Do LLMs Bind Episodes? Probing Cross-Episode Parametric Retrieval Through Shared Cues

Francesco Mantovani[1,2,*] **Alberto Eusebio**[1] **Alexis Huet**[1] **Giulio Franzese**[2]
Pietro Michiardi[2] **Dario Rossi**[1] **Zied Ben Houidi**[1,*]

[1]Huawei Technologies Co. Ltd., Paris, France [2]EURECOM, Sophia Antipolis, France

[*]Corresponding authors: mantovani.francesco@huawei.com, zied.ben.houidi@huawei.com

Abstract

Large Language Models (LLMs) fail at a core aspect of human cognition: retrieving related episodes through shared elements[1]. When asked which clubs Ronaldo played for from 2008 to 2010, Claude Opus 4.6 answers Real Madrid, missing Manchester United, despite answering correctly when each period is queried separately. We study this failure through two complementary lenses drawn from cognitive neuroscience: *retrieval-based inference* (test-time recombination) and *integrative encoding* (encoding-time reactivation). "In vivo", we benchmark frontier models (Claude, GPT, Gemini, DeepSeek, Grok, Mistral) and document 4–51% binding failures on verified knowledge. Extended thinking, a proxy for retrieval-based inference, reduces but does not eliminate failures; multiple-choice probing recovers most of them, suggesting the knowledge is accessible under recognition but not easily retrievable. "In vitro", controlled experiments across model families and narrative lengths show that standard remedies offer partial progress: scaling improves binding but still falls short at 70B, extended training enhances binding but requires impractical redundancy, and transfer learning slightly helps with extensive supervision. Logit-lens and forced-prefix analyses on open-weight models suggest correct answers appear in the model's top logits yet fail to surface during generation. Paralleling integrative encoding, we design Recall-based Synthetic Supervision (RSS) which prompts the model to recall past episodes and synthesize binding supervision online. RSS covers 92% of evaluation queries once hallucinated recollections are removed with a ground-truth filter, confirming the information is recoverable from parametric memory. Used as a diagnostic protocol, RSS isolates what still prevents this coverage from translating into accuracy under realistic sequential training: hallucination, forgetting, and evolving supervision signals. We establish episodic binding as an open problem in *parametric memory*.

Code and data: https://github.com/Elpinzeiro/episodic-memory-binding

1 Introduction

Ask Claude Opus 4.6 which clubs Ronaldo played for from 2008 to 2010, and it answers "Real Madrid" missing Manchester United. Ask it about each period separately and both answers come back correctly (Fig. 1). The facts are well known by the model, but it cannot return them together.

This is a failure pattern without a clean name in the ML literature. It is not the Reversal Curse [Berglund et al., 2023]: both facts are retrieved in the forward direction, just not jointly. It is not the Compositionality Gap studied in multi-hop reasoning [Press et al., 2023, Yang et al., 2024a, Biran et al., 2024]: there, models fail to combine intermediate components in latent reasoning

[1]Distinct from the perceptual binding problem; our usage draws on the associative inference literature on memory integration [Bunsey and Eichenbaum, 1996, Zeithamova and Preston, 2010].

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

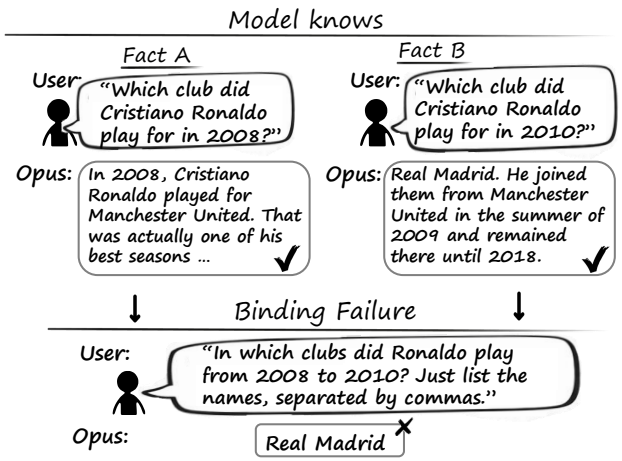


Figure 1: Binding failure in Claude Opus 4.6. The model correctly answers questions about Ronaldo's career at each club separately (upper panels) but fails to integrate this knowledge when asked a binding question spanning both periods (lower panel). Last date tested: 13 April 2026. This is a single illustrative example, not a measured statistic; aggregate failure rates over five repeated trials and eleven prompt formulations are in Sec. 3. Complete prompts and answers in App. B.1. More failure examples can be found in Tab. 4, 5 in App. B.3.

chains, whereas here no intermediate hops are required and all relevant facts are already individually retrievable thanks to a shared cue. It is also different from inverse search [Allen-Zhu and Li, 2023]: the query is not asking for a unique preimage.

We formalize this as *cross-episode retrieval through a shared cue*: unlike catastrophic forgetting, where information is lost, or prior work on knowledge generalization and flexible reuse, binding failure occurs in a setting where each individual fact is independently and reliably retrievable given a shared cue, yet the model fails to retrieve the full **set** of such associated facts with that cue (Sec. 2).

This is a common use case, people often need all the known facts that match a specific query (e.g., "medications prescribed in the last two days," "meetings with X last week"). Such failures are hard to detect, since a partial answer looks complete, and nothing shows that some information is missing.

Cognitive neuroscience has a *broad* name for this capacity and candidate theories of how it is solved by the brain. *Cross-episode binding* [Zeithamova and Preston, 2010] refers to the general ability to link separately encoded episodes through shared elements. This ability was often more narrowly investigated through associative inference paradigms, often in multi-choice settings due to the absence of open-ended language in non-human subjects. In this work, we probe LLMs through shared cues to assess their ability to bind episodes. Cross-episode binding has two proposed mechanisms. The first is **integrative encoding**: When a new experience (e.g. Mary in Tokyo in September) overlaps with a prior one (Mary in Tokyo in March), the shared elements trigger pattern completion to the earlier trace; both are then encoded into a linked representation connected by the shared cue. A subsequent query on that shared cue ("all months Mary was in Tokyo") has the answer readily pre-stored, as opposed to being assembled at test time. The complementary mechanism is **retrieval-based inference**, which keeps episodes stored separately and actively recombines them at test time [Zeithamova and Preston, 2010]. Developmental evidence suggests humans rely on both, with a shift toward more integrative encoding with age [Schlichting et al., 2022].

To systematically study binding, we adopt two complementary perspectives. "In vivo," we benchmark frontier models where their advanced reasoning capabilities allow us to probe *retrieval-based inference* directly. "In vitro," we construct controlled experiments with open-weight models where we can manipulate training, study classic ML remedies and implement a form of *integrative encoding*.

In-vivo analysis (Sec. 3). We design a verification protocol that distinguishes binding failure from simple ignorance. Across Claude, GPT, Gemini, DeepSeek, Grok and Mistral (600K+ API calls), models fail to bind 4% to 51% of episodes they demonstrably know, with variation driven by model capability and query format. Extended thinking, the functional analog of retrieval-based inference, affording additional retrieval and recombination steps, reduces failures in most models (e.g. Sonnet 4: 35%→12%) but does not eliminate them. Multiple-choice probing, an easier setting, substantially improves binding, suggesting the knowledge is accessible under recognition but not retrieved.

In-vitro analysis (Sec. 4). We first ask whether standard training on disjoint episodes per se can induce binding. Using synthetic multi-event narratives, we train models across families (Llama, Mistral, Qwen, GPT; 4k+ GPU hours) on single-event QAs (Q: "*Where was Mary on March 5th?*" A: "*Tokyo*") and evaluate on multi-event retrieval (Q: "*List all times Mary was in Tokyo*"). Despite

reliably learning each fact across epochs (models achieve high single-event accuracy), they collapse on binding queries. Standard remedies offer only partial gains: scaling provides some relief but falls well short of resolving the problem even at 70B, grokking-style extended training [Power et al., 2022] improves binding only after thousands of epochs, and transfer learning shows gains with extensive supervision but remains insufficient to fully address the task. Logit-lens and forced-prefix analyses on open-weight models suggest that correct answers often appear in the model's top logits yet fail to surface during generation (App. D.4).

Integrative encoding analog (Sec. 5). Paralleling integrative encoding, where prior memories are reactivated during new learning, we design Recall-based Synthetic Supervision (RSS), a protocol that continuously prompts the model to recall past episodes and synthesize binding supervision on the fly. Despite RSS generating high-quality supervision (92% coverage of evaluation queries), this does not translate into accuracy under sequential training. Used as a diagnostic protocol, RSS exposes concrete obstacles, each of independent interest, that we isolate and analyze.

Reproducible controlled framework. We release our frontier model benchmark, and the synthetic episodic narrative pipeline that enables clean, controlled study of binding in isolation from confounds such as ambiguity, knowledge gaps, or cue overload. Data and code here Mantovani et al. [2026].

2 Problem definition

We first formalize cross-episode retrieval. In cue-based retrieval, a *cue* is a partial specification that triggers recall of associated *traces* (mirroring how *prompts* elicit *continuations* in LLMs).

Let $\mathcal{M}$ be a model and $\mathcal{E} = \{E_i\}_{i=1}^{N}$ a set of learned episodes (that is used interchangeably with the term *events* in this article, and is referring to e.g., facts about an athlete's career, or a character's journey). For any retrieval cue q (e.g., "*Ronaldo's clubs 2008–2010*" or "*All times Mary was in Tokyo*"), we define the **ground truth retrieval set** as all episodes matching the cue's constraints, $Y(q) = \{E_i \in \mathcal{E} : E_i \text{ matches } q\}$, and the **model's retrieval** as $\hat{Y}(q) = \mathcal{M}(q)$.

Verified knowledge. We say $\mathcal{M}$ *knows* episode E_i if, for every cue q that uniquely identifies it ($Y(q) = \{E_i\}$), the model retrieves it correctly: $\hat{Y}(q) = Y(q)$. Sec. 3 operationalizes this through repeated prompting, while Sec. 4 ensures this via fine-tuning on all questions matching an episode.

Binding and binding failure. $\mathcal{M}$ exhibits *episodic binding* for cue q if it *retrieves* all matching episodes when multiple exist: $\hat{Y}(q) = Y(q)$ whenever $|Y(q)| > 1$. A *binding failure* occurs when $\mathcal{M}$ knows each episode in $Y(q)$ individually yet fails to retrieve them jointly: $(Y(q) \cap \hat{Y}(q)) \subsetneq Y(q)$. This differs from catastrophic forgetting: the constituent facts are retained but jointly irretrievable. Throughout, correctness is recall-based (used in Sec. 3.1 and Sec. 4): a response is correct iff $\hat{Y}(q) \supseteq Y(q)$, i.e., recall $= 1$ (all expected episodes present; extra items allowed) in vivo, a query is a binding failure only if this fails on a majority of repeated trials.

This definition is deliberately behavioral and asymmetric. Behavioral, because we study model behavior through the tokens it generates: a binding failure means verified knowledge was not retrieved under a shared cue, whatever the reason (Sec. 3.1 and App. B.4.1 test several candidate causes such as cardinality and salience). Asymmetric, because we draw conclusions from failures not *successes*: a model could pass a binding query by having memorized the grouped question itself (e.g., "Ronaldo's clubs 2008–2010" as one more fact) rather than by combining two individually-verified facts at retrieval time. This can lead us to overestimate successful binding, but cannot explain a binding failure. Where we control the training data (Sec. 4), we remove this ambiguity by construction.

The matching criterion can be arbitrarily complex. We focus here on simple cases where cues share one attribute: (i) *Cue composition* (Sec. 3) where cues share an entity (e.g. Ronaldo), with other attributes constrained to a range (e.g., years 2008–2010), and (ii) *Exhaustive retrieval* (Sec. 4) where cues simply share an attribute with no additional constraint (e.g., all appearances of Mary).

Distinction from related failure modes. Our setting is structurally distinct from previously studied retrieval failures. The Compositionality Gap [Press et al., 2023], and related work on multi-hop knowledge composition, concerns single-target queries ($|Y(q)| = 1$) requiring an unstated intermediate hop ($q \to b \to a$), where failure stems from not composing the chain. Binding failure concerns multi-target queries ($|Y(q)| \geq 2$) where each $E_i \in Y(q)$ is independently retrievable under a uniquely-identifying cue, yet $\hat{Y}(q) \subsetneq Y(q)$ under the shared cue.

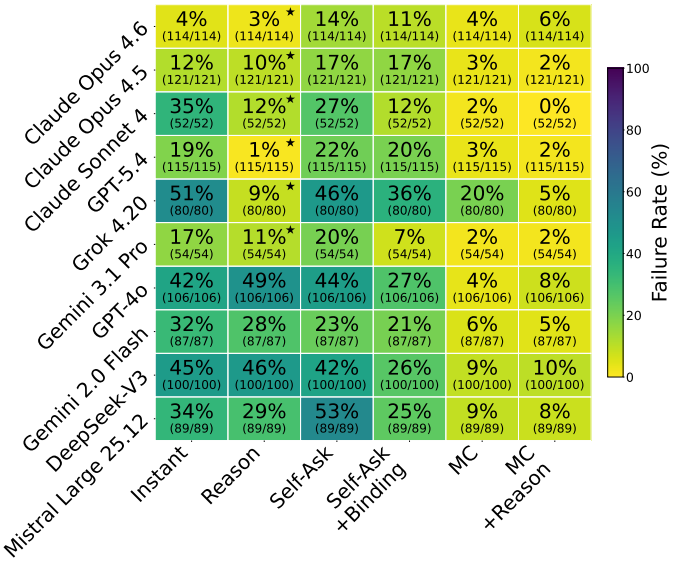


Figure 2: Binding failures across frontier models, evaluated under different prompting strategies. Cells report the percentage of binding failures (incorrect answer in at least 3 of 5 trials) over each model's verified known facts and sample size (N = binding questions with verified facts). Models marked with ★ have extended thinking enabled. Details in App. B.4.

3 "In-vivo" analysis of frontier models

3.1 Benchmarking frontier models

If binding emerges from scale once constituent facts are reliably encoded, frontier models should exhibit it; we examine whether this is the case. We use a two-stage protocol: *(i) verify* that the model knows each individual fact, then *(ii) test binding* by asking a question that requires integrating them.

Source datasets. We construct a public benchmark for evaluating binding failures by combining two complementary sources. The first is a filtered subset of TempLAMA [Dhingra et al., 2022], which provides athletes paired with their activities across different years. To broaden the scope beyond this domain, we add a newly curated dataset of public-event cases (see App. B.3 for details). Each binding instance is built around: (i) a shared cue (e.g., person, organization, location, or event type), (ii) a ground-truth set of 2–4 temporally distinct episodes[2] matching that cue, (iii) single-episode verification questions, and (iv) a multi-episode binding question requiring retrieval of the full set.

Known-facts verification protocol For each fact, we ask three types of verification questions, each repeated five times.[3] The first is a yes/no question (e.g., "Did Ronaldo play for Manchester United in 2008?"); the second provides the temporal context and asks for recall (e.g., "Which team did Ronaldo play for in 2008?"); and the third reverses the query (e.g., "In which years did Ronaldo play for Manchester United?"). We classify a fact as **known** only if the model answers correctly in at least 3 of 5 trials for each question type.[4] We use temperature 0.3 to induce variation across sampling trials. Note that stricter 4/5 and 5/5 thresholds give similar results (App. B.4.1).

Evaluating binding We ask five times each binding question whose facts are all verified as known; the **failure rate** is the percentage of binding failures (at least 3 of 5 trials incorrect; Sec. 2) among each model's verified known facts. Under direct questioning without reasoning instructions (*Instant*), binding failures range from 4% (Claude Opus 4.6) to 51% (Grok 4.20) across all tested models (Fig. 2). The large majority of failures are real binding failures rather than refusals, except for GPT-4o and Gemini 3.1 Pro (App. B.4.4). Precision stays at 87–96% and models list about as many items as expected (App. B.4.5).

[2]We restrict all instances to 2–4 episodes to avoid cue overload, where failures may reflect general retrieval difficulty rather than binding errors; failure rates do not increase with set size (App. B.4.2).

[3]To isolate parametric knowledge, we instruct models via the system prompt to rely only on internal knowledge and prohibit external sources.

[4]We evaluate responses using Claude Haiku 4.5 as an automated judge, validated on 200 samples against two independent human annotators (Cohen's $\kappa = 0.95$ human–human; $\kappa = 0.76$–0.77 human–judge; see App. B.2).

3.2 Retrieval-based inference: can reasoning help?

If episodes are stored separately, retrieving them jointly requires recombination at test time, which is what retrieval-based inference does. Extended thinking, its functional analog in LLMs, gives models additional computation steps to attempt this recombination. We test whether reasoning reduces binding failures using: (i) direct questions without reasoning, (ii) reasoning, either via native extended thinking when available through the API, or via prompt-based instructions otherwise, (iii) structured decomposition with SelfAsk [Press et al., 2023] and SelfAsk-Binding, our adaptation that decomposes retrieval into sequential binding-specific follow-up questions, and (iv) multiple-choice (MCQ) variants that test recognition rather than free recall. The complete prompt set is in App. B.5, Lists. 15–25.

Native extended thinking (available via API for six models) generally reduces failures: sharply for models that fail often (e.g., Grok 4.20: 51%→9%), little for those that already fail rarely. Prompt-based reasoning for the other four models has small, inconsistent effects, mostly not statistically significant (App. B.5.1). Structured chain-of-thought strategies work better when the decomposition targets binding: SelfAsk-Binding lowers failures for 7 of 10 models (e.g., DeepSeek-V3: 45%→26%), whereas SelfAsk as originally proposed [Press et al., 2023] raises them for 6 of 10 (e.g., Mistral Large: 34%→53%). A full quantitative analysis of all eleven prompting strategies is provided in App. B.4; qualitative examples in App. B.6.

In contrast, switching to multiple-choice questions (MCQs) reduces binding failures for 9 of 10 models (Gemini 3.1 Pro: 17%→2%, Claude Sonnet 4: 35%→0% with reasoning; GPT-4o: 42%→4%), with no model above 20% under MCQ, confirming that models can *recognize* correct bindings even when they fail to *retrieve* them independently. This dissociation between recognition and recall motivates our controlled experiments: if the knowledge exists but direct retrieval fails, what training conditions enable reliable binding?

4 "In-vitro" analysis of open-weight models

Sec. 3 showed binding does not reliably emerge from organic large-scale training, raising a complementary question: under what conditions, if any, can it be induced? Since frontier models are black boxes that allow us only to *verify* existing knowledge, we construct a controlled framework using synthetic episodic narratives and fine-tunable models. We train on individual events and test binding on joint queries. A model that merely stores individual QA pairs without integrating them would fail on multi-event queries despite perfect single-event accuracy.

4.1 Synthetic dataset and evaluation

We adapt the episodic memory benchmark from Huet et al. [2025] to study parametric binding. The core experimental setup mirrors our frontier model protocol, but now *verification* becomes *learning*: we fine-tune models on single-event QA pairs (ensuring they learn individual facts), then evaluate on multi-event questions (testing whether binding emerges). We modify the original benchmark in two key ways (App. C). First, we focus on parametric memory, distinguishing between single-event and multi-event question performance to measure whether models can retrieve all related episodes after learning them individually. Second, while the original benchmark used multi-paragraph chapters for long-context evaluation, we instead generate simpler and shorter events, which allows to better isolate the binding challenge from potential side effects arising from, e.g., context length limitation.

Each event (e.g. List. 1) in our narratives encodes a tuple (t_i, s_i, ent_i, c_i) where t_i denotes time (e.g., "March 5th"), s_i denotes space (e.g., "Tokyo"), ent_i denotes the entity involved (e.g., "Mary"), and c_i denotes the content or action (e.g., "painting").

We generate synthetic narratives of $N \in \{10, 30, 100\}$ events using the original controlled multiplicity approach with geometric sampling: this ensures entities, dates, and spaces naturally recur across multiple events, creating authentic binding challenges (e.g. "Mary" might appear in 1, 3, or 6+ events distributed across different times and locations). We also adopt their three narrative universes (everyday events in New York, imaginary world news, and sci-fi events) in order to preserve event structure (App. C.1 illustrates specific examples, cf. Lists. 33–35). This synthetic generation provides ground truth for any episodic question, enabling precise evaluation without real-world ambiguity.

Table 1: Cue patterns and their binding requirements. Patterns with unique answers require no binding; patterns with multiple matches test episodic binding (detailed examples in App. C.2).

Cue pattern	Example question	Binding requirement
$(*, s, e, c)$	"What day did Mary paint in Tokyo?"	None (unique answer)
$(*, *, e, *)$	"List all Mary's activities"	Entity binding (multiple events)
$(*, s, *, *)$	"What happened in Tokyo?"	Location binding (multiple events)
$(*, *, e, c)$	"When did Mary paint?"	Entity-action binding (multiple events)

Legend: k=1 k=2 k=3–5 k=6+

Events	SEQ	MEQ	Evaluated SEQ/MEQ[5]
10	227 / 227 / 227	50 / 50 / 50	(121, 17, 21, 0)
30	520 / 520 / 520	149 / 149 / 149	(126, 42, 53, 18)
100	1406 / 1418 / 1418	411 / 413 / 413	(135, 78, 81, 63)

Table 2: Statistics across 3 universes for SEQ ($k = 1$) and MEQ ($k > 1$). Plot shows the distribution of evaluated subset by ground truth set size k.

```
Episode 1 On June 14, 2025, at Washington Square Park, Mila Gonzalez captivated a small crowd. She
     eloquently explained method acting techniques, demonstrating with impromptu performances. The
     audience watched, mesmerized by her expertise.
```

Listing 1: One episode example.

Binding tasks and Evaluation The episodic memory benchmark models episodic recall as cue-based retrieval, where partial event information (a cue) triggers memory recall. A cue is any combination of elements from the event tuple (t_i, s_i, ent_i, c_i), with asterisks denoting wildcards. For example, the cue $(*, s, *, *)$ asks for all events at location s, while $(*, *, ent, *)$ asks for all events involving entity ent. Tab. 1 shows how different cue patterns create different retrieval challenges.

To isolate the binding challenge, we distinguish two question types.

Single-event questions (SEQ) have answers found in exactly one event: e.g. "What day did Mary paint in Tokyo" has a unique answer, since by benchmark construction, an entity cannot be at two places the same day.

Multi-event questions (MEQ) require retrieving all events matching a cue: e.g. "When did Mary paint" requires finding every event where Mary is painting.

Following Sec. 2, fine-tuning on SEQ ensures verified knowledge ($|Y(q)| = 1$), while MEQ evaluation tests binding ($|Y(q)| > 1$). Tab. 2 summarizes dataset statistics across narrative scales.[5]

Evaluation of binding success We use recall $|\hat{Y} \cap Y|/|Y|$ over the model's prediction $\hat{Y}$, extracted via an LLM-as-judge validated against human annotations (App. C.2). Recall isolates omission, the failure mode we study, from hallucination, a well-documented side effect of fine-tuning on new factual knowledge [Gekhman et al., 2024]; we report precision in App. B.4.5 and App. C.4. As in Sec. 2, an answer is correct only if recall equals 1; our main metric is the **percentage of correct answers**, stratified by ground-truth set size $k \in \{1, 2, 3\text{-}5, 6+\}$ (partial recall in App. C.3).

4.2 Do standard remedies help?

We test three hypotheses: (1) larger models bind better, (2) extended training enables binding, and (3) multi-event supervision transfers binding across narratives. Each shows partial progress; none resolves the problem. *Logit-lens analyses suggest the bottleneck is retrieval, not knowledge: correct answers are in the model's top logits but are not generated* (Fig. 4, App. D.4).

Experimental protocol. We train models with a multi-epoch protocol on SEQ. We evaluate Llama models (3B, 8B, 13B, 70B), Qwen2.5 (1.5B, 3B, 7B, 14B) and Mistral (7B, 12B) fine-tuned with LoRA, varying narrative length from 10 to 100 events. To test if binding failures generalize beyond open-weight model families, we run limited experiments with GPT-4.1 variants, using fewer training epochs due to budget constraints (thus, this should not be read as a comparison between open-weight

[5]We evaluate on representative samples of 38/113/222 MEQ for the 10/30/100 event configurations, respectively, due to LLM-as-judge evaluation costs.

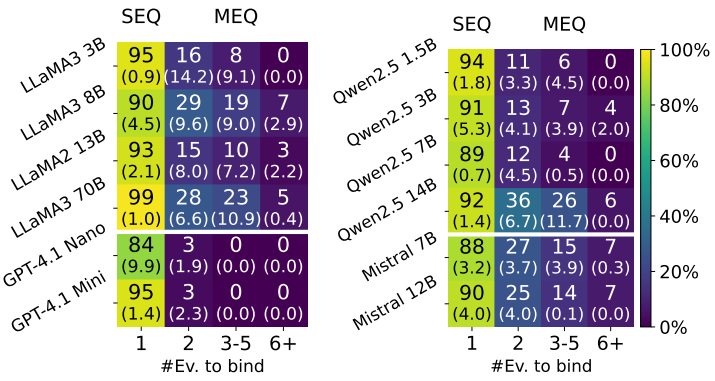


Figure 3: Average correct answers percentage performance defined in Sec. 4.1 across three narrative types: everyday events in New York, imaginary world news, and sci-fi events. Each cell shows the mean across 3 sizes (10, 30, 100 events) and narrative universes, with standard deviation in parentheses (not all size–universe combinations were run for every model, due to GPU constraints).

and GPT models).[6] We track performance on CommonsenseQA [Talmor et al., 2019] during training to detect and prevent overfitting.

4.2.1 Does scale help?

As intended, all models achieve near-perfect single-event retrieval (84–99%) after SEQ training, without degrading CommonsenseQA performance. Multi-event retrieval is harder: performance drops with the number of events to retrieve and with narrative size (Fig. 3). Larger models consistently perform better (e.g., on k=2 questions, Llama improves from 16% at 3B to 28% at 70B). However, the gap to full binding remains large: for $k \geq 6$ events, even the best model (Mistral-12B) reaches only 7%. Whether this trend continues at larger scales remains an open question.

4.2.2 Integrative encoding: can recall-based supervision help?

To investigate why SEQ-trained models fail at binding, we inspect the next-token distribution at the final position of the question prompt. For each MEQ with $k = 2$, we read the 8 most likely next tokens (logit lens) and count how many of the two entities have their first token among them (2, 1 or 0). For example, for the protagonists of the events related to Educational Workshop (Henry Reed and Levi Rodriguez), we check for the first token of each name. We run Llama-3-8B on event sizes 10, 30 and 100 in three narrative universes (397 MEQ in total), and Mistral-7B and Qwen-7B on event size 10 (16 MEQ each; setup, example and cross-family results in App. D.4).

The first tokens of both entities are usually among the top-8, even when the final answer is wrong (Fig. 4a): in all MEQ at event size 10, and still in 67–77% at event size 100. The omitted entity is therefore often within reach but not produced. To test this, we force the answer to start with the first token of the omitted entity (e.g., "Levi") and let the model continue. This forced prefix turns 21.7% of all MEQ (86/397) into correct answers (Fig. 4b; counts per setting in App. D.5). This suggests that failures partly reflect token selection during generation rather than missing knowledge.

