Evaluating Long-Term Memory for Long-Context Question Answering

Anonymous ACL submission

Abstract

In order for large language models to achieve true conversational continuity and benefit from experiential learning, they need memory. While research has focused on the development of complex memory systems, it remains unclear which types of memory are most effective for long-context conversational tasks. We present a systematic evaluation of memory-augmented methods using LoCoMo, a benchmark of synthetic long-context dialogues annotated for question-answering tasks that require diverse reasoning strategies. We analyse full-context prompting, semantic memory through retrieval-augmented generation and agentic memory, episodic memory through in-context learning, and procedural memory through prompt optimization. Our findings show that memory-augmented approaches reduce token usage by over 90% while maintaining competitive accuracy. Memory architecture complexity should scale with model capability, with small foundation models benefitting most from RAG, and strong instruction-tuned reasoning model gaining from episodic learning through reflections and more complex agentic semantic memory. In particular, episodic memory can help LLMs recognise the limits of their own knowledge.

1 Introduction

Memory is at the core of how humans think, learn, and make decisions. Similarly, for large language model (LLM) based agents, memory is fundamental to carry out extended conversations, reason over longer time frames, learn from experience, and act in a coherent way (Liu et al., 2025). LLMs often struggle to maintain coherence and accuracy in long, multi-turn interactions and very extended contexts. They also lack reliable mechanisms for metacognition, for instance they struggle with recognising the limits of their own knowledge or "knowing what they don't know" (Johnson et al., 2024).

These challenges gain more importance as context windows become larger, conversations can span hundreds of turns, and learning from experience becomes essential in real-world applications. Simply feeding models longer context is not always practical or efficient, as it increases inference cost and does not address the need for models to also learn from past information. Techniques such as retrieval-augmented generation (RAG) (Lewis et al., 2020) address context limitations and allow updating and controlling data sources to improve the factuality of generated text. Additionally, recent advancements have introduced mechanisms to augment LLMs with memory components, to enhance their ability of retaining information over long contexts, and of learning from experience (Shinn et al., 2023; Zhao et al., 2024).

In this paper, we examine different memory architectures, semantic (RAG and A-Mem), episodic (in-context learning with reflections), and procedural (prompt optimization), within a unified evaluation framework. We focus on long-context conversational QA, and reflect on how memory augmentation can support learning from experience and improve the models' ability to reason about their own knowledge. Our findings show that memory-augmented approaches reduce token usage by over 90% while maintaining competitive accuracy. The complexity of the memory architecture should scale with the model's capabilities: smaller foundation models benefit most from RAG, while more advanced instruction-tuned models from episodic memory and richer semantic memory structures. Additionally, episodic memory plays an important role in mitigating LLMs' metacognitive limitations, helping them recognise the limits of their knowledge. This comparison establishes baselines and identifies trade-offs between performance, interpretability, and resource constraints that can inform memory system design.

2 Background

2.1 Memory for LLM agents

The need for a dedicated memory component in LLM-based agents seems fundamental to capture the model's internal reasoning, its task-specific context, historical dialogue, and evolving objectives (Liu et al., 2025; Zhang et al., 2024).

Similarly to human memory, memory modules for LLMs can be categorised into long and short-term memory. Short-term memory – usually the immediate context window of the LLM – is sufficient to deal with simple tasks, but long-term memory becomes essential in more complex and realistic interaction scenarios (Khosla et al., 2023). Following the organization typically used for human memory, long-term memory is organised into three different mechanisms: **semantic memory** stores general world knowledge and facts, making it possible for the model to provide informed responses; **procedural memory** encodes knowledge about how to perform the task at hand; **episodic memory** finally captures specific past interactions or experiences, allowing the model to gather information that can inform its future decision making (Hatalis et al., 2023).

Early approaches to memory through appending conversation history to the input prompt have evolved (Liu et al., 2025), leading to the use of vector embeddings for retrieving memories and selective incorporation of reasoning steps into subsequent inference calls (Liu et al., 2023a). A growing body of work has been investigating dedicated memory systems for LLMs, exploring memory storage and management (Zhong et al., 2024; Roucher et al., 2025). Some systems maintain logs of each agent interaction (Zhong et al., 2024), while others employ read-write mechanisms that allow models to update memory content as needed (Modarressi et al., 2023). Recent research has been exploring more dynamic and agent-driven approaches to memory. A-mem (Xu et al., 2025) proposes a semantic memory system where each memory is structured with contextual tags and dynamically linked to related memories. This enables the system to form an evolving network of knowledge that supports updates over time and memory reorganization.

