALL WITHOUT FULL-FACT WRITING

For "Alice has an umbrella," answering Alice to "Who has the umbrella?" may reflect the Alice–umbrella pairing or simply selection of the only supplied name. To test what a correct answer tells us, we use four placeholders, `X X X X`. At layer 1, we write the mean states μ for Alice and umbrella at their positions, leaving the has and an positions as placeholders. Full activation writing supplies the corresponding states at all four positions; full text and no writing serve as references. We ask "Who has the umbrella?" and, separately, ask the model to recite "Alice has an umbrella." The recitation test checks whether the model can reproduce the complete fact without separate writes at the has and an positions, even if it recalls Alice in response to the question.

For the recall question, answering Alice by name or in a sentence counts; merely listing Alice and umbrella without a relation does not. For recitation, we check whether the answer begins with the complete four-word fact. Scoring details and results are in Appendix A.2.2 (Table 4).

Across the three models, recall after writing only the name and item remains close to full text, yet none recites the complete fact. Thus, the model can answer Alice even when it fails to recite the original sentence. That answer alone does not distinguish use of the Alice–umbrella pairing from selection of the only supplied name. The few failed answers are described in Appendix A.2.4.

3.2 AN UNPROVIDED ITEM CAN STILL ELICIT THE SAME NAME

The preceding result has two possible explanations: the model may use what it has learned earlier to pair Alice with umbrella, or it may select the supplied name without relying on a pairing. To investigate further, we keep Alice and umbrella but ask "Who has the bottle?" The input does not say who owns the bottle. If the model still answers Alice, we cannot rule out that it merely selected the supplied name rather than finding the person paired with the item in the question.

We use the two writing conditions from the previous subsection: writing the complete fact, and writing only the name and item. We also compare the text inputs `Alice has an umbrella` and `Alice X X umbrella` to see whether the effect occurs with ordinary text. For each condition, we either include or omit an instruction to answer unknown when the facts do not say. This tests whether reminding the model that it can say it does not know reduces these errors. Without the instruction, the model can still say it does not know.

After writing only the name and item, and without the unknown instruction, all three models attribute the unprovided item to the supplied person. With the instruction, Gemma no longer does so in this test, but Llama-3.1 and Llama-2-Chat often still do. For some models and instruction conditions, the same error occurs even with the complete fact or ordinary text input. The detailed results are in Appendix A.2 (Table 5).

Thus, when the model answers Alice to "Who has the umbrella?" that alone does not show it used the Alice–umbrella pairing: it may also answer Alice when asked about the unprovided bottle. Section 4 keeps the same names and items but changes where their states are placed, testing whether the answers follow the original pairing or the receiving positions.

4 BINDING EMERGES FROM POSITION

For the two item questions, our central finding is that early pairing follows the receiving position, whereas at middle layers a moved state can preserve the original relationship. Consider Alice–umbrella and Ben–bottle: if the umbrella state already carries information about its partner, moving it should preserve Alice–umbrella; if pairing is established from where the content is received, moving it beside Ben should change its partner. We first exchange states across layers to test these possibilities, then separate Slot and Content to examine how they help establish pairing and whether Slot needs to distinguish its source positions.

4.1 EXCHANGING SLOT AND CONTENT SEPARATELY TO TRACK CHANGES IN PAIRING

We first let the model read "Alice has an umbrella; Ben has a bottle." We then exchange the whole states h at the item positions while leaving the names and text unchanged. Before the first selected

binding head, all three models mainly switch partners: Alice–bottle and Ben–umbrella. Writing the states back in place, moving each name with its item, and unrelated-state replacement serve as controls (Appendix Table 6).

Changing Slot and Content separately. Exchanging whole states does not tell us which components of a state affect pairing. We therefore separate the question: Does changing only Slot pair an item with a different name? Does moving only Content carry the original relationship with it? If moving each factor alone differs from moving them together, we can see whether preserving their combination matters.

At each layer, we subtract the corresponding Slot from umbrella's mean states at its two source positions and average the differences to estimate its Content (Section 2); we do the same for bottle. We write the combinations below at the two item positions, leaving the name states and input text unchanged. Each test changes one layer, after which computation continues.

σ_0 and σ_1 are the Slots estimated beside Alice and Ben, respectively. Their subscripts identify the source positions; the table columns identify the positions that receive the vectors.

Operation	Item position beside Alice	Item position beside Ben
No exchange	$\sigma_0 + \Delta c(\text{umbrella})$	$\sigma_1 + \Delta c(\text{bottle})$
Slot only	$\sigma_1 + \Delta c(\text{umbrella})$	$\sigma_0 + \Delta c(\text{bottle})$
Content only	$\sigma_0 + \Delta c(\text{bottle})$	$\sigma_1 + \Delta c(\text{umbrella})$
Both factors	$\sigma_1 + \Delta c(\text{bottle})$	$\sigma_0 + \Delta c(\text{umbrella})$

With Content-only and both-factor exchange, umbrella Content moves beside Ben; only in the latter does it retain its original Slot. Comparing them tests whether that Slot preserves pairing at the same receiving position. Slot-only exchange changes Slot while leaving item Content in place.

We ask what Alice and Ben have, scoring the two answers jointly as crossed (bottle, umbrella), original (umbrella, bottle), or neither. Every layer uses the same examples and the unexchanged combination as reference. Unrelated-example and random replacements check whether these positions still affect answers; norm matching and full layer-wise results are in Appendix A.3.

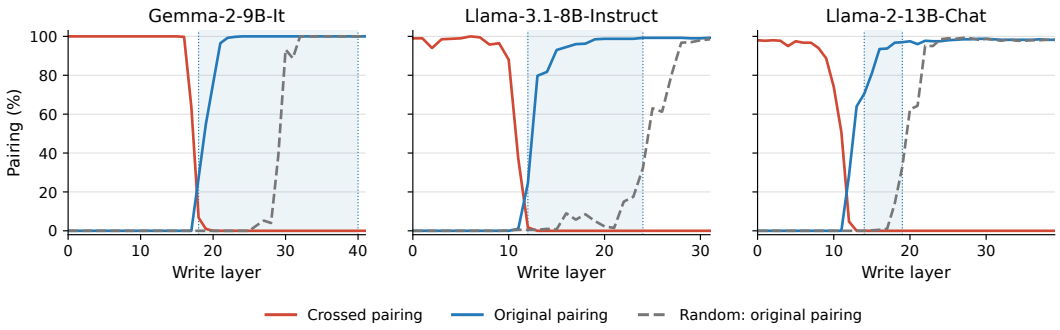

Figure 3: Layer-wise pairing after exchanging Slot and Content together.

The same exchange has different effects across layers. With both factors exchanged (Figure 3), early layers mainly pair umbrella with Ben despite its original Slot, whereas middle layers mainly preserve Alice–umbrella after the move. Thus, early pairing depends more on the receiving position, while the moved state can later preserve the relationship. The shaded range marks the selected binding heads; in Llama-2-Chat, the transition begins before the first selected head.

Table 1 separates the factors at middle layers. In Gemma, moving either factor alone mainly crosses the pairing, whereas moving both mainly preserves it: the original Slot changes the result even with umbrella at the same new position.

The two Llama models' single-factor outcomes often fall into neither category; moving both factors together mostly preserves the original pairing. Across all three models, keeping the factors together is more effective at these middle layers than moving either alone. At early layers, by contrast, Slot-only exchange mostly preserves pairing, whereas Content movement mainly changes it with or without its original Slot (Appendix A.3).

Table 1: Middle-layer factor exchanges and controls (percent). Layer ranges summarize the observed sweep; each entry averages the joint outcomes for the two item questions over the listed layers. Full early- and late-layer results are in Appendix A.3.

	Gemma			Llama-3.1			Llama-2-Chat		
Condition	Crossed	Original	Neither	Crossed	Original	Neither	Crossed	Original	Neither
Middle	19–29			13–22			14–19		
No exchange	0.0	100.0	0.0	0.0	98.6	1.5	0.0	97.9	2.1
Slot only	**71.5**	**5.5**	**23.0**	**28.5**	**38.2**	**33.3**	**19.1**	**11.8**	**69.2**
Content only	**93.4**	**0.5**	**6.1**	**37.8**	**33.7**	**28.5**	**51.2**	**2.4**	**46.4**
Both factors	**0.1**	**93.3**	**6.6**	**0.0**	**93.6**	**6.4**	**0.0**	**88.8**	**11.3**
Unrelated-example states	0.5	2.1	97.4	0.2	2.4	97.4	0.5	3.3	96.2
Norm-matched random states	0.1	4.6	95.3	0.0	4.9	95.1	0.0	8.1	91.9

Unrelated and random replacements still disrupt middle-layer answers, so these positions remain consequential. At late layers, even random replacement often preserves the answer (gray dashed line); an unchanged answer there cannot identify which factor preserves pairing.

4.2 The roles of Slot and Content in binding

Next we ask how Slot and Content establish umbrella's pairing with Alice rather than Ben. We initially hypothesized that Slot lets the model identify a state's position, while Content supplies information about the item. Slot alone might therefore let attention find the position without allowing the model to answer.

Building on the experiment in Section 4.1, we use "Alice has an umbrella. Ben has a bottle." At layers 0–8, we successively replace the states at the four name and item placeholders with vectors for those layers. This range precedes the first selected head in each model. We then measure K/V in subsequent binding heads, attention from the answer position, and the answer itself.

Text input and each position's own μ provide references. We compare Slot alone, Content alone, and both together. Slot plus same-norm random Content tests whether specific content is needed; no write and random write serve as controls.

We also ask whether Slot must retain its source-position distinction. We average the Slots for Alice and Ben so both names use the same Slot, and likewise share the average Slot for umbrella and bottle. Each word's Content, receiving position, and the relation words remain unchanged. This "shared Slot" condition tests whether the original pairing survives.

Table 2 reports R_K and R_V for K/V direction recovery (0: no improvement over no write; 1: text-reference direction), QK gain for the correct fact's attention advantage over the other fact relative to no write, and accuracy on "Who has the umbrella?" (Alice).

Table 2: K/V recovery, QK gain, and answer accuracy after factor writes (400 examples per condition).

	Gemma				Llama-3.1				Llama-2-Chat			
Condition	R_K	R_V	QK gain	Acc.	R_K	R_V	QK gain	Acc.	R_K	R_V	QK gain	Acc.
Text facts	–	–	1.595	1.000	–	–	.733	.983	–	–	.488	.998
Full μ	.995	.996	1.601	1.000	.970	.979	.731	.980	.964	.963	.500	1.000
Slot plus Content	.994	.995	1.598	1.000	.959	.972	.740	.988	.956	.954	.501	.998
Shared Slot + Content	.989	.992	1.582	1.000	.929	.952	.564	.945	.929	.930	.428	.998
Slot only	.608	.472	.001	.000	.660	.519	−.001	.000	.555	.471	-.010	.000
Content only	−1.188	−.019	.057	.033	−1.029	.027	.010	.308	-1.240	-.255	.058	.108
Slot + random Content	.583	.430	-.002	.000	.640	.476	.001	.000	.558	.456	-.019	.000
Matched-norm random	−1.757	−.524	.025	.000	−1.976	−.643	−.016	.003	-1.664	-.534	-.046	.003
No write	–	–	.000	.000	–	–	.000	.000	–	–	.000	.000

Slot alone partially recovers K/V directions (R_K around .6; R_V around .5), yet answer accuracy is zero in all three models. Content alone produces some correct answers but does not reliably support pairing. Both factors together nearly match each position's own μ across K/V, attention, and answers; same-norm random Content does not.

Our initial division of labor—Slot finds a position, Content supplies the item—is too simple: both factors are needed to recover fact selection and the pairing answer here.

Shared Slot retains most of the effect, though Llama-3.1 declines. Thus, source-position differences in Slot are not necessary for these answers.

Appendix A.4.4 further compares how changing the written item and changing Q/K affect the answer. Appendix A.4.5 tests whether stronger writes can make up for the loss of binding heads.

5 What structure supports two-hop use of written facts

Section 4 showed how receiving positions, Slot, and Content affect pairing. Its pairing question does not yet establish whether two linked written facts can be used together for a two-hop answer. We first test separate writes, then examine what happens when the states are pooled or their Content is mixed.

5.1 Writing two facts separately supports two-hop answers

We first test whether two linked facts written separately at their entity positions can support a two-hop answer. The synthetic task asks "Who has the tag?" after "Alice has an umbrella; the umbrella has a tag," with "Ben has a bottle" as a distractor. We compare both facts as text, one written fact with the other as text, and both facts written separately at their entity positions. The controls replace both facts with random writes of matching norm (Table 3). Writes occur at layers 0–8. We also test selected Wikidata film–director–country chains whose answers the models miss without supplied facts.