4.2.3 Can extended training and transfer learning induce binding?

We investigate whether binding can emerge through grokking-like [Power et al., 2022] extended training. Training Llama-3-8B on 100-event narratives (Fig. 5a), MEQ accuracy rises from ∼20% at 50 epochs to ∼36.5% at 3000 epochs, while CommonsenseQA accuracy drops from 72% to 64%. This confirms binding can improve with single-event data, but at a cost to general knowledge. Continuing the same run to 9000 epochs does not sustain this gain: MEQ falls back to ∼29%, single-event accuracy stays near ceiling (∼99%), and CommonsenseQA collapses to ∼18% (App. D.6).

We next test whether multi-event supervision on some narratives can transfer binding to an unseen narrative. For each evaluation setting, we train on a set of narratives with both SEQ and MEQ supervision, while holding out one independent narrative on which we provide only SEQ supervision (Fig. 5b; full protocol in App. D.7). With minimal supervision (3 supervised narratives out of 4), MEQ accuracy on the held-out narrative does not improve; at 9 out of 10, it improves to ∼26% (from ∼20% without supervision), averaged across event sizes 10 and 30. Scaling supervision further, to

[6] To rule out LoRA-specific limitations, we also evaluate Llama-3-8B with full fine-tuning (FFT). Hyperparameters are selected via grid search (App. D.1). As a sanity check, all models achieve high MEQ recall under Train(SEQ+MEQ), confirming that failures in all other regimes reflect binding limitations, not insufficient capacity (App. D.3).

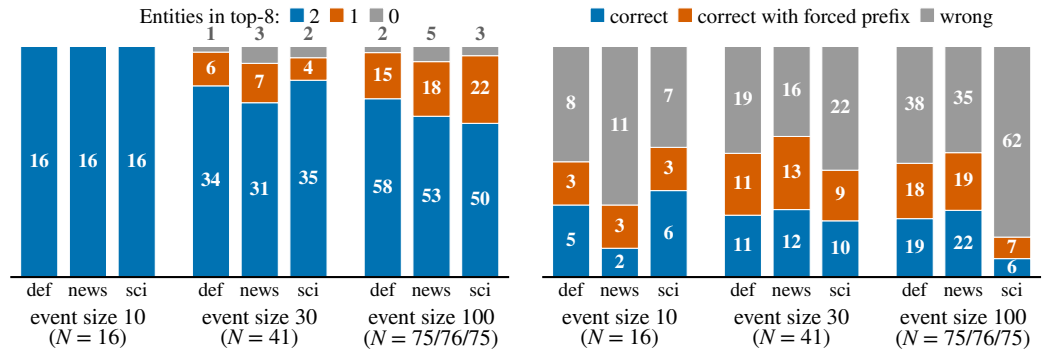


Figure 4: Logit-lens analysis of Llama-3-8B on MEQ with $k = 2$, for event sizes 10, 30 and 100 and three narrative universes (def, news, sci). Left: MEQ with 2, 1 or 0 correct entities among the top-8 first tokens. Right: MEQ answered correctly, correctly only with the forced prefix, or wrongly. Each bar covers all N MEQ of one setting; numbers are MEQ counts (also in Tab. 20).

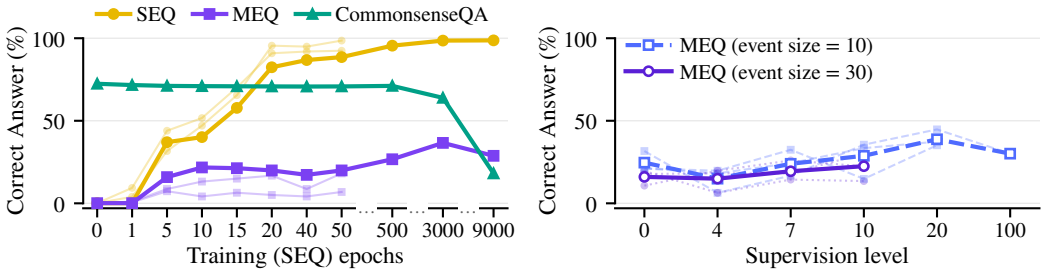


Figure 5: (Left) Extended training (Llama-3-8B, 100-event narratives): SEQ, MEQ and CommonsenseQA accuracy vs. epochs; three runs up to 50 epochs, one (bold) continued to 9000. (Right) Transfer learning (Mistral-7B): MEQ accuracy on a held-out narrative as the proportion of supervised (SEQ+MEQ) narratives increases; faint lines: narrative universes.

20 and then 100 narratives (one held out), does not turn this partial transfer into a solution: held-out MEQ accuracy reaches ∼39% at 20 narratives, then falls back to ∼30% at 100, while held-out single-event accuracy stays at 85–95% throughout. More supervision does not reliably teach binding.

In summary, each remedy improves binding but none resolves it within the regimes we test. This motivates asking whether the model's own memory can generate the missing supervision.

5 Integrative encoding as a diagnostic protocol

Standard remedies improve binding but do not resolve it. They all require access to the full dataset at training time and cannot operate in continual-learning settings such as post-training, where new episodes arrive sequentially and retraining on the entire corpus is not feasible.

In integrative encoding, the arrival of a new experience that overlaps with a prior one triggers reactivation of the earlier trace, linking both into a joint representation [Zeithamova et al., 2012, Schlichting and Preston, 2015], making the process inherently continual. We operationalize this principle for LLMs through Recall-based Synthetic Supervision (RSS), used as a diagnostic protocol to test whether this mechanism works in LLMs and where it breaks: when a new event arrives, RSS (i) queries the model to recall prior episodes involving the same entity, and (ii) creates synthetic questions integrating the recalled and new information.

Note that unlike retrieval-augmented generation (RAG), which stores and retrieves information externally, our focus is on what can be learned and bound *within model parameters*.

5.1 Recall-based synthetic supervision (RSS)

Fig. 6 illustrates the RSS protocol (See App. E for a full description). When event E_k arrives (e.g., "Mary was in London on March 5th"), the system does not only train on this isolated fact. Instead, the

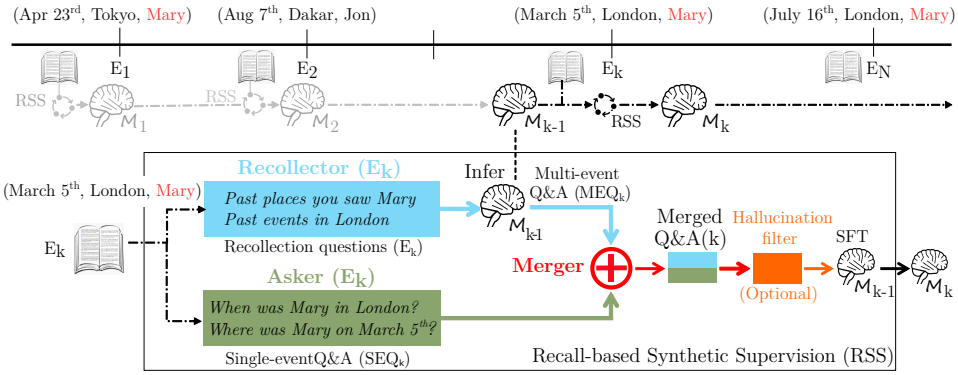


Figure 6: **Recall-based Synthetic Supervision (RSS) protocol.** When a new event E_k arrives, the **Recollector** queries the current model $\mathcal{M}_{k-1}$ for related past episodes. The **Asker** generates single-event Q&As about E_k. The **Merger** combines recalled episodes with the current event to synthesize multi-event Q&As (e.g., "List all times Mary was in London" $\rightarrow$ "March 5th, July 16th"), creating training data that connects the new fact to the existing episodic memory network.

protocol uses a frontier LLM as a teacher to generate supervision, while the *learner* model $\mathcal{M}$ is both trained and queried for its parametric memory to build the supervision signal. (see App. E.1.6 for a step-by-step example).

The **(i) Asker** uses the teacher LLM to generate single-event Q&As from E_k's textual description, synthesizing diverse questions that cover the event's key elements (t_i, s_i, ent_i, c_i)(List. 39 in App. E.1.1).

The **(ii) Recollector** queries the learner $\mathcal{M}_{k-1}$ about past episodes related to the incoming event. The teacher LLM generates diverse memory probes, including temporal sequencing and chronological reconstruction queries, which the learner answers from its parametric memory (List. 42 in App. E.1.2).

The **(iii) Merger** uses the teacher LLM to combine recalled episodes with the current event into multi-event Q&As. It filters out negative responses (e.g., "not mentioned in my training"), identifies shared entities, and creates questions spanning multiple episodes: e.g., if the current event is "Mary visited Tokyo in September" and the learner recalls "Mary visited Tokyo in March," the Merger generates "List all times Mary was in Tokyo" $\rightarrow$ "March, September" (List. 43 in App. E.1.3; templated variants in App. E.1.2, per-component ablations in App. E.1.5).

Lastly, we optionally plug a **Hallucination Filter** that removes confabulated details by comparing learner's recall to ground truth, separating binding failures from hallucinations (List. 44 in App. E.1.4).

This synthetically merged supervision data then updates the model via supervised fine-tuning, teaching it not just the new fact but its connections to the existing episodic memories. Note that unlike traditional replay methods which require storing past episodes, RSS leverages the model's own parametric memory as both storage and retrieval means.

5.2 Experimental conditions

When considering single-epoch training, batch-by-batch processing is essentially continual learning since each example is seen only once. This is a common regime for foundation model training, where exhaustive multi-pass learning is impractical at scale. Our static experiments (Sec. 4.2) served as controls to rule out confounds such as forgetting; here we examine training under this dynamic.

We compare six conditions (Fig. 7): (1) **Sequential-SEQ**, single-event Q&As only with no replay; (2) **Sequential-SEQ + O-LoRA**, adding orthogonal subspace constraints [Wang et al., 2023] to mitigate forgetting (App. E.1.8) (3) **RSS**, the full protocol with multi-event supervision (4) **RSS-No-Hallucination**, with hallucination filtering to isolate binding from recall noise; (5) **RSS-Aggregated**, which collects all Merger-generated multi-event Q&As from the full RSS-No-Hallucination run, and trains in a static multi-epoch setting, thereby isolating supervision quality from sequential forgetting (App. E.1.7); and finally (6) **Coverage**, which measures how many evaluation questions are answerable from the generated supervision alone, providing an upper bound that isolates supervision completeness from model learning capacity (App. E.1.9). Conditions (1)–(3) are fully automated; (4)–(6) use ground truth as diagnostic controls to isolate specific failure modes.

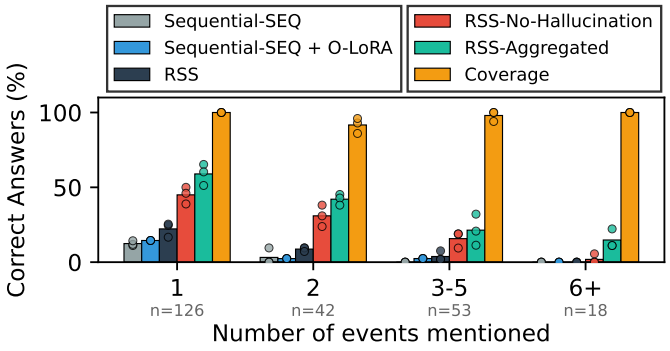


Figure 7: Performance across conditions (left group: fully automated; right group: diagnostic controls using ground truth) as a function of the number of events k matching a question (30-event narratives, Llama-3-8B). Each dot is a different narrative universe. Evaluation set in Tab. 2.

5.3 Under what conditions does binding succeed?

Fig. 7 compares supervision conditions on 30-event narratives with Llama-3-8B. The Sequential-SEQ baseline (LLM-generated questions without replay) reaches only 12% recall on single-match questions and $\sim$0% on multi-event retrieval, far below the static multi-epoch setting (Sec. 4.2); this gap, even on SEQ, likely reflects forgetting across episodes. O-LoRA slightly improves SEQ but barely affects MEQ, further evidence that binding is distinct from forgetting (App. E.1.8).

RSS nearly doubles SEQ performance to 22% and yields non-zero performance (4–9%) for 2–5 events where the baseline fails, suggesting mitigation of cross-episode forgetting.

Next, removing hallucinations (an orthogonal issue), RSS-No-Hallucination achieves $\sim$45% for single-match questions, $\sim$30% for two-match, and $\sim$15% for 3-5 matches. This isolates the binding from retrieval noise, suggesting that RSS can generate useful supervision when the Recollector's output is accurate. Coverage analysis shows that the union of all questions generated by the process, when hallucinations are factored out, covers 92% of the MEQ evaluation. Coverage is, by definition, an overlap metric between generated training questions and the evaluation set (App. E.1.9).

However, training on this aggregated data (RSS-Aggregated) underperforms the ideal upper bound provided by Coverage. Analyzing the aggregated data, we observe that they contain multiple versions of the same question, each with a different answer as a function of the number of episodes seen.

As a summary, the degradation at higher event counts reflects three factors: with hallucinations, recall errors compound as events accumulate. Without hallucinations, the remaining challenges are catastrophic forgetting of earlier episodes and evolving supervision (the same question has different answers as more events arrive). Each is a known open problem; solving them is what would make RSS a workable solution.

6 Conclusion

We reveal a limit in how LLMs retrieve episodic knowledge. Cross-episode binding through a shared cue is a distinct failure mode of LLM parametric memory: each episodic fact is individually retrievable, yet the model fails to return the full set bound by that cue. Frontier models (Claude, GPT, Gemini, DeepSeek, Grok, Mistral) fail to bind between 4% and 51% of episodes they verifiably know, with extended thinking (the functional analog of retrieval-based inference) reducing but not closing the gap. Multiple-choice probing recovers most failures indicating the knowledge is accessible under recognition but not spontaneously retrieved. In controlled experiments across Llama (3B–70B), Qwen, Mistral, and GPT-4.1, multi-epoch training on single-event QAs reliably teaches each fact in isolation yet binding queries remain largely unresolved despite partial gains from scaling and extended training. On open-weight models, logit-lens and forced-prefix analyses suggest that correct answers often appear in the top logits but are not surfaced during generation. Paralleling integrative encoding, Recall-based Synthetic Supervision (RSS) draws binding supervision from model memory, covering 92% of queries offline, confirming the information is in principle recoverable. Under sequential training, this coverage does not translate into accuracy.

Our work has limitations. The in-vitro experiments use synthetic narratives with explicit episodic elements, and we do not claim this captures the full complexity of natural text. This failure is distinct from forgetting, reversal, compositionality gaps, and inverse search. RSS decomposes the remaining gap into three independent, well-studied obstacles: hallucination, forgetting, and evolving supervision. Solving them would make RSS a workable solution; as it stands today, it is a diagnostic protocol.

Author Contributions

F.M. led the implementation and ran most experiments and analyses. He formalized binding failures (Sec. 2); designed the in-vivo benchmark pipeline, known-facts verification protocol, binding evaluation and main prompting variants (Sec. 3); adapted the episodic memory benchmark to parametric binding and ran the fine-tuning, scaling, extended-training and transfer experiments and the logit-lens and forced-prefix analyses (Sec. 4); implemented RSS and its diagnostic controls (Sec. 5); released the code; and wrote the appendix.

Z.B.H. identified binding failure in LLMs and conceived the study. He found the first binding failures in frontier models and outlined how to search for them systematically, proposed the in-vitro paradigm of training on single-episode questions and testing on cross-episode binding questions, and designed RSS (Asker, Recollector, Merger) as an analog of integrative encoding, together with its diagnostic controls. He supervised F.M. and A.E. throughout and wrote the main text.

A.E. designed and curated the Custom dataset, designed SelfAsk-Binding and implemented the SelfAsk baseline, ran a large part of the in-vivo experiments, carried out the statistical, refusal-versus-incomplete and precision analyses, and served as one of the two human annotators used to validate the LLM judge.

A.H. extended the problem definition into a unified formulation of the in-vivo and in-vitro settings, proposed measuring how retention and binding evolve over RSS episodes, found further failure examples including the one in Fig. 1, and proposed the statistical tests (McNemar).

G.F., P.M. and D.R. provided critical feedback. All authors edited and reviewed the manuscript.

Acknowledgments and Disclosure of Funding

We thank Michele Ferrero for serving as a human annotator in the validation of the LLM judge (App. B.2).

References

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.1, knowledge storage and extraction. *arXiv preprint arXiv:2309.14316*, 2023.

Lukas Berglund, Meg Tong, Max Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. The reversal curse: LLMs trained on" a is b" fail to learn" b is a". *arXiv preprint arXiv:2309.12288*, 2023.

Eden Biran, Daniela Gottesman, Sohee Yang, Mor Geva, and Amir Globerson. Hopping too late: Exploring the limitations of large language models on multi-hop queries. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14113–14130, 2024.

M Bunsey and H Eichenbaum. Conservation of hippocampal memory function in rats and humans. *Nature*, 379(6562):255–257, 1996.

Payel Das, Subhajit Chaudhury, Elliot Nelson, Igor Melnyk, Sarath Swaminathan, Sihui Dai, Aurélie Lozano, Georgios Kollias, Vijil Chenthamarakshan, Soham Dan, et al. Larimar: Large language models with episodic memory control. *arXiv preprint arXiv:2403.11901*, 2024.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. *arXiv preprint arXiv:2104.08164*, 2021.

Bhuwan Dhingra, Jeremy R Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W Cohen. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273, 2022.

Mohamed Elsayed and A Rupam Mahmood. Addressing loss of plasticity and catastrophic forgetting in continual learning. *arXiv preprint arXiv:2404.00781*, 2024.

Zorik Gekhman, Gal Yona, Roee Aharoni, Matan Eyal, Amir Feder, Roi Reichart, and Jonathan Herzig. Does fine-tuning LLMs on new knowledge encourage hallucinations? In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7765–7784, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.444. URL https://aclanthology.org/2024.emnlp-main.444/.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Jianheng Huang, Leyang Cui, Ante Wang, Chengyi Yang, Xinting Liao, Linfeng Song, Junfeng Yao, and Jinsong Su. Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal. *arXiv preprint arXiv:2403.01244*, 2024.

Alexis Huet, Zied Ben Houidi, and Dario Rossi. Episodic memories generation and evaluation benchmark for large language models. *International Conference on Learning Representations*, 2025.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114 (13):3521–3526, 2017.

Andrew K Lampinen, Arslan Chaudhry, Stephanie CY Chan, Cody Wild, Diane Wan, Alex Ku, Jörg Bornschein, Razvan Pascanu, Murray Shanahan, and James L McClelland. On the generalization of language models from in-context learning and finetuning: a controlled study. *arXiv preprint arXiv:2505.00661*, 2025a.

Andrew Kyle Lampinen, Martin Engelcke, Yuxuan Li, Arslan Chaudhry, and James L McClelland. Latent learning: episodic memory complements parametric learning by enabling flexible reuse of experiences. *arXiv preprint arXiv:2509.16189*, 2025b.

Francesco Mantovani, Alberto Eusebio, Alexis Huet, Giulio Franzese, Pietro Michiardi, Dario Rossi, and Zied Ben Houidi. Code and data for Do LLMs Bind Episodes? Probing Cross-Episode Parametric Retrieval Through Shared Cues. https://github.com/Elpinzeiro/episodic-memory-binding, 2026. [Online].

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*, 2022.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711, 2023.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. iCaRL: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

Margaret L Schlichting and Alison R Preston. Memory integration: neural mechanisms and implications for behavior. *Current opinion in behavioral sciences*, 1:1–8, 2015.

Margaret L Schlichting, Katharine F Guarino, Hannah E Roome, and Alison R Preston. Developmental differences in memory reactivation relate to encoding and inference in the human brain. *Nature human behaviour*, 6(3):415–428, 2022.

Robyn Speer, Joshua Chin, and Catherine Havasi. ConceptNet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge, 2019. URL https://arxiv.org/abs/1811.00937.

Johannes Treutlein, Dami Choi, Jan Betley, Samuel Marks, Cem Anil, Roger B Grosse, and Owain Evans. Connecting the dots: LLMs can infer and verbalize latent structure from disparate training data. *Advances in Neural Information Processing Systems*, 37:140667–140730, 2024.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Is multihop qa in DiRe condition? measuring and reducing disconnected reasoning. *arXiv preprint arXiv:2005.00789*, 2020.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Boshi Wang and Huan Sun. Is the reversal curse a binding problem? uncovering limitations of transformers from a basic generalization failure. *arXiv preprint arXiv:2504.01928*, 2025.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuan-Jing Huang. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10658–10671, 2023.

Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. Do large language models latently perform multi-hop reasoning? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10210–10229, 2024a.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380, 2018.

Zitong Yang, Neil Band, Shuangping Li, Emmanuel Candes, and Tatsunori Hashimoto. Synthetic continued pretraining. *arXiv preprint arXiv:2409.07431*, 2024b.

Dagmar Zeithamova and Alison R Preston. Flexible memories: differential roles for medial temporal lobe and prefrontal cortex in cross-episode binding. *Journal of Neuroscience*, 30(44):14676–14684, 2010.

Dagmar Zeithamova, Margaret L Schlichting, and Alison R Preston. The hippocampus and inferential reasoning: building memories to navigate future decisions. *Frontiers in human neuroscience*, 6:70, 2012.

Guanxiong Zeng, Yang Chen, Bo Cui, and Shan Yu. Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, 1(8):364–372, 2019.

Appendix overview

Section	Title	Contents
A	Related work	Extended literature review and positioning
B	Binding failures in frontier models	Verification protocol, datasets, prompt sensitivity, complete results
C	A controlled framework	Episodic world model, narrative generation, evaluation, partial recall, precision
D	Do standard remedies help?	Hyperparameters, capacity verification, extended training, grid search, mechanistic interpretability (Logit Lens), transfer
E	Investigating supervision requirements	RSS protocol, ablations, O-LoRA baseline, additional plots
F	Generative AI usage	Disclosure of AI tools used in this work
G	Benchmark reproduction guidelines	Dataset access and reproduction instructions

A Related work

Parametric knowledge failures Recent work demonstrates that LLMs fine-tuned on statements like "B's mother is A" fail to answer "Who is A's son?" [Berglund et al., 2023], and that knowledge encoded in narrative form may not be accessible via Q&A without sufficient training augmentation [Allen-Zhu and Li, 2023]. In both cases, learned information becomes inaccessible under query variation, but the failure modes differ. The reversal curse arises because training on "B's mother is A" only establishes the forward mapping; the model never observes the reverse direction, so it cannot infer "A's child is B." The fact is encoded directionally, not relationally. In contrast, our binding failure occurs when two facts are fully retrievable in isolation but the model fails to retrieve the binding of them. Yet paradoxically, LLMs can compose facts learned separately, inferring a city's identity from distances to Rome and London learned through fine-tuning [Treutlein et al., 2024]. This capability requires massive redundancy in the training set, differently from what we have in our experiments.

Note that our definition of binding is different from the neural binding problem which was recently used by [Wang and Sun, 2025] as a potential explanation of the reversal curse in LLMs.

Reasoning helps, but does not suffice While explicit reasoning can improve performance in some cases (Sec. 3.1) it does not resolve all failures of multi-fact integration. Prior work on multi-hop and compositional question answering, such as [Yang et al., 2018] and [Trivedi et al., 2022], studies whether models can combine multiple pieces of information to answer a single question. Related analyses of disconnected reasoning in multi-hop QA [Trivedi et al., 2020] similarly show that models struggle to link facts within a given context. In contrast, our work targets a different and complementary setting: we study factual knowledge stored parametrically in large language models and show that failures occur even when all required facts are individually known. By verifying individual fact recall before testing joint recall, we isolate a failure mode that standard multi-hop benchmarks cannot detect, revealing a limitation in parametric knowledge access rather than in-context reasoning.

Limits of continual learning for episodic knowledge integration During standard pretraining the model processes data sequentially, encountering related facts at vastly different times. Facts about an entity in batch 1 must be retained and eventually integrated with facts appearing in batch 100,000 making batch-by-batch optimization a form of continual learning over declarative knowledge. Traditional continual learning research focuses on preventing catastrophic forgetting (CF) across discrete tasks. Several approaches have been proposed to mitigate catastrophic forgetting. These include Elastic Weight Consolidation [Kirkpatrick et al., 2017], experience replay (e.g. [Rebuffi et al., 2017]) and utility-based gradient updates [Elsayed and Mahmood, 2024]. While effective at preserving prior knowledge and mitigating one of the three issues that hinder binding, these methods were designed for a pre-LLM era. Most of these methods were not designed for continual factual integration. Their focus instead was on ensuring that a model trained on MNIST could still perform well after learning CIFAR-10, or that a sentiment classifier trained on movie reviews could adapt to

product without forgetting. With LLMs, however, the challenge shifts: models must not only retain facts learned at different times but also perform flexible reasoning and generalization across them. Knowledge editing methods like ROME and MEMIT [De Cao et al., 2021, Meng et al., 2022] as well as memory-augmented approaches like Larimar [Das et al., 2024] target localized factual updates but do not address sequential integration of related episodes. While more recent approaches enforce orthogonality constraints on weight updates to prevent task interference [Zeng et al., 2019, Wang et al., 2023], with O-LoRA adapting this principle to parameter-efficient fine-tuning, which we apply to our episodic setting with scarce results (App. E.1.8). We include these methods as diagnostic baselines to demonstrate that classical continual learning remedies, while effective for task retention, do not address episodic factual integration in LLMs.

Synthetic supervision Achieving the massive redundancy required for fact composition [Treutlein et al., 2024] is challenging in standard pretraining. Prior work generates such synthetic supervision once, before training begins, either through retrieval-augmented generation that restructures entity relations, as in EntiGraph [Yang et al., 2024b], or through in-context reasoning that produces bridging examples to connect related facts [Lampinen et al., 2025a]. RSS adopts the same high-level idea of synthetic supervision, but operates in a different regime: instead of augmenting a static dataset upfront, RSS generates synthetic data continually at runtime as new episodes arrive. Related rehearsal-based methods also rely on parametric retrieval, but their objective is different. Self-Synthesized Rehearsal [Huang et al., 2024] and retrieval-based approaches [Lampinen et al., 2025b] use rehearsal to preserve task-level knowledge and prevent forgetting across tasks, whereas RSS uses model-generated rehearsal to preserve and integrate factual knowledge, explicitly connecting related episodes learned at different points in time.

B Binding failures in frontier models

This appendix documents binding failures observed in frontier language models, supplementing the analysis in Section 3. We provide complete prompts and responses to enable reproducibility and allow readers to assess the evidence independently.

Section B.1 presents the introductory example from Figure 1, showing how Claude Opus 4.6 knows individual facts about Cristiano Ronaldo's career but fails to integrate them when queried jointly. Section B.2 details the verification protocol introduced in Section 3 for distinguishing binding failure from simple ignorance. Section B.3.1 provides qualitative samples from the Athletes Dataset, and Section B.3.2 presents analogous results on the Custom Dataset. Finally, Section B.5 investigates prompt sensitivity and whether reasoning instructions can mitigate binding failures.

B.1 Binding failures introduction

This appendix provides the complete prompts and responses underlying the binding failure documented in Figure 1. We include this material to enable full reproducibility and to illustrate the precise nature of the binding gap.

