MEMORY DECODER AT SCALE: A PRETRAINED, PARAMETRIC LONG-TERM MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Decoder-only language models (LMs) entangle long-term memory and reasoning in a single parameter set, making it difficult to scale memory capacity independently. Memory Decoder (Cao et al., 2026) introduces an external parametric long-term memory and shows primitive results, albeit on small scales. In this work, we present ***Memory Decoder at Scale*** and systematically study memory scaling with model and data size, parameter allocation between memory and the LM, and pretrained memory's ability to capture general and domain-specific knowledge. We scale external memory to 6.9B parameters and pretrain it on 300B tokens. At this scale, we address retrieval scalability and storage overhead through a distributed pipeline for FAISS (Douze et al., 2025) indexing and retrieval and sparse kNN distribution storage, respectively. Our results show that pairing larger memories with smaller LMs outperforms scaling the LM alone. This architectural disentanglement of long-term memory from the LM substantially improves parameter efficiency. A 6.9B general memory enables Pythia-410M to surpass Pythia-12B on average across 17 benchmarks with 39% fewer total parameters. Domain-specific memory outperforms the strongest reported baseline among CPT, LoRA, and RAG by 4.05–8.53 points on average. Overall, our results establish the architectural advantage of external memory over conventional decoder-only LMs, paving the way toward disentangling long-term memory from the LM.

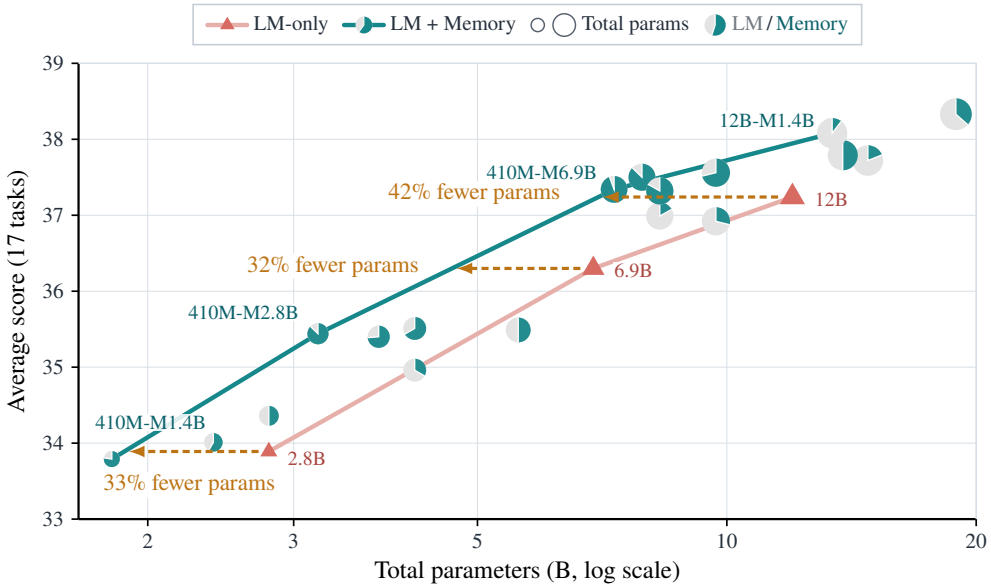

Figure 1: Across scales, pairing larger memories with smaller models is more parameter-efficient than scaling the model alone. Across 17 tasks, Pythia-410M + Mem-6.9B reaches an average score of 37.34, surpassing Pythia-12B at 37.24 with 39% fewer total parameters. Circle size indicates total parameters, and teal sectors show the memory fraction. Dashed arrows show parameter savings estimated by interpolating the LM + Memory curve at matched scores.

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

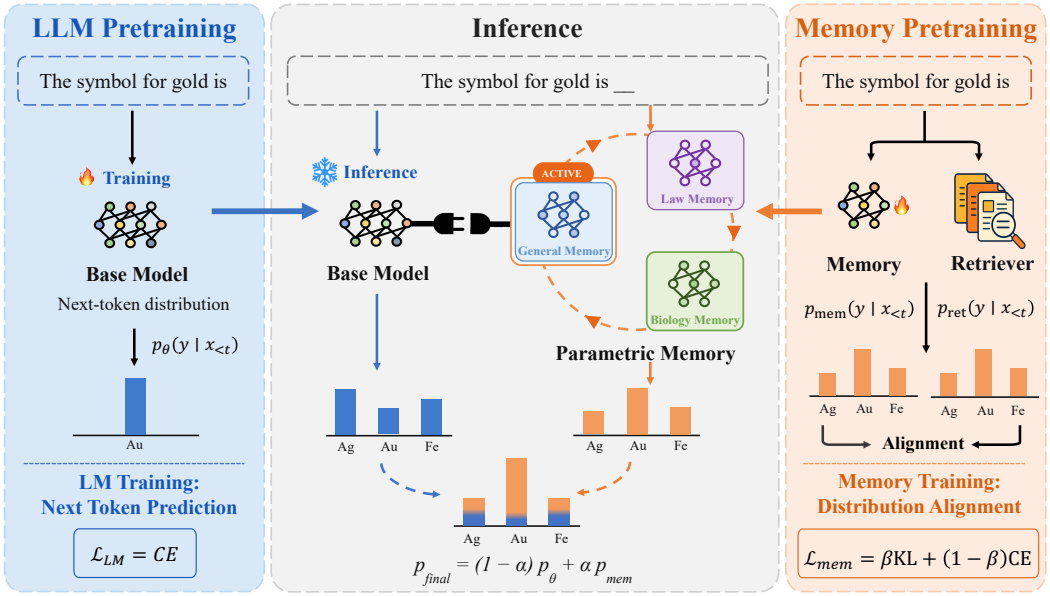

Figure 2: Overview of parametric long-term memory pretraining. **Left**, a standard language model is pretrained with next-token prediction and then frozen for inference. **Right**, a parametric memory is pretrained to align its output distribution with a retriever while retaining a next-token prediction objective. **Middle**, the frozen base model and a swappable memory process the same context in parallel, and their next-token distributions are interpolated to produce the final prediction.

1 Introduction

The human brain is organized into specialized functional systems that interact to support cognition (Fair et al., 2009). Memory and reasoning likewise rely on partially distinct sets of neurons, allowing memory storage and cognitive computation to be functionally dissociated (Squire, 2009; Baddeley & Warrington, 1970). In contrast, standard decoder-only language models (Xu et al., 2026; Team, 2026; Zeng et al., 2026; Singh et al., 2025) entangle memory and reasoning within a single set of parameters. Long-term memory cannot be pretrained or scaled independently, and increasing memory size requires a corresponding increase in the total parameter count of the model. Under this shared parameterization, domain adaptation through continued pretraining or full fine-tuning requires optimization of the entire parameter set, incurs substantial training cost and catastrophic forgetting (Kirkpatrick et al., 2017). These limitations raise a central architectural question: can long-term memory be disentangled from the backbone and trained or scaled without expanding or updating the entire base model?

Existing memory mechanisms for language models have not fully answered this question. Much of the literature instead improves access to information within the current context. Segment-level recurrence (Dai et al., 2019), bounded KV caches (Xiao et al., 2024b), retrieval from distant context (Xiao et al., 2024a), and RoPE rescaling (Ding et al., 2024) improve access to contextual information during inference, but leave the entanglement between long-term memory and reasoning unchanged. Methods aimed more directly at long-term memory remain comparatively limited. RAG (Lewis et al., 2020) retrieves passages from an external corpus as additional context, whereas kNN language models (Khandelwal et al., 2019) interpolate model predictions with a distribution constructed from nearest neighbors in an external datastore. Both require external retrieval during inference, with additional context processing for RAG and substantial storage and nearest-neighbor search costs for kNN language models. Titans (Behrouz et al., 2024) instead uses a neural long-term memory that encodes information from historical context at test time rather than knowledge acquired from a pre-training corpus. Memory Decoder (Cao et al., 2026) and MLP Memory (Wei et al., 2025) pretrain parametric memory modules to imitate the output distributions of non-parametric kNN retrievers. In particular, Memory Decoder (Cao et al., 2026) demonstrates the effectiveness of pretrained parametric memory with models scaling up to 1B parameters, using WikiText-103 and domain-specific cor-

pora containing only millions of tokens. However, its scaling behavior on substantially larger models and diverse pretraining corpora remains unexplored. It therefore remains unclear how memory pretraining scales with model and data size, how the parameter budget should be allocated between the base model and memory, and whether pretrained memory can capture general or domain-specific knowledge.

In this work, we present ***Memory Decoder at Scale***, a systematic study of parametric memory at LLM pretraining scale. We examine how memory scales with model and data size, how parameters should be allocated between memory and the LM, and whether pretrained memory can capture general or domain-specific knowledge. We scale memory up to 6.9B parameters, with the largest setting pretrained on 300B tokens. We find that downstream performance improves with memory capacity. Pairing larger memories with smaller LMs is also more parameter-efficient than scaling the LM alone. To support pretraining at this scale, we develop a distributed FAISS pipeline that addresses indexing and search costs through embedding compression, index sharding, and parallel search. Memory pretraining additionally uses sparse kNN distribution storage to reduce space requirements and distributed streaming to read only the entries required by each batch. Figure 2 summarizes memory pretraining and parallel inference with a **frozen LM**.

We evaluate both general and domain memory:

- **General memory.** Across 17 benchmarks, pairing the frozen Pythia-410M LM with a 6.9B memory raises the average score from 29.86 to 37.34. This configuration surpasses Pythia-12B at 37.24 while using 39% fewer total parameters (Figure 1).
- **Domain memory.** Across Qwen3 LMs from 0.6B to 14B, 1.7B domain memories improve the average over BioInst, LawBench, and FinEval by more than 9 points and outperform the strongest reported baseline among CPT, LoRA, and RAG by 4.05–8.53 points on average. After transfer across vocabularies using only 20% of the standard memory training budget, these memories improve the domain averages of OLMo-2-7B and OLMo-3-7B by 4.26 and 7.77 points, respectively.

Further analyses show memory remains effective under few-shot prompting, benefits knowledge-intensive tasks most, and improves with training budget.

2 Preliminary: Memory Decoder

Memory Decoder (Cao et al., 2026) is a standalone parametric memory trained to imitate the behavior of a non-parametric retriever. For a sequence $x = (x_1, \ldots, x_T)$ over vocabulary $\mathcal{V}$, each position t has context $c_t = x_{<t}$ and observed next token x_t. A frozen base language model M_θ encodes this context as a query. Each datastore key is a corpus context representation from M_θ, and its value is the observed next token. The retriever searches the datastore with the query and forms the kNN distribution $p_{\mathrm{ret}}(\cdot \mid c_t)$ from the values of the K nearest keys and their normalized distance weights.

Unlike language modeling with a single observed target x_t, p_{ret} offers richer supervision by capturing the diversity of plausible continuations. A transformer decoder M_ψ predicts $p_\psi(\cdot \mid c_t)$ directly from the original context without taking retrieved text as input. M_ψ and M_θ use the same tokenizer and output vocabulary. Retrieval is used only to construct offline supervision for memory pretraining.

Memory Decoder combines distribution alignment with the corpus language modeling objective (Bengio et al., 2003). The loss for each position is

$$\mathcal{L}_{\mathrm{mem}} = \beta D_{\mathrm{KL}}(p_{\mathrm{ret}}(\cdot \mid c_t) \,\|\, p_\psi(\cdot \mid c_t)) + (1-\beta)\big[-\log p_\psi(x_t \mid c_t)\big], \tag{1}$$

where D_{KL} denotes the KL divergence (Van Erven & Harremos, 2014) and $\beta \in [0,1]$ controls the contribution of retrieval supervision. The KL term trains M_ψ to imitate the kNN distribution. The language modeling term prevents excessive deviation from the underlying corpus distribution.

At inference, M_θ and M_ψ process the same context in parallel. Let $p_\theta(\cdot \mid c_t)$ and $p_\psi(\cdot \mid c_t)$ denote their output distributions. The predictions are combined as

$$p_{\mathrm{final}}(y \mid c_t) = (1-\alpha)p_\theta(y \mid c_t) + \alpha p_\psi(y \mid c_t), \qquad y \in \mathcal{V}, \tag{2}$$

where $\alpha \in [0,1]$ controls the memory contribution. Through this interpolation, the final prediction incorporates the retrieval behavior learned by M_ψ during memory pretraining.

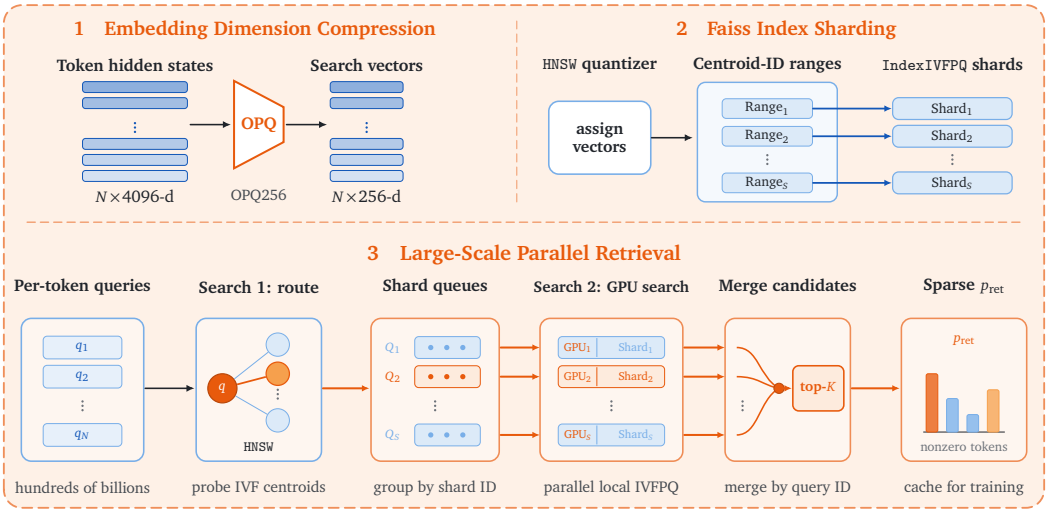

Figure 3: Distributed Faiss pipeline for constructing kNN distribution at pretraining scale. (1) An OPQ transform compresses contextual representations before indexing. (2) An IVF index with an HNSW quantizer partitions the compressed representations into `IndexIVFPQ` shards over contiguous centroid ranges. (3) The quantizer routes queries to the relevant GPU shards, and writer workers merge local candidates into the final K neighbors and store p_{ret} as sparse training targets.

3 Memory Decoder at Scale

Scaling Memory Decoder requires constructing retrieval supervision for every training context while retaining the same objective and inference interface. The central challenge is a data construction problem: the contextual representations used as datastore entries and the contexts used as queries both grow with the training corpus. Section 3.1 introduces a distributed Faiss pipeline that separates global routing from local search and materializes the results as sparse training targets. Section 3.2 then describes how these targets are stored and streamed into memory pretraining. The same training procedure applies to general memories and domain memories.

3.1 Large-Scale kNN Distribution Construction

Let $\mathcal{P}_{\text{mem}}$ denote the set of context and target pairs extracted from the training corpus $\mathcal{D}_{\text{mem}}$. Each position t defines a context $c_t = x_{<t}$ and a target token x_t. We derive retrieval keys from the frozen base language model M_θ. For each context c_t, $\phi_\theta(c_t)$ denotes the hidden state from the final layer of M_θ at the final position of c_t. We set $k_t = \phi_\theta(c_t)$, yielding the datastore

$$\mathcal{R}_{\text{mem}} = \{(k_t, x_t) \mid k_t = \phi_\theta(c_t),\ (c_t, x_t) \in \mathcal{P}_{\text{mem}}\}. \tag{3}$$

For a query context $c_t = x_{<t}$, let $q_t = \phi_\theta(c_t)$. We retrieve its K nearest datastore entries $\mathcal{N}_K(q_t) \subset \mathcal{R}_{\text{mem}}$ under distance d, discarding the closest entry if its key equals q_t to avoid trivial self-retrieval. Each retrieved pair (k_i, x_i) is assigned a normalized weight

$$\lambda_i(q_t) = \frac{\exp(-d(q_t, k_i)/\tau)}{\sum_{(k_j,x_j)\in\mathcal{N}_K(q_t)} \exp(-d(q_t, k_j)/\tau)}, \qquad (k_i, x_i) \in \mathcal{N}_K(q_t), \tag{4}$$

where τ is a temperature. The kNN distribution is a weighted vote over the retrieved next tokens

$$p_{\text{ret}}(y \mid c_t) = \sum_{(k_i,x_i)\in\mathcal{N}_K(q_t)} \lambda_i(q_t)\mathbf{1}[x_i = y], \qquad y \in \mathcal{V}. \tag{5}$$

Neighbors with the same target token contribute to the same probability mass, so the support of p_{ret} remains sparse over $\mathcal{V}$. The resulting distribution serves as the non-parametric supervision distilled into M_ψ.

Scaling Bottleneck. Constructing p_{ret} for every training context requires both the datastore and the query stream to grow with the training corpus. A centralized FAISS index cannot accommodate

the resulting storage and search workload, while naive index sharding would require each query to access every shard. We address this limitation by coupling distributed index construction with routed local search, so that the index partition determines query dispatch, local retrieval, and result aggregation. The overall design is illustrated in Figure 3.

Distributed Index Construction. We apply a learned Optimized Product Quantization transform (Ge et al., 2013) to compress the contextual representations before indexing. The same transform is applied to queries, placing keys and queries in the same compressed space. An IVF partition organizes the compressed representations by centroid, and contiguous centroid ranges are assigned to independent `IndexIVFPQ` shards (Jegou et al., 2011). At search time, the probed centroids determine which shards process each query. This organization reduces the storage and search cost of each shard while allowing the shards to be constructed and searched independently.

Routed Parallel Search. Naively sharding the datastore does not make retrieval scalable, because each query would still need to be searched against every shard. We therefore decompose retrieval into two stages. *Search 1* performs lightweight global routing: an `IndexHNSW` quantizer (Malkov & Yashunin, 2018) identifies the IVF centroids probed by each query and routes it only to the shards that own those centroids. *Search 2* performs local retrieval: queries routed to the same shard are batched and searched against its local `IndexIVFPQ` index. The two stages are sequential per query, while local retrieval runs in parallel across shards. Because the shards own disjoint centroid ranges and each search accesses only shard-local index state, GPU workers can process their query batches independently. Writer workers merge candidates with the same query identifier, retain the final K neighbors, and aggregate their values into $p_{\mathrm{ret}}(\cdot \mid c_t)$.

3.2 Memory Pretraining

Memory pretraining uses the constructed kNN distributions as offline supervision for M_ψ. General and domain memories share the same architecture, objective, and data pipeline, but use kNN targets constructed from broad pretraining data and the corresponding domain corpus, respectively. At pretraining scale, storing a dense distribution over the vocabulary for every token is prohibitive, while loading irregular target rows can disrupt the language model training pipeline. We address these constraints by storing only the support of each kNN distribution and aligning distributed streaming with the packed training examples.