Table 3: Two-hop accuracy for text and activation writing. Synthetic: 400 per model; Wikidata: 237 chains for Gemma and Llama-3.1, 216 for Llama-2-Chat.

	Gemma		Llama-3.1		Llama-2-Chat	
Condition	Synthetic	Wikidata	Synthetic	Wikidata	Synthetic	Wikidata
Both facts as text	.995	1.000	.760	.764	.927	.968
Write Fact 1; Fact 2 text	.993	1.000	.770	.738	.927	.963
Write Fact 2; Fact 1 text	.993	1.000	.807	.806	.940	.972
Write both facts	.993	1.000	.823	.793	.940	.977
Random writes for both	.000	.000	.000	.025	.003	.014

Writing both facts separately supports two-hop answers close to text; random writes remain low. This provides a reference for testing whether two-hop answers survive when the written states are combined.

5.2 Content mixing reduces two-hop accuracy despite fixed positions

Having established that separate writes support two-hop answers, we ask whether the answers survive combining each fact's two states. In a separate run with the same synthetic task type, we average the two states in each fact and write the average at one position. Two-hop accuracy falls from 99.0% to 1.8% on Gemma and from 75.7% to 0% on Llama-3.1 (Appendix Table 15). Pooling changes both position count and state composition, so this comparison cannot isolate the cause. We therefore keep all four receiving positions and their original states, adding only the other entity's Content within each fact. This tests whether keeping positions and original states remains sufficient for two-hop use under Content mixing.

Section 3 showed that a correct recall answer alone does not establish use of a pairing. We therefore also ask "Who has the umbrella?" alongside the two-hop question; both have Alice as the answer.

At layers 0–8, we write into four placeholders: Alice, both umbrella occurrences, and tag. Relation words and the distractor remain text. We score against fixed candidates: Alice, Ben, umbrella, and lowercase `unknown`.

We add umbrella Content at Alice's position and Alice Content at umbrella's; the second fact is mixed likewise.

For the first fact, A and U denote Alice's and umbrella's positions; σ_A and σ_U are their Slots. $\Delta c(s, v)$ is entity v's full μ at position s minus its Slot. We write:

$$\text{Alice position}: \quad \underbrace{\sigma_A + \Delta c(A, \text{Alice})}_{\text{original state}} + g\Delta c(U, \text{umbrella}),$$

$$\text{umbrella position}: \quad \underbrace{\sigma_U + \Delta c(U, \text{umbrella})}_{\text{original state}} + g\Delta c(A, \text{Alice}).$$

The original state stays in place; g ranges from 0 to 1, where $g = 0$, 0.5, and 1 add none, half, or one full copy of the other Content. We treat the second fact likewise. Factor estimation is detailed in Appendix A.1.

In the random control, the added Content at each position is replaced by a same-norm random vector, while its original Slot and Content remain. This separates interference from another entity's Content from a perturbation of equal size. Each model uses a fixed set of 400 new examples; curves start from their own unmixed baseline.

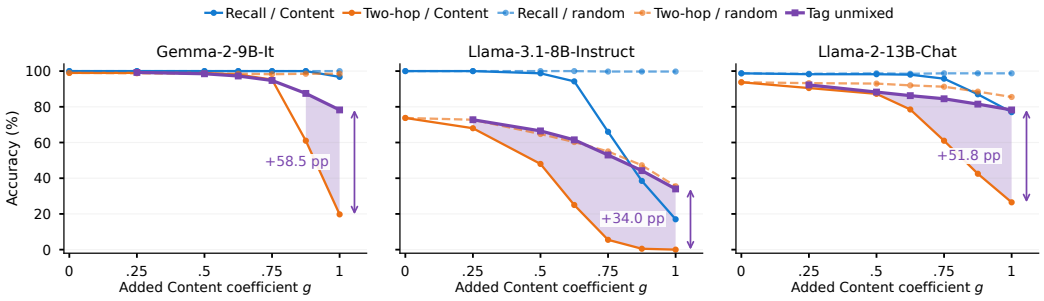

Figure 4: Recall (blue) and two-hop (orange) accuracy after adding partner Content (solid) or equal-norm random vectors (dashed); four-candidate score, 400 questions per model and condition. Purple: two-hop with all but tag mixed. Shading and arrows mark the gain.

In Figure 4, recall stays near its unmixed level over part of the range while two-hop accuracy falls. The random control largely retains two-hop performance for Gemma and Llama-2-Chat; Llama-3.1 also declines, but less than with actual Content. Thus the added entity's Content matters beyond vector size (full values: Table 23).

Wrong entities and unknown count as failures. Because Llama-2-Chat failed the original abstention control, Gemma and Llama-3.1 provide the main evidence. The results suggest that mixing impairs pairing formation or use while recall can remain available.

5.3 Keeping the Content of the entity named in the question unmixed improves two-hop answers

The preceding comparison showed that keeping the receiving positions does not prevent two-hop accuracy from falling under Content mixing. We now ask whether leaving the Content at the entity named in the question unmixed can improve two-hop answers while the other positions remain mixed. For "Who has the tag?", that entity is tag.

At $g = 1$, we compare no mixing, mixing only tag, mixing all four positions, and mixing all except tag. Each mixed position retains its original state and receives its fact partner's Content; at tag, the addition is umbrella Content. The first pair tests whether mixing tag alone hurts, and the second whether sparing tag helps despite mixing elsewhere. All other settings stay fixed.

At $g = 1$, mixing only tag sharply reduces two-hop accuracy while direct accuracy barely changes. Conversely, leaving only tag unmixed raises two-hop accuracy from 19.75% to 78.25% on Gemma, 0% to 34.00% on Llama-3.1, and 26.50% to 78.25% on Llama-2-Chat. Direct accuracy rises slightly for Gemma but falls for both Llama models. Appendix Table 24 gives all values. The purple curves in Figure 4 show the two-hop comparison across mixing amounts.

When the two facts are written separately at their corresponding entity positions, two-hop accuracy is close to that obtained when both facts are given as text. In our synthetic tests, however, adding the other entity's Content at each of the four receiving positions lowers two-hop accuracy even though every position retains its original state. Keeping only the tag position, which is directly named in the question, unmixed improves two-hop accuracy while the other positions remain mixed, but does not restore the fully unmixed performance. These results show that, beyond the receiving positions, the Content carried at each position affects whether the written facts support two-hop answers. The final answers do not identify which intermediate step is affected. Appendix A.6.4 reports a changed answer-candidate set.

6 Related Work

Entity binding. Binding-ID vectors and activation interventions show that entity–attribute relations can be represented, read, and edited in model states (Feng & Steinhardt, 2024; Dai et al., 2024; Prakash et al., 2024; Gur-Arieh et al., 2025; Oh & Demberg, 2026; Feng et al., 2025). We ask how the pairing changes when the supplied state and its receiving position are varied independently across layers.

Activation interventions. Activation Addition and Contrastive Activation Addition steer outputs with residual-stream directions (Turner et al., 2024; Panickssery et al., 2024); function and context vectors carry task or example information (Todd et al., 2024; Li et al., 2025b). We use activation writing to control the supplied state, its receiving position, and the layer where computation resumes, so that we can test which of these determines a pairing.

Compressed memory. RAG supplies retrieved text (Lewis et al., 2020). Gist Tokens compress prompts, while xRAG and Context Embeddings encode retrieved context in compact representations (Mu et al., 2023; Cheng et al., 2024; Rau et al., 2024). At higher compression ratios, DeCAL loses more on HotpotQA than TriviaQA (Panwar, 2025); a comparison of prompt-compression methods also reports the largest loss on multi-hop QA among its tested tasks (Łajewska et al., 2025). BRIEF produces query-aware textual summaries that integrate evidence for multi-hop answers (Li et al., 2025a). We hold receiving positions fixed and mix only the written Content to compare its effects on recall and two-hop answering.

7 Conclusion

We estimate a factor associated with position (Slot) and a factor associated with entity content (Content) from mean activation states, and compare how moving either or both affects pairing at different layers. At the tested early layers, pairing mainly follows Content's new receiving position even when it moves together with its original Slot; at middle layers, moving both components together preserves the original pairing better than moving either alone. Shared Slots retain most pairing performance, further showing that these source-position distinctions need not be supplied in the written states for early pairing to form.

For injected knowledge, supplying entity content is not enough: the way it is written must support entity pairing and the joint use of multiple facts. Writing two facts separately at corresponding positions supports two-hop performance close to text on synthetic and Wikidata tasks. However, controlled comparisons on the synthetic task show that even with receiving positions and original states retained, adding another entity's Content reduces two-hop accuracy even where recall remains near its unmixed level. Keeping only the Content of the entity named in the question unmixed improves two-hop answering even while other positions remain mixed. These results show that preserving positions is insufficient to maintain two-hop use of written facts and suggest selectively preserving content as a concrete direction; its benefit for practical memory systems remains to be tested.

AI USE STATEMENT

In this work, we used generative AI tools to discuss refinements to aspects of the method and experimental design, implement methods, write code for experiments and figures, identify relevant literature, and draft and edit portions of the paper for clarity and readability. The core research ideas and methodological decisions originated with the authors. We reviewed and validated AI-assisted outputs before use, verified the existence and relevance of AI-found literature, and proofread AI-proposed paper edits before incorporating them. We did not use generative AI for any other tasks on ICLR's required-disclosure list. We take full responsibility for the final content of this work.

REFERENCES

Mikita Balesni, Tomek Korbak, and Owain Evans. Lessons from studying two-hop latent reasoning, 2024. arXiv:2411.16353.

Xin Cheng, Xun Wang, Xingxing Zhang, Tao Ge, Si-Qing Chen, Furu Wei, Huishuai Zhang, and Dongyan Zhao. xrag: Extreme context compression for retrieval-augmented generation with one token, 2024.

Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics*, 2024. arXiv:2307.12976.

Qin Dai, Benjamin Heinzerling, and Kentaro Inui. Representational analysis of binding in language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 17468–17493, 2024. doi: 10.18653/v1/2024.emnlp-main.967. URL https://aclanthology.org/2024.emnlp-main.967/.

Jiahai Feng and Jacob Steinhardt. How do language models bind entities in context? In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.17191.

Jiahai Feng, Stuart Russell, and Jacob Steinhardt. Monitoring latent world states in language models with propositional probes. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=0yvZm2AjUr.

Yoav Gur-Arieh, Mor Geva, and Atticus Geiger. Mixing mechanisms: How language models retrieve bound entities in-context, 2025. URL https://arxiv.org/abs/2510.06182.

Weronika Łajewska, Momchil Hardalov, Laura Aina, Neha Anna John, Hang Su, and Lluís Màrquez. Understanding and improving information preservation in prompt compression for LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 17520–17541, 2025. doi: 10.18653/v1/2025.findings-emnlp.949. URL https://aclanthology.org/2025.findings-emnlp.949/.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen-tau Yih, Tim Rocktaschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, 2020. URL https://arxiv.org/abs/2005.11401.

Yuankai Li, Jia-Chen Gu, Di Wu, Kai-Wei Chang, and Nanyun Peng. BRIEF: Bridging retrieval and inference for multi-hop reasoning via compression. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 5464–5485, 2025a. doi: 10.18653/v1/2025.findings-naacl.301. URL https://aclanthology.org/2025.findings-naacl.301/.

Zhuowei Li, Zihao Xu, Ligong Han, Yunhe Gao, Song Wen, Di Liu, Hao Wang, and Dimitris N. Metaxas. Implicit in-context learning. In *International Conference on Learning Representations*, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/fc034d186280f55370b6aca7a3285a65-Paper-Conference.pdf.

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

Soyoung Oh and Vera Demberg. A retrieval conditioned rebinding circuit for dynamic entity tracking in large language models, 2026. arXiv:2606.08644.

Nina Panickssery, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering llama 2 via contrastive activation addition, 2024.

Sameer Panwar. DeCAL tokenwise compression, 2025. URL https://arxiv.org/abs/2508.08514.

Nikhil Prakash, Tamar Rott Shaham, Tal Haklay, Yonatan Belinkov, and David Bau. Fine-tuning enhances existing mechanisms: A case study on entity tracking. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2402.14811.

David Rau, Shuai Wang, Herve Dejean, and Stephane Clinchant. Context embeddings for efficient answer generation in rag, 2024.

Eric Todd, Millicent L. Li, Arnab Sen Sharma, Aaron Mueller, Byron C. Wallace, and David Bau. Function vectors in large language models. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.15213.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering, 2024.