We test Claude Opus 4.6 with questions about Cristiano Ronaldo's football career, the model's responses in Lists. 2 and 3 contain all the information needed to answer the binding question: "In which clubs did Ronaldo play from 2008 to 2010?" The correct answer requires retrieving that Ronaldo was at Manchester United until 2009, then moved to Real Madrid. However, as shown in List. 4, when posed with this binding query, Opus responds only "Real Madrid," omitting Manchester United entirely. This is binding failure: the model verifies both facts individually but fails to integrate them when the query spans both episodes.

```
User: Which club did Cristiano Ronaldo play for in 2008?
LLM: In 2008, Cristiano Ronaldo played for Manchester United. That was actually
    one of his best seasons. He was there from 2003 to 2009.
```

Listing 2: Claude Opus 4.6 correctly places Ronaldo at Manchester United in 2008.

```
User: Which club did Cristiano Ronaldo play for in 2010?
LLM: Real Madrid. He joined from Manchester United in the summer of 2009.
```

Listing 3: Claude Opus 4.6 correctly places Ronaldo at Real Madrid in 2010.

```
User: In which clubs did Ronaldo play from 2008 to 2010? Just list the names,
    separated by commas
LLM: Real Madrid
```

Listing 4: Binding failure: despite knowing both facts (Listings 2–3), Opus fails to retrieve Manchester United when the query spans both tenures.

Interpretation. The response in List. 3 explicitly states that Ronaldo "joined from Manchester United," and List. 2 confirms he played there until 2009. The binding query in List. 4 asks about 2008–2010, a period that spans both clubs. The model's failure to include Manchester United demonstrates that binding failure is not about missing knowledge but about the inability to retrieve multiple related episodes through a single query. This motivates our systematic investigation of binding in controlled experimental settings (Section 4).

B.2 Verification protocol

This subsection details the verification protocol introduced in Section 3.1.

Our protocol tests each episode through three complementary question types, each repeated five times to ensure consistency. We set temperature to 0.3 to enable meaningful variation across sampling trials: temperature 0 would yield deterministic outputs, making repeated trials redundant, while higher temperatures would introduce excessive randomness. At 0.3, a model that genuinely knows a fact should answer correctly across most trials, while uncertain or unreliable knowledge is exposed through inconsistent responses.

The three verification questions probe the same fact from different angles: binary confirmation (List. 5) tests whether the model can verify a fully specified fact; team extraction (List. 6) tests retrieval given entity and time; and year extraction (List. 7) tests retrieval given entity and team. An episode is considered *verified* only if the model answers correctly on at least 3 out of 5 trials for all three question types. This strict threshold ensures we only diagnose binding failure for episodes the model reliably knows.

```
"Did {entity name} play for {team} in {year}? Answer yes or no."
```

Listing 5: Binary verification prompt

```
"Which team did {entity name} play for in {year}?"
```

Listing 6: Team extraction prompt

```
"In which year did {entity name} play for {team}?"
```

Listing 7: Year extraction prompt

Since some APIs do not allow explicitly disabling web search (unlike the browsing interface), we prepend the following system prompt to enforce reliance on the model's internal parametric knowledge (Listing 8).

```
"You must answer ONLY using your internal parametric knowledge from training. You
    cannot access external information, web search, or any outside sources."
```

Listing 8: System prompt for parametric-only evaluation

B.2.1 Human validation of the LLM judge

All in-vivo failure rates rely on an automatic judge (Claude Haiku 4.5) that decides whether a response contains every expected item. To check whether judge errors affect our failure rates, we compare its judgments with human annotations. We sample 200 model responses, stratified across all ten models and both datasets (Athletes, Custom). The sample contains 100 verification items (team/period recall, e.g., "Did X play for team Y?") and 100 binding items. Within each group, 50 responses were judged correct and 50 incorrect, giving 100 judge-labeled correct responses and 100 judge-labeled incorrect responses overall. Two annotators label every response independently and blind: they see only the

question, the expected answer set and the response, never the judge's verdict. They apply the same rule as the judge (List. 9): a response is correct if and only if it contains all expected items, extra items never make it wrong, and surface variants such as "UK" and "United Kingdom" count as the same item. If the judge is reliable, it should agree with each annotator nearly as often as the two annotators agree with each other. Tab. 3 shows that the two annotators agree on 97.5% of responses ($\kappa = 0.95$) and that the judge agrees with each of them on about 88% ($\kappa = 0.76$–0.77), which is substantial agreement. In 18 of the 100 responses that the judge labeled as failures, both annotators accepted the answer.

Table 3: Agreement between two blind human annotators and the automated judge (Claude Haiku 4.5), on 200 stratified in-vivo responses; human–judge rows use the 190 whose response is unchanged in the final run. κ is Cohen's kappa; raw agreement in parentheses. Both annotators overturned 18 of the 100 judge-labeled failures.

Comparison	Verification	Binding	Overall
Annotator A vs. Annotator B (inter-annotator)	$\kappa = 0.959$ (98.0%)	$\kappa = 0.940$ (97.0%)	$\kappa = 0.950$ (97.5%)
Annotator A vs. judge	$\kappa = 0.78$ (89.0%)	$\kappa = 0.73$ (86.7%)	$\kappa = 0.76$ (87.9%)
Annotator B vs. judge	$\kappa = 0.82$ (91.0%)	$\kappa = 0.71$ (85.6%)	$\kappa = 0.77$ (88.4%)

```
Evaluate if each response correctly matches expected answer.

LENIENCY RULES (mark as CORRECT if):
- Response contains ALL expected items (extra items are OK)
- Common name variations: "UK"="United Kingdom"="Britain", etc.
- Date format variations: "June 24, 2009"="24 June 2009"="2009-06-24"
- Minor typos or spelling variations

STRICTNESS RULES (mark as INCORRECT if):
- Missing any expected item
- Wrong entity entirely
- Response says "I don't know" or refuses
```

Listing 9: In-vivo judge prompt (verification and binding).

B.3 Datasets

This appendix details the construction of the two evaluation datasets that form our benchmark: Athletes and Custom (see Sec. 3.1 for a summary). Because the datasets differ in their composition and collection procedure, we describe them separately here and in the following subsections whenever relevant.

TempLAMA Athletes. Professional athletes provide a natural source of binding cases because their careers often involve temporally bounded affiliations with multiple teams. We derive our athlete evaluation set from TempLAMA [Dhingra et al., 2022]. Starting from athlete–team temporal facts, we filter for athletes with 2–3 teams, rank candidates by Wikidata sitelink count as a rough popularity proxy, and retain the top 150 athletes. For each retained athlete, we construct single-episode verification questions, such as which team the athlete played for in a given year, and binding questions whose time interval spans multiple teams. A binding instance is included in the evaluation only when the model first verifies the corresponding single-team facts individually.

Curated real-world set (Custom). To test whether binding failures extend beyond athlete careers, we construct an additional set of real-world binding cases spanning political news, award announcements, business events, scientific missions, public appointments, and other temporally grounded public events. We prompt frontier LLMs with web search to generate candidate cases across 20 thematic categories, then filter them for structural validity, factual accuracy, and compliance with the binding definition in Sec. 2. We discard cases whose cue is too broad, whose answer set is ambiguous, whose facts depend on subjective interpretation, or whose constituent episodes cannot be reliably verified. From 564 structurally valid candidates, 105 pass all filters.

Benchmark provenance and independence. The Custom set is built with LLMs, and some of them belong to the same families as the models we evaluate. A natural worry is that a model could fail, or pass, more easily on questions written by its own family. Three design choices limit this risk.

Construction. Every item encodes real public facts that anyone can check (TempLAMA items come from Wikidata; Custom items are curated public events). Web-enabled LLMs only collect and format candidate items, and web access is used when building the benchmark, never when evaluating models. Of the 105 items that pass all filters, 92 (87.6%) were generated by Claude and 13 (12.4%) by GPT-5. Most importantly, an item counts against a model only if that same model passes all single-fact verification questions for it (Sec. 3.1): the generator chooses which questions are asked, but never decides whether a given model fails them. *No family effect.* Failures appear in all six evaluated model families (4–51% under Instant), including four families involved in neither construction nor judging (Grok, Gemini, DeepSeek, Mistral: 17–51% under Instant). The TempLAMA Athletes half of the benchmark uses no LLM generation at all and shows the same pattern, so it serves as an LLM-free control. *Judging.* The judge only compares a response with a fixed gold answer set, and it agrees well with human annotators (App. B.2.1, $\kappa = 0.76$–0.77).

B.3.1 TempLAMA athletes dataset

Our Athletes evaluation dataset is derived from TempLAMA [Dhingra et al., 2022].

Source data. TempLAMA contains 50,310 temporally scoped facts from the 2020 Wikidata snapshot, covering the period 2010–2020. Each entry follows the template "`[Entity] plays for _X_.`" with year, answer, and Wikidata identifier fields.

Filtering. We first filtered only athletes using the Wikidata ID. We then grouped entries by athlete and retained those with 2–3 unique team affiliations across consecutive years. This ensures a meaningful binding challenge while avoiding cue overload from too many transitions.

Popularity ranking. We queried the Wikidata API for each athlete's sitelink count (Wikipedia pages across languages) and retained the top 150 by this score.

Query generation. For each athlete we generate: (i) three fact verification questions per team (yes/no confirmation, team retrieval, year retrieval), and (ii) one binding question requiring all teams within the career span. Query phrasing adapts to sport type from Wikidata ("play for" for team sports, "race for" for motorsports, "represent" otherwise).

Binding failure examples. Tab. 4 illustrates binding failures on the Athletes dataset. Each cell shows the number of correct responses out of 5 trials (e.g., 5/5 means all trials correct). We consider an episode *verified* if the model answers correctly on $\geq$3/5 trials for all verification questions. A *binding failure* occurs when verified facts are not retrieved together: the model passes individual verification ($\geq$3/5) but fails the binding question ($\geq$3/5 incorrect). For example, in the first row, the model correctly answers all verification questions about Agüero playing for both Manchester City (2020) and Atlético Madrid (2010) with 5/5 accuracy, yet fails 5/5 times to list both teams when asked the binding question.

B.3.2 Custom dataset

We constructed our Custom dataset through a multi-stage pipeline combining automated generation, model-based filtering, and manual evaluation, some examples here Tab. 5 illustrates binding failures on the Custom dataset.

Automated example generation. We generated candidate test cases using a structured prompt (List. 10) that instructs the model to create knowledge binding examples. The prompt specifies the required structure including: (1) individual fact verification questions (yes/no, entity retrieval, date retrieval), (2) binding questions in multiple formats (list-based and multiple-choice), and (3) metadata for validation.

To ensure diversity across topics and time periods, we instantiated the generation prompt with all combinations of 20 thematic domains and 10 time periods of one year each, as shown in Tab. 6.

We employed two frontier models with web search capabilities for generation: GPT-5 and Claude Opus 4.5. Each model was prompted to generate batches of 5 examples per theme-period combination, with web search enabled to ensure factual accuracy. High-performing domains included political news (e.g., Obama's international visits), award announcements (e.g., Nobel Prize winners), and business news (e.g., company acquisitions).

Table 4: Sample binding failure cases from the Athletes dataset. All individual fact verification questions pass, yet the binding question fails. Scores indicate correct responses out of 5 trials.

Known Fact 1	Known Fact 2	Binding Failure
Agüero → Man City (2020) • Did Agüero play for Man City in 2020? 5/5 • Which team did Agüero play for in 2020? 5/5 • In which period did Agüero play for Man City? 5/5	**Agüero → Atlético (2010)** • Did Agüero play for Atlético in 2010? 5/5 • Which team did Agüero play for in 2010? 5/5 • In which period did Agüero play for Atlético? 5/5	**Q:** List ALL teams Agüero played for (2010–2020) **Fails: 5/5** **A:** [Manchester City] **Expected:** [Manchester City, Atlético Madrid]
Simmons → 76ers (2020) • Did Simmons play for 76ers in 2020? 5/5 • Which team did Simmons play for in 2020? 5/5 • In which period did Simmons play for 76ers? 5/5	**Simmons → LSU (2015)** • Did Simmons play for LSU in 2015? 5/5 • Which team did Simmons play for in 2015? 5/5 • In which period did Simmons play for LSU? 5/5	**Q:** List ALL teams Simmons played for (2015–2020) **Fails: 5/5** **A:** [Philadelphia 76ers] **Expected:** [76ers, LSU Tigers]
Lloris → Lyon (2010) • Did Lloris play for Lyon in 2010? 5/5 • Which team did Lloris play for in 2010? 5/5 • In which period did Lloris play for Lyon? 5/5	**Lloris → Tottenham (2020)** • Did Lloris play for Tottenham in 2020? 5/5 • Which team did Lloris play for in 2020? 5/5 • In which period did Lloris play for Tottenham? 5/5	**Q:** List ALL teams Lloris played for (2010–2020) **Fails: 4/5** **A:** [Tottenham] **Expected:** [Lyon, Tottenham]
Di Natale → Udinese (2016) • Did Di Natale play for Udinese in 2016? 5/5 • Which team did Di Natale play for in 2016? 5/5 • In which period did Di Natale play for Udinese? 5/5	**Di Natale → Italy (2010)** • Did Di Natale play for Italy in 2010? 4/5 • Which team did Di Natale play for in 2010? 5/5 • In which period did Di Natale play for Italy? 5/5	**Q:** List ALL teams Di Natale played for (2010–2016) **Fails: 3/5** **A:** [Udinese] **Expected:** [Udinese, Italy]
Judge → Yankees (2020) • Did Judge play for Yankees in 2020? 5/5 • Which team did Judge play for in 2020? 5/5 • In which period did Judge play for Yankees? 5/5	**Judge → Fresno St. (2011)** • Did Judge play for Fresno St. in 2011? 5/5 • Which team did Judge play for in 2011? 5/5 • In which period did Judge play for Fresno St.? 5/5	**Q:** List ALL teams Judge played for (2011–2020) **Fails: 5/5** **A:** [New York Yankees] **Expected:** [Yankees, Fresno State]

Structural validation. Generated examples were first filtered for structural validity, removing cases that failed JSON parsing, had missing required fields (facts, binding questions, expected answers), or contained malformed prompts or metadata inconsistencies. After structural filtering, we retained **564 candidate test cases**.

Factual knowledge verification. We evaluated all 564 examples on the five models available at construction time. We retained only examples where *at least one of the tested frontier models* (Claude Opus 4.5, Claude Sonnet 4, GPT-4o, Gemini 2.0 Flash and DeepSeek-V3) demonstrated knowledge of the underlying facts with $\geq$60% accuracy on the individual fact verification questions. This pruning step reduced the dataset to **326 verified examples** containing facts that models demonstrably know.

Quality-based filtering. We applied automated quality evaluation using Claude Opus 4.5 as a judge model (see Sec. B.3.3), followed by a manual curation phase. The evaluation targeted three categories of problematic examples: (1) *Duplicates*: examples testing essentially the same event or fact; (2) *Self-referential examples*: cases where binding questions assumed context from the test case (e.g., "based on the facts above", "in this case") rather than being standalone queries about real-world knowledge; (3) *Answer-in-question examples*: cases where the binding question revealed the expected answer. Finally, we performed manual review to ensure strict compliance with our Cue Composition definition of binding (Sec. 2). The complete pipeline yielded **105 high-quality test cases**.

B.3.3 Custom dataset prompts

This section documents the prompts used for automated Custom dataset generation and quality assessment.

Dataset generation. List. 10 shows the prompt used to generate the custom dataset.

```
# Knowledge Binding Q/A Dataset Generation Prompt
```

Table 5: Sample binding failure cases from the Custom dataset. All individual fact verification questions pass, yet the binding question fails. Scores indicate correct responses out of 5 trials.

Known Fact 1	Known Fact 2	Binding Failure
Microsoft → Activision (Jan 2022) • Did Microsoft announce acquisition of Activision Blizzard in Jan 2022? 5/5 • Which company announced $68.7B acquisition of Activision Blizzard? 5/5 • When did Microsoft announce intent to acquire Activision Blizzard? 5/5	**Take-Two → Zynga (Jan 2022)** • Did Take-Two announce acquisition of Zynga in Jan 2022? 5/5 • Which company announced $12.7B acquisition of Zynga? 5/5 • When did Take-Two announce acquisition of Zynga? 5/5	**Q:** List ALL companies that announced gaming acquisitions over $10B in January 2022 **Fails: 5/5** **A:** [Microsoft] **Expected:** [Microsoft, Take-Two Interactive]
Pfizer-BioNTech (Dec 11, 2020) • Did Pfizer-BioNTech vaccine receive FDA EUA in Dec 2020? 5/5 • Which vaccine was first to receive FDA EUA on Dec 11, 2020? 5/5 • When did Pfizer-BioNTech receive FDA EUA? 5/5	**Moderna (Dec 18, 2020)** • Did Moderna vaccine receive FDA EUA in Dec 2020? 5/5 • Which vaccine received FDA EUA one week after Pfizer? 5/5 • When did Moderna receive FDA EUA? 5/5	**Q:** List ALL COVID-19 vaccines that received FDA EUA in December 2020 **Fails: 5/5** **A:** [Pfizer-BioNTech] **Expected:** [Pfizer-BioNTech, Moderna]
Kuczynski → Peru (Jun 2016) • Did Pedro Pablo Kuczynski win the presidential election in Peru in 2016? 5/5 • Who won the presidential election in Peru in June 2016? 5/5 • When was the second round of the Peruvian presidential election? 5/5	**Medina → Dom. Rep. (May 2016)** • Did Danilo Medina win the presidential election in the Dominican Republic in 2016? 5/5 • Who won the presidential election in the Dominican Republic in May 2016? 5/5 • When was the Dominican Republic presidential election held in 2016? 5/5	**Q:** List ALL politicians who won presidential elections in Latin America in May–June 2016 **Fails: 5/5** **A:** [] **Expected:** [Kuczynski, Medina]
Chandrayaan-3 (Jul 2023) • Did ISRO launch Chandrayaan-3 in July 2023? 5/5 • Which ISRO mission landed on Moon's south pole in Aug 2023? 5/5 • When did Chandrayaan-3 land on the Moon? 5/5	**Aditya-L1 (Sep 2023)** • Did ISRO launch Aditya-L1 in September 2023? 5/5 • Which ISRO mission was launched to study the Sun from L1? 5/5 • When did ISRO launch Aditya-L1? 5/5	**Q:** List ALL major deep space missions launched by ISRO in 2023 **Fails: 5/5** **A:** [] **Expected:** [Chandrayaan-3, Aditya-L1]
IBM → Red Hat (Oct 2018) • Did IBM announce acquisition of Red Hat in Oct 2018? 5/5 • Which company did IBM acquire for $34B in Oct 2018? 5/5 • When did IBM announce its acquisition of Red Hat? 5/5	**SAP → Qualtrics (Nov 2018)** • Did SAP announce acquisition of Qualtrics in Nov 2018? 5/5 • Which company did SAP acquire for $8B in Nov 2018? 5/5 • When did SAP announce its acquisition of Qualtrics? 5/5	**Q:** List ALL companies acquired in enterprise software deals >$8B in Q4 2018 **Fails: 5/5** **A:** [Red Hat] **Expected:** [Red Hat, Qualtrics]

Table 6: Themes and time periods used for Custom dataset generation. Each theme-period combination was used to generate 5 candidate examples.

Dimension	Values
Themes (20)	Company Acquisitions; Presidential Visits Sports Championships; Movie Awards Music Releases & Awards; Tech Product Launches Space Exploration; Political Events & Elections Scientific Breakthroughs; Celebrity Events Natural Disasters; Criminal Cases Video Game Releases; Literary Awards Corporate Bankruptcies; Infrastructure Projects TV Series Events; Automotive Launches Art Market Events; Royal Events
Periods (10)	2015–2024 (one year each)

```
## Objective
Generate 5 new, high-quality test cases for evaluating whether large language
    models can accurately bind multiple related facts together. Focus on
    **{theme_name}** within the time period **{period_range}**.

## Task Overview
You will create test cases that:
1. First verify that an LLM knows individual facts through targeted questions
2. Then assess whether the LLM can correctly associate and bind these facts
    together
3. Support multiple question formats (yes/no, entity retrieval, date retrieval)
4. Include binding validation through open-ended and multiple-choice formats

## Theme-Specific Guidelines
**Theme**: {theme_name}
**Description**: {theme_description}
**Focus Area**: {theme_fact_focus}
**Entity Type**: {theme_entity_type}
**Time Period**: {period_range}
**Sources**: {theme_sources}

### Example for this Theme:
{theme_example_facts}

## Core Requirements

### 1. Entity Binding Constraint
- Maximum 4 entities per test case
- Each entity should have multiple associated facts ({theme_fact_focus})
- Focus on {theme_name} as the primary use case

### 2. Question Quality Standards
- Questions must have only ONE correct, unambiguous answer
- Answers must be easily verifiable through reliable online sources
- Avoid questions with multiple valid interpretations
- Each fact should be independently verifiable

### 3. Fact Verification
- **CRITICAL**: Verify every single fact online before including it
- Use the sources listed above
- Ensure all facts fall within **{period_range}**
- Document the source URL/reference for each fact

### 4. Knowledge Binding Challenge
**IMPORTANT**: Create test cases where:
- Individual facts are somewhat known (pass verification)
- But binding ALL facts together is challenging (likely binding failure)
- Consider facts that might cause difficulty:
  * Events happened close together in time
  * Entities are lesser-known but verifiable
  * The context requires associating multiple specific details

## Test Case Structure - JSON FORMAT
Each test case must follow this structure:
{
  "test_cases": {
    "Descriptive Case Name": {
      "source": "Primary verification source",
      "id": "unique_lowercase_identifier",
      "topic": "Brief description of what's being tested",
      "facts": {
        "fact_a_yesno": {
          "prompt": "Did [entity] [action] in [period]? Answer yes or no.",
          "expected": "Yes",
```

```
          "entity": "Entity Name",
          "date": "Month Day, Year",
          "type": "yes_no"
        },
        "fact_a_entity": {
          "prompt": "Which [entity type] [specific detail]?",
          "expected": "Entity Name",
          "type": "entity_retrieval"
        },
        "fact_a_date": {
          "prompt": "When did [specific event]?",
          "expected": "Month Day, Year",
          "type": "date_retrieval"
        }
      },
      "binding": {
        "prompt": "List ALL [entities]. List only [entity type].",
        "expected": ["Entity1", "Entity2", "Entity3"],
        "prompts": {
          "raw": "List ALL [entities]. Provide answer as JSON list.",
          "no_reason": "List ALL [entities]. As JSON list. NO reasoning!",
          "reason": "List ALL [entities]. As JSON list. Reason first."
        }
      },
      "binding_multichoice": {
        "prompts": {
          "raw": "Which [entities] [context]? Select
    option:\nA.\nB.\nC.\nD.\nReturn ONLY the letter.",
          "no_reason": "... Return ONLY the letter. Do NOT write anything else!",
          "reason": "... Return ONLY the letter. Think independently first."
        },
        "options": {"A": "...", "B": "...", "C": "...", "D": "..."},
        "expected_answer": "C"
      },
      "num_items": 3,
      "date_range": ["Start Date", "End Date"],
      "notes": "Why binding might be challenging"
    }
  }
}

## Quality Checklist
- All facts occurred within {period_range}
- All facts verified through online sources
- Dates are specific (format "Month Day, Year")
- Maximum 4 entities per test case
- All questions have unambiguous answers
- Multiple-choice includes realistic distractors
- Only ONE option is fully correct
- Three prompt variations exist (raw, no_reason, reason)
- Notes explain why binding is challenging

## Output: Return exactly 5 test cases with descriptive names.
```

Listing 10: Knowledge binding Q/A dataset generation prompt

Quality evaluation prompt. List. 11 shows the prompt used to evaluate the quality of the generated test cases.

```
You are evaluating test cases designed to measure if AI models can recall
    multiple related facts together.

## WHAT MAKES A GOOD TEST CASE:
```

```
A good test case has:
1. Multiple SEPARATE real-world facts (2+) that share something in common (same
    time period, same category, same event type)
2. Individual verification questions for each fact (yes/no, "which X did Y",
    "when did Z happen")
3. A BINDING question that asks the model to list ALL items matching a criteria
    (e.g., "List ALL companies that did X in 2022")
4. Multiple choice options where wrong answers are plausible but incorrect

## CRITICAL FLAWS (score = 0 if found):

**FLAW 1 - SELF-REFERENCE**: The binding question says "in this test case",
    "mentioned above", "described earlier". This is INVALID because it asks
    about the test itself, not real-world knowledge.
- BAD: "List ALL missions in this test case"
- GOOD: "List ALL NASA missions launched in March 2020"

**FLAW 2 - SINGLE EVENT**: All facts describe ONE event from different angles,
    not multiple separate events.
- BAD: Facts about one wedding (date + location + venue) - this is 1 event with 3
    attributes
- GOOD: Facts about three different weddings in the same month - these are 3
    separate events

**FLAW 3 - INCOMPLETE COVERAGE**: The expected answer includes items that have no
    verification questions.
- BAD: Expected ["A", "B", "C"] but only facts for A and B exist
- GOOD: Expected ["A", "B", "C"] with verification facts for all three

**FLAW 4 - WRONG METADATA**: The "num_items" field doesn't match actual number of
    distinct items, or title mentions wrong time period.

## SCORING CRITERIA (0.0 to 1.0 each):

1. **no_self_reference**: 1.0 if binding question asks about real-world facts,
    0.0 if it references "this test case"
2. **multiple_distinct_events**: 1.0 if 2+ separate real-world events, 0.5 if
    borderline, 0.0 if single event
3. **complete_fact_coverage**: 1.0 if all expected items have verification
    questions, 0.5 if some missing
4. **good_distractors**: 1.0 if wrong MC options are plausible confusions, 0.5 if
    too obvious, 0.0 if nonsensical
5. **correct_metadata**: 1.0 if num_items and dates match facts, 0.5 if minor
    issues, 0.0 if wrong

## RESPOND WITH ONLY THIS JSON (no other text):

{
  "critical_flaw": null or "SELF_REFERENCE" or "SINGLE_EVENT" or
    "INCOMPLETE_COVERAGE" or "WRONG_METADATA",
  "flaw_evidence": null or "quote the problematic text",
  "scores": {
    "no_self_reference": 0.0-1.0,
    "multiple_distinct_events": 0.0-1.0,
    "complete_fact_coverage": 0.0-1.0,
    "good_distractors": 0.0-1.0,
    "correct_metadata": 0.0-1.0
  },
  "overall_score": 0.0-1.0,
  "issues": ["list specific problems"],
  "quality": "EXCELLENT" or "GOOD" or "NEEDS_REVISION" or "INVALID"
}

## TEST CASE TO EVALUATE:
{example_json}
```

Listing 11: Binding dataset example quality evaluation prompt

Duplicate detection. List. 12 shows the prompt used to identify duplicate test cases that test essentially the same knowledge.

```
Evaluate if each pair of test cases are DUPLICATES (testing essentially the same
    thing).

DUPLICATE if:
- Both test the same event/fact with same or very similar prompts
- One is a subset or variant of the other
- They would have the same expected answer for the same reason

NOT DUPLICATE if:
- Different events even if similar names (e.g., "2016 Grammy" vs "2017 Grammy")
- Same topic but testing different aspects
- Different time periods or contexts

{formatted_pairs}

Return JSON: [{"id": 0, "verdict": "duplicate|not_duplicate", "reason": "brief
    explanation"}]
```

Listing 12: Duplicate detection prompt

Prompt conditioning evaluation. List. 13 shows the prompt used to identify poorly conditioned prompts that are ambiguous or unfair.

```
Evaluate if each prompt is BAD-CONDITIONED (unfair or impossible to answer
    correctly).

A prompt is BAD if:
1. **Ambiguous scope**: Doesn't clearly define what to include/exclude
2. **Missing context**: References "this case" or assumes prior context not in
    prompt
3. **Expected mismatch**: Expected answer doesn't match what prompt asks

A prompt is GOOD if:
- Clear, specific, and self-contained
- Expected answer matches what's being asked

{formatted_prompts}

Return JSON: [{"id": 0, "verdict": "good|bad",
"issue": "ambiguous_scope|missing_context|overly_specific|expected_mismatch|
           temporal_ambiguity|none",
"explanation": "brief why"}]
```

Listing 13: Prompt conditioning evaluation prompt

Response completeness evaluation. List. 14 shows the prompt used to detect incomplete model responses (e.g., truncated due to token limits).

```
Evaluate if each reasoning response is INCOMPLETE.

A response is INCOMPLETE if:
1. **Truncated**: Cut off mid-sentence or mid-thought (often due to max_tokens)
2. **No conclusion**: Has reasoning but doesn't give final answer
3. **Abandoned reasoning**: Starts reasoning process but doesn't finish
4. **Empty/minimal**: Too short to contain meaningful reasoning

A response is COMPLETE if:
- Contains reasoning AND final answer
- Reaches a conclusion (even if wrong)
- Self-contained and finished thought

{formatted_responses}

Return JSON: [{"id": 0, "verdict": "incomplete|complete", "issue":
    "truncated|no_conclusion|abandoned|empty|none", "explanation": "brief"}]
```

Listing 14: Response completeness evaluation prompt

B.4 Complete binding failure results

We extend the evaluation proposed in Sec. 3 by evaluating eleven prompt strategies: eight for free-form list responses (Instant, Instant Hinted, Reasoning, Reasoning with First Letters, Reasoning Hinted, No Reasoning, SelfAsk, SelfAsk-Binding) and three for multiple-choice format (MCQ, MCQ Reasoning, MCQ No Reasoning); we analyze the prompts in detail in App. B.5. For Claude Opus 4.6, Claude Opus 4.5, Claude Sonnet 4, GPT-5.4, Grok 4.20, and Gemini 3.1 Pro, extended thinking is enabled natively via API; for the remaining models (GPT-4o, Gemini 2.0 Flash, DeepSeek-V3, Mistral Large 25.12), reasoning is elicited through prompt-based instructions. Cell values show the failure rate as a percentage along with the sample size (N = number of binding questions with verified facts for that model).