Sparse kNN Distribution Storage. The support of $p_{\mathrm{ret}}(\cdot \mid c_t)$ contains at most K token identifiers and becomes smaller when multiple neighbors share the same target token. We aggregate duplicate targets in higher precision and remove negligible probabilities using a small threshold ϵ, which gives

$$\mathcal{S}_t = \{y \in \mathcal{V} \mid p_{\mathrm{ret}}(y \mid c_t) > \epsilon\}. \tag{6}$$

Only the token identifiers and probabilities in $\mathcal{S}_t$ are stored in sharded flat arrays, with offsets marking the boundary of each distribution row. For N rows and M retained token and probability pairs, storage scales as $\mathcal{O}(N + M)$, rather than $\mathcal{O}(N|\mathcal{V}|)$ for dense distributions. [1] This representation requires no padding and preserves direct access to every row, while the memory model continues to predict over the complete vocabulary.

Distributed Streaming of kNN Distributions. During preprocessing, each packed language model example records the aligned range of distribution rows. Because this alignment is part of the example, dataset shuffling does not separate tokens from their retrieval targets. Distributed workers resolve the stored offsets and use memory mapping to read only the required contiguous slices from the corresponding shards. The collator retains the standard language modeling labels and represents p_{ret} as sparse coordinate triplets, so memory pretraining uses the same batching interface as language model pretraining. On the accelerator, the retained probabilities are renormalized before the KL divergence in Equation 1 is computed. The training pipeline therefore consumes retrieval supervision for the entire corpus without reconstructing dense vocabulary distributions.

[1] On the Pile, each row retains 64.95 token and probability pairs on average for a $50{,}304$-token vocabulary. Accounting for amortized INT64 row offsets, INT32 token identifiers and labels, and FP16 probabilities, the sparse representation requires approximately 402 bytes per row, compared with $100{,}608$ bytes for a dense FP16 distribution, yielding an approximately $250\times$ reduction.

Table 1: General memory scaling across Pythia backbones on 17 tasks. Scores are percentages. Base columns report frozen backbones, while +Mem-x.xB columns add an x.xB memory to the corresponding backbone. Each memory is trained on 300B tokens, matching the training budget of its backbone. † marks LAMBADA (OpenAI), and GPQA-main scores average 10 random seeds.

Benchmark (Metric)	1.4B		2.8B		6.9B		12B
	Base	+Mem-1.4B	Base	+Mem-2.8B	Base	+Mem-6.9B	Base
General Tasks							
ARC-Easy (Acc.)	61.83	62.42	63.64	65.82	68.39	70.12	70.58
ARC-Challenge (Acc.)	27.39	27.56	30.03	30.29	33.11	35.67	33.45
LAMBADA† (Acc.)	61.91	63.30	65.09	65.75	68.76	69.94	70.99
LogiQA (Acc.)	22.43	22.73	21.66	21.97	23.04	23.04	21.97
PIQA (Acc.)	72.20	73.23	74.05	74.92	76.01	76.17	76.28
SciQ (Acc.)	86.30	88.00	88.10	90.20	90.90	91.50	92.70
WinoGrande (Acc.)	56.12	59.43	58.56	61.25	62.98	64.72	65.59
Knowledge							
MMLU (Acc.)	23.76	24.75	24.72	24.87	26.26	26.95	25.67
NQ-Open (EM)	2.80	3.88	3.82	5.15	4.68	6.32	6.32
TriviaQA (EM)	8.49	10.08	8.30	17.11	22.21	25.58	26.80
PopQA (Acc.)	10.65	11.90	11.45	12.83	14.00	15.13	14.95
2WikiMultiHopQA (EM)	16.89	22.57	18.96	21.64	20.60	21.11	20.57
Bamboogle (EM)	1.60	2.40	0.80	2.40	4.00	5.60	2.40
HotpotQA (EM)	6.75	8.44	8.83	9.49	8.97	11.78	11.11
GPQA-main (Acc.)	24.53	27.90	24.91	25.92	24.11	26.36	25.49
Hallucination							
TruthfulQA (Acc.)	30.63	30.46	28.47	27.92	28.49	27.81	26.91
HaluEval (Acc.)	42.67	45.13	44.68	45.83	40.53	44.59	41.23
AVG	32.76	**34.36** (+1.60)	33.89	**35.49** (+1.60)	36.30	**37.79** (+1.49)	37.24

4 EXPERIMENTS

Overview. We organize our evaluation of pretrained parametric memory into four parts: (1) experimental setup (§4.1), (2) the effectiveness of general memory at matched backbone and memory scales, together with its transfer across backbone scales (§4.2), (3) the transferability of domain memory across backbone scales, model families, and vocabularies (§4.3), and (4) analyses of few-shot robustness, knowledge gains, and the memory training objective (§5). Appendix E further examines memory capacity, training budget, extractable memorization, and retrieval fidelity.

4.1 EXPERIMENTAL SETUP

Datasets, Backbones, and Baselines. For general memory, we train 1.4B, 2.8B, and 6.9B memories on the deduplicated Pile (207B tokens) (Gao et al., 2020) and evaluate them with deduplicated Pythia backbones (410M–12B) (Biderman et al., 2023) on 17 benchmarks covering general abilities, knowledge, and factuality. We compare against decoder-only Pythia under matched total parameter counts and training budgets. For domain memory, we train 1.7B memories on Biology-Instructions, DISC-Law-SFT, and FinTrain, and evaluate on BioInst, LawBench, and FinEval, respectively. Backbones include Qwen3 Base (0.6B–14B) (Yang et al., 2025) and OLMo-2/3 (7B) (OLMo et al., 2024; Olmo et al., 2025). Baselines are CPT with the same training budget, parameter-matched LoRA, and top-5 RAG. Appendices A.1 and B.1 provide full dataset, backbone, and baseline descriptions.

Implementation Details. General memories train for 300B tokens using Megatron-LM (Shoeybi et al., 2019). We use 2,000 warmup steps for general memory. We use AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay 0.01, and cosine decay to 10% of the peak learning rate. Domain memories are initialized from Qwen3-1.7B-Base and trained with a peak learning rate of 3×10^{-4} under a compute budget matching one Qwen3-8B-Base CPT epoch. Base models remain frozen when paired with memory. We combine predictions using Eq. 2, with α selected on validation data when available and fixed for testing. Appendices B and C provide full training and evaluation settings.

Table 2: Domain memory results with Qwen3 backbones under zero-shot domain evaluation. BioInst, LawBench, and FinEval use their respective benchmark metrics. Score changes relative to the corresponding frozen backbone are shown in smaller type. +Mem-1.7B denotes a 1.7B domain memory attached to a frozen Qwen3 backbone.

Model	BioInst	LawBench	FinEval	Avg
		Qwen3 Family		
Qwen3-0.6B-Base	5.78	17.87	23.31	15.65
+*CPT*	7.94 +2.16	20.39 +2.52	14.90 -8.41	14.41(-1.24)
+*LoRA*	10.36 +4.58	21.10 +3.23	19.54 -3.77	17.00(+1.35)
+*RAG*	2.96 -2.82	17.82 -0.05	21.37 -1.94	14.05(-1.60)
+*Mem-1.7B*	**19.94** +14.16	**26.53** +8.66	**30.12** +6.81	**25.53**(+9.88)
Qwen3-1.7B-Base	5.39	26.86	27.26	19.84
+*CPT*	4.84 -0.55	28.86 +2.00	29.02 +1.76	20.91(+1.07)
+*LoRA*	9.01 +3.62	28.53 +1.67	**38.74** +11.48	25.43(+5.59)
+*RAG*	8.07 +2.68	29.61 +2.75	36.63 +9.37	24.77(+4.93)
+*Mem-1.7B*	**23.21** +17.82	**30.78** +3.92	34.45 +7.19	**29.48**(+9.64)
Qwen3-4B-Base	5.02	27.47	37.43	23.31
+*CPT*	9.14 +4.12	36.10 +8.63	30.22 -7.21	25.15(+1.85)
+*LoRA*	8.06 +3.04	33.65 +6.18	33.52 -3.91	25.08(+1.77)
+*RAG*	8.34 +3.32	32.62 +5.15	39.13 +1.70	26.70(+3.39)
+*Mem-1.7B*	**23.48** +18.46	**36.89** +9.42	**39.54** +2.11	**33.30**(+10.00)
Qwen3-8B-Base	4.82	32.67	43.51	27.00
+*CPT*	9.79 +4.97	39.24 +6.57	40.00 -3.51	29.68(+2.68)
+*LoRA*	5.32 +0.50	33.55 +0.88	42.43 -1.08	27.10(+0.10)
+*RAG*	7.78 +2.96	40.67 +8.00	**46.97** +3.46	31.81(+4.81)
+*Mem-1.7B*	**20.02** +15.20	**41.91** +9.24	46.33 +2.82	**36.09**(+9.09)
Qwen3-14B-Base	4.01	35.45	44.25	27.90
+*RAG*	7.70 +3.69	44.38 +8.93	42.81 -1.44	31.63(+3.73)
+*Mem-1.7B*	**21.97** +17.96	**44.42** +8.97	**47.29** +3.04	**37.89**(+9.99)

4.2 General Memory

General Memory at Scale. In Table 1, adding a memory of the same size to each frozen Pythia backbone raises AVG on 17 benchmarks from 32.76 to 34.36 at 1.4B, 33.89 to 35.49 at 2.8B, and 36.30 to 37.79 at 6.9B. The 1.4B backbone with 1.4B memory outperforms the 2.8B base (34.36 vs. 33.89) at matched total parameters and training budgets. The 6.9B backbone with 6.9B memory also exceeds the frozen 12B base (37.79 vs. 37.24). Memory improves 47 of 51 task-scale combinations and matches one more. Knowledge gains include TriviaQA at 2.8B (8.30 $\rightarrow$ 17.11), 2WikiMultiHopQA at 1.4B (16.89 $\rightarrow$ 22.57), and HotpotQA at 6.9B (8.97 $\rightarrow$ 11.78). Gains also extend beyond knowledge tasks, with improvements of 3.31 on WinoGrande and 3.37 on GPQA-main at 1.4B, and 4.06 on HaluEval at 6.9B.

Transfer between Different LM Sizes. Figure 1 shows that all 18 cross-scale Pythia configurations improve zero-shot AVG over their frozen backbones without memory retraining. Many exceed the base-model scaling trend. Memory and backbone capacities need not match, with larger memories particularly benefiting smaller backbones. A 410M backbone with 6.9B memory raises AVG from 29.86 to 37.34, surpassing the 12B base (37.24) with 39% fewer total parameters. With memory training exposure fixed, the Base + Memory curve matches the AVG of the 2.8B, 6.9B, and 12B bases using 33%, 32%, and 42% fewer parameters, respectively. These results favor reusable memory over scaling the backbone alone. Both components can run in parallel, potentially lowering inference latency at matched total parameter counts. Appendix D reports individual task results.

4.3 Domain Memory

Transfer between Different LM Sizes. Table 2 evaluates 1.7B domain memories on BioInst, LawBench, and FinEval with frozen Qwen3 backbones from 0.6B to 14B. Memory improves all 15 backbone-domain combinations and achieves the best three-domain average at every scale. AVG

gains are 9.88, 9.64, 10.00, 9.09, and 9.99 points for the 0.6B, 1.7B, 4B, 8B, and 14B backbones, respectively. Memory exceeds the strongest baseline by 4.05–8.53 points, with the largest margin at 0.6B. At 14B, gains remain 17.96, 8.97, and 3.04 points on BioInst, LawBench, and FinEval, respectively. These results demonstrate reusable domain knowledge across backbone scales. Switching domains only requires activating the corresponding memory at inference, while the backbone remains frozen and requires no continued pretraining. Appendix F reports results for 0.6B memories.

Transfer Across Model Families and Vocabularies. We adapt the Qwen3 memories for biology, law, and finance from Table 2 on datastores using the OLMo-3 vocabulary. With only 20% of the standard memory training budget, the memories achieve the best average for both OLMo backbones in Table 3. AVG rises from 19.57 to 23.83 on OLMo-2-7B and from 18.67 to 26.44 on OLMo-3-7B, gains of 4.26 and 7.77 points. Memory improves all six evaluations, gaining 4.09 and 1.77 points on BioInst, 8.52 and 9.21 on LawBench, and 0.18 and 12.33 on FinEval for OLMo-2 and OLMo-3, respectively. CPT leads on BioInst for both backbones and LawBench for OLMo-2, while memory leads on average for both backbones.

Table 3: Domain memory transfer across vocabularies to OLMo backbones. Training uses 20% of the budget in Table 2, with the same evaluation setting.

Model	BioInst	LawBench	FinEval	Avg
OLMo Family				
OLMo-2-7B	3.90	6.37	48.43	19.57
+*CPT*	**9.51**	**22.95**	34.61	22.36(+2.79)
+*LoRA*	9.44	13.21	32.34	18.33(-1.24)
+*RAG*	3.70	12.11	40.88	18.90(-0.67)
+*Mem-1.7B*	7.98	14.89	**48.61**	**23.83**(+4.26)
OLMo-3-7B	6.39	13.19	36.43	18.67
+*CPT*	**13.69**	20.29	38.89	24.29(+5.62)
+*LoRA*	6.95	16.40	28.21	17.19(-1.48)
+*RAG*	5.95	21.85	33.10	20.30(+1.63)
+*Mem-1.7B*	8.16	**22.40**	**48.76**	**26.44**(+7.77)

5 Analysis

5.1 Few-Shot Robustness and Knowledge Task Improvements

Figure 4 compares zero-, three-, and five-shot AVG on the same 13 tasks at three model scales, with Bamboogle kept zero-shot (Appendix C.1). Memory improves AVG by 1.43–1.87, 1.39–1.52, and 1.22–1.62 points, respectively, with gains at every scale and prompt setting, indicating that it complements in-context examples. It also improves all eight zero-shot knowledge tasks at every scale (Table 1). The bottom row reports zero-shot results on TriviaQA, 2WikiMultiHopQA, and

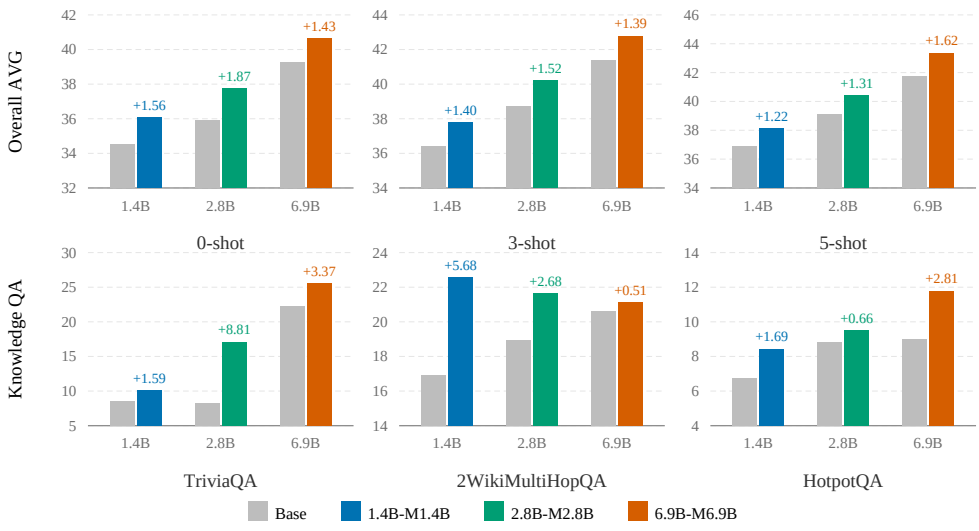

Figure 4: **Top:** AVG over the same 13 tasks under zero-shot, three-shot, and five-shot settings. **Bottom:** Zero-shot results on TriviaQA, 2WikiMultiHopQA, and HotpotQA. Gray bars denote frozen backbones. The notation $x.x$B-M$x.x$B represents an $x.x$B backbone paired with an $x.x$B memory.

HotpotQA, whose mean gains over scales are 4.59, 2.96, and 1.72 points, respectively. The benefits are not driven by a single benchmark. These open-domain and multi-hop QA gains suggest that memory supplies factual evidence beyond the prompt.

5.2 Memory Training Objective

To isolate the memory objective, we compare our 1.7B memory with an attached Qwen3-1.7B-Base CPT model, matching domain training data, FLOPs, capacity, and the interpolation interface. With frozen Qwen3-Base backbones of 1.7B and 8B, memory exceeds attached CPT by 10.21 and 8.71 points on BioInst and 1.68 and 3.93 on LawBench, respectively (Table 4). These consistent gains support the contribution of the memory objective beyond the interpolation interface.

Table 4: Attached module control with frozen Qwen3 backbones. CPT and memory modules are both 1.7B.

Domain	Module	1.7B	8B
BioInst	None	5.39	4.82
	CPT	13.00	11.31
	Memory	**23.21**	**20.02**
LawBench	None	26.86	32.67
	CPT	29.10	37.98
	Memory	**30.78**	**41.91**

6 Related Work

Memory-Augmented Pretraining. Most memory methods augment language models at inference time, whereas few incorporate memory into pretraining. RETRO (Borgeaud et al., 2022) conditions an autoregressive model on retrieved chunks through a retrieval encoder and cross-attention. TRIME (Zhong et al., 2022) trains with accessible in-batch examples, enabling adaptation to local, long-term, and external memories at inference time. Limited Memory Language Models (LMLMs) (Zhao et al., 2025) annotate the pretraining corpus with database lookups and mask retrieved factual values from the loss. These methods use non-parametric memory at inference time. Our work instead distills the memory signal learned during pretraining into a standalone parametric module, requiring neither external memory construction nor online retrieval at inference time.

Short-term and Long-term Memory. Short-term memory improves access to input context through segment-level recurrence (Dai et al., 2019), bounded KV caches (Xiao et al., 2024b), retrieval from distant context (Xiao et al., 2024a), and RoPE rescaling (Ding et al., 2024). These methods focus primarily on memory within the input context, whereas our work learns long-term memory from the entire pretraining corpus. Long-term memory can be non-parametric or parametric. kNN-LM (Khandelwal et al., 2019) interpolates model predictions with a distribution over next tokens of retrieved training contexts, whereas RAG (Lewis et al., 2020) conditions generation on retrieved passages. Both require external retrieval at inference time, incurring storage and search costs, with additional context processing for RAG. Parametric memory encodes knowledge in learned weights. Memory Decoder (Cao et al., 2026) and MLP Memory (Wei et al., 2025) distill retrieval distributions into plug-and-play neural memories, while Titans (Behrouz et al., 2024) encodes information from historical context at test time. These approaches learn memory at small scale or update it online at test time. Our work instead learns long-term memory at pretraining scale.

7 Conclusion

In this work, we ask whether long-term memory can be scaled as an independent component of language models. ***Memory Decoder at Scale*** provides evidence that it can. We scale external parametric memory to 6.9B parameters and pretrain it on up to 300B tokens, supported by distributed infrastructure for large-scale kNN indexing, retrieval, storage, and streaming.

More importantly, our experiments consistently show that pairing larger memories with smaller base models outperforms scaling the base model alone. A 6.9B general memory enables a frozen Pythia-410M backbone to surpass Pythia-12B with 39% fewer total parameters, while domain-specific memories provide substantial gains across backbone scales and transfer across model families and vocabularies. These findings suggest a different scaling path for language models, in which knowledge and reasoning can be scaled, specialized, and reused more independently, paving the way toward disentangling long-term memory from reasoning.

AI USE STATEMENT

We used generative AI tools, including ChatGPT and Codex, to assist with manuscript drafting, language polishing, organization, and LaTeX formatting. The authors retain full responsibility for the research design, experimental procedures, analysis, and all content and claims in this paper.