2.2 Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) (Lewis et al., 2020) can improve LLMs efficiency and traceability, providing a way to capture and update knowledge in a modular and more interpretable way (Guu et al., 2020). Retrieval-augmented LMs can be more reliable, adaptable, and attributable thanks to the use of large-scale datastores for knowledge-intensive tasks (Asai et al., 2024).

The standard RAG process involves chunking a set of documents, embedding and indexing them in a vectorstore, and retrieving relevant chunks based on semantic similarity with the query; this retrieved context can then be used to augment the LLM's prompt for generation. Recent research has made these systems agentic by introducing more autonomy in determining what to retrieve and when, possibly having different retrievers available and refining the search strategies based on intermediate results (Shao et al., 2023).

2.3 Learning through memory

A growing interest has developed in leveraging LLM-based agents' experience to improve them without relying on costly fine-tuning or parametric updates. Traditional reinforcement learning approaches, while effective in many domains, often require extensive interaction data and prolonged training times. In response to these limitations, several frameworks have been proposed that enable language agents to learn through memory and linguistic feedback.

One such approach is Reflexion (Shinn et al., 2023), which allows agents to self-improve through natural language reflection stored in an episodic memory buffer, rather than weight updates. Building on Reflexion, the Experiential Learning (ExpeL) framework (Zhao et al., 2024) proposes a memory-based strategy in which agents autonomously learn from experience without gradient updates, highlighting their capacity for transfer learning and emergent reasoning abilities.

Our approach to episodic memory, motivated by this literature, also makes use of textual feedback signals and knowledge from past experiences, but it is built with conversational tasks in mind, and to be modular and easy to evaluate in conjunction with different LLM memory components. These developments point to a paradigm shift in how learning is implemented in LLM-based agents, moving towards dynamic, introspective, and context-aware memory systems.

3 Methodologies

3.1 Dataset and evaluation

Our goal is to test different memory strategies on a task which models realistic long-context conversations, which is why we decided to use LoCoMo (Long-term Conversational Memory) (Maharana et al., 2024). LoCoMo is a publicly available evaluation benchmark consisting of very long-context synthetic conversational data generated through a machine-human pipeline. While synthetic data has limitations regarding real-world noise and ambiguous user intent, it provides controlled evaluation conditions that allow us to isolate memory mechanism effects without confounding variables from dialogue structure or inconsistent annotation quality. While previous datasets contain dialogues with around 1,000 tokens, spanning over 4-5 sessions, LoCoMo consists of ten conversations, spread across up to 35 chat sessions, each extending over 300 turns and averaging 9,000 tokens. The dataset is granularly annotated for the question-answering and event-summarization tasks and presents a comprehensive benchmark to evaluate long-term memory in models.

An additional reason to choose this dataset, is that the questions are divided into five distinct reasoning types: single-hop, multi-hop, temporal, open-domain or world knowledge, and adversarial. In particular, temporal reasoning questions require the model to use the date of conversations, while adversarial questions do not have an answer in the given data, presenting both a challenge and possible insights into the trustworthiness of the examined systems.

For evaluation, we employ the F1 score to assess answer accuracy. As QA annotations are directly taken from the conversations as much as possible (Maharana et al., 2024), we instruct the models to answer with exact words from the conversation. As a measure of relative performance, we report the average F1 ranking across categories for each approach examined, and we use the average token length per query to measure efficiency. Following Maharana et al. (2024), for adversarial questions we set the evaluation score to 1 if the generated answer contains 'no information available' and to 0 otherwise.

Unlike previous research that used different prompts depending on the type of question (Maharana et al., 2024; Xu et al., 2025), we prompt the models in the same way regardless of the question type, making the task more realistic and allowing better isolation of memory mechanism effects.

3.2 Memory for Question Answering

We compare several approaches to memory under the QA setting. Table 1 shows which types of memory are implemented by each approach.