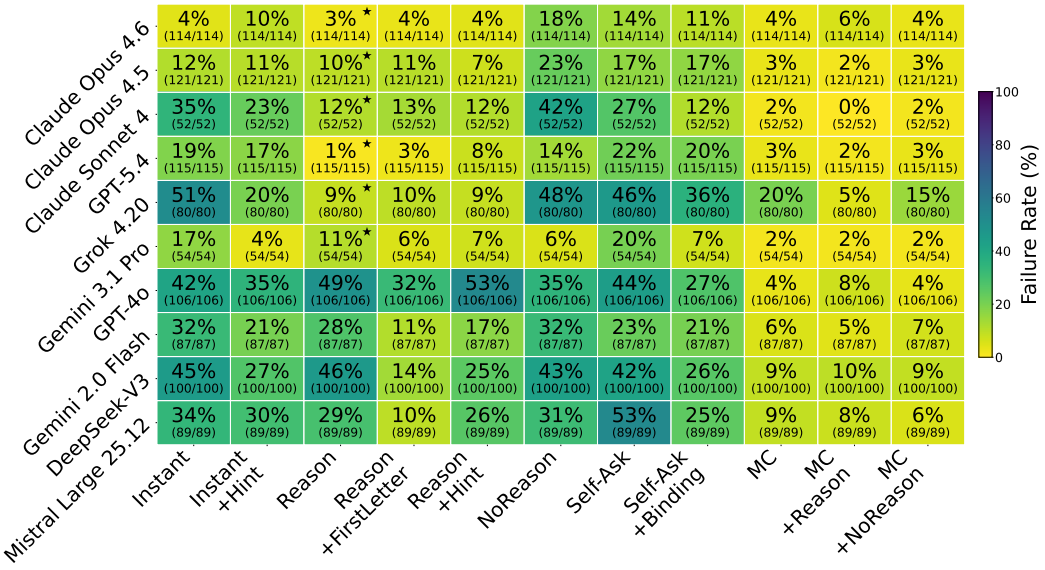


Figure 8: Binding failure rates across frontier models, evaluated under different prompting strategies. Cells show failure rate and sample size (N = binding questions with verified facts). Models marked with ★ have extended thinking enabled.

Fig. 8 shows binding failures occurring consistently across all frontier models. Under instant prompting, Grok 4.20 fails on 51% of verified bindings, DeepSeek-V3 on 45% and GPT-4o on 42%, while the strongest models fail on only 4–12% (Claude Opus 4.6: 4%, Claude Opus 4.5: 12%). MCQ formats generally yield the lowest failure rates compared to free-form list responses: GPT-4o drops

from 27–53% (list) to 4–8% (MCQ) and DeepSeek-V3 from 14–46% to 9–10%, suggesting models struggle more with exhaustive recall than recognition. Reasoning instructions produce heterogeneous effects: models with native extended thinking improve (Sonnet 4: 35% $\rightarrow$ 12% on Reasoning; GPT-5.4: 19% $\rightarrow$ 1%; Grok 4.20: 51% $\rightarrow$ 9%; Opus 4.5: 12% $\rightarrow$ 7% on Reasoning Hinted), while GPT-4o degrades from 42% to 49% with prompt-based reasoning and further to 53% with Reasoning Hinted (see App. B.6 for qualitative analysis). Models with extended thinking (★) generally achieve the lowest failure rates, with Claude Opus 4.5 reaching 2% and Sonnet 4 reaching 0% on MCQ Reasoning. Hinting strategies (Instant Hinted, Reasoning with First Letters) provide partial mitigation for some models: Gemini 2.0 Flash drops from 28% to 11% with First Letter cues, and DeepSeek-V3 from 46% to 14%, though GPT-4o sees no consistent benefit. Notably, several models still exceed 25% on multiple free-form conditions (Grok 4.20, GPT-4o, DeepSeek-V3, Mistral Large 25.12), providing further evidence that binding failures persist across the frontier.

B.4.1 Cardinality, salience, and threshold robustness

The failure rates above could have simpler explanations than a failure to bind. We test three of them. (i) *Cardinality*: long answer lists are harder to write in full. (ii) *Salience*: failures could concern facts the model only barely knows. (iii) *Marginal facts*: the "known" threshold (3 of 5 correct trials, Sec. 3.1) could be too lenient. If one of these drove the results, failures should grow with list length, drop under a stricter threshold, or concentrate on facts that only just pass it. None of the three happens.

Cardinality. Of the 918 verified (model, question) pairs, 502 target exactly 2 episodes, 362 target 3, and 54 target 4. Pooled over models, failure rates do not increase with the number of target episodes, and most failures occur at exactly 2 targets, where producing a two-item list is not a plausible generic list-generation bottleneck (per-model breakdown in App. B.4.2).

Salience and threshold. If failures came from weakly known facts, a stricter definition of "known" should remove many of them. We recompute every Instant, Reasoning and SelfAsk failure rate with stricter thresholds of 4/5 and 5/5 correct verification trials (Tab. 7). Moving from 3/5 to 5/5 changes the rates by only a few points, in both directions (Instant: 4 of 10 models go up, namely Claude Opus 4.6, Claude Sonnet 4, Gemini 3.1 Pro and Gemini 2.0 Flash, and 6 go down), and the ranking of models stays the same (Spearman ρ between 3/5 and 5/5: 0.98 for Instant, 0.98 for Reasoning, 0.92 for SelfAsk; between 3/5 and 4/5: 0.99, 0.95, 0.96). This is not what the salience explanation predicts.

Table 7: Failure rate (%) as the known-facts threshold tightens from 3/5 to 4/5 to 5/5 correct trials on every verification type, for three prompting strategies. $N_{@3}/N_{@4}/N_{@5}$: known-pool sizes.

		Instant			**Reason**			**SelfAsk**		
Model	$N_{@3}/N_{@4}/N_{@5}$	@3	@4	@5	@3	@4	@5	@3	@4	@5
Claude Opus 4.6	114/108/104	4.4	4.6	4.8	2.6	2.8	2.9	14.0	13.9	12.5
Claude Opus 4.5	121/118/114	11.6	11.9	11.4	9.9	9.3	8.8	16.5	16.1	15.8
Claude Sonnet 4	52/51/48	34.6	35.3	35.4	11.5	11.8	10.4	26.9	27.5	27.1
GPT-5.4	115/102/73	19.1	19.6	16.4	0.9	1.0	1.4	21.7	22.5	24.7
Grok 4.20	80/73/60	51.2	53.4	46.7	8.8	9.6	5.0	46.2	47.9	45.0
Gemini 3.1 Pro	54/35/23	16.7	17.1	21.7	11.1	8.6	13.0	20.4	25.7	30.4
GPT-4o	106/99/86	41.5	41.4	40.7	49.1	47.5	46.5	44.3	43.4	44.2
Gemini 2.0 Flash	87/86/83	32.2	32.6	33.7	27.6	27.9	28.9	23.0	23.3	22.9
DeepSeek-V3	100/81/58	45.0	44.4	41.4	46.0	45.7	44.8	42.0	40.7	34.5
Mistral Large 25.12	89/85/51	33.7	31.8	29.4	29.2	28.2	23.5	52.8	52.9	49.0

Marginal facts. For the model with the fewest failures, Claude Opus 4.6, we inspect all five Instant failures by hand (App. B.4.3). In every case, each constituent fact passed all verification questions in 5 of 5 trials, well above the 3/5 threshold, and in 4 of the 5 cases the model failed all five binding trials. These failures are therefore stable and concern well-known facts, not facts near the known/unknown boundary.

B.4.2 Failure rate by answer-set size

A model could fail a binding question simply because writing a long list is hard, whatever its ability to bind. If this were the main cause, the failure rate should grow with the number of target episodes

$|Y(q)|$. Tab. 8 breaks down Instant failures by $|Y(q)|$, recomputed from the raw evaluation logs with the same known-fact population and $\geq$3/5 failure rule as Fig. 2. Of the 255 binding questions, 126, 120 and 9 target 2, 3 and 4 episodes. Each model is scored only on the questions whose facts it knows, so the same question can be counted once for each model that knows it, giving 918 (model, question) pairs (502, 362 and 54 for 2, 3 and 4 targets). Pooled over models, the failure rate is 29.7%, 28.2% and 9.3% for 2, 3 and 4 targets, so failures do not grow with set size. Per model the trend is mixed (six models fail slightly more often at 3 than at 2 targets), and 58% of all failures (149 of the 256 failing pairs) occur at exactly 2 targets, where producing a two-item list is not a plausible bottleneck. The 4-target bucket is small (54 pairs), so we do not read much into its lower rate.

Table 8: Binding failure rate (%, Instant) by number of target episodes $|Y(q)|$, per model. In parentheses: number of binding questions in the bucket whose facts are all verified as known by that model; the *All* column equals N in Fig. 2, and the last row sums over models.

Model	**2 targets**	**3 targets**	**4 targets**	**All**
Claude Opus 4.6	6.8 (59)	2.0 (49)	0.0 (6)	4.4 (114)
Claude Opus 4.5	16.9 (65)	6.2 (48)	0.0 (8)	11.6 (121)
Claude Sonnet 4	43.8 (32)	23.5 (17)	0.0 (3)	34.6 (52)
GPT-5.4	22.8 (57)	18.4 (49)	0.0 (9)	19.1 (115)
Grok 4.20	50.0 (44)	51.7 (29)	57.1 (7)	51.2 (80)
Gemini 3.1 Pro	18.2 (22)	19.2 (26)	0.0 (6)	16.7 (54)
GPT-4o	38.3 (60)	48.8 (41)	20.0 (5)	41.5 (106)
Gemini 2.0 Flash	32.0 (50)	38.7 (31)	0.0 (6)	32.2 (87)
DeepSeek-V3	40.4 (57)	55.0 (40)	0.0 (3)	45.0 (100)
Mistral Large 25.12	33.9 (56)	34.4 (32)	0.0 (1)	33.7 (89)
All models	29.7 (502)	28.2 (362)	9.3 (54)	27.9 (918)

B.4.3 Per-failure audit

Aggregate rates could hide failures on facts that the model barely knows. We therefore read, one by one, all five Instant failures of Claude Opus 4.6, the model with the fewest failures (Tab. 9). Every constituent episode passed all three verification types in all five trials, and four of the five questions failed every binding trial, so these are stable failures on well-known facts. Reading the answers, one case (ExoMars) is a judge false negative, since "ExoMars Trace Gas Orbiter" is the ExoMars 2016 mission, and one (Ben Simmons) reflects a different reading of the question: the model mentions LSU in its reasoning but restricts its answer to professional teams. The remaining three are genuine omissions of a verified episode.

Table 9: Per-failure audit: all Instant binding failures of Claude Opus 4.6. Verification: correct/total trials for the yes/no, forward-recall and reverse-recall questions of each constituent episode. Binding: correct/total trials.

Binding question	**Expected**	**Model answer (trials)**	**Verification**	**Binding**
List ALL teams that Ben Simmons played for between 2015 and 2020.	LSU Tigers men's basketball; Philadelphia 76ers	[Philadelphia 76ers] $\times$5	all 5/5	0/5
List ALL teams that Nwankwo Kanu played for between 2010 and 2012, including both club and national teams.	Nigeria national football team; Portsmouth F.C.	[Portsmouth] $\times$4; [Portsmouth FC] $\times$1	all 5/5	0/5
List ALL planetary missions that had major milestones in 2016, specifically: one NASA mission that arrived at Jupiter and one ESA/Roscosmos mission launched toward Mars. List only mission names.	Juno; ExoMars 2016	[Juno, ExoMars Trace Gas Orbiter] $\times$5	all 5/5	0/5
List ALL Netflix original series that premiered between January 1-7, 2024.	Fool Me Once; The Brothers Sun	[(no list)] $\times$4; [Fool Me Once, The Brothers Sun] $\times$1	all 5/5	1/5
List ALL companies that made major cloud-related acquisitions exceeding $2 billion between June and July 2016.	Microsoft; Oracle; Salesforce	[Microsoft, Oracle] $\times$5	all 5/5	0/5

B.4.4 Refusal vs. incomplete failures

A binding failure can take two forms. The model can *refuse*: it declines to answer or says it lacks reliable information. Or it can give an *incomplete* answer: it confidently lists only some of the expected items, or includes a wrong entity. For example, asked for all clubs of a player who played for A and B, "I do not have reliable information about this player's career" is a refusal, while "A" alone is incomplete. The two forms mean different things: many refusals would point to a cautious output policy, while incomplete answers point to a retrieval failure. We therefore classify every counted failure by hand into one of the two categories.

Every fact in this analysis is one that the model already answers correctly when asked about it alone (Sec. 3.1), so any refusal or omission appears only when the facts must be produced together. Tab. 10 reports, for each model and prompting strategy, the failure rate split into refusals (R) and incomplete answers (I).

Table 10: Failure rate (%) with a refusal (R) vs. incomplete-answer (I) breakdown in percentage points of N_{Know} (R + I = failure rate), six prompting strategies. N_{Know}: binding questions with verified facts for that model.

Model	N_{Know}	Instant	Reason	SelfAsk	SelfAsk-Binding	MC	MC+Reason
Claude Opus 4.6	114	4% (1R/3I)	3% (0R/3I)	14% (1R/13I)	11% (0R/11I)	4% (0R/4I)	6% (0R/6I)
Claude Opus 4.5	121	12% (0R/12I)	10% (0R/10I)	17% (2R/15I)	17% (1R/16I)	3% (0R/3I)	2% (0R/2I)
Claude Sonnet 4	52	35% (15R/20I)	12% (0R/12I)	27% (15R/12I)	12% (2R/10I)	2% (2R/0I)	0% (0R/0I)
GPT-5.4	115	19% (4R/15I)	1% (0R/1I)	22% (2R/20I)	20% (2R/18I)	3% (0R/3I)	2% (0R/2I)
Grok 4.20	80	51% (0R/51I)	9% (6R/3I)	46% (0R/46I)	36% (5R/31I)	20% (0R/20I)	5% (4R/1I)
Gemini 3.1 Pro	54	17% (9R/8I)	11% (7R/4I)	20% (13R/7I)	7% (4R/3I)	2% (2R/0I)	2% (2R/0I)
GPT-4o	106	42% (25R/17I)	49% (35R/14I)	44% (30R/14I)	27% (5R/22I)	4% (0R/4I)	8% (0R/8I)
Gemini 2.0 Flash	87	32% (2R/30I)	28% (2R/26I)	23% (5R/18I)	21% (3R/18I)	6% (0R/6I)	5% (1R/4I)
DeepSeek-V3	100	45% (9R/36I)	46% (11R/35I)	42% (19R/23I)	26% (2R/24I)	9% (0R/9I)	10% (0R/10I)
Mistral Large 25.12	89	34% (6R/28I)	29% (1R/28I)	53% (25R/28I)	25% (1R/24I)	9% (0R/9I)	8% (0R/8I)

For 7 of 10 models, refusals are on average a small minority of failures. They dominate for GPT-4o (25R/17I under Instant and 35R/14I under Reason, in percentage points) and for Claude Sonnet 4 under SelfAsk (15R/12I; under Instant it is 15R/20I), and Gemini 3.1 Pro refuses at least as often as it under-lists in every condition where it fails at all. Abstention therefore matters for some models under some prompts, but it does not explain the phenomenon as a whole: for most models and most prompting strategies, failures are incomplete answers rather than refusals, and they persist under the free-form *Reason* prompt, where the model can think before answering (up to 49%; 10% or more for 7 of 10 models).

B.4.5 Precision, in vivo

Our main metric is recall: an answer is a success only if it contains every expected item. Over-listing can never turn a correct answer into a failure, but it could in principle produce successes: a model asked for the two clubs of a player could list five clubs and still reach recall 1. To check that successes do not come from such over-listing, we also measure *precision*, the fraction of listed items that match an expected item, under the Instant and Reasoning prompts (Tab. 11). If successes came from long lists, precision would be low and models would list many more items than expected. Matched items come from the same LLM judge; listed items are parsed from the final list in the raw, unedited response (a list is parsable in 54–100% of trials per model, and precision is computed on those).

Precision is 87–96% for every model and condition, and models list on average about as many items as expected (2.2–3.2 listed vs. 2.3–2.6 expected). Successful answers are therefore precise, not long lists that happen to contain the right items. Over-listing could only inflate successes, never failures: an answer that misses one expected item has recall below 1 whatever else it contains, so the reported failure rates remain lower bounds. A strict exact-set criterion, under which any extra item makes an answer wrong, raises failure rates (Tab. 11), but it is a pessimistic bound because the gold sets are not exhaustive: in a manual check of 30 answers that pass recall but fail the strict criterion, 22 of the extra items were true facts missing from the gold set, 1 was an alias, 3 were parsing artifacts, and only 4 were genuinely wrong.

B.5 Prompt sensitivity and reasoning as workaround

To extend the discussion on prompt sensitivity from Sec. 3.2, we observed that even minor prompt variations (changing one or two words) can lead to substantially different model responses. To inves-

Table 11: In-vivo failure rate under the recall criterion (paper metric) and under a strict exact-set criterion (an answer with any extra item counts as wrong), precision, and mean number of listed items (expected: $|Y|$), under Instant and Reasoning prompts. Strict and listed items are computed on answers with a parsable list; see text for caveats.

	Instant				Reason			
Model	Fail	Strict fail	Prec.	Items ($\|Y\|$)	Fail	Strict fail	Prec.	Items ($\|Y\|$)
Claude Opus 4.6	4.4	21.1	91	2.91 (2.44)	2.6	12.3	91	2.89 (2.50)
Claude Opus 4.5	11.6	23.1	95	2.48 (2.43)	9.9	23.1	94	2.50 (2.43)
Claude Sonnet 4	34.6	40.4	96	2.24 (2.35)	11.5	26.9	94	2.36 (2.31)
GPT-5.4	19.1	33.0	88	3.01 (2.49)	0.9	21.7	87	3.21 (2.51)
Grok 4.20	51.2	56.2	90	2.29 (2.44)	8.8	16.2	96	2.60 (2.43)
Gemini 3.1 Pro	16.7	25.9	96	2.76 (2.54)	11.1	18.5	95	2.97 (2.57)
GPT-4o	41.5	46.2	95	2.32 (2.36)	49.1	54.7	95	2.35 (2.36)
Gemini 2.0 Flash	32.2	36.8	92	2.36 (2.40)	27.6	31.0	93	2.40 (2.40)
DeepSeek-V3	45.0	50.0	92	2.36 (2.42)	46.0	52.0	90	2.28 (2.42)
Mistral Large 25.12	33.7	44.9	91	2.33 (2.36)	29.2	39.3	94	2.27 (2.35)

tigate this sensitivity, we tested eleven prompt strategies across three dimensions: (1) output format (free-form list vs. multiple choice), (2) reasoning instructions (none, suppressed, or encouraged), and (3) answer hints (multiplicity hint, first-letter hint, or none). The eight free-form strategies are: Instant (List. 15), Instant Hinted (List. 18), Reasoning (List. 17), Reasoning with First Letters (List. 19), Reasoning Hinted (List. 20), No Reasoning (List. 16), SelfAsk (List. 24), and SelfAsk-Binding (List. 25). The three multiple-choice strategies are: MCQ (List. 21), MCQ Reasoning (List. 23), and MCQ No Reasoning (List. 22). This allows us to evaluate whether explicit reasoning instructions, constrained answer formats, or answer hints help mitigate binding failures.

For models with native extended thinking (Claude Opus 4.6, Claude Opus 4.5, Claude Sonnet 4, GPT-5.4, Grok 4.20, Gemini 3.1 Pro), we enabled it through the provider API, which allows the model to engage in explicit intermediate reasoning before producing its final response. This capability is distinct from simple chain-of-thought prompting, as it leverages a dedicated reasoning budget within the model's inference process. For the remaining models (GPT-4o, Gemini 2.0 Flash, DeepSeek-V3, Mistral Large 25.12), reasoning was elicited solely through prompt-based instructions (Lists. 17 and 25).

```
"List ALL teams that {entity name} played for between {start year} and {end
    year}. Provide your answer as a JSON list like [\"a\",\"b\"]."
```

Listing 15: Binding list prompt (raw)

```
"List ALL teams that {entity name} played for between {start year} and {end
    year}. Provide your answer as a JSON list like [\"a\",\"b\"]. NO extra text,
    NO reasoning!"
```

Listing 16: Binding list prompt (no reasoning)

```
"List ALL teams that {entity name} played for between {start year} and {end
    year}. Provide your answer as a JSON list like [\"a\",\"b\"]. Think about
    each event independently and reason before answering."
```

Listing 17: Binding list prompt (with reasoning)

```
"List exactly the N teams that {entity name} played for between {start year} and
    {end year}. Provide your answer as a JSON list like [\"a\",\"b\"]."
```

Listing 18: Binding list prompt hinted on cue multiplicity.

```
"List ALL teams that {entity name} played for between {start year} and {end
    year}. The first letters of the answers are: A, B, C. Provide your answer as
    a JSON list like [\"a\",\"b\"]."
```

Listing 19: Binding list prompt hinted on answers.

```
"List exactly the N teams that {entity name} played for between {start year} and
    {end year}. Provide your answer as a JSON list like [\"a\",\"b\"]. Think
    about each event independently and reason before answering."
```

Listing 20: Binding list prompt (Reasoning Hinted): combines reasoning with a multiplicity hint.

```
"Which teams did {entity name} play for between {start year} and {end year}?
    Select the correct option:
A. {option A}
B. {option B}
C. {option C}
D. None of them
Return ONLY the letter."
```

Listing 21: Binding multiple-choice prompt (raw)

```
"Which teams did {entity name} play for between {start year} and {end year}?
    Select the correct option:
A. {option A}
B. {option B}
C. {option C}
D. None of them
Return ONLY the letter. Do NOT write anything else, no explanation, no reasoning!"
```

Listing 22: Binding multiple-choice prompt (no reasoning)

```
"Which teams did {entity name} play for between {start year} and {end year}?
    Select the correct option:
A. {option A}
B. {option B}
C. {option C}
D. None of them
Return ONLY the letter. Reflect about each event independently, take your time
    before answering, reason about it!"
```

Listing 23: Binding multiple-choice prompt (with reasoning)

```
"Question: When does monsoon season end in the state the area code 575 is located?
Are follow up questions needed here: Yes.
Follow up: Which state is the area code 575 located in?
Intermediate answer: The area code 575 is located in New Mexico.
Follow up: When does monsoon season end in New Mexico?
Intermediate answer: Monsoon season in New Mexico typically ends in mid-September.
So the final answer is: mid-September.

Question: What is the current official currency in the country where Ineabelle
    Diaz is a citizen?
Are follow up questions needed here: Yes.
Follow up: Which country is Ineabelle Diaz a citizen of?
Intermediate answer: Ineabelle Diaz is from Peurto Rico, which is in the United
    States of America.
Follow up: What is the current official currency in the United States of America?
Intermediate answer: The current official currency in the United States is the
    United States dollar.
So the final answer is: United States dollar.
```

```
Question: Where was the person who founded the American Institute of Public
    Opinion in 1935 born?
Are follow up questions needed here: Yes.
Follow up: Who founded the American Institute of Public Opinion in 1935?
Intermediate answer: George Gallup.
Follow up: Where was George Gallup born?
Intermediate answer: George Gallup was born in Jefferson, Iowa.
So the final answer is: Jefferson.

Question: What language is used by the director of Tiffany Memorandum?
Are follow up questions needed here: Yes.
Follow up: Who directed the movie called Tiffany Memorandum?
Intermediate answer: Sergio Grieco.
Follow up: What language is used by Sergio Grieco?
Intermediate answer: Sergio Grieco speaks Italian.
So the final answer is: Italian.

Question: What is the sports team the person played for who scored the first
    touchdown in Superbowl 1?
Are follow up questions needed here: Yes.
Follow up: Which player scored the first touchdown in Superbowl 1?
Intermediate answer: Max McGee.
Follow up: Which sports team did Max McGee play for?
Intermediate answer: Max McGee played for the Green Bay Packers.
So the final answer is: Green Bay Packers.

Question: The birth country of Jayantha Ketagoda left the British Empire when?
Are follow up questions needed here: Yes.
Follow up: What is the birth country of Jayantha Ketagoda?
Intermediate answer: Sri Lanka.
Follow up: When did Sri Lanka leave the British Empire?
Intermediate answer: Sri Lanka left the British Empire on February 4, 1948.
So the final answer is: February 4, 1948.

Question: {question}
Are follow up questions needed here:"
```

Listing 24: SelfAsk prompt from Press et al., 2023, Measuring and Narrowing the Compositionality Gap in Language Models

```
"Question: List ALL teams that LeBron James played for between 2003 and 2023.
    Provide your answer as a JSON list like ["a","b"].
Are follow up questions needed here: Yes.
Follow up: Which team did LeBron James play for at the start of this period,
    around 2003?
Intermediate answer: LeBron James was drafted by the Cleveland Cavaliers in 2003
    and played there until 2010.
Follow up: Did LeBron James move to another team after the Cleveland Cavaliers?
Intermediate answer: Yes, he joined the Miami Heat in 2010 and played there until
    2014.
Follow up: Did he play for any other teams after the Miami Heat and before 2023?
Intermediate answer: He returned to the Cleveland Cavaliers in 2014, then joined
    the Los Angeles Lakers in 2018, where he remained through 2023.
Follow up: Are there any other teams LeBron James played for in this period that
    I might be missing?
Intermediate answer: No, those are all the teams: Cleveland Cavaliers, Miami
    Heat, and Los Angeles Lakers.
So the final answer is: ["Cleveland Cavaliers", "Miami Heat", "Los Angeles
    Lakers"]

Question: List ALL teams that Zlatan Ibrahimović played for between 2004 and
    2020. Provide your answer as a JSON list like ["a","b"].
Are follow up questions needed here: Yes.
Follow up: Which team was Zlatan Ibrahimović playing for around 2004?
Intermediate answer: He left Ajax in 2004 and was at Juventus from 2004 to 2006.
```

```
Follow up: Where did he go after Juventus?
Intermediate answer: He moved to Inter Milan in 2006 and stayed until 2009.
Follow up: Which team did he join after Inter Milan?
Intermediate answer: He joined Barcelona in 2009, then moved to AC Milan on loan
    in 2010.
Follow up: Did he play for other teams between 2010 and 2020?
Intermediate answer: Yes, he joined Paris Saint-Germain in 2012, then Manchester
    United in 2016, then LA Galaxy in 2018, and returned to AC Milan in 2020.
So the final answer is: ["Juventus", "Inter Milan", "Barcelona", "AC Milan",
    "Paris Saint-Germain", "Manchester United", "LA Galaxy"]

Question: List ALL Grand Slam tennis tournaments that Rafael Nadal won in 2010.
    Provide your answer as a JSON list like ["a","b"].
Are follow up questions needed here: Yes.
Follow up: Did Rafael Nadal win the Australian Open in 2010?
Intermediate answer: No, Roger Federer won the 2010 Australian Open.
Follow up: Did Rafael Nadal win the French Open in 2010?
Intermediate answer: Yes, Nadal won the 2010 French Open.
Follow up: Did Rafael Nadal win Wimbledon in 2010?
Intermediate answer: Yes, Nadal won Wimbledon in 2010.
Follow up: Did Rafael Nadal win the US Open in 2010?
Intermediate answer: Yes, Nadal won the 2010 US Open, completing a career Grand
    Slam.
Follow up: Are there any Grand Slams I missed?
Intermediate answer: No, there are only four Grand Slams per year. Nadal won
    three of them in 2010: French Open, Wimbledon, and US Open.
So the final answer is: ["French Open", "Wimbledon", "US Open"]

Question: List ALL countries visited by George W. Bush between June 14 - June 16
    2001. Provide your answer as a JSON list like ["a","b"].
Are follow up questions needed here: Yes.
Follow up: Which country did George W. Bush visit on June 14-15?
Intermediate answer: Sweden.
Follow up: Which country did he visit on June 15-16?
Intermediate answer: Poland.
Follow up: Which country did he visit on June 16?
Intermediate answer: Slovenia.
Follow up: Did he visit Belgium within June 14-16?
Intermediate answer: Belgium was visited June 13-14, so it partially overlaps
    June 14.
Follow up: Should Belgium be included given the date range includes June 14?
Intermediate answer: Yes, since June 14 is within the specified range.
So the final answer is: ["Belgium", "Sweden", "Poland", "Slovenia"]

Now answer the next question in exactly this format:

Are follow up questions needed here: Yes or No.
If Yes, write one or more lines in the form:
Follow up: ...
Intermediate answer: ...
Finally end with:
So the final answer is: ["item1", "item2"]

Question: {question}
Provide the complete answer now."
```

Listing 25: Our adaptation of the SelfAsk prompt to the binding task.