ETHICS STATEMENT

Our work uses existing language models, corpora, and benchmarks. Parametric memories may inherit biases and retain sensitive information from their training data. Our extractable memorization analysis in Appendix E.3 shows increased recoverability of training content under the tested protocols, underscoring the need to consider data provenance and privacy when training and deploying memory models.

REPRODUCIBILITY STATEMENT

We describe our method in Section 3 and summarize the experimental setup in Section 4.1. Appendices A–C provide detailed descriptions of the datasets, implementation, training settings, and evaluation protocols. Section 5 and Appendix E report our analyses and ablation studies.

REFERENCES

Alan D Baddeley and Elizabeth K Warrington. Amnesia and the distinction between long-and short-term memory. *Journal of verbal learning and verbal behavior*, 9(2):176–189, 1970.

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time, 2024.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International conference on machine learning*, pp. 2397–2430. PMLR, 2023.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pp. 2206–2240. PMLR, 2022.

Jiaqi Cao, Jiarui Wang, Rubin Wei, Qipeng Guo, Kai Chen, Bowen Zhou, and Zhouhan Lin. Memory decoder: A pretrained, plug-and-play memory for large language models. *Advances in Neural Information Processing Systems*, 38:115487–115510, 2026.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX security symposium (USENIX Security 21)*, pp. 2633–2650, 2021.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations*, 2022.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. Longrope: Extending llm context window beyond 2 million tokens. *arXiv preprint arXiv:2402.13753*, 2024.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library. *IEEE Transactions on Big Data*, 2025.

Bangde Du, Minghao Guo, Songming He, Ziyi Ye, Xi Zhu, Weihang Su, Shuqi Zhu, Yujia Zhou, Yongfeng Zhang, Qingyao Ai, et al. Twinvoice: A multi-dimensional benchmark towards digital twins via llm persona simulation. *arXiv preprint arXiv:2510.25536*, 2025.

Damien A. Fair, Alexander L. Cohen, Jonathan D. Power, Nico U. F. Dosenbach, Jessica A. Church, Francis M. Miezin, Bradley L. Schlaggar, and Steven E. Petersen. Functional brain networks develop from a "local to distributed" organization. *PLoS Computational Biology*, 5(5):e1000381, May 2009. doi: 10.1371/journal.pcbi.1000381.

Zhiwei Fei, Xiaoyu Shen, Dawei Zhu, Fengzhe Zhou, Zhuo Han, Alan Huang, Songyang Zhang, Kai Chen, Zhixin Yin, Zongwen Shen, et al. Lawbench: Benchmarking legal knowledge of large language models. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 7933–7962, 2024.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Tiezheng Ge, Kaiming He, Qifa Ke, and Jian Sun. Optimized product quantization for approximate nearest neighbor search. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2946–2953, 2013.

Minghao Guo, Qingcheng Zeng, Xujiang Zhao, Yanchi Liu, Wenchao Yu, Mengnan Du, Haifeng Chen, and Wei Cheng. Deepsieve: Information sieving via llm-as-a-knowledge-router. In *Findings of the Association for Computational Linguistics: EACL 2026*, 2026.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 8342–8360, 2020.

Jamie Hayes, Marika Swanberg, Harsh Chaudhari, Itay Yona, Ilia Shumailov, Milad Nasr, Christopher A Choquette-Choo, Katherine Lee, and A Feder Cooper. Measuring memorization in language models via probabilistic extraction. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 9266–9291, 2025.

Haonan He, Yuchen Ren, Yining Tang, Ziyang Xu, Junxian Li, Minghao Yang, Di Zhang, Dong Yuan, Tao Chen, Shufei Zhang, et al. Biology-instructions: A dataset and benchmark for multi-omics sequence understanding capability of large language models. *arXiv preprint arXiv:2412.19191*, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, 2020.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Herve Jegou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128, 2011.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, 2017.

Zixuan Ke, Yifei Ming, Xuan-Phi Nguyen, Caiming Xiong, and Shafiq Joty. Demystifying domain-adaptive post-training for financial llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 31021–31047, 2025.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*, 2019.

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

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Junyi Li, Xiaoxue Cheng, Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. Halueval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 6449–6464, 2023.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 3214–3252, 2022.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*, 2020.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Yu A Malkov and Dmitry A Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836, 2018.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 9802–9822, 2023.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc-Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 1525–1534, 2016.

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

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

Larry R Squire. Memory and brain systems: 1969–2009. *Journal of Neuroscience*, 29(41):12711–12716, 2009.

Qwen Team. Qwen3. 5-omni technical report. *arXiv preprint arXiv:2604.15804*, 2026.

Tim Van Erven and Peter Harremos. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014.

Rubin Wei, Jiaqi Cao, Jiarui Wang, Jushi Kai, Qipeng Guo, Bowen Zhou, and Zhouhan Lin. Mlp memory: A retriever-pretrained memory for large language models. *arXiv preprint arXiv:2508.01832*, 2025.

Johannes Welbl, Nelson F Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pp. 94–106, 2017.

Chaojun Xiao, Pengle Zhang, Xu Han, Guangxuan Xiao, Yankai Lin, Zhengyan Zhang, Zhiyuan Liu, and Maosong Sun. Infllm: Training-free long-context extrapolation for llms with an efficient context memory. *Advances in neural information processing systems*, 37:119638–119661, 2024a.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, volume 2024, pp. 21875–21895, 2024b.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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

Xinlei Yu, Chengming Xu, Guibin Zhang, Zhangquan Chen, Yudong Zhang, Yongbo He, Peng-Tao Jiang, Jiangning Zhang, Xiaobin Hu, and Shuicheng Yan. Vismem: Latent vision memory unlocks potential of vision-language models. *arXiv preprint arXiv:2511.11007*, 2025.

Shengbin Yue, Wei Chen, Siyuan Wang, Bingxuan Li, Chenchen Shen, Shujun Liu, Yuxuan Zhou, Yao Xiao, Song Yun, Xuanjing Huang, et al. Disc-lawllm: Fine-tuning large language models for intelligent legal services. *arXiv preprint arXiv:2309.11325*, 2023.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. Glm-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, et al. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025.

Linxi Zhao, Sofian Zalouk, Christian K Belardi, Justin Lovelace, Jin Peng Zhou, Ryan Thomas Noonan, Dongyoung Go, Kilian Q Weinberger, Yoav Artzi, and Jennifer J Sun. Pre-training limited memory language models with internal and external knowledge. *arXiv preprint arXiv:2505.15962*, 2025.

Zexuan Zhong, Tao Lei, and Danqi Chen. Training language models with memory augmentation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 5657–5673, 2022.

APPENDIX

A TRAINING DATASETS

Table 5 summarizes the corpora used to train the memory modules. The general memory uses the deduplicated Pile directly as a language modeling corpus. For domain memories, we first normalize each domain dataset into plain continued pretraining text before tokenization. BioInst is converted from biology instruction data into task-oriented training text. DISC-Law-SFT is deduplicated and formatted as legal domain instruction text, preserving supporting legal materials when available. FinTrain-unsup is drawn from the unsupervised financial text subset and chunked as plain CPT data. We append the Qwen3 EOS token to each training example and pack the resulting corpus with a block size of 4096 tokens.

Table 5: Training datasets used for memory pretraining.

Memory	**Dataset**	**Source**	**Tokens**
General	The Pile deduplicated	EleutherAI/the_pile_deduplicated	207B
Domain	BioInst	hhnqqq/Biology-Instructions	552M
Domain	DISC-Law-SFT	ShengbinYue/DISC-Law-SFT	125M
Domain	FinTrain-unsup	Salesforce/FinTrain	4.4B

A.1 TRAINING CORPORA AND EVALUATION BENCHMARKS

The general memory is trained on the deduplicated Pile (Gao et al., 2020), which contains 207B tokens. We evaluate it across three task categories: general tasks from the Pythia suite (Biderman et al., 2023), including ARC-Easy, ARC-Challenge (Clark et al., 2018), LAMBADA (Paperno et al., 2016), LogiQA (Liu et al., 2020), PIQA (Bisk et al., 2020), SciQ (Welbl et al., 2017), and Wino-Grande (Sakaguchi et al., 2021); knowledge-intensive tasks, including MMLU (Hendrycks et al., 2020), Natural Questions (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023), 2WikiMultiHopQA (Ho et al., 2020), Bamboogle (Press et al., 2023), HotpotQA (Yang et al., 2018), and GPQA-main (Rein et al., 2023); and factuality and hallucination robustness on TruthfulQA (Lin et al., 2022) and HaluEval (Li et al., 2023). We evaluate domain memories in biology, law, and finance. For biology, we train the memory on Biology-Instructions (He et al., 2024) after reformatting the dataset as unsupervised text and evaluate it on the Biology-Instructions benchmark (He et al., 2024). For law, we use the same procedure with DISC-Law-SFT (Yue et al., 2023) for training and LawBench (Fei et al., 2024) for evaluation. For finance, we train the memory on the book and finance subsets of FinTrain (Ke et al., 2025) and evaluate it on FinEval (Ke et al., 2025).

B IMPLEMENTATION DETAILS

All memory modules are trained separately from the frozen base backbone and are combined with the backbone distribution only at inference time. The general memory runs use GPT-NeoX configurations from the Pythia family and scale the memory from 1.4B to 6.9B parameters. We report these settings in Table 6. For domain memories, we initialize from Qwen3-1.7B-Base and list the shared training hyperparameters in Table 7.

B.1 BACKBONES AND BASELINES

We evaluate parametric memory across diverse model families and scales. Throughout the paper, we use deduplicated Pythia checkpoints (Biderman et al., 2023). For general memory experiments, we evaluate frozen backbones from 410M to 12B parameters and train 1.4B, 2.8B, and 6.9B memories initialized from the corresponding checkpoints. Domain memory experiments use Qwen3 Base backbones ranging from 0.6B to 14B parameters (Yang et al., 2025). To evaluate transfer across vocabularies, we use OLMo-2-1124-7B (OLMo et al., 2024) and OLMo-3-1025-7B (Olmo et al.,

2025) backbones. For general memory, we compare Base + Memory configurations against decoder-only Pythia models under matched total parameter counts and training budgets. For domain memory, we compare against three baselines. CPT (Gururangan et al., 2020) trains the backbone on the corresponding corpus using the same training budget as memory. LoRA (Hu et al., 2022) is applied to the query, key, value, and MLP layers, with its rank adjusted to match the parameter count of our memory modules. RAG (Lewis et al., 2020) employs Qwen3-Embedding-0.6B (Zhang et al., 2025) as the retrieval model and retrieves the top-5 documents.

B.2 Training and Evaluation Settings

All experiments are conducted on 256 NVIDIA A800 80GB GPUs with Megatron-LM (Shoeybi et al., 2019). For general memory, we construct the datastore from the deduplicated Pile using Pythia-6.9B and train 1.4B, 2.8B, and 6.9B memories for 300B tokens by default, matching the Pythia pretraining budget and corresponding to approximately 1.5 epochs over the datastore. The 1.4B, 2.8B, and 6.9B memories use peak learning rates of 3×10^{-4}, 2.5×10^{-4}, and 2×10^{-4}, respectively. All runs use AdamW (Loshchilov & Hutter, 2017) with $\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay 0.01, cosine decay to 10% of the peak learning rate, and $2{,}000$ warmup steps. For domain memory, we perform one epoch of continued pretraining with Qwen3-4B-Base on each domain corpus and use the model to construct the datastore. Each memory is initialized from Qwen3-1.7B-Base and trained with a peak learning rate of 3×10^{-4} under a standard budget matching the compute of one Qwen3-8B-Base CPT epoch. For transfer across vocabularies, a trained Qwen3 memory is adapted to OLMo-2-1124-7B and OLMo-3-1025-7B by replacing the embedding layer and language model head, followed by continued training on an OLMo-3-1025-7B datastore using 20% of the standard memory training budget. We follow the standard metric for each benchmark. We use exact match for NQ-Open, TriviaQA, 2WikiMultiHopQA, Bamboogle, and HotpotQA, and standard accuracy metrics for the remaining general benchmarks. For domain evaluation, we report the mean over the metrics defined for BioInst and LawBench, while FinEval combines exact match, Matthews correlation coefficient (MCC), and ROUGE-1 across tasks. When validation data is available, the interpolation coefficient α is tuned on the validation split and then fixed for test evaluation.

Table 6: Implementation details for general memories.

Configuration / Hyperparameter	**Memdec-1.4B**	**Memdec-2.8B**	**Memdec-6.9B**
Model configuration			
Architecture	GPT-NeoX	GPT-NeoX	GPT-NeoX
Hidden size	2048	2560	4096
Intermediate size	8192	10240	16384
Number of hidden layers	24	32	32
Number of attention heads	16	32	32
Activation function	GELU	GELU	GELU
Vocabulary size	50,277	50,304	50,432
Maximum sequence length	2048	2048	2048
Maximum position embeddings	2048	2048	2048
BOS / EOS token id	0 / 0	0 / 0	0 / 0
Model dtype	float32	bfloat16	bfloat16

Continued on next page

Table 6 continued

Configuration / Hyperparameter	Memdec-1.4B	Memdec-2.8B	Memdec-6.9B
Training hyperparameters			
Training data	The Pile deduplicated	The Pile deduplicated	The Pile deduplicated
Training tokens	300B	300B	300B
Training steps	148,000	148,000	148,000
Global batch size	2M tokens	2M tokens	2M tokens
Sequence length	2048	2048	2048
Optimizer	AdamW	AdamW	AdamW
Learning rate	3×10^{-4}	2.5×10^{-4}	2×10^{-4}
Learning rate schedule	Cosine decay	Cosine decay	Cosine decay
Minimum learning rate	3×10^{-5}	2.5×10^{-5}	2×10^{-5}
Warmup steps	2,000	2,000	2,000
Adam β_1	0.9	0.9	0.9
Adam β_2	0.95	0.95	0.95
Weight decay	0.01	0.01	0.01
Gradient clipping	1.0	1.0	1.0
Memory loss weight	0.5	0.5	0.5
Training precision	bf16	bf16	bf16
Number of GPUs	64	64	256

Table 7: Training hyperparameters for domain memories.

Configuration / Hyperparameter	BioInst	DISC-Law-SFT	FinTrain-unsup
Training hyperparameters			
Memory initialization	Qwen3-1.7B-Base	Qwen3-1.7B-Base	Qwen3-1.7B-Base
Sequence length	4096	4096	4096
Global batch size	0.5M tokens	0.5M tokens	0.25M tokens
Optimizer	AdamW	AdamW	AdamW
Learning rate	3×10^{-4}	3×10^{-4}	3×10^{-4}
Learning rate schedule	Cosine decay	Cosine decay	Cosine decay
Minimum learning rate	3×10^{-5}	3×10^{-5}	3×10^{-5}
Warmup steps	1,000	1,000	200
Adam β_1	0.9	0.9	0.9
Adam β_2	0.95	0.95	0.95
Weight decay	0.01	0.01	0.01
Gradient clipping	1.0	1.0	1.0
Memory loss weight	0.5	0.5	0.5
Training precision	bf16	bf16	bf16

C Evaluation Details

C.1 General Evaluation

We use the LM Evaluation Harness[2] and its accompanying evaluators for the knowledge tasks. The primary results in Table 1 follow the prompt setting defined for each benchmark. The rows receive no dynamically sampled examples. In Section 5.1, all three AVG bars use the same 13 tasks: ARC-Easy, ARC-Challenge, LAMBADA, LogiQA, MMLU, PIQA, SciQ, WinoGrande, NQ-Open, TriviaQA, 2WikiMultiHopQA, Bamboogle, and HotpotQA. Bamboogle retains zero-shot prompting. PopQA, TruthfulQA-MC, HaluEval, and GPQA-main are omitted from all three averages. TruthfulQA retains its canonical prefix of six question and answer pairs as part of the task definition rather than as sampled context.

The reported `acc` tasks are ARC-Easy, ARC-Challenge, LAMBADA-OpenAI, LogiQA, MMLU, PIQA, SciQ, WinoGrande, PopQA, and GPQA-main. NQ-Open, TriviaQA, 2WikiMultiHopQA, Bamboogle, and HotpotQA use exact match. We report TruthfulQA-MC as the mean of MC1 and MC2. For HaluEval, a fixed seed of 42 selects either the hallucinated or reference candidate for each example. The model then predicts `Yes` or `No`, and a response containing both labels or neither is incorrect. The HaluEval score averages accuracy across QA, dialogue, and summarization. GPQA-main averages results over ten independently permuted answer orders.

C.2 Domain Evaluation

BioInst and LawBench use OpenCompass[3], while FinEval uses the LM Evaluation Harness[2]. All methods receive plain base model prompts without tokenizer chat templates and use greedy generation. Prompts, demonstrations, generation budgets, and evaluators remain fixed across methods. By default, RAG uses the top-5 retrieved passages as additional context. Table 8 summarizes the generation budget for each task.

Table 8: Values of `max_new_tokens` for the domain evaluations.

Domain	Tasks	Value
BioInst	All 21 tasks	256
LawBench	`knowledge_question_answering`, `dispute_focus_identification`, `issue_topic_identification`, `argument_mining`, `case_analysis`	256
LawBench	`marital_disputes_identification`, `named_entity_recognition`, `event_detection`, `fact_based_article_prediction`, `prison_term_prediction_wo_article`, and `prison_term_prediction_w_article`	384
LawBench	`document_proofreading`	512
LawBench	`article_recitation`	768
LawBench	`reading_comprehension`, `opinion_summarization`, `trigger_word_extraction`, `scene_based_article_prediction`, `charge_prediction`, `criminal_damages_calculation`, and `consultation`	1,024
FinEval	All tasks except the four listed below	16
FinEval	`Flare-TATQA` and `NER`	128
FinEval	`ECTSUM` and `EDTSUM`	512

[2] EleutherAI/lm-evaluation-harness

[3] open-compass/opencompass

BioInst. We evaluate the 21 `biodata_task_gen` tasks on each `sample_1k` split with a sequence length of 16,384. BioInst combines heterogeneous benchmark metrics. We multiply each task metric by 100 and report the unweighted mean over the 21 tasks. BioInst often expects structured answers specific to each task, which can cause a semantically correct base model prediction to be rejected solely for its format. We therefore apply a deterministic relaxed answer extractor uniformly to all methods before computing each benchmark metric. The references and metric definitions remain unchanged, and no model judge is used.

Extraction starts after the final `</think>` tag when present. It then considers the final complete `\boxed{...}` span, the text following the last explicit answer cue, and the final 500 characters in that order. For binary tasks, the extractor recognizes common label pairs and unambiguous natural language evidence. For five DNA classification tasks, the extractor gives precedence to conservative negative cues when positive and negative evidence coexist. For Protein-Solubility, it uses a dedicated soluble versus insoluble parser. For numeric tasks, the extractor accepts signed values, scientific notation, and percentages, with percentages mapped to $[0, 1]$ when appropriate. For structured tasks, the extractor parses dictionaries or key and value expressions, including aliases for DNA enhancer activity and explicit `OFF`, `ON`, and `ON_OFF` fields for RNA switches. Categorical labels are matched without case or separator sensitivity, and EC numbers are normalized before computing $F_{\max}$. A fixed fallback is used for each task family when no answer can be extracted, with the same fallback applied to every model and example.