Zexuan Zhong, Zhengxuan Wu, Christopher D. Manning, Christopher Potts, and Danqi Chen. MQuAKE: Assessing knowledge editing in language models via multi-hop questions. In *Proceedings of EMNLP*, 2023. arXiv:2305.14795.

A EXPERIMENTAL DETAILS AND SUPPLEMENTARY COMPARISONS

The appendix follows the questions in the main text. We begin with how we write states and select heads, then describe the direct-answer and recitation tests for Section 3, the placement and factor comparisons for Section 4, and the two-hop and mixing experiments for Section 5. For each experiment, we explain what we do, how we score answers, which conditions we compare, and what the results mean.

The models are Gemma-2-9B-Instruct, Llama-3.1-8B-Instruct, and Llama-2-13B-Chat. Some supplementary comparisons use only Gemma and Llama-3.1; we state this at the comparison. Unless specified otherwise, a condition contains 400 examples, and compared conditions within a model use the same examples. Reusing those examples across layers or conditions does not create additional independent samples.

Question	Where to find the comparison
What is written, and which heads are measured?	Appendix A.1
Does one correct answer show that the model used a pairing?	Appendix A.2
Does moving an item state change which person it is paired with?	Appendix A.3
What do Slot, Content, and the selected heads contribute?	Appendix A.4
Can written facts support a two-hop answer?	Appendix A.5
Which positions can receive added Content before answers change?	Appendix A.6
Can attention patches recover pooled writes?	Appendix A.5.4
What happens when full states are written late?	Appendix A.5.6

A.1 HOW WE WRITE STATES AND MEASURE THEIR EFFECTS

A.1.1 OBTAINING THE STATES TO WRITE

A donor prompt provides a state for the value to be written. For example, to estimate umbrella's mean state μ at an object position, we keep umbrella at that position and vary the other values in the donor prompts. A target prompt replaces the relevant entity words with placeholders. Writing then supplies states at selected placeholder positions, after which the model continues its forward computation and answers the question. In the two-hop experiments, relation words and the distractor remain text.

The location and timing of a write are part of the experiment. The delivery comparison in Appendix A.2 writes at the input to layer 1. The factor-reconstruction and two-hop comparisons replace states successively at layers 0–8. The cross-layer exchange instead changes the two item states at one tested layer. Layer and head indices start at zero.

Slot and receiving position refer to different objects. Slot is an estimated component of a source state; the receiving position is the token position where a state is placed. A Slot factor can move while the receiving positions remain fixed. The cross-layer exchange averages Content differences across two source positions (Appendix A.3.1); the mixing experiment retains each position's own difference so that its original Slot plus Content equals its full μ (Appendix A.6.1). We give each estimator where its experiment is introduced.

A.1.2 SCORING ANSWERS AND COMPARING INTERNAL STATES

The experiments ask different questions and therefore score answers differently. The direct-answer and recitation tests score generated text, including whether an answer says who owns the item (Appendix A.2.2). The factor-exchange tests judge the two item answers together (Appendix A.3.3). The synthetic two-hop tests compare the first-token scores of fixed answer candidates; the real-entity tests score names that can contain multiple tokens (Appendix A.5). Candidate order and tie handling are stated with each experiment.

Internal similarity and answer accuracy are separate measurements. Appendix A.4 defines R_K, R_V, and QK gain before using them to compare factor writes. A high similarity to text does not itself establish that the model selects the correct fact or answers correctly.

A.1.3 SELECTING THE HEADS USED IN THE COMPARISONS

We locate attention heads whose outputs contribute to retrieving the correct owner from two textual facts. On a separate screen of 32 items, we replace each head's output at each position with its mean across the screen items at that position, and measure the decrease in the logit difference between the correct and alternative names. We screen all 672 heads on Gemma, 1024 on Llama-3.1, and 1600 on Llama-2-Chat.

Among heads with a positive individual effect, we rank heads by this decrease and start with the most influential head and progressively increase the number ablated. We select the smallest number of heads whose combined ablation reaches 90% of the decrease from jointly ablating all positive-effect heads. This selects 22 heads in layers 18–40 on Gemma, 13 in layers 12–24 on Llama-3.1, and 5 in layers 14–19 on Llama-2-Chat. For Llama-2-Chat, the selected $(\text{layer}, \text{head})$ pairs, in ranking order, are $(15, 33)$, $(19, 36)$, $(16, 39)$, $(18, 22)$, and $(14, 31)$; indices start at zero. Ablating these heads together reduces the original logit difference between the correct and alternative names by fractions 0.888, 0.926, and 1.032, respectively. A fraction above one means that the alternative name scores higher after ablation. Same-size random sets drawn from heads whose individual ablation does not reduce that difference produce smaller decreases.

The writing experiments reuse the heads selected with textual facts. To check whether the selected heads also matter when the facts are written, a supplementary experiment on Gemma and Llama-3.1 compares their ablation under text and writes on the measurement scaffold. For the question about the first fact's object, the fractions by which ablation reduces the original name-logit difference are 0.9345 under text and 0.9342 under writes on Gemma, and 1.0042 and 1.0012 on Llama-3.1. Thus, the selected heads contribute similarly under these two ways of providing the facts. This ablation tests what is lost when these heads are removed; it does not show that these heads alone can perform the full binding computation.

The complete head list is in Appendix A.7.

A.2 DIRECT ANSWERS, RECITATION, AND UNPROVIDED ITEMS

Section 3 asks whether a correct direct answer is enough to establish that the model used the name–item pairing. The controls below distinguish answering with a supplied name, recovering the supplied object, and reciting the complete fact.

A.2.1 WHAT THE MODEL SEES AND WHERE WE WRITE

A donor prompt contains one fact, for example `Memory: [ Andy has a dog ]`. The target replaces the four fact positions with placeholders. This example shows the memory and forward question; the actual target also includes a case identifier between them:

> Memory: [X X X X]
> Question: Who has the dog?
> Answer:

At each written position, μ averages the states from 16 donor prompts that fix the value at that position while varying the remaining values. The write replaces the corresponding placeholder state at the input to layer 1, with zero-based layer indices. Adding μ minus the neutral placeholder state is equivalent to replacement at this site. The write is reapplied at the same site when generation feeds the full sequence through the model again.

Activation writing supplies states at all four fact positions; an endpoint write supplies the name and object positions, leaving the other two as placeholders. Single-endpoint controls supply only the name or only the object. Text and no-write references establish performance with explicit facts and with placeholders alone. Random writes at the same positions, matched to each carrier's norm, test whether vectors of the same size but without the target content still produce the effect. Table 4

compares text, activation writing, endpoint writes, and no write on all three models; the other carriers provide supplementary controls. Each condition contains 400 items per model.

A.2.2 Checking whether an answer says who owns the item

For Tables 4 and 5, we rescore the saved answers rather than judging them only by their first word. We make no changes to the prompts, writes, or generated tokens. Table 4 queries the supplied item; Table 5 queries a different, unprovided item. For both tables, we ask whether the answer says that the queried item belongs to the supplied person. This is correct when the item was supplied, but unsupported when it was not. Recitation remains a separate, unchanged score.

Generation uses raw completion prompts without a chat template, greedy decoding, and at most 12 tokens. We read only the answer before a generated `Question:`, `Memory:`, or `Case:` field. A bare supplied name or an explicit ownership statement about the queried item counts as saying who owns the item. A name with only punctuation, brackets, or trailing X placeholders also counts; a sequence such as `Alice X X umbrella` without a relation does not. Additional name modifiers do not change this decision, but are not themselves verified. Merely mentioning the name, repeating a fact about another item, or explicitly retracting the initial answer does not count. We retain explanations when checking whether the answer retracts its initial assignment. Incomplete responses are judged only from the recorded tokens.

We record whether the answer names the supplied person as owner, says the answer is unknown, explicitly denies ownership, names another owner, or cannot be classified. Unknown and statements that information is unavailable count as abstentions. Explicit negations and `None` count as denials. Contradictory answers and incomplete owner descriptions are undetermined. The tables report the number of answers naming the supplied person as owner, divided by 400. Other answers can be wrong or explicitly deny ownership; they do not all express uncertainty. Two independent classification passes agreed on whether the answer names the supplied person as owner for all 4,800 Table 4 answers and all 9,600 Table 5 answers. These are repeated conditions on the same examples, not independent samples of those sizes.

Table 4: Direct-answer and recitation pass rates (400 examples per condition). Recitation explicitly asks for the complete fact.

	Gemma		Llama-3.1		Llama-2-Chat	
Condition	Recall	Recitation	Recall	Recitation	Recall	Recitation
Activation writing	1.000	1.000	1.000	1.000	1.000	.998
Name and item only	**1.000**	**.000**	**.998**	**.000**	**.968**	**.000**
No write	.000	.000	.000	.000	.000	.000
Full text	1.000	1.000	1.000	1.000	1.000	.990

A.2.3 Recovering an object and reciting a fact

The forward question for Table 4 asks `Who has the {OBJECT}?`, without an unknown instruction. We score the recorded answers as described above.

A separate test asking what the person has, conducted on Gemma and Llama-3.1, checks whether the written object can also be recovered. It asks `What does {NAME} have?`, preceded by `If the facts do not say, answer unknown.` The scorer examines the first six alphabetic words of the generated answer, skips articles, the queried name, and specified linking words, and scores the first remaining word as the object answer. Endpoint writes recover the supplied object on $399/400$ Gemma items and $378/400$ Llama-3.1 items. Holding the name fixed and changing the supplied object makes the answer follow the new object on $399/400$ and $375/400$ items, respectively. Activation writing succeeds on all items in both comparisons. Thus, in this two-model control, the model can name the written object when asked what the person has, although endpoint success is not perfect on every sample.

Recitation asks the model to repeat the complete fact. For the recitation results in Table 4, the prompt inserts `Repeat the memory above exactly, then stop.` before `Memory: [`. Success requires the expected four-word fact at the start of the completion; trailing text is not scored. Text and activation writing reach 1.000 on Gemma and Llama-3.1; on Llama-2-Chat they

reach .990 and .998, respectively. Endpoint recitation is zero on all three models. Endpoint forward recall, however, reaches 1.000, .998, and .968 on Gemma, Llama-3.1, and Llama-2-Chat respectively, compared with 1.000 for text on each model. These comparisons separate direct answering from complete recitation; they do not establish that endpoint states contain no relation information.

To check the role of the response instruction, a separate prompt control on Gemma and Llama-3.1 omitted the recitation instruction and ended at `Memory: [`. Gemma's recitation rates were .715 for text and .795 for activation writing, while both were 1.000 on Llama-3.1. The improvement under the explicit instruction shows why Table 4 states the requested response format.

Asking about an item that was not supplied. The unprovided-item test asks whether the model returns the supplied name even when no fact links that name to the queried item. It compares prompts with and without the instruction to answer unknown, and counts supplied-name answers rather than accuracy. Such answers show why returning the supplied name alone does not establish that the model used the pairing. The test does not determine whether the original name–item pairing formed.

Table 5: Proportion of answers that attribute an unprovided item to the supplied person (400 examples per cell). Columns compare absence or presence of the unknown instruction; higher values mean more such answers without a corresponding fact.

	Gemma		Llama-3.1		Llama-2-Chat	
Condition	No hint	Unknown hint	No hint	Unknown hint	No hint	Unknown hint
Full text	.003	.000	.998	.403	.855	.885
Name-and-item text	.538	.000	.980	.678	.893	.623
Activation writing	.003	.000	1.000	.443	.863	.880
Name-and-item write	**.423**	**.000**	**.990**	**.720**	**.863**	**.640**

A.2.4 Direct-answer and recitation failures

With name-and-item writing, Llama-2-Chat identifies the expected owner on $387/400$ examples. The 13 remaining answers comprise two other-name answers (`Bob` instead of `Alice`), three X placeholders, three titles (`The king` twice and `The Queen` once, for Charles), four unfinished owner descriptions, and one response that treats `Mark X` as part of a chair's name. Llama-3.1's one non-passing answer is `Frank X X ball`, which repeats the endpoints without saying who owns the item. The 36 Llama-2-Chat responses missed by the old first-word scorer, such as `The camera belongs to Jason.`, now count as correctly answering who owns the item.

On Llama-2-Chat, the rare recitation failures are object substitutions. Under activation writing, the single failure says `Andy has a cat` instead of `Andy has a dog`. Under complete text, four failures retain the expected name and relation but replace the object: dog with cat for Andy, Grace and Simon, and lamp with book for Thomas.