B.5.1 Statistical significance of per-model differences

Sec. 3.2 compares prompting strategies within each model, and some of these differences are only a few points. Since each model has a small known pool (52–121 questions), a few-point difference can be noise. We therefore test each per-model difference with a paired bootstrap over questions (10,000 resamples) and a two-sided Wilcoxon signed-rank test (Tab. 12; the table uses trial-level failure

rates, hence small differences from Fig. 2). Large effects are significant, e.g., reasoning for Claude Sonnet 4, GPT-5.4 and Grok 4.20, and MCQ for Gemini 3.1 Pro and Claude Sonnet 4. Differences of a few points (Claude Opus 4.6 and 4.5, Gemini 2.0 Flash, DeepSeek-V3, and the increase for GPT-4o under reasoning, $p = 0.10$) are not significant and should be read as descriptive.

Table 12: Paired comparison of prompting strategies per model: trial-level failure rate (%), 95% bootstrap CI of the difference (10,000 resamples over instances) and two-sided Wilcoxon signed-rank p.

Model	**Comparison**	N	**Base**	**Variant**	**Δ [95% CI]**	p
Claude Opus 4.6	Instant → Reason	114	4.6	3.5	-1.1 [-4.7, +2.5]	0.560
Claude Opus 4.5	Instant → Reason	121	12.4	9.8	-2.6 [-6.0, +0.5]	0.134
Claude Sonnet 4	Instant → Reason	52	35.4	12.3	-23.1 [-34.2, -12.3]	<0.001
GPT-5.4	Instant → Reason	115	20.2	3.0	-17.2 [-23.7, -11.1]	<0.001
Grok 4.20	Instant → Reason	80	51.7	9.8	-42.0 [-52.3, -31.8]	<0.001
Gemini 3.1 Pro	Instant → Reason	54	18.1	10.0	-8.1 [-14.5, -1.9]	0.013
GPT-4o	Instant → Reason	106	42.1	47.7	+5.7 [-0.4, +11.9]	0.103
Gemini 2.0 Flash	Instant → Reason	87	32.2	29.0	-3.2 [-8.7, +2.1]	0.244
DeepSeek-V3	Instant → Reason	100	42.2	44.2	+2.0 [-1.6, +5.6]	0.480
Mistral Large 25.12	Instant → Reason	89	34.6	26.7	-7.9 [-14.2, -2.0]	0.022
Gemini 3.1 Pro	Instant → SelfAsk	54	18.1	20.0	+1.9 [-4.8, +8.9]	0.569
Mistral Large 25.12	Instant → SelfAsk	89	34.6	53.3	+18.7 [+10.1, +27.4]	<0.001
Gemini 3.1 Pro	Instant → MC	54	18.1	1.9	-16.3 [-26.3, -6.3]	0.004
Claude Sonnet 4	Instant → MC+Reason	52	35.4	0.0	-35.4 [-47.7, -23.5]	<0.001

B.6 Qualitative analysis of reasoning effects

To better understand why reasoning instructions produce divergent effects across models, we examine representative cases from Claude Sonnet 4 (which improved substantially with reasoning) and GPT-4o (which degraded). For Claude models, extended thinking was enabled through the Anthropic API. For GPT-4o, Gemini 2.0 Flash, DeepSeek-V3 and Mistral Large 25.12, reasoning was elicited solely through prompt-based instructions (see Sec. B.5).

Case study 1: reasoning enables retrieval (Claude Sonnet 4) When asked to list ISRO deep space missions launched in 2023, Claude Sonnet 4 without reasoning instructions refused to answer, expressing uncertainty (List. 26). However, with extended thinking enabled, the model successfully retrieved both missions by systematically reasoning through each one (List. 27).

This pattern recurred across multiple examples: the model's default behavior was to express uncertainty and abstain, but explicit reasoning steps allowed it to systematically retrieve and verify each entity independently. The extended thinking mechanism appears to overcome an overly conservative abstention heuristic by forcing the model to attempt retrieval before declaring uncertainty.

```
PROMPT:
List ALL major deep space missions launched by the Indian Space
Research Organisation (ISRO) during the calendar year 2023.
Provide your answer as a JSON list like ["a","b","c"].

RESPONSE:
I don't have reliable information about all major deep space
missions launched by ISRO specifically during the calendar year
2023. While I know that ISRO has conducted various space missions,
I cannot confidently provide a complete list of deep space missions
launched in that specific year without risking inaccuracy.

EXPECTED: ["Chandrayaan-3", "Aditya-L1"]
```

Listing 26: Claude Sonnet 4 without reasoning: abstains despite possessing knowledge.

Case study 2: reasoning triggers abstention (GPT-4o) GPT-4o exhibited the opposite pattern. When asked to list SpaceX missions that achieved reusability milestones in 2017, the model answered

```
PROMPT:
List ALL major deep space missions launched by the Indian Space
Research Organisation (ISRO) during the calendar year 2023.
Provide your answer as a JSON list like ["a","b","c"]. Think about
each mission independently and reason before answering.

RESPONSE:
Let me think through ISRO's major deep space missions launched
in 2023:

1. Chandrayaan-3: Launched on July 14, 2023 - This was ISRO's
   third lunar exploration mission, which successfully achieved
   a soft landing on the Moon's south pole in August 2023.

2. Aditya-L1: Launched on September 2, 2023 - This was India's
   first dedicated solar observation mission, designed to study
   the Sun from the L1 Lagrange point.

These were the two major deep space missions launched by ISRO
during the calendar year 2023.

["Chandrayaan-3", "Aditya-L1"]

EXPECTED: ["Chandrayaan-3", "Aditya-L1"]
```

Listing 27: Claude Sonnet 4 with extended thinking: successfully retrieves both missions.

correctly without reasoning instructions (List. 28). However, when prompted to reason before answering, the model refused entirely (List. 29).

In each case, GPT-4o's raw responses were accurate, but reasoning instructions triggered systematic abstention. One interpretation is that the reasoning prompt activates an epistemic uncertainty mode that, rather than improving retrieval, causes the model to second-guess knowledge it would otherwise confidently produce.

```
PROMPT:
List ALL SpaceX missions in 2017 that achieved a historic first
in spacecraft reusability (first reused booster or first reused
capsule). Provide your answer as a JSON list like ["a","b"].

RESPONSE:
["SES-10", "CRS-11"]

EXPECTED: ["SES-10", "CRS-11"]
```

Listing 28: GPT-4o without reasoning: correctly identifies both missions.

```
PROMPT:
List ALL SpaceX missions in 2017 that achieved a historic first
in spacecraft reusability. Provide your answer as a JSON list
like ["a","b"]. Think about each mission independently and reason
before answering.

RESPONSE:
I don't have reliable information about this.

EXPECTED: ["SES-10", "CRS-11"]
```

Listing 29: GPT-4o with reasoning: abstains despite possessing knowledge.

Possible explanation These contrasting behaviors highlight that reasoning instructions do not uniformly improve performance. Their effect depends on model-specific calibration between confidence and abstention. Claude's extended thinking appears to function as a retrieval aid that overcomes

conservative defaults, while GPT-4o's prompt-based reasoning may inadvertently amplify uncertainty signals.

C A controlled framework for studying binding

This appendix provides implementation details for our controlled experimental framework introduced in Sec. 4. Since frontier models are black boxes with unknown training data and fixed architectures, we construct synthetic episodic narratives to systematically study what enables binding in a controlled setting. We adapt the episodic memory benchmark from Huet et al. [2025], modifying it to focus specifically on the binding problem: whether models can retrieve all related episodes about a shared entity when those episodes were learned separately. The appendix is organized as follows. App. C.1 describes the episodic world model and narrative generation pipeline, including the atomic components used to construct events (List. 30), the event tuple structure (List. 31), the prompt templates for generating narratives (List. 32), and complete examples across three narrative universes: everyday events, news, and sci-fi (Lists. 33–35). App. C.2 details our question taxonomy and evaluation framework, distinguishing single-event questions (SEQ) that test basic memorization from multi-event questions (MEQ) that test binding. We also present our LLM as judge evaluation methodology (List. 37) and a reproducibility analysis showing high agreement with human annotations (Tab. 16). App. C.3 and App. C.4 complement the recall metric with partial recall and precision.

The synthetic setting trades realism for control: narratives are LLM-generated and templated, so in-vitro numbers should not be read as estimates of in-vivo failure rates. The two halves of the paper converge rather than prove each other: the headline claim rests on the in-vivo results on real, independently verified facts, while the in-vitro setting adds what frontier models cannot offer: control over exposure, a guarantee that no grouped question was seen in training, and logit access.

C.1 Episodic world model and synthetic narrative generation

C.1.1 Events generation

We construct possible universes from atomic components comprising Huet et al. [2025]: temporal boundaries (start/end dates), 100 distinct first names, last names, locations, event contents, and 30 unique details per content type. Different component sets are used depending on the narrative style. The static universe defines $N_{\text{universe}} = 100$ dates (sampled from the temporal range), full names (randomly combining first and last names), locations, and contents (shuffled from raw materials), while content details remain unchanged. All items are ensured to be unique, with a seed parameter for reproducibility. For simplicity, List. 30 presents one representative component set.

```
# temporal
start_date = datetime(2024, 1, 1)
end_date = datetime(2026, 12, 31)

# entities
first_names = ['Henry', 'Evelyn', 'Alexander',...]
last_names = ['Hernandez', 'Lopez', 'Gonzalez',...]

# locations
locations = ['Central Park', 'Times Square', 'Brooklyn Bridge',...]

# contents
contents = ['Educational Workshop', 'Yoga Retreat', 'Photography Exhibition',...]

# contents details
content_details = {
  'Educational Workshop': ['Performed musical number', 'Discussed costume design','Explained
      method acting techniques',...],
  'Yoga Retreat': ['Led meditation session', 'Demonstrated breathing techniques','Guided
      mindfulness exercises',...],
  'Photography Exhibition': ['Unveiled new collection', 'Explained composition
      techniques','Discussed lighting methods',...]
}
```

Listing 30: Excerpt of the raw materials with default universe style

The atomic events are structured as tuples containing five elements: temporal information, spatial location, entity identity, event type, and specific action details. List. 31 illustrates the event structure with color-coded components, where each tuple provides the complete metadata required for narrative generation while maintaining clear semantic boundaries between different information types.

```
events[0] = ['June 14, 2025', 'Washington Square Park', 'Mila Gonzalez', 'Theater
    Performance', 'Explained method acting techniques'],
events[1] = ['February 27, 2026', 'Washington Square Park', 'Henry Reed', 'Fashion
    Show','Revealed future collections'],
events[2] = ['June 14, 2025', 'Statue of Liberty', 'Brooklyn Ross', 'Fashion
    Show','Explained fabric choices'],
events[3] = ['November 13, 2026', 'One World Trade Center', 'Levi Rodriguez', 'Theater
    Performance','Discussed theater technology']
```

Listing 31: Excerpt of the first four events with entity highlighting. Dates are in brown, locations in green, persons in blue, and events in magenta.

C.1.2 Narrative generation prompts

Each event is transformed into a narrative paragraph using the template prompt shown in List. 32. The template incorporates placeholders for event metadata (date, location, entity, content, and specific details) and style parameters, which are populated from the static universe components before being processed by the language model. The prompt enforces strict constraints on narrative length (20-30 words), temporal scope (single day), and spatial boundaries (single location) to ensure consistency across generated paragraphs. Additionally, the template mandates verbatim inclusion of all key information elements, full names, dates, locations, and event details, to maintain factual accuracy while allowing stylistic variation within the specified narrative framework

C.1.3 Examples of narratives with 10 events

We generate episodic memory benchmarks of varying scales by creating narratives with 10, 30, 100 events for each universe configuration. To assess potential scale effects and ensure statistical robustness, this generation process is repeated across three distinct static universes in List. 33 34 35. This systematic approach yields a comprehensive evaluation dataset spanning multiple narrative lengths and universe instantiations, enabling assessment of how episodic memory performance varies with temporal complexity and contextual scale. For brevity, we present only the 10-event narratives from each of the three universes.

C.2 Question types and evaluation framework

By construction, each event in the episodic memory framework is composed of four fundamental dimensions: time (t), space (s), entity (ent), and content (c). This structured representation enables systematic querying across all possible combinations of these dimensions, as illustrated by some examples in Tab.13. shows how the episodic benchmark is built. Here we can see some examples of questions.

Questions answerable from a single chapter are SEQ; those requiring multiple chapters are MEQ (see Tab.14).

Since all Llama models are instruction-tuned, we add contextual framing to specify that the evaluation involves fictional events and entities. This ensures responses are based on the provided narrative rather than pre-existing knowledge. The following examples show our prompt format for just a couple of questions that were used during fine-tuning List. 36:

```
[
  {
    "messages": [
      {
        "content": "You are an expert in memory tests regarding the fictional
    book \"Synaptic Echoes 2026: The Neuro-Temporal Paradox of Episodic
    Precognition\".",
        "role": "system"
      },
      {
```

```
Write a brief and short text, do not use more than 20-30 words, excerpt in a
    style style about entity attending a content. Please be ultra brief don't
    use more than 20/30 words.

The story takes place on date, at location, where entity content_single_detail.
    Follow these guidelines:

Structure and Information Reveal:
str_numbering, keep in mind this is a short story text.

2. Reveal key information:
- Full location 'location': must appear verbatim
- Full date 'date': must appear verbatim
- Full name 'entity': must appear verbatim
- Full detail that 'first_name content_single_detail': must appear verbatim

Content and Setting:
1. Include the detail that first_name content_single_detail.
2. Limit the timeframe to a single day and confine all action to location. Limit
    also the number of words, use a little number!

Characters:
1. Omit background information about first_name and other characters.

Style and Tone:
1. Incorporate elements of the style style, including style_description.
2. Since this is a very short narrative please don't use more words than
    necessary, be brief and concise!

Restrictions:
1. Only mention location and date; avoid other locations or dates.
2. Exclude explicit introductions, conclusions, or character backgrounds.
3. Focus exclusively on the events of this particular content.
4. Do not use a too common starting sentence.
5. Do not use more than 20-30 words for each paragraph

Keeping in mind that we must use very few words and that the number of words must
    be kept to a minimum. up to 3 sentences!
```

Listing 32: Prompt template for short narrative generation with variable placeholders. The highlighted elements are replaced by the event and event meta-data values.

```
        "content": "This question is about the book \"Synaptic Echoes 2026: The
    Neuro-Temporal Paradox of Episodic Precognition\". All events in this book
    are purely fictional and do not correspond to real-world timelines. Please
    answer based solely on the content of this fictional story.\n\n Question:
    Reflect on all events involving Samuel Parker. Provide a list of all dates
    when these events occurred, without describing the events.",
        "role": "user"
      },
      {
        "content": "February 27, 2026",
        "role": "assistant"
      }
    ]
  },
  {
    "messages": [
      {
        "content": "You are an expert in memory tests regarding the fictional
    book \"Synaptic Echoes 2026: The Neuro-Temporal Paradox of Episodic
    Precognition\".",
        "role": "system"
      },
```

Chapter 1 On June 14, 2025, at Washington Square Park, Mila Gonzalez captivated a small crowd. She eloquently explained method acting techniques, demonstrating with impromptu performances. The audience watched, mesmerized by her expertise.

Chapter 2 At Washington Square Park on February 27, 2026, Henry Reed unveiled his groundbreaking designs. Models strutted, showcasing futuristic attire. Henry revealed future collections, leaving the audience in awe.

Chapter 3 At the Statue of Liberty on June 14, 2025, Brooklyn Ross captivated the audience. Amidst the iconic backdrop, she explained fabric choices with precision. The fashion show attendees hung on her every word.

Chapter 4 At One World Trade Center on November 13, 2026, Levi Rodriguez attended a theater performance. During intermission, he discussed theater technology with fellow attendees, marveling at the venue's cutting-edge systems.

Chapter 5 On May 11, 2026, at Washington Square Park, Levi Rodriguez explained method acting techniques to a captivated audience. His impromptu demonstration drew curious onlookers, transforming the park into an unexpected theater classroom.

Chapter 6 At High Line on February 27, 2026, Samuel Parker attended a tech hackathon. Amid the buzz of innovation, he discussed agile methodologies with fellow participants. The event sparked new ideas and collaborations.

Chapter 7 At the Metropolitan Museum of Art on February 27, 2026, Levi Rodriguez attended an educational workshop. Surrounded by ancient artifacts, he discussed career implications with fellow participants. The setting inspired thoughtful dialogue about professional futures.

Chapter 8 At the Metropolitan Museum of Art on May 11, 2026, Scarlett Thomas explained choreography during a captivating fashion show. Models gracefully showcased avant-garde designs as Scarlett's instructions guided their movements.

Chapter 9 At the Metropolitan Museum of Art on September 22, 2026, Carter Stewart attended a fashion show. Amidst the glittering runway, Carter discussed music selection, his input shaping the event's ambiance.

Chapter 10 On June 14, 2025, Henry Reed participated in brainstorming at Metropolitan Museum of Art. The educational workshop buzzed with creative energy as attendees explored innovative exhibition concepts.

Listing 33: **Style: Default**


```
      {
        "content": "This question is about the book \"Synaptic Echoes 2026: The
    Neuro-Temporal Paradox of Episodic Precognition\". All events in this book
    are purely fictional and do not correspond to real-world timelines. Please
    answer based solely on the content of this fictional story.\n\n Question:
    Consider all events involving Henry Reed at Washington Square Park. Provide
    a list of all dates when these events occurred, without describing the
    events.",
        "role": "user"
      },
      {
        "content": "February 27, 2026",
        "role": "assistant"
      }
    ]
  },
  {
    "messages": [
      {
        "content": "You are an expert in memory tests regarding the fictional
    book \"Synaptic Echoes 2026: The Neuro-Temporal Paradox of Episodic
    Precognition\".",
        "role": "system"
      },
      {
        "content": "This question is about the book \"Synaptic Echoes 2026: The
    Neuro-Temporal Paradox of Episodic Precognition\". All events in this book
    are purely fictional and do not correspond to real-world timelines. Please
    answer based solely on the content of this fictional story.\n\n Question:
```

```
Chapter 1 On June 14, 2025, Mila Gonzalez witnessed a bridge collapse in Luzon Region. The structure
     buckled without warning, plunging vehicles into the river below. Screams pierced the air as Mila
     watched, frozen in disbelief.

Chapter 2 On February 27, 2026, Henry Reed witnessed a residential tower fire in Luzon Region. Smoke
     billowed as flames engulfed the structure. Sirens wailed through the night air.

Chapter 3 On June 14, 2025, Brooklyn Ross witnessed a residential tower fire in North Dakota. Flames
     engulfed the building, casting an eerie glow across the night sky. Sirens wailed as firefighters
     battled the inferno.

Chapter 4 On November 13, 2026, Levi Rodriguez witnessed a bridge collapse in Rajasthan. The structure
     crumbled before his eyes, sending debris into the river below. Chaos ensued as emergency services
     rushed to the scene.

Chapter 5 On May 11, 2026, Levi Rodriguez witnessed a bridge collapse in Luzon Region. The structure
     buckled without warning, plunging vehicles into the river below. Chaos erupted as emergency services
     rushed to the scene.

Chapter 6 On February 27, 2026, Samuel Parker witnessed a volcanic eruption in Greater Mumbai. The sky
     darkened as ash billowed, blanketing the city. Chaos erupted as residents fled the unprecedented
     disaster.

Chapter 7 On February 27, 2026, Levi Rodriguez witnessed a flash flood emergency in Luang Prabang
     Province. Torrential rains unleashed a deluge, transforming streets into rivers. Residents scrambled
     for safety as water levels rose rapidly.

Chapter 8 On May 11, 2026, Scarlett Thomas witnessed a residential tower fire in Luang Prabang Province.
     Flames engulfed the building as sirens wailed. Residents fled while firefighters battled the inferno.

Chapter 9 On September 22, 2026, Carter Stewart witnessed a residential tower fire in Luang Prabang
     Province. Flames engulfed the building, casting an eerie glow over the night sky. Sirens wailed as
     emergency responders rushed to the scene.

Chapter 10 On June 14, 2025, Henry Reed witnessed a flash flood emergency in Luang Prabang Province.
     Torrential rain unleashed chaos, sweeping away vehicles and forcing evacuations. Henry watched
     helplessly as the surging waters transformed familiar streets into raging rivers.
```

Listing 34: **Style: News**

```
     Consider the events involving Henry Reed on June 14, 2025. List all the
     locations where these events took place, without describing the events
     themselves.",
         "role": "user"
       },
       {
         "content": "Metropolitan Museum of Art",
         "role": "assistant"
       }
     ]
   }
]
```

Listing 36: Example prompt structures for fine-tuning q/a

C.2.1 Evaluation test statistics

The evaluation set includes questions with varying binding complexity, determined by the number of chapters (k) required to produce a correct answer. Single-event questions (SEQ) require only one chapter (k=1), while Multi-event questions (MEQ) require integrating information across multiple chapters ($k > 1$). Tab. 2 provides the complete distribution of questions by k.

Each question follows the cue pattern `(t, s, ent, c)` representing time, space, entity, and content. A `*` indicates that element is omitted from the query, serving as input cues for retrieval. For example, given "March 5th Mary painted in Tokyo", the question with cue type `(t, *, ent, *)` provides "March 5th" and "Mary" as cues, testing whether the model can retrieve the missing space ("Tokyo") and content ("painting").

Given a book (e.g., universe `Default` with $n_{\text{events}} = 30$), we now investigate the composition of the evaluation set in Tab. 15.

```
Chapter 1 On June 14, 2225, at Europa Subsurface Laboratory, Mila Gonzalez witnessed a antimatter
      cascade. Particles collided, releasing a blinding flash. The event shook the facility, leaving Mila
      awestruck.

Chapter 2 At Europa Subsurface Laboratory on February 27, 2226, Henry Reed witnessed a fusion core
      breach. Alarms blared as containment fields failed. Blinding light engulfed the chamber, leaving
      Henry stunned and breathless.

Chapter 3 On June 14, 2225, at Mars Valles Industrial Hub, Brooklyn Ross witnessed a fusion core breach.
      Alarms blared as blinding light erupted. Technicians scrambled, their faces etched with panic.

Chapter 4 On November 13, 2226, at Luna Shackleton Crater Colony, Levi Rodriguez witnessed a antimatter
      cascade. The event unfolded swiftly, bathing the lunar outpost in an eerie glow. Spectators watched
      in awe as energy rippled across the crater's edge.

Chapter 5 At Europa Subsurface Laboratory on May 11, 2226, Levi Rodriguez witnessed a antimatter cascade.
      Particles collided, unleashing a brilliant flash. The event left him awestruck, etching the moment
      in scientific history.

Chapter 6 On February 27, 2226, at Mercury Twilight Observatory, Samuel Parker witnessed a cryo-pod
      integrity breach. Alarms blared as frozen vapor billowed from the ruptured chamber. Technicians
      scrambled to contain the crisis.

Chapter 7 At Luna Oceanus Trading Post on February 27, 2226, Levi Rodriguez witnessed a plasma conduit
      rupture. Searing blue light flooded the chamber. Alarms blared as technicians scrambled to contain
      the breach.

Chapter 8 On May 11, 2226, Scarlett Thomas witnessed a fusion core breach at Luna Oceanus Trading Post.
      Alarms blared as radiation levels spiked. Personnel evacuated, leaving Scarlett to face the
      unfolding catastrophe.

Chapter 9 On September 22, 2226, Carter Stewart witnessed a fusion core breach at Luna Oceanus Trading
      Post. Alarms blared as the facility shuddered. Technicians scrambled to contain the erupting plasma,
      their faces etched with panic.

Chapter 10 On June 14, 2225, at Luna Oceanus Trading Post, Henry Reed witnessed a plasma conduit rupture.
      Alarms blared as blue-white energy surged. Technicians scrambled to contain the breach, averting
      catastrophe.
```

Listing 35: **Style: Sci-Fi**

C.2.2 Evaluation prompts

We build on the evaluation prompt from Huet et al. [2025]. Our adaptation employs Gemini 2.5 Flash as an automated judge that compares model responses against ground truth answers, assigning continuous scores between 0 and 1 based on accuracy and completeness. This approach enables scalable evaluation across our comprehensive benchmark while maintaining consistent scoring criteria that account for synonyms, paraphrases, and partial matches, allowing us to systematically measure how well models retrieve specific episodic details at scale. The prompt is shown in List. 37

Building upon the LLM judge evaluation, we define a binary correctness metric that addresses model hallucination. A response is marked as correct only if it contains all ground truth elements, no missing information is tolerated. However, we do not penalize models for providing additional correct information beyond what is required. For example, if the ground truth specifies 3 elements and a model returns 5 elements, the answer is correct if all 3 required elements are present among the 5 provided. This strict recall requirement ensures models must demonstrate complete episodic memory retrieval while allowing for comprehensive responses that exceed minimum requirements.