LawBench. The primary evaluation covers all 20 tasks without demonstrations, using 500 examples per task. OpenCompass concatenates `instruction`, a newline, and `question` without retrieving additional examples. We report the unweighted mean across tasks. Results with one demonstration are included only as supplementary evidence and are aggregated separately. In this one-shot setting, the demonstration is embedded directly in `instruction`, and the generation budgets remain unchanged. The seven tasks assigned 1,024 tokens retain this longer limit because shorter budgets can truncate the response before a scorable final answer is produced.

FinEval. We evaluate the 25 `fineval_em` tasks on their test splits without demonstrations. Each task uses `generate_until` with greedy decoding and a context length of 32,768. We report the unweighted mean over 20 exact match tasks, two MCC tasks (CRA-CCF and CRA-Polish), and three ROUGE-1 tasks (ECTSUM, EDTSUM, and NER). The deterministic evaluator extracts explicit answer labels when available and otherwise uses the final answer marker or final nonempty line. Nonnumeric answers are normalized for case, punctuation, and whitespace. Numeric answers are compared after removing thousands separators, common currency symbols, and a trailing percent sign, using an absolute tolerance of 10^{-6}. Unrecognized predictions on the two MCC tasks count as errors rather than being discarded. The ROUGE tasks collapse whitespace in the full generated continuation before computing stemmed ROUGE-1 F1.

D ADDITIONAL GENERAL MEMORY RESULTS

Figure 5 reports results for individual tasks under the same total parameter view used in the main analysis.

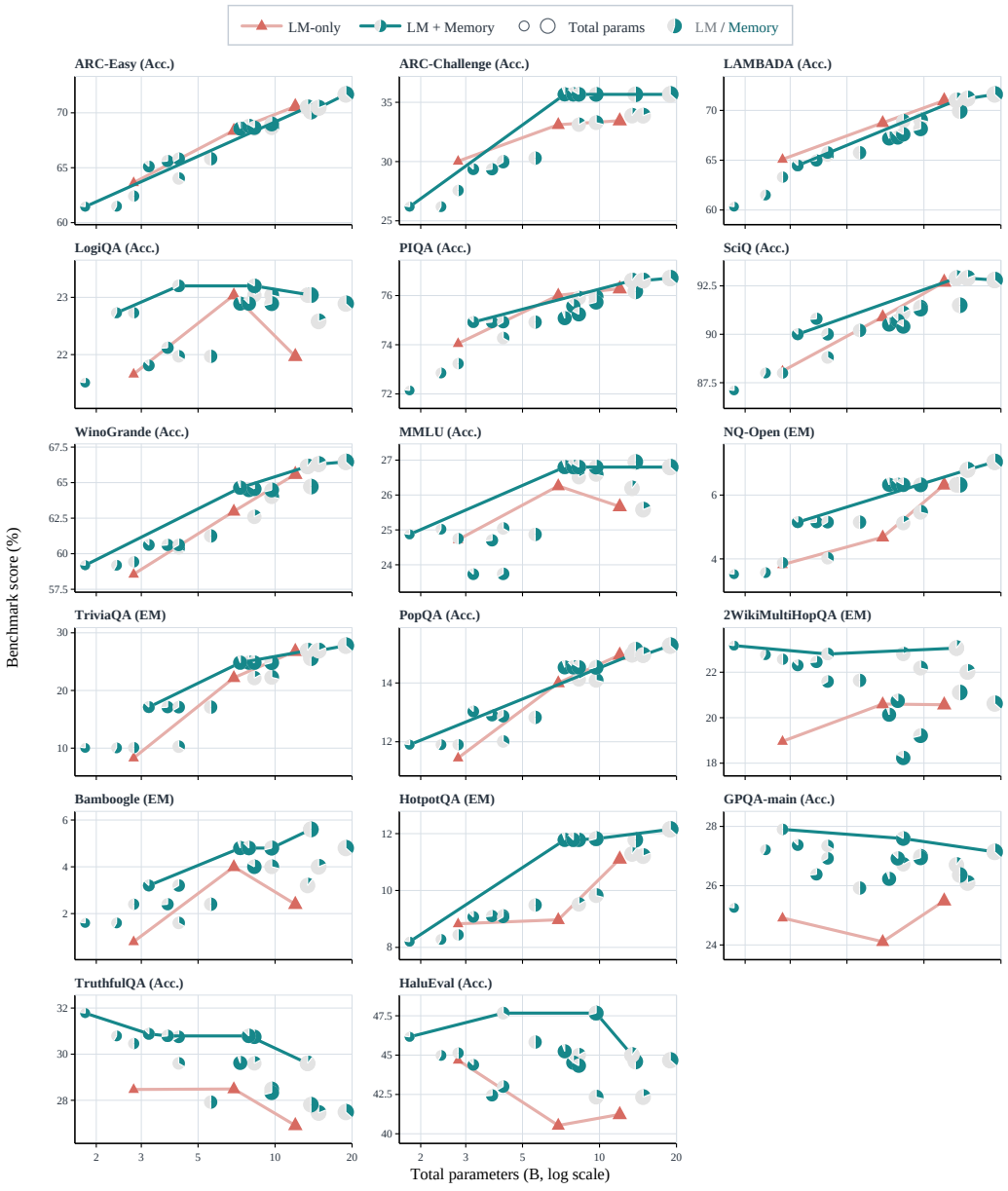

Figure 5: General memory transfer by task under matched parameter counts and training budgets. Marker area denotes total parameters. Triangles denote base models, while pie markers show the Base and Memory parameter shares. Lines trace Base and Base + Memory scaling.

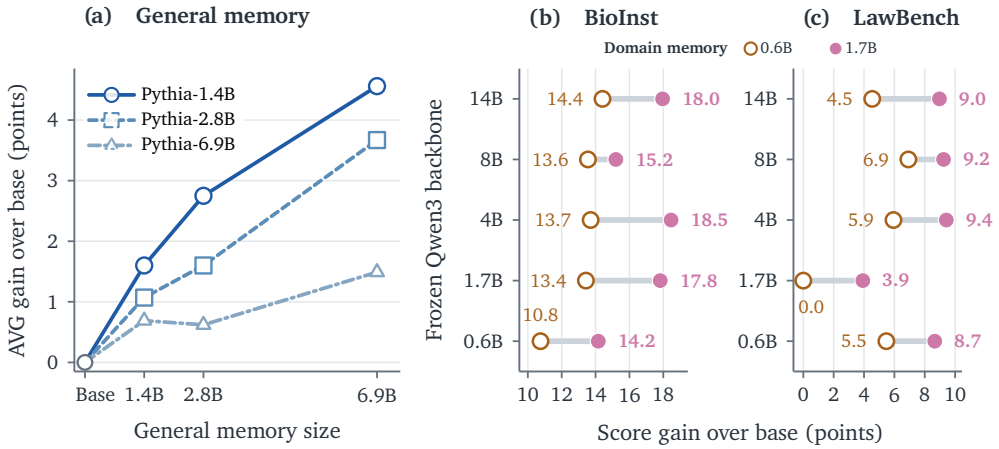

Figure 6: Effect of memory size. **(a)** AVG gains for three general memory sizes on frozen Pythia backbones. **(b,c)** Gains on BioInst and LawBench for frozen Qwen3 backbones. A 0.6B or 1.7B marker denotes the backbone paired with a domain memory of that size.

E Additional Ablation Studies and Analyses

E.1 Memory Size

Figure 6 examines how performance changes as memory capacity increases while keeping the backbone fixed. In the general setting, every tested memory size improves AVG over the corresponding base across all three Pythia backbones. For each fixed backbone, the gain also grows overall with memory capacity. The 6.9B memory therefore yields the largest AVG gain in every case: 4.56, 3.67, and 1.49 points for the 1.4B, 2.8B, and 6.9B backbones, respectively. This consistent trend shows that general memory scales across backbone sizes, with the largest gains on the smaller frozen backbones.

In the domain setting, the 1.7B memory outperforms the 0.6B memory for every Qwen3 backbone on both BioInst and LawBench. Across backbones, adding the 1.7B memory increases scores by as much as 18.5 points on BioInst and 9.4 points on LawBench. Increasing memory capacity from 0.6B to 1.7B provides up to an additional 4.8 points on BioInst and 4.4 points on LawBench. Taken together, the general and domain results show a consistent scaling trend, with larger memory capacities generally improving performance across backbone sizes and domains.

E.2 Memory Training Budget

Table 9 examines how the amount of memory training affects performance. For general memory, extending training from 1.5 to 5 epochs raises the average across general tasks from 56.67 to 57.33 and the knowledge average from 13.99 to 14.17. The domain memory experiments follow the setting in Table 3 and compare 20% of the standard training budget with the full budget. Using the full budget improves Avg for every combination of OLMo backbone and memory size. The gains are 0.34 and 3.51 points for the 0.6B and 1.7B memories on OLMo-2, and 0.94 and 1.75 points on OLMo-3. Across both settings, increasing the memory training budget consistently improves aggregate performance, showing that memory continues to benefit from additional training.

E.3 Extractable Memorization Evaluation

We next ask whether the memory makes training content more extractable, beyond its effect on downstream accuracy. We conduct this analysis in the domain memory setting. Following the discoverable extraction protocol that conditions on a prefix to recover its suffix (Carlini et al., 2022; Hayes et al., 2025), we evaluate the 1.7B BioInst memory model against a Qwen3-1.7B-Base CPT model on the same 1,024 training prompts. Both models are trained on the same BioInst training data under the same budget and differ only in the training mechanism. *Verbatim Continuation* pro-

Table 9: Effect of training budget for general and domain memory. The domain evaluation setting follows Table 3. Bold marks the better result within each comparison.

General memory: Pythia-1.4B + Mem-1.4B			
Memory checkpoint	**Epochs**	**General Tasks**	**Knowledge**
Default	1.5	56.67	13.99
Longer training	5	**57.33**	**14.17**

Domain memory: OLMo transfer						
Backbone	**Memory**	**Budget**	**BioInst**	**LawBench**	**FinEval**	**Avg**
OLMo-2-7B	0.6B	20%	9.86	16.78	**50.15**	25.60
	0.6B	Full	**11.15**	**17.76**	48.92	**25.94**
	1.7B	20%	7.98	14.89	48.61	23.83
	1.7B	Full	**11.72**	**17.15**	**53.14**	**27.34**
OLMo-3-7B	0.6B	20%	9.28	22.34	**46.66**	26.09
	0.6B	Full	**10.79**	**24.02**	46.29	**27.03**
	1.7B	20%	8.16	22.40	48.76	26.44
	1.7B	Full	**11.64**	**23.74**	**49.20**	**28.19**

Table 10: Extractable memorization on BioInst training examples. N denotes the number of evaluated training examples for each probe.

Probe	N	CPT model	memory model
Verbatim continuation, EM@8, 16	1,024	42.4%	**49.7%**
Domain anchor completion	1,024	22.6%	**56.5%**

Verbatim continuation (FunctionEC)
Prefix `... Output: EC3.4.24.-`
Gold `,EC3.4.24.69 identifies the enzyme's function`
× CPT model `,EC3.4.24.69.`
✓ memory model `,EC3.4.24.69 identifies the enzyme's function`

Domain anchor completion (EMP)
Prefix `... Output: EMP, or Epigenetic Marks Prediction, aims to identify`
Gold `epigenetic changes, successfully confirmed`
× CPT model `acetylation and methylation nucleosome occupancies, ...`
✓ memory model `epigenetic changes, successfully confirmed ...`

Figure 7: Model outputs for the two extractable memorization probes. Each panel shows the provided prefix, gold target, and continuations from the CPT and memory models. Ellipses abbreviate the task context and text beyond the scored span.

vides the task context and the first eight output tokens from an enzyme function annotation task (`FunctionEC`). EM@8, 16 is the percentage of examples for which greedy decoding exactly reproduces the next 16 target tokens. *Domain Anchor Verbatim Completion* instead hides a four-word span from an epigenetic mark prediction task (`EMP`) and tests whether generation starts with that exact span.

Table 10 shows that the memory model raises strict EM@8, 16 from 42.4% to 49.7%. On domain anchor completion, the gap widens from 22.6% to 56.5%. Figure 7 compares the outputs of the memory and CPT models under the two probes. Together, these results show that the memory provides stronger traceability to its training data, suggesting that it retains content from the training domain in a form that can be more reliably recovered from partial context. Appendix I gives the construction, analysis by suffix frequency, paired counts, and additional examples.

E.4 Memory Fidelity to Retrieval Supervision

We evaluate how closely a parametric memory matches its retrieval-based training targets. We use the 1.7B BioInst domain setting and draw 1,024 samples from the training datastore, each containing a context and its next token. For sample i, P_i is the stored kNN target distribution and Q_i is the memory distribution without datastore access. Their agreement measures how well the memory internalizes retrieval supervision. We use four complementary metrics. Top token match compares modes, while KL divergence and total variation (TV) compare full distributions. Smaller values indicate closer agreement. KL is sensitive when Q_i underweights tokens with high probability under P_i. TV is half the ℓ_1 distance and equals the probability mass that must be reassigned for exact agreement. Pearson r measures whether the memory probability for the token selected by kNN tracks the corresponding target probability across samples.

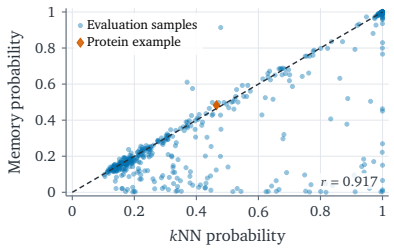

Figure 8: kNN and memory probabilities for the mode of the kNN distribution. The dashed line marks equality.

Table 11: Agreement between kNN targets and memory distributions over 1,024 BioInst samples. Top token match compares modes, KL and TV compare full distributions, and Pearson r compares probabilities for the kNN mode.

Metric	All samples
Top token match	86.62%
Mean KL	0.1820
Mean TV	0.0823
Pearson r	0.9174

Figure 8 shows strong correspondence between kNN and memory probabilities for the mode of the kNN distribution. Most samples lie near equality, indicating that memory tracks the target probability rather than merely selecting the same token. Table 11 confirms this agreement across 1,024 samples. The modes match in 86.62% of cases, with Pearson $r = 0.9174$, mean KL 0.1820, and mean TV 0.0823. Together, these metrics show alignment in both token choice and probability allocation. Additional results are provided in Appendix J. In the protein example in Figure 9, both distributions select *task*, with KL 0.0013 and TV 0.0198. Overall, parametric memory learns the structure of the retrieval distribution rather than only its argmax. Residual errors arise mainly for diffuse targets, suggesting that matching uncertainty is harder than identifying the mode. The learned signal can be interpolated with the frozen base distribution at inference without an online datastore lookup.

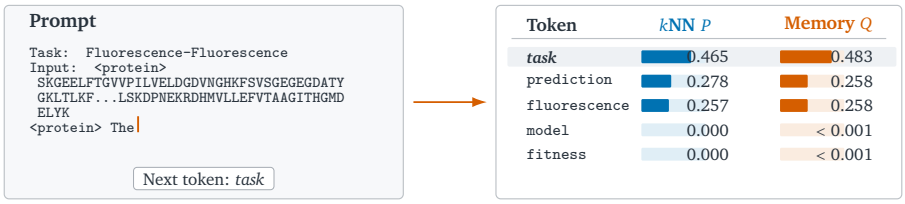

Figure 9: Protein example. The sequence is abbreviated. The target continuation is highlighted. The right panel compares kNN and memory probabilities for five candidate tokens.

F Additional Domain Memory Results

F.1 Results with 0.6B Domain Memory

Table 12 reports the 0.6B domain memory under the zero-shot domain setup used in the main results. The average score improves for every Qwen3 backbone and both OLMo backbones. This result shows that the gains are not limited to the larger 1.7B memory. The OLMo rows use 20% of the memory training budget in Table 2.

Table 12: Results for the 0.6B domain memory under the zero-shot setting. OLMo memory rows use 20% of the training budget in Table 2. Metrics follow Tables 2 and 3.

Model	BioInst	LawBench	FinEval	Avg
Qwen3 Family				
Qwen3-0.6B-Base	5.78	17.87	23.31	15.65
+*CPT*	7.94 +2.16	20.39 +2.52	14.90 -8.41	14.41(-1.24)
+*LoRA*	10.36 +4.58	21.10 +3.23	19.54 -3.77	17.00(+1.35)
+*RAG*	2.96 -2.82	17.82 -0.05	21.37 -1.94	14.05(-1.60)
+*Mem-0.6B*	**16.53** +10.75	**23.34** +5.47	**29.42** +6.11	**23.10**(+7.44)
Qwen3-1.7B-Base	5.39	26.86	27.26	19.84
+*CPT*	4.84 -0.55	28.86 +2.00	29.02 +1.76	20.91(+1.07)
+*LoRA*	9.01 +3.62	28.53 +1.67	**38.74** +11.48	25.43(+5.59)
+*RAG*	8.07 +2.68	**29.61** +2.75	36.63 +9.37	24.77(+4.93)
+*Mem-0.6B*	**18.82** +13.43	26.86 +0.00	34.54 +7.28	**26.74**(+6.90)
Qwen3-4B-Base	5.02	27.47	37.43	23.31
+*CPT*	9.14 +4.12	**36.10** +8.63	30.22 -7.21	25.15(+1.85)
+*LoRA*	8.06 +3.04	33.65 +6.18	33.52 -3.91	25.08(+1.77)
+*RAG*	8.34 +3.32	32.62 +5.15	39.13 +1.70	26.70(+3.39)
+*Mem-0.6B*	**18.73** +13.71	33.41 +5.94	**42.06** +4.63	**31.40**(+8.09)
Qwen3-8B-Base	4.82	32.67	43.51	27.00
+*CPT*	9.79 +4.97	39.24 +6.57	40.00 -3.51	29.68(+2.68)
+*LoRA*	5.32 +0.50	33.55 +0.88	42.43 -1.08	27.10(+0.10)
+*RAG*	7.78 +2.96	**40.67** +8.00	46.97 +3.46	31.81(+4.81)
+*Mem-0.6B*	**18.37** +13.55	39.59 +6.92	**47.36** +3.85	**35.11**(+8.11)
Qwen3-14B-Base	4.01	35.45	44.25	27.90
+*RAG*	7.70 +3.69	**44.38** +8.93	42.81 -1.44	31.63(+3.73)
+*Mem-0.6B*	**18.42** +14.41	39.99 +4.54	**47.71** +3.46	**35.37**(+7.47)
OLMo Family (cross-vocabulary transfer, 20% training budget)				
OLMo-2-7B	3.90	6.37	48.43	19.57
+*CPT*	9.51 +5.61	**22.95** +16.58	34.61 -13.82	22.36(+2.79)
+*LoRA*	9.44 +5.54	13.21 +6.84	32.34 -16.09	18.33(-1.24)
+*RAG*	3.70 -0.20	12.11 +5.74	40.88 -7.55	18.90(-0.67)
+*Mem-0.6B*	**9.86** +5.96	16.78 +10.41	**50.15** +1.72	**25.60**(+6.03)
OLMo-3-7B	6.39	13.19	36.43	18.67
+*CPT*	**13.69** +7.30	20.29 +7.10	38.89 +2.46	24.29(+5.62)
+*LoRA*	6.95 +0.56	16.40 +3.21	28.21 -8.22	17.19(-1.48)
+*RAG*	5.95 -0.44	21.85 +8.66	33.10 -3.33	20.30(+1.63)
+*Mem-0.6B*	9.28 +2.89	**22.34** +9.15	**46.66** +10.23	**26.09**(+7.42)

F.2 LawBench One-Shot Results

Table 13 reports one-shot LawBench results, with one demonstration included in each prompt. On Qwen3, +Mem-1.7B achieves the highest score for all five backbones and outperforms RAG. On OLMo, CPT achieves the highest score for both backbones, while both memory sizes still improve over the frozen base using 20% of the memory training budget.