A.2.5 Supported conclusion

Endpoint writes can support direct answers while failing complete recitation, with lower direct-answer performance on Llama-2-Chat than on the other two models. Together with answers to questions about unprovided items, these results show why correct direct recall alone is insufficient to establish use of the pairing. The subsequent pairing tests hold the available content fixed and change its assignment to distinguish those possibilities.

A.3 Moving Slot and Content across layers

When umbrella's state is moved beside Ben, does the model pair umbrella with Ben, or retain its earlier pairing with Alice? Section 4 tests this first with full states and then with estimated Slot and Content factors. Separating the factors asks what must move together for the original pairing to survive. We explain how we estimate and exchange Slot and Content, how the two answers determine the pairing, and how the results change across layers.

Full-state exchange and early-layer controls. The model reads "Alice has an umbrella; Ben has a bottle." At the tested layer, we take the residual states h at the umbrella and bottle positions and place each at the other's position. The name states and input text remain unchanged, and computation continues through the remaining layers. We then ask separately what Alice and Ben have. Answering bottle and umbrella, respectively, gives the crossed pairing; answering umbrella and bottle gives the original pairing. Every other joint outcome is neither. Each test starts again from the unmodified input.

We write the object states back to their original positions to check whether reading and writing them changes the answers. We also move the Alice and umbrella states together to the positions originally occupied by Ben and bottle, and move the Ben and bottle states in the other direction, keeping each name with its own object. Compared with moving only the objects, this control helps separate disruption from moving states from the effect of changing the neighboring name. Unrelated states test whether modifying these positions can still change the answers.

Table 6 averages the results from layer 0 through the layer before the first selected binding head. This checks whether the pairing mainly follows the receiving position before computation reaches those heads. Each layer uses the same 400 examples: we compute pairing proportions within each layer, then average across layers.

Table 6: Mean pairing proportions before the first selected binding head, scored jointly on the two item questions (400 examples per layer).

	Gemma (0–17)			Llama-3.1 (0–11)			Llama-2-Chat (0–13)		
Condition	Crossed	Original	Neither	Crossed	Original	Neither	Crossed	Original	Neither
Write back in place	0.000	1.000	0.000	0.000	0.965	0.035	0.000	0.980	0.020
Exchange item states	0.965	0.001	0.035	0.853	0.002	0.145	0.752	0.078	0.171
Move names and items together	0.120	0.736	0.145	0.183	0.497	0.320	0.235	0.484	0.281
Unrelated states	0.000	0.000	1.000	0.000	0.000	1.000	0.000	0.005	0.995

A.3.1 Estimating the Slot and Content factors to exchange

Section 2 describes the donor averaging and additive assumption. Here we give the estimator used at each layer. Let $\mu(s, v)$ be the donor mean for item v at source item position $s \in \{0, 1\}$, with its paired name and the other fact varied. Averaging over the estimation items $\mathcal{V}$ gives Slot; averaging the two position-specific residuals gives Content:

$$\sigma(s) = \frac{1}{|\mathcal{V}|} \sum_{v' \in \mathcal{V}} \mu(s, v'), \qquad \Delta c(v) = \frac{[\mu(0, v) - \sigma(0)] + [\mu(1, v) - \sigma(1)]}{2}.$$

We estimate name factors separately in the same way. Because Content averages differences from two positions, $\sigma + \Delta c$ need not equal the full μ at either position. We compare each exchange with the unexchanged factor combination, rather than assuming that this combination equals the full state.

A.3.2 Which states we exchange and what we compare them with

The four factor combinations are specified in Section 4.1: no exchange, Slot-only, Content-only, and joint exchange. Each intervention changes the two item states at one layer while keeping the name states and questions fixed. Below we specify the replacement controls and scoring used for the full layer sweep. Unrelated-example and random states replace the same item states to check whether interventions at these positions can still affect answers.

For the random control, we collect the exchanged Slot plus Content vectors across layers, draw a random array of the same shape, and scale the entire array to the norm of the factor array. A test at one layer uses the corresponding row. Norm matching thus applies across the full array, rather than independently at each layer.

A.3.3 Judging the pairing from both answers

Table 7 uses the two item questions (WHAT), "What does Alice have?" and "What does Ben have?", to describe the answers after exchanging factors at each layer. The original joint answer is umbrella

for Alice and bottle for Ben; the crossed answer reverses those assignments. Each test changes only the two item positions at one layer; name states and questions remain fixed. Each layer's Slot and Content factors follow the estimator above. The source pool contains the tested names and items, so the table does not test names or items that were absent when the factors were estimated.

Each question compares first-token logits for the two items and unknown; tied maxima are marked as ties. The two item answers must jointly form the original or crossed pairing. Every other pattern is neither, including ties, unknown, and selecting the same item for both questions. We first compute each layer's proportions, then average them arithmetically over the listed range. The same examples recur at every layer and are not counted as new independent samples. Rounding may prevent displayed percentages from summing to exactly 100%.

The questions specified in advance for testing the hypothesis asked who owns an item (WHO). Table 7 instead describes how answers to what each person has (WHAT) change across layers. Random replacement did not disrupt pairing at every layer: at late layers it often preserved the original answers. Unchanged late-layer answers therefore do not establish that the exchanged states retain the pairing information.

A.3.4 How the effect changes with depth

Figure 3 follows the joint exchange layer by layer. Table 7 adds the individual-factor exchanges and controls. Its early, middle, and late ranges summarize the observed sweep; they differ from the range in Table 6, which was specified in advance from the first selected head. Start with the Content-only and both-factors rows: both move the item, but only one preserves the factor combination.

Table 7: Factor-exchange outcomes by layer range (percent); ranges summarize the observed sweep.

	Gemma			Llama-3.1			Llama-2-Chat		
Condition	Crossed	Original	Neither	Crossed	Original	Neither	Crossed	Original	Neither
Early		0–17			0–10			0–13	
No exchange	0.0	100.0	0.0	0.0	99.2	0.8	0.0	98.0	2.0
Slot only	**0.0**	**98.7**	**1.3**	**0.0**	**98.7**	**1.3**	**2.6**	**83.1**	**14.3**
Content only	**100.0**	**0.0**	**0.0**	**98.6**	**0.0**	**1.4**	**94.1**	**0.0**	**5.9**
Both factors	**98.0**	**0.0**	**2.0**	**97.1**	**0.0**	**2.9**	**77.8**	**6.6**	**15.6**
Unrelated-example states	0.5	2.0	97.5	0.5	2.0	97.5	1.2	2.3	96.5
Norm-matched random states	0.0	0.0	100.0	0.0	0.0	100.0	0.1	0.0	99.9
Middle		19–29			13–22			14–19	
No exchange	0.0	100.0	0.0	0.0	98.6	1.5	0.0	97.9	2.1
Slot only	**71.5**	**5.5**	**23.0**	**28.5**	**38.2**	**33.3**	**19.1**	**11.8**	**69.2**
Content only	**93.4**	**0.5**	**6.1**	**37.8**	**33.7**	**28.5**	**51.2**	**2.4**	**46.4**
Both factors	**0.1**	**93.3**	**6.6**	**0.0**	**93.6**	**6.4**	**0.0**	**88.8**	**11.3**
Unrelated-example states	0.5	2.1	97.4	0.2	2.4	97.4	0.5	3.3	96.2
Norm-matched random states	0.1	4.6	95.3	0.0	4.9	95.1	0.0	8.1	91.9
Late		30–41			23–31			20–39	
No exchange	0.0	100.0	0.0	0.0	99.0	1.0	0.0	98.4	1.6
Slot only	0.1	91.4	8.5	0.0	96.6	3.4	0.3	87.4	12.3
Content only	0.3	89.7	10.0	0.0	96.2	3.8	0.5	88.3	11.2
Both factors	0.0	100.0	0.0	0.0	99.1	0.9	0.0	98.1	1.9
Unrelated-example states	**0.0**	**82.2**	**17.8**	**0.0**	**70.7**	**29.3**	**0.0**	**88.9**	**11.1**
Norm-matched random states	**0.0**	**98.5**	**1.5**	**0.0**	**71.6**	**28.4**	**0.0**	**94.6**	**5.4**

Early layers: pairing mainly follows where Content is received. First compare Slot-only with Content-only exchange. The former leaves Content in place and mostly preserves pairing; the latter moves Content beside the other name and mainly produces crossed pairing. Moving both factors also fails to preserve the original pairing, despite retaining the original Slot. These comparisons support an early receiving-position effect.

Middle layers: joint movement differs from separate exchanges. Compare Content-only with joint exchange: both move Content, but only the latter retains its original Slot. In Gemma, exchanging either factor alone mainly crosses pairing, whereas moving both mainly preserves it. Llama-3.1 has more dispersed single-factor outcomes, and Llama-2-Chat has more neither outcomes, but all three mainly retain the original pairing under joint exchange. The individual-factor effects thus dif-

fer across models; the shared pattern is that retaining the factor combination helps preserve pairing after movement.

Late layers: answers often stay unchanged even with random states. Unrelated-state and random replacements mainly disrupt answering at middle layers, showing that interventions at these positions remain effective. At late layers, both replacements often preserve original pairing as well, so unchanged answers do not identify what carries the pairing. This control comparison is necessary for interpreting the layerwise curves in the main text.

A.3.5 Repeating the test at layers chosen in advance

A separate confirmation selected five layers per model before execution: Llama-3.1 0, 8, 12, 17, and 31; Gemma 0, 8, 18, 25, and 41. Tables 8 and 9 report the six conditions used in the depth comparison at all of these layers. We show WHAT and WHO separately: WHAT asks what Alice and Ben have; WHO asks who has umbrella and bottle. Each pair of answers is classified as original, crossed, or neither. Tied candidate maxima count as neither.

The two question types do not always support the same interpretation. At Gemma layer 25, random replacement disrupts the WHAT pairing but leaves the WHO pairing unchanged, so WHO fails the control that checks whether replacing these states affects answers. Llama-3.1's confirmation does not provide equally strong later-layer evidence. The full tables retain these failures alongside the successful conditions.

The Ties column counts tied maxima across all four questions (1,600 probes per row); tied answers count as neither. Some Llama-3.1 conditions at layers 8, 12, and 17 exceed the preregistered 1% tie threshold, so these rows describe the observed behavior rather than establish a clean confirmation.

Table 8: Gemma confirmation at every layer chosen in advance. Counts out of 400 on the two WHAT questions and, separately, the two WHO questions; each row uses the same examples. Ties counts tied maxima across all 1,600 question probes.

		WHAT			WHO			
Layer	Condition	Crossed	Original	Neither	Crossed	Original	Neither	Ties
0	No exchange	0	400	0	0	400	0	0
0	Slot only	0	400	0	0	400	0	0
0	Content only	400	0	0	400	0	0	0
0	Both factors	400	0	0	400	0	0	0
0	Unrelated states	2	8	390	2	8	390	0
0	Random states	0	0	400	0	0	400	10
8	No exchange	0	400	0	0	400	0	0
8	Slot only	0	400	0	0	400	0	0
8	Content only	400	0	0	400	0	0	0
8	Both factors	400	0	0	400	0	0	0
8	Unrelated states	2	8	390	3	10	387	2
8	Random states	0	0	400	0	3	397	8
18	No exchange	0	400	0	0	400	0	0
18	Slot only	3	234	163	0	400	0	9
18	Content only	400	0	0	0	400	0	0
18	Both factors	27	109	264	0	400	0	11
18	Unrelated states	2	8	390	0	400	0	3
18	Random states	0	0	400	0	400	0	5
25	No exchange	0	400	0	0	400	0	0
25	Slot only	397	0	3	0	400	0	0
25	Content only	400	0	0	0	400	0	0
25	Both factors	0	400	0	0	400	0	0
25	Unrelated states	2	8	390	0	400	0	2
25	Random states	0	0	400	0	400	0	9
41	No exchange	0	400	0	0	400	0	0
41	Slot only	0	400	0	0	400	0	0
41	Content only	0	400	0	0	400	0	0
41	Both factors	0	400	0	0	400	0	0
41	Unrelated states	0	400	0	0	400	0	0
41	Random states	0	400	0	0	400	0	0

Table 9: Llama-3.1 confirmation at every layer chosen in advance. Counts out of 400 on the two WHAT questions and, separately, the two WHO questions; each row uses the same examples. Ties counts tied maxima across all 1,600 question probes.

