

Unable to Forget: Proactive Interference Reveals Working Memory Limits in LLMs Beyond Context Length

Chupei Wang*
University of Virginia
Charlottesville, VA, USA
cw4bb@virginia.edu

Jiaqiu Vince Sun*
Center for Neural Science
New York University, New York, NY, USA
vince.sun@nyu.edu

Abstract

Large language models are often assumed to benefit from longer context windows, yet retrieval from context requires not only locating the target information but also suppressing competing information tied to the same cue. Inspired by the proactive interference (PI) paradigm in cognitive science, we introduce PI-LLM, an evaluation in which interleaved key-value updates are streamed and models are queried to retrieve only the final value of each key, therefore isolating interference rather than search as the primary variable. Across 35+ models ranging from 0.6B open-weight to frontier-scale proprietary systems, retrieval accuracy declines approximately log-linearly as repeated same-key updates accumulate, with errors dominated by outdated values. Similar log-linear degradation patterns emerge across multiple independent load dimensions, and persist when total input length is held constant, pointing to a working-memory-like bottleneck in maintaining and retrieving the current binding under interference; resistance scales with effective parameter size rather than context window length. This degradation is not alleviated by explicit ‘forget’ instructions or chain-of-thought reasoning, and shows no sign of leveling off within the tested range, in contrast to the plateau observed in human PI studies. Together, these results suggest that LLMs’ working-memory-like capacity, though large, is governed by less flexible control than the executive processes that support human resilience to interference.

1 Introduction

Large language models (LLMs) are increasingly deployed in settings that require retrieving and updating information from long contexts—summarizing evolving documents, tracking entity states across dialogues, or processing streamed sensor logs. Current research indicates that LLMs generally struggle with retrieval tasks when closely related pieces of information are present (Vodrahalli et al., 2024), and even advanced reasoning models do not effectively overcome these retrieval challenges (OpenAI, 2025b). However, most studies—having already labeled retrieval as a “long-context” challenge—prioritize input length as the primary determinant of retrieval difficulty, relegating other factors to a secondary role.

This framing conflates two distinct challenges in retrieval: *search*—locating a target “needle” in a vast contextual haystack—and *interference*—correctly identifying that needle when other items sharing the same cue or similar features compete for retrieval. Recent long-context benchmarks, such as Needle-in-a-Haystack, DeepMind’s Michelangelo (Vodrahalli et al., 2024), and OpenAI’s MRCR (OpenAI, 2025c), primarily raise task difficulty by lengthening the prompt. Although these studies acknowledge that interfering information affects retrieval, interference as an independent variable has been largely overlooked.

Our work draws on the proactive interference (PI) paradigm from cognitive science, where earlier associations for a repeated cue disrupt recall of the most recent one. We introduce

*Equal contribution. Listing order is random.

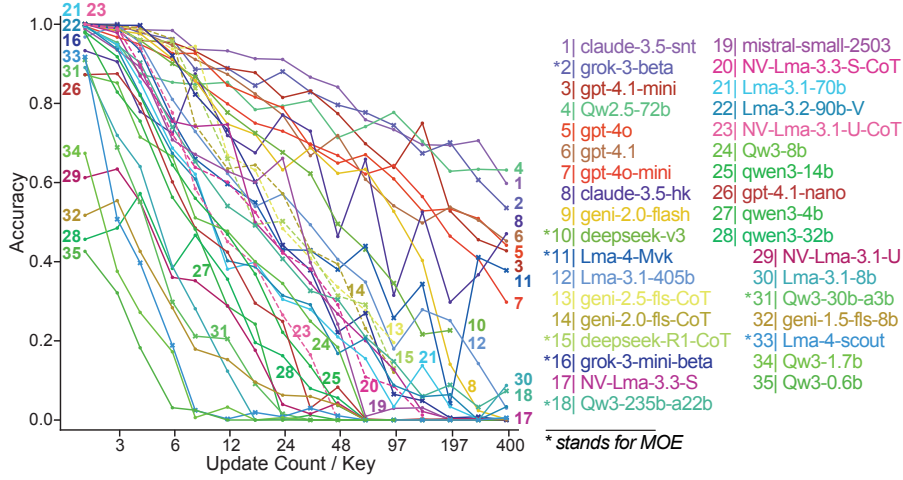


Figure 1: Consistent *log-linear* decline in retrieval performance due to interference. As same-key updates accumulate, accuracy declines log-linearly across diverse models. The retrieval target is always the most recent update for each key, minimizing search difficulty and isolating interference effects. (Asterisk: MoE models.)

PI-LLM, an evaluation in which interleaved key-value updates are streamed and the model is queried for only the final value of each key. Because the retrieval target is always the last-presented value for each key, search difficulty is minimized and interference is isolated as the primary variable. Consider, for example, a sequence of blood-pressure (BP) readings: BP: 120 (triage) → BP: 128 (10 min) → BP: 125 (discharge). The correct answer is “BP: 125,” but prior same-key values act as distractors.

Across 35+ models, from 0.6B open-weight models to frontier proprietary systems, retrieval accuracy declines approximately log-linearly as same-key updates accumulate (Figure 1), with errors dominated by outdated values. Crucially, this degradation persists even when total input length is held constant, implicating interference as an independent factor. Within the range of update counts we test, the decline shows no plateau, in contrast to the above-chance stabilization observed in human PI studies (Oberauer & Vockenberg, 2009; Hartshorne, 2008), and remains robust across the mitigation strategies we test. These findings point to limited flexible control over competing in-context bindings. We use PI and working memory throughout as a functional framework, not as a claim of mechanistic equivalence with humans.

Our contributions are as follows:

- We introduce PI-LLM, the first evaluation that isolates interference from search in LLM retrieval, revealing a consistent log-linear decline in accuracy across all tested models as same-key updates accumulate.
- We demonstrate that this interference bottleneck is independent of input length and that a similar log-linear degradation pattern emerges across multiple load dimensions (number of updates, number of tracked keys, value token length).
- We show that no test-time mitigation we tested—neither editing the input (e.g., direct forgetting instructions) nor scaling inference-time reasoning (e.g., chain-of-thought)—eliminates the decline, and that the failure is not one of reasoning: models correctly localize the most recent update yet still report an outdated value.
- We provide a focused naturalistic validation via a Product Launch State Tracking task on five models across three families, demonstrating that the interference effect extends beyond synthetic settings.

2 Related work

Long-context retrieval and in-context interference. The Needle-in-a-Haystack paradigm (Liu et al., 2024) and its successors—Michelangelo (Vodrahalli et al., 2024) and MRCR (OpenAI, 2025c)—have become standard long-context evaluations, with MRCR now adopted as a benchmark for frontier models such as Gemini 3 Pro and GPT-5 (Google DeepMind, 2025; OpenAI, 2025a). These benchmarks manipulate context length and acknowledge that co-referenced distractors hurt performance, but they do not isolate interference as an independent variable. Complementary to our findings, Du et al. (2025) show that input length alone degrades LLM performance even when retrieval is perfect and distractors are minimized, pointing to a length-related performance degradation distinct from interference. Our work isolates the converse: interference degrades retrieval even when input length is held constant. PI-LLM can be viewed as a minimal abstraction of MRCR that strips away the haystack/search component and directly controls the number of same-key rebindings. Work on entity binding in transformers (Dai et al., 2024; Feng & Steinhardt, 2024; Gur-Arieh et al., 2025) and the role of induction heads (Anthropic, 2022) provides theoretical context for why same-key interference is mechanistically non-trivial: when a model encodes bindings such as “A is associated with B,” its internal representations occupy structured subspaces from which both identity and bound value are decodable, and overwriting these bindings may be inherently difficult for attention-based architectures.

State tracking and entity updating. Tracking the current value of a repeatedly updated key is a state-tracking problem, and PI-LLM is that problem in its simplest form: key–value binding with a trivial update rule (newest overwrites old), leaving same-key interference as the only variable. Prior state-tracking work asks whether models can compute an evolving state under semantic or symbolic transformations—entity and procedural tracking (Kim & Schuster, 2023), algorithmic state tracking (Li et al., 2025)—whereas we strip the transformation away and scale up the interference. The binding literature (Feng & Steinhardt, 2024; Dai et al., 2024) binds each key to a single object and asks whether it is read out correctly; our distinct axis is binding one key to a long *succession* of values, the repeated-rebinding regime that produces proactive interference.

Proactive interference and working memory. Proactive interference (PI) refers to the phenomenon whereby previously learned associations impair recall of more recent ones. In the classic A-B, A-C paired-associate paradigm, learning a new association (A-C) is hindered by a prior association (A-B) to the same cue (Oberauer & Vockenberg, 2009). Human resilience to PI is attributed to executive control mechanisms—particularly gating (automatically suppressing outdated bindings during encoding) and directed forgetting (intentionally discarding specific associations upon instruction) (Festini & Reuter-Lorenz, 2014). In human studies, resistance to PI is strongly correlated with working memory capacity: individuals with greater working memory show less impairment from prior associations (Ackerman et al., 2005). Importantly, human recall under PI typically plateaus: beyond a certain interference load, further accumulation produces minimal additional impairment (Oberauer & Vockenberg, 2009; Hartshorne, 2008). Recent studies have applied human working-memory tests such as the N-back paradigm to LLMs (Gong et al., 2024). N-back probes a related but distinct ability: the target is the item n positions back, so tracking serial position is itself part of the task. PI-LLM always queries the most recent binding, which makes position tracking trivial and leaves interference as the sole source of difficulty.

3 PI-LLM: evaluation design

3.1 Task structure

Each input sequence in PI-LLM consists of three parts (Figure 2): (1) an *instruction* specifying the task and which keys to track; (2) an *update stream* of key–value pairs, where a fixed set of keys each receive an equal number of updates with the constraint that the same key does

not appear in two consecutive pairs; and (3) a *query* directing the model to retrieve the final value for each tracked key.

1) Instruction	2) Update Stream	3) Query
As my secretary, I need you to carefully read a text stream where the values of multiple keys are being continuously updated. The 3 keys to track include <u>visual art</u> , <u>tools</u> , <u>landform</u> . I will ask you to identify the current value of each key later. The text stream starts on the next line.	<u>visual art</u> : Braque; <u>tools</u> : hook remover; <u>visual art</u> : Pollock; <u>landform</u> : moraine; <u>visual art</u> : Basquiat; <u>tools</u> : plunger; <u>landform</u> : plain; <u>tools</u> : ruler; <u>visual art</u> : <u>seascape</u> ; <u>landform</u> : valley; <u>tools</u> : <u>hammer</u> ; <u>landform</u> : <u>dune</u> .	What are the current value of each key (<u>visual art</u> , <u>tools</u> , <u>landform</u>) you are tracking? End your response with: 'The current value of <key> is <value>.'

Figure 2: Basic input example for PI-LLM. Three keys (“visual art,” “tools,” “landform”)—underlined in distinct colors for clarity—each undergo four same-key updates. In actual experiments, up to 46 keys are used with up to 400 updates each. The model is instructed to retrieve the final value for each tracked key (bold). Keys to check are cued both before and after the update stream.

For each key with update count X (the number of same-key updates that key receives), the preceding $X-1$ key-value pairs share the same key but carry different values, serving as interfering same-key updates. Because the retrieval target is always the last occurrence of each key, search difficulty is low by design—the main challenge is identifying the correct value in the presence of multiple earlier, competing updates for the same key. This mirrors realistic data environments where many variables are updated concurrently and interference—rather than search—becomes the limiting factor.

This design contrasts with Lost-in-the-Middle (Liu et al., 2024), which varies the *position* of a single target within the context. PI-LLM instead holds the target’s relative position constant (always the most recent update) and varies the *amount of same-key interference* preceding it.

3.2 Evaluation protocol

In our primary experiment, we used 46 unique keys, each receiving a variable number of same-key updates (from 3 to 400, log-spaced). To maintain comparability with human performance, we constructed a word dictionary with up to 46 categories, each comprising 400 words with similar token lengths within each category. Keys were drawn from category names and values were randomly selected from the corresponding categories. Words were randomly sampled in each test run to eliminate potential effects of specific semantic combinations. Accuracy is defined as the proportion of correctly retrieved final values across all tracked keys. We report 95% confidence intervals computed via nonparametric bootstrapping over repeated sessions; decoding settings, sessions per condition, and the bootstrap procedure are given in Appendix B.3, and response-parsing rules in Appendix B.4.

