LoRA Readout: Learning to Read Parametric Memories for Context-Free Inference

Anonymous authors
Paper under double-blind review

Abstract

Encoding context into model parameters can reduce the inference costs of in-context learning, but conventional fine-tuning can be costly and require context-specific supervision. We introduce LoRA Readout, a framework that separates context writing from memory reading to enable context-free inference. The writer encodes raw context into a memory LoRA through next-token prediction, without requiring context-specific question-answer pairs or other auxiliary supervision. A shared reader is trained across frozen memories to access the encoded knowledge and use it to answer queries. Incorporating a new context requires only an offline memory-writing step, while the reader is reused without further adaptation. At inference, the model combines the memory and reader to answer queries without including the original context tokens in the prompt. Experiments across multiple datasets show that LoRA Readout outperforms related approaches with a small training-token budget: only 33.75M training input tokens for source-memory writing and reader training on Qwen3-8B, less than $1/26$ of the smallest context-training corpus reported by these approaches.

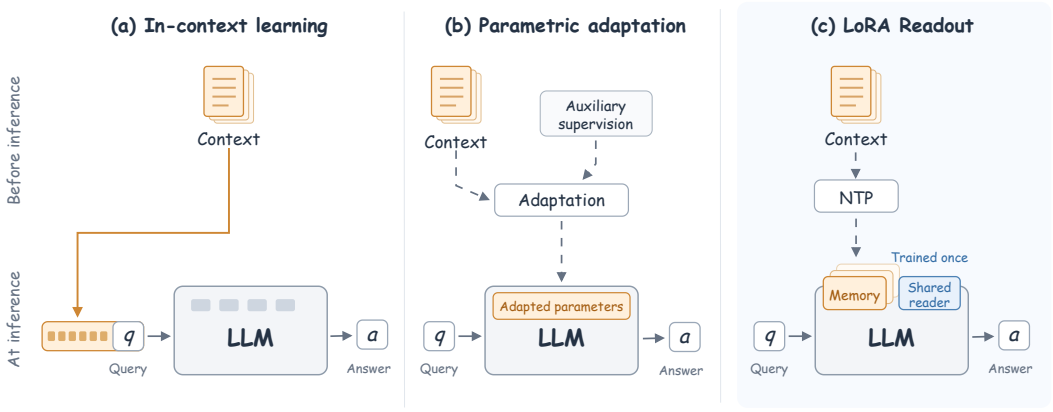


Figure 1: Approaches to incorporating new knowledge. (a) ICL includes context tokens alongside each query. (b) Parametric adaptation can use auxiliary supervision to encode context into model parameters. (c) LoRA Readout writes raw context into a memory LoRA through NTP and reuses a shared reader across contexts. Faint memory outlines denote alternative context-specific memories.

1 Introduction

Many applications of large language models (LLMs) require models to incorporate new knowledge. Two common approaches are in-context learning (ICL), which provides this knowledge in the input context, and parametric adaptation, which encodes it in model parameters. ICL incurs higher inference costs as context length grows, while parametric adaptation can require costly optimization and additional supervision, such as annotated examples, synthetic data, or teacher distillation signals. This motivates a practical goal: rapidly compress raw context into model parameters, then access its information through those parameters without including the original text in the inference prompt.

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

Next-token prediction (NTP) offers a natural starting point because it derives supervision directly from raw text. This standard pretraining objective enables models to acquire knowledge by predicting each token from the preceding text, without context-specific annotations, synthetic data, or teacher signals. We envision using NTP to encode the knowledge in new contexts as reusable *parametric memories*. If models can reliably access this knowledge in response to queries, NTP could offer a simple, broadly applicable route to context-free inference.

However, acquiring knowledge through NTP does not guarantee that a model can use it to answer questions. Prior work shows that models trained with NTP can achieve low perplexity on newly learned text yet struggle to answer questions about it, a phenomenon known as the *perplexity curse* (Jiang et al., 2024). Studies of parametric knowledge likewise distinguish what a model has learned from what it can use in downstream tasks (Kazemnejad et al., 2023). We therefore focus on learning to access the knowledge encoded by NTP.

The training process of LLMs suggests a way to bridge this gap. During NTP pretraining, models acquire knowledge from raw text; subsequent post-training helps them use that knowledge to respond to instructions (Zhou et al., 2023). Following this division, we separate knowledge acquisition from knowledge access. After NTP encodes a context in parameters, we freeze the resulting memory and learn an additional parameter update (a *shared reader*) across these memories to help the model answer questions using their contents. We hypothesize that this reading ability can transfer to new contexts, allowing the learned update to be reused with newly written memories.

We introduce *LoRA Readout*, which combines a context-specific memory LoRA with a shared reader LoRA on a frozen backbone (Hu et al., 2021). We construct training data from public QA datasets by consistently replacing entities and dates across contexts and question–answer pairs while preserving factual relations. These replacements encourage the reader to recover newly assigned associations. After writing each training context into a memory through NTP, we freeze the memories and train the reader across context episodes using Reptile (Nichol et al., 2018), combining answer prediction with answer-level contrastive learning. For a new context, we construct a memory offline using NTP and pair it with the trained reader, without additional QA supervision or reader adaptation. Together, the two adapters enable the model to answer questions using knowledge encoded in its parameters without accessing the original context text (Figure 1).

Across multiple datasets, LoRA Readout outperforms the evaluated generator checkpoints under matched-backbone evaluation while using only 33.75M training input tokens on Qwen3-8B, less than $1/26$ of the smallest context-training corpus reported by these approaches (Chen et al., 2024; Charakorn et al., 2026; Liu et al., 2026). In our diagnostics, controlled comparisons show gains beyond general QA adaptation, while memory interventions identify fact-relevant signals that influence answers. The benefits depend on the writing regime and query formulation, separating document fit from reliable question answering. We also measure the training budget and the repeated-query savings from moving context processing offline. These results support learning reusable access to independently written NTP memories and characterize the limits of that access.

Our contributions are threefold:

- We introduce LoRA Readout, which pairs independently NTP-written memories with a reusable reader, enabling context-free inference on new contexts without target-context QA supervision or reader adaptation.
- We develop an episodic reader-training procedure that combines Reptile updates with answer prediction and contrastive supervision to learn access across frozen memories.
- We demonstrate gains over matched generator checkpoints across five datasets and three backbones with only 33.75M training input tokens on Qwen3-8B, and quantify repeated-query savings.

2 Related Work

2.1 Context-Free Inference and Parametric Context Compression

Hypernetwork-based approaches compress contexts into model parameters by learning to generate context-specific adapters. Generative Adapter maps context hidden states to layer-wise weight up-

dates using a learned adapter generator (Chen et al., 2024). SHINE reuses the frozen LLM within an in-context hypernetwork and adds a lightweight Transformer to generate LoRA parameters across layers (Liu et al., 2026). Doc-to-LoRA meta-learns context distillation, generating a LoRA that approximates the behavior of a teacher with access to the source text (Charakorn et al., 2026). Their common idea is to learn a context-to-parameter mapping, so that new contexts can be internalized through forward computation. LoRA Readout places the shared learning in knowledge access: raw-text NTP writes each memory independently, and a reader learns across the resulting frozen memories. This avoids training a parameter generator and allows the same reader to access new memories without further adaptation. Each new context requires offline NTP optimization, whose cost can be amortized across repeated queries.

LLoCO (Tan et al., 2024) uses LoRA instruction tuning to improve the use of compressed summary embeddings, which remain part of the inference context. Deep Context Distillation (DCD) (Caccia et al., 2025) pairs document-specific LoRA modules with a shared knowledge extractor, but its NTP-written modules provide limited gains over the extractor alone in the reported experiments. We share the goal of learning to access stored knowledge and train a reusable reader across independently written, frozen NTP memories. New contexts require only raw-text memory writing, without target-context QA supervision or reader adaptation, and inference requires no document-derived context embeddings. See Appendix A.1 for a detailed comparison.

2.2 Next-Token Prediction and Knowledge Accessibility

Raw-text NTP offers a simple way to encode document knowledge in parameters because it derives supervision from the document itself. However, fitting text and accessing its information through downstream questions are distinct capabilities (Kazemnejad et al., 2023; Allen-Zhu & Li, 2023). The perplexity curse makes this gap concrete: low document perplexity can coexist with poor QA performance (Jiang et al., 2024). Document-LoRA experiments likewise find raw-text NTP less effective than QA supervision in their evaluated settings (Zuo et al., 2026). Prior approaches seek to make document learning more useful for subsequent queries. Pre-instruction-tuning develops QA ability before document learning (Jiang et al., 2024), while CaMeLS meta-learns token-loss weights to improve QA after adaptation (Hu et al., 2023). Denoising objectives, synthetic continued pretraining, and mixed training on storage and query formats further explore how learning objectives and data presentation affect knowledge use (Saito et al., 2024; Yang et al., 2024; Zhang et al., 2025).

Our work asks whether a reader trained across independently written NTP memories can learn a reusable way to access their contents. We retain raw-text NTP as the writer: each memory is optimized with the reader inactive and then frozen, while a shared reader learns from source-context QA across these memories. This concentrates supervision for knowledge access in a reusable reader-training stage. Improvements with the same frozen memories and sensitivity to replacing the attached memory support the reader's use of context-specific information (Section 5.3 and Appendix A).

3 LoRA Readout

3.1 Overview and Design Rationale

LoRA Readout combines a frozen backbone, a context-specific memory LoRA, and a shared reader LoRA (Figure 2). NTP writes each memory independently; source-context QA then trains the reader across frozen memories. At deployment, the trained reader is paired with a newly written memory.

Memory and reader parameterization. Let θ denote the frozen backbone parameters, ϕ_d the memory parameters for context C_d, and ψ the shared reader parameters. Both adapters use low-rank updates (Hu et al., 2021). For an adapted projection with base weight W_ℓ, the effective weight is

$$W_{\ell,d}^{\mathrm{eff}} = W_\ell + \Delta W_{\ell,d}^{\mathrm{mem}} + \Delta W_\ell^{\mathrm{read}}, \tag{1}$$

where $\Delta W = (\alpha/r)BA$ has rank r and scale α; memory and reader ranks may differ. Switching contexts replaces ϕ_d while retaining θ and ψ. We write p_{θ,ϕ_d} for the memory-only model and $p_{\theta,\phi_d,\psi}$ for the combined model.

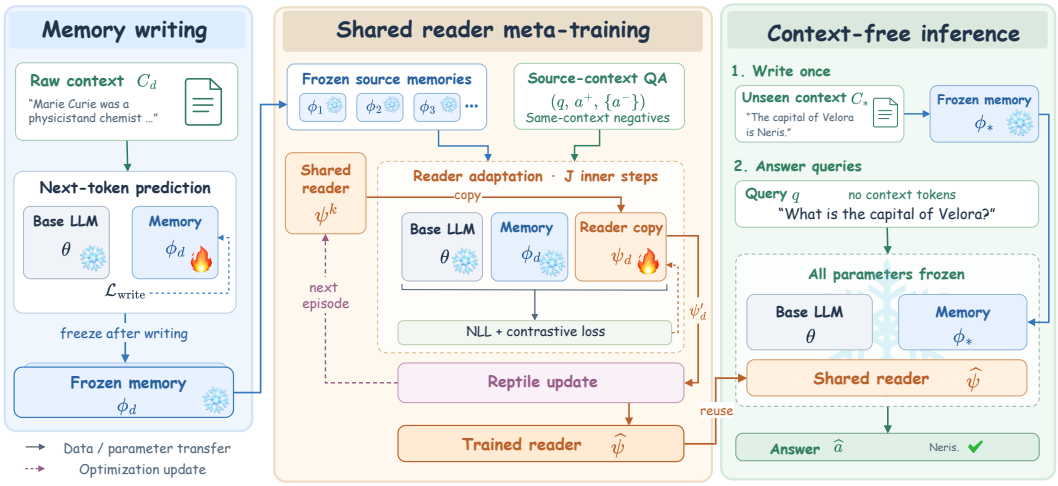


Figure 2: LoRA Readout. Left: NTP writes raw text into a memory LoRA. Center: NLL and contrastive losses adapt a reader copy within each episode; Reptile updates the shared reader across frozen memories. Right: write an unseen context once and reuse the trained reader for question-only inference. Snowflakes and flames denote frozen and trainable parameters.

Design rationale. NTP fits document continuations, whereas QA requires answers under question prompts (Jiang et al., 2024). Training across memories written with the same backbone and procedure may bridge this gap: the reader can change hidden states entering frozen memory branches and computation following their outputs. Section 5.3 tests these interactions.

3.2 Independent Memory Writing with NTP

For each context C_d, we initialize a memory adapter at ϕ_0 and optimize only its parameters, with the backbone frozen and the reader inactive. The context is tokenized with a fixed shared writing prefix and a termination token. For a token window w, the writing objective is

$$\mathcal{L}_{\text{write}}(\phi; w) = -\frac{1}{|w| - 1} \sum_{t=2}^{|w|} \log p_{\theta,\phi}(w_t \mid w_{<t}), \tag{2}$$

where conditioning is restricted to preceding tokens within the window. We use AdamW to optimize these sequence losses and denote the resulting memory by $\phi_d = \text{Write}_\theta(C_d; \phi_0)$. Writing uses the context text without QA targets or teacher-distillation signals.

We store and freeze each memory before reader training. Source and deployment contexts use the same writing procedure, independently of the reader parameters.

3.3 Learning a Shared Reader across Memories

Each frozen memory ϕ_d and QA collection $\mathcal{Q}_d$ form a training task, with the memory active and source text absent from the prompt. Consistent entity replacement encourages learning newly assigned associations (Section 4.1). Reptile (Nichol et al., 2018) combines context episodes trained with the following objectives.

Answer prediction. For a question q and candidate answer a, we compute the mean negative log-likelihood over the answer tokens:

$$\ell_\psi(q, a; \phi_d) = -\frac{1}{|a|} \sum_{t=1}^{|a|} \log p_{\theta,\phi_d,\psi}(a_t \mid q, a_{<t}). \tag{3}$$

Here, a includes its termination token; question, prompt-formatting, and padding positions are masked. Per-answer normalization permits comparison across candidate lengths.

Distinguishing answers within a context. We supplement reference-answer NLL with a ranking loss against same-context candidates, using related answers and entity, date, or number substitutions (Section 4.2).