	Short-term	Long-term			
Strategy	**Working**	**Semantic**	**Procedural**	**Episodic**	**Agentic**
Full Context	✓				
RAG	✓	✓			
A-Mem	✓	✓			✓
RAG + PromptOpt	✓		✓		
RAG + EpMem	✓	✓		✓	
RAG + PromptOpt + EpMem	✓	✓	✓	✓	

Table 1: **Comparison of memory strategies.** Each approach is categorised by memory types: short-term working memory (via context window), long-term memory (semantic, procedural, episodic), and whether it includes agentic control.

Full Context Prompting As a strong upper bound, we evaluate a naive approach in which the entire conversation history is appended to the prompt before the query. This allows the model full access to all past turns, eliminating the need for memory management or retrieval.

RAG: Retrieval-Augmented Generation We implement RAG using a top-k retrieval pipeline over the full conversation history. Our RAG component represents semantic memory, as it stores the essential knowledge that grounds an agent's responses. At inference time, for each question, the model is provided with the top-k relevant utterances and their respective timestamps, ranked by cosine similarity to the query using bge-m3 embeddings. We initially test $k = 5, 10, 20$ to examine the impact of context window size on performance (Appendix A), and run all of our experiments with $k = 10$. The retrieved snippets are appended to the model prompt before answer generation.

PromptOpt: Procedural Memory Through Prompt Optimization Procedural memory encodes how an agent should behave and respond. We start with the same prompts used for the base RAG approach (Appendix B) that define the core agent behaviour and then evolve through feedback

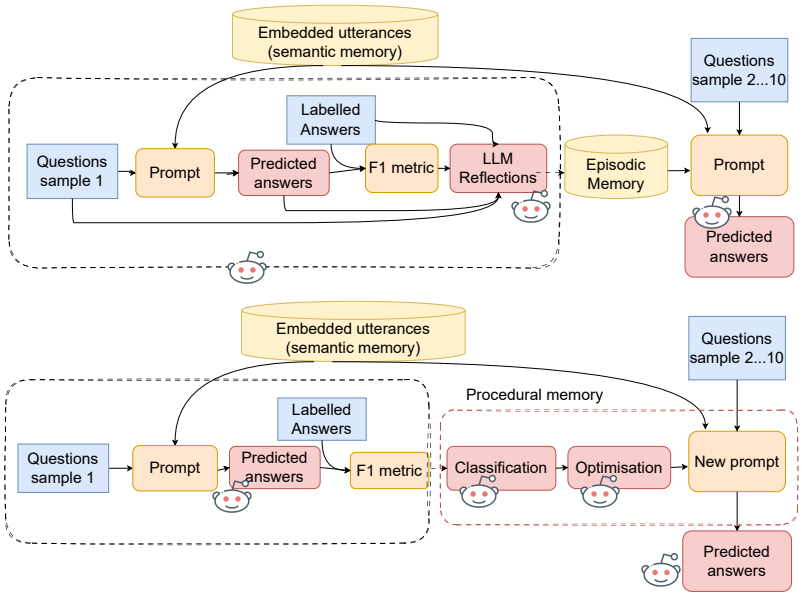

Figure 1: **Top figure: EpMem, episodic memory through in-context learning.** The model generates answers for a sample of questions and reflects on its performance using F1 scores and labelled answers to produce natural language reflections. These previous examples and reflections are stored in episodic memory and retrieved as in-context examples to guide the model's answers on future samples. **Bottom figure: PromptOpt, procedural memory through prompt optimization.** The model answers a sample of questions using an initial prompt and predictions are compared with labelled answers to compute F1 scores. A classification and optimization step is then used to update the prompt, forming procedural memory. The optimized prompt is applied to subsequent samples to guide future predictions.

and experience. In particular, after generating answers for all the questions related to the first conversation, we record the model's responses, the correct labels, and its performance, and iteratively prompt the model with batches of 5 examples to use to refine its instructions, aiming to learn which approaches work best for different situations (Figure 1). This optimization follows LangMem's implementation of procedural memory (LangChain, 2024) and it is achieved by a classification step, where the model is prompted to select which parts of its prompt caused errors, and an optimisation step where the model is prompted to generate an updated version of those prompt parts. We use the optimized prompt for the next sets of questions.