Table 13: One-shot LawBench results using the generation budgets of the primary evaluation. OLMo memory columns use 20% of the training budget in Table 2. Bold marks the best score for each backbone, and underlining marks the second best.

Model	Base	CPT	LoRA	RAG	+Mem-0.6B	+Mem-1.7B
Qwen3 Family						
Qwen3-0.6B-Base	17.19	21.09	22.37	18.79	<u>24.87</u>	**28.48**
Qwen3-1.7B-Base	29.45	29.52	25.97	<u>33.25</u>	32.20	**34.58**
Qwen3-4B-Base	37.22	34.23	33.81	<u>40.77</u>	39.56	**41.64**
Qwen3-8B-Base	42.50	42.71	37.05	43.92	<u>44.13</u>	**45.70**
Qwen3-14B-Base	43.68	–	–	46.18	<u>46.46</u>	**48.68**
OLMo Family (cross-vocabulary transfer, 20% training budget)						
OLMo-2-7B	12.56	**23.46**	14.63	15.29	<u>18.64</u>	17.99
OLMo-3-7B	16.37	**27.21**	19.32	<u>25.92</u>	22.33	24.07

F.3 Effect of RAG Retrieval Depth

Table 14 compares top-3 and top-5 retrieval for RAG under the same evaluation settings as Tables 2 and 3. Top-5 retrieval performs better in 11 of the 21 backbone and domain comparisons, while top-3 performs better in the remaining 10. Neither retrieval depth consistently dominates across backbones or domains.

Table 14: Top-3 versus top-5 retrieval for the RAG baseline. Bold marks the better retrieval depth for each backbone and metric.

Model	RAG	BioInst	LawBench	FinEval	Avg
Qwen3-0.6B-Base	top-3	**3.89**	**18.36**	**24.86**	**15.70**
	top-5	2.96	17.82	21.37	14.05
Qwen3-1.7B-Base	top-3	7.33	**29.81**	36.60	24.58
	top-5	**8.07**	29.61	**36.63**	**24.77**
Qwen3-4B-Base	top-3	7.60	**33.53**	**40.16**	**27.10**
	top-5	**8.34**	32.62	39.13	26.70
Qwen3-8B-Base	top-3	7.18	40.46	45.78	31.14
	top-5	**7.78**	**40.67**	**46.97**	**31.81**
Qwen3-14B-Base	top-3	7.35	43.97	**42.86**	31.39
	top-5	**7.70**	**44.38**	42.81	**31.63**
OLMo-2-7B	top-3	**4.16**	11.88	**45.91**	**20.65**
	top-5	3.70	**12.11**	40.88	18.90
OLMo-3-7B	top-3	5.45	**22.52**	32.74	20.24
	top-5	**5.95**	21.85	**33.10**	**20.30**

G SENSITIVITY TO THE INTERPOLATION COEFFICIENT

At inference, we interpolate the output distributions of the frozen backbone and memory as $p_{\text{final}} = (1-\alpha)p_{\text{base}} + \alpha p_{\text{mem}}$. When validation data is available, the interpolation coefficient α is tuned on the validation split and then fixed for test evaluation. Table 15 summarizes the interpolation coefficients for the results in Table 1 that use a single task-level coefficient. We additionally evaluate $\alpha \in \{0.00, 0.05, \ldots, 1.00\}$ on six representative tasks spanning general and knowledge-intensive benchmarks.

Table 15: Interpolation coefficients for the results in Table 1 that use a single task-level coefficient. Column headers indicate the parameter scale of both the backbone and memory.

Benchmark	**1.4B**	**2.8B**	**6.9B**
	General Tasks		
ARC-Easy	0.30	0.60	0.45
ARC-Challenge	0.25	0.75	1.00
LAMBADA	0.50	0.60	0.45
LogiQA	0.15	0.00	0.00
PIQA	0.55	1.00	0.60
SciQ	0.35	0.45	0.75
WinoGrande	0.70	0.55	0.85
	Knowledge		
MMLU	0.95	0.10	0.60
NQ-Open	0.80	1.00	1.00
TriviaQA	0.90	1.00	0.80
PopQA	1.00	0.85	0.55
2WikiMultiHopQA	0.95	0.90	0.30
Bamboogle	0.40	0.60	0.55
HotpotQA	0.80	0.30	1.00
	Hallucination		
TruthfulQA	0.15	0.15	0.20

The coefficient distributions in Table 15 provide a useful view of how the memory contribution changes with evaluation requirements. For general tasks, the coefficients concentrate in a moderate range, with a median of 0.55 and an interquartile range of 0.35–0.70, reflecting substantial contributions from both distributions. Knowledge-intensive tasks are more heavily represented at larger coefficients, with a median of 0.80 and an interquartile range of 0.55–0.95, consistent with stronger use of the memory distribution when stored knowledge is central. The overlap between the two ranges indicates a gradual change in emphasis rather than a sharp separation.

Figure 10 reports performance gains relative to the $\alpha = 0$ endpoint from the same sweep. This within-sweep reference isolates the effect of interpolation from small differences between separately executed base evaluations. Across the six tasks, 17 of the 18 task–scale curves stay above $\alpha = 0$ for all 20 nonzero coefficients. WinoGrande, NQ-Open, TriviaQA, PopQA, and HotpotQA exhibit this behavior at all three scales, showing that their improvements do not rely on a single isolated coefficient. For ARC-Easy, the 1.4B curve remains positive over the central portion of the sweep, while both larger configurations remain positive throughout. Taken together, the curves show that the improvements persist over broad coefficient intervals rather than arising from sharply localized peaks.

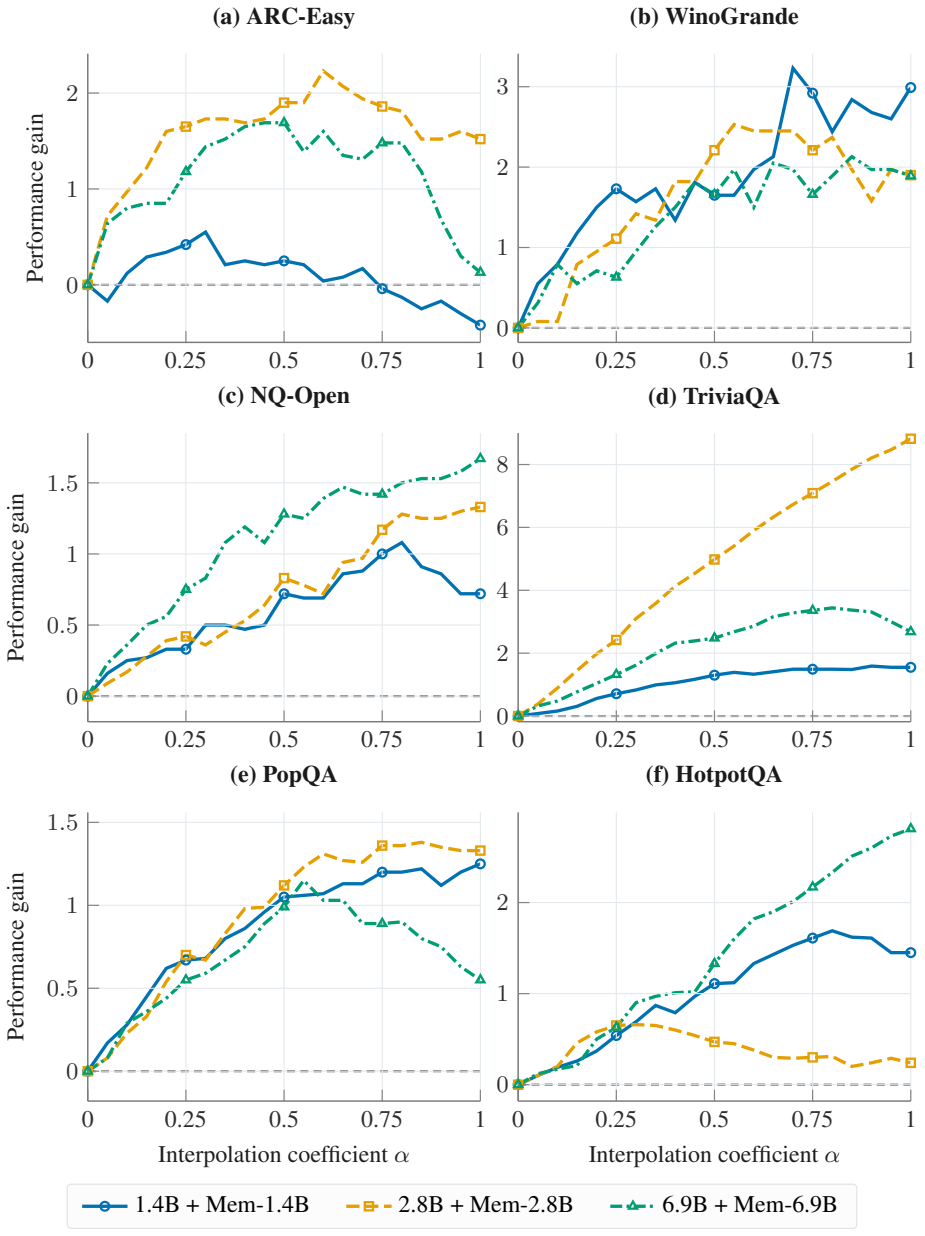

Figure 10: Sensitivity to the interpolation coefficient across six representative tasks. Each panel reports the performance gain relative to the within-sweep $\alpha = 0$ endpoint; positive values indicate an improvement over the no-memory setting. Lines include all 21 coefficients at intervals of 0.05, with markers shown every 0.25 for readability. Panels use task-specific vertical scales.

H CASE STUDIES

H.1 MEMORY STRENGTHENS FACTUAL EVIDENCE

Figure 11 compares gold label probabilities at the word level within a supporting HotpotQA paragraph titled *Brooklyn Nine-Nine* (Yang et al., 2018). The case study uses frozen Pythia-1.4B as the base model and the 1.4B memory in Table 1. For each displayed word w_i, let $p_{\text{base}}(w_i)$ and $p_{\text{mem}}(w_i)$ denote the geometric mean gold label probabilities of its constituent subword tokens. We visualize the normalized memory share

$$s_i = \frac{p_{\text{mem}}(w_i)}{p_{\text{base}}(w_i) + p_{\text{mem}}(w_i)}. \tag{7}$$

A value of $s_i = 0.5$ indicates equal support, while $s_i > 0.5$ indicates greater support from memory. The memory model assigns larger shares to many entity and attribute words. This pattern is consistent with the aggregate QA gains and suggests that memory reinforces factual evidence.

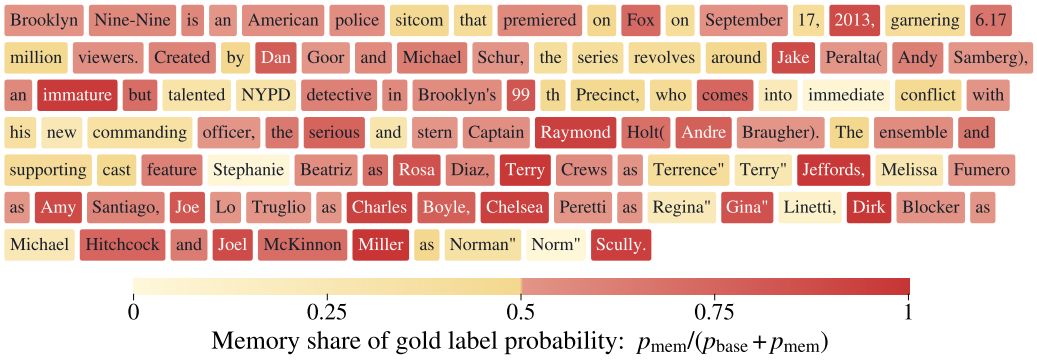

Figure 11: Memory probability share on the HotpotQA context titled *Brooklyn Nine-Nine*. Colors encode $p_{\text{mem}}/(p_{\text{base}} + p_{\text{mem}})$. Values below 0.5 favor the frozen base model, while values above 0.5 favor the memory model.

H.2 Additional Examples

We provide additional case study examples in Figures 12–15.

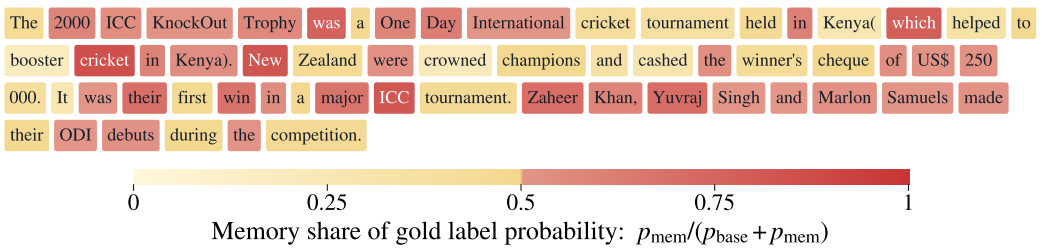

Figure 12: Additional case study on a factual question about the *2000 ICC KnockOut Trophy*.

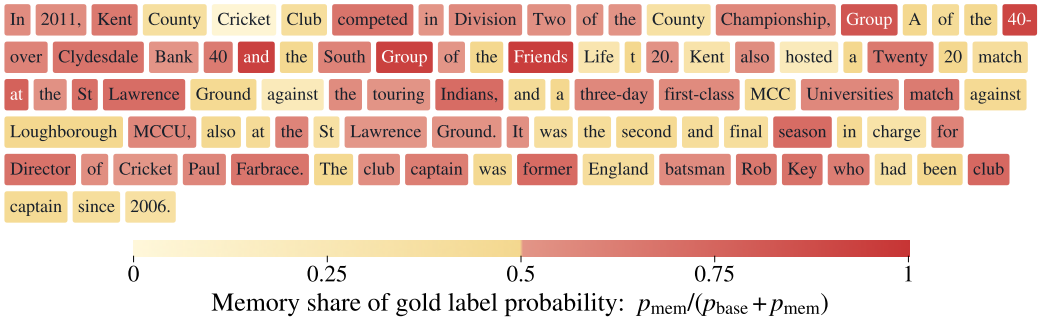

Figure 13: Additional case study on a factual question about *Kent County Cricket Club in 2011*.

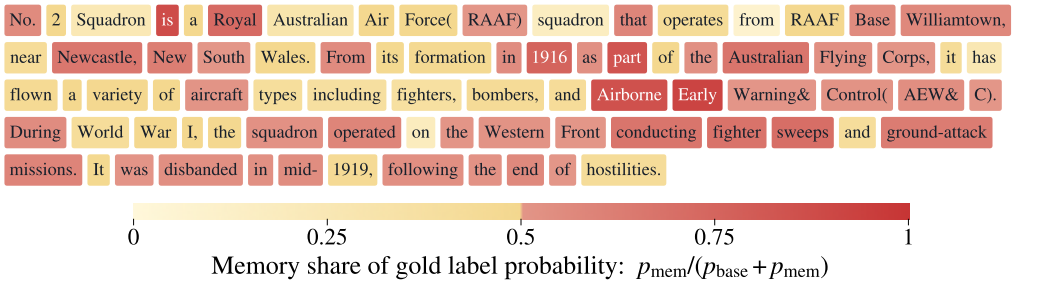

Figure 14: Additional case study on a factual question about *No. 2 Squadron RAAF*.

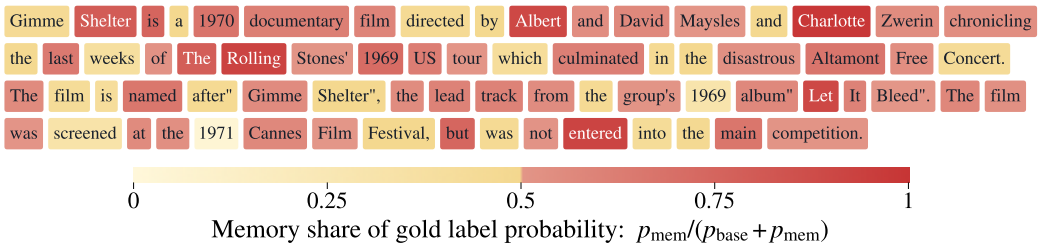

Figure 15: Additional case study on a factual question about the documentary film *Gimme Shelter*.

I EXTRACTABLE MEMORIZATION EVALUATION DETAILS

This appendix specifies the two BioInst traceability probes summarized in Section E.3. Both use examples from the raw BioInst training corpus and compare the 1.7B memory model with a Qwen3-1.7B-Base CPT model trained on the same corpus. Thus, the comparison isolates the memory training mechanism from mere exposure to BioInst data.

I.1 METRIC AND DECODING PROTOCOL

For a training output token sequence, let p_i^K be an output prefix of K tokens and s_i^L the immediately following suffix of L tokens. The model input combines the original task context with p_i^K. The task context is not counted in K. With deterministic greedy decoding, we define

$$\mathrm{EM@}K, L = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\left[\mathrm{Greedy}_L(M, c_i, p_i^K) = s_i^L\right].$$

Equality is evaluated on token IDs, not normalized strings. We use one deterministic query per example and do not estimate probabilistic extraction under repeated stochastic queries (Hayes et al., 2025). We also report k-eidetic EM, restricting the same rate to suffixes occurring in at most k distinct BioInst training rows (Carlini et al., 2021). For paired comparisons, the CPT only and memory only columns count prompts solved by exactly one model.

I.2 VERBATIM CONTINUATION

We scan 15,537 `FunctionEC-FunctionEC` rows. Of these, 10,178 contain an output prefix of eight tokens followed by a suffix of 16 tokens. Requiring each selected prefix to have a unique observed continuation leaves 1,112 candidates, from which we sample $N = 1{,}024$ with seed 41. Each model receives the full protein and task context together with the eight output tokens, then greedily generates 16 tokens. The full prompts have a median length of 208.5 tokens, a mean of 225.4, and a range from 69 to 582.

Table 16: Full results for FunctionEC verbatim continuation.

Metric	N	CPT model	memory model	CPT only	memory only	Both	Neither
EM@8, 16	1,024	434 (42.4%)	509 (49.7%)	56	131	378	459

Table 17 stratifies exact recovery by suffix row frequency. The direction favors the memory model for all three thresholds. However, these subsets are small, so we treat them as supporting rather than standalone evidence.

Table 17: FunctionEC continuation for suffixes with limited frequency. A suffix is included when it occurs in at most k BioInst training rows.

Subset	N	CPT model	memory model	CPT only	memory only
$k \leq 1$	30	6 (20.0%)	11 (36.7%)	3	8
$k \leq 5$	76	13 (17.1%)	21 (27.6%)	7	15
$k \leq 10$	101	16 (15.8%)	24 (23.8%)	9	17

The FunctionEC example in Figure 7 corresponds to training sample 52,021 (example `bioinst_em_000616`). Its gold suffix occurs in one training row. The memory model reproduces all 16 tokens, whereas the CPT model predicts the EC number but terminates with a shorter continuation.