3.3 Models

We evaluated a broad spectrum of 35+ state-of-the-art open-source and proprietary LLMs, ranging from 0.6B parameters (Qwen3-0.6B) to frontier-scale systems (DeepSeek-V3, GPT-5), including major proprietary models from OpenAI, Anthropic, Google, xAI, and Meta. Our benchmark covers both dense and Mixture-of-Experts (MoE) architectures, as well as both reasoning-enabled and standard inference modes. A complete list of model versions is provided in Table 1 (Appendix B.5).

4 Consistent log-linear decline under interference

4.1 Main results

In our primary experiment we varied the update count per key from 3 to 400 with the number of unique keys fixed at 46, so that higher counts correspond to greater interference load. Across models of varying parameter sizes, we observe a robust log-linear decline in retrieval accuracy as same-key updates accumulate (Figure 1; for a detailed breakdown

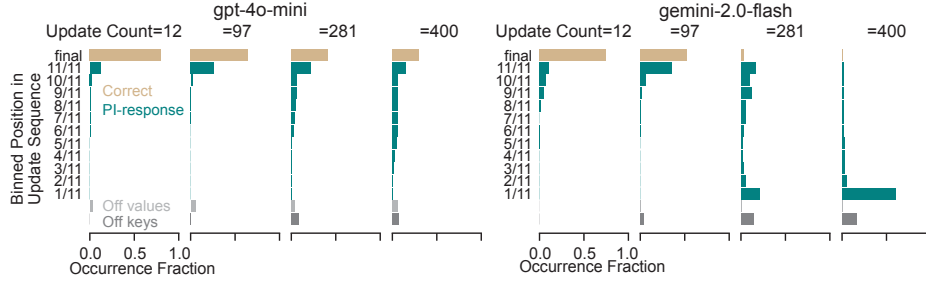


Figure 3: Distribution of model responses across update positions as interference increases (left to right), shown for gpt-4o-mini (left) and gemini-2.0-flash (right). Green bars (Bins 1–11) cover the update sequence; earthy yellow indicates the correct final update. Light gray (“off values”) denotes hallucinations; dark gray (“off keys”) denotes failures to return any value. As update count grows, errors shift from clustering near the final update to earlier bins. For additional models, see Figure 7 in Appendix C.

by model size tier, see Figure 6 in Appendix C). This log-linear trend reflects rapid initial accuracy loss, with each subsequent doubling of interference producing a smaller absolute but proportionally consistent additional decline. Larger models exhibit a more gradual slope than smaller ones, but no model is immune: even frontier-scale models degrade substantially at high update counts. Fitting accuracy against $\log_2(\text{update count})$ per model (Table 2 in Appendix C), log-linear gives the best single-form description overall (28 models, median $R^2 = 0.95$; quadratic term non-significant for 15 of 24). We keep “approximately log-linear” as a descriptive summary rather than a claim about the underlying form, since a subset of models is systematically curved.

4.2 Error analysis: three stages of interference

Analysis of error distributions reveals a systematic three-stage progression as interference increases (Figure 3):

Stage 1—Low interference, localized errors. When interference is low, retrieval accuracy is high and the few errors that occur are concentrated on updates immediately preceding the correct final value (“recency errors”). The model’s confusion is narrowly constrained and spatially localized.

Stage 2—Moderate interference, dispersed errors. As interference accumulates, accuracy drops and errors shift further back in the update history. The error distribution broadens, with the model increasingly favoring earlier updates, marking a drift from recency toward primacy. A small but growing fraction of errors involve hallucinated values never presented in the input.

Stage 3—High interference, primacy-dominated errors. At high interference levels, retrieval accuracy remains near zero. However, the error distribution continues to evolve: errors migrate further toward the earliest updates, with primacy effects becoming dominant. Hallucinated values—outputs never present in the input—increase modestly, but the majority of errors still reflect previously presented values, consistent with the signature of proactive interference rather than random guessing.

This three-stage progression aligns with a “limited resource” interpretation: interference progressively depletes a finite capacity, causing the model’s responses to drift further from the correct values even after accuracy has collapsed.

4.3 Interference Endurance Score

To quantify each model’s resistance to interference, we define the Interference Endurance Score (IES) as the area under the curve (AUC) of retrieval accuracy across log-scaled update counts (Figure 8 in Appendix C). A higher IES indicates greater resistance to interference.

Regression analysis shows that model parameter size is a significant predictor of IES ($t = 3.02$, $p = 0.006$, $N = 30$), while context window length has no significant effect ($t = -0.144$, $p = 0.887$; Table 3). Among models with similar context lengths (128k–131k tokens), the Spearman correlation between parameter size and IES remains strong ($\rho = 0.673$, $p = 0.0016$; Figure 9).

Model size—not context window length—is the predictor of resistance to interference. This regression is descriptive: size is entangled with family, training data, tokenizer, and inference settings. As partial controls, the OpenAI and qwen3 ladders hold family fixed and still show a monotonic size effect; a causal test would require controlled training at varied scale. Furthermore, MoE architectures match or decline faster than dense models of equal or smaller total size (Figure 10), suggesting that the effective parameter size—the parameters active per token, not the total count—is what supports resistance to interference.

4.4 Robustness

Although absolute retrieval accuracy can shift with changes in prompt wording (He et al., 2024), our study focuses on the relative trend of performance decline rather than raw accuracy, which neutralizes variability from individual prompt formulations. We validated the robustness of the observed log-linear decline across multiple dimensions of experimental variation, including alternative prompt templates, different update formats (e.g., natural-language statements such as “The value of X is A” versus symbolic notation such as “X = A”; the primary experiments use colon-separated pairs), and varied instruction styles. Across all variations, the qualitative log-linear trend persisted. It is likewise insensitive to decoding: re-running gpt-4.1-nano and qwen3-14b across temperatures $\{0, 0.3, 0.7, 1.0\}$ leaves accuracy and slope essentially unchanged (Figure 11). Nor is it a parsing artifact: errors are overwhelmingly well-formed values presented earlier for the same key, while not-in-stream and unparseable outputs stay a small minority throughout (pooled mean $\leq 9\%$; see Appendix B.4 for parser rules).

We additionally tested sequential update ordering (same-key updates grouped contiguously rather than interleaved; see Figure 12 in Appendix C). In this mode, accuracy remains near-perfect until a model-specific threshold, after which it drops abruptly to near-zero—a step-function pattern that contrasts with the gradual log-linear decay under random interleaving. Within the same model family, larger models exhibit higher thresholds. The consistent interference-induced decline across both input organizations underscores the robustness and generalizability of the PI phenomenon.

5 Interference load, not context length, drives the decline

The log-linear decline in the previous experiment might be attributable to increasing context length rather than interference per se, since both co-vary when update count grows. To dissociate these factors, we designed two additional experimental settings (Section 5.1), one of which varies interference load through the instruction alone and therefore holds the input sequence itself fixed. We then test a further load axis—value token length—which raises load without adding any bindings to resolve (Section 5.2).

5.1 Tracking load at fixed input length

5.1.1 Experimental setup

1. **Setting A—Number of Updated Keys (N_U):** We fixed the update count per key (either 125 or 350) and varied the number of distinct keys updated in the sequence (N_U , from 2 to 46). The model was instructed to track all updated keys. Here, input length grows with N_U .
2. **Setting B—Number of Tracked Keys (N_T) at Fixed Input Length:** Both the update count per key and the number of updated keys ($N_U = 46$) were fixed, so each input sequence contained the same total number of key–value pairs. We varied only the number of keys the model was instructed to track and retrieve (N_T , from 1 to

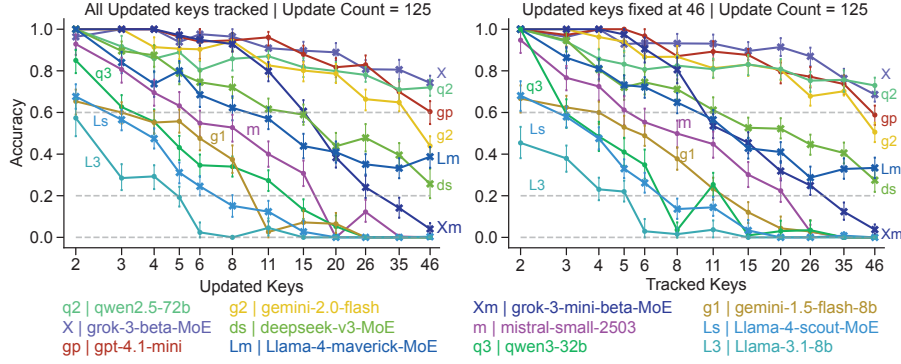


Figure 4: Varying the number of updated keys (left, Setting A) versus tracked keys at fixed input length (right, Setting B) yields nearly identical log-linear declines. Update count is fixed at 125 per key. MoE models: “X” markers. Error bars: bootstrapped 95% CIs. Results with update count = 350 show the same pattern and are provided in Figure 14 in Appendix C.

46). Here, input length is held constant (see Figure 13 in Appendix C for an input example).

If both settings produce similar declines, the performance drop cannot be attributed to input length alone and must instead reflect tracking load.

5.1.2 Results

Settings A and B exhibit nearly identical log-linear declines in retrieval accuracy (Figure 4). Notably, Setting B holds input length constant while Setting A allows it to grow, yet the decline patterns are virtually indistinguishable. This demonstrates that the decline tracks the number of bindings to be resolved rather than the amount of text in context, identifying anti-interference capacity as a distinct resource. A separate line of evidence points the same way at the model level: regression analysis (Section 4) shows that parameter size significantly predicts interference resistance while context window length does not. Neither more text nor a larger window is what governs interference management.

Comparing the two axes at matched total volume—deep-narrow (2–8 keys $\times$ 125 updates) against shallow-wide (46 keys $\times$ 6–24 updates; Table 4, Appendix C)—deep-narrow is more accurate in 34 of 36 matched pairs (median +0.145), suggesting keys are the more damaging axis, as expected since they enlarge the set of bindings to resolve at query time.

5.2 Item length as a further load axis

Sections 4 and 5.1 established two dimensions of interference load—update count and number of tracked keys—and showed that the decline is not explained by input length. We now test a third dimension: the token length of each value. In human memory research, increasing the length of words to be remembered impairs performance—the classic word-length effect (Baddeley et al., 1975). To test whether LLMs exhibit a similar sensitivity, we held constant the number of updates, updated keys, and tracked keys, and systematically varied value length by concatenating multiple dictionary words (e.g., **AppleOrangeBanana**; see Figure 15 in Appendix C for an input example).

LLMs exhibit a consistent, approximately log-linear decline in retrieval accuracy as value token length increases (Figure 16 in Appendix C), with a steeper slope than for other interference dimensions: increasing value length from one to ten words drives accuracy below 40% for every model, and extending it to forty words reduces accuracy to under 5%. This demonstrates that item length constitutes an independent dimension of interference load, paralleling the word-length effect in human working memory.

6 PI persists in naturalistic settings

A natural concern is whether the interference effects observed in synthetic key-value tracking generalize to more realistic settings. To address this, we designed a Product Launch State Tracking task, in which key-value updates are embedded within a naturalistic multi-turn team chat history simulating a Slack export (see Figure 17 in Appendix C for an input example). The model must track state changes of four specific entities (Budget, Song, Place, Date) amidst conversational noise and retrieve the final agreed-upon values.

We evaluate five models across three families (gpt-4.1, gpt-4.1-mini, gpt-4.1-nano, claude-haiku-4-5, qwen3-14b), with $N = 30$ sessions per model and update count. All five exhibit an approximately log-linear decline as the number of updates grows from 4 to 256 (Figure 18 in Appendix C), with accuracy falling from 95–100% to 34–88%; within the OpenAI family the slope flattens from nano to mini to full, mirroring the size-IES relation in Section 4. The error distribution at moderate update counts shows that errors are concentrated on recent but outdated updates—characteristic of proactive interference—rather than random values from the distant past.

This remains a focused naturalistic stress test rather than a broad ecological-validity benchmark: it covers five models but a single domain, and extending it to further domains is a natural next step. Nevertheless, the convergence between synthetic and naturalistic results provides evidence that the interference bottleneck identified by PI-LLM reflects a genuine constraint on LLM retrieval rather than an artifact of the synthetic setup.