EpMem: Episodic Memory Through In-Context Learning Episodic memory preserves past interactions as learning examples that guide future behaviour. We implement it by first generating answers for all the questions related to the first conversation, noting the model's responses and the correct labels, and then prompting the model to reflect on each experience and generate a "reflection" string. We store dictionaries containing question, prediction, label, and reflection and we retrieve the top-3 most similar experiences to use as in-context examples whenever answering a new question (Figure 1). The structure and content of the prompts used is found in Appendix B.

A-Mem: Agentic Semantic Memory We replicate the A-Mem architecture as introduced by Xu et al. (2025), where the system maintains structured memory notes representing each utterance in the conversational data. As the memory agent is presented with a new utterance for the same conversation, it updates its memory with new entries and can make decisions about updating old ones and their connections. For QA, top-10 semantic memory entries are retrieved and inserted into the prompt. We follow the authors' implementation and prompting strategy, excluding task-specific rewording for adversarial and temporal questions.

Both RAG and A-Mem implement semantic memory, but with different levels of complexity. A-Mem supplements snippet embeddings with relevant keywords, contextual tags, and arrays of related memories. While RAG retrieves based purely on semantic similarity, A-Mem includes query ex-

pansion and provides the complete memory note context.

3.3 Implementation details

Appendix B details all the prompts we employed for our experiments. We deploy LLama 3.2-3B, Mistral-7B-v0.3, Qwen2.5-7B, and the respective instruction-tuned versions through huggingface, we access GPT-4o mini through the official OpenAI API. We report our main findings in tables 5 and3, and the full experimental results in Appendix C.

We conduct all of our experiments on a 24 GB GPU machine for a single run. For the retrieval components, we employ k=10 for snippet and A-Mem memories selection, for episodic memories we limit the number of examples to 3. We use bge-m3 text embeddings for our approaches and the all-minilm-l6-v2 model used by the authors for A-Mem. We report hyperparameter details in Appendix A.

The results on GPT-4o mini validate our findings on a widely-used commercial reasoning model. Our focus on open weights smaller models, instead, serves an important practical purpose, as these represent the scale most practitioners can deploy in resource-constrained environments.

4 Experiments and Results

In our empirical evaluation, we compare the effectiveness of various memory augmentation strategies across multiple language models. The task spans five reasoning categories, with different challenges and considerations for each of them: Single-Hop, Multi-Hop, Temporal, Open-Domain and Adversarial. We evaluate the Full Context baseline against different combinations of memory mechanisms: RAG, A-Mem, RAG+PomptOpt, RAG+EpMem, and RAG+PromptOpt+EpMem. We report average F1 across categories and the average number of tokens used per query as a proxy for efficiency. We calculate the average F1 ranking across categories, ordering the approaches by their F1 scores, from highest to lowest within each model (rank 1 to the highest F1 score, rank 2 to the next, and so on). As we use the first data sample to generate episodic and procedural memories, we report experimental results on the remaining 9 samples (1787 data points out of 1986). While in a real-world system these memories would be generated along all conversations, fixing a "training" sample and using the rest for evaluation allows us to capture a snapshot of model behaviour and learning.

4.1 Overall Performance

The results in Table 5 show significant variance in performance across different memory augmentation strategies. Retrieval and memory augmented generation approaches improve performance across most models, with instruction-tuned variants benefitting the most (Table 3). In most cases memory and retrieval-based approaches achieve competitive or superior F1 scores to the Full Context baseline, while using significantly fewer tokens.

RAG outperforms other approaches at Multi-Hop reasoning, suggesting that direct retrieval of similar utterances provides useful context for multi-step reasoning. Temporal-reasoning, adversarial and open-domain knowledge questions were the most challenging settings, with differences in performance based on the approach used. LLMs struggle to understand time concepts within dialogues, which is consistent with findings on single-turn-based benchmarks and temporal grounding (Qiu et al., 2023; Wang and Zhao, 2023). While this effect is somewhat less noticeable in GPT-4o mini, LLMs' performance for open-domain QA often degrades in memory-augmented settings. This suggests that, even when some of the open-domain knowledge may be parametric, introducing irrelevant context from memory and further instructing the model to use the provided context can lead to worse performance.