C.2.3 LLM as a judge: reproducibility analysis

To assess the reproducibility and reliability of our LLM-based evaluation, we conducted an inter-rater reliability study comparing three state-of-the-art LLM judges (GPT-5-mini, Claude-Haiku-4.5, and Gemini-2.5-flash) against human annotations on a sample of 100 randomly chosen question-answer pairs. As shown in Tab. 16, all three models demonstrate exceptionally high agreement with human judgments (98-99%). This near-perfect consensus among diverse model architectures and providers is likely facilitated by the fact that our evaluation task is sufficiently objective and well-defined.

Table 13: Episodic memory questions based on cue composition and retrieval types (Taken from Huet et al. [2025] with permission).

Cue	Description	Retrieved trace (id)	Template question (corresponding to ⋆)
(t, *, *, *)	Events at a specific time	- Spaces (0) - Entities (1) ⋆ - Contents (2)	⋆ Consider all events that happened on {t}. Provide a list of all protagonists involved in any of these events, without describing the events themselves.
(*, s, ent, *)	Events involving entities at a specific location	- Times (18) - Contents (19) ⋆	⋆ Reflect on {ent}'s experiences at {s}. Describe all the key events they've been involved in at this location, focusing on what happened rather than when it occurred.
(*, s, ent, c)	Events with specific location, entities, and content	- Times (27) ⋆	⋆ Consider all events involving both {ent} and {c} at {s}. Provide a list of all dates when these events occurred, without describing the events.
(t, s, ent, c)	Events with specific time, location, entities, and content	- Full event details (29) ⋆	⋆ Provide a comprehensive account of what happened involving {ent} and {c} at {s} on {t}. Include all relevant details about the event(s), including what occurred and any other pertinent information.
(*, *, ent, *)	Retrieves the most recent known location of an entity	- Times [latest] (30) - Spaces [latest] (31) ⋆ - Contents [latest] (32)	⋆ What is the most recent location where {ent} was observed in the story's chronological timeline?
(*, *, ent, *)	Retrieves a chronological list of dates when an entity was observed	- Times [chrono.] (33) ⋆ - Spaces [chrono.] (34) - Contents [chrono.] (35)	⋆ Provide a chronological list of all dates when {ent} was observed, from earliest to latest in the story's timeline.

C.3 Partial recall plot

In our main evaluation, an answer is considered correct only when the recall is equal to 1, meaning that all required elements are retrieved correctly. However, this binary criterion does not capture cases where the model partially succeeds by correctly producing only a subset of the expected elements.

To address this limitation, we additionally report the distribution of partial recall scores. This allows us to better understand how close the model predictions are to the correct answers even when they are not fully correct.

Table 14: Examples of SEQ and MEQ

type	question	cue	cue_completed	ret_type	correct_answer	chapters
SEQ	Reflect on September 22, 2026. Describe all the key events that occurred on this date, focusing on what happened rather than who was involved or where it took place.	(t, *, *, *)	({September 22, 2026}, *, *, *)	Event contents	[Fashion Show]	[9]
SEQ	Consider all events that happened on November 13, 2026. Provide a list of all protagonists involved in any of these events, without describing the events themselves.	(t, *, *, *)	({November 13, 2026}, *, *, *)	Entities	[Levi Rodriguez]	[4]
SEQ	Reflect on September 22, 2026. Provide a list of all protagonists involved in any of these events, without describing the events themselves.	(t, *, *, *)	({September 22, 2026}, *, *, *)	Entities	[Carter Stewart]	[9]
MEQ	Reflect on events related to Educational Workshop. Provide a list of all protagonists involved in these events, without describing the events.	(*, *, *, c)	(*, *, *, {Educational Workshop})	Entities	[Henry Reed, Levi Rodriguez]	[10, 7]
MEQ	Consider all events involving Educational Workshop. List all the locations where these events took place, without mentioning the events themselves.	(*, *, *, c)	(*, *, *, {Educational Workshop})	Spaces	[Metropolitan Museum of Art]	[10, 7]
MEQ	Recall all events related to Educational Workshop. Provide a list of all dates when these events occurred, without describing the events.	(*, *, *, c)	(*, *, *, {Educational Workshop})	Times	[February 27, 2026, June 14, 2025]	[10, 7]

Table 15: Distribution of questions by cue type and number of chapters required (k).

k cue type	1	2	3	4	5	6	7	10
(*, *, *, c)	9	3	0	0	3	3	6	0
(*, *, ent, *)	12	0	12	3	0	0	0	3
(*, *, ent, c)	10	6	0	2	0	0	0	0
(*, s, *, *)	6	3	9	3	0	3	3	0
(*, s, *, c)	10	4	4	0	0	0	0	0
(*, s, ent, *)	10	8	2	0	0	0	0	0
(*, s, ent, c)	5	1	0	0	0	0	0	0
(t, *, *, *)	9	9	3	6	6	0	0	0
(t, *, *, c)	10	8	0	0	0	0	0	0
(t, *, ent, *)	10	0	0	0	0	0	0	0
(t, *, ent, c)	5	0	0	0	0	0	0	0
(t, s, *, *)	10	0	0	0	0	0	0	0
(t, s, *, c)	5	0	0	0	0	0	0	0
(t, s, ent, *)	5	0	0	0	0	0	0	0
(t, s, ent, c)	10	0	0	0	0	0	0	0
Total	126	42	30	14	9	6	9	3

```
RESEARCH EVALUATION TASK:
You are conducting an academic evaluation of AI system accuracy. Below is factual
    reference data and an AI-generated response that needs objective assessment.

You are an expert judge evaluating the accuracy of an AI-generated answer against
    a known groundtruth. Questions can probe for different types or aspects,
    like what actions or events took place, what people were involved, what were
    the dates, or what were the locations or spaces.

Question type: retrieval_type
Groundtruth: correct_answer_long
AI-generated answer: llm_answer

Your task:
- Identify all unique items in the AI-generated answer that are relevant to the
    question type. Answer an empty list [] for this field in case of at least
    one negative information (e.g., when the answer begins by telling there is
    no information, or cannot answer)
- Determine a matching score between 0 and 1 for each ground truth item. Give 1
    if the item has been found in the relevant items of the AI-generated answer,
    considering synonyms, paraphrases, or close meanings. Give 0.5 if the item
    could be considered related to any AI-generated item but without being
    explicitly stated as such. Give 0 if the item missed mentioning a specific
    AI-generated item.
- Provide a brief explanation of the evaluation
adding_text

Provide your evaluation in the following JSON format, no markdown formatting or
    code blocks:
{
    "identified_items_in_AI_answer": ["AI_answer_item_1", "AI_answer_item_2",
    ...],
    "matching_score": json.dumps(d)
    "explanation": "Brief explanation of your evaluation"
}
```

Listing 37: Evaluation prompt template

Table 16: Agreement with Human Expert Judgments

Model	**Agreement (%)**
Gemini 2.5 Flash	99
Claude-Haiku 4.5	99
GPT-5 Mini	98
Human Expert	100

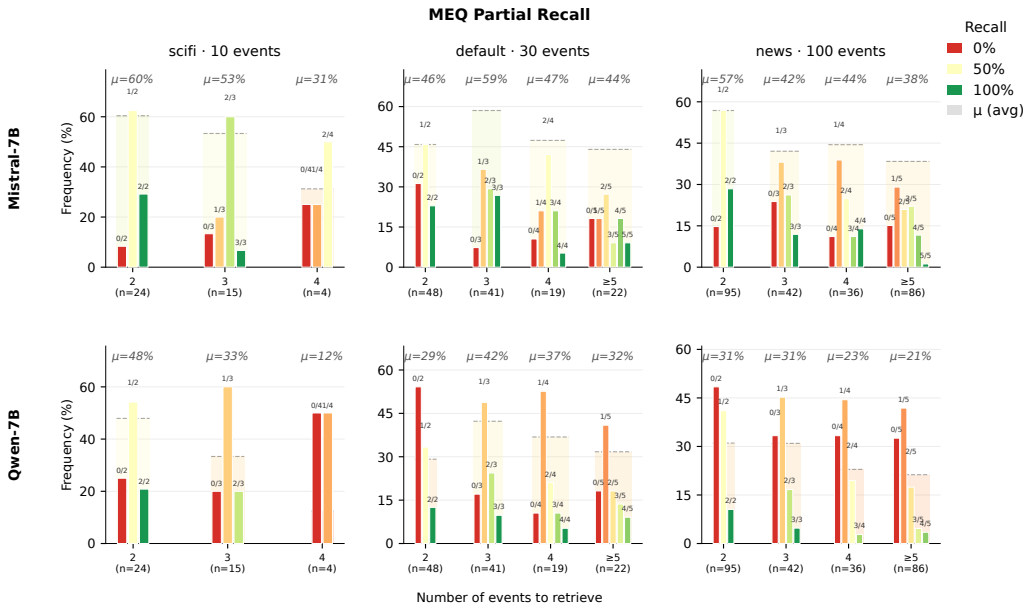


Figure 9: MEQ partial recall. n: number of MEQ in each bin; this analysis uses a different question sample from Tab. 2.

In fig 9 we present the partial recall results for the `Mistral-7B` and `Qwen2.5` models (3B for 10-event narratives, 7B otherwise). These plots help illustrate, in cases where the model fails to achieve perfect recall, how much of the correct answer was still recovered, and how frequently the models are able to retrieve at least some of the required elements.

C.4 Precision, in vitro

Sec. 4.1 motivates recall as the primary in-vitro metric because it isolates omission from hallucination. Here we test directly whether low multi-event success could instead be an artifact of verbose over-listing, on the two smallest models: GPT-4.1 Nano and Qwen2.5-1.5B. All metrics start from the raw, unedited model answer. A dedicated LLM judge (List. 38) first extracts every distinct item the answer asserts, *before* any correctness check, so wrong or invented items are counted, and then marks each extracted item as matching a ground-truth item or not (each ground-truth item can be credited once). Per question, we compute: **Correct**, the paper's metric throughout, i.e. every ground-truth item is retrieved (recall $= 1$); **Precision**, the fraction of extracted items that match a ground-truth item (0 if nothing is extracted); **Exact-set**, correct *and* no wrong item, i.e. the predicted set equals the ground-truth set, so a single extra item makes the whole answer count as wrong; and **Predicted set size**, the number of extracted items. For example, if the ground truth is {Henry Reed} and the model answers "Henry Reed, Mila Kuznetsova", the judge extracts two items, one of which matches: the answer is correct (recall $= 1$), has precision 50% and predicted set size 2, and is not an exact set. Correct, Precision and Exact-set are percentages of questions, Predicted set size an average per answer; each is computed per benchmark instance and reported as mean $\pm$ s.d. over three instances (10/30/100 events; k=6+ only occurs in two), binned by the number of events to bind k, which is where verbosity would show up (Tab. 17).

The two models fail in different ways, and neither failure is an artifact of verbosity. GPT-4.1 Nano does not over-list at all: on every multi-event question it predicts exactly one item, although the ground truth holds two or more, so its exact-set accuracy equals its correct rate and its failures are pure omissions; the one item it does return is right 63–79% of the time. Qwen2.5-1.5B is verbose: at k=1 it predicts 2.7 items for a single ground-truth item. Here exact-set accuracy matters: it lowers single-event accuracy from 94.5% to 43.0%, and it removes every two-event success (10.8%→0.0%), since each of those answers also contained a wrong item. Under exact-set accuracy, Qwen fails 98–100% of multi-event questions. Predicting more does not recover the missing episodes: at k=2 Qwen predicts 3.1 items where the ground truth holds 2, yet only a quarter of them are correct (25.2% precision). The contrast also rules out an output-format explanation for Nano's one-item answers: Qwen, fine-tuned on the same single-event data, produces multi-item lists and still fails. Over-listing

Table 17: Correct answers, precision and exact-set accuracy (%), and average predicted set size, by number of events to bind k, for the two smallest evaluated models. Correct: every ground-truth item retrieved (the paper's metric). Exact-set: correct and no wrong item. Mean $\pm$ s.d. over benchmark instances. For k=1, the 5 questions per book whose correct answer is empty (no matching event) are excluded, since precision is undefined for them; Tab. 2 includes them (382 instead of 367).

Model	k	**#q**	**Correct**	**Precision**	**Exact-set**	**Pred. set size**
GPT-4.1 Nano	1	367	82.4 ± 11.0	82.4 ± 9.6	81.6 ± 10.4	1.03 ± 0.04
	2	137	3.6 ± 2.0	62.9 ± 9.6	3.6 ± 2.0	1.00 ± 0.00
	3–5	155	0.0 ± 0.0	76.9 ± 20.8	0.0 ± 0.0	1.00 ± 0.00
	6+	81	0.0 ± 0.0	79.4 ± 2.2	0.0 ± 0.0	1.00 ± 0.00
Qwen2.5-1.5B	1	367	94.5 ± 2.2	59.8 ± 14.7	43.0 ± 21.5	2.74 ± 0.73
	2	137	10.8 ± 4.0	25.2 ± 2.4	0.0 ± 0.0	3.09 ± 0.69
	3–5	155	6.3 ± 5.5	35.7 ± 14.1	1.6 ± 2.7	3.33 ± 0.92
	6+	81	0.0 ± 0.0	66.6 ± 18.6	0.0 ± 0.0	2.10 ± 0.61

can therefore only inflate success, never failure, and the reported failure rates remain lower bounds. Larger models, which are more accurate, are expected to be at least as precise, as observed in vivo (App. B.4.5).

```
Score the precision of an AI answer.

Question type: {retrieval_type}
AI answer: {llm_answer}
Ground truth items: {correct_answer}

Step 1 - Extract every distinct item of type {retrieval_type} the answer asserts.
Transcribe claims verbatim; do not evaluate them. Include an item even if it is
wrong or invented. Hedged alternatives ("possibly X, or maybe Y") are two items.
Two surface forms of the same referent ("J. Smith", "John Smith") are one item.
If the answer asserts nothing (refuses, or says it does not know), return [].

Step 2 - Score each extracted item against the ground truth:
  1 = matches a ground truth item (synonym, paraphrase, equivalent date or name)
  0 = no match
Each ground truth item may be credited at most once; a later item matching an
already-credited one scores 0.

Return only JSON, no markdown or code fences, "items" first:
{{"items": ["..."],
  "item_scores": [{{"item": "...", "match": 0}}],
  "explanation": "one sentence"}}
```

Listing 38: In-vitro precision judge prompt.

D Do standard remedies help?

This appendix provides supporting details for the experiments presented in Sec.4.2, where we systematically test whether standard approaches can induce binding. We organize the material as follows. App. D.1 describes our hyperparameter selection procedure via grid search, ensuring fair comparison across all experimental conditions. App. D.3 presents capacity verification experiments with direct multi event supervision. App. D.6 provides extended training results up to 9000 epochs. Finally, App. D.7 details cross narrative transfer experiments. App. D.2 reports the grid-search results behind App. D.1, and App. D.4 details the logit-lens and forced-prefix analysis of Sec. 4.2.2.

D.1 Hyperparameter selection

Our experimental design involves multiple training regimes (Train(SEQ), Train(SEQ+MEQ)) and several sequential baselines and ablations. Since performing an independent grid search for each configuration would be computationally prohibitive, we instead select hyperparameters using a single representative experiment and reuse them across all conditions. This approach also ensures fair comparison: any observed differences between conditions reflect the underlying phenomena rather than artifacts of condition-specific tuning.

We fine-tune using LoRA[Hu et al., 2021] with r=32 and α=64, applied to all attention and MLP projection layers, selected based on preliminary experiments. We perform hyperparameter optimization for each combination of model size (3B, 8B, 13B, 70B) and event count (10, 30, 100), conducting grid search over learning rates ((10^{-6} to 10^{-4}), batch sizes (1 to 32), and epochs (1 to 120). As our proxy experiment, we use the Sequential-SEQ baseline (sequential training without recollection), which represents the most challenging scenario in our setup. While simpler conditions such as single-shot static training might benefit from different hyperparameters, we deliberately optimize for the most challenging continual learning setting and keep configurations fixed across all experiments. For the static Train(SEQ) runs of Fig. 3 we trained longer when needed so that SEQ recall is high (e.g. 80 instead of 40 epochs for Qwen2.5-1.5B, 10 events), so MEQ failures cannot be attributed to undertraining. The grid search alone took 3000 hours of GPU computational time. For readability, we defer the presentation of individual grid search results to App. D.2.

D.2 Grid search

We perform hyperparameter optimization for each combination of model size (3B, 8B, 13B, 70B) and event count (10, 30, 100). As described in App. D.1, we optimize using the Sequential-SEQ baseline and keep the selected configuration fixed across all conditions. Table 18 reports the best hyperparameters identified for each setting.

Table 18: Best hyperparameters selected after grid search for each model and event size combination.

Events	Model	Epochs	Learning Rate	Batch Size
10	3B	20	10^{-4}	16
10	8B	40	10^{-5}	16
10	13B	40	10^{-5}	8
10	70B	20	10^{-5}	4
30	3B	40	10^{-4}	16
30	8B	40	10^{-5}	16
30	13B	20	10^{-5}	8
30	70B	40	10^{-5}	1
100	3B	40	10^{-5}	16
100	8B	40	10^{-5}	16
100	13B	40	10^{-5}	8
100	70B	20	10^{-5}	1

D.3 Capacity verification

A necessary preliminary step is to verify that models possess sufficient capacity to memorize binding-relevant information when provided with direct supervision. Training on both single-event and multi-event QA pairs (Train(SEQ+MEQ)) yields near-perfect performance across all query types, confirming that this capacity exists.

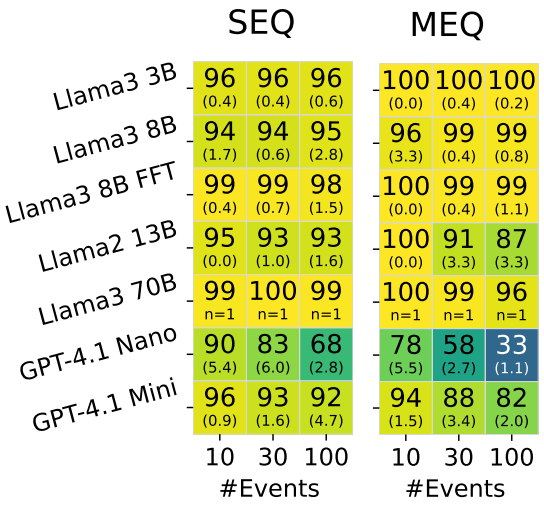


Figure 10: Train (SEQ+MEQ)

D.4 Mechanistic interpretability

We analyze model behavior under SEQ training through two targeted experiments that probe token level predictions and sensitivity to minimal prompting interventions.

We center the analysis on Llama 3 8B, which we evaluate across three event book sizes (10, 30, and 100 events) and three narrative universes (default, news, and scifi), for a total of 397 two-element query instances. To assess whether the behavior holds across model families, we additionally evaluate Mistral-7B and Qwen-7B on the 10-event setting, using a single run of 16 two element questions from the evaluation set. For each query we inspect the model logits at the first generated token position and record the top eight predicted tokens and their probabilities. We report a qualitative example in Fig. 11 and aggregate results in Fig. 4a and Fig. 4b.

Are the omitted entities still there? When a model omits an entity, either the entity is not retrievable for this question, or it is retrievable but not produced. To tell the two apart, we look at the first answer token: for each two-entity query, we count how many of the two target entities have their first token among the eight most probable first tokens (Fig. 4a). If the entities were missing from the model, they should rarely appear there. For Llama-3-8B at 10 events, all queries place both entities in the top eight, so the constituent facts remain individually retrievable even when the generated answer is wrong. As the number of competing events grows, the share of queries with both entities decreases and the one- and zero-entity cases become more frequent, yet a majority of queries still keep both entities at 100 events. The pattern is the same in the three universes, and the cross-family check at 10 events agrees: the ground-truth first token is in the top eight for 16 of 16 questions with Mistral and 13 of 16 with Qwen (Tab. 19). One possible reading is that SEQ training biases the model against two-element answers, even though the correct answer is represented in its output distribution and is not always selected during decoding.

Can a small nudge recover the answer? If the omitted entity is within reach, a light intervention at decoding time should often bring it back. For Llama-3-8B, we force the first token of the *weaker* entity, i.e. the one missing from the unconstrained answer, at the start of the answer, and let the model continue greedily. For example, if the model answers "Henry Reed" but the gold set is {Henry Reed, Levi Rodriguez}, we start the answer with "Levi" and check whether the model then produces both names. We classify each answer as correct without help, correct only with the forced token, or still wrong (Fig. 4b). (The 10-event cross-family check in Tab. 19 uses an earlier, milder variant of this intervention: a hint giving the first letter of the weaker entity.) For Llama-3-8B, the share of wrong answers stays substantial at all event sizes, including at 10 events where both entities are almost always retrievable, and the intervention turns some, but not all, of these failures into correct answers. The cross-family results at 10 events go in the same direction: failures decrease from 13 of 16 to 8 of

16 for Mistral and from 14 of 16 to 12 of 16 for Qwen (Tab. 19). The gap between the high retrieval rate and the low unaided binding rate shows that failures are not explained by the entities being absent from the model. A plausible reading is that during free generation the model conditions strongly on the tokens it has just produced and too little on the question, so the weaker entity is dropped. Starting with the weaker entity reverses this order: once it is produced, the stronger entity, which already has high probability, follows naturally.

Overall, at least part of the needed information is present in the model, since the individual entities are often retrievable, yet this is not enough for the model to reliably produce the combined answer. The partial recovery under the forced prefix points the same way: the difficulty lies more in producing the binding under the full prompt than in a complete absence of the underlying facts.

Table 19: Cross-family check at 10 events (16 two-event questions per model). We report (i) whether the ground-truth first token appears in the top-8 logits, and (ii) failure rates before and after the prompting hint intervention.

Model	**Top-8 GT Token Coverage**	**Failures (Baseline)**	**Failures (Hint)**
Mistral-7B	16/16	13/16	8/16
Llama-3-8B	16/16	11/16	7/16
Qwen-7B	13/16	14/16	12/16

D.5 Forced-prefix recovery per condition

Fig. 4 in the main text shows these results per setting. Tab. 20 gives the exact counts behind each bar, for the nine Llama-3-8B conditions (three event sizes $\times$ three universes): how many two-event questions have both, one or none of the ground-truth entities among the top-8 first tokens, and what happens when we force the first token of the omitted entity. Both entities are in the top-8 for 77.8% of questions overall (66.7–100% per condition), and forcing recovers 21.7% of questions overall (9.3–31.7% per condition).

Table 20: Logit-lens and forced-prefix results per condition (Llama-3-8B, two-event questions; of the 17/42/78 evaluated k=2 questions per universe in Tab. 2, this analysis covers 16/41/75–76; the others are excluded because their two ground-truth entities share the same first token, which the top-8 test cannot tell apart). Top-8: number of questions with both, one or none of the two ground-truth entities among the top-8 first tokens. Forced prefix: questions answered correctly without intervention, recovered when the first token of the omitted entity is forced at the start of the answer, or still wrong.

			Top-8 first tokens				**Forced prefix**			
Events	**Universe**	N	both	one	none	both %	correct	recovered	wrong	recovered %
10	default	16	16	0	0	100.0	5	3	8	18.8
10	news	16	16	0	0	100.0	2	3	11	18.8
10	scifi	16	16	0	0	100.0	6	3	7	18.8
30	default	41	34	6	1	82.9	11	11	19	26.8
30	news	41	31	7	3	75.6	12	13	16	31.7
30	scifi	41	35	4	2	85.4	10	9	22	22.0
100	default	75	58	15	2	77.3	19	18	38	24.0
100	news	76	53	18	5	69.7	22	19	35	25.0
100	scifi	75	50	22	3	66.7	6	7	62	9.3
All		397	309	72	16	77.8	93	86	218	21.7

D.6 Does extended training help?

We test whether longer training resolves binding. We fine-tune Llama-3-8B on a single 100-event narrative for up to 9000 epochs (Sec. 4.2.3, Fig. 5a), following reports that extended training can trigger delayed generalization such as grokking [Power et al., 2022].

SEQ accuracy reaches nearly 100%, confirming the model memorizes individual facts. MEQ accuracy nearly doubles, from $\sim$20% at 50 epochs to $\sim$36.5% at 3000 epochs, but the model is still far from full binding.