7 Interventions reshape but do not eliminate interference

Evidence from cognitive science indicates that humans actively unbind prior associations before encoding new information (Oberauer & Vockenberg, 2009) and benefit from explicit instructions to deprioritize outdated items (Festini & Reuter-Lorenz, 2014). We tested whether analogous strategies could help LLMs manage interference. We evaluate each against two criteria that come apart. (1) Does it raise overall accuracy? Interventions differ here: forgetting and focusing instructions help some models and hurt others, while session-boundary framing helps more consistently. (2) Does it remove the decline as update count grows? None does; “does not eliminate” below refers to this criterion.

7.1 Natural language forgetting and focusing prompts

We inserted targeted prompts into the update stream instructing models to disregard prior updates for specific keys (per-key forget), to concentrate on upcoming information (forward focus), to first locate the target before extracting it (locate-then-retrieve), or to treat a verbal cue as marking a new session (soft session reset). The effect of these prompts varies in sign across models (Figure 5, left): the per-key forget prompt lifts mistral-small-2503 from near-floor yet costs gpt-4.1-mini about half its accuracy, and the soft session reset helps three of four models modestly. None helps consistently, and all converge to the floor by 100–150 updates.

Error analysis reveals a telling failure mode (Figure 5, right): rather than enabling effective forgetting, the per-key prompt causes errors to cluster around the position where the instruction was injected. The model treats the forget cue as an additional anchor point rather than a signal to suppress earlier information; the same pattern holds across all tested models (Figure 20, Appendix C). Rather than mitigating interference, these prompts reshape its spatial distribution without reducing it. A recency cue appended at the final query, run separately on two models (gpt-4.1-nano and qwen3-14b), raises mid-interference accuracy but still reaches the floor by 128 updates (Figure 21, Appendix C).

7.2 Mock QA reset

We also tested a non-natural-language intervention inspired by LLM “hacking” studies (Kuo et al., 2025): a mock QA reset that simulates a completed user–assistant dialogue turn,

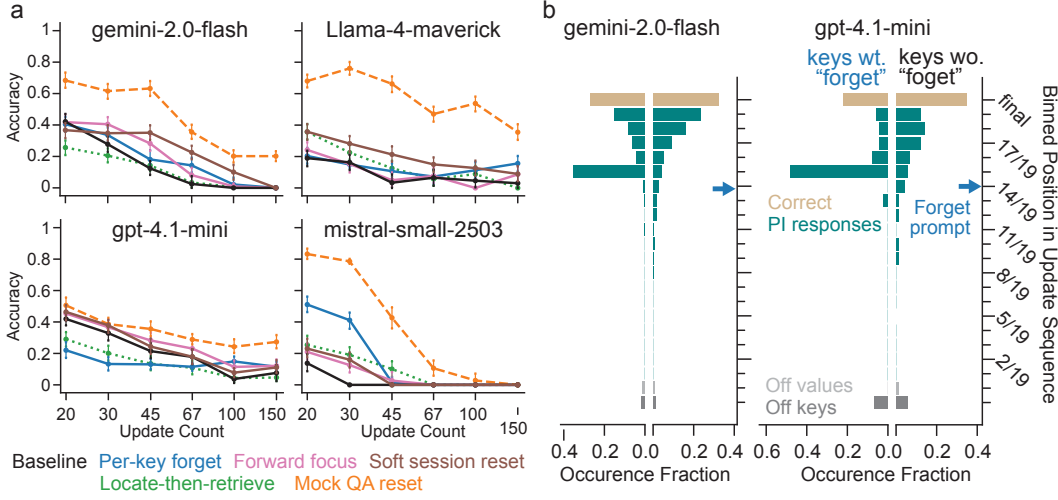


Figure 5: Intervention results. (Left) Explicit forgetting and focusing prompts yield only marginal improvements across four representative models. The mock QA reset (orange dashed) substantially improves accuracy but does not eliminate the log-linear decline. (Right) The per-key forget prompt reshapes rather than reduces interference: keys that received a forget instruction show errors clustered around the injection point, while control keys do not. “Off values” (light gray) denote hallucinated values not present in the input; “off keys” (dark gray) denote failures to return any value. For full prompt designs, see Figure 19 in Appendix C.

framing prior updates as an already-processed task (see Appendix B.6 for the full prompt design). This reset substantially improved retrieval accuracy (orange dashed line in Figure 5, left), consistently outperforming all natural language interventions. However, the overall log-linear decline persists, demonstrating that artificial task boundaries can delay but not eliminate the interference effect.

7.3 Reasoning about the task does not improve retrieval

We tested whether reasoning could overcome the interference bottleneck through two escalating interventions. First, native reasoning models (DeepSeek-R1, Gemini Flash 2.5, Nvidia-Llama) exhibited the same log-linear decline as their base counterparts, performing equal or worse in three of four comparisons despite substantially higher latency (see Figure 22 in Appendix C). Second, a locate-then-retrieve prompt—which first required the model to state the position of the target before extracting it—showed that models could correctly identify the answer’s location but still failed to retrieve it accurately (Figure 5, left, green dotted line). Decomposing these runs quantifies the gap (Figure 23, Appendix C): localization stays correct in 43–100% of sessions while value accuracy collapses—at 150 updates per key, gpt-4.1-mini localizes correctly in 100% of sessions but recovers only 4.7% of values.

Finally, even GPT-5 under maximum reasoning effort (≈ 39.9 k reasoning tokens, 10-minute latency) reached only 71.7% accuracy (Table 5 in Appendix C), confirming that stronger models with larger reasoning budgets can partially mitigate but not eliminate interference. Together these results reveal a dissociation between analytical reasoning and retrieval execution, highlighting the limits of top-down executive control over retrieval.

8 Discussion

Methodological contribution. PI-LLM adapts the A-B, A-C paired-associate paradigm—a workhorse of human PI research—into a diagnostic tool for LLMs. It can be viewed

as a minimal abstraction of MRCR (OpenAI, 2025c) that removes the haystack/search component and isolates the multi-binding core, providing a quantitative measure of the underlying difficulty.

Theoretical implications. Our results suggest that current LLMs possess only an implicit, resource-bounded form of memory selectivity that suffices for low-interference conditions but cannot be strengthened or re-allocated on demand. As a result, when the interference load exceeds this capacity, retrieval accuracy degrades monotonically to near-zero. Within our tested range this degradation shows no sign of plateauing, in contrast to the above-chance stabilization observed in human PI studies, indicating that the control these models exert over their in-context bindings is less flexible than the adaptive executive control that enables humans to resist interference—a difference established behaviorally rather than mechanistically.

Scope of the claim. Our claim is diagnostic, not prescriptive. PI-LLM measures the bottleneck of the LLM component under controlled same-cue interference: a base-prompted LLM need not be the best complete state-tracking solution, and an external state store resolves the overwrite by construction once updates are in a clean structured table. But updates often arrive as open-ended natural language—evolving documents, chats, logs—and the LLM is then the component that must extract, update, and retrieve state from that text, where same-cue interference exposes a bottleneck that external-memory or agentic design should explicitly address.

Limitations and future directions. Several limitations suggest directions for future research. First, while the synthetic PI-LLM benchmark is complemented by a naturalistic validation on five models (Section 6), that validation covers a single domain; more domains are needed to demonstrate the full ecological scope of the interference bottleneck. Second, all models tested are transformer-based; testing state-space models (e.g., Mamba) would clarify whether this bottleneck is specific to attention-based architectures or reflects a more general limitation. Third, every experiment here is inference-only on frozen models, so we cannot distinguish an architectural ceiling from the absence of training pressure; fine-tuning on synthetic PI-style data is the discriminating test, and we do not assert the stronger architectural reading without it. Other untested solution-side directions include curriculum design and explicit memory modules. Fourth, in humans an item’s corpus frequency modulates the long-/short-term memory interaction (Popov & Reder, 2020; Hulme et al., 1997); whether pretraining frequency similarly modulates susceptibility to in-context interference is open, and corpus proxies would make this tractable. Finally, in concurrent work, we conduct representational analyses to investigate the mechanistic basis of this bottleneck; results are to be reported separately.

9 Conclusion

We have introduced PI-LLM, an evaluation framework inspired by proactive interference in cognitive science, and used it to reveal a pervasive interference bottleneck across 35+ LLMs. Retrieval accuracy declines log-linearly as same-key updates accumulate—independently of input length and across multiple load dimensions—with errors dominated by outdated values. Neither forgetting prompts, mock QA resets, nor chain-of-thought reasoning (including GPT-5 under maximum effort) eliminate this decline; asked to first locate the most recent update, models identify its position correctly yet still report an outdated value, revealing a dissociation between verbal reasoning and retrieval execution. A naturalistic validation task on five models confirms that the effect extends beyond synthetic settings. Unlike the plateau observed in human PI studies, the LLM decline shows no sign of leveling off within the range we test, suggesting that current models exercise less flexible control over their in-context bindings than the executive processes that support human working memory. These findings call for approaches—whether through training, inference-time strategies, or architectural design—that strengthen models’ ability to maintain and selectively retrieve current bindings under interference.

Reproducibility

The code for experiments is publicly available at <https://github.com/zhuangziGiantfish/Unable-to-Forget>

Ethics statement

This work evaluates large language models on synthetic and semi-naturalistic key-value tracking tasks. No human subjects, personal data, or sensitive real-world data were used.

Acknowledgments

Special thanks to Sam Zheyang Zheng, Simons Foundation’s Flatiron Institute CCN researcher and NYU Ph.D. candidate, whose clear, contemplative discussions with Chupei(CW)—linking human and LLM focus to dynamical systems—encouraged the initiation of this investigation.

References

- Phillip L. Ackerman, Margaret E. Beier, and Mary O. Boyle. Working Memory and Intelligence: The Same or Different Constructs? *Psychological Bulletin*, 131(1):30–60, 2005. ISSN 1939-1455. doi: 10.1037/0033-2909.131.1.30. Place: US Publisher: American Psychological Association.
- Anthropic. In-context learning and induction heads, March 2022. URL <https://www.anthropic.com/news/in-context-learning-and-induction-heads>. Publisher: Anthropic.
- Alan D. Baddeley, Neil Thomson, and Mary Buchanan. Word length and the structure of short-term memory. *Journal of Verbal Learning and Verbal Behavior*, 14(6):575–589, December 1975. doi: 10.1016/S0022-5371(75)80045-4. URL [https://doi.org/10.1016/S0022-5371\(75\)80045-4](https://doi.org/10.1016/S0022-5371(75)80045-4).
- Qin Dai, Benjamin Heinzerling, and Kentaro Inui. Representational analysis of binding in language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 17468–17493, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.967. URL <https://aclanthology.org/2024.emnlp-main.967/>.
- Yufeng Du, Minyang Tian, Srikanth Ronanki, Subendhu Rongali, Sravan Bodapati, Aram Galstyan, Azton Wells, Roy Schwartz, Eliu A Huerta, and Hao Peng. Context length alone hurts LLM performance despite perfect retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 23281–23298, 2025. URL <https://arxiv.org/abs/2510.05381>.
- Jiahai Feng and Jacob Steinhardt. How do language models bind entities in context? *arXiv preprint*, 2024. doi: 10.48550/arXiv.2310.17191. URL <https://arxiv.org/abs/2310.17191>.
- Sara B. Festini and Patricia A. Reuter-Lorenz. Cognitive control of familiarity: Directed forgetting reduces proactive interference in working memory. *Cognitive, Affective, & Behavioral Neuroscience*, 14(1):78–89, March 2014. ISSN 1531-135X. doi: 10.3758/s13415-013-0231-1. URL <https://doi.org/10.3758/s13415-013-0231-1>.
- Dongyu Gong, Xingchen Wan, and Dingmin Wang. Working Memory Capacity of ChatGPT: An Empirical Study. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(9): 10048–10056, March 2024. ISSN 2374-3468. doi: 10.1609/aaai.v38i9.28868. URL <https://ojs.aaai.org/index.php/AAAI/article/view/28868>. Number: 9.