For a reference answer a^+ and K negative candidates $\{a_j^-\}_{j=1}^K$, let $\ell^+ = \ell_\psi(q, a^+; \phi_d)$ and $\ell_j^- = \ell_\psi(q, a_j^-; \phi_d)$. The per-example objective is

$$\mathcal{L}_{\text{read}} = \ell^+ + \frac{\lambda}{K} \sum_{j=1}^{K} \text{softplus}\big(m + \ell^+ - \ell_j^-\big)\,, \tag{4}$$

where $\text{softplus}(z) = \log(1 + e^z)$, λ weights the ranking term, and m is its margin. The loss favors lower mean NLL for the reference than for competing answers and is averaged over each minibatch.

Reference-answer NLL teaches the token sequence to generate, while contrastive learning separates it from plausible alternatives under the same question and memory. Sharing the source context emphasizes which fact answers the question. Negative candidates are used only in training; open-ended inference generates answers directly.

Episodic updates across memories. At outer iteration k, we select a source context and initialize a temporary reader at ψ^k. With θ and ϕ_d fixed, we take J AdamW steps on minibatches from $\mathcal{Q}_d$ using Equation 4, resetting optimizer state for each episode. Denoting these steps by U_d^J, Reptile updates the shared reader as

$$\psi_d' = U_d^J(\psi^k), \qquad \psi^{k+1} = \psi^k + \epsilon_k\big(\psi_d' - \psi^k\big), \tag{5}$$

where ϵ_k is the outer step size. Interpolation operates on the reader's LoRA factors A and B, without differentiating through the inner optimization trajectory. The next episode starts from the updated shared reader.

Temporary adaptations occur only during training; deployment retains the final shared parameters $\widehat{\psi}$. Section 5.4 compares Reptile with direct optimization under both objectives.

3.4 Deployment on Unseen Contexts

Given an unseen context C_*, we first construct $\phi_* = \text{Write}_\theta(C_*; \phi_0)$ from its raw text. We then activate ϕ_* together with $\widehat{\psi}$ and generate answers autoregressively from $p_{\theta,\phi_*,\widehat{\psi}}(y \mid q)$. The prompt contains the question and shared task formatting. The original context is represented by its memory adapter, and all parameters remain fixed during question answering.

Subsequent queries reuse this memory–reader pair, amortizing per-context writing cost; reader training is shared across contexts.

4 Training Data and Procedure

4.1 Context Construction

We construct Adapted Public QA (APQA) from SQuAD 2.0 (Rajpurkar et al., 2018), MuSiQue (Trivedi et al., 2022), DROP (Dua et al., 2019), DuoRC (Saha et al., 2018), and Qasper (Dasigi et al., 2021). These sources provide contexts with varied content and questions about their facts and relations.

Familiar names can evoke facts already present in the base model; entity replacement creates new associations for the reader to recover. In a single LLM request, we introduce fictional names and replacement dates consistently throughout each context and its QA collection, preserving coreference, factual relations, and the quantities needed to answer questions. Figure 15 in Appendix G.3 illustrates the transformation; Section 5.4 evaluates its effect.

We check the generated record structure and extractive answer grounding, and flag duplicate questions. Source data is split by complete contexts, with each context and its QA collection assigned together to training or heldout evaluation.

4.2 Reader Supervision Construction

QA construction accompanies entity replacement. We transform usable source QA pairs and generate additional questions about the transformed context, covering fact recall, entity and relation binding, paraphrases, and comparisons. Each question and reference answer is paired with the memory written from that context. This provides multiple ways to query the information encoded in a single memory.

The QA collection also includes boundary questions whose premises are absent or false in the context. Their reference answer is "Not stated." These examples teach the reader to distinguish supported answers from requests that the memory cannot resolve.

For contrastive learning, we construct competing answers from the same context's QA pool. We select answers associated with lexically similar QA pairs and make substitutions using entities, dates, or numbers of the same type. These candidates draw on the context's own vocabulary and facts, encouraging the reader to associate each question with the appropriate answer. Boundary QA supplies a reference response to an unsupported question; negative candidates provide alternatives against which each reference is scored in Equation 4.

4.3 Training Procedure

We precompute a frozen memory bank using Section 3.2 and pair each memory with its QA collection. Reader training follows the episodic procedure in Section 3.3; neither source memories nor the backbone are updated. Appendix F reports the Qwen3-8B training exposure and update budget.

5 Experiments

We evaluate transfer to unseen contexts, then test memory use, reader training, and deployment costs.

5.1 Experimental Setup

We train on APQA (Section 4) and evaluate on unseen computer-paper summaries (CP; Appendix G.1), entity-replaced CUAD contracts (Hendrycks et al., 2021) and RepliQA content (Monteiro et al., 2024), plus original-entity ROPES (Lin et al., 2019) and QuAIL (Rogers et al., 2020). Released GenerativeAdapter (Chen et al., 2024), Doc-to-LoRA (Charakorn et al., 2026), and SHINE (Liu et al., 2026) checkpoints are compared with LoRA Readout on matched backbones and questions. We use normalized-token ROUGE-L F1 for CP, CUAD, RepliQA, and ROPES, and option accuracy for QuAIL. Scores are averaged over questions within each dataset; Avg. reports the unweighted mean of the five dataset scores. New contexts require only raw-text writing, without reader adaptation.

We use natural-document and synthetic counterfactual diagnostics to test how the reader accesses information written by NTP. They examine whether better document fitting improves question answering, whether gains depend on the attached memory beyond general QA adaptation, and whether interventions on memory signals change answer selection and generation (Appendix A.2).

5.2 Effectiveness on Unseen Contexts

LoRA Readout scores highest on each dataset within each backbone group (Table 1), exceeding GenerativeAdapter, Doc-to-LoRA, and SHINE by 0.092, 0.191, and 0.177 on average across the five datasets.

The ROPES and QuAIL gains extend these results to original-entity contexts and multiple-choice answers, complementing CP summaries and entity-replaced CUAD and RepliQA. The following controls test whether these gains depend on the written memories.

Method	CP	CUAD	RepliQA	ROPES	QuAIL (Acc.)	Avg.
Mistral-7B-Instruct-v0.2						
Base LM (question only)	0.127	0.095	0.080	0.086	0.360	0.149
GenerativeAdapter	0.272	0.208	0.204	0.543	0.400	0.326
LoRA Readout	**0.301**	**0.337**	**0.359**	**0.578**	**0.513**	**0.418**
Qwen3-4B-Instruct-2507						
Base LM (question only)	0.164	0.209	0.229	0.076	0.292	0.194
Doc-to-LoRA	0.361	0.190	0.147	0.082	0.495	0.255
LoRA Readout	**0.372**	**0.417**	**0.449**	**0.490**	**0.501**	**0.446**
Qwen3-8B						
Base LM (question only)	0.254	0.251	0.268	0.092	0.346	0.242
SHINE	0.190	0.151	0.149	0.373	0.507	0.274
LoRA Readout	**0.357**	**0.387**	**0.410**	**0.573**	**0.526**	**0.451**

Table 1: Context-free inference performance: ROUGE-L F1 on CP, CUAD, RepliQA, and ROPES; option accuracy on QuAIL (all scores on a 0–1 scale). QuAIL is multiple choice: accuracy directly measures correct option selection, whereas text overlap can give partial credit to an incorrect option. Bold marks the best score within each backbone group. Avg. equally weights the five dataset scores. Unparsed or contradictory QuAIL outputs count as incorrect.

Configuration	CP	CUAD	RepliQA	ROPES	QuAIL (Acc.)
Base LM: question + context	0.604	0.673	0.685	0.762	0.715
Reader only	0.260	0.326	0.318	0.271	0.287
Correct memory only	0.300	0.239	0.250	0.248	0.232
Correct memory + reader	**0.357**	**0.387**	**0.410**	**0.573**	**0.526**
Wrong memory + reader	0.241	0.300	0.321	0.245	0.299

Table 2: Memory dependence on Qwen3-8B. Evaluation sets and metrics follow Table 1 (QuAIL: accuracy of multiple-choice option selection). Only the first row includes context tokens. Bold marks the best context-free score; additional controls appear in Appendix G.

5.3 Understanding Access to NTP Memories

Document fit versus query access. The writing-budget diagnostic separates fitting document continuations from accessing their contents under question prompts. Increasing writing from 6 to 20 updates lowers median document NTP loss, while memory-only token F1 falls from 24.86% to 15.85%. This separation motivates learning a shared reader to access the written information. Appendix A.3 reports the loss distribution and writing-budget comparisons.

Dependence on the attached memory. The reader improves QA with frozen memories on all five datasets (Table 2). Replacing the correct memory lowers ROUGE-L by 0.115, 0.088, 0.089, and 0.328 on CP, CUAD, RepliQA, and ROPES, and QuAIL accuracy by 0.227.

Correct memory plus the reader also outperforms either component alone on ROPES and QuAIL, supporting context-specific memory use in both answer generation and option selection. The full-context reference quantifies the remaining quality gap; Section 5.5 examines execution-cost savings.

Beyond general QA adaptation. Attaching saved 20-update memories improves a task-only adapter by 6.85 natural-document F1 points and the memory-trained reader by 14.01 (Figure 3a). The difference is 7.17 points $[1.61, 12.71]$, supporting an additional benefit from training with memories. The comparison matches capacity, gold-QA exposure and order, and update counts. It retains the historical reader; remaining differences in initialization and negative draws are documented in Appendix B.4.

Intervening on fact-relevant signals. Effective-memory interpolation between synthetic factual variants shifts candidate preference toward the corresponding answer (Figure 3b): 44 of 48 trajectories are nondecreasing with the reader, versus 25 without it. Disabling late-MLP memory branches

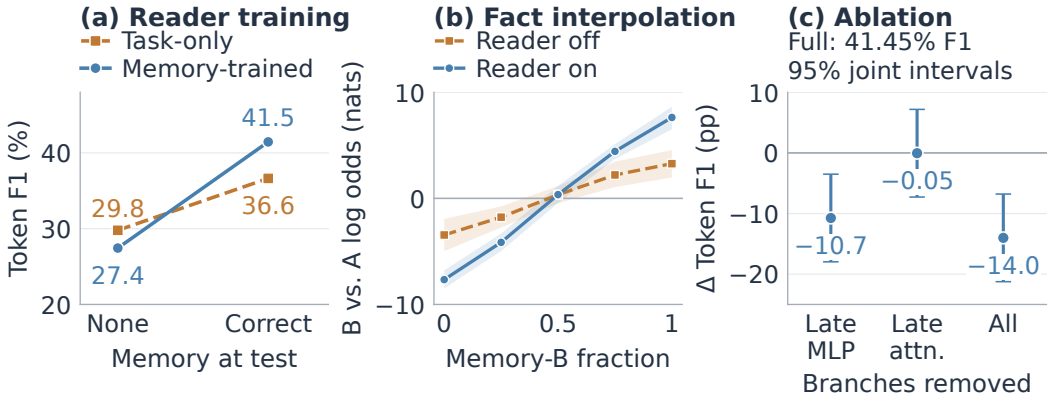


Figure 3: Access to 20-update NTP memories. (a) Natural-document F1 with/without memory; lines connect point estimates. (b) Synthetic B-versus-A candidate log odds under effective-memory interpolation, with pointwise 95% pair-cluster intervals. (c) Natural-generation F1 changes after disabling memory branches (late: layers 24–35), with simultaneous 95% document-cluster intervals. Intervals condition on fixed readers; the panels distinguish candidate preference from generated-answer quality.

Update rule	Objective	CP	CUAD	RepliQA	ROPES	QuAIL (Acc.)
Direct	NLL	0.332	0.353	0.373	0.413	0.338
Direct	NLL + contrastive	0.353	0.361	0.395	0.439	0.392
Reptile	NLL	0.348	0.375	0.395	0.440	0.371
Reptile	NLL + contrastive	**0.357**	**0.387**	**0.410**	**0.573**	**0.526**

Table 3: Reader-training configurations on Qwen3-8B. Evaluation sets and metrics follow Table 1 (QuAIL: accuracy of multiple-choice option selection). Bold marks the best score per dataset.

reduces natural-generation F1 by 10.74 points $[-17.97, -3.51]$, versus -0.05 $[-7.28, 7.18]$ for late attention (panel c). These interventions connect fact-relevant memory signals to answer generation and measure the effects of disabling specific layer groups (Appendix B).

Scope of the access evidence. A fixed memory branch $m_\ell(h) = \Delta W_\ell^{\text{mem}} h$ can respond to reader-induced changes in its input. Fixed-output interventions support this access pathway on synthetic probes; its contribution to natural-QA gains remains to be quantified (Appendix A.5). In the updated natural counterfactual evaluation, the trained reader follows both factual variants on 320/1,000 pairs, compared with 125/1,000 for memory alone and 112/1,000 for the task-only reader (Appendix C). The trained reader also shifts candidate preference toward the edited fact by 34.89 nats (95% CI [31.71, 38.34]). Correct-memory benefits also depend on the evaluation population: the main QuAIL benchmark shows a gain, while the separate natural-document diagnostic finds no clear advantage. Synthetic probes further distinguish writing budgets and scoring criteria: fact presence decreases with the reader at 6 updates, while strict matching improves at 20 updates (Appendices A and D).

5.4 Reader Training Design

Table 3 compares update rules and objectives with the same backbone, source QA pool, and frozen memories.

Reptile improves over direct optimization with either objective, and contrastive supervision improves both update rules across all five datasets. Their combination achieves the highest scores.

Because these configurations use identical memories, the comparison holds the stored context parameters fixed. Reptile groups updates by memory, while contrastive supervision distinguishes com-

peting answers within it. The ROPES and QuAIL gains extend to both answer generation and option selection.

Entity replacement. Training on replaced rather than original entities raises ROUGE-L from 0.396 to 0.436 on the same entity-replaced APQA holdout. This comparison supports entity replacement for transfer under the evaluated transformation protocol; Appendix G.3 discusses its relationship to training–evaluation distribution alignment.

5.5 Writing and Inference Costs

Training budget. Source-memory writing and reader training use 33.75M cumulative input-token presentations on Qwen3-8B. The reader-training stage takes approximately 3.1 hours on one RTX 5090; source-memory construction and data preparation are separate stages. Appendix F specifies the accounting scope for this run and for the published baseline budgets.

This count includes repeated NTP passes and complete reader input sequences, including prompts and contrastive negatives. The source-memory bank is constructed once and reused across reader episodes. Writing memories for target contexts is a separate deployment cost, accounted for in the inference analysis below.