I.3 Domain Anchor Verbatim Completion

We scan 1,215,490 BioInst rows and retain the EMP tasks `emp-H3`, `emp-H4`, `emp-H3K4me1`, and `emp-H3K9ac`. Candidate outputs contain a span of four words beginning with the domain anchors `DNA`, `epigenetic`, or `marks`. The scan yields 10,321 candidates. After removing prompt leakage and capping overly frequent spans, 6,676 remain. We sample 1,024 examples with seed 41 and score whether greedy generation begins with the exact hidden span. The full prompts have median length 295 tokens, a mean of 295.6, and a range from 268 to 322. The output prefix has a median of six words, and every target contains four words.

Table 18: Full EMP domain anchor completion results.

Metric	CPT model	memory model	CPT only	memory only
Starts with target span	231 (22.6%)	579 (56.5%)	41	389
Target appears anywhere	369 (36.0%)	660 (64.5%)	72	363
Generation occurs in source	242 (23.6%)	587 (57.3%)	43	388

The advantage persists after restricting by target span frequency in the candidate pool. For spans observed at most 50 times, the memory model recovers 194 of 312 targets (62.2%), compared with 98 (31.4%) for the CPT model. The two models exclusively recover 111 and 15 prompts, respectively. The EMP example in Figure 7 corresponds to training sample 1,016,559 from `emp-H3`. The target span appears 35 times in the extracted pool. The memory model restores "epigenetic changes, successfully confirmed," whereas the CPT model begins with the different continuation "acetylation and methylation nucleosome occupancies."

I.4 Additional Qualitative Examples

Together with Figure 7, the eight examples below complete two qualitative sets of five examples. Each prefix omits the shared task context, and ellipses abbreviate text beyond the scored span.

Domain anchor completion (EMP 1)
Prefix `... Output: ... our analysis of yeast`
Gold `DNA did not achieve`
× CPT model `did not achieve this.`
✓ memory model `DNA did not achieve this.`

Domain anchor completion (EMP 2)
Prefix `... Output: ... epigenetic markers, and our yeast`
Gold `DNA analysis supports these`
× CPT model `300 bp DNA sequence supports these predictions.`
✓ memory model `DNA analysis supports these predictions.`

Domain anchor completion (EMP 3)
Prefix `... Output: The purpose of the EMP task is to predict`
Gold `epigenetic markers, and our`
× CPT model `epigenetic modifications, and our ...`
✓ memory model `epigenetic markers, and our ...`

Domain anchor completion (EMP 4)
Prefix `... Output: ... could not confirm the presence of epigenetic`
Gold `marks in the DNA.`
× CPT model `36me3 marks in the DNA.`
✓ memory model `marks in the DNA.`

Figure 16: Four additional domain anchor completions solved only by the memory model.

Verbatim continuation (FunctionEC 1)
Prefix `... Output: 5.3.1.9,`
Gold `EC5.3.1.- corresponds to the enzyme's function in this sequence`
× CPT model `EC5.3.1.-.`
✓ memory model `EC5.3.1.- corresponds to the enzyme's function in this sequence`

Verbatim continuation (FunctionEC 2)
Prefix `... Output: EC2.7.7.2`
Gold `4,EC2.7.7.- identifies the enzyme's function within it`
× CPT model `4,EC2.7.7.- is the enzyme's function.`
✓ memory model `4,EC2.7.7.- identifies the enzyme's function within it`

Verbatim continuation (FunctionEC 3)
Prefix `... Output: EC3.3.1.1`
Gold `,EC3.3.1.- identifies the enzyme's function within it.`
× CPT model `,EC3.3.1.- corresponds to the enzyme's function in this ...`
✓ memory model `,EC3.3.1.- identifies the enzyme's function within it.`

Verbatim continuation (FunctionEC 4)
Prefix `... Output: number EC4.3.3.`
Gold `7,EC4.3.3.- identifies the enzyme's function within it`
× CPT model `7,EC4.3.3.-4 indicates the presence of protein modifications`
✓ memory model `7,EC4.3.3.- identifies the enzyme's function within it`

Figure 17: Four additional FunctionEC continuations solved only by the memory model. All target suffixes occur in one BioInst training row.

J Additional Details on Memory Fidelity to Retrieval Supervision

This appendix expands the direct comparison between the retrieval target and the trained 1.7B BioInst domain memory in Section E.4. The aggregate metrics and examples show how the memory learns its kNN supervision and contributes this knowledge to the final prediction through interpolation with a frozen base model.

J.1 Sampling Protocol

We uniformly draw 1,024 samples without replacement from the complete BioInst training datastore. Each sample consists of a context and its corresponding next token. For each sample, we compare the stored kNN target P_i with the distribution Q_i produced by the same trained 1.7B BioInst domain memory used in the main results.

J.2 Metrics

Let $v_i^\star = \arg\max_v P_i(v)$. Top token match is the sample mean of $\mathbb{1}[v_i^\star = \arg\max_v Q_i(v)]$. The two distribution distances are

$$D_{\mathrm{KL}}(P_i \| Q_i) = \sum_{v \in \mathcal{V}} P_i(v) \log \frac{P_i(v)}{Q_i(v)}, \qquad \mathrm{TV}(P_i, Q_i) = \frac{1}{2} \sum_{v \in \mathcal{V}} |P_i(v) - Q_i(v)|.$$

KL emphasizes cases where the memory assigns too little mass to a likely kNN token. Total variation is bounded between zero and one and equals the amount of probability mass that must be reassigned to make the distributions equal. Finally, Pearson correlation is computed across the 1,024 paired values $(P_i(v_i^\star), Q_i(v_i^\star))$. Unlike top token match, this statistic tests whether confidence on the same target token varies consistently across samples.

J.3 Target Distribution Structure and Memory Fidelity

Table 19: Distribution reproduction for targets supported by one or multiple tokens. All rows use the same random set of samples. KL is measured in nats. Pearson is reported only for all samples.

Target type	N	Share	Top token match	Mean KL	Mean TV	Pearson r
All samples	1,024	100.00%	86.62%	0.1820	0.0823	0.9174
Single token	599	58.50%	99.50%	0.0206	0.0050	-
Multiple tokens	425	41.50%	68.47%	0.4095	0.1913	-

To examine how fidelity varies with target distribution structure, Table 19 separates targets supported by one token from those that distribute probability across several tokens. Single token targets reach 99.50% top token agreement and mean TV 0.0050.

Targets with multiple support tokens are harder, with 68.47% top token agreement, mean KL 0.4095, and mean TV 0.1913. Total variation exceeds 0.5 for 63 of the 1,024 samples, corresponding to 6.15%. Thus, the aggregate metrics combine near exact reproduction of many concentrated targets with a smaller set of more ambiguous samples. The strong overall agreement shows that the memory learns both concentrated targets and targets that divide probability among several plausible tokens, although the latter remain more difficult.

To assess the stability of the aggregate estimates, Table 20 reports 95% bootstrap confidence intervals based on 10,000 resamples. The narrow intervals across all four metrics show that the estimates remain stable under resampling.

Table 20: Bootstrap confidence intervals over evaluation samples, based on 10,000 resamples.

Metric	Estimate	95% interval
Top token match	86.62%	[84.47%, 88.67%]
Mean KL	0.1820	[0.1395, 0.2273]
Mean TV	0.0823	[0.0708, 0.0943]
Pearson r	0.9174	[0.8954, 0.9383]

J.4 How Memory Contributes to Model Predictions

The distribution comparison shows how the BioInst memory contributes to the main results. The kNN targets encode which continuations are supported by the BioInst datastore and how probability is divided among plausible tokens. The high top token agreement, strong probability correlation, and small distances show that the trained 1.7B memory has absorbed this knowledge into its parameters. The examples provide the same evidence for individual samples, where the memory recovers both the preferred token and the relative weights of its main alternatives.

At inference time, the memory processes the same context as the frozen base model. Their distributions are linearly interpolated for each next token prediction, so the retrieval signal learned from BioInst contributes directly to the final prediction without an online datastore lookup. This mechanism connects the training objective to the strong gains in the main BioInst results. The memory signal is mixed directly with the base distribution, allowing continuations favored by BioInst retrieval to influence each token prediction while the base model remains frozen. The analysis therefore connects the retrieval signal learned by the memory to each next token prediction.

J.5 Additional Examples

The following cases use the same format as Figure 9. They complement the protein example with DNA and RNA samples from the fixed random evaluation set. These figures are qualitative illustrations. All quantitative conclusions use the complete random sample.

In the DNA example, *enh* is the target continuation and the mode of both distributions. Their probability allocations remain close across the displayed candidates, with KL 0.0020 and total variation 0.0249.

In the RNA example, *does* is the target continuation and is ranked second by both distributions, while *RNA* is their shared mode. The close probabilities, KL 0.0019, and total variation 0.0235 show that the trained memory reproduces the kNN target. This distinction is why the example is included here rather than used as the main illustration.

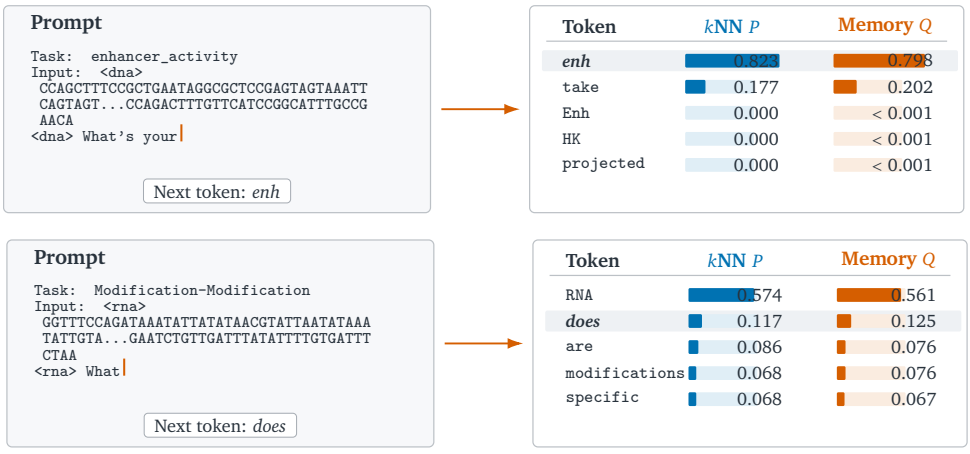

Figure 18: Additional examples in the same format as the main example. The DNA continuation *enh* is the shared mode. The RNA continuation *does* is ranked second behind *RNA*.

K PER-TASK DOMAIN RESULTS

Tables 21 through 41 expand the domain results from domain averages to individual tasks. All evaluations use zero-shot prompting, and RAG retrieves the five highest-ranked passages. We group the tables by domain, with one compact table for each base backbone. Each table compares Base, CPT, LoRA, RAG, and the two memory sizes. In the six OLMo tables, both memory columns use 20% of the memory training budget in Table 2.

K.1 BIOINST

Table 21: Per-task domain memory results on BioInst for the Qwen3-0.6B-Base backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
DNA cpd	MCC	2.96	21.27	32.83	-0.17	**34.95**	31.68
DNA emp	MCC	1.40	7.08	-4.28	-3.22	**10.19**	7.06
DNA pd	MCC	-2.03	10.79	18.17	-3.12	29.96	**40.24**
DNA tf h	MCC	0.54	6.71	-1.07	-5.92	21.06	**21.45**
DNA tf m	MCC	1.15	2.75	8.51	-4.67	**23.89**	20.55
Antibody antigen	MCC	1.21	2.47	-0.37	-2.83	**4.53**	**4.53**
Promoter enhancer	MCC	1.22	0.04	0.00	-4.00	**4.54**	3.95
RNA protein interaction	MCC	0.46	-2.99	0.00	3.16	4.23	**12.73**
DNA enhancer activity	PCC	-1.19	1.33	15.43	0.21	2.26	**21.65**
CRISPR on target	Spearman	-2.32	-1.51	0.00	6.38	2.18	**8.24**
Protein fluorescence	Spearman	-0.26	0.00	0.00	-0.43	**18.51**	15.37
Protein stability	Spearman	3.74	0.68	6.66	-1.62	8.71	**15.00**
Protein thermostability	Spearman	3.70	5.70	9.96	-2.36	**21.11**	6.68
RNA isoform	R2	0.06	12.30	0.00	0.04	**32.79**	29.31
Mean ribosome loading	R2	0.00	0.58	0.00	0.00	4.48	**42.17**
Programmable RNA switches	R2	0.00	2.19	**18.68**	0.18	0.78	15.72
RNA modification	AUC	50.59	49.95	50.04	50.08	50.39	**51.06**
Protein solubility	Acc.	19.90	8.90	28.20	0.10	24.80	**29.50**
Noncoding RNA family	Acc.	5.10	0.90	0.00	5.50	**8.40**	4.80
Protein function EC	Score	**2.04**	**2.04**	**2.04**	**2.04**	**2.04**	**2.04**
siRNA efficiency	Mixed	33.02	35.59	32.73	22.71	**37.34**	35.08
AVG	Avg.	5.78	7.94	10.36	2.96	16.53	**19.94**

Table 22: Per-task domain memory results on BioInst for the Qwen3-1.7B-Base backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
DNA cpd	MCC	4.09	0.00	0.00	-1.90	**35.86**	31.04
DNA emp	MCC	1.26	0.00	0.00	3.92	**10.90**	6.91
DNA pd	MCC	-0.77	0.00	0.00	5.11	55.83	**58.31**
DNA tf h	MCC	5.48	0.00	0.00	6.04	22.43	**25.96**
DNA tf m	MCC	-2.83	0.00	0.00	-2.72	**34.90**	34.02
Antibody antigen	MCC	1.42	-3.43	4.14	-3.31	**5.26**	3.97
Promoter enhancer	MCC	2.98	0.00	0.00	-1.81	5.69	**7.63**
RNA protein interaction	MCC	-0.04	0.00	0.00	6.64	0.07	**14.53**
DNA enhancer activity	PCC	-0.89	9.89	15.61	-1.51	3.23	**21.65**
CRISPR on target	Spearman	-5.55	0.30	3.60	**12.25**	0.18	9.87
Protein fluorescence	Spearman	1.25	0.00	-2.20	-0.37	20.72	**33.72**
Protein stability	Spearman	-3.00	0.91	4.08	-2.32	**14.36**	11.25
Protein thermostability	Spearman	-1.29	1.11	13.81	6.84	**21.11**	5.85
RNA isoform	R2	0.04	0.00	**52.14**	0.18	32.98	38.98
Mean ribosome loading	R2	0.08	0.20	0.14	0.02	4.48	**41.72**
Programmable RNA switches	R2	0.26	7.86	9.39	0.00	1.07	**17.37**
RNA modification	AUC	50.37	50.00	50.00	51.33	50.79	**51.36**
Protein solubility	Acc.	19.30	0.00	0.10	**47.00**	27.10	29.70
Noncoding RNA family	Acc.	6.10	0.00	0.00	8.70	**8.80**	6.50
Protein function EC	Score	**2.04**	**2.04**	**2.04**	**2.04**	**2.04**	**2.04**
siRNA efficiency	Mixed	32.82	32.73	36.40	33.44	**37.34**	35.00
AVG	Avg.	5.39	4.84	9.01	8.07	18.82	**23.21**

Table 23: Per-task domain memory results on BioInst for the Qwen3-4B-Base backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
DNA cpd	MCC	-3.82	28.93	3.65	-1.47	**36.80**	35.40
DNA emp	MCC	-2.96	3.81	2.65	2.56	**12.03**	6.33
DNA pd	MCC	-2.98	2.96	-2.39	-0.24	49.59	**65.53**
DNA tf h	MCC	1.88	-0.51	-1.02	0.62	**29.23**	26.00
DNA tf m	MCC	-5.77	-0.55	-1.01	-0.34	**34.02**	31.67
Antibody antigen	MCC	4.22	4.59	2.92	**5.70**	2.28	3.80
Promoter enhancer	MCC	**4.02**	-3.16	1.67	0.10	3.17	3.36
RNA protein interaction	MCC	-2.57	0.00	0.00	12.51	4.74	**14.53**
DNA enhancer activity	PCC	-1.81	**31.59**	13.77	1.48	3.70	21.65
CRISPR on target	Spearman	1.46	0.00	0.00	10.46	4.01	**10.60**
Protein fluorescence	Spearman	-7.68	3.36	0.00	1.10	22.83	**26.97**
Protein stability	Spearman	-1.27	0.83	0.00	-5.48	6.08	**10.94**
Protein thermostability	Spearman	1.35	8.67	-0.20	5.12	**21.11**	6.61
RNA isoform	R2	0.11	1.18	21.26	0.09	35.04	**35.65**
Mean ribosome loading	R2	0.01	0.00	0.00	0.01	4.83	**42.45**
Programmable RNA switches	R2	0.00	20.58	4.49	0.16	1.35	**24.64**
RNA modification	AUC	50.26	50.00	50.00	50.86	50.74	**51.17**
Protein solubility	Acc.	33.70	4.90	37.80	**45.50**	23.10	30.50
Noncoding RNA family	Acc.	6.50	0.00	0.80	**12.00**	9.30	8.20
Protein function EC	Score	**2.04**	**2.04**	**2.04**	**2.04**	**2.04**	**2.04**
siRNA efficiency	Mixed	28.70	32.73	32.73	32.30	**37.34**	35.04
AVG	Avg.	5.02	9.14	8.06	8.34	18.73	**23.48**

Table 24: Per-task domain memory results on BioInst for the Qwen3-8B-Base backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
DNA cpd	MCC	-2.33	17.33	6.93	3.35	**36.29**	29.57
DNA emp	MCC	-0.89	-0.01	-2.11	2.72	**10.06**	7.19
DNA pd	MCC	-7.98	**43.17**	6.06	4.87	36.03	38.62
DNA tf h	MCC	-2.27	3.91	0.00	1.82	**27.21**	22.19
DNA tf m	MCC	0.24	-5.11	0.00	0.18	**31.41**	25.94
Antibody antigen	MCC	3.16	**4.62**	-1.27	0.27	4.12	1.38
Promoter enhancer	MCC	2.83	0.00	0.00	-2.29	**7.10**	6.32
RNA protein interaction	MCC	2.01	0.00	0.00	6.36	1.45	**16.26**
DNA enhancer activity	PCC	3.76	18.26	14.59	-0.14	1.77	**22.04**
CRISPR on target	Spearman	-1.95	0.00	0.00	2.64	4.01	**6.84**
Protein fluorescence	Spearman	-3.63	1.90	0.00	-2.88	22.19	**34.01**
Protein stability	Spearman	-2.25	3.30	0.00	4.17	**14.33**	13.22
Protein thermostability	Spearman	1.44	0.75	0.00	8.83	**21.11**	5.12
RNA isoform	R2	0.00	0.00	0.62	0.29	**33.86**	32.57
Mean ribosome loading	R2	0.04	0.00	0.00	0.03	4.48	**7.74**
Programmable RNA switches	R2	0.00	**23.59**	0.06	0.27	1.15	20.53
RNA modification	AUC	50.20	50.02	49.90	50.04	**50.80**	50.32
Protein solubility	Acc.	40.20	7.30	0.00	**41.30**	30.60	35.70
Noncoding RNA family	Acc.	6.20	1.80	2.10	7.70	**8.40**	7.70
Protein function EC	Score	**2.04**	**2.04**	**2.04**	**2.04**	**2.04**	**2.04**
siRNA efficiency	Mixed	10.40	32.73	32.73	31.78	**37.34**	35.17
AVG	Avg.	4.82	9.79	5.32	7.78	18.37	**20.02**