- Google DeepMind. Introducing gemini 3 pro. <https://blog.google/products/gemini/gemini-3/>, 2025. Accessed: 2025-11-20.
- Yoav Gur-Arieh, Mor Geva, and Atticus Geiger. Mixing mechanisms: How language models retrieve bound entities in-context. *arXiv preprint arXiv:2510.06182*, 2025.
- Joshua K Hartshorne. Visual working memory capacity and proactive interference. *PLoS one*, 3(7):e2716, 2008.
- Jia He, Mukund Rungta, David Koleczek, Arshdeep Sekhon, Franklin X. Wang, and Sadid Hasan. Does Prompt Formatting Have Any Impact on LLM Performance?, November 2024. URL <http://arxiv.org/abs/2411.10541>. arXiv:2411.10541 [cs] version: 1.
- Charles Hulme, Steven Roodenrys, Richard Schweickert, Gordon D. A. Brown, Sarah Martin, and George Stuart. Word-frequency effects on short-term memory tasks: Evidence for a redintegration process in immediate serial recall. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 23(5):1217–1232, 1997.
- Najoung Kim and Sebastian Schuster. Entity tracking in language models. *arXiv preprint arXiv:2305.02363*, 2023.
- Martin Kuo, Jianyi Zhang, Aolin Ding, Qinsi Wang, Louis DiValentin, Yujia Bao, Wei Wei, Da-Cheng Juan, Hai Li, and Yiran Chen. H-CoT: Hijacking the Chain-of-Thought Safety Reasoning Mechanism to Jailbreak Large Reasoning Models, Including OpenAI o1/o3, DeepSeek-R1, and Gemini 2.0 Flash Thinking, February 2025. URL <http://arxiv.org/abs/2502.12893>. arXiv:2502.12893 [cs] version: 1.
- Belinda Z. Li, Zifan Carl Guo, and Jacob Andreas. (how) do language models track state? *arXiv preprint arXiv:2503.02854*, 2025.
- Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the Middle: How Language Models Use Long Contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, February 2024. ISSN 2307-387X. doi: 10.1162/tacl_a.00638. URL https://doi.org/10.1162/tacl_a.00638.
- Klaus Oberauer and Kerstin Vockenberg. Updating of working memory: lingering bindings. *Quarterly Journal of Experimental Psychology (2006)*, 62(5):967–987, May 2009. ISSN 1747-0226. doi: 10.1080/17470210802372912.
- OpenAI. Introducing GPT-5 for developers. <https://openai.com/index/introducing-gpt-5-for-developers/>, 2025a. Accessed: 2025-11-20.
- OpenAI. Introducing GPT-4.1 in the API, April 2025b. URL <https://openai.com/index/gpt-4-1/>. Publisher: OpenAI.
- OpenAI. OpenAI MRCR: Long-context “multiple-needle-in-a-haystack” benchmark, 2025c. URL <https://huggingface.co/datasets/openai/mrcr>.
- Vencislav Popov and Lynne M. Reder. Frequency effects on memory: A resource-limited theory. *Psychological Review*, 127(1):1–46, 2020.
- Kiran Vodrahalli, Santiago Ontanon, Nilesch Tripuraneni, Kelvin Xu, Sanil Jain, Rakesh Shivanna, Jeffrey Hui, Nishanth Dikkala, Mehran Kazemi, Bahare Fatemi, Rohan Anil, Ethan Dyer, Siamak Shakeri, Roopali Vij, Harsh Mehta, Vinay Ramasesh, Quoc Le, Ed Chi, Yifeng Lu, Orhan Firat, Angeliki Lazaridou, Jean-Baptiste Lespiau, Nithya Attaluri, and Kate Olszewska. Michelangelo: Long Context Evaluations Beyond Haystacks via Latent Structure Queries, September 2024. URL <http://arxiv.org/abs/2409.12640>. arXiv:2409.12640 [cs].

A Author contributions

Equal contribution. Listing order is random. Chupei Wang (CW) and Jiaqiu Vince Sun (JVS) co-initiated the development of cognitive-inspired working memory stress tests to reveal this limitation. CW proposed framing LLM retrieval errors as an interference problem and led the iterative trial-and-error testing. JVS implemented and automated the main experiment pipelines, and managed data collection and analysis.

CW designed Experiment 1 revealing how increasing interference degrades LLM retrieval accuracy. Both authors collaborated to engineer Experiment 2, carefully controlling input size to isolate the effects of interference. JVS expanded the framework with Experiment 3, showing that increasing value length intensifies interference. This indicates LLMs’ interference resistance as a unified capacity, exhaustible across different dimensions.

To address the deficits exposed, JVS developed a natural-language intervention and thoroughly analyzed prompt effects and error distributions, while CW pioneered a prompt-hacking strategy, drawing on detailed empirical observations of LLM behavior to mitigate interference.

B Experimental details

B.1 Notes

The initial set of experiments were concluded by May 5th, 2025. We used large language models to assist with restructuring, condensing, and proofreading the manuscript. All scientific content, experimental design, analysis, and interpretation are solely the work of the authors.

B.2 Data construction and task description

Data and performance evaluation. To maintain comparability with human performance, we constructed a word dictionary with up to 46 categories, each comprising 400 words. The token lengths of words within each category were selected to fall within a similar range. Keys were drawn from these category names, and values were randomly selected from the corresponding categories. This dictionary design aligns with cognitive psychology proactive interference tests related to human working memory. Words were randomly selected from the dictionary in each test run to eliminate potential effects of specific semantic combinations. Confidence intervals (CI95) were computed using bootstrap methods after multiple test repetitions.

Detailed task description. Each input sequence consisted of three parts: 1. Instruction—a brief directive indicating the task and specifying which keys to track for value updates. 2. Update stream—the input consists of a sequence of key–value pairs, where a fixed set of keys each receive an equal number of updates. The updates for different keys are randomly interleaved and organized such that the same key does not appear in two consecutive key–value pairs. 3. Query—a prompt directing the model to retrieve the final value associated with each tracked key, reconfirming which keys’ values are to be extracted.

The retrieval objective was to return the most recent value associated with each specified key. For each key with update count X , the preceding $(X-1)$ key–value pairs served as irrelevant, interfering updates—sharing the same key but differing in value.

B.3 Decoding settings, repetitions, and confidence intervals

Decoding. All main experiments use greedy decoding: temperature = 0, top- p = 1.0, frequency and presence penalties at their API defaults, and max_tokens set between 1000 and 4000 depending on the number of tracked keys (large enough that no response was truncated). Reasoning-enabled variants use each provider’s default reasoning configuration except in Table 5, where the reasoning-effort parameter is swept explicitly. The temperature sweep in Figure 11 re-runs two models on paired inputs at temperatures $\{0, 0.3, 0.7, 1.0\}$.

Sessions and bootstrap CIs. A *session* is one complete input–response pair; its score is the proportion of queried keys retrieved correctly. All error bars are 95% bootstrap confidence intervals from 5000 resamples of the session-level scores within a condition, using the percentile method. For the synthetic experiments, sampling was conducted adaptively in batches capped at $N = 20$ or $N = 50$ sessions per run, depending on the experiment, with a floor of $N = 6$ and stopping once two consecutive CI widths fell below 0.10. When a condition was rerun, valid sessions from previous runs with identical parameters were included in the stopping calculation and pooled for analysis. Consequently, final pooled sample sizes per model-by-condition cell could exceed the per-run batch cap, reaching up to 315 sessions. The naturalistic experiment uses a fixed $N = 30$ per model and update count; the locate-then-retrieve analysis pools 83 sessions (gemini-2.0-flash) and 97 sessions (gpt-4.1-mini), with per-point counts annotated in Figure 23.

B.4 Response parsing and error composition

Models are asked to return one value per tracked key, and every reported value must be associated with a key. Parsing is therefore not a single regex but a set of rules assembled by cataloguing how models across the evaluated suite actually phrase their answers: numbered and bulleted lists, “key: value” and “key = value” lines, prose of the form “The final value of X is Y,” answers wrapped in code fences or JSON, and responses that restate the instruction before answering. Values are matched case-insensitively after stripping punctuation and surrounding whitespace; a key with no recoverable value is counted as an error rather than dropped.

Each retrieved value is classified into one of four categories: (i) the correct final value; (ii) an earlier same-key value, i.e. one that genuinely appeared in the update stream for that key but was subsequently overwritten; (iii) an *off value*, a string not present anywhere in the stream for that key; or (iv) *missing/unparseable*. Category (ii) is the proactive-interference signature and dominates the error mass throughout the tested range (Figures 3 and 7). Categories (iii) and (iv) together remain a small minority across the full range up to 400 updates (pooled mean $\leq 9\%$, median 0%). A few weak models eventually produce degenerate output, but only at update counts well beyond the point where their decline is already established, so this cannot account for the decline itself.

B.5 Model versions

The main model suite was evaluated up to May 5, 2025; experiments added during revision were run up to June 8, 2026, and use the model snapshots current at that time. Models with explicit date stamps in their identifiers (e.g., gpt-4o-2024-11-20) represent fixed snapshots, while other identifiers represent the latest available API endpoints as of the corresponding cutoff date. The -thinking suffix indicates the model was evaluated with its native reasoning/Chain-of-Thought (CoT) mode enabled; the counterpart without the suffix was evaluated with this mode disabled.

B.6 Intervention prompt designs

The following intervention prompt designs were tested (see Figure 19 for examples):

Baseline. No intervention prompt is inserted. The model receives the standard instruction, update stream, and query.

Per-key forget. An instruction to disregard previous updates for a specific key is inserted before a new update (e.g., “Forget all the previous updates to visual art”).

Forward focus. An instruction to prioritize information that follows (e.g., “Pay close attention to the following updates”).

Locate-then-retrieve prompt. A prompt that first asks the model to state the position of the target key–value pair within the input before extracting it, designed to test whether explicit reasoning about target location can improve retrieval under interference.

Model Name	Version/Snapshot
<i>Google Gemini Models</i>	
gemini-2.5-flash	gemini-2.5-flash-preview-04-17
gemini-2.0-flash	gemini-2.0-flash
gemini-1.5-flash-8b	gemini-1.5-flash-8b
gemini-2.0-flash-thinking-exp	gemini-2.0-flash-thinking-exp-01-21
<i>OpenAI Models</i>	
gpt-5	gpt-5
gpt-4.1	gpt-4.1-2025-04-14
gpt-4.1-mini	gpt-4.1-mini-2025-04-14
gpt-4.1-nano	gpt-4.1-nano-2025-04-14
gpt-4o	gpt-4o-2024-11-20
gpt-4o-mini	gpt-4o-mini-2024-07-18
<i>Anthropic Claude Models</i>	
claude-haiku-4-5	claude-haiku-4-5-20251001
claude-3.5-sonnet	claude-3-5-sonnet-20241022
claude-3.5-haiku	claude-3-5-haiku-20241022
<i>Alibaba Qwen Models</i>	
qwen2.5-72b	qwen2.5-72b-instruct
qwen3-0.6b	qwen3-0.6b
qwen3-1.7b	qwen3-1.7b
qwen3-4b	qwen3-4b
qwen3-8b	qwen3-8b
qwen3-14b	qwen3-14b
qwen3-32b	qwen3-32b
qwen3-30b-a3b	qwen3-30b-a3b
qwen3-235b-a22b	qwen3-235b-a22b
qwen3-0.6b-thinking	qwen3-0.6b
qwen3-4b-thinking	qwen3-4b
qwen3-8b-thinking	qwen3-8b
qwen3-14b-thinking	qwen3-14b
qwen3-32b-thinking	qwen3-32b
qwen3-30b-a3b-thinking	qwen3-30b-a3b
qwen3-235b-a22b-thinking	qwen3-235b-a22b
<i>Meta LLaMA Models</i>	
Llama-4-maverick	llama-4-maverick-17b-128e-instruct-maas
Llama-4-scout	llama-4-scout-17b-16e-instruct-maas
Llama-3.1-405b	llama-3.1-405b-instruct-maas
Llama-3.2-90b-V	llama-3.2-90b-vision-instruct-maas
Llama-3.1-70b	llama-3.1-70b-instruct-maas
Llama-3.1-8b	llama-3.1-8b-instruct-maas
<i>DeepSeek Models</i>	
deepseek-v3	deepseek-chat
deepseek-R1	deepseek-reasoner
<i>xAI Grok Models</i>	
grok-3-beta	grok-3-beta
grok-3-mini-beta	grok-3-mini-beta
<i>Mistral Models</i>	
mistral-small-2503	mistral-small-2503
<i>NVIDIA Models</i>	
NV-Llama-3.1-ultra	nvidia.llama-3.1-nemotron-ultra-253b-v1
NV-Llama-3.3-super	nvidia.llama-3.3-nemotron-super-49b-v1
NV-Llama-3.1-nano	nvidia.llama-3.1-nemotron-nano-8b-v1

Table 1: Model versions used in our evaluation. The **Model Name** column gives the short names used throughout the paper’s figures and tables; the **Version/Snapshot** column gives the exact API identifier, so this table is the lookup between the two. The “-maas” suffix on the Llama identifiers denotes Google Vertex AI’s Model-as-a-Service endpoint and is not part of the model name.