Writing and inference. Moving context processing offline increases repeated-query throughput in the synthetic timing benchmark. On an RTX PRO 6000 Blackwell, throughput reaches $6.6\times$ the full-context reference at 32k tokens, and writing plus 128 queries takes $0.31\times$ the full-context time (Figure 4). Under fresh prefill without cross-query prefix reuse, savings recover writing cost after 25–32 queries. The sweep measures execution cost at fixed output length; QA quality is evaluated separately in Section 5.2. Appendix E specifies the timing protocol and included costs.

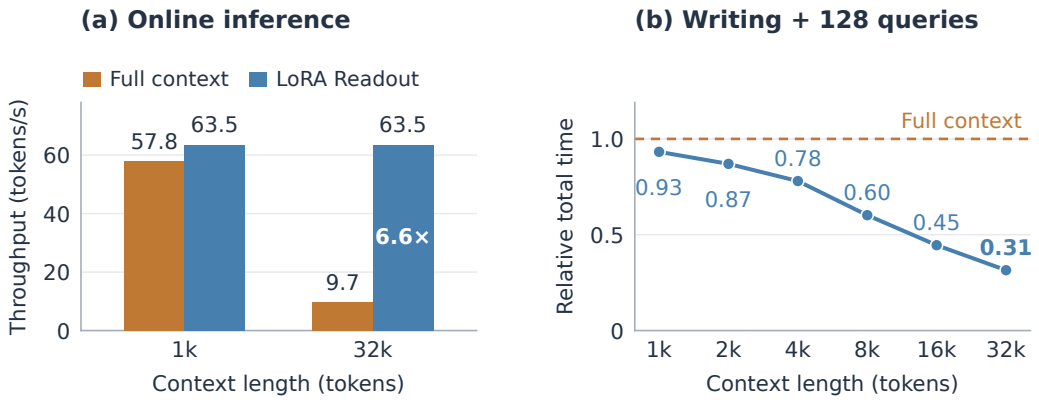


Figure 4: Inference costs on Qwen3-8B with synthetic contexts and 32-token outputs. (a) Online throughput at 1k and 32k. (b) Writing-plus-query time relative to full-context inference for 128 queries; lower is better. Queries use merged weights and fresh prefill without cross-query prefix reuse. Ratios include writing and exclude adapter merging and model loading.

6 Conclusion

LoRA Readout pairs independently written NTP memories with a shared reader trained across frozen memories. New contexts require only raw-text memory writing; the reader is reused without adaptation, and question answering omits the original context tokens.

The method improves over the evaluated parametric context-compression checkpoints. Controls and interventions support context-specific memory use beyond general QA adaptation, but benefits depend on the writing regime and task. Document fit, candidate preference, and correct generation remain distinct. Repeated queries can amortize writing cost while reducing online computation; extending reliable access across these conditions remains an important direction.

AI USE STATEMENT

We used generative AI tools to assist with data generation and manuscript writing. The data-related use concerns the construction of question–answering resources for LoRA Readout, while the writing-related use concerns the preparation and revision of the manuscript. We describe these uses and the associated validation procedures below.

Data generation. We used large language models to transform public question–answering resources into the adapted contexts and supervision described in the paper. For APQA construction, a model receives a source context and its associated questions, introduces fictional entity names and replacement dates, and produces the transformed context together with its question–answer collection. The generation instructions require consistent replacements across the context, questions, and answers, with the aim of preserving coreference, factual relations, and answer-relevant quantities. Models also assist in generating additional questions and reference answers grounded in the transformed context, including questions whose premises are unsupported and whose reference response is "Not stated." The resulting resources provide supervision for learning to read context-specific memories. AI-assisted construction also applies to the entity-replaced evaluation resources identified in the experimental setup.

Data validation. Generated records are subjected to programmatic checks for the expected structure and counts, document-length changes, extractive-answer presence, and label consistency. Duplicate questions are flagged, and the source-data materialization pipeline accepts records that pass its required validation checks. Each source context and its associated question–answer collection are assigned together to training or heldout evaluation. In addition to these automated checks, the authors manually reviewed all AI-generated data, checking the consistency of entity and date replacements, the correspondence between questions and reference answers, and the support for answers in the associated contexts. Questions labeled as unsupported were also checked against their contexts.

The paper distinguishes the use of generated QA supervision for reader training from memory writing, which derives its next-token-prediction targets from the context text. Writing a memory for a new context does not require generating additional QA supervision.

Manuscript writing. Generative AI tools assisted with writing and revising the manuscript, including rephrasing text and improving clarity, readability, and organization. The authors manually reviewed all AI-assisted text and retained control over the final wording and scientific content. Technical descriptions and claims were checked against the methods and experimental evidence, and citations were checked against the referenced sources. The authors reviewed the resulting manuscript for factual accuracy, consistency, and appropriate attribution.

Author responsibility. The authors take full responsibility for the final submission, including the generated data, written text, scientific claims, and other artifacts prepared with AI assistance. All AI-assisted materials were subject to human review, and the authors remain accountable for the accuracy, originality, and integrity of the work and for addressing any errors or inconsistencies.

REFERENCES

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of Language Models: Part 3.1, Knowledge Storage and Extraction, 2023. URL https://arxiv.org/abs/2309.14316.

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", 2023. URL https://arxiv.org/abs/2309.12288.

Lucas Caccia, Alan Ansell, Edoardo Ponti, Ivan Vulić, and Alessandro Sordoni. Training Plug-and-Play Knowledge Modules with Deep Context Distillation, 2025. URL https://arxiv.org/abs/2503.08727v4.

Rujikorn Charakorn, Edoardo Cetin, Shinnosuke Uesaka, and Robert Tjarko Lange. Doc-to-LoRA: Learning to Instantly Internalize Contexts, 2026. URL https://arxiv.org/abs/2602.15902.

Tong Chen, Hao Fang, Patrick Xia, Xiaodong Liu, Benjamin Van Durme, Luke Zettlemoyer, Jianfeng Gao, and Hao Cheng. Generative Adapter: Contextualizing Language Models in Parameters with A Single Forward Pass, 2024. URL https://arxiv.org/abs/2411.05877.

Xin Chen, Junchao Wu, Shu Yang, Runzhe Zhan, Zeyu Wu, Min Yang, Shujian Huang, Lidia S. Chao, and Derek F. Wong. Neuron-aware data selection in instruction tuning for large language models. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/pdf?id=uq6UWRgzMr.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4599–4610, 2021. doi: 10.18653/v1/2021.naacl-main.365.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2368–2378, 2019. doi: 10.18653/v1/N19-1246.

Dan Hendrycks, Collin Burns, Anya Chen, and Spencer Ball. CUAD: An expert-annotated NLP dataset for legal contract review, 2021. URL https://arxiv.org/abs/2103.06268.

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.

Nathan Hu, Eric Mitchell, Christopher Manning, and Chelsea Finn. Meta-Learning Online Adaptation of Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4418–4432, 2023. doi: 10.18653/v1/2023.emnlp-main.268. URL https://aclanthology.org/2023.emnlp-main.268/.

Zhengbao Jiang, Zhiqing Sun, Weijia Shi, Pedro Rodriguez, Chunting Zhou, Graham Neubig, Xi Victoria Lin, Wen tau Yih, and Srinivasan Iyer. Instruction-tuned Language Models are Better Knowledge Learners, 2024. URL https://arxiv.org/abs/2402.12847.

Amirhossein Kazemnejad, Mehdi Rezagholizadeh, Prasanna Parthasarathi, and Sarath Chandar. Measuring the Knowledge Acquisition-Utilization Gap in Pretrained Language Models, 2023. URL https://arxiv.org/abs/2305.14775.

Ouail Kitouni, Niklas Nolte, Diane Bouchacourt, Adina Williams, Mike Rabbat, and Mark Ibrahim. The Factorization Curse: Which Tokens You Predict Underlie the Reversal Curse and More, 2024. URL https://arxiv.org/abs/2406.05183.

Kevin Lin, Oyvind Tafjord, Peter Clark, and Matt Gardner. Reasoning over paragraph effects in situations. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pp. 58–62. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-5808. URL https://aclanthology.org/D19-5808/.

Yewei Liu, Xiyuan Wang, Yansheng Mao, Yoav Gelbery, Haggai Maron, and Muhan Zhang. SHINE: A Scalable In-Context Hypernetwork for Mapping Context to LoRA in a Single Pass, 2026. URL https://arxiv.org/abs/2602.06358.

João Monteiro, Pierre-André Noël, Étienne Marcotte, Sai Rajeswar, Valentina Zantedeschi, David Vázquez, Nicolas Chapados, Christopher Pal, and Perouz Taslakian. RepLiQA: A question-answering dataset for benchmarking LLMs on unseen reference content. In *Advances in Neural Information Processing Systems*, volume 37, pp. 24242–24276, 2024. doi: 10.52202/079017-0764.

Alex Nichol, Joshua Achiam, and John Schulman. On First-Order Meta-Learning Algorithms, 2018. URL https://arxiv.org/abs/1803.02999.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 784–789, 2018. doi: 10.18653/v1/P18-2124.

Anna Rogers, Olga Kovaleva, Matthew Downey, and Anna Rumshisky. Getting closer to AI complete question answering: A set of prerequisite real tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8722–8731, 2020. doi: 10.1609/aaai.v34i05.6398. URL https://ojs.aaai.org/index.php/AAAI/article/view/6398.

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. DuoRC: Towards complex language understanding with paraphrased reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1683–1693, 2018. doi: 10.18653/v1/P18-1156.

Kuniaki Saito, Kihyuk Sohn, Chen-Yu Lee, and Yoshitaka Ushiku. Where is the answer? Investigating Positional Bias in Language Model Knowledge Extraction, 2024. URL https://arxiv.org/abs/2402.12170.

Sijun Tan, Xiuyu Li, Shishir G Patil, Ziyang Wu, Tianjun Zhang, Kurt Keutzer, Joseph E. Gonzalez, and Raluca Ada Popa. LLoCO: Learning long contexts offline. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 17605–17621, 2024. doi: 10.18653/v1/2024.emnlp-main.975. URL https://aclanthology.org/2024.emnlp-main.975/.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022. doi: 10.1162/tacl_a_00475.

Zitong Yang, Neil Band, Shuangping Li, Emmanuel Candès, and Tatsunori Hashimoto. Synthetic continued pretraining, 2024. URL https://arxiv.org/abs/2409.07431.

Ying Zhang, Benjamin Heinzerling, Dongyuan Li, and Kentaro Inui. Understanding Fact Recall in Language Models: Why Two-Stage Training Encourages Memorization but Mixed Training Teaches Knowledge, 2025. URL https://arxiv.org/abs/2505.16178.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. LIMA: Less Is More for Alignment, 2023. URL https://arxiv.org/abs/2305.11206.

Yuanchang Zhou, Siyu Hu, Xiangyu Zhang, Hongyu Wang, Guangming Tan, and Weile Jia. MatRIS: Toward reliable and efficient pretrained machine learning interatomic potentials. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/8cd1080c50e15e72414666c2f7637506-Abstract-Conference.html.

Chunsheng Zuo, Liaoyaqi Wang, William Jurayj, William Fleshman, and Benjamin Van Durme. Rethinking LoRA Memory Through the Lens of KV Cache Compression, 2026. URL https://arxiv.org/abs/2606.05698.

A When Can a Reader Use NTP Memories?

This appendix separates three questions: whether raw-text next-token prediction (NTP) fits a document, whether its memory parameters carry answer-relevant information, and whether a shared reader makes that information useful in question answering. These properties need not move together. We consolidate controlled natural-document experiments and deterministic synthetic interventions, and add a continuous counterfactual memory-output intervention. The latter is an exploratory extension motivated by the preceding localization results, rather than an independent confirmatory test.

A.1 Reconciling Our Findings with Prior NTP Results

The perplexity curse describes low language-modeling loss coexisting with weak question answering; it does not establish that NTP stores no usable information (Jiang et al., 2024). Acquisition–utilization studies similarly distinguish learning information from accessing it through a downstream query (Kazemnejad et al., 2023; Allen-Zhu & Li, 2023). NTP conditions on the preceding document tokens, whereas context-free QA conditions on a question. Positional bias and reversal/factorization results show that changing this conditioning can limit retrieval (Saito et al., 2024; Berglund et al., 2023; Kitouni et al., 2024). These findings motivate testing access to a frozen memory, rather than using document loss alone as a proxy for QA performance.

Comparison with LLoCO. LLoCO shares the goal of moving context processing offline and learning to use the resulting representation (Tan et al., 2024). It uses AutoCompressor to cache continuous document-summary embeddings, then instruction-tunes a LoRA adapter for a document group to answer questions conditioned on those embeddings. Its serving pipeline retrieves the compressed document representation and the associated adapter; summary embeddings remain in the model's online input. LoRA Readout stores each context in an independently NTP-written memory LoRA. At deployment, the shared reader is reused with that memory, and the prompt contains the question and shared formatting, without document-summary embeddings. The methods therefore share a learned-access motivation while using different memory representations, reader-training procedures, and inference inputs.

Comparison with Deep Context Distillation. DCD is a close architectural precedent: it already trains a shared knowledge extractor (KE) over frozen document-specific LoRA knowledge modules and reports an NTP-memory-plus-KE baseline (Caccia et al., 2025). Document DCD writes a module by matching a context-conditioned teacher's output distributions with KL divergence and normalized hidden states with an L_1 loss. Its QA-trained KE transfers to unseen document modules, with learned layer-specific weights combining module and extractor updates. LoRA Readout instead studies a raw-text NTP writer without teacher targets, an episodic Reptile reader trained with answer NLL and contrastive ranking, and additive memory–reader composition (Equations 1–5). Its focus is the writing and reader-learning recipe for accessing independently written NTP memories. DCD's Table 1 is summarized in Table 4. Thus, a missing reader cannot explain the difference between that study and ours. Adding KE helps LM memory relative to LM memory alone in these settings, but the gain over KE alone is small or negative; document distillation remains stronger. Their conclusion that NTP is comparatively weak is compatible with these results.

Backbone / dataset	LM	KE	LM + KE	Doc. DCD + KE
Phi-3 / NarrativeQA	15.2	19.6	20.7	28.0
Phi-3 / QuALITY	26.4	39.4	36.0	40.4
Llama-3.1 / NarrativeQA	17.9	20.9	22.3	31.2
Llama-3.1 / QuALITY	27.0	39.3	40.9	45.8

Table 4: Published closed-book results from Caccia et al. (2025), Table 1 (arXiv v4). NarrativeQA uses ROUGE-L; QuALITY uses accuracy, both on a 0–100 scale. These are literature results, not our reruns or matched comparisons with LoRA Readout. LM denotes raw-document language-model training.