		WHAT			WHO			
Layer	Condition	Crossed	Original	Neither	Crossed	Original	Neither	Ties
0	No exchange	0	397	3	0	376	24	6
0	Slot only	0	397	3	0	376	24	6
0	Content only	396	0	4	384	0	16	6
0	Both factors	396	0	4	384	0	16	6
0	Unrelated states	2	8	390	5	12	383	13
0	Random states	0	0	400	15	17	368	38
8	No exchange	0	398	2	0	377	23	2
8	Slot only	0	388	12	0	356	44	10
8	Content only	397	0	3	336	0	64	4
8	Both factors	383	0	17	330	1	69	17
8	Unrelated states	2	8	390	12	228	160	24
8	Random states	0	0	400	0	310	90	19
12	No exchange	0	398	2	0	380	20	7
12	Slot only	9	111	280	0	365	35	23
12	Content only	387	0	13	0	321	79	25
12	Both factors	7	98	295	0	367	33	27
12	Unrelated states	2	8	390	0	373	27	13
12	Random states	1	2	397	0	371	29	6
17	No exchange	0	392	8	0	375	25	6
17	Slot only	77	47	276	0	373	27	25
17	Content only	114	27	259	0	373	27	32
17	Both factors	0	384	16	0	372	28	11
17	Unrelated states	0	8	392	0	373	27	12
17	Random states	0	23	377	0	372	28	15
31	No exchange	0	397	3	0	376	24	5
31	Slot only	0	397	3	0	377	23	6
31	Content only	0	397	3	0	376	24	7
31	Both factors	0	397	3	0	377	23	6
31	Unrelated states	0	395	5	0	374	26	8
31	Random states	0	394	6	0	374	26	7

A.3.6 LAYER-BY-LAYER VALUES BEHIND THE DEPTH CURVES

Tables 10, 11, and 12 give every layer behind Figure 3. They show joint Slot-and-Content exchange and random replacement, including the transition layers omitted from the range summaries. Crossed and original counts under joint exchange produce the red and blue curves; original counts under random replacement produce the gray dashed curve. Divide each count by 400 to obtain its proportion. The same examples recur across layers.

Table 10: Gemma: layer-by-layer pairing counts for joint factor exchange and random replacement. Each condition uses the same 400 examples at every layer.

	Both factors exchanged			Random replacement		
Layer	Crossed	Original	Neither	Crossed	Original	Neither
0	400	0	0	0	0	400
1	400	0	0	0	0	400
2	400	0	0	0	0	400
3	400	0	0	0	0	400
4	400	0	0	0	0	400
5	400	0	0	0	0	400
6	400	0	0	0	0	400
7	400	0	0	0	0	400
8	400	0	0	0	0	400
9	400	0	0	0	0	400
10	400	0	0	0	0	400
11	400	0	0	0	0	400
12	400	0	0	0	0	400
13	400	0	0	0	0	400
14	400	0	0	0	0	400
15	400	0	0	0	0	400
16	399	0	1	0	0	400
17	255	0	145	0	0	400
18	27	109	264	0	0	400
19	4	219	177	0	0	400
20	0	303	97	0	0	400
21	0	386	14	0	0	400
22	0	397	3	0	0	400
23	0	399	1	0	0	400
24	0	400	0	0	0	400
25	0	400	0	0	0	400
26	0	400	0	0	11	389
27	0	400	0	2	21	377
28	0	400	0	1	16	383
29	0	400	0	0	155	245
30	0	400	0	0	373	27
31	0	400	0	0	355	45
32	0	400	0	0	400	0
33	0	400	0	0	400	0
34	0	400	0	0	400	0
35	0	400	0	0	400	0
36	0	400	0	0	400	0
37	0	400	0	0	400	0
38	0	400	0	0	400	0
39	0	400	0	0	400	0
40	0	400	0	0	400	0
41	0	400	0	0	400	0

Table 11: Llama-3.1: layer-by-layer pairing counts for joint factor exchange and random replacement. Each condition uses the same 400 examples at every layer.

	Both factors exchanged			Random replacement		
Layer	Crossed	Original	Neither	Crossed	Original	Neither
0	396	0	4	0	0	400
1	396	0	4	0	0	400
2	376	0	24	0	0	400
3	394	0	6	0	0	400
4	395	0	5	0	0	400
5	396	0	4	0	0	400
6	400	0	0	0	0	400
7	398	0	2	0	0	400
8	383	0	17	0	0	400
9	386	0	14	0	0	400
10	352	0	48	0	1	399
11	148	4	248	2	2	396
12	7	98	295	1	2	397
13	0	319	81	0	2	398
14	0	327	73	0	4	396
15	0	372	28	0	3	397
16	0	378	22	0	36	364
17	0	384	16	0	23	377
18	0	385	15	0	34	366
19	0	394	6	0	20	380
20	0	395	5	0	9	391
21	0	395	5	0	6	394
22	0	395	5	0	60	340
23	0	395	5	0	70	330
24	0	397	3	0	129	271
25	0	397	3	0	252	148
26	0	397	3	0	245	155
27	0	397	3	0	322	78
28	0	397	3	0	387	13
29	0	396	4	0	388	12
30	0	396	4	0	392	8
31	0	397	3	0	394	6

Table 12: Llama-2-Chat: layer-by-layer pairing counts for joint factor exchange and random replacement. Each condition uses the same 400 examples at every layer.

	Both factors exchanged			Random replacement		
Layer	Crossed	Original	Neither	Crossed	Original	Neither
0	392	0	8	0	0	400
1	391	0	9	0	0	400
2	392	0	8	0	0	400
3	391	0	9	0	0	400
4	380	0	20	3	0	397
5	390	0	10	0	0	400
6	387	0	13	0	0	400
7	387	0	13	1	0	399
8	376	0	24	0	0	400
9	355	0	45	0	0	400
10	296	0	104	0	0	400
11	203	0	197	0	0	400
12	19	114	267	0	0	400
13	0	256	144	0	0	400
14	0	282	118	0	0	400
15	0	324	76	0	1	399
16	0	374	26	0	2	398
17	0	375	25	0	4	396
18	0	387	13	0	56	344
19	0	388	12	0	132	268
20	0	390	10	0	250	150
21	0	384	16	0	257	143
22	0	391	9	0	381	19
23	0	390	10	0	380	20
24	0	390	10	0	394	6
25	0	392	8	0	396	4
26	0	393	7	0	395	5
27	0	394	6	0	397	3
28	0	394	6	0	396	4
29	0	395	5	0	393	7
30	0	395	5	0	395	5
31	0	394	6	0	393	7
32	0	393	7	0	391	9
33	0	393	7	0	394	6
34	0	393	7	0	391	9
35	0	393	7	0	391	9
36	0	393	7	0	392	8
37	0	394	6	0	392	8
38	0	393	7	0	393	7
39	0	393	7	0	393	7

A.4 How answers change after factor writes, Q/K edits, and head ablation

The placement experiments ask which pairing survives a move. Here we ask what happens after early states are written: do Slot and Content make later K/V resemble the text condition, does changing Q/K matching change the selected owner, and can a stronger write compensate when selected heads are ablated? These are separate comparisons. The main factor table and the ablation analysis below include all three models; the Q/K editing comparison uses Gemma and Llama-3.1.

A.4.1 Comparing K/V after writing with the text condition

Writing before the measured heads. To examine what the factors contribute to pairing, we compare full states, Slot plus Content, and each factor on its own. The target prompt contains two placeholder facts and asks which person owns one of the objects. Each written state replaces the residual state at its corresponding fact position over layers 0–8. This range ends before the first selected binding head: layer 18 on Gemma, 12 on Llama-3.1, and 14 on Llama-2-Chat. The experiment therefore does not directly replace the key that it later measures.

Table 2 in the main text gives the complete three-model factor comparison. Its answer rate scores who owns the first fact's object ($n = 400$), rather than pooling both query directions. Below we explain how we measure internal changes and what the factor comparisons can tell us.

Comparing K/V, attention, and answers. At each selected binding head, we record the keys and values at the written fact positions. Let $C_X(c)$ be the mean cosine similarity to the text reference for representation $X \in \{K, V\}$ under condition c, averaged over the measured heads, positions, and examples. We normalize this mean as

$$R_X(c) = \frac{C_X(c) - C_X(\text{no write})}{1 - C_X(\text{no write})}.$$

Cosines are averaged first; the recovery ratios are not averaged per example. $R_X = 0$ is the no-write reference and $R_X = 1$ is the text reference. *QK gain* is the change over no write in the log attention-mass ratio between the correct and competing facts, computed at the answer position and averaged over selected heads and examples. *Answer* is the fraction of items on which the intended owner has the largest score among the answer candidates fixed before the run. The candidates are ordered correct owner, other owner, unknown, and exact ties select the first candidate.

The σ-only rows recover part of the similarity to text K/V, but QK gain remains near zero and the intended-owner answer rate is zero. In this writing setup, σ alone is insufficient to establish correct selection and answering. The Δc-only state also fails to reconstruct a key and, on Llama-3.1, sometimes biases the final name. Because its key is farther from the text reference than the no-write key, this failure shows insufficiency under this intervention; it does not prove that Δc contains no information that contributes to Q/K matching.

A.4.2 Using the same Slot at two source positions

The shared-Slot condition in the main text writes a synthesized state $\bar{\sigma} + \Delta c$. For each role separately (name or object), it averages the Slot estimates from source positions 0 and 1, $\bar{\sigma} = (\sigma_0 + \sigma_1)/2$, and adds the Content factor for the value being written. For each role and layer, the estimators give

$$\bar{\sigma} + \Delta c(v) = \frac{\mu(0, v) + \mu(1, v)}{2}.$$

Thus, for a fixed value, the complete replacement vector does not depend on which source position was assigned to that value. Its lookup does not depend on which entity is currently paired with it, either. Each value's Content, the receiving position, and the target-prompt structure remain intact. The K/V similarity to text and the mostly correct answers therefore show that, for the tested pairing question, the early written vector need not preserve this source-position distinction. This does not mean the entire Slot can be omitted; the experiment also does not separately measure the effects of the receiving position and the surrounding prompt.

A.4.3 ADDING THE FOUR OFFSETS TOGETHER AT THE ANSWER POSITION

Does adding all four offsets at the answer position have the same effect as writing each fact position separately? We add this comparison to the factor-writing experiment; the answer position occurs after the four fact positions. At each layer 0–8, we subtract each role's placeholder state from its intended mean state, sum the four offsets (two names and two objects), and add that sum to the answer-position residual:

$$h^{\ell}_{\text{ans}} \leftarrow h^{\ell}_{\text{ans}} + \sum_{k\in\{\text{name,object}\}} \sum_{s=0}^{1} \left(\mu^{\ell}_{k,c_s,s} - h^{\ell}_{\text{ph},k,s}\right).$$

Here c_s is the value of role k at source position s. This is a sum of offsets added at the answer position, whereas the fact-position intervention replaces four separate residual states. Causal masking keeps K/V at the earlier fact positions unchanged, so $R_K = R_V = 0$. The answer-position query can change: QK gain is $-.003$ on Gemma and $-.005$ on Llama-3.1, close to zero but not forced to zero by masking. This control can change the current answer; the experiment does not test whether its effect can be reused by a later query.

A.4.4 CHANGING THE WRITTEN OBJECTS OR CHANGING ONLY Q/K

Can Q/K edits change the answer while the written facts stay fixed? Suppose the written facts pair Alice with umbrella and Ben with kettle. We can change the owner returned for umbrella by changing what is written. Can we also change that answer while leaving all four written states intact and editing only Q/K matching? Comparing these interventions separates a change to the supplied objects from a change to the matching signals. This comparison uses Gemma and Llama-3.1. The target contains two placeholder facts; mean states μ written at layers 0–8 supply, for example, `Alice--umbrella` and `Ben--kettle`. Each mean is estimated from 16 donor prompts. The object-swap condition writes the other object's mean, estimated at the source position corresponding to its new receiving position, at each of the two object positions, while leaving the name writes fixed. Thus, umbrella is written beside Ben and kettle beside Alice.

Choosing the vectors to add to Q/K. The Q/K interventions retain the original four writes. In this written setting, 64 estimation examples provide a mean name-key vector $\bar{K}_s$ for each source position $s \in \{0, 1\}$ and a mean answer-colon query $\bar{Q}_s$ conditional on querying the object at that position. At every selected head, a key-side edit adds $\alpha(\bar{K}_{1-s} - \bar{K}_s)$ at each name position s. A query-side edit instead adds $\alpha(\bar{Q}_{1-s} - \bar{Q}_s)$ at the answer colon, where s is the position of the queried object. The two sides are tested separately at $\alpha \in \{1, 2, 4\}$. Queries are edited at the selected query heads; keys are edited once at each distinct K/V head shared by those heads. The runs use the fixed corrected head sets: 22 heads on Gemma and 13 on Llama-3.1, selected by the procedure in Appendix A.1.3.