Table 25: Per-task domain memory results on BioInst for the Qwen3-14B-Base backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
DNA cpd	MCC	0.09	–	–	8.45	**35.29**	31.63
DNA emp	MCC	-6.01	–	–	-1.72	**10.32**	9.78
DNA pd	MCC	1.72	–	–	-0.27	32.60	**42.82**
DNA tf h	MCC	1.29	–	–	2.17	**27.32**	17.49
DNA tf m	MCC	-3.08	–	–	4.34	**29.08**	19.05
Antibody antigen	MCC	0.50	–	–	0.10	**9.27**	4.02
Promoter enhancer	MCC	-2.59	–	–	1.82	**5.55**	4.37
RNA protein interaction	MCC	-7.13	–	–	1.07	4.67	**12.28**
DNA enhancer activity	PCC	-1.12	–	–	0.55	3.70	**22.11**
CRISPR on target	Spearman	-3.61	–	–	**9.40**	4.27	7.83
Protein fluorescence	Spearman	-2.81	–	–	1.58	22.84	**32.78**
Protein stability	Spearman	-3.06	–	–	-0.89	8.38	**13.77**
Protein thermostability	Spearman	-2.22	–	–	-0.96	**21.11**	4.04
RNA isoform	R2	0.11	–	–	0.37	34.09	**42.87**
Mean ribosome loading	R2	0.05	–	–	0.72	4.48	**44.44**
Programmable RNA switches	R2	0.00	–	–	0.16	1.20	**19.67**
RNA modification	AUC	49.05	–	–	48.20	**50.65**	50.56
Protein solubility	Acc.	21.10	–	–	**43.20**	35.40	37.00
Noncoding RNA family	Acc.	6.30	–	–	**8.20**	7.20	7.70
Protein function EC	Score	**2.04**	–	–	**2.04**	**2.04**	**2.04**
siRNA efficiency	Mixed	33.52	–	–	33.24	**37.34**	35.06
AVG	Avg.	4.01	–	–	7.70	18.42	**21.97**

Table 26: Per-task domain memory results on BioInst for the OLMo-2-7B backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
DNA cpd	MCC	0.00	13.34	0.95	0.00	**15.64**	5.72
DNA emp	MCC	-6.14	5.77	5.02	0.00	**6.72**	4.11
DNA pd	MCC	0.00	-5.31	**18.49**	0.00	5.39	8.07
DNA tf h	MCC	0.00	-8.91	-1.84	-3.11	**10.63**	5.96
DNA tf m	MCC	0.00	3.37	-0.84	-0.13	5.04	**6.15**
Antibody antigen	MCC	0.00	3.75	-4.38	0.00	**4.53**	4.31
Promoter enhancer	MCC	0.00	0.00	-0.64	-3.16	4.35	**4.49**
RNA protein interaction	MCC	0.00	-4.22	9.06	0.00	13.96	**14.82**
DNA enhancer activity	PCC	0.49	**29.28**	15.06	-1.34	13.75	1.15
CRISPR on target	Spearman	0.00	-2.49	-1.26	0.00	**2.16**	1.44
Protein fluorescence	Spearman	–	**10.45**	-4.65	0.00	4.00	6.38
Protein stability	Spearman	-0.11	4.89	**8.01**	0.00	5.85	4.76
Protein thermostability	Spearman	-1.18	0.89	-11.80	-0.81	**2.48**	1.47
RNA isoform	R2	0.00	2.20	**26.01**	0.00	4.80	0.91
Mean ribosome loading	R2	0.00	**1.41**	0.02	0.00	0.43	0.29
Programmable RNA switches	R2	0.00	**11.68**	3.93	0.57	0.40	0.28
RNA modification	AUC	50.00	50.00	49.98	50.00	50.00	**50.06**
Protein solubility	Acc.	0.00	**48.80**	**48.80**	0.00	17.80	7.90
Noncoding RNA family	Acc.	0.00	0.00	3.30	0.00	0.90	**3.90**
Protein function EC	Score	**2.04**	**2.04**	**2.04**	**2.04**	**2.04**	**2.04**
siRNA efficiency	Mixed	32.80	32.73	32.95	33.62	**36.09**	33.41
AVG	Avg.	3.90	9.51	9.44	3.70	**9.86**	7.98

Table 27: Per-task domain memory results on BioInst for the OLMo-3-7B backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
DNA cpd	MCC	1.50	**47.01**	-4.29	3.96	14.90	5.24
DNA emp	MCC	-0.69	**5.84**	0.00	-0.01	2.78	5.31
DNA pd	MCC	7.04	**20.87**	0.68	0.11	6.30	8.10
DNA tf h	MCC	2.47	3.73	-5.51	0.95	**6.66**	5.55
DNA tf m	MCC	6.11	-1.17	**11.00**	4.53	5.36	7.25
Antibody antigen	MCC	-1.88	0.00	2.42	2.42	**4.62**	2.56
Promoter enhancer	MCC	-0.36	0.00	0.00	1.36	2.67	**4.49**
RNA protein interaction	MCC	-0.35	0.00	3.35	2.95	**15.48**	4.67
DNA enhancer activity	PCC	1.70	4.70	11.39	-0.97	**14.93**	1.80
CRISPR on target	Spearman	-3.16	4.77	1.35	2.76	3.13	**5.23**
Protein fluorescence	Spearman	3.87	**6.53**	-0.65	2.32	3.34	**6.53**
Protein stability	Spearman	0.18	**22.78**	-1.56	2.09	1.93	5.76
Protein thermostability	Spearman	1.38	4.71	-2.86	**9.51**	-0.86	0.02
RNA isoform	R2	0.02	**30.25**	0.34	0.01	5.57	0.52
Mean ribosome loading	R2	0.01	**4.57**	0.00	0.14	0.42	0.78
Programmable RNA switches	R2	0.00	**2.31**	1.84	0.18	0.55	0.32
RNA modification	AUC	49.66	49.99	50.09	48.26	50.57	**51.33**
Protein solubility	Acc.	24.90	**42.10**	36.40	3.30	16.50	16.20
Noncoding RNA family	Acc.	**7.00**	3.70	6.00	6.60	4.60	4.70
Protein function EC	Score	**2.04**	**2.04**	**2.04**	**2.04**	**2.04**	**2.04**
siRNA efficiency	Mixed	32.76	32.75	**33.85**	32.53	33.42	32.94
AVG	Avg.	6.39	**13.69**	6.95	5.95	9.28	8.16

K.2 LawBench

Table 28: Per-task domain memory results on LawBench for the Qwen3-0.6B-Base backbone.

Task	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
1.1 Article recitation	14.72	17.40	**17.55**	1.73	7.23	8.54
1.2 Knowledge question answering	23.40	8.00	6.40	**30.00**	23.80	29.60
2.1 Document proofreading	**2.88**	1.48	1.12	2.09	2.69	2.78
2.2 Dispute focus identification	5.80	3.60	4.40	3.40	5.00	**6.60**
2.3 Marital disputes identification	15.24	23.41	10.75	**38.32**	30.10	37.41
2.4 Issue topic identification	10.60	**28.40**	28.20	21.20	17.80	20.60
2.5 Reading comprehension	**33.86**	24.96	25.00	3.96	28.20	29.97
2.6 Named entity recognition	**18.36**	2.00	2.20	6.15	12.72	12.72
2.7 Opinion summarization	5.58	**24.25**	23.80	5.33	19.03	17.52
2.8 Argument mining	18.60	3.00	4.20	8.40	**20.40**	**20.40**
2.9 Event detection	11.28	22.95	28.34	14.55	28.88	**30.33**
2.10 Trigger word extraction	0.55	0.61	0.65	0.73	0.43	**1.69**
3.1 Fact based article prediction	9.95	6.37	32.68	18.09	24.84	**39.44**
3.2 Scene based article prediction	22.45	22.43	**23.01**	10.01	18.69	20.09
3.3 Charge prediction	10.49	28.04	26.88	15.47	37.60	**45.43**
3.4 Prison term prediction without article	47.08	**71.96**	71.00	51.71	58.13	62.97
3.5 Prison term prediction with article	48.39	59.57	58.81	49.92	**72.44**	63.88
3.6 Case analysis	24.00	10.60	10.60	32.80	22.60	**35.80**
3.7 Criminal damages calculation	22.20	33.80	31.20	28.20	30.40	**39.00**
3.8 Consultation	11.92	14.87	**15.18**	14.29	5.79	5.79
AVG	17.87	20.39	21.10	17.82	23.34	**26.53**

Table 29: Per-task domain memory results on LawBench for the Qwen3-1.7B-Base backbone.

Task	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
1.1 Article recitation	10.37	24.38	**26.27**	14.53	17.23	20.76
1.2 Knowledge question answering	41.40	21.40	23.40	**72.40**	27.80	37.00
2.1 Document proofreading	4.21	3.00	2.14	**4.53**	3.80	3.62
2.2 Dispute focus identification	**15.40**	5.40	5.00	8.40	9.80	10.00
2.3 Marital disputes identification	27.16	43.91	44.60	**46.82**	33.49	38.85
2.4 Issue topic identification	23.20	**34.60**	29.80	22.40	22.20	23.20
2.5 Reading comprehension	**50.82**	29.75	29.71	20.03	36.66	42.47
2.6 Named entity recognition	6.91	4.55	2.28	**6.94**	3.77	3.79
2.7 Opinion summarization	21.73	**30.74**	27.51	15.07	20.70	20.42
2.8 Argument mining	**25.20**	1.80	8.80	21.80	19.40	19.20
2.9 Event detection	16.79	**60.54**	53.94	23.01	31.12	37.19
2.10 Trigger word extraction	2.94	3.41	**4.23**	1.51	1.81	2.00
3.1 Fact based article prediction	27.04	41.02	33.16	35.97	32.47	**53.50**
3.2 Scene based article prediction	14.33	**24.53**	23.07	14.63	17.52	16.96
3.3 Charge prediction	34.99	45.77	34.43	37.45	36.67	**46.29**
3.4 Prison term prediction without article	69.25	71.84	**75.48**	70.50	72.38	73.20
3.5 Prison term prediction with article	63.73	42.80	59.49	59.67	**72.44**	66.37
3.6 Case analysis	33.60	26.80	27.00	**64.80**	24.40	44.40
3.7 Criminal damages calculation	34.40	43.80	44.20	40.20	43.20	**44.40**
3.8 Consultation	13.75	**17.11**	16.12	11.64	10.34	12.03
AVG	26.86	28.86	28.53	29.61	26.86	**30.78**

Table 30: Per-task domain memory results on LawBench for the Qwen3-4B-Base backbone.

Task	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
1.1 Article recitation	20.11	10.83	23.83	**25.47**	19.20	21.78
1.2 Knowledge question answering	21.60	28.00	32.00	**83.20**	52.40	62.60
2.1 Document proofreading	**6.01**	5.08	3.79	3.54	5.87	5.88
2.2 Dispute focus identification	**20.20**	4.20	9.20	14.20	9.00	12.60
2.3 Marital disputes identification	27.12	**53.55**	26.28	52.03	33.41	40.44
2.4 Issue topic identification	31.40	**33.60**	27.40	4.80	31.40	32.00
2.5 Reading comprehension	3.20	**30.25**	30.18	21.73	18.60	22.10
2.6 Named entity recognition	10.81	21.73	**32.40**	18.96	9.69	10.65
2.7 Opinion summarization	12.53	**31.28**	29.47	7.01	17.49	18.82
2.8 Argument mining	0.20	22.60	29.20	18.40	27.60	**33.40**
2.9 Event detection	40.45	**69.98**	54.92	27.23	31.37	38.94
2.10 Trigger word extraction	11.19	5.23	**12.28**	6.04	5.91	5.58
3.1 Fact based article prediction	55.44	65.82	39.20	74.88	78.19	**79.56**
3.2 Scene based article prediction	16.42	24.61	26.82	**39.35**	16.51	16.72
3.3 Charge prediction	45.94	**56.46**	42.02	42.72	46.05	48.39
3.4 Prison term prediction without article	79.43	**83.40**	76.80	60.02	79.70	79.70
3.5 Prison term prediction with article	79.66	77.31	74.98	40.64	79.93	**80.21**
3.6 Case analysis	11.20	33.40	34.00	44.80	38.80	**57.80**
3.7 Criminal damages calculation	37.80	46.60	51.00	49.40	49.60	**53.00**
3.8 Consultation	**18.72**	18.14	17.18	18.03	17.44	17.71
AVG	27.47	36.10	33.65	32.62	33.41	**36.89**

Table 31: Per-task domain memory results on LawBench for the Qwen3-8B-Base backbone.

Task	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
1.1 Article recitation	24.34	**48.07**	42.17	27.46	25.82	24.91
1.2 Knowledge question answering	37.20	31.40	38.40	**92.00**	66.20	72.80
2.1 Document proofreading	**8.54**	6.31	5.47	6.33	7.82	8.14
2.2 Dispute focus identification	13.40	10.60	5.20	13.80	12.80	**14.00**
2.3 Marital disputes identification	25.97	60.11	33.16	**63.28**	36.77	41.08
2.4 Issue topic identification	**32.40**	31.40	30.20	20.60	31.60	31.60
2.5 Reading comprehension	17.53	33.81	**35.84**	20.10	29.36	30.72
2.6 Named entity recognition	42.87	13.04	30.79	20.92	**46.00**	45.20
2.7 Opinion summarization	6.79	**33.53**	29.61	11.65	19.61	19.25
2.8 Argument mining	3.60	31.00	24.60	37.60	39.00	**42.80**
2.9 Event detection	43.96	**62.21**	51.04	37.10	40.39	43.25
2.10 Trigger word extraction	8.76	8.89	**11.31**	8.52	5.04	5.30
3.1 Fact based article prediction	79.00	59.42	8.64	**83.17**	79.09	80.58
3.2 Scene based article prediction	26.44	24.13	27.54	**35.23**	25.57	26.71
3.3 Charge prediction	38.21	**56.26**	49.55	43.23	43.42	49.30
3.4 Prison term prediction without article	80.24	82.24	**82.38**	81.80	79.77	79.77
3.5 Prison term prediction with article	79.90	80.01	62.91	57.24	**80.32**	**80.32**
3.6 Case analysis	3.40	32.00	41.60	**80.80**	43.20	60.60
3.7 Criminal damages calculation	61.00	**62.00**	42.00	53.60	60.40	**62.00**
3.8 Consultation	**19.86**	18.42	18.49	18.95	19.53	19.85
AVG	32.67	39.24	33.55	40.67	39.59	**41.91**

Table 32: Per-task domain memory results on LawBench for the Qwen3-14B-Base backbone.

Task	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
1.1 Article recitation	31.50	–	–	30.95	30.29	**31.59**
1.2 Knowledge question answering	17.20	–	–	**97.40**	51.40	70.20
2.1 Document proofreading	8.76	–	–	**9.57**	8.52	8.52
2.2 Dispute focus identification	**29.40**	–	–	25.60	**29.40**	**29.40**
2.3 Marital disputes identification	21.00	–	–	**69.57**	42.56	39.77
2.4 Issue topic identification	33.20	–	–	**36.60**	33.20	33.20
2.5 Reading comprehension	3.79	–	–	12.94	26.91	**28.93**
2.6 Named entity recognition	35.51	–	–	23.73	**37.86**	37.08
2.7 Opinion summarization	21.80	–	–	18.78	22.14	**22.36**
2.8 Argument mining	0.20	–	–	25.80	9.20	**43.60**
2.9 Event detection	48.08	–	–	26.76	48.79	**50.19**
2.10 Trigger word extraction	**21.64**	–	–	9.53	14.07	14.26
3.1 Fact based article prediction	85.57	–	–	84.41	**85.91**	**85.91**
3.2 Scene based article prediction	26.35	–	–	**36.40**	26.70	26.20
3.3 Charge prediction	38.59	–	–	49.33	44.38	**51.68**
3.4 Prison term prediction without article	82.10	–	–	**82.82**	81.98	81.98
3.5 Prison term prediction with article	80.15	–	–	69.00	**80.27**	**80.27**
3.6 Case analysis	38.60	–	–	**92.20**	40.80	67.00
3.7 Criminal damages calculation	66.40	–	–	66.40	66.40	**67.20**
3.8 Consultation	19.24	–	–	**19.78**	19.08	19.08
AVG	35.45	–	–	44.38	39.99	**44.42**

Table 33: Per-task domain memory results on LawBench for the OLMo-2-7B backbone.

Task	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
1.1 Article recitation	0.00	**15.80**	4.15	0.38	3.98	7.10
1.2 Knowledge question answering	0.00	7.20	6.00	10.20	**20.20**	**20.20**
2.1 Document proofreading	0.56	**1.34**	1.05	0.94	0.90	1.05
2.2 Dispute focus identification	3.60	3.60	**4.20**	2.80	**4.20**	2.80
2.3 Marital disputes identification	7.12	**40.12**	8.13	20.61	2.59	2.40
2.4 Issue topic identification	6.80	**27.60**	10.60	19.60	7.60	7.60
2.5 Reading comprehension	2.03	**27.11**	26.85	2.95	16.46	18.52
2.6 Named entity recognition	2.36	2.00	**3.25**	2.34	2.42	2.46
2.7 Opinion summarization	7.37	**27.66**	21.65	9.15	12.00	11.17
2.8 Argument mining	0.20	0.80	0.20	0.80	**21.40**	20.20
2.9 Event detection	12.55	34.85	**38.55**	17.41	14.80	14.93
2.10 Trigger word extraction	0.30	0.87	**2.96**	2.12	0.37	0.70
3.1 Fact based article prediction	0.14	**26.52**	2.34	3.08	11.52	10.91
3.2 Scene based article prediction	3.28	**23.56**	22.94	5.05	13.57	11.03
3.3 Charge prediction	7.27	**40.81**	21.57	12.73	27.52	20.85
3.4 Prison term prediction without article	21.61	59.17	32.10	46.01	**67.71**	59.15
3.5 Prison term prediction with article	32.94	55.42	16.63	45.10	**56.47**	37.12
3.6 Case analysis	0.00	12.80	5.20	13.00	**19.40**	**19.40**
3.7 Criminal damages calculation	13.00	**34.60**	21.20	18.20	24.40	23.40
3.8 Consultation	6.34	**17.08**	14.68	9.71	8.10	6.81
AVG	6.37	**22.95**	13.21	12.11	16.78	14.89

Table 34: Per-task domain memory results on LawBench for the OLMo-3-7B backbone.