Our question is therefore conditional: can a reader trained across NTP memories exploit information that direct QA does not reliably express? We test memory identity, writing budget, task-only adaptation, and internal memory signals below. DCD's layerwise learned combination, reader objective, backbones, adapter capacities, documents, and training budgets differ from ours. Our experiments establish behavior within our setting; they do not isolate which of these cross-paper differences causes the performance gap. In particular, the task-only controls below are not a reproduction of DCD's memory-conditioned KE. These controls also do not independently identify the contributions of Reptile and contrastive learning.

Property	DCD	LLoCO	LoRA Readout
Document state	Document LoRA module	Cached summary embeddings	NTP memory LoRA
Writing	Teacher logit/hidden-state distillation	AutoCompressor forward encoding	Raw-text NTP optimization
Learned access	Shared QA-trained KE	Instruction-tuned LoRA for a document group	Shared episodic reader; NLL + contrastive
Composition	Learned layerwise module/KE weights	Summary embeddings with the adapted decoder	Additive memory/reader updates
Online input	Question in the closed-book setting	Question + summary embeddings	Question with shared formatting

Table 5: Method-level comparison with DCD (Caccia et al., 2025) and LLoCO (Tan et al., 2024). The DCD writing row describes its document-distillation method; its separate LM baseline uses raw-document language modeling. This table compares designs, not scores or costs under a matched experimental protocol.

A.2 Experimental Protocol and Uncertainty

We fix the Qwen3-8B backbone and the July 24 reader checkpoint trained for 6,000 episodes with Reptile, NLL, and contrastive learning. Natural-document diagnostics contain 64 documents: 16 each from NewsQA-MRQA, StreamingQA-MultiQA, QuAIL, and ROPES, with three original questions per document. Documents and questions are selected by fixed hash ranks without using model outcomes. Exact question overlap with the reader's source QA is zero; this does not establish absence from backbone pretraining. This diagnostic population is distinct from the five-dataset main benchmark, including its separate ROPES and QuAIL evaluation sets.

The raw-text writer uses rank-32/alpha-64 memory LoRA, AdamW with learning rate 10^{-3}, batch size one, gradient clipping at one, and BF16 arithmetic. We retain memories after 6 and 20 updates under initialization seeds 42, 43, and 44. The rank-64/alpha-128 reader is frozen at evaluation. All question-only conditions share the historical IM4 chat prompt; ICL supplies the original document. Generation is greedy with a 96-token cap. Natural-document scores use maximum-reference token F1 and normalized exact match. An isolated QuAIL option letter is resolved to its option text for scoring. These F1 values are not the main table's ROUGE-L.

The synthetic study contains 24 counterfactual document pairs, six each for person, city, color, and number facts. A pair differs in one factual value. Both variants and two fixed questions yield 96 generation items per condition. Candidate probes compare two single-token answers before any answer is supplied. The continuous intervention uses the first question and both directions of all 24 pairs, without selecting pairs by correctness. Deterministic writing was checked by an identical repeat, and saved memories are reused.

Uncertainty is conditional on the fixed readers and diagnostic populations. We use 4,000 bootstrap resamples, stratified by source or fact family, with documents or counterfactual pairs as independent clusters. Questions, memory seeds, and opposite directions remain within their cluster. Existing analyses use seed 20260921; the added dose/loss analyses use 20260923. Intervals are pointwise 95% intervals without multiplicity adjustment. Three memory seeds do not constitute three reader-training replications.

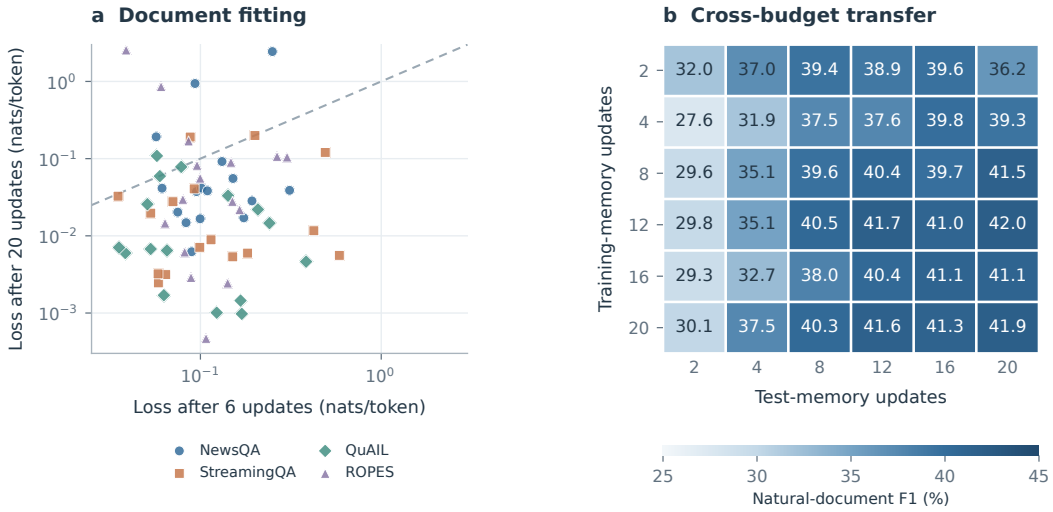


Figure 5: Writing budget and access. (a) Each point is a natural document, with post-update NTP loss averaged over three memory seeds. Points below the diagonal improve with additional writing; both axes are logarithmic. The exceptions explain why the overall mean loss does not decrease. (b) Natural-document token F1 across reader-training and test-memory budgets. Each row uses a separately trained reader and each cell contains 576 predictions. Cells reuse the same documents and are not independent replications.

A.3 Writing Loss and Answer Accessibility Are Distinct

After averaging the three memory seeds within each natural document, 56 of 64 documents have lower post-update NTP loss at 20 updates than at 6. The median document loss falls from 0.0975 to 0.0238 nats/token. However, the mean changes from 0.1368 to 0.1438 because some writers become less stable; we therefore do not claim a uniform decrease in loss (Figure 5a). Memory-only QA F1 drops from 24.86 to 15.85, a paired difference of -9.01 percentage points $[-14.53, -3.67]$. With the frozen reader, F1 is 38.22 and 41.45, respectively; the difference, 3.23 points $[-1.35, 7.51]$, is inconclusive. Increasing the writing budget is therefore not a monotonic route to better direct QA, and low training loss alone is insufficient to establish access.

An existing 6×6 scan trains separate readers with source-memory budgets of 2, 4, 8, 12, 16, or 20 updates and evaluates each across the same six test-memory budgets (Figure 5b). Each reader is trained for 6,000 episodes; each cell contains the same 192 questions and three memory seeds. The 12/20 and 20/20 configurations reach 41.98 and 41.93 F1, with paired difference 0.06 points $[-2.00, 2.04]$. These are descriptive high-scoring cells, not independently selected optima. Exact training/test budget matching is not necessary, although the 2/20 configuration scores 36.23. This sensitivity supports studying the compatibility between writing and reading; it does not identify a unique internal mechanism.

A.4 Memory Dependence Beyond General QA Adaptation

Table 6 shows that the 20-update memory and reader combination exceeds memory alone by 25.60 F1 points $[21.09, 30.18]$. It also exceeds reader alone by 14.01 points $[6.81, 21.22]$ and a same-source mismatched memory by 13.31 points $[7.37, 19.11]$. Mismatches use the next document in a fixed source-specific order. These controls show that the aggregate gain depends on the corresponding memory, not just the presence of an additional adapter.

We further reuse two completed controls trained with memory inactive: direct answer CE and Reptile with CE and contrastive learning. Both replay the historical gold QA draws in order, matching 48,000 optimizer updates, 96,000 gold QA exposures, source pool, and adapter capacity. They are evaluated with the same saved test memories as the reader. LoRA Readout exceeds these controls at 20 writing updates by 20.59 points $[14.80, 26.44]$ and 4.82 points $[1.66, 8.07]$, respectively. At 6

Condition	Token F1 (%)	Exact match (%)
Base, question only	10.90	3.65
Reader only	27.44	16.67
Memory only, 6 updates	24.86	11.81
Memory + reader, 6 updates	38.22	23.44
Memory only, 20 updates	15.85	3.65
Memory + reader, 20 updates	**41.45**	**29.34**
Mismatched memory + reader, 20 updates	28.14	17.01
Task-only CE + memory, 20 updates	20.87	11.81
Task-only Reptile + memory, 20 updates	36.64	23.44
Original-context ICL	79.28	54.69

Table 6: Natural-document diagnostics, macro-averaged over sources and documents. Task-only controls are trained without memory, then combined with the saved NTP memories at evaluation. Paired uncertainty is reported in the text.

writing updates the advantage over task-only Reptile is only 1.38 points $[-2.39, 4.92]$. Thus, general QA adaptation explains a substantial part of the improvement, while training with memories provides an additional benefit at 20 updates. The controls share a new initialization; the historical reader initialization and historical negative-answer draws are not fully reconstructable. Direct CE also requires less computation, so this is not a fully initialization-, negative-sample-, or FLOP-matched experiment.

Memory dependence is heterogeneous (Figure 6c). Correct versus mismatched memory improves F1 by 17.84 points $[9.49, 27.05]$ on NewsQA and 24.17 $[12.90, 34.76]$ on StreamingQA. QuAIL gives -2.63 $[-12.62, 6.63]$, and ROPES gives 13.88 $[-2.88, 30.46]$. A post hoc QuAIL option-accuracy diagnostic also finds no correct-memory advantage: 47.22% versus 47.92%. We do not generalize the aggregate memory dependence to every source. All these context-free conditions remain well below the original-context ICL reference.

A.5 Counterfactual Interventions on the Memory Signal

Let a_r and a_d denote recipient and donor answers. For an intervention, we measure the change in $\log p(a_d \mid q) - \log p(a_r \mid q)$ relative to the recipient's unmodified forward pass. A positive shift favors the donor fact. At 20 writing updates, swapping the complete memory weights produces a shift of 15.49 nats $[13.97, 16.81]$. Swapping only MLP memory weights in layers 24–35 (zero-indexed) produces 11.55 $[10.14, 12.88]$; replacing that group's cached prompt outputs gives 11.54 $[10.20, 12.83]$. Module/token-wise norm-matched random perturbations produce 0.32 $[0.02, 0.69]$. The latter is small but not statistically zero under this pointwise interval. The swaps show that memory parameters carry a fact-specific signal that affects answer preference. Group effects are non-additive; their ratio is not a fraction of causal mediation.

Continuous replacement. We extend the complete swaps to $\alpha \in \{0, .25, .5, .75, 1\}$. On the same question prompt, we cache memory-branch outputs m_r and m_d under the recipient and donor memories with reader active, and replace selected outputs by $(1 - \alpha)m_r + \alpha m_d$. We intervene on all memory branches or only the late MLP group. Every unselected branch retains recipient weights. Both directions of every pair are included, giving $24 \times 2 \times 2 \times 5 = 480$ scored forwards.

The mean donor-directed shifts increase with dose: 0.00, 3.63, 7.74, 11.85, and 15.49 nats for all branches, and 0.00, 2.61, 5.51, 8.57, and 11.54 for late MLP branches (Figure 6a). This average dose dependence is consistent with an answer-relevant signal carried through those outputs. It is not a claim that every pair is monotonic, that the intervention follows a natural training trajectory, or that candidate preference implies free-generation correctness. All 192 zero/unit-dose comparisons with independently saved baseline and output-swap records match exactly, including the localized zero-dose controls. The backbone and reader remain unchanged.

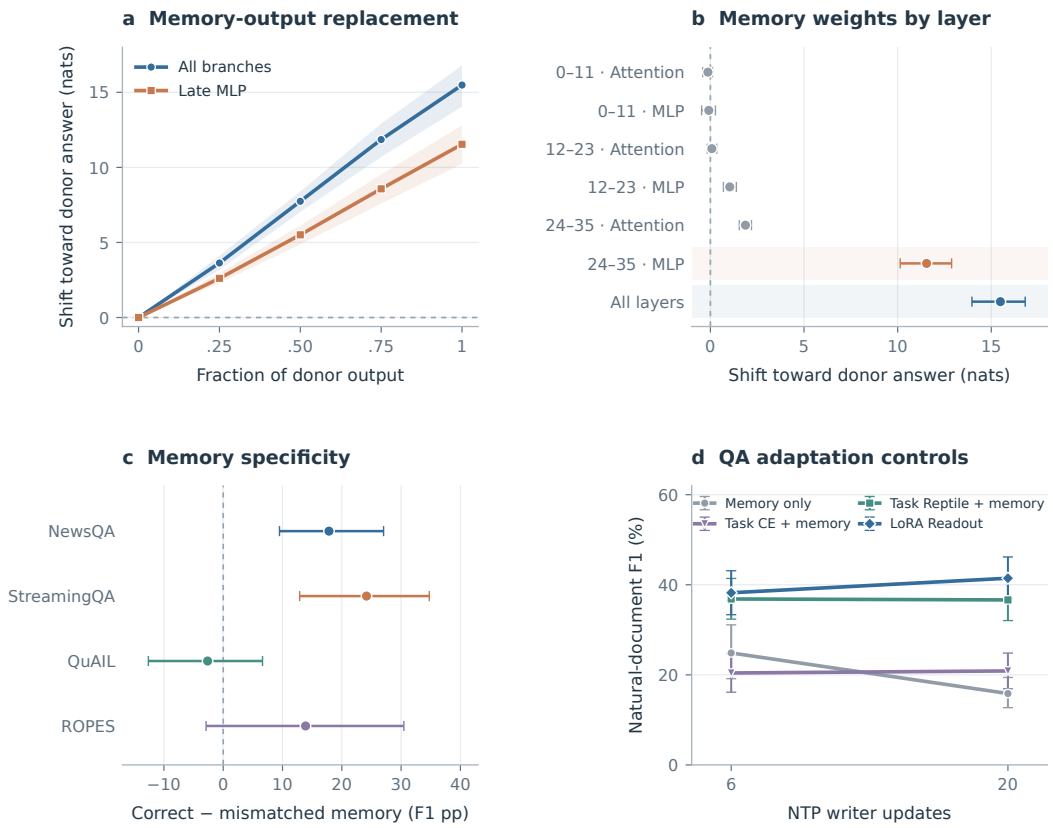


Figure 6: Memory identity, signal location, and adaptation controls. (a) Continuous output replacement across all 24 counterfactual pairs and both directions; bands are pointwise pair-cluster intervals. (b) Synthetic memory-weight swaps by layer group. (c) Natural-document correct-minus-mismatched-memory F1 at 20 writing updates. (d) Natural QA scores for memory only, task-only adaptations combined with memory, and the memory-trained reader. Error bars use pair clusters in (b) and source-stratified document clusters in (c,d); they do not reflect reader training-seed variation.