What we compare and how we score the results. For each side and amplitude, a random control replaces the mean difference with a random vector of the same norm at the same positions and heads. An absent-object control asks about a third object that was not written. The answer candidates are the two owners (Alice and Ben in the example) and *unknown*; the candidate with the highest final-position logit wins. For edited queries about the first fact, *expected* means the other owner (Ben in the example), and *stale* means the original owner (Alice). At baseline, expected means the original correct owner. The attention readout sums attention to each fact's name and object positions, pools it across the selected heads, and counts an attention flip when the other fact has strictly more mass than the originally queried fact.

At baseline, the queried fact receives mean attention mass .131 versus .023 on Gemma and .086 versus .024 on Llama-3.1. The attention-flip readout compares which fact has more mass, rather than measuring the absolute amount of attention. The object swap changes both the selected owner and the fact receiving more attention. Query-side edits also change both readouts. At unit amplitude, key-side edits redirect attention on Llama-3.1, but the answer does not reliably follow: the stale answer remains on .805 of items. Table 13 puts the two outcomes side by side: selecting the other owner and moving more attention to the other fact. The table retains the recorded rounding; every cell uses 400 items.

The baseline and object-swap rows come from the object-swap comparison; the Q/K rows include the subsequent per-amplitude controls. Both use the same corrected head sets, writing layers, donor

count, and first-fact question. Owner columns always refer to Alice and Ben in the original facts, so the baseline and edited rows can be compared directly.

Table 13: Baseline, object swap, and Q/K edits (400 items per cell). Q/K edits retain the original fact writes. For the first-fact question, "other owner" is Ben and "original owner" is Alice. Attention flip means the other fact receives more attention. Random reports the other-owner rate under a matched random edit; absent reports unknown for an unwritten object. Dashes mark controls not reported for that row.

Model	Condition	α	Other owner	Original owner	Attention flip	Random	Absent
Gemma	Baseline	–	0.000	1.000	0.000	–	1.000
Gemma	Object swap	–	1.000	0.000	1.000	–	–
Gemma	K	1	0.775	0.225	0.978	0.000	1.000
Gemma	K	2	1.000	0.000	1.000	0.000	1.000
Gemma	K	4	1.000	0.000	1.000	0.033	1.000
Gemma	Q	1	0.968	0.033	1.000	0.000	1.000
Gemma	Q	2	1.000	0.000	1.000	0.000	1.000
Gemma	Q	4	1.000	0.000	1.000	0.010	0.998
Llama-3.1	Baseline	–	0.010	0.980	0.010	–	1.000
Llama-3.1	Object swap	–	0.995	0.005	1.000	–	–
Llama-3.1	K	1	0.182	0.805	0.927	0.037	1.000
Llama-3.1	K	2	0.787	0.198	0.995	0.125	1.000
Llama-3.1	K	4	0.507	0.175	0.547	0.268	1.000
Llama-3.1	Q	1	0.925	0.068	0.995	0.020	1.000
Llama-3.1	Q	2	0.998	0.000	1.000	0.070	1.000
Llama-3.1	Q	4	0.695	0.287	1.000	0.155	1.000

At unit amplitude, matched random edits have expected-answer rates near zero. Across the tested absent-object cells, at least $399/400$ items select *unknown*; this is the scope in which these edits rarely produce an answer to an unwritten object.

The original pass rule and the comparisons added later. The rule fixed before the experiment first requires baseline expected-answer rates of at least .80 and stale-answer rates of at most .20 in both query directions, and a baseline absent-object unknown rate of at least .60. For each intervention side, an amplitude is eligible only if expected plus stale is at least .60. The side passes if any eligible amplitude reaches an expected-answer rate of .50, while its random control at amplitude 4 is at most .10. Gemma passes on both sides; Llama-3.1 passes on neither. Per-amplitude random controls were added later without changing the frozen rule. Applying the same .50 effect and .10 random-control thresholds at each amplitude supports Llama-3.1 query-side amplitudes 1 and 2, but no key-side amplitude. These are supplementary per-amplitude findings, not passes under the original rule.

A.4.5 Testing whether stronger writes help after head ablation

The experiments in Section 4 show that early receiving positions affect pairing and that later states can preserve the relationship. We next ask whether strengthening the early state can help when some binding heads are impaired. We hypothesize that early states may already contain pairing-related information: if the remaining heads can use a stronger input, increasing writing strength could improve answering after ablation. This comparison tests whether stronger early states can compensate for part of these heads' contribution.

We retain the four receiving positions from Section 4.2 and layer 0–8 writing setup, provide Alice–umbrella and Ben–bottle through full μ, and ask "Who has the umbrella?" We jointly vary the number of ablated binding heads and the strength of the same activation writing. We also ablate other heads matched in count and layer to test whether the loss is specific to the selected binding heads.

At writing strength $\alpha = 1$, the receiving state is replaced by μ. We first record the placeholder state h_{ph} at each target layer and position in a forward pass without writing. Increasing the strength scales the change from this fixed reference to μ:

$$h \leftarrow h_{\mathrm{ph}} + \alpha(\mu - h_{\mathrm{ph}}), \qquad \alpha \in \{0.5, 1, 2, 4\}.$$

For example, $\alpha = 2$ doubles this change. Each strength includes a no-ablation condition, so we can distinguish the effect of strengthening itself from recovery of the ablation loss.

Ablation changes the outputs of subsequent heads. Using mean ablation, we replace each selected head's output at all token positions with its mean output from a separate set of written examples, before the attention output projection. We retain the screening order: heads whose individual ablation caused a larger drop in the correct-name versus other-name score difference are removed first. We progressively increase the number removed. Every writing strength uses the same order and mean outputs, allowing comparison of signal strength under the same ablation. A matched-norm random write is the write control. Each cell uses 400 examples. Owner rates compare first-token logits in the order correct owner, other owner, unknown; exact ties select the first candidate.

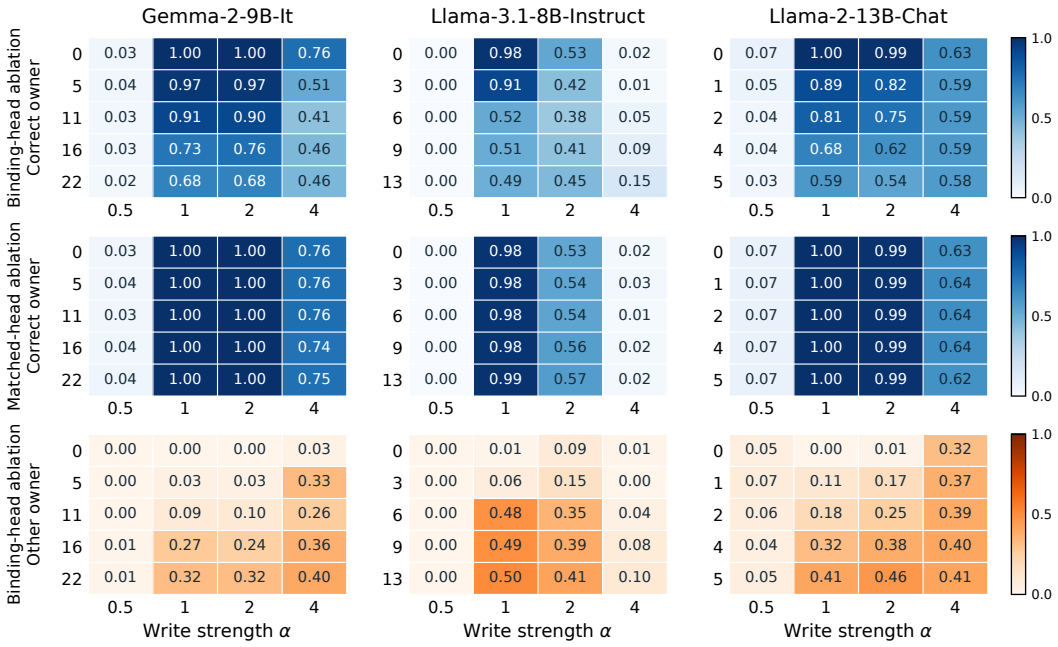

Figure 5: Writing strength and head ablation (400 examples per cell). Columns: Gemma, Llama-3.1, and Llama-2-Chat. Top and middle rows: correct-owner rates after ablating binding heads or matched other heads (blue). Bottom row: other-owner rates after binding-head ablation (orange). Horizontal axes: write strength; vertical axes: number of ablated heads.

The top row of Figure 5 shows that increasing writing strength does not recover the loss in correct answers after binding-head ablation. In the middle row, ablating other heads matched in count and layer does not cause a comparable loss. The effect is thus associated with the selected heads' contribution rather than simply removing the same number of heads. The bottom row shows how the answers change: at the ordinary writing strength $\alpha = 1$, ablating the selected binding heads increases answers naming the other owner in all three models. Thus, wrong answers include naming the other person as owner, rather than only saying the answer is unknown. For example, the model answers Ben when asked who has the umbrella, although the written facts are unchanged.

This result answers our initial hypothesis: if the early signal were simply too weak, increasing write strength should improve answers after ablation, but increasing strength does not consistently compensate for the loss. Without ablation, Gemma answers every question correctly at both the original and doubled strengths; after ablation, doubling does not restore this performance, although one ablation level shows a small improvement. Doubling also fails to help Llama-2-Chat, while stronger writes already impair Llama-3.1 without ablation. Under these tested conditions, a stronger input cannot replace the selected heads' contribution to subsequent computation. Both the supplied state and how the model processes it affect whether it can use the pairing correctly.

Mean ablation replaces whole head outputs. The comparison therefore measures their contribution to answering under these writes; it does not separate their roles in forming a pairing, selecting a fact, and reading content.

A.5 Designing two-hop questions and comparing ways to write the facts

Table 3 in Section 5.1 gives the main text/write comparison. Below we describe the question construction, real-entity examples, and two changes to the writing interface: pooling states into fewer positions and adding offsets instead of replacing states. Each supplementary comparison uses its stated baseline.

We also test whether restoring later Q/K/V changes pooled answers and whether the same full states still work when supplied at later layers.

A.5.1 Questions built from two connected facts

Each item contains two chain facts and one textual distractor:

> Fact 1: Alice has the umbrella.
> Fact 2: The umbrella has the tag.
> Distractor: Ben has the kettle.
> Question: Who has the tag? Answer:

The two-hop question asks for the owner reached through both facts: "Who has the tag?" The direct question asks "Who has the umbrella?" and needs only the first fact. Two control questions ask about the unchanged distractor and an object not supplied in any fact. Candidates are ordered correct owner (Alice), competing owner (Ben), intermediate item (umbrella), and unknown. We compare their first-token logits; the largest wins, with exact ties assigned to the first candidate. A correct-answer rate is the fraction of examples on which the correct candidate ranks first; it is not a mean logit or a free-generation score. Each condition contains 400 examples.

Each chain fact is independently presented as text or written at its corresponding fact positions. When both facts are written, writes successively replace all four entity positions at layers 0–8; the relation words and distractor remain text. A matched-norm random write provides a comparison without the intended content. Writing Fact 1 while leaving Fact 2 as text tests whether the two forms of input can jointly support the answer. The question omits the intermediate item, `umbrella`, so the task requires the connection between the two facts. We score the final answer, without requiring the model to output an intermediate query. Table 3 reports this condition alongside the other text/write combinations.

The distractor question remains at ceiling with text and with writes at all fact positions, and the unprovided-item question returns *unknown*. Matched-norm random writes score $.000$ on the two-hop question on both models. These controls show that writes preserve answering about the textual distractor, while matched-norm random states fail to support the two-hop answer.

A.5.2 Questions about real films, directors, and countries

We repeat the text/write comparison with film–director–country chains from Wikidata. Fact 1 links the film to its director; Fact 2 links the director to a country. The question asks which country the film's director is from.

Selection starts from 2,240 candidate chains. We first retain chains on which Gemma and Llama-3.1 fail both direct questions and the two-hop question without supplied facts, then retain those with enough same-token-length donors for each role under both tokenizers. Entities may span multiple tokens. These selection and preparation requirements leave 237 shared chains. Llama-2-Chat uses the 216-chain subset meeting its tokenizer and donor requirements; it also fails all three questions without the facts on these chains.

Writes use layers 0–8. For entity candidates we compare mean log probability per token; unknown is selected when its first-token logit exceeds those of all entity candidates. Entity-score ties select the first maximum in the order correct country, alternative country, intermediate director. Unknown

overrides the entity scores only when its first-token logit is strictly greater than every entity candidate's first-token logit; an unknown/entity tie does not trigger the override.