Soft session reset. A verbal cue marking the start of a new input segment (e.g., “This text stream ends here and a new session of text stream begins next”).

Mock QA reset. A simulated dialogue turn including an initial update segment, a query, a mock assistant response, and remaining updates in a new user turn. This creates an artificial task boundary intended to prompt the model to deprioritize earlier input.

C Supplementary figures and tables

This section presents supplementary figures and tables organized by the main-text sections they support.

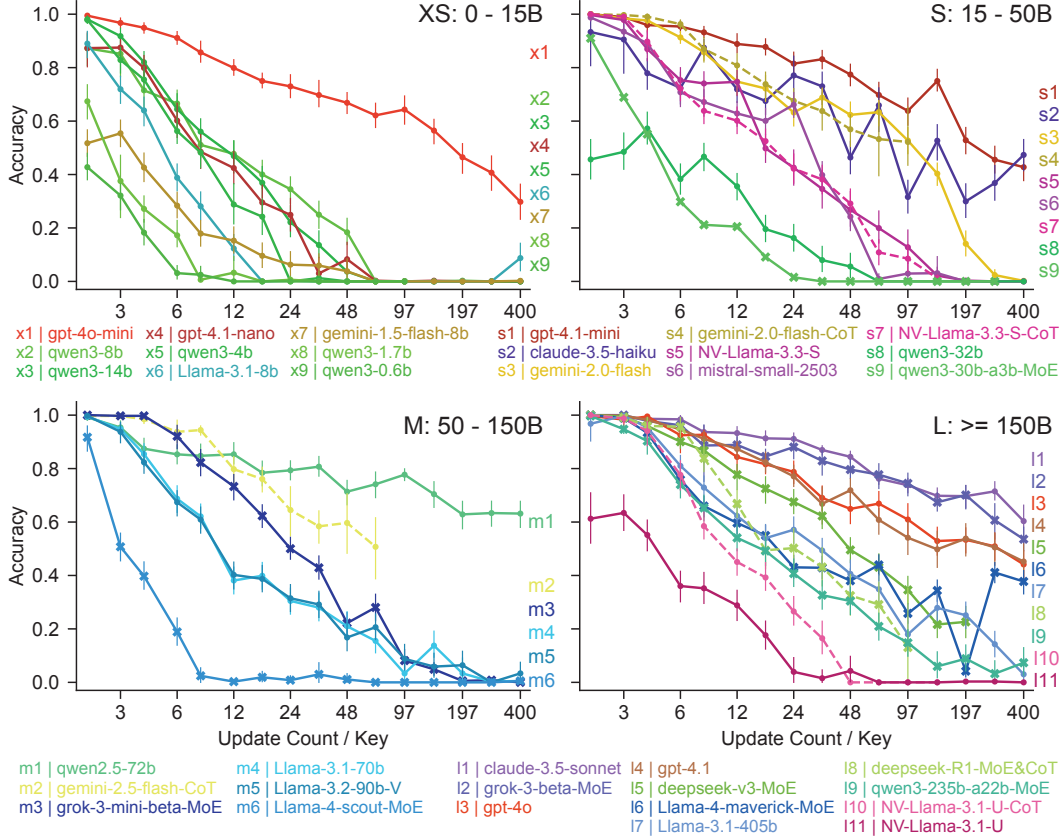


Figure 6: Retrieval accuracy declines approximately log-linearly as the number of same-key updates increases. Models are grouped by estimated parameter size into four tiers—XS, S, M, and L—shown from top to bottom. Larger models degrade more slowly, while smaller models decline fastest. The x-axis is log-scaled, covering update counts from 3 to 400. The number of keys is fixed at 46. Models are color-coded by developer; MoE models are indicated with “X” markers and CoT variants with dashed lines. Error bars indicate bootstrapped 95% confidence intervals.

Model	n_{cond}	n_{sess}	Slope [95% CI]	R^2	Quad. p	Curvature
claude-3.5-haiku	14	1210	-0.073 [-0.081, -0.065]	0.681	0.831	none
claude-3.5-sonnet	14	319	-0.057 [-0.062, -0.052]	0.948	0.154	none
deepseek-v3	12	577	-0.140 [-0.147, -0.133]	0.984	0.042	concave
gemini-1.5-flash-8b	7	801	-0.113 [-0.130, -0.096]	0.879	0.007	convex
gemini-2.0-flash	12	212	-0.116 [-0.122, -0.110]	0.876	0.165	none
gemini-2.5-flash	9	152	-0.125 [-0.145, -0.107]	0.964	0.873	none
gpt-4.1	14	334	-0.085 [-0.091, -0.078]	0.964	0.174	none
gpt-4.1-mini	14	488	-0.080 [-0.086, -0.075]	0.904	0.018	concave
gpt-4.1-nano	8	452	-0.203 [-0.220, -0.186]	0.957	0.243	none
gpt-4o	14	646	-0.082 [-0.088, -0.076]	0.982	0.213	none
gpt-4o-mini	14	397	-0.087 [-0.093, -0.081]	0.958	0.059	none
grok-3-beta	14	167	-0.059 [-0.066, -0.052]	0.940	0.028	concave
grok-3-mini-beta	10	419	-0.200 [-0.208, -0.191]	0.982	0.678	none
Llama-3.1-405b	13	971	-0.123 [-0.129, -0.116]	0.947	0.043	convex
Llama-3.1-70b	11	667	-0.144 [-0.153, -0.135]	0.911	0.005	convex
Llama-3.2-90b-V	12	659	-0.132 [-0.139, -0.125]	0.933	<0.001	convex
Llama-4-maverick	14	875	-0.086 [-0.090, -0.081]	0.695	0.009	convex
mistral-small-2503	8	587	-0.143 [-0.158, -0.128]	0.818	0.404	none
NV-Llama-3.3-super	10	360	-0.169 [-0.181, -0.158]	0.967	0.818	none
qwen2.5-72b	14	288	-0.038 [-0.044, -0.033]	0.885	0.294	none
qwen3-14b	7	505	-0.215 [-0.233, -0.196]	0.992	0.710	none
qwen3-235b-a22b	14	693	-0.127 [-0.133, -0.121]	0.945	<0.001	convex
qwen3-32b	8	846	-0.145 [-0.162, -0.129]	0.934	0.779	none
qwen3-8b	8	703	-0.148 [-0.165, -0.130]	0.981	0.688	none
Llama-3.1-8b	4	375	-0.322 [-0.366, -0.277]	0.982	—	—
NV-Llama-3.1-ultra	5	500	-0.160 [-0.193, -0.126]	0.922	—	—
qwen3-30b-a3b	5	170	-0.196 [-0.233, -0.156]	0.862	—	—
qwen3-4b	5	209	-0.252 [-0.289, -0.214]	0.975	—	—
Llama-4-scout	2	143	—	—	—	excluded
qwen3-0.6b	1	20	—	—	—	excluded
qwen3-1.7b	2	97	—	—	—	excluded

Table 2: Per-model log-linear fits of accuracy against $\log_2(\text{update count})$. **Slope** is the change in accuracy per doubling of the update count, with a bootstrapped 95% CI. R^2 is computed on condition means (the quantity plotted throughout the paper); session-level R^2 is necessarily lower (median 0.34) because of within-condition variance across sessions, and is reported in the released data. Quad. p is the analytic p -value for a quadratic term added to the log-linear fit; Curvature labels its sign where that term is significant at $\alpha = 0.05$ and “none” otherwise. Models with fewer than 6 conditions in their dynamic range (bottom block) yield a slope but not a stable curvature test, marked “—”. Three models (llama-4-scout, qwen3-0.6b, qwen3-1.7b) fall to the accuracy floor after 1–2 conditions, leaving too few points to fit at all, and are excluded from the summary statistics reported in Section 4.

Variable	Coef.	t	P > t	[0.025	0.975]
Intercept	0.1866	2.045	0.051	-0.001	0.374
Parameter Size	0.1030	3.016	0.006	0.033	0.173
Context Size	-3.334e-09	-0.144	0.887	-5.08e-08	4.41e-08

Table 3: OLS regression of IES on model parameter size (ordinal class) and context window length ($N = 30$ models). Parameter size is a significant predictor; context length is not.

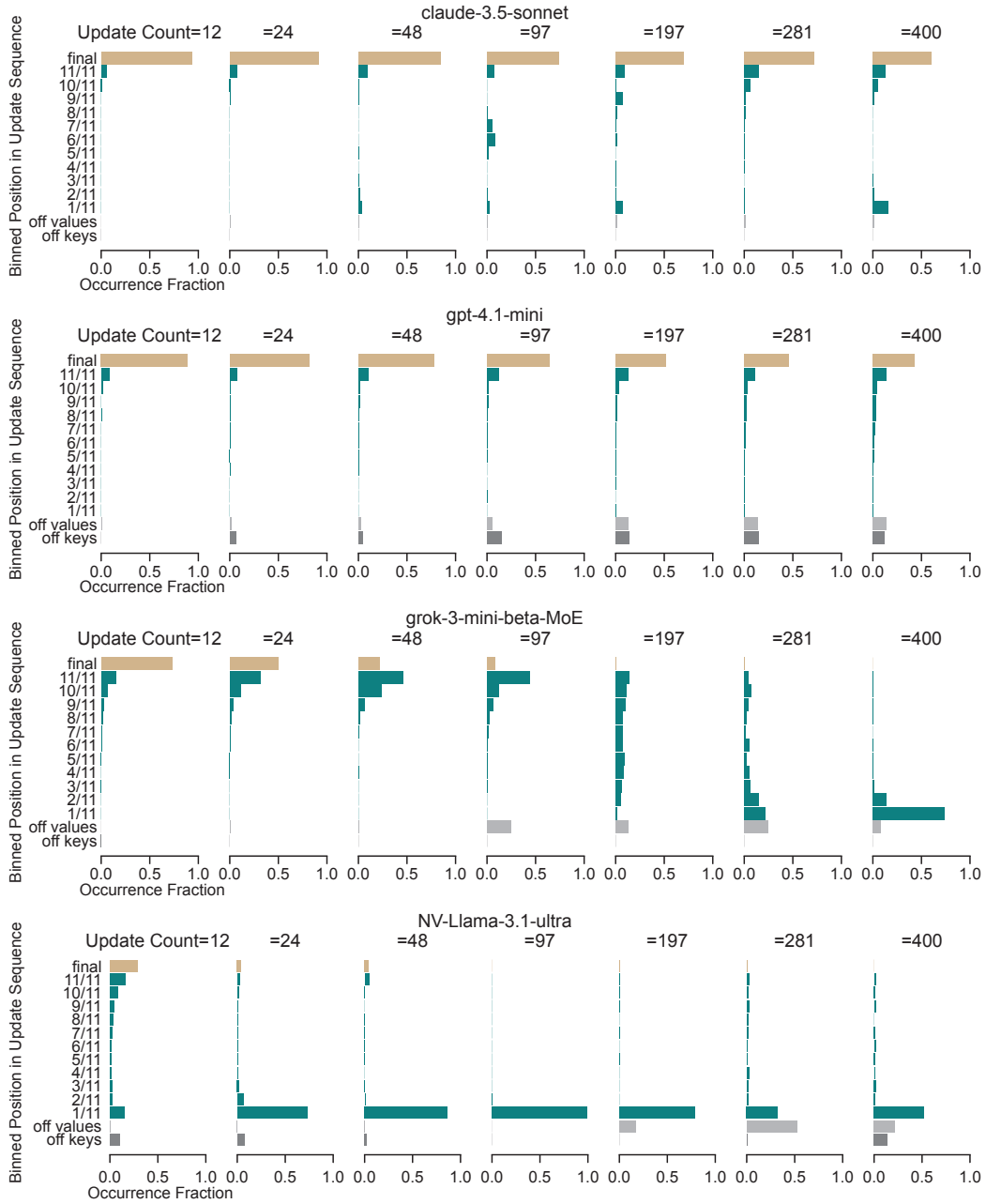


Figure 7: Full response distributions across update positions for more models, extending the two-model summary in the main text.