How can a frozen memory change its signal? At a linear memory branch, $m_\ell(h) = \Delta W_\ell^M h$. Even with ΔW_ℓ^M fixed, a reader can change upstream hidden states h and thus the memory output. We test this possibility by caching prompt memory outputs with reader off/on, and fixing those outputs in recipient forward passes with reader off/on. On synthetic pairs, the unmodified reader gain in correct-answer log-odds margin is 4.59 nats $[3.56, 5.63]$. When memory outputs are fixed to the reader-off cache, the gain is 1.09 $[0.37, 1.74]$. With recipient reader on, switching from reader-off to reader-on cached outputs gives 3.50 $[2.83, 4.15]$. This supports an interaction between the reader's computation and the signal produced by frozen memory branches, while leaving a residual effect outside that intervention.

The same interpretation is not established as the dominant mechanism on natural text. For full reference-answer mean log probability, the reader gain is 4.60 nats/token and remains 4.51 with reader-off prompt memory outputs fixed; changing those outputs gives only 0.089 $[-0.003, 0.194]$. Restricting scoring to the first answer token still leaves gains of 10.89 and 10.67 nats, respectively. Moreover, absolute log probability and two-candidate log odds are different estimands: on the same synthetic records, the correct candidate's absolute log-probability gain is 14.21 nats and remains 10.99 under fixed reader-off outputs. Relative odds remove changes shared by both candidates. These distributed interventions therefore support a bounded synthetic mechanism, not a universal decomposition of natural QA gains. General task adaptation, output formatting, and computation at subsequent answer positions can also contribute.

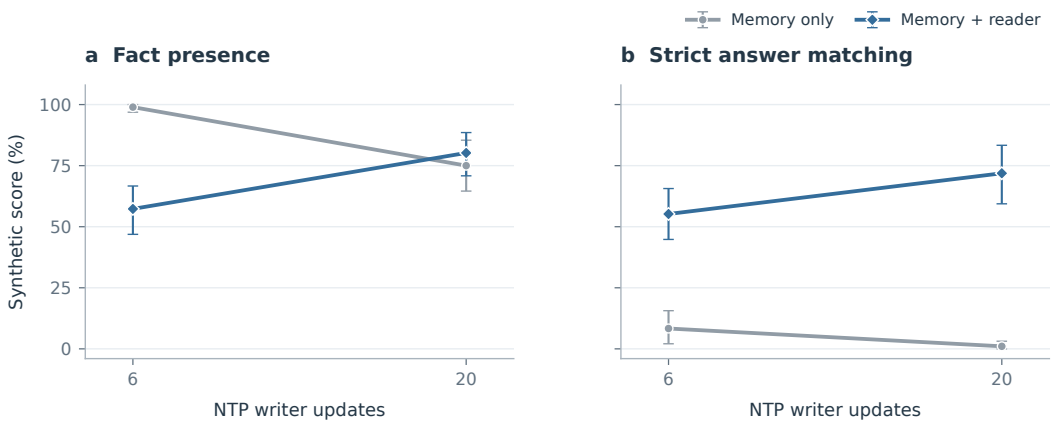


Figure 7: The apparent reader benefit depends on the evaluation question. (a) Rule-based fact presence in free generation can deteriorate with the reader, especially at six writing updates. (b) Strict short-answer matching improves substantially. Each condition contains 96 generation items; intervals resample 24 counterfactual pairs within four fact families.

A.6 Output Format and Failure Cases

Synthetic fact presence, strict short-answer matching, and two-candidate choice answer different questions. Fact presence requires the correct value and excludes the opposite value and predefined refusal phrases; it can count a fact inside a long recitation and is not a comprehensive semantic judge. At 6 writing updates, memory-only fact presence is 98.96%, versus 57.29% with the reader, a paired change of -41.67 points $[-53.12, -31.25]$. At 20 updates it changes from 75.00% to 80.21%, with inconclusive difference 5.21 $[-2.08, 14.58]$. In contrast, strict matching rises from 1.04% to 71.88% at 20 updates. The large exact-match improvement therefore includes a substantial formatting effect (Figure 7). The reader's two-candidate choice accuracy is 100% at 6 updates despite its much weaker free generation, further demonstrating why these metrics must remain separate.

Taken together, the experiments support a conditional account: raw NTP can encode answer-relevant information; a memory-trained reader can improve its use in natural QA beyond memory alone and matched-budget task-only adaptation; and counterfactual memory signals influence answer selection. They do not show that NTP always succeeds, that earlier NTP findings were incorrect, or that the reader uniformly improves factual correctness. A fully matched comparison with DCD, additional reader-training seeds, and broader counterfactual natural-text interventions are needed to attribute the cross-paper difference to a particular design choice.

B Further Controlled Memory Interventions

We extend the analysis in Appendix A with two experiments: interpolating the effective memory updates throughout generation, and removing memory branches during natural-document answer generation. These address whether candidate-preference interventions correspond to changes in generated content and whether the synthetic localization results extend to natural QA. We also report a supplementary training-control contrast and visualize the score dependence of the fixed-output analysis. These additions are exploratory, motivated by the completed diagnostic studies.

B.1 Protocol and Validation

We reuse the same frozen Qwen3-8B backbone, historical episode-6,000 reader, and deterministic saved memories as Appendix A.2. Synthetic interpolation covers all 24 document pairs, both question wordings, 6/20 writing steps, and reader off/on. Natural-document ablations cover all 64 documents, all three original questions per document, and all three writing-initialization seeds. Generation remains greedy with a 96-token cap and the historical question-only prompt. No new reader training or target-question adaptation is performed.

All added confidence intervals use 10,000 bootstrap resamples with seed 20260923, stratified by source or fact family. Documents or counterfactual pairs are the sampling units; their questions and writing seeds remain together. The three primary natural-ablation F1 contrasts use simultaneous intervals: the critical value is the 95th percentile of the maximum absolute deviation of the jointly bootstrapped mean contrasts. Other intervals are pointwise and exploratory. This uncertainty conditions on the fixed reader and does not measure reader-training seed variance.

Coverage is checked against every expected condition key, and generation scores are recomputed from raw predictions. Each interpolation endpoint is checked against the corresponding directly installed checkpoint and archived prediction. Natural full-memory and all-memory-disabled runs must reproduce the archived full-memory and reader-only conditions. Smoke runs are excluded from formal statistics.

B.2 Interpolating Effective Memories Throughout Generation

The cached-output intervention in Appendix A.5 holds selected signals to values recorded on a prompt. Here we instead mix the two saved memories' effective updates and let their inputs evolve with the intervened model. At each adapted projection, we apply

$$\Delta W_\ell(t) = (1-t)\Delta W_{\ell,A} + t\Delta W_{\ell,B}, \qquad t \in \{0, 0.25, 0.5, 0.75, 1\}. \tag{6}$$

Both branches receive the same current hidden state at every prompt and generated token. We mix their BF16 outputs in FP32 and cast the result back to BF16. Directly interpolating LoRA factors would introduce cross terms and implement a different intervention. The effective update can have rank up to 64; this probe is not a claim about the capacity or storage cost of an individually trained rank-32 memory.

For all 24 pairs and both question wordings, we evaluate five mixture values at 6/20 writing steps, with the reader disabled/enabled: 960 scored generations. We retain both single-token candidate probabilities and exclusive A/B fact mentions in generated text. Exclusive mentions exclude the alternative value and predefined abstention phrases. There is no uniquely correct fact for an interior mixture, so these curves are not labeled as mixture accuracy.

Figure 8 shows that the mean candidate preference tracks the effective memory mixture. With 20-step memories and the reader enabled, $\log p(B) - \log p(A)$ changes from -7.66 through -4.15, 0.36, and 4.43 to 7.63 nats. The paired endpoint shift is 15.29 nats [13.92, 16.53], compared with 6.72 nats [4.54, 8.92] without the reader. At six writing steps, endpoint shifts are similar in magnitude with and without the reader: 13.39 and 13.75 nats.

The average trend does not imply that every trajectory is monotonic. Across the 48 pair–question trajectories at twenty writing steps, 44 (91.7%) are nondecreasing across the five mixture values with the reader, compared with 25 (52.1%) without it. At six steps, the counts are 48 and 46, respectively. These descriptive rates reinforce the dependence of the reader's effect on the writing regime; they are not independent replications across reader training runs.

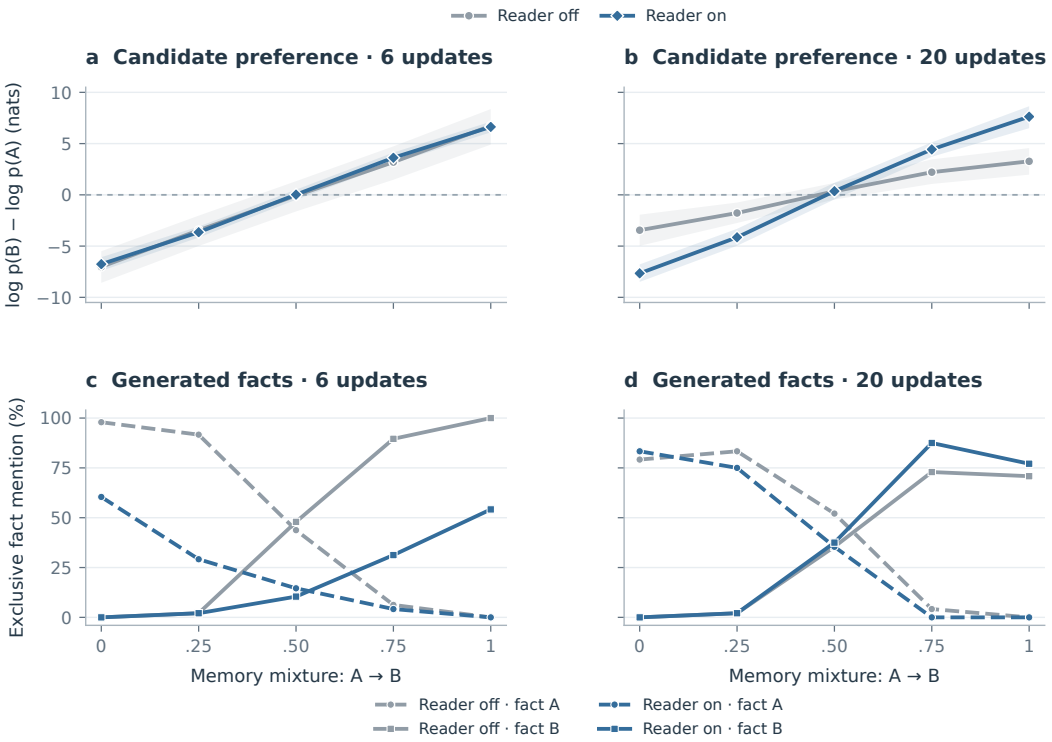


Figure 8: Effective-memory interpolation applied at every prompt and generated token. (a,b) B-versus-A candidate log odds, with pointwise 95% pair-cluster bootstrap bands. (c,d) Exclusive A/B fact-mention rates in free generation (means); the remaining outputs mention both/neither fact or contain an abstention phrase. Each point summarizes all 24 pairs and both questions. Interior mixtures have no uniquely correct answer.

Generation remains distinct from candidate preference. At the equal six-step mixture, other or abstaining outputs account for 75.0% with the reader and 8.3% without it, although both mean candidate margins are near zero. At twenty steps with the reader, the B-only mention rate is 87.5% at $t = 0.75$ and 77.1% at $t = 1$, even as the mean B-versus-A margin increases. These are descriptive generation patterns, not accuracy judgments for mixed memories. Thus effective memory content steers candidate preference and generated content, but does not imply monotonic improvement in generated answers. All 384 endpoint checks reproduce the directly installed memories and archived predictions/scores exactly.

B.3 Natural-Document Memory Ablations During Free Generation

To test generated answers directly, we use the fixed reader and 20-step memories for every natural question and writing seed. We disable only the memory updates in late MLP projections (layers 24–35), late attention projections (the same layers), or all adapted projections. Backbone and reader branches stay active, and the intervention applies throughout prefill and autoregressive generation. With the full-memory condition, this yields $192 \times 3 \times 4 = 2{,}304$ predictions.

Removing late MLP memory reduces F1 by 10.74 pp, with simultaneous 95% interval $[-17.97, -3.51]$. Removing all memory reduces it by 14.01 pp $[-21.24, -6.78]$. Removing late attention memory changes F1 by only -0.05 pp $[-7.28, 7.18]$ (Table 7; Figure 9). The late MLP result extends the synthetic localization evidence to a natural-document necessity test on generated answers. It does not locate all stored knowledge exclusively in this group or measure an additive contribution.

Late MLP removal changes F1 by -17.08 pp $[-25.20, -9.59]$ on NewsQA and -17.29 pp $[-27.86, -6.78]$ on StreamingQA, but by -1.02 pp $[-4.37, 2.30]$ on QuAIL and -7.56 pp $[-17.76, 0.64]$ on ROPES. These source-specific intervals are pointwise. Thus the necessity evidence, like the wrong-

Condition	F1 (%)	ΔF1 (pp)	Simultaneous 95% CI
Full memory	41.45	0.00	—
Late MLP memory off	30.72	-10.74	[-17.97, -3.51]
Late attention memory off	41.40	-0.05	[-7.28, 7.18]
All memory off	27.44	-14.01	[-21.24, -6.78]

Table 7: Natural-document free-generation ablations with the fixed reader and 20-step memories. Differences are relative to full memory. Intervals are simultaneous across the three F1 contrasts, using 10,000 source-stratified document-cluster bootstrap resamples.