The two-hop candidates are the correct country, an alternative country, the intermediate director, and *unknown*. *Two-hop* in Table 14 is the fraction of items on which the correct country ranks first. *Distractor* checks the unchanged textual fact, and *unknown* is the rate of returning *unknown* for an entity that was never delivered.

Table 14: Questions about real films, directors, and countries ($n = 237$ per condition). Every entry is the fraction of items that pass the readout named by the column.

	Gemma			Llama-3.1		
Condition	two-hop	distractor	unknown	two-hop	distractor	unknown
Both facts as text	1.000	.979	1.000	.764	.806	.857
Fact 1 written	1.000	.979	1.000	.738	.814	.852
Fact 2 written	1.000	.975	1.000	.806	.781	.857
Both facts written	1.000	.975	1.000	.793	.793	.848
Matched-norm random	.000	1.000	.996	.025	.937	.114

The text/write two-hop comparison in Table 14 remains near its own text baseline, while the random write gives low two-hop accuracy. The controls also show a model difference: Llama-3.1 frequently fails to answer unknown for an absent entity under random writing. That row does not show that only two-hop answering is affected, because answers about absent entities change too.

The direct intermediate-entity question is not used to decide whether a run is valid. Its candidate set contains two countries and one person, so the model can often select the only person without identifying the director. The two-hop question does not have this artifact because its correct and alternative answers are both countries. We therefore use the two-hop, distractor, and absent-entity columns above and do not interpret the direct intermediate-entity rate.

A.5.3 Averaging states and writing them at fewer positions

We ask what changes when the states of a fact are averaged and written to fewer positions. Table 15 compares these pooled writes with separate writes at all four fact positions. Pooling changes both the number of written positions and the state components, whereas Section 5.2 retains all four positions and adds Content. This supplementary comparison uses replacement writes at layers 0–8; its full-write baseline is measured in the same run, rather than taken from Table 3.

Keeping fewer written positions. Activation writing places one state at each of four fact positions. The *per-fact pooled write* averages the two written states of each fact and writes the resulting state only at that fact's first position. The *per-chain pooled write* averages all four written states and writes one state at the first chain position. Thus, these pooled writes reduce the number of written positions and mix Slot-factor and Content-factor components. They compare compression on the same scaffold; it is not a pure position intervention.

Table 15: Writes at all fact positions and pooled-write proxies on the same two-hop scaffold ($n = 400$). Two-hop and one-hop are correct-answer rates; *unknown* is the rate of answering `unknown` to the question about an absent object.

	Gemma			Llama-3.1		
Write	two-hop	one-hop	unknown	two-hop	one-hop	unknown
Activation writing at all fact positions	.990	1.000	1.000	.757	1.000	1.000
Per-fact pooled write	.018	.255	1.000	.000	.000	1.000
Per-chain pooled write	.000	.000	1.000	.000	.000	1.000
Matched-norm random	.000	.000	1.000	.000	.000	.665

The per-fact comparison leaves some direct answering on Gemma but almost no two-hop answering. Llama-3.1 loses both in this setting. Pooling the whole chain reduces both readouts to zero on both models. These results establish the failure of these pooled writes; because pooling changes both placement and state composition, they do not identify which change caused the loss. The Content-mixing experiment below holds receiving positions fixed to examine a different, more specific change.

A.5.4 RESTORING Q/K/V AFTER A POOLED WRITE

The pooled comparison leaves open whether the loss comes from retrieval address or from the information available when a fact is selected. If damaged keys were sufficient to explain the loss, replacing them with the same item's keys from successful activation writing should improve two-hop answering. We test this proposed restoration, then extend it to Q, V, and head outputs. This experiment uses pooled full states, not the position-preserving Content mixing in Section 5.2; the two interventions need not fail for the same reason.

What is pooled and what is restored. We retain the two connected facts and textual distractor of Appendix A.5, with 400 examples per model and replacement writes at layers 0–8. The paper's pooled condition writes each fact's mean at its first receiving position only. A second condition writes that same mean at both positions of the fact. The first leaves the other position as a placeholder; the second supplies a state to all four positions. Each condition has its own unmodified baseline.

For each question, we first run activation writing for the same item and save the selected binding heads' K and V at the four fact positions and Q at the queried-object position and the final answer position. We rerun the pooled condition and replace the indicated projections with these saved values. K and Q are replaced before RoPE. K/V replacement acts on the native shared K/V groups, so it also affects unselected query heads that share those groups. Q replacement acts on the selected query heads. For the extended comparison, we replace selected heads' outputs before the output projection at the question and answer positions, the final four positions, or all 38 positions. These output replacements supply a later computed result; they do not isolate an individual projection's role.

Scoring and comparisons. Tables 16 and 17 use the original four-candidate score: the correct name, the other name, the intermediate object, and lowercase `unknown`. We compare first-token scores; exact ties select the first candidate. This is not free-generation accuracy. The baseline-readability check instead compares only the three entity candidates and requires the correct name to strictly exceed the two entity rivals on at least half of the direct questions. Gemma passes for both pooled variants (210/400 for first-position-only, 274/400 for both positions); Llama-3.1 does not (94/400 and 108/400). Its patch rows are therefore descriptive, not a test that can reject the recovery hypothesis.

K-only and Q-only rows marked * come from the initial Q/K experiment; the remaining rows come from its Q/K/V extension. Their shared conditions use identical examples and have identical saved candidate logits. This permits the displayed descriptive comparison without treating them as one execution. Restored two-hop accuracy above .50 was required for full recovery; the extension also tracked paired gains of at least 40 examples as partial improvements under the original score.

Table 16: Gemma: restoring projections or head outputs after pooling. Each cell reports correct answers out of 400 under the original four-candidate score. The activation-writing row is the common donor reference.

	First position only		Both positions	
Replacement	Two-hop	Direct	Two-hop	Direct
No replacement	6	104	0	214
K*	6	103	0	221
Q*	7	111	7	223
V	24	111	3	281
K+Q	11	113	10	242
K+V	26	114	3	286
Q+V	49	133	30	313
K+Q+V	60	173	45	323
Head output: question + answer	45	177	36	349
Head output: final four positions	89	275	76	381
Head output: all positions	175	384	146	396
Activation writing	396	400	396	400

Table 17: Llama-3.1: restoring projections or head outputs after pooling. Each cell reports correct answers out of 400 under the original four-candidate score. The activation-writing row is the common donor reference.

	First position only		Both positions	
Replacement	Two-hop	Direct	Two-hop	Direct
No replacement	0	0	0	14
K*	0	0	2	16
Q*	0	0	3	27
V	0	0	2	19
K+Q	0	0	3	28
K+V	0	0	2	20
Q+V	0	2	5	34
K+Q+V	0	6	6	40
Head output: question + answer	0	37	4	77
Head output: final four positions	0	62	5	99
Head output: all positions	2	98	6	132
Activation writing	296	400	296	400

On Gemma, K-only replacement leaves the two-hop count unchanged in either pooled variant. Q and V together give a larger increase than either alone, but no replacement reaches the full-recovery threshold. This observation does not establish that keys are irrelevant: random K in the reverse experiment reduces original two-hop success from 396 to 368 on Gemma and from 296 to 103 on Llama-3.1, so it is not an inert comparison, and the first-position-only condition also receives donor K/V at positions where no state was originally written.

With all Q/K/V replaced, Gemma's two-hop counts rise from 6 to 60 in the first-position-only condition and from 0 to 45 in the two-position condition. These are improvements under the original four-candidate score; neither reaches the full-recovery threshold of two-hop accuracy above .50.

What the controls establish. Table 18 compares same-item donors with values from activation writing for another item at the same positions and with norm-matched random vectors. Replacing the receiver's values with themselves reproduces its saved candidate logits. The full-output replacement has the largest original-score effect, but a substantial part is already obtained by replacing outputs at the final four positions. Moving these outputs cannot establish that the pooled facts themselves have been repaired.

Table 18: Two-hop controls for pooled-write replacements. Each cell reports correct answers out of 400 under the original four-candidate score. Same-item and other-item donors use activation writing; random vectors match donor norms. Self-replacement reproduces the unmodified receiver logits.

Model	Pooled write	Replacement	Same item	Other item	Random
Gemma	First only	K	6	2	1
Gemma	First only	Q	7	14	0
Gemma	First only	K+Q	11	16	0
Gemma	First only	V	24	1	1
Gemma	First only	K+Q+V	60	1	0
Gemma	First only	All head outputs	175	4	0
Gemma	Both	K	0	0	0
Gemma	Both	Q	7	3	0
Gemma	Both	K+Q	10	8	0
Gemma	Both	V	3	0	0
Gemma	Both	K+Q+V	45	0	0
Gemma	Both	All head outputs	146	3	0
Llama-3.1	First only	K	0	0	0
Llama-3.1	First only	Q	0	0	0
Llama-3.1	First only	K+Q	0	0	1
Llama-3.1	First only	V	0	0	0
Llama-3.1	First only	K+Q+V	0	0	0
Llama-3.1	First only	All head outputs	2	0	0
Llama-3.1	Both	K	2	2	1
Llama-3.1	Both	Q	3	2	0
Llama-3.1	Both	K+Q	3	3	2
Llama-3.1	Both	V	2	0	0
Llama-3.1	Both	K+Q+V	6	1	0
Llama-3.1	Both	All head outputs	6	1	0

Table 19: Reverse controls begin with activation writing. Pooled donors use the two-position variant. Counts are correct answers out of 400 under the original four-candidate score.

Model	Replacement	Two-hop	Direct
Gemma	Activation writing	396	400
Gemma	K from two-position pooled write	395	400
Gemma	Random K	368	400
Gemma	Head outputs from pooled write	9	340
Gemma	Random head outputs	233	340
Llama-3.1	Activation writing	296	400
Llama-3.1	K from two-position pooled write	279	399
Llama-3.1	Random K	103	289
Llama-3.1	Head outputs from pooled write	8	120
Llama-3.1	Random head outputs	86	141

In the reverse comparison (Table 19), inserting two-position pooled outputs into the activation-writing condition reduces both direct and two-hop answers. Randomizing those outputs does not eliminate all successful two-hop answers on Gemma, so this comparison does not identify the selected heads as the only route. Together, these controls support a bounded result: the tested donor replacements change answering, but they do not provide the proposed recovery of two-hop reasoning from pooled writes.

A.5.5 Adding an offset at one layer instead of replacing the state

The main mechanism experiments use replacement writes. A separate two-hop run compares that interface with an additive write at one shallow layer. Its replacement baseline is measured in the same run, rather than copied from the main two-hop table:

$$h \leftarrow h + \alpha(\mu - h_{\text{neutral}}).$$

At layer 0, the neutral placeholder state equals the incoming residual state at that target-prompt fact position, so additive writing and replacement are identical. At layer 1, the object states already depend on context, so we can compare adding an offset with replacing the state.

The wrong-value control changes only the name supplied at the first fact's name position. For example, it replaces the Alice offset with an offset for a different name, while keeping the three object offsets unchanged. The chain still connects through the same intermediate item, but its owner has changed. The random control replaces each of the four offsets with a random vector of the same norm. Both controls use layer 1 and $\alpha = 1$, the best-performing additive setting selected from the tested combinations of layer and strength in each model. The table scores both controls against the original owner.

Table 20: Single-layer additive write and controls ($n = 400$). Entries are two-hop correct-answer rates. All controls use the same item set.

Condition	Gemma	Llama-3.1
Replacement, layers 0–8	.985	.770
Additive write, layer 1, $\alpha = 1$	.988	.757
Additive write, layers 0–8, $\alpha = 1$	.003	.003
Wrong-value additive write	.000	.000
Matched-norm random additive write	.000	.000

A single shallow additive write gives accuracy close to replacement in this run. Repeating the additive write over layers 0–8 fails, as do the random and wrong-value controls when scored against the original answer. We also record a higher probability for the substituted name under the wrong-value write, showing that changing the supplied content changes the answer distribution. Under these tested conditions, adding an offset at one layer can also support two-hop answering.

A.5.6 Supplying the same full states at later layers

Successful early writing does not tell us how late the same information can arrive. We test whether delaying activation writing prevents later computation from using the two facts. The primary comparison moves the nine replacement writes from layers 0–8 to layers 18–26 on Gemma and 12–20 on Llama-3.1. Positions, examples, questions, and layer-matched full means μ remain fixed. Each layer receives its own mean, rather than a copy of the layer-0 state. These windows begin at the first selected binding-head layer; they are not after all selected heads. Only these two models were run. The original proposal also predicted $R_K < .20$: if writing misses the computation that builds retrieval addresses, later keys might no longer resemble the text condition. We test this key-similarity prediction separately from answer failure.