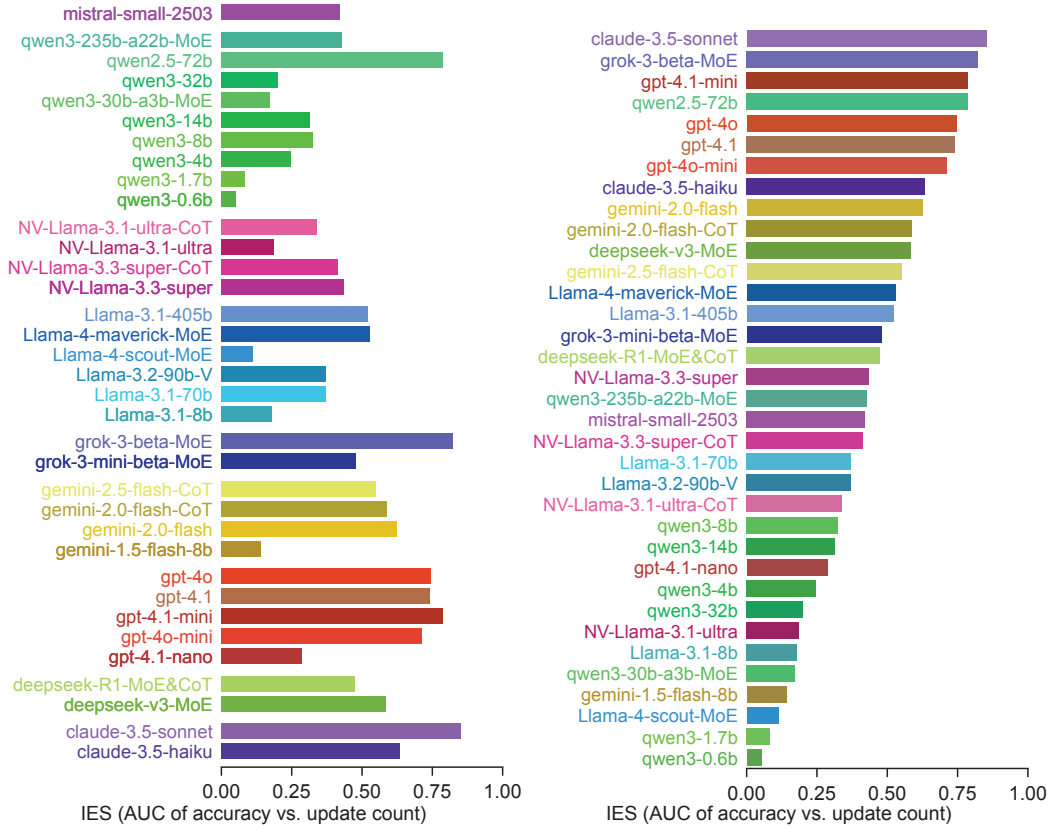


Figure 8: Interference Endurance Score (IES) for all models, computed as the AUC of their accuracy–update-count functions. Left: models are grouped by family and sorted by parameter size; right: ranking by magnitude in descending order.

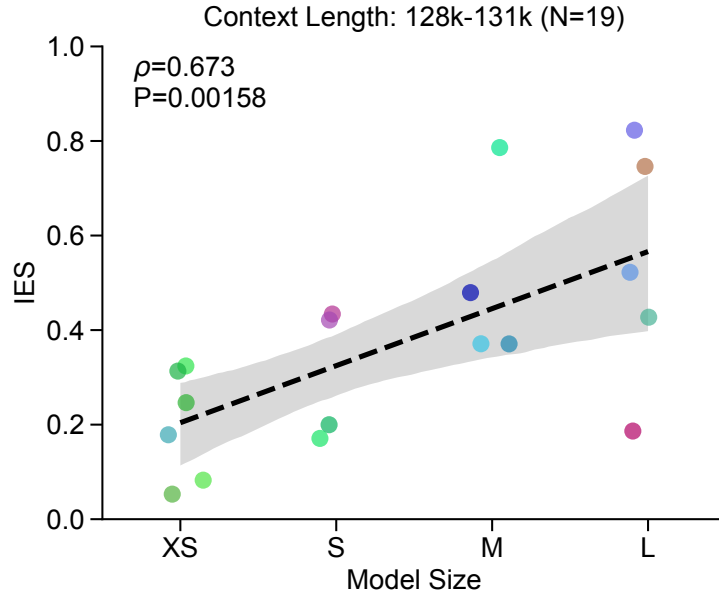


Figure 9: IES correlates strongly with model size class (XS, S, M, L). Analysis restricted to models with context lengths of 128k–131k tokens. Spearman's ρ is reported.

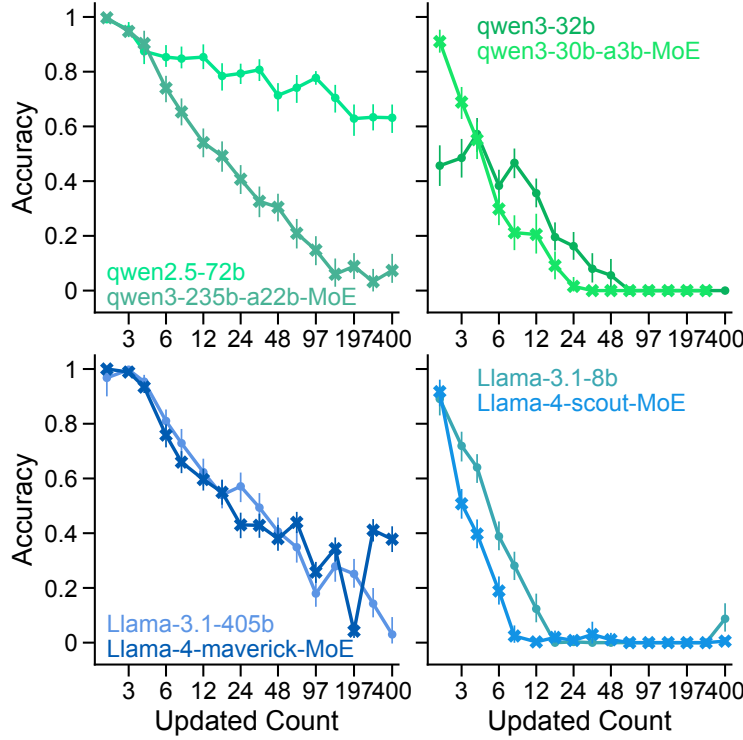


Figure 10: Comparison of retrieval accuracy between MoE and dense models (MoE: “X” markers; active parameters per token are 22B, 3B, 17B, and 17B for the four MoE models shown). In three of four pairs the MoE model declines faster as interference accumulates, and in none does it decline more slowly—including where it holds far more total parameters (Llama-4-scout, 109B total, versus the dense Llama-3.1-8b). The tightest comparison is qwen3-30b-a3b versus qwen3-32b: same generation and tokenizer, near-identical total size, roughly an order of magnitude fewer active parameters, and a steeper decline. These comparisons are observational—MoE and dense variants also differ in post-training, and qwen3-30b-a3b is itself distilled from qwen3-32b—so they indicate that total parameter count overstates the effective capacity of MoE models here rather than isolating active parameters as the cause.

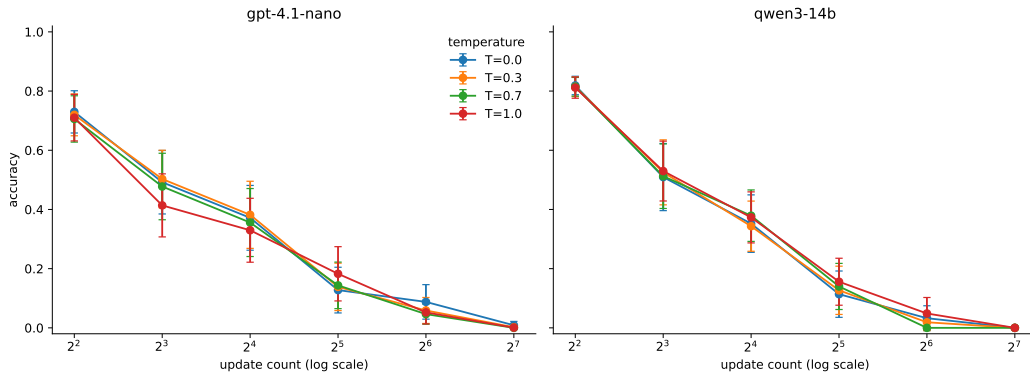


Figure 11: Decoding temperature does not affect the interference decline. gpt-4.1-nano and qwen3-14b re-run on paired inputs at temperatures $\{0, 0.3, 0.7, 1.0\}$; curves overlap and the slope is essentially unchanged (maximum accuracy spread ≤ 0.09 and ≤ 0.05 , respectively). All main experiments use temperature = 0.

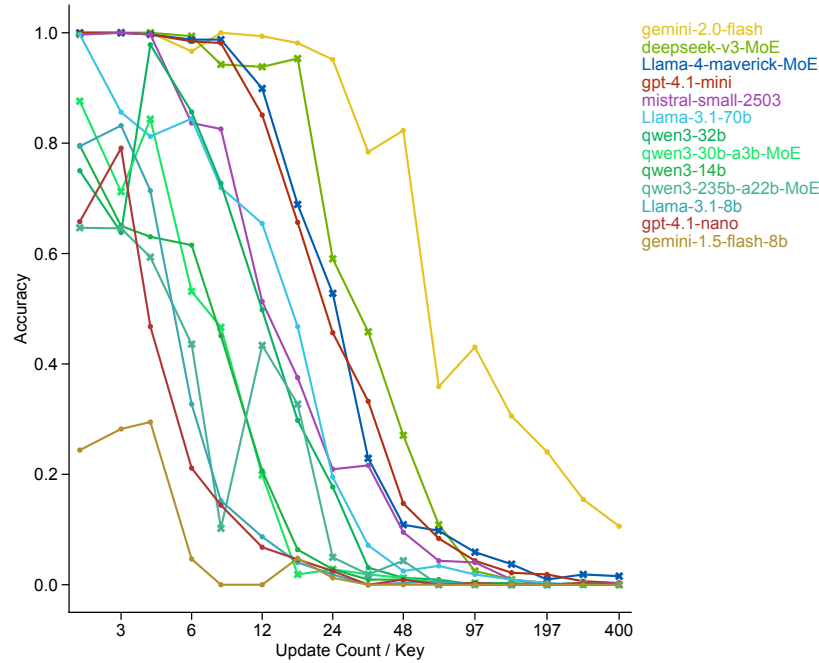


Figure 12: Step-like failure in sequential mode. Accuracy remains near-perfect until a model-specific threshold, then drops abruptly—contrasting with the gradual log-linear decay under random update order. (x-axis: number of same-key updates, log-scaled; asterisk: MoE models.)

Tracked Keys = 2; Updated Keys = 3

INPUT

As my secretary, I need you to carefully read a text stream where the values of multiple keys are being continuously updated. The 2 keys to track include **visual art**, **landform**. I will ask you to identify the current value of each key later. The text stream starts on the next line.

1*visual art: Braque; 1*tools: hook remover; 2*visual art: Pollock; 1*landform: moraine; 3*visual art: Basquiat; 2*tools: plunger; 2*landform: plain; 3*tools: ruler; 4***visual art: seascape**; 3*landform: valley; 4*tools: hammer; 4***landform: dune**.

What are the current value of each key (**visual art**, **landform**) you are tracking? End your response with: 'The current value of <key> is <value>.'

DESIRED ANSWER

The current value of **visual art** is seascape. The current value of **landform** is dune.

Figure 13: Input example illustrating how the model is prompted to track a subset of updated keys. Here, “visual art” (blue) and “landform” (orange) are tracked, while “tools” (gray prefix) appears in the update stream but is not queried. Bold text highlights the target key-value pairs.