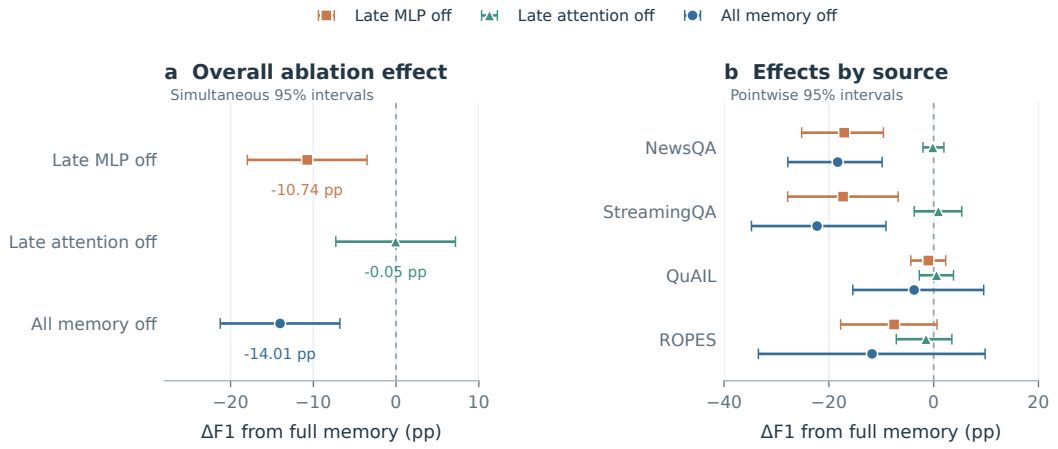


Figure 9: Effect of disabling memory branches throughout natural answer generation. (a) Overall F1 changes with simultaneous 95% intervals across the three contrasts. (b) Source-specific changes with pointwise exploratory intervals. The reference is full memory with reader enabled in every panel.

memory control, is heterogeneous rather than universal across sources. The supplementary option parser gives 47.22% QuAIL accuracy with both full memory and late MLP memory disabled (paired difference 0.00 pp [−3.47, 2.78]); parse coverage is 95.14% and 99.31%, respectively. The lack of a clear QuAIL effect survives this format check.

Complete-reference mean log probability also declines when late MLP memory is removed: −0.331 nats/token [−0.477, −0.188]. However, late attention removal increases this score by 0.131 nats/token [0.085, 0.179] while leaving aggregate F1 nearly unchanged. These scores use all three questions per document, unlike the earlier first-question intervention. Reference likelihood and greedy lexical answer quality therefore need separate interpretation even under the same intervention. Ablation is a perturbation of the computation; it is not a minimally edited natural-document counterfactual.

The 576 full-memory and 576 all-memory-disabled observations exactly reproduce their archived full-memory and reader-only predictions and reference scores. Duplicate seed-independent reader-only results remain within their document cluster and do not increase the effective sample size.

B.4 Additional Memory Benefit Beyond Task-Only Adaptation

The task-only adapter trained with Reptile and contrastive learning also benefits from attaching a saved 20-step memory: its F1 rises from 29.79% to 36.64%, a paired gain of 6.85 percentage points (pp; 95% CI [1.20, 12.40]). The memory-trained reader rises from 27.44% without memory to 41.45% with memory. A supplementary paired difference-in-differences asks whether the memory

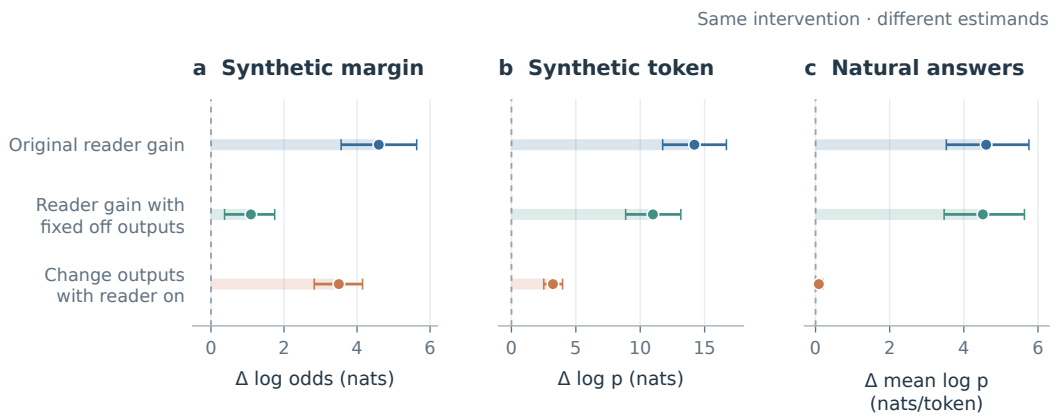


Figure 10: The fixed-output result depends on both task and estimand. (a) Synthetic correct-minus-alternative log odds. (b) Absolute log probability of the same correct synthetic token. (c) Mean log probability of complete natural reference answers. "Fixed off outputs" substitutes the reader-off memory-output cache; "change outputs" switches to reader-on values while the receiving reader remains enabled. Pointwise 95% cluster intervals are from the existing 4,000-resample analysis. Axes have different units/scales.

benefit itself is larger for the memory-trained reader:

$$\begin{aligned}\big(F_1(R+M)-F_1(R)\big)-\big(F_1(R_{\text{task}}+M)-F_1(R_{\text{task}})\big)\\ =7.17\ \text{pp}\quad[1.61,12.71].\end{aligned}\tag{7}$$

Here R is the memory-trained reader and R_{task} is the control trained without memory. The contrast uses the same questions and saved memories in all four cells, broadcasting the seed-independent reader-only scores within each document before paired resampling. The difference is consistent with an additional benefit from memory-conditioned training, while showing that general QA adaptation already enables some memory use. The controls match adapter capacity, source pool, gold-QA exposures and order, and optimizer updates; the historical reader's original initialization and negative-answer draws are incompletely reconstructable. This is therefore not a fully isolated causal estimate of memory-conditioned training.

B.5 Score Dependence of the Fixed-Output Interpretation

Figure 10 visualizes the three distinct estimands in Appendix A.5. On synthetic pairs, the reader's correct-minus-alternative log-odds gain falls from 4.59 nats to 1.09 nats when memory outputs are fixed to the reader-off cache. For the *same correct synthetic token's absolute log probability*, the gain falls from 14.21 to 10.99 nats. The alternative token's absolute probability also increases, so relative log odds cancel part of a shared probability change. On natural complete answers, the mean-log-probability gain is 4.60 nats/token and remains 4.51 with reader-off memory outputs fixed. Switching those outputs to the reader-on cache adds only 0.089 nats/token $[-0.003, 0.194]$.

The natural-document caches contain prompt positions only; donor prompt activations contain no reference answer, and subsequent answer positions remain unpatched. Even the first-answer-token diagnostic preserves most of the gain (10.89 versus 10.67 nats), although a first token may be a common word or subword. Thus the synthetic margin result does not establish that changes to cached prompt memory outputs account for most natural QA improvement. Memory access, general task adaptation, answer formatting, and later answer-token computation can all contribute. Intervention effects are non-additive and do not define a unique mediation percentage.

Reproducibility. The supplementary experiment bundle preserves the protocol written before these additional runs, code/model provenance, references to saved memories, raw generations, exact condition-coverage checks, baseline/endpoint restoration audits, and scripts for regenerating the figures and numeric tables. Large model and adapter files remain on the experiment server; the analysis bundle operates on saved observations without loading them.

C NATURAL-DOCUMENT FACTUAL COUNTERFACTUALS

Evaluation. We evaluate whether generated answers follow changes to facts in the written context using 1,000 original–counterfactual passage pairs. Each pair retains the question while changing its target fact. The two passages are written into separate NTP memories. We compare memory alone, memory with the trained reader, and memory with a task-only reader. Full-context ICL supplies the passage directly and provides a reference for answering with the source text available.

Metrics. The generation metric is exclusive target-value following: an answer must contain its variant's target value, exclude the opposite value, and avoid a predefined refusal expression. The Original and Counterfactual columns report successes for the respective members; Both variants requires both answers in the same pair to succeed. This metric evaluates the specified factual edit.

Generation-rate comparisons below are differences between the reported point estimates.

We also measure how changing the memory changes relative preference between the two target answers. For original answer a_A and edited answer a_B, let $L(M) = \log p(a_B \mid q, M) - \log p(a_A \mid q, M)$, summing teacher-forced token log probabilities over each candidate and excluding EOS. The paired preference shift is $L(M_B) - L(M_A)$, where M_A and M_B denote the original and edited memories, or the corresponding input passages for ICL. A positive shift favors the edited answer after the factual change. We report the mean shift in nats with its 95% confidence interval. Candidate answers are used only for scoring and are absent from generation prompts.

Configuration	Original /1000	Counterfactual /1000	Both variants /1000	Preference shift nats [95% CI]
Memory only	210	187	125	18.01 [14.38, 21.46]
Memory + trained reader	435	382	320	34.89 [31.71, 38.34]
Memory + task-only reader	230	193	112	11.59 [8.46, 15.49]
Full-context ICL	951	925	898	68.31 [63.96, 71.84]

Table 8: Natural factual counterfactual evaluation on 1,000 passage pairs. Counts measure exclusive target-value following; Both variants requires success on both members of the same pair. Preference shift measures the change in edited-versus-original candidate log odds, with 95% confidence intervals.

Results. The trained reader improves factual consistency across both versions of a passage (Table 8). It succeeds on 435 original variants and 382 counterfactual variants, with both answers satisfying the target-value criterion for 320 pairs. Its paired success rate is therefore 32.0%, compared with 12.5% for memory alone and 11.2% for the task-only reader: gains of 19.5 and 20.8 percentage points, respectively. The improvement appears in both original and counterfactual answers, supporting the trained reader's use of the factual content of the attached memory.

The trained reader also yields a positive preference shift of 34.89 nats [31.71, 38.34], compared with 18.01 [14.38, 21.46] for memory alone and 11.59 [8.46, 15.49] for the task-only reader. These shifts characterize sensitivity to the changed fact at the candidate-scoring level; the generation counts separately measure whether the model produces the corresponding answer.

These generation results complement the memory-identity controls by testing responses to changed facts within a passage. The comparison with a task-only reader further supports training the reader to operate with written memories. Full-context ICL provides the complementary reference in which the source passage remains directly available during inference.

D NTP FIT, QUERY ROBUSTNESS, AND FACT LOCATION

We add three exploratory follow-ups with the reader checkpoint fixed: writer fit versus downstream QA, query formulation, and fact position versus document length. They characterize access to NTP memories within our setting, without isolating the causes of cross-paper performance differences.

D.1 DOCUMENT FIT DOES NOT UNIQUELY PREDICT ACCESSIBLE KNOWLEDGE

We reuse all 64 natural documents, 192 questions and three memory initializations, averaging seeds and questions within each document. At 20 writer updates, Spearman correlations of NTP loss with memory-only F1, reader F1 and correct-minus-wrong-memory F1 are $-0.124[-0.380, 0.134]$, $-0.271[-0.481, -0.038]$ and $-0.077[-0.330, 0.195]$, respectively. Brackets give pointwise 95% intervals from 4,000 source-stratified document bootstrap samples.

Residualizing ranks on source indicators gives $-0.257[-0.482, -0.005]$ for reader F1. The correlation between changes in loss and reader F1 from six to twenty updates is $-0.061[-0.312, 0.204]$. Better fit is associated with reader QA here, but loss alone is an incomplete proxy for correct-memory access. Intervals crossing zero do not prove independence. Full Spearman, Kendall, source-adjusted and change estimates are released; no multiplicity correction is applied.

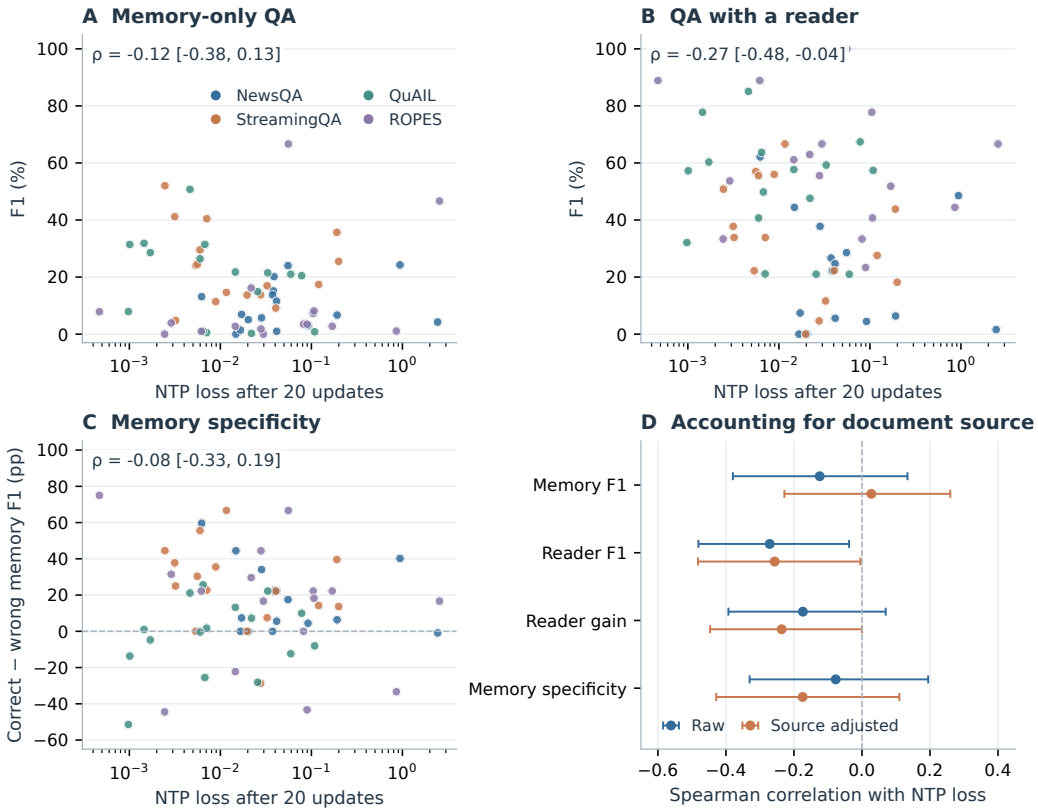


Figure 11: **Document fit and QA are distinct measurements.** Each point is one natural document, averaging its three memory seeds and three questions. Panels A–C use 20-update memories; the horizontal axis is logarithmic. Memory specificity is correct-minus-wrong-memory reader F1. Panel D compares raw and source-adjusted Spearman correlations; bars are source-stratified document bootstrap intervals. Reader gain is reader F1 minus memory-only F1.