We also test the declared intermediate windows and windows after the final selected head. Because fewer layers remain there, the latter comparison uses a matched early window with the same number of writes: one layer on Gemma (L1 versus L41) and seven on Llama-3.1 (L0–6 versus L25–31). Norm-matched random states test whether the intervention has an effect unrelated to the supplied fact. Tables 21 and 22 report all measured windows, keeping these shorter interventions separate from the nine-layer comparison.

Which keys are measured. We record fact-position keys during the same two-hop forward passes, using only selected heads strictly after the last written layer. For each fact position, we average its cosine similarity to text over examples and eligible heads, normalize against the no-write reference as in Appendix A.4, and then average over four positions. Selected heads receive equal weight even when some share a K/V group. As the write window moves, both the eligible heads and the no-write denominator can change; the entries do not form a fixed-head recovery curve. If no selected head remains downstream, R_K is unavailable.

Table 21: Gemma: delayed activation writing and matched random writes. First selected binding head: L18. Correct counts are out of 400. Entity-only two-hop excludes unknown and requires strict superiority over entity rivals; it is not the original answer score. * marks random controls that fail the distractor/absent-item eligibility rule. A dash for R_K means no selected head remains after the window.

	Activation writing				Random write	
Window	Direct	Two-hop	Entity-only	R_K	Direct	Two-hop
Text	400	396	400	1.000	–	–
No write	0	0	0	0.000	–	–
L0–8	400	396	400	0.994	0	0
L2–10	400	396	400	0.991	0	0
L4–12	400	393	400	0.987	0	0
L6–14	400	383	400	0.985	0	0
L12–20	5	50	322	0.978	0	0
L18–26	0	0	148	0.978	0	0
L19–27	0	0	146	0.978	0	0
L24–32	0	0	136	0.983	0	0
L30–38	0	0	99	0.974	0	0
L33–41	0	0	98	–	0	0
L0–0	400	396	400	1.000	0	0
L1–1	400	396	400	1.000	0	0
L18–18	0	0	157	0.959	0	0
L41–41	0	0	0	–	0	0

Table 22: Llama-3.1: delayed activation writing and matched random writes. First selected binding head: L12. Correct counts are out of 400. Entity-only two-hop excludes unknown and requires strict superiority over entity rivals; it is not the original answer score. * marks random controls that fail the distractor/absent-item eligibility rule. A dash for R_K means no selected head remains after the window.

	Activation writing				Random write	
Window	Direct	Two-hop	Entity-only	R_K	Direct	Two-hop
Text	400	268	368	1.000	–	–
No write	0	0	0	0.000	–	–
L0–8	400	296	374	0.981	0*	0*
L1–9	400	319	371	0.978	1*	0*
L2–10	120	60	348	0.974	0*	0*
L4–12	117	90	347	0.967	0*	0*
L8–16	0	0	227	0.972	0	0
L12–20	0	0	173	0.978	0	0
L13–21	0	0	158	0.980	0	0
L16–24	0	0	121	–	0	0
L20–28	0	0	100	–	0	0
L23–31	0	0	100	–	0	0
L0–6	400	277	370	0.987	0*	0*
L12–18	0	0	173	0.978	0	0
L25–31	0	0	43	–	0	0
L0–0	400	268	368	1.000	0*	0*
L1–1	400	273	367	0.999	0*	0*
L12–12	0	0	171	0.955	0	0
L25–25	0	0	45	–	0	0

The requested late windows eliminate both direct and two-hop correct answers under the original four-candidate score, while downstream R_K remains about .978 on both models. Every late answer to these two questions is lowercase unknown. Excluding abstention from the competition leaves 148/400 entity-only two-hop successes on Gemma and 173/400 on Llama-3.1; these are not correct answers under the original score. The late activation-writing and matched-random conditions retain all 400 distractor and absent-item control answers on each model.

Thus, the prediction of failed answers is observed, but the proposed low-key-recovery outcome ($R_K < .20$) is not. Similar key directions do not by themselves establish successful attention selection or answering. The loss is also not specific to two-hop composition, since direct answering fails. In the intermediate scan, a large decrease already occurs between starts L6 and L12 on Gemma and L1 and L2 on Llama-3.1, before the first selected heads. These are intervals between tested windows, not exact boundaries of a binding mechanism. This delayed-supply experiment is distinct from moving already-computed states between positions in Appendix A.3.

A.6 Which positions affect answers when another Content is added

Pooling reduces the number of written positions and changes the states at the same time. Section 5.2 instead keeps the four receiving positions, their Slots, and the full original Content. It adds another Content to each state. This asks whether different contents sharing a state can impair the use of two connected facts even when a direct answer remains available. We first explain how we add Content and compare the added amounts, then vary which positions receive it.

The running example is Alice–umbrella followed by umbrella–tag. The direct question is "Who has the umbrella?"; the two-hop question is "Who has the tag?". The latter connects tag to umbrella and then to Alice.

A.6.1 Estimating the Content to add

We obtain content by subtracting Slot from the mean state μ. At Alice's position, we first average the μ vectors for eight tested names to obtain Slot, then subtract it from Alice's μ to obtain the factor used for mixing, $\Delta c(s, v) = \mu(s, v) - \sigma(s)$. At each of the other three positions, we use eight tested objects in the same calculation. We retain each position's difference directly, so its original Slot plus its own Content equals the full μ.

For example, Alice's position receives $\sigma_A + \Delta c(A, Alice) + g\Delta c(U, umbrella)$, where U is the umbrella position in Fact 1. Its own state is retained, and g controls the additional partner Content. The other positions are treated in the same way within their facts. At $g = 1$, the two Content factors have equal coefficients, not necessarily equal norms. Both facts are written successively at layers 0–8.

A.6.2 Which questions we ask and how we compare the answers

The same 400 examples receive a direct question about Fact 1 and a two-hop question. Candidates are ordered correct owner, competing owner, intermediate item, unknown. The largest first-token logit wins; ties select the first candidate. We use this scoring both when varying the added amount and when leaving the item named in the question unmixed. The fixed candidates include lowercase `unknown`.

The original Content coefficient is 1 for every model. Random conditions replace the added partner Content with a vector matched to its norm before multiplication by g. This compares partner Content with a perturbation of the same size while retaining the original state.

A.6.3 Direct and two-hop answers across the mixing sweep

Table 23 reports the values behind Figure 4. Read each model from its own no-mixing row. Over part of the sweep, two-hop accuracy falls while direct accuracy remains closer to its starting value. At larger amounts, direct answering can also fail, especially on Llama-3.1. Direct answers remain available while two-hop answers decline only over part of the tested range; direct answering is not unaffected at every amount.

Adding random vectors of the same norm tests whether answers decline simply because an extra vector was added. Gemma and Llama-2-Chat retain much more two-hop accuracy with random additions than with partner Content. Llama-3.1 is harmed by random additions too, but partner Content causes a larger loss. Adding another Content matters beyond the size of the added vector, although generic perturbation also contributes on Llama-3.1. As in the main text, Llama-2-Chat is supplementary because it failed the original abstention criterion for unprovided-item controls.

Table 23: Direct and two-hop accuracy with full original Content and added partner Content or matched-norm random perturbations (400 examples per row). The lower panel gives correct counts for the purple curves in Figure 4, using the original scoring.

	Gemma		Llama-3.1		Llama-2-Chat	
g	Direct	Two-hop	Direct	Two-hop	Direct	Two-hop
No mixing	1.000	.990	1.000	.738	.988	.938
.25	1.000	.990	1.000	.680	.983	.905
.5	1.000	.990	.988	.480	.983	.873
.625	1.000	.980	.943	.250	.980	.785
.75	1.000	.953	.660	.055	.958	.610
.875	1.000	.610	.385	.005	.870	.425
1	.968	.198	.170	.000	.770	.265
.25 (random)	1.000	.988	1.000	.728	.985	.933
.5 (random)	1.000	.985	1.000	.648	.988	.930
.625 (random)	1.000	.985	1.000	.603	.985	.920
.75 (random)	1.000	.983	.998	.550	.988	.913
.875 (random)	1.000	.985	.998	.473	.988	.885
1 (random)	1.000	.985	.998	.355	.988	.855

Only the queried position left unmixed: two-hop correct counts out of 400

g	Gemma	Llama-3.1	Llama-2-Chat
0	396	295	375
.25	397	291	369
.5	394	266	353
.625	389	246	345
.75	379	212	338
.875	350	177	326
1	313	136	313

A.6.4 Leaving the Content of the item named in the question unmixed

The preceding experiment changes all four positions at each mixing amount. To ask whether one position can account for much of the loss, we now vary which positions receive partner Content. For the question "Who has the tag?", mixing only tag tests whether that change alone can impair the answer; leaving only tag unmixed tests whether the answer can improve while the other three positions remain mixed.

The comparisons below use full original Content, partner Content added with coefficient 1 where mixed ($g = 1$), and 400 examples per model. "Only queried position mixed" changes the tag position in the running example; "Queried position unmixed" instead mixes Alice and both umbrella positions. Slots, receiving positions, and the layer 0–8 writing range remain fixed.

The purple curves in Figure 4 leave the item named in the question unmixed at each tested amount: the queried entity retains its full original Content and Slot, with no partner Content added, while the other three positions use the same mixture as the baseline. The curves use the original scoring above. Both comparisons reuse the same examples; repeated baseline counts agree at $g = 0, .5, 1$.

Table 24: Accuracy after changing which positions are mixed (%, 400 examples per condition). Direct: "Who has the umbrella?" Two-hop: "Who has the tag?" The direct question tests umbrella to Alice; the two-hop question requires tag to umbrella to Alice. Each mixed position retains its full original Content and receives one full copy of the other endpoint's Content.

	Gemma		Llama-3.1		Llama-2-Chat	
Condition	Direct	Two-hop	Direct	Two-hop	Direct	Two-hop
No mixing	100.00	99.00	100.00	73.75	98.75	93.75
Mix only tag	100.00	32.50	100.00	0.25	98.50	35.50
Mix all four	96.75	19.75	17.00	0.00	77.00	26.50
Leave only tag unmixed	98.75	**78.25**	8.25	**34.00**	58.75	**78.25**

The original score uses the four synthetic-task candidates defined in Appendix A.5. The expanded score also includes the first token of capitalized `Unknown` and requires the correct name to score strictly above all four alternatives; any tie for the highest score counts as failure. For Llama-2-Chat this is a token prefix rather than the complete word, so this comparison checks whether adding another candidate token changes the result; it does not classify the complete answers the model generates. The improvement persists under this expanded comparison, but its magnitude changes substantially for Llama-3.1. Both scores measure the final answer; neither establishes which fact the model retrieves first or next.

Table 25: Two-hop correct counts under the original and expanded candidate sets (400 examples per condition).

	Gemma		Llama-3.1		Llama-2-Chat	
	Original	Expanded	Original	Expanded	Original	Expanded
Unmixed	396	396	295	74	375	368
Only queried position mixed	130	108	1	0	142	94
All four positions mixed	79	55	0	0	106	80
Queried position unmixed	313	294	136	37	313	304

When only the queried position is mixed, direct-answer correct counts remain at $400/400$ for Gemma and Llama-3.1, and change from $395/400$ to $394/400$ for Llama-2-Chat, relative to no mixing. These counts use the original candidate set.

Keeping the same writes and leaving tag unmixed, we also ask the direct question, "Who has the umbrella?". Compared with mixing all four positions, leaving tag unmixed changes accuracy from 96.75% to 98.75% for Gemma, from 17% to 8.25% for Llama-3.1, and from 77% to 58.75% for Llama-2-Chat. Thus direct answering declines for the two Llama models but not for Gemma in this setting. The preserved position is chosen from the known question; this comparison tests a mechanism hypothesis, rather than an automatic compression or recovery procedure.

A.7 Complete list of selected heads

Table 26 lists the selected heads. Layers and heads are indexed from zero; each entry gives a layer followed by its head indices.

Table 26: Selected query heads, grouped by layer.

Model	Layer: head indices
Gemma	$18:13;\ 23:12;\ 24:5;\ 25:7;\ 26:4,12;\ 27:9,15;$ $28:4,8,12,13;\ 29:0,1;\ 30:1;\ 31:14;\ 32:11;$ $33:4;\ 34:14;\ 36:6;\ 37:15;\ 40:10$
Llama-3.1	$12:4;\ 13:18;\ 14:5,7;\ 15:8;\ 16:0,1,22;$ $17:24,27;\ 21:10;\ 22:19;\ 24:27$
Llama-2-Chat	$14:31;\ 15:33;\ 16:39;\ 18:22;\ 19:36$