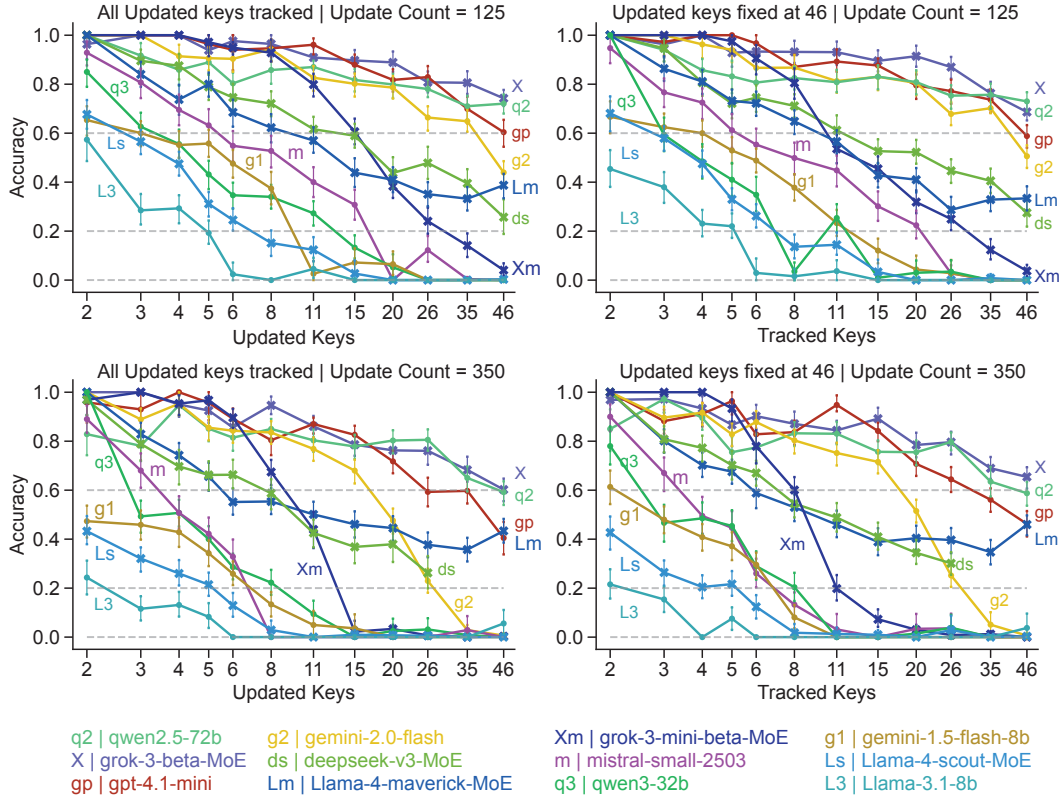


Figure 14: Full version of the updated-keys vs. tracked-keys comparison, including both update count = 125 (upper panels) and update count = 350 (lower panels). Left panels: Setting A (number of updated keys). Right panels: Setting B (number of tracked keys at fixed input length). MoE models: “X” markers. Error bars: bootstrapped 95% CIs.

Model	≈250 pairs			≈500 pairs			≈1000 pairs		
	deep / wide	Δ		deep / wide	Δ		deep / wide	Δ	
deepseek-v3	1.00 / 0.90	+0.099		0.88 / 0.78	+0.098		0.72 / 0.68	+0.046	
gemini-1.5-flash-8b	0.65 / 0.28	+0.369		0.55 / 0.15	+0.399		0.37 / 0.06	+0.311	
gemini-2.0-flash	1.00 / 0.91	+0.087		0.91 / 0.75	+0.166		0.94 / 0.63	+0.310	
gpt-4.1-mini	1.00 / 0.95	+0.046		1.00 / 0.89	+0.111		0.95 / 0.81	+0.131	
grok-3-beta	0.96 / 0.96	+0.001		1.00 / 0.89	+0.111		0.96 / 0.88	+0.084	
grok-3-mini-beta	1.00 / 0.92	+0.079		1.00 / 0.73	+0.267		0.93 / 0.50	+0.429	
Llama-3.1-8b	0.57 / 0.39	+0.185		0.29 / 0.12	+0.170		0.00 / 0.00	-0.003	
Llama-4-maverick	1.00 / 0.76	+0.241		0.74 / 0.60	+0.142		0.62 / 0.43	+0.192	
Llama-4-scout	0.68 / 0.19	+0.487		0.47 / 0.00	+0.471		0.15 / 0.01	+0.143	
mistral-small-2503	0.93 / 0.71	+0.221		0.69 / 0.63	+0.066		0.53 / 0.66	-0.133	
qwen2.5-72b	1.00 / 0.85	+0.147		0.86 / 0.85	+0.006		0.86 / 0.79	+0.065	
qwen3-32b	0.85 / 0.38	+0.467		0.56 / 0.36	+0.199		0.34 / 0.16	+0.178	

Table 4: Matched-volume comparison of the two interference axes. At each total volume of key-value pairs we compare a *deep-narrow* configuration (2, 4, or 8 keys $\times$ 125 updates each) against a *shallow-wide* one (46 keys $\times$ 6, 12, or 24 updates each). Cells give deep-narrow / shallow-wide accuracy; Δ is their difference, so positive values mean the deep-narrow condition is more accurate and therefore that adding keys costs more than adding updates at matched volume. $\Delta > 0$ in 34 of 36 matched pairs (median +0.145). The two exceptions both occur at ≈1000 pairs with at least one side at or near the accuracy floor.

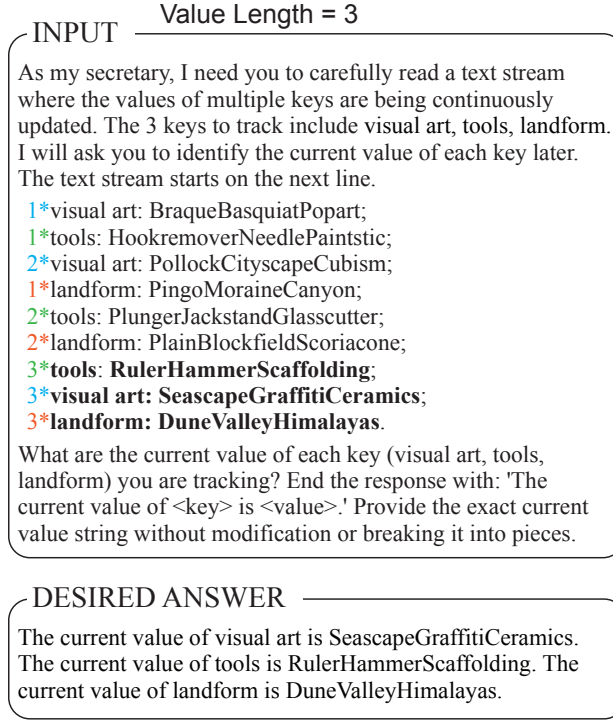


Figure 15: Input example with manipulation of value length. Three items from the same category are concatenated into a single value. Bold text indicates the retrieval targets.

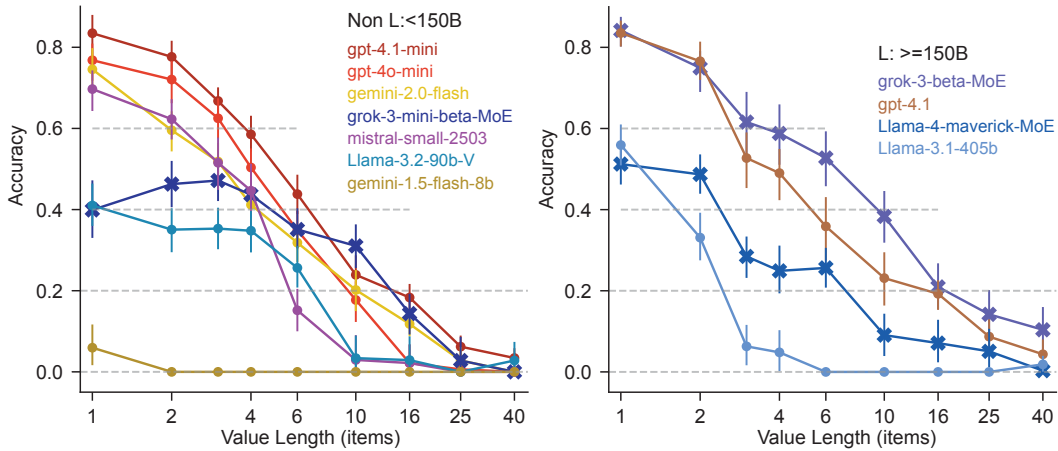


Figure 16: Retrieval accuracy declines approximately log-linearly as value token length increases, with a steeper slope than for other interference dimensions. Update count is fixed at 20. Error bars: bootstrapped 95% CIs.

Setting	Reasoning tokens	Time	Correct / 46	Accuracy
Medium effort	15.9k	6 min 25 s	18	39.1%
High effort	39.9k	10 min 00 s	33	71.7%

Table 5: GPT-5 PI recall under medium- and high-effort reasoning. Even with ≈ 39.9 k reasoning tokens and 10-minute latency, accuracy reaches only 71.7%.

--- START OF SLACK HISTORY EXPORT ---

Channel: #project-alpha-launch

Topic: Launch Event Logistics & Planning

[System]: Project initialized. Waiting for decisions.

[Day 1 09:41] Alice: Budget update: The total funding is now set at \$541,603.

[Day 1 10:17] Eve (Finance): Budget change: The authorized spending limit is now \$1,684,675.

[Day 1 12:33] Bob: Rights cleared for the new song. We are using "Wild Midnight World".

[Day 1 14:33] Alice: We are locking the final budget at \$1,073,223.

[Day 1 15:45] Frank (Legal): Team, regarding the venue: We are officially switching the location to Neo Valley Hall, Building C3.

[Day 1 16:59] Bob: Music choice change: The official theme song will be "Wild Midnight Fire".

[Day 1 18:49] Frank (Legal): Confirmed: The official date for the product launch is May 13, 2026.

[Day 2 09:10] Charlie: Let's finalize the location. We are going with Grand Valley Arena, Building A9.

[Day 2 10:52] Charlie: Finance has adjusted the budget cap. We now have \$1,694,079.

[Day 2 13:56] Alice: We need to reschedule. The event will now happen on this date: March 4, 2026.

[Day 2 15:31] Charlie: Update on the place: Management wants us to use Blue Link Center, Building A4 instead.

[Day 2 17:39] Dave (PM): Logistics update: We will hold the event at Prime Harbor Arena, Building C4.

[Day 3 09:22] Bob: We need to reschedule. The event will now happen on this date: February 8, 2026.

[Day 3 12:35] Frank (Legal): The team voted on the song. The winner is "Eternal Summer Dreams".

[Day 3 15:34] Eve (Finance): The team voted on the song. The winner is "Urban Morning Soul".

[Day 3 17:55] Eve (Finance): We need to reschedule. The event will now happen on this date: May 24, 2026.

--- END OF SLACK HISTORY ---

Based on the chat history above, identify the FINAL agreed upon details for the launch event.

Please output the answer in the following format:

Place: <value>

Date: <value>

Budget: <value>

Song: <value>

Figure 17: Input example for the Product Launch State Tracking task (update count = 4). Updates are embedded within a naturalistic multi-turn team chat history (simulating a Slack export). The model must track state changes of four entities (Budget, Song, Place, Date) amidst conversational noise and retrieve the final agreed-upon values.

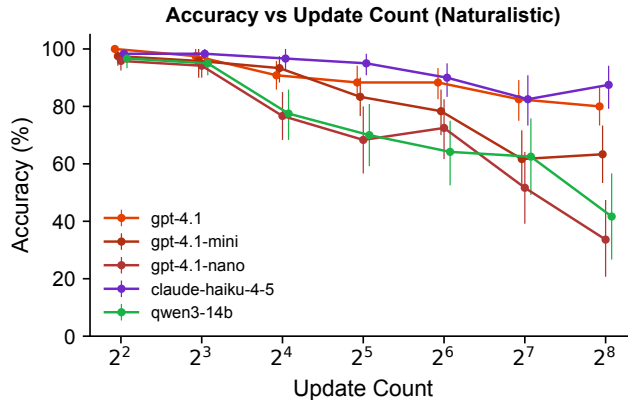


Figure 18: Naturalistic validation on the Product Launch State Tracking task. Five models across three families all decline approximately log-linearly as the update count grows from 4 to 256, replicating the synthetic pattern in a multi-turn team-chat setting. $N = 30$ sessions per model and update count; error bars are bootstrapped 95% CIs.