D.2 SENSITIVITY TO THE QUESTION AND INSTRUCTION FORMAT

We evaluate the same 64 documents, 192 questions and saved 20-update memories under five fixed prompts: the historical IM4 prompt; a plain user question; an answer-only instruction; a brief-answer request; and one fixed question paraphrase under the original IM4 system message and trigger. Multiple-choice options are unchanged. The latter comparison changes the question wording alone; the other prompt bundles jointly change instructions and the trigger, and do not isolate those factors. Paraphrases were fixed before inference, without selection on performance. All conditions use greedy generation with the same 96-token limit. Correct and wrong memories use the same within-source donor mapping as before; reader-only is evaluated once per question and prompt.

Table 9: **Prompt robustness on natural documents.** F1 is in percent. Memory and Reader denote each adapter alone; Both and Wrong combine the reader with correct and mismatched memories. The final column gives the paired correct-minus-wrong gap in percentage points with its 95% document-cluster interval. Each row uses all 192 questions and three memory seeds.

Prompt	Memory	Reader	Both	Wrong	Correct − wrong
IM4	15.85	27.44	41.45	28.14	13.31 [7.33, 19.20]
Plain question	5.45	29.17	40.49	28.07	12.42 [6.22, 18.33]
Answer only	30.30	30.08	43.25	29.99	13.26 [7.29, 18.98]
Brief answer	21.19	32.37	43.43	30.87	12.55 [7.10, 18.01]
Paraphrased IM4	17.32	28.74	37.03	27.78	9.25 [3.98, 14.30]

The correct-minus-wrong-memory advantage is positive under all five prompts, ranging from 9.25 to 13.31 percentage points, with every pointwise interval above zero. This includes three prompts without the IM4 trigger. At the same time, memory-only F1 varies from 5.45% to 30.30%, whereas memory-plus-reader F1 ranges from 37.03% to 43.43%. Consequently, the gain over memory-only QA is substantially more prompt-sensitive than the correct-minus-wrong-memory comparison. Output format is a material part of the apparent gain against memory-only QA.

The paraphrased IM4 prompt gives memory-plus-reader F1 37.03%, with memory specificity 9.25 [3.98, 14.30] percentage points. Relative to original IM4, its paired changes in memory-plus-reader F1 and memory specificity are -4.42 [-8.16, -0.62] and -4.06 [-7.83, -0.06] percentage points, respectively. Thus, usable memory information persists under these alternatives, but access is not invariant to question wording. QuAIL's memory-specificity interval includes zero under every prompt. These are pointwise exploratory intervals, conditional on this reader. We report exact match, complete-reference likelihoods, output lengths and cap-hit rates alongside F1 to distinguish output behavior from memory dependence. The historical IM4 endpoint reconstructs all 1,920 reference predictions exactly in generated text, with maximum reference-log-probability error 0.

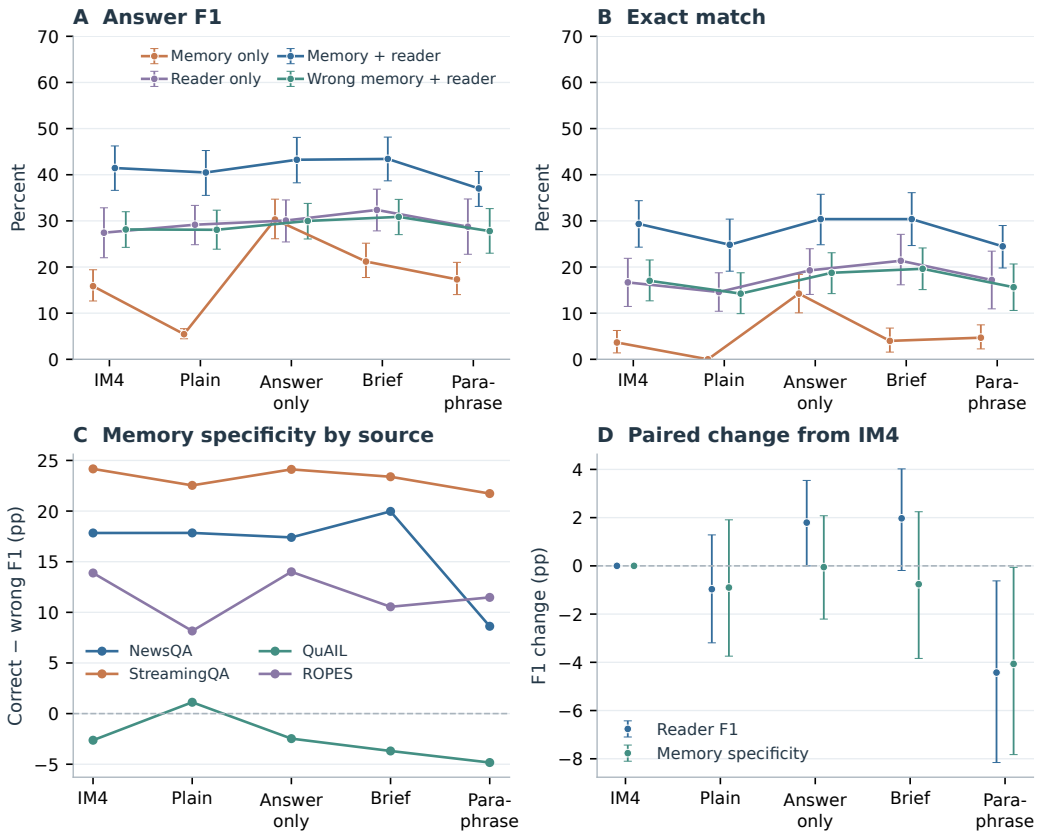


Figure 12: **The same memories under five query formulations.** A–B: mean token F1 and exact match with 95% document-cluster intervals. C: correct-minus-wrong-memory F1 by source (point estimates). D: paired changes from IM4 in reader F1 and memory specificity. Paraphrases retain the original IM4 instructions and trigger. Intervals cluster all questions and seeds within documents.

D.3 CONTROLLED FACT POSITION AND DOCUMENT LENGTH

We construct a complete grid from 24 synthetic counterfactual pairs, six per relation family. In each cell, the target value occurs at the token nearest 10%, 50% or 90% of a 128-, 512-, 2,048- or 4,096-token document, excluding the IM4 prefix and EOS. Each pair differs in exactly one contextual answer token, and has two fixed question formulations. Compact entity identifiers are shared between documents and questions. The same neutral filler tokens are moved around the fact within a length condition, holding their multiset constant; filler can contain truncated sentences at insertion boundaries. Longer conditions repeat the fixed filler bank. This is a controlled synthetic stress test, rather than a sample of natural long documents. Serialized token sequences are the writer input and are supplied with decoded text and offset audits.

We train memory adapters from the same saved seed-42 initialization for 6/20 updates using the original rank, learning rate and optimizer, with the reader frozen and inactive. All lengths use a 128-token chunked vocabulary projection to compute full causal NTP without truncating long documents. Against the native loss, the maximum discrepancy at 256/512/1,024 tokens is 2.4×10^{-7} and the audited gradients are identical. Repeated writes of an excluded 4,096-token smoke document reproduce six- and twenty-update weights exactly. Fixed update counts expose longer documents to more tokens, so the length comparison is not compute-matched.

We evaluate memory-only and memory-plus-reader candidate preferences and greedy free generation. The opposite counterfactual member provides the wrong-memory control under the same query. It is not an extra independent observation. Base and reader-only controls are shared across the grid because their prompts contain no document. Candidate choice without memory averages 50% over balanced counterfactual values by construction. Fact-hit generation requires the intended value, absence of the opposite value, and absence of a refusal; candidate log odds and generation remain separate outcomes.

Table 10: **Fact-hit generation by document length.** Percent, averaged over all three fact positions, both values and both questions. Intervals resample the 24 pairs within relation families, keeping the full grid together.

	6 updates		20 updates	
Tokens	Memory only	Memory + reader	Memory only	Memory + reader
128	78.8 [72.6, 85.1]	29.9 [24.7, 35.4]	47.2 [39.2, 54.5]	76.0 [70.5, 81.2]
512	78.1 [72.9, 83.3]	5.9 [3.1, 9.0]	66.7 [60.1, 73.3]	59.7 [53.5, 66.0]
2048	82.3 [75.7, 88.5]	6.9 [3.8, 10.8]	67.7 [60.8, 74.7]	64.6 [58.3, 70.8]
4096	74.3 [70.5, 78.1]	2.8 [1.4, 4.2]	78.1 [70.1, 85.4]	49.0 [41.3, 56.2]

Candidate preference need not produce a correct answer. At 4,096 tokens and six updates, the reader chooses the correct candidate in 98.61% of comparisons, with mean correct-versus-counterfactual log odds of 3.364 nats, yet free-generation fact-hit is only 2.78%. At twenty updates, these quantities are 98.96%, 6.168 nats and 48.96%, respectively. Both base and reader-only controls have zero fact-hit generation. The candidate probe compares only two specified values at the answer position; it does not require either value to beat every other vocabulary token or sustain the subsequent generation. Its high accuracy therefore establishes a relative preference, not reliable free-form retrieval. Figure 14 further separates this query-conditioned preference from target-token fit under the original document prefix.

The reader's generation benefit has boundary conditions. With six updates, adding the reader reduces generation fact-hit at all four lengths. With twenty updates, its paired advantage is 28.82 [19.44, 38.89] percentage points at 128 tokens but -29.17 [-40.62, -17.01] points at 4,096 tokens; the 512- and 2,048-token intervals include zero. These method contrasts were added after inspecting the complete grid and are reported as post hoc, pointwise comparisons. Thus, the reader is not a universal improvement over direct generation from an NTP memory. The synthetic task also changes entity style and filler relative to the natural-document population, so the negative result does not isolate length alone as the cause of failure.

Position effects vary with length and writing budget. At 128 tokens and twenty updates, reader fact-hit decreases from 90.62% at the front to 72.92% in the middle and 64.58% at the end. The paired middle-minus-front and end-minus-front differences are -17.71 [-29.17, -6.25] and -26.04 [-41.67, -10.42] percentage points. At the other three lengths with twenty updates, both reader generation position-contrast intervals include zero. This grid does not establish a universal middle-position deficit or elimination of positional bias by the reader. We retain every cell and both writing budgets in Figure 13.

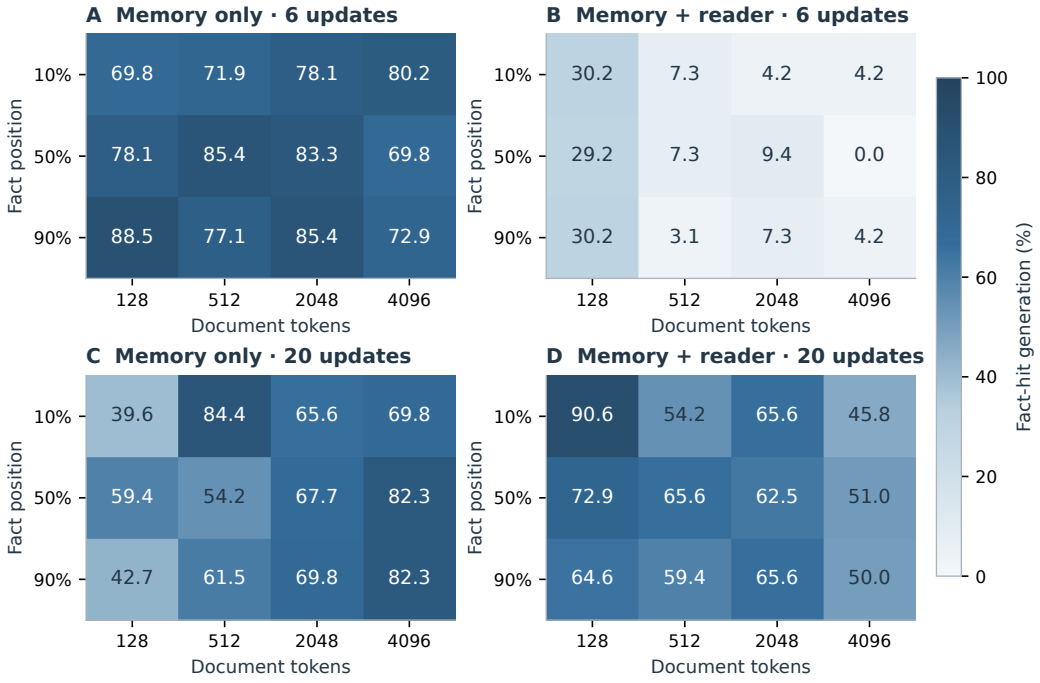


Figure 13: **Fact location and length under a fixed writer policy.** Each heatmap cell includes all 24 pairs, both fact values and two questions; these 96 outputs form 24 independent pair clusters. All 12 cells are retained. A common color scale permits comparison between memory-only and reader generation and between writing budgets.

Scope and validation. The position grid contains 576 independently executed document writes and 4,608 memory-conditioned prediction records, plus 96 shared base/reader-only records. Reload-and-rescore checks reconstruct all 2,304 tested endpoints exactly. All twenty-update weights and the six-update weights for the first pair in each family are retained (672 checkpoints); all writing trajectories, tensor hashes and exact input IDs are retained. The remaining six-update states are used directly during evaluation and are reproducible from the deterministic recipe. Reader/backbone invariance is checked throughout. All uncertainty here concerns the sampled documents or synthetic pairs conditional on a single reader checkpoint; it does not measure variation across reader training seeds. Results are reported at both writing budgets and for every prompt, source, length and position, without filtering on reader success.