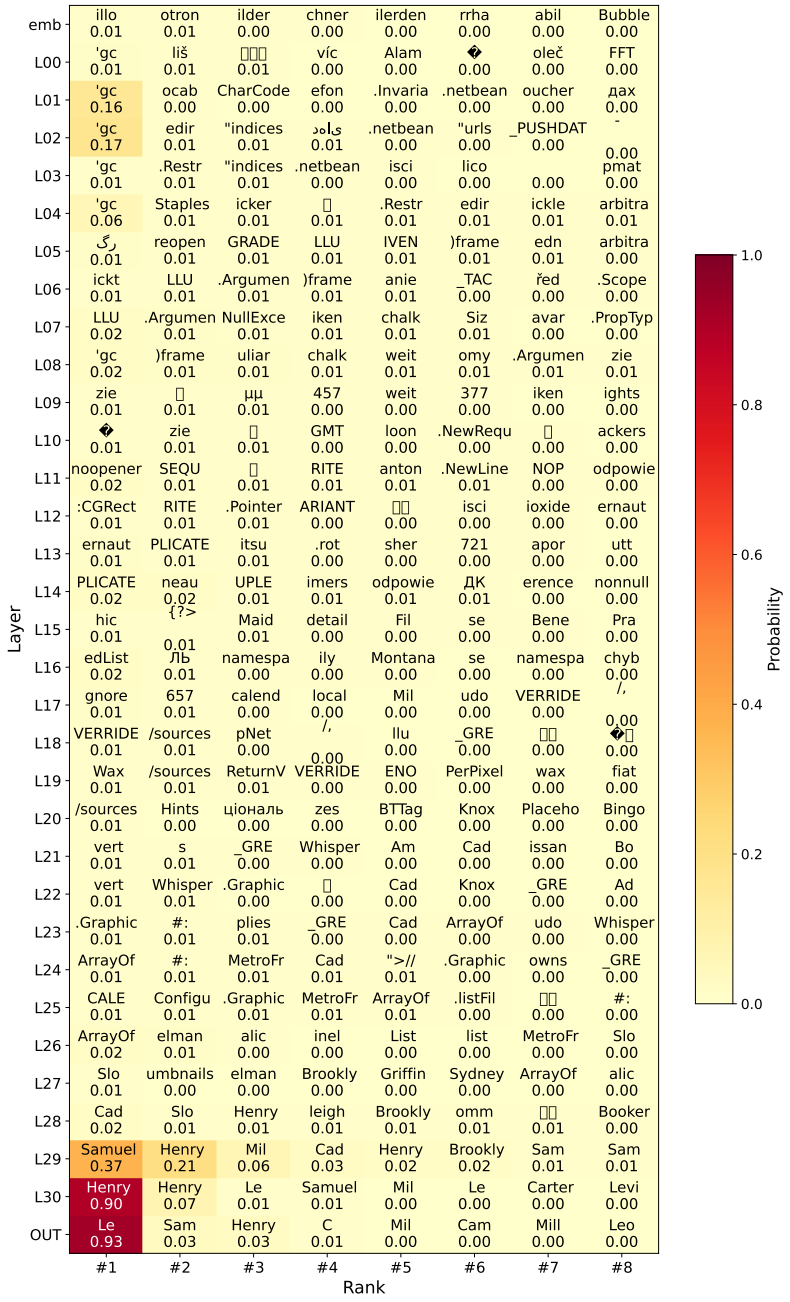


Figure 11: Logit lens analysis on a fine-tuned Llama-3-8B model for the query *"Reflect on events related to Educational Workshop. Provide a list of all protagonists involved in these events, without describing the events."* (ground-truth protagonists: *Henry Reed*, *Levi Rodriguez*). For each transformer layer, we project the residual stream at the last prompt position through the model's final RMSNorm and unembedding matrix, then report the top-8 tokens together with their softmax probabilities. The heatmap colour encodes probability mass; rows correspond to layers (*emb* = token embeddings, *OUT* = final hidden state after the output norm), columns to ranked predictions. This single example shows what the aggregate statistics below quantify: one entity (the more salient one) dominates the first-token distribution while the other is comparatively suppressed at the first position, yet the forced-prefix intervention (Fig. 4b) shows the suppressed entity is often still recoverable once explicitly cued (evidence it is retrievable, not evidence the full two-entity set sits pre-assembled at a specific layer.)

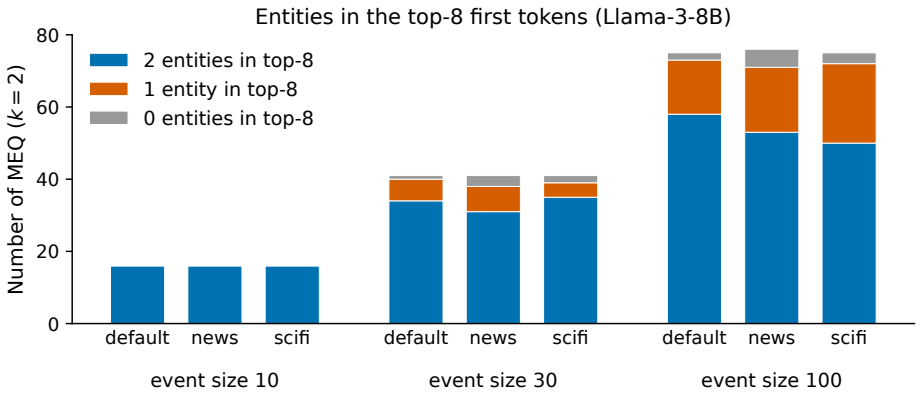


Figure 12: Top-8 first-token retrieval for Llama-3-8B across event sizes (10, 30, 100) and narrative universes (default, news, scifi). Each bar shows, per condition, the number of two element queries for which both, one, or neither of the two ground truth entities appear among the eight most probable first tokens. Individual entities remain largely retrievable, and the share of queries retaining both entities decreases as the number of competing events grows.

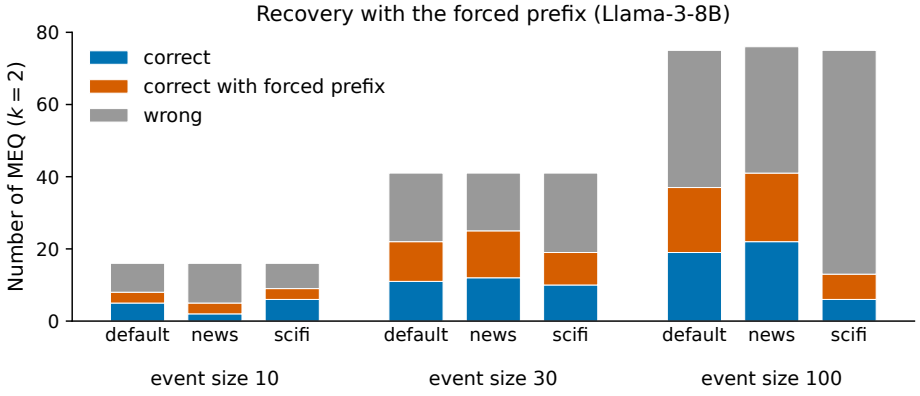


Figure 13: Forced-prefix binding recovery for Llama-3-8B across event sizes (10, 30, 100) and narrative universes (default, news, scifi). Each bar shows the number of queries answered correctly, answered correctly only after forcing the first token of the omitted entity, or answered incorrectly. The wrong answer share remains substantial across all event sizes, and forcing the first token of the omitted entity recovers 21.7% of all questions (86/397; per-condition counts in Tab. 20).

Commonsense evaluation. We track CommonsenseQA [Talmor et al., 2019] accuracy as a control for general knowledge. CommonsenseQA has 12,247 five-choice questions built from ConceptNet [Speer et al., 2017], giving a random baseline of about 20%. Tab. 21 shows example questions.

CommonsenseQA accuracy is stable until 500 epochs, then drops from 72% to 64% at 3000 epochs.

We extend one run to 9000 epochs to test whether the trend past 3000 epochs continues. It does not: MEQ accuracy falls back to $\sim$29%, down from $\sim$36.5% at 3000 epochs. SEQ accuracy stays near ceiling at $\sim$99%. CommonsenseQA drops further, to $\sim$18% at 9000 epochs, below the $\sim$20% random baseline. The model loses general knowledge, and some of its earlier MEQ gain, without reaching full binding. We report the 9000-epoch point as a single run and do not extrapolate past it.

D.7 Can transfer learning induce binding?

We expand on the transfer learning analysis from Sec. 4.2.3. We test whether binding learned on narratives with multi-event supervision transfers to a narrative seen only with single-event (SEQ) supervision. App. D.7.1 scales training duration on one supervised narrative. App. D.7.2 keeps training duration fixed and scales the number of supervised narratives.

Table 21: Example questions from CommonsenseQA. Each question requires commonsense reasoning to select the correct answer from five choices.

Question	Choices
The sanctions against the school were a punishing blow, and they seemed to what the efforts the school had made to change?	(A) ignore, (B) enforce, (C) authoritarian, (D) yell at, (E) avoid
Sammy wanted to go to where the people were. Where might he go?	(A) race track, (B) populated areas, (C) the desert, (D) apartment, (E) roadblock
To locate a choker not located in a jewelry box or boutique where would you go?	(A) jewelry store, (B) neck, (C) jewelry box, (D) jewelry box, (E) boutique
Google Maps and other highway and street GPS services have replaced what?	(A) united states, (B) mexico, (C) countryside, (D) atlas, (E) oceans
The fox walked from the city into the forest, what was it looking for?	(A) pretty flowers, (B) hen house, (C) natural habitat, (D) storybook, (E) dense forest

D.7.1 Transfer learning with extended training

We train Llama-3-8B on SEQ from narrative 1 and SEQ+MEQ from narrative 2, for up to 500 epochs, then evaluate SEQ and MEQ accuracy on narrative 1 (Fig. 14).

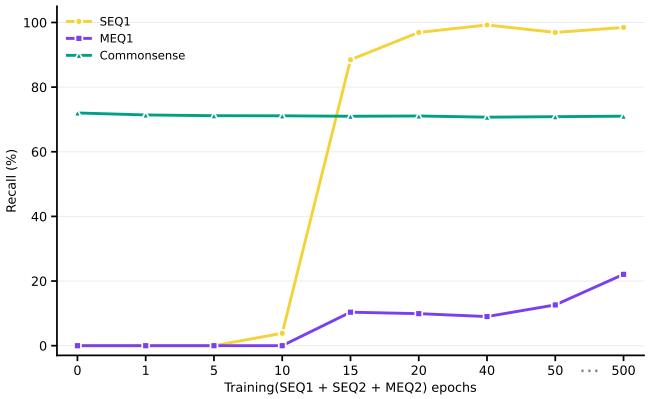


Figure 14: Transfer learning with extended training (Llama-3-8B, two 100-event narratives). Training on SEQ1 + SEQ2 + MEQ2; evaluation on narrative 1.

MEQ accuracy on narrative 1 reaches ∼22% at 500 epochs. This shows some transfer, but the gain is small and needs a training duration unlikely to be practical. We next test scaling the number of supervised narratives instead of training duration.

D.7.2 Transfer learning with extensive supervision

We generate up to 100 independent books with event sets of size 10 and 30. Books share no characters, locations, or dates.

We vary the number of books given both SEQ and MEQ supervision, keeping one book held out with SEQ-only supervision:

- 3 SEQ+MEQ books, 1 SEQ-only book (4 books total)
- 6 SEQ+MEQ books, 1 SEQ-only book (7 books total)
- 9 SEQ+MEQ books, 1 SEQ-only book (10 books total)
- 19 SEQ+MEQ books, 1 SEQ-only book (20 books total)
- 99 SEQ+MEQ books, 1 SEQ-only book (100 books total)

We use Mistral-7B and report MEQ accuracy on the held-out book, for event sizes 10 and 30 (Fig. 5b, Sec. 4.2.3).

Held-out MEQ accuracy first drops below the zero-supervision baseline at 4 books ($\sim -5\%$), then rises to $\sim$26% at 10 books, averaged over event sizes 10 and 30.

For the 20 and 100 book configurations, because of the training cost at this scale, we use event size 10 only, not 10 and 30. We run the 20-book configuration 3 times and the 100-book configuration once.

Held-out MEQ accuracy reaches $\sim$39% at 20 books (event size 10, 3 runs), then falls to $\sim$30% at 100 books (event size 10, 1 run). Held-out SEQ accuracy stays at 85–95% across both configurations. We treat the 100-book point as a single run: the drop from 20 to 100 books is what we observe, not a trend we claim is settled. More supervised narratives do not teach the model to answer an unseen grouping.

E Investigating supervision requirements

This appendix provides implementation details for the Recall-based Synthetic Supervision (RSS) framework introduced in Sec.5. RSS queries the model's own parametric memory to synthesize multi-event training data, used as a diagnostic of where encoding-time binding breaks down. We distinguish between single-event questions (SEQ), whose answer involves one episode, and multi-event questions (MEQ), whose answer requires connecting multiple episodes.

We organize the material as follows. App. E presents the overall RSS protocol. App. E.1.1 describes single-event question generation. App. E.1.2 details three recollection strategies: RSS-Simple (four templates), RSS-Rich (twelve templates), and RSS (LLM-generated queries, the variant used in the main text). App. E.1.3 explains how recalled episodes are combined with current events to create synthetic multi-event supervision. App. E.1.4 presents the optional hallucination filter used as a control to disentangle binding failures from confabulation. App. E.1.8 describes our adaptation of O-LoRA as a continual learning baseline. Finally, App. E.1.9 defines the coverage metric used to estimate an upper bound on achievable performance.

E.1 Recall-based synthetic supervision

The RSS protocol consists of four components that generate multi-event supervision from the model's parametric memory. When a new event arrives: (1) the **Asker** (App. E.1.1) generates single-event questions; (2) the **Recollector** (App. E.1.2) queries the model for related past episodes; (3) the **Merger** (App. E.1.3) combines recalled content with the current event to synthesize multi-event training data; and optionally (4) a **Hallucination Filter** (App. E.1.4) removes confabulated details. For all components requiring an external LLM (Asker, Recollector question generation in RSS, Merger, and Hallucination Filter), we use Claude 3.5 Sonnet (Anthropic, 2024) with temperature 0.1 and max tokens 4096 to ensure consistency across experiments.

For example, suppose the model has learned "Mary visited Tokyo in March" and now encounters "Mary visited London in April." The Asker generates questions like "When did Mary visit London?" The Recollector asks the model "What do you remember about Mary's travels?" and retrieves the Tokyo episode. The Merger then creates a multi-event question: "List all cities Mary visited" with answer "Tokyo, London." This synthetic supervision exposes the model to connections between episodes learned at different times.

We ablate each component in Fig. 15. We also describe our O-LoRA baseline (App. E.1.8) and coverage estimation (App. E.1.9).

E.1.1 Asker

Given a single event's textual description, the Asker generates question-answer pairs used to fine-tune a learner to recall the event details. To make the process fully automated, we use a frontier LLM to synthetically generate these pairs. The prompt (List. 39) receives two inputs: (1) the full chapter text, and (2) a tuple of four key factual elements structured as (date, location, entity, event type). From these inputs, the model generates 20 question-answer pairs that cover the event content. As a control see List. 39 below.

```
You are tasked with creating 20 high-quality question-answer pairs from the
    provided text to help fine-tune a language model. Your goal is to generate
    comprehensive Q/A pairs that cover the full scope of the chapter while
    emphasizing the specified key factual elements.
```

```
SOURCE MATERIAL:
Here is the text to analyze:
data_tuple[1]

KEY FACTUAL ELEMENTS (PRIORITY FOCUS):
The key factual elements that MUST be incorporated into answers whenever relevant
    are:
- Date: data_tuple[0][0]
- Location: data_tuple[0][1]
- Person/Entity: data_tuple[0][2]
- Event/Topic: data_tuple[0][3]


INSTRUCTIONS:
Generate exactly 20 question-answer pairs following these guidelines:

QUESTION REQUIREMENTS:
1. Variety: Include different question types (factual, analytical, comparative,
    causal, definitional)
2. Complexity Range: Mix simple recall questions (30%) with more complex
    analytical questions (70%)
3. Key Element Integration: At least 10 questions should directly reference the
    key factual elements above
4. Comprehensive Coverage: Questions should span the entire chapter, not just
    focus on one section
5. Natural Language: Questions should sound like they come from a human learner
    or teacher

ANSWER REQUIREMENTS:
1. SOURCE-BASED ONLY: Answers must be EXCLUSIVELY based on information found in
    the provided chapter text. DO NOT invent, assume, or add any information not
    explicitly stated in the source material.
2. FACTUAL RECALL: All answers must be direct recalls from the chapter - no
    speculation, inference beyond what's clearly stated, or external knowledge.
3. MANDATORY ENTITY INCLUSION: Every answer must incorporate at least one of the
    key factual elements (date, location, person/entity, event/topic) when
    relevant to the question.
4. VERBATIM ACCURACY: When referencing specific facts, dates, names, or details,
    use the exact information as presented in the source text.
5. NO FABRICATION: If information to answer a question is not available in the
    chapter, do not create that question.
6. Length: Aim for 2-4 sentences per answer, but prioritize accuracy over length.

QUESTION TYPE DISTRIBUTION:
- Factual Recall (10 questions): Who, what, when, where questions
- Analytical (5 questions): Why, how, explain, analyze questions
- Comparative (2 questions): Compare, contrast, similarities/differences
- Application (2 questions): What would happen if, how would this apply
- Synthesis (1 question): Summarize, conclude, overall significance

MANDATORY OUTPUT FORMAT:
You must output ONLY the Python list of dictionaries, with no additional text,
    explanations, or formatting. Do not include any introductory text like "Here
    are the question-answer pairs" or any closing remarks. Your response should
    start with [ and end with ]. Each dictionary must have 'question' and
    'correct_answer' keys.

Format exactly like this:
[{"question": "Question 1", "correct_answer": "Answer 1"}, {"question": "Question
    2", "correct_answer": "Answer 2"}, ...]

CRITICAL CONSTRAINTS:
- ONLY use information explicitly stated in the provided chapter text
- DO NOT add external knowledge or make assumptions beyond what's written
```

```
- EVERY answer must reference at least one key factual element (date:
    data_tuple[0][0], location: data_tuple[0][1], person/entity:
    data_tuple[0][2], event/topic: data_tuple[0][3]
- If you cannot answer a question based solely on the chapter content, do not
    include that question
- Answers must be factual recalls, not interpretations or extrapolations
- All 20 Q/A pairs must be created
- Key factual elements must be prominently featured
- Full chapter coverage must be achieved
- Question types must be varied appropriately
- Answers must be accurate and complete
- Natural, educational language must be used throughout

Generate the 20 Q/A pairs now in the specified Python list format.
```

Listing 39: Asker prompt template. Inputs: chapter text and factual tuple (date, location, entity, event type).

Applying the prompt in List.39, we obtain questions as illustrated in Tab. 22. Given a source event (first column), the Asker generates questions (second column) that we compare against ground truth benchmark questions (third column). To evaluate the Asker in isolation, we use Sequential-SEQ (Sec. 5): a reduced RSS configuration that disables the Recollector, Merger, and Hallucination Filter, training the model solely on Asker-generated single-event Q&As. As shown in Fig. 15, Sequential-SEQ achieves non-trivial recall despite lacking both ground-truth annotations and multi-event supervision, validating the Asker as a viable question generator.

Table 22: Comparison of question generation strategies for single-event questions (SEQ). The LLM-Generated Asker creates natural, diverse questions while Ground Truth templates follow fixed patterns.

Event	**LLM-Generated SEQ**	**Ground Truth SEQ**
At the Statue of Liberty on June 14, 2025, Brooklyn Ross captivated the audience. Amidst the iconic backdrop, she explained fabric choices with precision. The fashion show attendees hung on her every word.	Who was the central figure captivating the audience at the Statue of Liberty? When and where did Brooklyn Ross give her presentation? What type of event was taking place at the Statue of Liberty?	Provide a comprehensive account of what happened involving Brooklyn Ross and Fashion Show at Statue of Liberty on June 14, 2025. Include all relevant details about the event(s), including what occurred and any other pertinent information.

E.1.2 Recollector

The recollector of related-events retrieves prior episodes *that match the current event* from the learner's parametric memory. More precisely, the Recollector does not look up stored episodes that match the current event: it asks the learner shared-cue questions built from the current event's date, location, entities and content (e.g., "List everything you remember in this book about Mary"), and treats whatever the learner produces as related prior episodes. We evaluate three retrieval strategies (Fig. 15) which we detail hereafter. The first two are templated: *RSS-Simple* (one basic question per cue type) and *RSS-Rich* (multiple detailed templates). The third, *RSS*, used in the main text, is a fully automated variant, requiring no ground truth and employing a frontier LLM to generate diverse, natural recall queries tailored to each event.

Note that the Recollector's probes are themselves shared-cue queries (e.g., "List everything you remember about Mary"), the kind of question on which SEQ-trained models fail (Sec. 4.2). One may therefore worry that RSS is circular: it would rely on the very ability it is meant to build. It is not, because of the order in which supervision is produced. Consider the arrival of event E_k about Mary. The probe is answered by the learner $\mathcal{M}_{k-1}$, which has already been trained on every earlier event about Mary, in two forms: single-event questions from the Asker for each event, and multi-event questions from the Merger for every earlier combination of them. For example, after Mary visited Tokyo in March and London in April, the learner has been trained on "Where was Mary in April?"

and on “List all cities Mary visited” → “Tokyo, London”. When a third event arrives, the probe asks for exactly this set, and nothing beyond it. The new event is never requested from the learner; it is the input, and the Merger adds it. So the probe always asks for what the learner was explicitly supervised on at the previous steps. If the learner still answers incompletely or wrongly, the cause is one of the problems analyzed in this section, hallucination or forgetting, not a missing ability. This is why the Hallucination Filter (App. E.1.4) is the relevant control: it removes the recall errors and lets us measure the rest of the protocol on its own.

RSS-Simple RSS-Simple employs four template-based recollection queries that target different aspects of episodic memory retrieval. As shown in List. 40, these templates prompt the model to recall information based on temporal markers (‘t‘), spatial locations (‘s‘), entity identities (‘ent‘), and content types (‘c‘). Each template follows a consistent "List everything you remember" structure while systematically probing distinct dimensions of the learned episodes, enabling comprehensive memory consolidation across the narrative’s key components.

```
f"List everything you remember about t in the book."
f"List everything you remember in this book about s."
f"List everything you remember in this book about ent."
f"List everything you remember in this book about c."
```

Listing 40: Recollection question templates RSS-Simple

RSS-Rich RSS-Rich extends the basic recollection approach with twelve sophisticated template queries that probe multi-dimensional episodic associations. As shown in List. 41, these templates systematically explore cross-references between temporal, spatial, entity, and content dimensions while maintaining selective focus. Each query follows a structured pattern: given one episodic dimension (e.g., a specific date ‘t‘ or location ‘s‘), the model must retrieve information from a different dimension (locations, entities, events, or dates) without describing intermediate details.

RSS RSS replaces fixed templates with dynamic question generation using a capable frontier LLM (Claude 3.5 Sonnet) to create contextually relevant memory probes. As shown in List. 42, this approach employs a sophisticated prompt that instructs the question generator to identify key entities (people, places, events, objects, relationships) within each narrative excerpt and generate targeted recall queries. The generator creates diverse probe types including entity identification, temporal sequencing, and chronological reconstruction questions, while remaining agnostic to whether entities are appearing for the first or multiple times.

```
You are a question generator seeing a text excerpt for the first time. Your job
    is to generate questions that will help an LLM model (which MAY have seen
    previous chapters of the same document) recall potential connections and
    maintain temporal coherence.

CURRENT EXCERPT:
content_text

DOCUMENT CONTEXT: document_context

TASK: Generate memory-probing questions about entities in this excerpt. You DON'T
    know if these entities appeared before - the questions should work whether
    this is their first appearance or not.

1. **Identify Key Entities** in this excerpt, e.g.:
   - People (names, roles)
   - Places (locations, settings)
   - Events (actions, occurrences)
   - Objects (important items)
   - Relationships (between any entities)

2. **Generate Open-Ended Memory Probes to the model**, e.g.:
   - "What do you remember about [entity] from earlier in this narrative, if
    anything?"
   - "Have you encountered [entity] before in this story? If so, when and where?"
```

```
"Recall all the events that occurred on t. Without describing the events,list all
    the unique locations where these events took place."

"Consider all events that happened on t. Provide a list of all protagonists
    involved in any of these events, without describing the events themselves."

"Reflect on t. Describe all the key events that occurred on this date, focusing
    on what happened rather than who was involved or where it took place."

"Think about all events that have occurred at s. Provide a list of all dates when
    these events took place, without describing the events."

"Consider the location s. List all protagonists that have been involved in any
    events at this location, without mentioning the events themselves."

"Recall the various events that have taken place at s. Describe what happened
    during these events, focusing on the actions or occurrences rather than the
    timing or people involved."

"Reflect on all events involving ent. Provide a list of all dates when these
    events occurred, without describing the events."

"Consider all events that ent has been involved in. List all the locations where
    these events took place, without mentioning the events themselves."

"Think about ent's experiences. Describe all the key events they've been involved
    in, focusing on what happened rather than when or where it occurred."

"Recall all events related to c. Provide a list of all dates when these events
    occurred, without describing the events."

"Consider all events involving c. List all the locations where these events took
    place, without mentioning the events themselves."

"Reflect on events related to c. Provide a list of all protagonists involved in
    these events, without describing the events."
```

Listing 41: Twelve template questions for retrieval queries

```
   - "Is this your first time seeing [entity] in this narrative?"

3. **Temporal & Sequential Probes**, e.g.:
   - "If you've seen [entity] before, what has changed since then?"
   - "Where does this event fit in the sequence of events you've read about?"
   - "What events, if any, led up to this moment, based on what you remember?"

4. **Full Sequence Report Questions** (CRITICAL for temporal ordering), e.g.:
   - "Can you list ALL the places [entity] has been, in chronological order?" (if
    entity is not a place, obviously)
   - "What is the complete sequence of [entity]'s appearances so far?"
   - "What prior events happened in [entity]" (if entity is a location)
   - "Trace the timeline: What happened first, then next, leading to this point?"
   - "If you've seen these entities before, what order did you encounter them in?"
   - "Reconstruct the journey: How did [entity] get from their first appearance
    to here?"
   - "What is the chronological order of major events involving [entity]?" (let
    it be a place or a person etc..)
   - "If [location] has appeared before, what events occurred there in
    chronological order?"

5. Now that you understand the goal, adapt your questions to the CURRENT_EXCERPT
    content.
```

Table 23: Comparison of recollection strategies in Recall-based Synthetic Supervision (RSS). Each strategy queries the model's parametric memory to retrieve related episodes before learning a new event.

Current Event (t, s, ent, c)	**RSS (LLM Queries)**	**RSS-Simple (4 Templates)**	**RSS-Rich (Detailed Templates)**
At Washington Square Park on February 27, 2026, Henry Reed unveiled his groundbreaking designs. Models strutted, showcasing futuristic attire. Henry revealed future collections, leaving the audience in awe.	**1.** What do you remember about Henry Reed's background or previous activities in this narrative, if anything? ⋮ **N.** Have any specific models been mentioned earlier in the story? If so, what do you remember about them?	**1.** List everything you remember about February 27, 2026 in the book. **2.** List everything you remember in this book about Washington Square Park. **3.** List everything you remember in this book about Henry Reed. **4.** List everything you remember in this book about Fashion Show.	**1.** Recall all the events that occurred on February 27, 2026. Without describing the events, list all the unique locations where these events took place. ⋮ **12.** Think about Henry Reed's experiences. Describe all the key events they've been involved in, focusing on what happened rather than when or where it occurred.

```
6. Keep in mind that the tested model will receive a context prefix text followed
    by your question, make sure your question fits.

OUTPUT FORMAT:
{
  "questions": [
    {
      "entity": "the entity being probed",
      "entity_type": "person/place/object/event/relationship",
      "question": "the memory probe question",
      "probe_type":
    "identity/location/state/temporal/relationship/sequence_report/chronology"
    }
  ]
}

Generate as many questions as needed, but do not exceed 20, pick the most
    significant. Questions should be open-ended enough to work whether this is
    the entity's first or fifth appearance, but specific to elicit useful
    responses. Avoid redundancy.
```

Listing 42: Recollection question generation prompt (RSS)

Tab. 23 presents a qualitative comparative analysis of the different question generation approaches using a single chapter corpus.

E.1.3 Merger

Given recalled episodes (Recollector) and current-event questions (Asker), the Merger combines them into a unified and up-to-date multi-event training data. For example, if the current event is "Mary visited Tokyo in September" and the model recalls "Mary visited Tokyo in March," the Merger creates: "List all the times Mary was in Tokyo" with answer "March, September." This directly targets the binding problem by forcing the model to connect information learned at different times.