Task	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
1.1 Article recitation	3.52	**18.26**	5.88	6.43	8.68	11.52
1.2 Knowledge question answering	9.40	14.60	6.40	**34.20**	28.20	27.80
2.1 Document proofreading	2.07	**2.69**	2.04	1.71	2.07	2.07
2.2 Dispute focus identification	7.80	4.80	5.00	5.00	**8.20**	7.80
2.3 Marital disputes identification	12.26	20.55	6.73	**54.61**	3.25	12.26
2.4 Issue topic identification	17.20	**34.60**	0.60	27.00	17.20	17.20
2.5 Reading comprehension	2.32	28.19	4.54	3.07	**42.72**	**42.72**
2.6 Named entity recognition	**16.08**	3.43	2.01	7.48	16.02	16.02
2.7 Opinion summarization	6.05	**25.84**	10.37	4.90	13.73	13.33
2.8 Argument mining	7.60	3.60	5.20	15.20	**20.00**	**20.00**
2.9 Event detection	14.62	42.00	**51.94**	16.89	14.69	14.69
2.10 Trigger word extraction	1.41	**2.62**	0.60	0.84	2.23	2.23
3.1 Fact based article prediction	1.18	2.52	18.79	22.70	**26.11**	25.61
3.2 Scene based article prediction	5.90	**24.48**	9.94	12.15	15.77	18.37
3.3 Charge prediction	12.50	**39.52**	28.00	23.22	26.33	23.41
3.4 Prison term prediction without article	35.11	30.90	53.03	57.53	**67.31**	58.77
3.5 Prison term prediction with article	**62.59**	20.63	51.10	60.48	**62.59**	**62.59**
3.6 Case analysis	9.20	20.20	18.00	**43.00**	30.80	30.80
3.7 Criminal damages calculation	28.80	**50.20**	35.80	29.40	29.60	29.60
3.8 Consultation	8.21	**16.25**	12.06	11.13	11.24	11.24
AVG	13.19	20.29	16.40	21.85	22.34	**22.40**

K.3 FINEVAL

All FinEval subtasks use `generate_until`. The metric column lists the primary lm-evaluation-harness metric for each task after multiplying by 100. EM denotes exact match after answer normalization, MCC denotes Matthews correlation coefficient over extracted labels, and ROUGE-1 denotes the unigram F-measure for generated text. AVG is the macro average over the 25 task scores.

Table 35: Per-task domain memory results on FinEval for the Qwen3-0.6B-Base backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
CFA-Challenge	EM	0.00	4.44	5.56	1.11	11.11	**25.56**
CFA-Easy	EM	22.67	32.27	21.12	25.10	**39.34**	29.84
CRA-Bigdata	EM	30.57	**54.96**	43.21	38.25	30.84	54.14
CRA-CCF	MCC	0.00	2.61	0.00	-1.37	0.58	**4.37**
CRA-CCFraud	EM	6.01	0.00	12.35	**38.04**	9.01	7.05
CRA-LendingClub	EM	78.93	16.16	59.35	23.86	**79.52**	79.23
CRA-Polish	MCC	0.36	0.00	**1.12**	0.00	0.54	0.88
CRA-ProtoSeguro	EM	3.02	3.02	**63.10**	11.38	3.02	4.28
CRA-Taiwan	EM	3.22	0.00	3.22	3.22	**30.62**	3.22
CRA-TravelInsurance	EM	49.88	1.10	0.00	34.69	80.98	**87.88**
ECTSUM	ROUGE-1	29.08	13.98	12.46	27.18	**29.34**	**29.34**
EDTSUM	ROUGE-1	**18.04**	5.09	8.97	17.71	17.98	17.98
FIQASA	EM	38.72	43.83	27.23	48.94	**61.70**	46.38
FOMC	EM	27.02	**30.85**	29.84	28.02	30.04	27.82
FPB	EM	41.86	17.11	25.26	50.21	**56.08**	52.27
FinanceBench	EM	78.93	16.16	59.35	23.86	**79.52**	79.23
Flare-Australian	EM	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Flare-German	EM	0.00	0.00	0.00	0.00	**13.50**	0.00
Flare-TATQA	EM	1.38	0.78	0.96	**1.62**	1.32	1.38
MA	EM	65.00	46.00	57.20	**65.20**	64.60	64.20
MLESG	EM	11.67	4.67	3.33	**22.67**	11.67	12.33
MMLU-finance	EM	14.85	6.02	6.23	0.14	14.65	**24.50**
NER	ROUGE-1	14.90	6.28	8.68	12.84	**16.15**	**16.15**
SM-ACL	EM	24.27	41.10	32.45	35.05	27.80	**45.30**
SM-CIKM	EM	22.48	26.07	7.61	26.60	25.46	**39.72**
AVG	Avg.	23.31	14.90	19.54	21.37	29.42	**30.12**

Table 36: Per-task domain memory results on FinEval for the Qwen3-1.7B-Base backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
CFA-Challenge	EM	0.00	4.44	11.11	1.11	12.22	**24.44**
CFA-Easy	EM	3.29	47.58	**55.52**	34.21	40.41	25.10
CRA-Bigdata	EM	30.64	43.48	46.60	40.69	31.86	**52.99**
CRA-CCF	MCC	-0.60	-1.59	0.00	-1.05	**0.87**	0.04
CRA-CCFraud	EM	55.00	5.86	77.31	**90.37**	56.20	56.20
CRA-LendingClub	EM	39.20	75.96	**79.19**	75.66	52.21	54.14
CRA-Polish	MCC	-1.37	0.79	0.00	-3.71	-1.37	**0.92**
CRA-ProtoSeguro	EM	28.59	37.57	**96.98**	80.27	31.11	31.11
CRA-Taiwan	EM	**96.78**	3.22	**96.78**	**96.78**	**96.78**	**96.78**
CRA-TravelInsurance	EM	1.62	1.46	1.50	1.50	**77.78**	1.54
ECTSUM	ROUGE-1	29.41	22.45	18.45	28.24	29.67	**29.71**
EDTSUM	ROUGE-1	19.01	12.77	15.09	17.87	**19.07**	**19.07**
FIQASA	EM	40.00	**71.49**	62.55	47.66	55.74	50.64
FOMC	EM	51.01	39.11	41.73	48.39	51.61	**52.02**
FPB	EM	55.46	63.09	**64.02**	49.90	60.72	60.41
FinanceBench	EM	39.20	75.96	**79.19**	75.66	52.21	54.14
Flare-Australian	EM	0.00	0.00	0.00	**24.46**	0.00	0.00
Flare-German	EM	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Flare-TATQA	EM	6.59	3.96	**18.94**	8.81	6.59	6.59
MA	EM	53.00	**69.80**	60.80	68.20	53.60	60.60
MLESG	EM	18.00	14.00	**26.67**	23.33	17.33	17.33
MMLU-finance	EM	**52.29**	32.10	12.66	32.58	51.61	51.81
NER	ROUGE-1	32.57	15.54	17.49	20.30	**33.94**	32.80
SM-ACL	EM	23.06	42.82	**46.08**	35.32	22.90	45.22
SM-CIKM	EM	8.84	**43.66**	39.81	19.16	10.50	37.62
AVG	Avg.	27.26	29.02	**38.74**	36.63	34.54	34.45

Table 37: Per-task domain memory results on FinEval for the Qwen3-4B-Base backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
CFA-Challenge	EM	0.00	**25.56**	24.44	11.11	12.22	24.44
CFA-Easy	EM	49.03	62.98	**67.34**	49.61	63.08	54.17
CRA-Bigdata	EM	54.01	40.42	**55.10**	52.65	54.55	**55.10**
CRA-CCF	MCC	0.00	1.38	0.00	0.08	0.58	**7.34**
CRA-CCFraud	EM	**9.82**	5.86	5.82	6.58	8.58	8.20
CRA-LendingClub	EM	**20.77**	19.29	0.00	**20.77**	20.59	20.62
CRA-Polish	MCC	15.25	-1.10	12.98	-2.24	**15.56**	**15.56**
CRA-ProtoSeguro	EM	86.73	1.51	0.00	**96.98**	84.47	84.47
CRA-Taiwan	EM	96.70	3.22	**96.78**	**96.78**	96.70	96.70
CRA-TravelInsurance	EM	1.50	1.50	1.50	1.50	**84.85**	1.50
ECTSUM	ROUGE-1	30.85	27.26	22.79	30.77	**30.94**	30.88
EDTSUM	ROUGE-1	20.43	**21.18**	19.21	17.74	20.50	20.50
FIQASA	EM	56.60	51.49	**71.91**	64.26	59.57	57.87
FOMC	EM	56.65	53.63	48.39	54.64	57.46	**58.06**
FPB	EM	77.11	56.29	71.34	70.93	77.63	**78.04**
FinanceBench	EM	**20.77**	19.29	0.00	**20.77**	20.59	20.62
Flare-Australian	EM	2.88	**56.83**	12.23	41.01	4.32	4.32
Flare-German	EM	10.00	28.00	17.00	**29.50**	11.00	11.00
Flare-TATQA	EM	21.22	17.09	**26.68**	11.75	20.68	20.68
MA	EM	84.20	**84.60**	72.40	80.00	**84.60**	**84.60**
MLESG	EM	24.00	9.00	**29.00**	27.67	23.33	23.33
MMLU-finance	EM	76.59	61.05	66.26	73.44	**76.93**	**76.93**

Table 37 continued

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
NER	ROUGE-1	26.68	27.87	**29.28**	19.82	28.46	29.04
SM-ACL	EM	43.71	42.15	43.63	48.60	44.14	**49.52**
SM-CIKM	EM	50.22	39.11	44.01	53.63	50.22	**55.12**
AVG	Avg.	37.43	30.22	33.52	39.13	**42.06**	39.54

Table 38: Per-task domain memory results on FinEval for the Qwen3-8B-Base backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
CFA-Challenge	EM	4.44	8.89	0.00	3.33	11.11	**27.78**
CFA-Easy	EM	36.63	47.00	37.79	31.30	**62.02**	48.26
CRA-Bigdata	EM	52.65	54.14	53.94	50.54	53.94	**54.82**
CRA-CCF	MCC	1.18	1.33	**6.66**	-0.20	4.07	4.07
CRA-CCFraud	EM	37.94	5.86	5.86	17.11	36.51	**38.08**
CRA-LendingClub	EM	62.54	42.55	**67.19**	58.60	62.02	62.02
CRA-Polish	MCC	13.64	2.34	11.17	11.42	14.49	**14.61**
CRA-ProtoSeguro	EM	**96.98**	**96.98**	**96.98**	**96.98**	**96.98**	**96.98**
CRA-Taiwan	EM	**96.70**	**96.70**	**96.70**	**96.70**	**96.70**	**96.70**
CRA-TravelInsurance	EM	41.71	0.04	5.96	91.55	**93.05**	46.25
ECTSUM	ROUGE-1	29.28	29.06	**29.46**	28.22	29.31	29.31
EDTSUM	ROUGE-1	20.11	19.06	19.39	17.74	**20.16**	**20.16**
FIQASA	EM	62.13	63.83	62.13	59.57	**66.81**	65.53
FOMC	EM	56.65	55.24	55.44	55.44	**57.46**	57.26
FPB	EM	78.97	74.95	77.42	72.47	**79.18**	**79.18**
FinanceBench	EM	62.54	42.55	**67.19**	58.60	62.02	62.02
Flare-Australian	EM	5.04	0.00	22.30	**42.45**	5.76	5.76
Flare-German	EM	0.00	0.00	0.00	**65.50**	0.00	0.00
Flare-TATQA	EM	3.54	**23.86**	12.35	13.61	3.84	3.84
MA	EM	79.00	79.60	76.20	78.20	78.60	**80.40**
MLESG	EM	**37.67**	30.33	34.67	34.67	37.33	37.33
MMLU-finance	EM	78.99	72.69	68.38	76.93	78.78	**79.40**
NER	ROUGE-1	45.60	45.66	44.57	17.78	46.99	**47.06**
SM-ACL	EM	45.51	**51.34**	**51.34**	47.82	47.15	50.03
SM-CIKM	EM	38.23	55.91	**57.66**	47.86	39.81	51.44
AVG	Avg.	43.51	40.00	42.43	46.97	**47.36**	46.33

Table 39: Per-task domain memory results on FinEval for the Qwen3-14B-Base backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
CFA-Challenge	EM	3.33	–	–	8.89	14.44	**28.89**
CFA-Easy	EM	8.04	–	–	18.80	**50.29**	29.36
CRA-Bigdata	EM	48.10	–	–	44.23	49.25	**53.53**
CRA-CCF	MCC	5.15	–	–	-0.24	5.09	**6.32**
CRA-CCFraud	EM	48.95	–	–	39.28	**49.00**	**49.00**
CRA-LendingClub	EM	62.17	–	–	49.39	63.17	**63.51**
CRA-Polish	MCC	**13.32**	–	–	11.87	12.76	12.77
CRA-ProtoSeguro	EM	**96.98**	–	–	**96.98**	**96.98**	**96.98**
CRA-Taiwan	EM	**96.70**	–	–	**96.70**	**96.70**	**96.70**
CRA-TravelInsurance	EM	63.58	–	–	15.07	**93.45**	69.49
ECTSUM	ROUGE-1	31.18	–	–	29.89	**31.35**	31.31
EDTSUM	ROUGE-1	**21.30**	–	–	18.22	21.20	21.20
FIQASA	EM	79.57	–	–	60.43	80.43	**81.28**
FOMC	EM	60.28	–	–	**61.49**	60.08	60.48
FPB	EM	80.31	–	–	78.14	**80.82**	**80.82**
FinanceBench	EM	62.17	–	–	49.39	63.17	**63.51**
Flare-Australian	EM	0.00	–	–	**51.80**	0.00	0.00
Flare-German	EM	0.00	–	–	**9.00**	0.00	0.00
Flare-TATQA	EM	2.76	–	–	**3.06**	2.70	2.70
MA	EM	**84.20**	–	–	82.80	83.80	84.00
MLESG	EM	32.67	–	–	**35.67**	32.67	33.00
MMLU-finance	EM	81.66	–	–	81.11	81.66	**81.72**
NER	ROUGE-1	**43.14**	–	–	38.67	42.77	42.77
SM-ACL	EM	46.51	–	–	47.04	46.21	**49.19**
SM-CIKM	EM	34.30	–	–	42.69	34.73	**43.83**
AVG	Avg.	44.25	–	–	42.81	**47.71**	47.29

Table 40: Per-task domain memory results on FinEval for the OLMo-2-7B backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
CFA-Challenge	EM	34.44	32.22	27.78	24.44	34.44	**35.56**
CFA-Easy	EM	53.49	48.45	51.55	**56.69**	53.97	54.17
CRA-Bigdata	EM	46.94	47.62	44.63	45.31	**48.98**	47.55
CRA-CCF	MCC	0.00	0.11	0.13	0.20	**2.28**	0.82
CRA-CCFraud	EM	83.41	**94.14**	86.37	32.13	84.22	84.22
CRA-LendingClub	EM	19.77	52.62	19.47	**65.66**	28.61	19.73
CRA-Polish	MCC	**9.73**	0.75	0.00	0.00	5.65	5.36
CRA-ProtoSeguro	EM	**96.98**	**96.98**	**96.98**	**96.98**	**96.98**	**96.98**
CRA-Taiwan	EM	**96.78**	6.01	5.86	52.31	**96.78**	**96.78**
CRA-TravelInsurance	EM	94.24	1.54	1.93	11.01	**98.38**	94.48
ECTSUM	ROUGE-1	16.15	19.08	14.46	**20.53**	17.34	19.21
EDTSUM	ROUGE-1	**40.79**	30.48	35.50	37.78	40.71	40.71
FIQASA	EM	50.21	22.98	**55.32**	41.70	53.19	51.06
FOMC	EM	52.22	**52.82**	52.22	52.42	**52.82**	52.42
FPB	EM	72.78	72.37	47.84	**75.46**	72.78	72.78
FinanceBench	EM	19.77	52.62	19.47	**65.66**	28.61	19.73
Flare-Australian	EM	43.17	0.00	0.00	19.42	**57.55**	43.88
Flare-German	EM	**66.00**	0.00	8.00	1.50	**66.00**	**66.00**
Flare-TATQA	EM	31.41	3.42	2.88	26.92	31.47	**31.59**
MA	EM	78.40	52.60	49.20	**84.80**	79.00	79.00
MLESG	EM	25.67	21.33	12.00	23.67	**26.00**	25.33
MMLU-finance	EM	56.95	52.09	54.14	**61.81**	56.74	56.74

Table 40 continued

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
NER	ROUGE-1	23.71	**27.96**	26.04	24.88	23.30	23.31
SM-ACL	EM	49.14	44.73	49.09	**50.99**	49.25	49.25
SM-CIKM	EM	48.64	32.28	47.59	**49.78**	48.56	48.47
AVG	Avg.	48.43	34.61	32.34	40.88	**50.15**	48.61

Table 41: Per-task domain memory results on FinEval for the OLMo-3-7B backbone.

Task	Metric	Base	CPT	LoRA	RAG	+Mem 0.6B	+Mem 1.7B
CFA-Challenge	EM	23.33	**26.67**	20.00	17.78	25.56	25.56
CFA-Easy	EM	56.30	49.90	52.42	52.71	**58.33**	56.78
CRA-Bigdata	EM	44.57	44.77	41.85	49.25	**49.46**	47.89
CRA-CCF	MCC	0.00	0.00	-0.45	0.00	**11.22**	2.18
CRA-CCFraud	EM	10.77	14.68	10.53	23.02	11.11	**77.93**
CRA-LendingClub	EM	45.00	**80.75**	13.79	26.20	44.93	44.93
CRA-Polish	MCC	0.00	-0.97	1.29	**3.79**	3.50	1.50
CRA-ProtoSeguro	EM	**96.98**	3.32	3.48	11.96	**96.98**	**96.98**
CRA-Taiwan	EM	**96.78**	79.34	**96.78**	**96.78**	**96.78**	**96.78**
CRA-TravelInsurance	EM	1.50	1.54	22.97	1.50	**98.50**	**98.50**
ECTSUM	ROUGE-1	26.85	26.01	26.05	25.67	**27.63**	**27.63**
EDTSUM	ROUGE-1	26.62	**38.07**	15.63	18.20	37.99	37.99
FIQASA	EM	71.06	**77.02**	56.60	63.83	71.06	71.91
FOMC	EM	22.58	47.58	21.17	44.56	**52.82**	52.42
FPB	EM	80.82	76.80	58.45	71.24	**80.93**	**80.93**
FinanceBench	EM	45.00	**80.75**	13.79	26.20	44.93	44.93
Flare-Australian	EM	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Flare-German	EM	0.00	**30.00**	0.00	0.00	0.00	0.00
Flare-TATQA	EM	13.67	16.07	5.94	5.04	40.29	**40.41**
MA	EM	50.40	64.20	48.40	75.80	**82.80**	82.00
MLESG	EM	16.33	27.00	29.33	**34.67**	32.67	32.67
MMLU-finance	EM	62.77	59.34	62.42	62.49	**63.52**	**63.52**
NER	ROUGE-1	15.87	**37.42**	16.78	12.38	31.76	31.64
SM-ACL	EM	50.22	48.23	41.77	50.05	50.13	**50.32**
SM-CIKM	EM	53.28	43.83	46.37	**54.33**	53.63	53.63
AVG	Avg.	36.43	38.89	28.21	33.10	46.66	**48.76**

L LIMITATIONS

Although inference requires no external retrieval, constructing the kNN target distributions still incurs indexing and retrieval overhead during memory pretraining. This offline cost remains an additional preprocessing burden that grows with corpus scale despite the compressed and sharded construction pipeline. Future work could replace the fixed interpolation coefficient α with an adaptive weighting mechanism that determines the memory contribution for each input based on the context or model confidence. It could also explore joint or staged training of the memory and backbone to improve their coordination while preserving modular deployment. Such training may improve integration but would increase training cost and could weaken cross-backbone transfer.