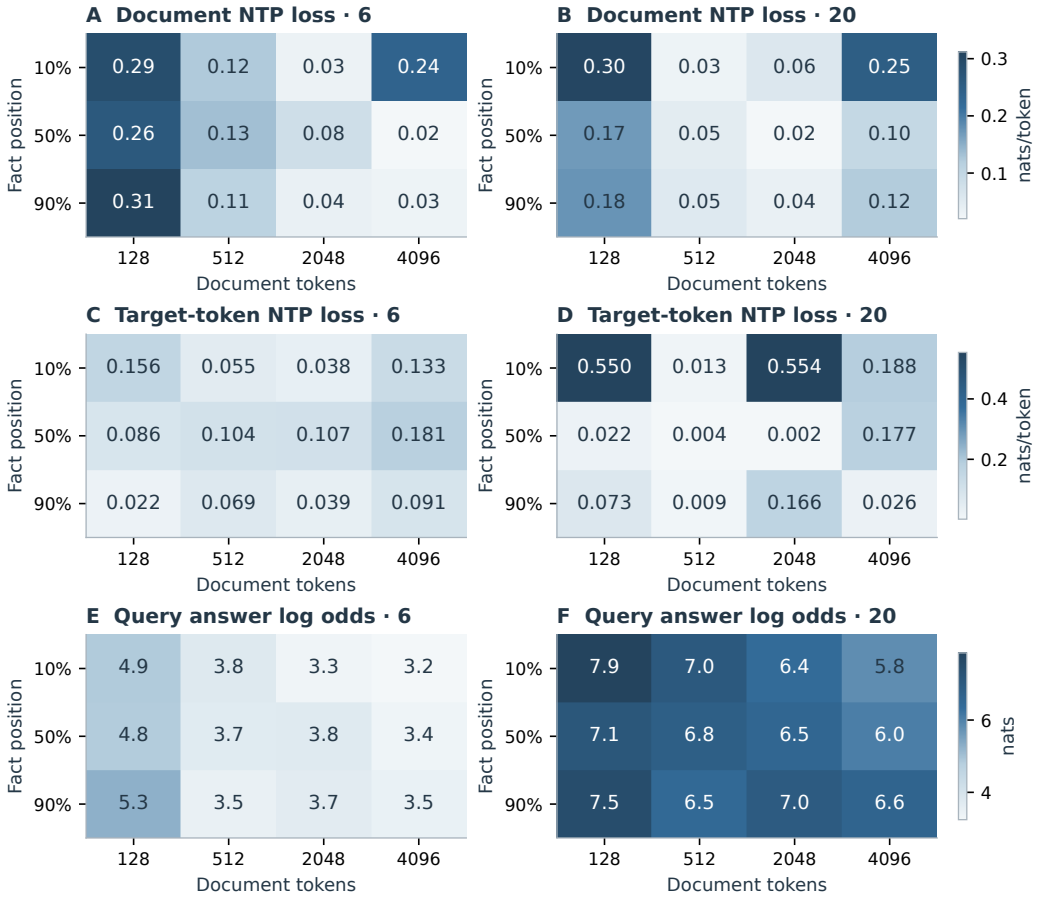


Figure 14: **Separating document fit, target fit and query access.** A–B: mean document NTP loss, which includes repeated neutral filler. C–D: NTP loss specifically at the target value, conditioned on its document prefix. E–F: correct-versus-counterfactual candidate log odds under the question prompt with the reader active, before any answer token is supplied. Color scales are shared within each row. Low document loss may reflect easy filler prediction; target-token fit and question-answer preference measure different access conditions. Full cell intervals and paired position contrasts are released with the figure data.

E EFFICIENCY MEASUREMENT

Timing protocol. The efficiency sweep uses Qwen3-8B on a single NVIDIA RTX PRO 6000 Blackwell Server Edition GPU with 96 GB of memory, PyTorch 2.8.0, and Transformers 5.13.1. We measure synthetic contexts of 1k, 2k, 4k, 8k, 16k, and 32k tokens, using batch size one, BF16, SDPA attention, and greedy generation. Query timing uses three measured runs after one warmup, with 32 output tokens. Throughput counts generated tokens per wall-clock second, including prompt prefill. The readout prompt has 89 tokens, while the full-context prompt contains the context plus 105 tokens. The full-context route performs a fresh prefill for each query, without reusing a context-prefix cache across queries.

Memory writing uses the main experiment's rank-32 NTP configuration with six passes over each context. We average three writing runs per length after a one-epoch warmup on a 1k-token context. For query timing, the memory and reader updates are merged into the base weights. The fixed readout timing uses one memory and is reused across the synthetic lengths; this sweep measures execution cost, separately from the QA evaluations in Section 5.2.

Amortizing memory writing. Both panels of Figure 4 use the same query measurements. For Q queries to one context, we combine mean writing and query times as

$$\mathcal{R}(Q) = \frac{T_{\text{write}} + Q\,T_{\text{readout}}}{Q\,T_{\text{full}}}, \tag{8}$$

where T_{readout} and T_{full} are the respective per-query times, including prefill and generation. Writing takes 1.55–69.03 s across the measured lengths; query savings recover this cost after 25–32 queries (Table 11). These ratios include memory optimization and query generation, and exclude adapter merging, model loading, checkpoint serialization, and disk I/O. The measured memory-merge time is 1.90 s per context; including it adds this one-time cost to the numerator of Equation 8.

Context length	1k	2k	4k	8k	16k	32k
Full-context query time (s)	0.553	0.604	0.702	1.002	1.635	3.313
Memory-writing time (s)	1.55	2.75	5.62	12.69	28.71	69.03
Queries to recover writing cost	32	28	29	26	26	25
Time ratio at $Q = 128$	0.93	0.87	0.78	0.60	0.45	0.31

Table 11: Writing and amortization on the same GPU as the query sweep. The fixed readout query time is 0.504 s. Ratios and crossover counts use unrounded mean times in Equation 8.

Memory and first-token latency. At 32k context tokens, peak memory allocated by PyTorch during query generation is 22.87 GiB for full-context inference and 15.45 GiB for readout. This measures allocated tensor memory, including model weights, rather than total device memory usage. A separate one-output-token run on the same GPU model measures time to first token: 2,588 ms for full-context inference at 32k and 22.4 ms for readout. This single-token timing is not used to compute throughput or amortization.

F TRAINING BUDGET ACCOUNTING

Scope and definition. The audit covers the Qwen3-8B reader trained with Reptile, NLL, and contrastive learning at episode 6,000 used in Table 1, together with construction of its source-memory bank. An input-token presentation counts one non-padding token in a sequence processed for a training example. It includes prompt tokens masked from the loss, shared formatting, and termination tokens. Repeated epochs and repeated QA draws are counted each time; each negative candidate contributes its own complete input sequence. Backpropagation and activation-checkpoint recomputation do not multiply this data-exposure count. The statistic measures training inputs, not FLOPs.

The recipe uses 1,197 source contexts containing 0.93M raw context tokens and a pool of 30,841 QA pairs. Each memory is written with six NTP passes. Reader training comprises 6,000 episodes, each with eight updates at batch size two: 96,000 reference QA presentations and 192,000 negative-answer presentations for the contrastive loss. The sampling logs contain 26,098 distinct reference QA pairs from 1,162 contexts; the writing budget includes all 1,197 source memories. The reader is initialized without a previously trained reader checkpoint and uses no teacher-logit distillation stage.

Training component	Non-padding input-token presentations
Source-memory NTP writing (six passes)	5,664,690
Reader reference-answer sequences	9,344,517
Reader contrastive-negative sequences	18,736,041
Total	33,745,248

Table 12: Cumulative training inputs for the Qwen3-8B main-result checkpoint. Reader sequences include the question prompt as well as the candidate answer. Including batch padding yields 35,337,030 input positions in total.

Reconstruction and exclusions. Counts are reconstructed from the archived source inputs, exported tokenizer, historical feature-construction and negative-selection functions, and the complete episode sampling log. Input-file hashes match the memory-construction manifest. Reconstructed maximum sequence lengths match the original preflight audit: 2,547 tokens for writing and 248 for reference QA. Negative-answer text was not logged individually and is recovered using the deterministic historical selection procedure. Thus, these are reconstructed counts rather than token counters recorded during training. The budget excludes backbone pretraining, LLM-assisted data construction, heldout and target memory writing, evaluation, and other experimental runs. Per-target memory writing is a deployment cost, reported separately in Section 5.5. The 33.75M total applies to this Qwen3-8B run, not to an aggregate over the three backbones.

Reader training time. The archived process-level resource log for the same Qwen3-8B checkpoint records 11,087.43 seconds (approximately 3.1 hours) for the 6,000-episode reader-training run on a single NVIDIA GeForce RTX 5090. This wall-clock measurement includes process initialization, memory loading, training, and checkpoint saving. Source memories were precomputed, and evaluation was disabled for the timed process. The measurement therefore excludes source-memory construction, data preparation, and evaluation, and is narrower in scope than the combined writing-and-reading token budget above. We report the measured duration without inferring a training-speed ratio against baselines run with different hardware and training protocols.

Interpreting the baseline comparison. SHINE reports a 6B-token pretraining corpus and one epoch, followed by instruction fine-tuning (Liu et al., 2026, Section 5.1). GenerativeAdapter reports a 1B-token SlimPajama sample for pretraining and subsequent instruction tuning (Chen et al., 2024, Section 3 and Appendices A–B). Doc-to-LoRA reports approximately 900M FineWeb-Edu tokens plus passage-grounded QA data for context-distillation meta-training (Charakorn et al., 2026, Appendix B.1). Its reported 80K-step and 20K-step stages repeatedly process training contexts; the 900M figure is the source corpus size, not cumulative training exposure. The three corpus sizes therefore describe only part of the corresponding training pipelines. We retain these published scopes in Table 13, rather than presenting them as fully matched token counters. The comparison supports a smaller training scale for our recipe; it does not isolate architecture as the sole cause of data efficiency or establish that a hypernetwork could not be trained with less data.

Method	Counted quantity	Tokens
SHINE	Pretraining context corpus	6B
Doc-to-LoRA	FineWeb-Edu context corpus	$\approx$ 900M
GenerativeAdapter	Pretraining context corpus	1B
LoRA Readout (Qwen3-8B)	Writing + reader input exposure	33.75M

Table 13: Training token scale with different accounting scopes. Baselines report raw context-corpus sizes (Liu et al., 2026; Charakorn et al., 2026; Chen et al., 2024); ours counts cumulative training inputs, including repeats and contrastive negatives. Baseline counts exclude additional instruction-tuning or QA/distillation sequences; all entries exclude backbone pretraining. These are not matched compute measurements.

G Additional Main-Benchmark Controls

G.1 Computer-Paper Summaries (CP): Source and Construction

CP denotes a computer-paper QA evaluation set constructed for this work from compact summaries of publicly available ICLR 2026 papers, including work on instruction-data selection and interatomic potentials (Chen et al., 2026; Zhou et al., 2026). Each archived context is a short fact ledger describing a paper's authors, method, and reported findings, rather than the full paper. Questions and reference answers were generated through an LLM API from these summaries; they are our evaluation material, not QA annotations released by the original paper authors. The frozen target set contains 16 paper summaries and 1,024 questions, with 64 questions per summary.

The target contexts and QA were inherited from the earlier CP artifact and kept fixed when constructing APQA. CP is excluded from shared-reader training and is not rewritten by the APQA one-call entity-replacement pipeline. The archived target contexts match their source summaries after removing formatting and memory-identifier prefixes. This provenance differs from the entity-replaced CUAD and RepliQA targets. CP uses normalized-token ROUGE-L F1 as specified in Table 1; compact-summary QA should not be interpreted as an evaluation over the full source papers. The accompanying CP provenance record maps target IDs to archived source IDs, paper titles, and construction files.

G.2 Additional Memory-Identity Controls

Table 14 retains all evaluated Qwen3-8B conditions behind Table 2. With the reader and question prompts fixed, removing or replacing the correct memory reduces performance across all five datasets. Random LoRA controls perform close to reader-only inference on CP, CUAD, and RepliQA. On ROPES and QuAIL, random LoRA plus the reader scores 0.231 ROUGE-L and 0.278 accuracy, below reader-only scores of 0.271 and 0.287 and correct-memory scores of 0.573 and 0.526. These evaluations are distinct from the natural-document token-F1 diagnostics in Appendix A.

Configuration	CP	CUAD	RepliQA	ROPES	QuAIL (Acc.)
Base LM: question only	0.254	0.251	0.268	0.092	0.346
Base LM: question + context	0.604	0.673	0.685	0.762	0.715
Reader only	0.260	0.326	0.318	0.271	0.287
Correct memory only	0.300	0.239	0.250	0.248	0.232
Correct memory + reader	**0.357**	**0.387**	**0.410**	**0.573**	**0.526**
Wrong memory + reader	0.241	0.300	0.321	0.245	0.299
Random LoRA + reader	0.259	0.325	0.316	0.231	0.278

Table 14: Main-benchmark memory controls, using the evaluation sets and metrics in Table 1 (QuAIL: accuracy of multiple-choice option selection). Only the full-context reference includes context tokens. Bold marks the best context-free score.

G.3 Entity Replacement during Reader Training

Figure 15 illustrates the consistent replacement of names, dates, and related terminology in a contract excerpt and its QA pair. The purchaser changes from Zebra Technologies Corporation to Luminara Devices Corporation, and the reference answer changes with it while the question's target relation is preserved. The same mapping is applied throughout the context and its QA collection so that the reader must recover the newly assigned associations. This example illustrates the transformation used in data construction; it is not an additional evaluation result.

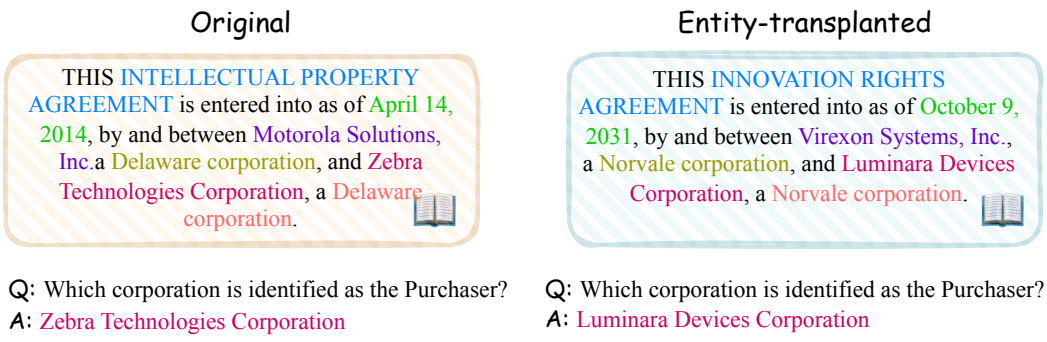


Figure 15: Illustration of consistent entity replacement. Corresponding colors mark substitutions in the original and transformed contract excerpts. The question retains its target relation, while the reference answer follows the entity named in the transformed context.

We compare readers trained on original-entity and entity-replaced APQA contexts, both using Reptile with NLL and contrastive learning. Both are evaluated on the same 965-question entity-replaced APQA holdout, with identical questions and context memories. Training on replaced entities raises ROUGE-L from 0.396 to 0.436 (Table 15). This supports the transformation for the evaluated setting, but does not isolate improved memory use from closer alignment between training and evaluation distributions. No memory-identity contrast is inferred from these two scores alone.

Reader-training contexts	ROUGE-L F1
Original entities	0.396
Replaced entities	**0.436**

Table 15: Effect of entity replacement during reader training, evaluated on the same entity-replaced APQA holdout.