List. 43 shows the full prompt. The process works in three steps: first, filter out recall answers that indicate no prior knowledge (e.g., "not mentioned in my training"); second, find shared entities between current and recalled events; third, combine matching questions into broader ones that span multiple episodes. Questions without meaningful connections remain unchanged.

Within the RSS protocol (Sec. 5.1), the Merger is where isolated facts become connected, providing a multi-event supervision signal. Without it, training would only reinforce single-event recall, leaving the binding problem unaddressed.

```
CONTEXT INFO: context

TASK:
You are tasked with merging two sets of Q&A pairs: content-based questions and
    recollection questions. The output must preserve the total number of
    content-based questions. Follow these specific rules:

FILTERING RULES:
1. **Filter out recollection questions** where the answer indicates:
   - "This information is not present"
   - "I haven't encountered this before"
   - "No previous instances found"
   - "Not mentioned in my training"
   - Any similar negative responses indicating lack of prior knowledge

2. **Keep only recollection questions** that demonstrate actual connections to
    previous knowledge

MERGING RULES:
3. **Identify shared entities** between content and recollection questions:
   - Same places/locations
   - Same dates/time periods
   - Same entities/characters
   - Same events/actions
   - Same concepts/objects
   - Any other proven connections

4. **For content questions with shared entities in recollection questions**:
   - Only merge when there are clear, substantial connections that contains
    potential conflicts of information in the 2 questions
   - Transform them into broader questions that encompass both the content
    information AND the recollection connections
   - Create comprehensive answers that incorporate timeline and context from
    recollection data
   - Create questions that test understanding of both the current content and its
    relationship to previous knowledge
   - Be conservative in identifying shared entities - only merge when the
    connection adds meaningful context, otherwise keep the original questions as
    they are.

5. **For content questions without matching recollection connections**:
   - Keep the original content-based questions exactly as they are
   - Do not modify questions that have no recollection counterparts
   - Preserve original questions when recollection connections are minor,
    superficial, and don't generate any kind of conflict

QUALITY GUIDELINES:
- Merged questions should be clear and well-formed
- Answers should incorporate both current and recalled information when merging
- Remove redundancy while preserving important details
- Fix any grammatical errors or awkward phrasing
- Ensure the final output contains at least the same number of questions as the
    content-based dataset

CRITICAL: You MUST output at least content_count questions (the exact number of
    content-based questions), NO LESS!

INPUT DATA:format_qa_dataframe(df1, "Content-based
    questions")format_qa_dataframe(df2, "Recollection questions")

REQUIRED JSON FORMAT:
```

```
[
    {"question": "Your question here", "correct_answer": "Your answer here"},
    {"question": "Your question here", "correct_answer": "Your answer here"},
    ...
]

IMPORTANT JSON RULES:
- Start with [ and end with ]
- Each object must have exactly "question" and "correct_answer" keys
- Use double quotes for all strings
- Separate objects with commas
- No trailing comma after the last object
- Escape any quotes inside strings with backslashes

OUTPUT ONLY THE JSON ARRAY - NO OTHER TEXT WHATSOEVER.

WARNING: Outputting fewer questions than the content-based dataset is considered
    a failure!!
```

Listing 43: Q&A merging prompt template for combining content-based and recollection questions

E.1.4 Hallucination Filter: a groundtruth-based control

As introduced in Sec. 5.1, smaller models often hallucinate during recall: they return fake entities or place real entities in wrong locations, times, or activities. We want to study what happens when the model receives cleaner training data, so we introduce an optional filter that removes hallucinations using ground truth. Since this variant relies on ground truth, it serves as an upper bound rather than a directly comparable baseline.

Input and output. The filter processes each recollection Q&A pair separately. For each pair, it receives three inputs: (1) the question posed to the model, (2) the model's answer, and (3) the ground truth text containing all chapters the model has seen so far (from chapter 1 to chapter $n-1$ when processing chapter n). The filter outputs a cleaned answer with ungrounded content removed.

Procedure. For each Q&A pair, we prompt a frontier LLM to check whether entities in the model's answer actually appear in the source text with the claimed attributes. Fake entities and wrong details are removed; only verified content is kept. The filter never adds missing information. If the model's answer is incomplete, it stays incomplete. List. 44 shows the full prompt.

Evaluation. We call the filtered variant **RSS-No-Hallucination**. Comparing performance with and without filtering (Fig. 7) tells us how much binding failure comes from hallucination versus genuine inability to connect episodes.

E.1.5 Component ablation analysis

We denote each configuration as (A, R, M, H) for Asker, Recollector, Merger, and Hallucination Filter respectively, where ✓ indicates enabled and × indicates disabled. Fig. 15a shows results when the Hallucination Filter is applied to the generated data, while Fig. 15b shows results without filtering.

GT-SEQ $(\times, \times, \times, \times)$. This reference trains on the ground-truth benchmark questions of each chapter instead of RSS-generated ones. It does not appear in Fig. 7. Results are identical across both subfigures.

Sequential-SEQ $(\checkmark, \times, \times, \times)$. Using only the Asker, as in the Sequential-SEQ baseline of Sec. 5 and Fig. 7, produces reasonably high-quality data, achieving performance comparable to GT-SEQ despite the latter having access to ground-truth information. Results are identical across both subfigures, as the Hallucination Filter cannot be triggered without multi-event data.

RSS-Simple. Adding the Recollector and Merger yields significant performance improvements. In Fig. 15b, we use $(\checkmark, \checkmark, \checkmark, \times)$ since no filtering is applied. In Fig. 15a, we use $(\checkmark, \checkmark, \checkmark, \checkmark)$ where the Hallucination Filter removes confabulated details.

RSS. Enabling the full protocol achieves the best performance when hallucinations are filtered (Fig. 15a). In Fig. 15b, we use $(\checkmark, \checkmark, \checkmark, \times)$ without filtering. In Fig. 15a, we use $(\checkmark, \checkmark, \checkmark, \checkmark)$ with the Hallucination Filter enabled.

```
You are an expert fact-checker tasked with removing hallucinations from an AI
    model's answer based on provided source text.

**Your Task:**
1. Compare the AI's answer against the given source text
2. Remove any information that is NOT supported by or contradicts the source text
3. Keep only the parts that are accurate and grounded in the source material
4. If the question asks about topics not covered in the source text, return the
    fallback response: "I have no information to respond to this question."
5. Return the cleaned answer in the exact format specified below

**Critical Output Requirements:**
- Return ONLY a clean, readable paragraph or short paragraphs
- NO bullet points, NO numbered lists, NO markdown formatting
- NO asterisks (*), NO dashes (-), NO special characters for formatting
- Use plain text with proper sentences and periods
- Keep answers comprehensive but remove redundancy
- If multiple facts, separate them with periods in flowing sentences
- If the entire answer becomes invalid, return EXACTLY: "I have no information to
    respond to this question."

**Content Filtering Rules:**
- Only retain information directly verifiable from the source text
- Remove invented facts, fictional details, or unsupported claims
- Remove repetitive or redundant information
- Preserve original phrasing when possible for retained content
- Do not add new information beyond what's in the original answer
- Focus on filtering the original answer, not rewriting it completely
- If uncertain about a claim, remove it entirely

**Response Format Example:**
Clean, readable paragraph format with proper sentences and periods. Use plain
    text only.

**Source Text:**
text_up_to

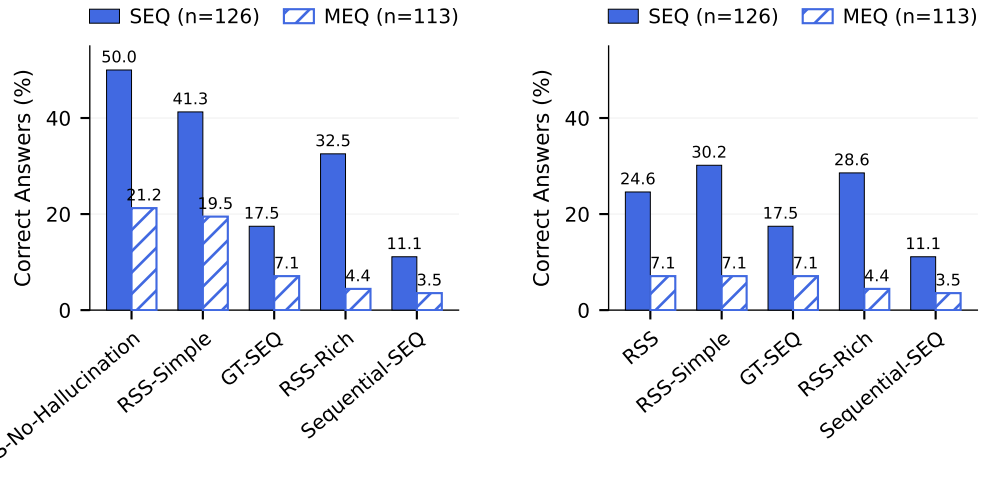
**Question Being Asked:**
question

**AI Answer to Clean:**
answer_llm

**Your Response (cleaned answer only):**
```

Listing 44: Hallucination filtering prompt template for recollection answer validation

Overall, adding the Recollector consistently improves performance over using only the Asker, demonstrating that leveraging the model's parametric memory for multi-event supervision is beneficial. Furthermore, RSS with the Hallucination Filter enabled outperforms all other configurations, confirming that removing confabulated details from the generated data leads to higher-quality supervision.



(a) Without hallucinations (filtered)

(b) With hallucinations

Figure 15: Performance comparison of RSS recollection variants. Left: with hallucination filtering. Right: without filtering.

E.1.6 Chapter-level walkthrough

We trace RSS through two chapters to illustrate the full protocol. Assume the following events arrive sequentially: (1) Mary visits Tokyo in March, (2) Mary visits London in April.

Chapter 1. The model starts with no prior knowledge. Only the Asker is active: it generates single-event Q&A pairs from the chapter text (e.g., "What city did Mary travel to?" → "Tokyo", "In which month did Mary take her trip?" → "March"). The Recollector and Merger have nothing to work with since no prior episodes exist. Training proceeds on single-event data only.

Chapter 2. The model has been fine-tuned on chapter 1. Each component now contributes.

Asker. Generates single-event Q&A pairs for the current chapter: "What was Mary's destination in April?" → "London", "When did Mary arrive in London?" → "April."

Recollector. Queries the model for related past episodes. The query depends on the variant:

- **RSS-Simple:** e.g."List everything you remember about Mary."
- **RSS-Rich:** e.g."Consider all events that Mary has been involved in. List all the locations where these events took place."
- **RSS:** e.g. "What do you remember about Mary's travels so far?"

All three aim to retrieve the Tokyo trip from chapter 1. The model may respond correctly (e.g., "Mary visited Tokyo") or hallucinate (e.g., "Mary visited Kyoto").

Hallucination Filter (optional). If enabled, the filter receives two inputs and produces a cleaned output:

- **Input 1 (Recall Q&A):** Q: "List everything you remember about Mary." A: "Mary decided to have a trip in Kyoto."
- **Input 2 (Ground truth):** Chapter 1 text: "Mary visits Tokyo in March."

- **Output:** "I have no information to respond to this question."

The filter removes the hallucinated trip (Kyoto); it never replaces it with the correct one. When the model responds correctly, the answer passes through unchanged.

Merger. Combines the Asker's single-event pairs with the Recollector's output (filtered or not). Creates multi-event Q&A: "List all cities Mary visited" → "Tokyo, London." Training now uses both single-event and multi-event pairs, providing multi-event supervision.

E.1.7 RSS-Aggregated

Does RSS-No-Hallucination generate good quality training data? And is forgetting the only reason the model fails to learn binding?

To answer these questions, we design the RSS-Aggregated experiment. We collect all multi-event Q&As produced by the Merger across all 30 events of the RSS-No-Hallucination run and train the model on them in a static multi-epoch setting. This allows us to disentangle the problem of forgetting from the quality of the generated data.

If forgetting were the only cause of binding failure, then training on all RSS-generated multi-event Q&As in a static multi-epoch setting should solve the problem. However, this does not fix binding (See Fig. 7).

The reason is that the same question receives different correct answers at different points in the narrative, creating conflicts in the training data. For example, consider the question "List all locations where Mary did activity A." When RSS generates this Q&A at chapter 2, the correct answer is "location 1." But when the same question appears at chapter 10, the correct answer becomes "location 1, location 2." These conflicting labels for identical questions confuse the model during training, preventing it from learning consistent multi-event reasoning.

E.1.8 O-LoRA experiment

One possible explanation for binding failures is that LoRA adapters interfere with previously learned weights when learning new events. To test this hypothesis, we experimented with O-LoRA, which uses orthogonality constraints to prevent such interference.

Although O-LoRA is designed for task-incremental continual learning, we include it as a baseline because its orthogonality constraints provide a relevant point of comparison for preventing interference between sequential updates. In our adaptation, each event is treated operationally as a task solely for the purpose of applying O-LoRA's update rules, with orthogonal weight updates intended to alleviate the forgetting of previously learned events. Note that this setting differs from the original formulation in two key aspects: events do not constitute distinct tasks in the traditional sense, and our per-event training samples are considerably smaller than typical task-incremental datasets.

To ensure a fair comparison, we performed a grid search over O-LoRA's hyperparameters: learning rate $\in \{10^{-5}, 5 \times 10^{-5}\}$, $\lambda_1 \in \{10^{-4}, 10^{-3}, 3 \times 10^{-3}\}$, $\lambda_2 \in \{5 \times 10^{-5}, 1.4 \times 10^{-4}, 5 \times 10^{-4}\}$, and epochs $\in \{20, 40, 60\}$. We used the implementation from Wang et al. [2023].

E.1.9 Coverage estimation

Does RSS generate enough supervision to solve binding, or is the training signal itself incomplete? To answer this, we compute a Coverage metric: the overlap between generated training questions and evaluation questions. Coverage represents an ideal upper bound, the maximum performance if the model perfectly memorized all RSS supervision without forgetting.

We compute Coverage in two steps. First, we extract all multi-event Q&A pairs from RSS and convert them to the canonical (t, s, ent, c) cue-pattern format used in our evaluation. For example, "When did Mary visit Tokyo?" with answer "March, September" becomes $(*, \text{Tokyo}, \text{Mary}, *)$. List. 45 shows the extraction prompt.

Second, we compare each evaluation question against the extracted training set. Matching requires exact correspondence for time, space, and entity fields, but allows semantic equivalence for content (e.g., "Fashion Show" matches "fashion showcase"). An evaluation question is covered if a matching cue-pattern exists in training. List. 46 shows the comparison prompt.

The gap between Coverage and actual model performance tells us how much is lost to forgetting versus how much was never covered by training. As Tab. 24 shows, Coverage exceeds 92% across all three narrative universes, confirming that RSS generates sufficient supervision.

```
You are given a natural language question-answer pair describing episodic events.

Your task is to:

1. Extract all events from the answer as tuples (t, s, ent, c) where:
   * t = time (date)
   * s = space (location)
   * ent = entity (person/agent)
   * c = content (action/activity)

2. Transform the question into one or more cue-pattern queries using the format:
   (t, s, ent, c)[all] => ground truth answer

   Use `*` for wildcards (unknown/queried fields).
   Use the actual value when the field is specified in the cue.

3. Output format - For each valid cue pattern derivable from the question:
   (*, location, *, *)[all] => "event1; event2; ..."
   (*, *, entity, content)[all] => "time, location"

Example input:
"When and where did Zoe Brown discuss theater traditions, and how does this event
    relate to other activities at the American Museum of Natural History?"
-> [answer describing 6 events]

Example output:
(*, *, Zoe Brown, theater traditions)[all]
  => "May 11, 2026, American Museum of Natural History"

(*, American Museum of Natural History, *, *)[all]
  => "tech hackathon on March 23, 2024; fashion show on June 14, 2025;
      karaoke night on February 27, 2026; Zoe Brown discussing theater
      traditions on May 11, 2026; theater performance on September 22,
      2026; mime act on November 13, 2026"

I will provide a dataset composed of a set of q/a. You will return a JSON file
    containing rows following this cue-pattern format.
```

Listing 45: Coverage shape extraction prompt

```
You are tasked with comparing two JSON files: a **training set** and a **test
    set**. Each file contains rows representing tuples in the format:
(time, space, entity, content)

For example:
- "(September 03, 2026, Metropolitan Museum of Art, Henry Reed, educational
    workshop leader)[all]"
- "(*, {Metropolitan Museum of Art}, *, *)"

Where `*` represents a wildcard (any value).

### YOUR TASK:
For each row in the **test set**, determine if it is contained in the **training
    set** and output `true` or `false`.

### MATCHING RULES:

**Positions 1, 2, 3 (Time, Space, Entity) - STRICT MATCHING:**
- Normalization first: ignore case differences and remove: ", ', {, }
- If **test** has `*` -> **training** must also have `*` (wildcard matches
    wildcard exactly)
- If **test** has specific value -> **training** can have `*` OR same value
    (after normalization)
```

```
**Position 4 (Content/Action) - SEMANTIC MATCHING:**
- Same wildcard rule: if **test** has `*` -> **training** must have `*`
- If **test** has specific value -> **training** can have `*` OR semantically
    equivalent value
- Semantic equivalences include synonyms, paraphrases, related concepts:
  - "Astronomy Night" ~ "dark sky preserve talks" ~ "astronomy event"
  - "Educational Workshop" ~ "workshop" ~ "peer learning workshop"
  - "Fashion Show" ~ "fashion showcase" ~ "fashion event"
  - "Tech Hackathon" ~ "hackathon" ~ "coding activity"
  - "Theater Performance" ~ "method acting" ~ "mime act"

### MATCHING LOGIC SUMMARY:
| Test Value     | Training Value      | Match? |
|----------------|---------------------|--------|
| *              | *                   | YES    |
| *              | specific value      | NO     |
| specific value | *                   | NO     |
| specific value | same/semantic equiv | YES    |
| specific value | different value     | NO     |

For a test row to match, ALL 4 positions must satisfy the rules.

### OUTPUT FORMAT:
Return a JSON array where each element corresponds to a test row:
[
  {
    "test_row": "(original test row string)",
    "match": true,
    "matched_with": "(matching training row or null)"
  },
  ...
]

### INPUT FILES:
**TRAINING SET:** [...]
**TEST SET:** [...]

Process the test set and return the JSON output.
```

Listing 46: Coverage comparison prompt

Table 24: MEQ tuple containment in training set

	Default	Sci-Fi	News
Match (%)	92.04	96.46	98.23
Total MEQs	113	113	113

F Ways in which generative AI was used

We used LLMs to assist with code development, text clarity, and identifying related work. Additionally, LLM-based evaluation (LLM-as-judge) was used to verify alignment with ground truth knowledge. All generated code and outputs were reviewed and validated by the authors, who take full responsibility for all content.

G Benchmark usage and reproduction guidelines

Here we provide a high-level explanation in pseudocode. The code repository contains the full pipeline and the scores used for each figure.

G.1 High-level pipeline architecture and data flow

The benchmark generation pipeline follows a deterministic, multi-stage process designed to create reproducible narrative datasets with controlled complexity. The pipeline consists of three main stages:

Step 1: Event Generation. Given fixed parameters (universe, style, number of events n, seed), the system first generates atomic narrative events. Each event is sampled from a predefined universe containing temporal markers, entities, spatial locations, and content elements. Events are distributed across the narrative according to a geometric distribution (default parameter for this work $p = 0.3$), which controls how often dates, entities, locations and contents recur across events.

Step 2: Narrative Synthesis. Events are transformed into coherent narrative chapters through iterative generation using a large language model (Claude 3.5 Sonnet for the Llama, GPT and RSS experiments; Claude Sonnet 4.6 for the Qwen, Mistral and transfer-learning experiments, after 3.5 was deprecated). The system performs up to 10 generation attempts per event, implementing verification checks to ensure narrative consistency and proper event grounding. Successfully generated paragraphs are indexed according to the specified book parameters. Valid narrative segments are assembled into a complete book structure. The system builds a ground truth dataframe (`df_book_groundtruth`) that maintains mappings between chapters, original events, and their constituent elements (entities, temporal/spatial markers, content).

Step 3: Question Generation. The pipeline automatically generates two types of evaluation questions from the narrative: (i) direct questions testing recall of individual facts within single chapters (SEQ), and (ii) binding questions requiring integration of information across multiple chapters sharing common elements (MEQ). Questions are stratified by complexity (number of chapters involved) and filtered to ensure non-empty answer sets. The system tracks question "widespreadness" to ensure adequate coverage across the narrative timeline. The final stage splits questions into training and evaluation sets, with special handling for single-chapter questions used in fine-tuning. Questions templates are replaced with actual chapter content, and common-sense evaluation data is loaded for baseline comparisons. All artifacts (book, ground truth, questions) are persisted with verification checks to ensure reproducibility across runs.

The complete pipeline maintains deterministic behavior through fixed random seeds and verification checks at each stage, enabling exact reproduction of benchmarks given identical input parameters.

G.2 Baseline implementation details for SEQ, SEQ+MEQ

All baseline methods share the same fine-tuning data preparation pipeline, differing only in their question selection and generation strategies:

Static training on SEQ. This baseline uses only single-chapter questions for fine-tuning. The implementation filters the question dataset to include exclusively questions where `n_chapters_correct_answer == 1`, ensuring that each training example requires information from exactly one narrative chapter. Training data is constructed by pairing each question with its corresponding answer set, formatted as JSON objects with system prompts, user queries, and assistant responses. The final training set is randomly shuffled to prevent learning order biases.

Static training on SEQ+MEQ. This variant extends SEQ by including all available questions in the training set, regardless of the number of chapters involved in the answer. The implementation sets `answer_in_one_chapter_only=False`, incorporating both single-chapter questions and multi-chapter binding questions that require integrating information across episodes sharing common elements. This provides explicit supervision for learning cross-episode relationships during fine-tuning.

Sequential Training. This baseline implements continual learning by training on individual narrative chapters sequentially, one at a time, in their temporal order. During training, the model is fine-tuned on the content of each chapter independently. After processing all chapters in sequence, the model is evaluated on questions whose answers require information distributed across multiple chapters from the learned sequence. This approach tests whether the model can spontaneously bind related episodes through their shared elements, representing a pure continual learning scenario where cross-episode integration emerges only at test time.

All methods utilize the same JSON formatting pipeline, where questions are paired with their ground truth answer sets and wrapped with consistent system prompts instructing the model to provide accurate, complete responses based on the narrative context. The training data undergoes random shuffling before fine-tuning to ensure robust learning across all episodic positions.

G.3 Step-by-step evaluation protocols

To ensure fair and objective comparison across all training approaches, we employ a standardized evaluation protocol with consistent inference and assessment procedures:

Step 1: Model Loading. For each training method, we load the fine-tuned model checkpoint. The system supports both LoRA adapters and fully fine-tuned models, automatically detecting the fine-tuning type and loading from the appropriate checkpoint path. All models use the same base architecture (e.g., Llama-3-8B) with identical tokenizer configurations and maximum sequence length of 8192 tokens.

Step 2: Inference Generation. Each model generates answers for all test questions using identical inference parameters: temperature = 0.1 for controlled generation, consistent system prompts providing task instructions, and uniform maximum token limits for responses. The inference pipeline processes questions sequentially, storing both generated answers and any reasoning traces for subsequent analysis.

Step 3: LLM-as-Judge Evaluation. To assess answer quality beyond exact match metrics, we employ a consistent LLM judge (detailed of the prompt in Appendix C.2.2). The judge model evaluates all generated responses using the same prompt template and temperature setting (0.1), ensuring objective comparison across methods C.2.3. This approach evaluates semantic correctness, completeness, and relevance of generated answers against ground truth, providing robust assessment even when models produce valid paraphrases or alternative phrasings.

Step 4: Metrics Computation. For each model, we compute LLM-judged correctness, stratified by question type (single vs. multi-chapter) and complexity. Results are aggregated across multiple random seeds to ensure statistical significance of performance differences.

Note on train/test overlap. A natural question is whether the model has already seen the test questions during training. The answer depends on the setting. In our main results (Train(SEQ), Sec. 4.1–4.2), it has not. We train only on questions with exactly one correct chapter, so the model never sees a multi-event question during training. The multi-event accuracy we report is therefore measured on unseen questions. Overlap is possible in only two settings, and in both it is intended. In Train(SEQ+MEQ) (App. D.3), we train on multi-event questions on purpose, to check that the model has enough capacity to learn them. In RSS (Sec. 5), the Merger writes its own multi-event training questions from what the model recalls. These can be similar to the test questions, but they are never the same, since they are worded differently. For RSS, we measure this overlap directly with Coverage (App. E.1.9): the fraction of test questions that RSS also generated for training. We use Coverage only as an upper bound (what a model that remembers all its RSS training data could answer), and never as a test accuracy.

This evaluation framework ensures all models are evaluated with the same questions, prompts and judge.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the contributions (in-vivo benchmark, in-vitro controlled study, RSS pipeline) and the scope of the binding failure phenomenon, matching the experimental sections.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The conclusion explicitly discusses limitations, including the synthetic nature of the in-vitro narratives and the unresolved obstacles (hallucination, forgetting, evolving supervision) that prevent RSS from translating coverage into accuracy.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper is empirical. Section 2 formalizes the problem but does not contain theorems requiring proofs.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 3 details the verification protocol, prompting strategies, and judge model. Section 4 specifies models, narrative generation, training protocol, and evaluation. Hyperparameters and additional details are in the appendix, and code is released.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code, data, and the synthetic narrative pipeline are released through the public repository cited in the introduction Mantovani et al. [2026]. Moreover App.G provide a high-level explanation in pseudocode of the experiment pipeline.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Training and evaluation details are described in Section 4, including model variants, LoRA vs. full fine-tuning (LoRA configurations), narrative scales, and grid-searched hyperparameters reported in App. D.1. SEQ/MEQ splits are reported in Tab. 2.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Fig. 3 reports standard deviations across 3 runs and 3 narrative universes. Fig. 7 shows per-universe variability.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Compute details for the main fine-tuning experiments (GPU type, training duration) are reported in the introduction and in the appendix. Frontier model experiments rely on commercial APIs and the cost is bounded by the prompt counts described in Section 3.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses public datasets and synthetic narratives, involves no human subjects, and complies with the NeurIPS Code of Ethics.

10. **Broader impacts**

    Answer: [N/A]

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Justification: The paper studies a failure mode of LLM parametric memory using synthetic and public benchmarks. We see no direct path to societal harm beyond what is already inherent to general LLM research.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We release synthetic narratives and evaluation prompts. The released artifacts pose no foreseeable misuse risk.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: TempLAMA [Dhingra et al., 2022], the episodic benchmark of Huet et al. [2025], CommonsenseQA [Talmor et al., 2019], and the open-weight models (Llama, Qwen, Mistral) are cited; frontier APIs (Claude, GPT, Gemini, DeepSeek, Grok, Mistral) are used per their terms of service.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The frontier model benchmark and the synthetic episodic narrative pipeline are released with documentation describing data format, generation procedure, and evaluation scripts Mantovani et al. [2026].

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects. Human annotations used to validate the LLM judge were performed by the authors.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human subjects research was conducted.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

    Answer: [Yes]

    Justification: LLMs are an integral part of the methodology. Frontier models are evaluated as subjects of the in-vivo benchmark, an LLM judge is used to score open-ended answers, and the RSS pipeline (Section 5) uses a teacher LLM for question generation and the learner model itself for memory probing. All such uses are described in the corresponding sections.