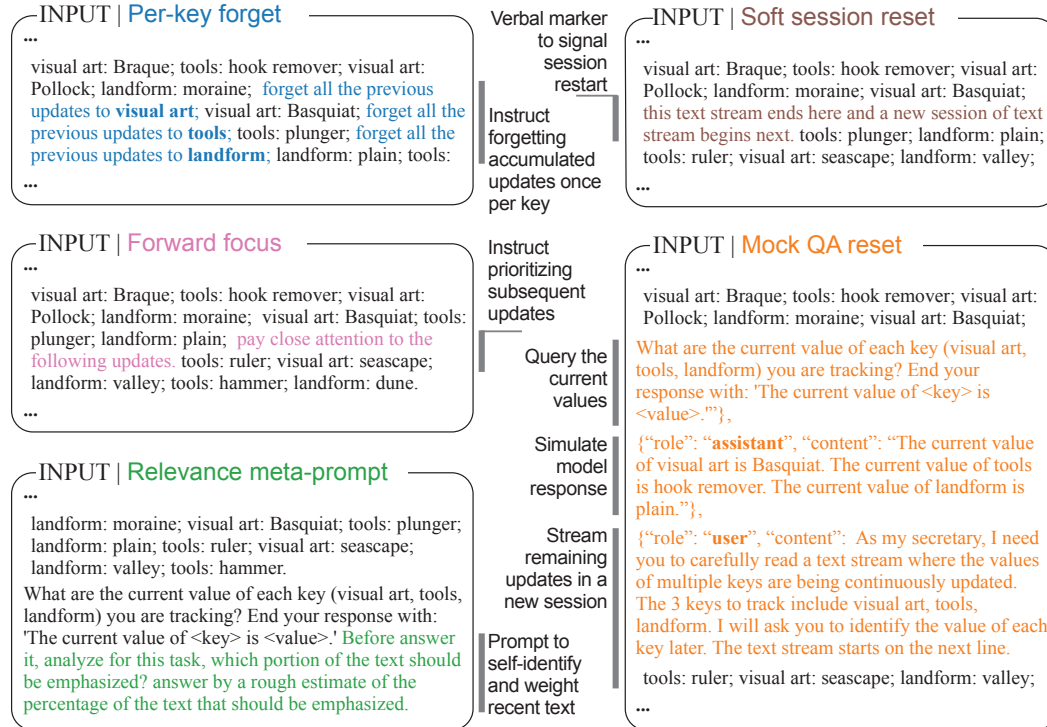


Figure 19: Example input illustrating intervention strategies for mitigating proactive interference. Five strategies are shown: *Per-key forget* (green), *Forward focus* (magenta), *Locate-then-retrieve* (green dotted), *Soft session reset* (brown), and *Mock QA reset* (orange). Instructional cues are shown in colored text; role labels in the Mock QA reset are in bold.

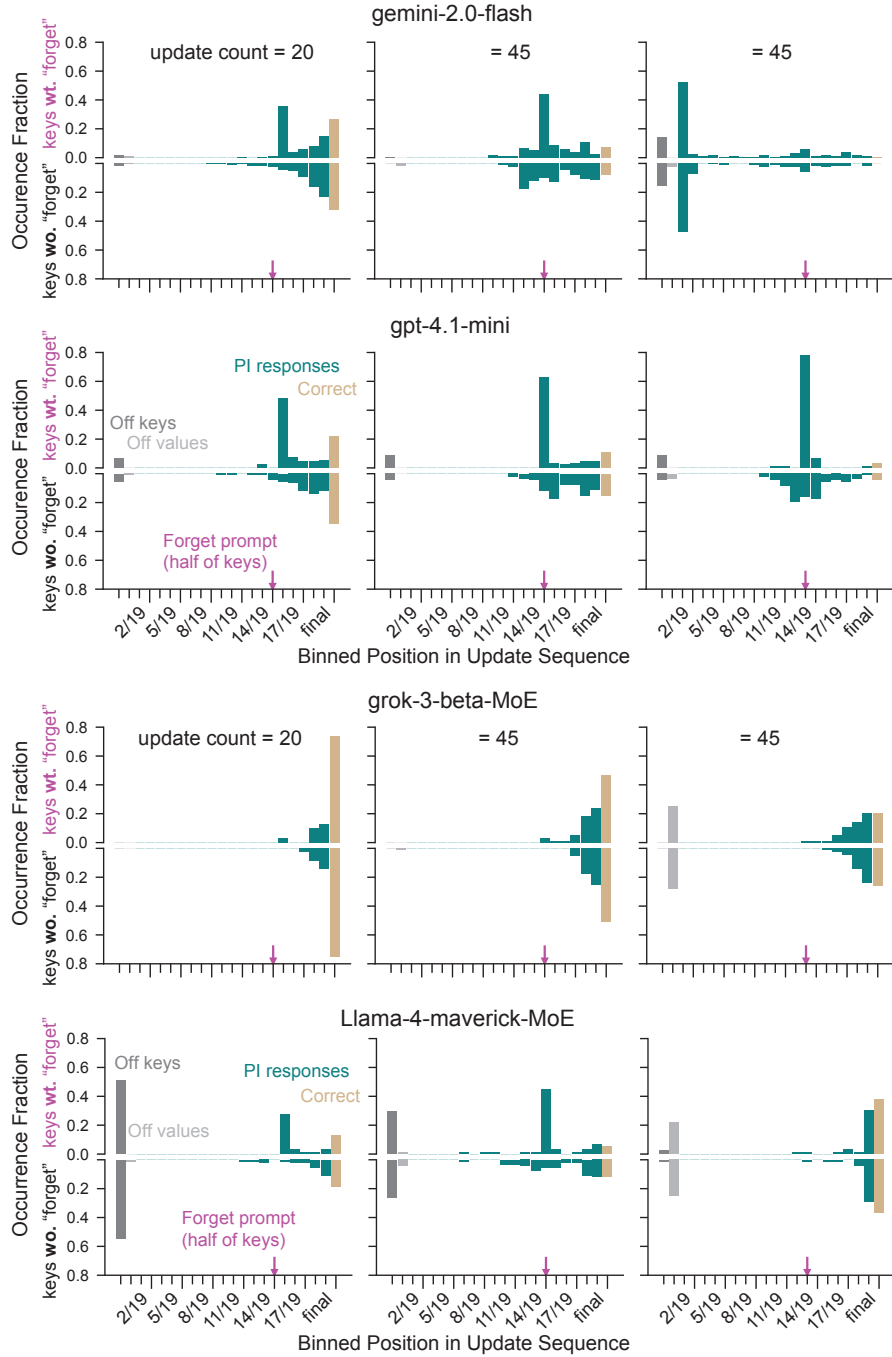


Figure 20: Full per-key forget error distributions across all tested models. Keys that received a forget instruction show error concentration around the injection point, while control keys do not.

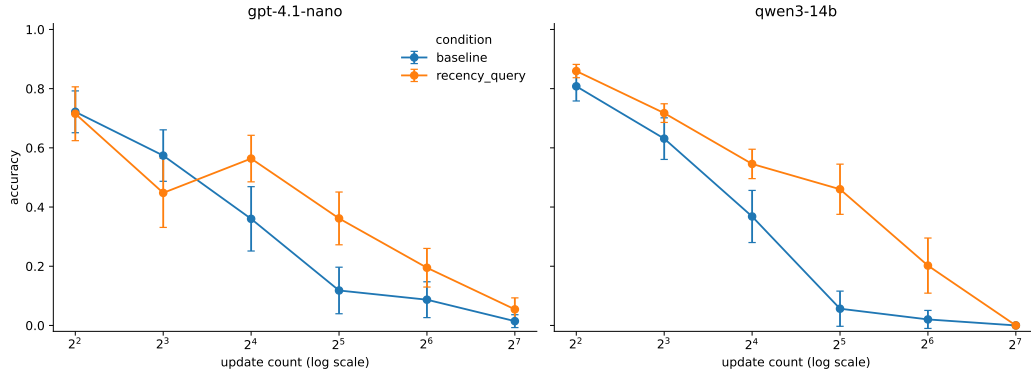


Figure 21: A recency-oriented cue appended at the final query (“report only the most recent value and disregard all earlier updates”) compared against the standard query on paired inputs. The cue lifts mid-interference accuracy on both models but does not flatten the decline: by 128 updates both converge to the floor regardless of the cue.

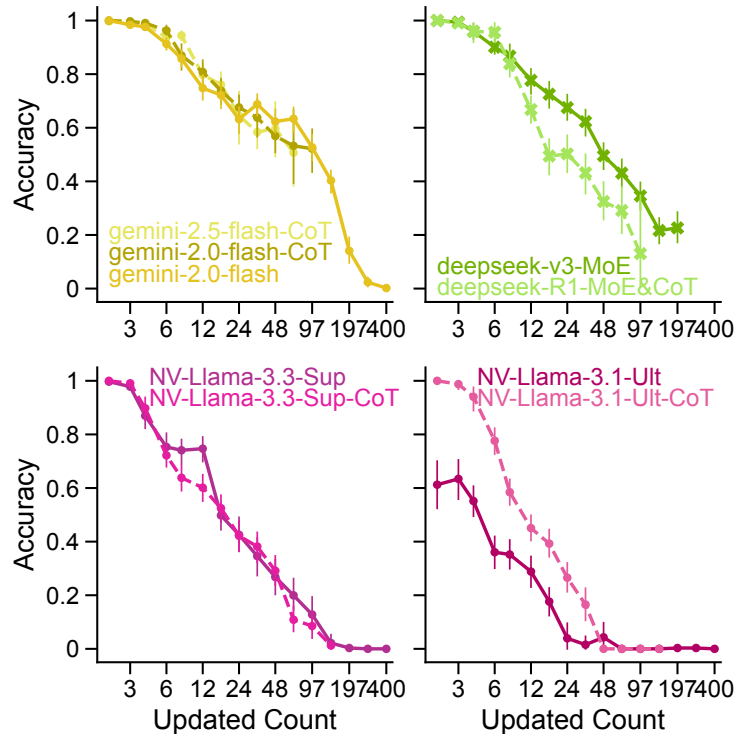


Figure 22: Chain-of-thought reasoning does not improve retrieval. Accuracy versus update count for four pairs of reasoning-enabled models and their base versions. In three of four comparisons, the reasoning variant performs equal to or worse than its base. Dashed: reasoning variants; solid: base models. x-axis is log-scaled.

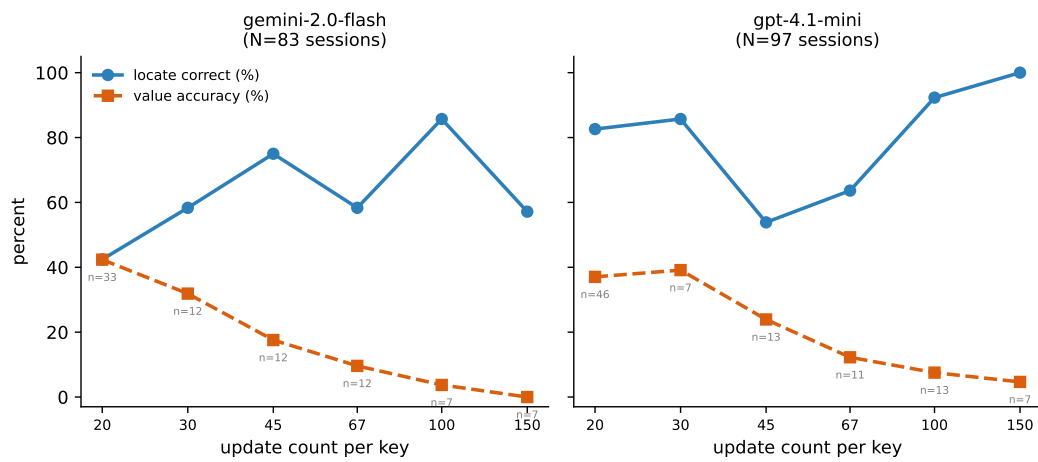


Figure 23: Locating the target and retrieving it dissociate. In the locate-then-retrieve runs, the rate at which the model correctly localizes the most-recent region (solid) stays high as interference grows, while the accuracy of the value it then reports (dashed) collapses. Per-point session counts are annotated; N is the total number of sessions per model.