The adversarial questions, approached with the same prompt used for all categories rather than being rephrased as MCQs, show how the Full Context approach is prone to mistakes, while memory-augmented methods provide more trustworthy answers but can over-generate "no information available" answers and degrade performance in other categories for less powerful models (6.59 F1 for multi-hop vs. 95.24 F1 for adversarial for Qwen with RAG+Ep Mem). The way adversarial questions are designed in LoCoMo often references something that has high surface-level semantic similarity with part of the conversation the question relates to (Figure 2). For this reason, even in retrieval-augmented and memory-augmented settings the models will still have some confounding information in their context, making it particularly challenging to give the correct "No information available" answer, unless the model has strong instruction-following capabilities (i.e. results from

Model	Approach	Category					Average	
		Single-Hop	Multi-Hop	Temp.	Open Dom.	Adv.	F1 Rank.	Tokens
Llama 3B Inst.	Full Context	**30.07**	**25.96**	**11.39**	**39.36**	26.90	**1.00**	23265.98
	RAG	5.96	8.73	4.29	5.18	17.26	4.50	658.11
	A-Mem	3.55	4.31	5.12	4.04	50.56	4.00	2480.37
	RAG+PromptOpt	10.51	10.16	6.88	8.31	33.08	2.83	821.42
	RAG+EpMem	19.96	11.80	5.59	13.33	67.71	1.83	1305.82
	RAG+PromptOpt+EpMem	17.40	9.78	3.85	10.78	**72.18**	2.83	1377.53
Qwen 7B Inst.	Full Context	10.61	3.39	3.72	8.24	78.02	4.50	23376.87
	RAG	**21.72**	8.05	4.89	**14.51**	94.84	**2.00**	695.74
	A-Mem	12.73	**8.67**	**6.28**	13.86	90.53	**2.00**	3307.27
	RAG+PromptOpt	7.99	1.97	4.00	9.11	73.18	4.66	1024.88
	RAG+EpMem	17.51	6.59	5.35	10.21	**95.24**	**2.00**	1452.88
	RAG+PromptOpt+EpMem	17.64	6.35	5.33	9.96	94.48	2.66	1453.28
GPT-4o mini	Full Context	31.68	20.40	12.04	**56.40**	52.13	2.66	23132.49
	RAG	29.98	30.06	10.13	49.47	84.46	2.66	649.17
	A-Mem	24.83	26.02	7.61	37.48	64.79	3.83	3514.45
	RAG+PromptOpt	15.58	13.22	5.96	22.98	**92.98**	4.00	668.90
	RAG+EpMem	**31.77**	**40.39**	**12.51**	51.78	77.69	**1.66**	969.26
	RAG+PromptOpt+EpMem	30.79	41.44	5.86	51.71	80.53	2.66	972.93

Table 2: **Performance comparison of memory augmentation approaches across language models and reasoning categories.** We report average F1 scores, rankings and average number of tokens used per query for each approach across five reasoning categories: Single-Hop, Multi-Hop, Temporal, Open-Domain, and Adversarial. Best results per model in bold.

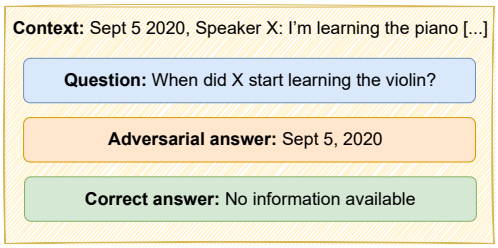

Figure 2: Example of adversarial question

Approach	Llama 3B		Qwen 7B	
	Base	Inst.	Base	Inst.
Full Context	2.00	1.00	2.33	4.50
RAG	2.16	4.50	1.16	2.00
A-Mem	3.33	4.00	3.16	2.00
RAG+PromptOpt	3.16	2.83	3.50	4.66
RAG+EpMem	3.50	1.83	3.83	2.00
RAG+PromptOpt +EpMem	3.33	2.83	3.50	2.66

Table 3: **Average F1 rankings comparison.** Base vs instruction-tuned models (lower is better). Across base models, Full Context and RAG have the top rankings; for instruction tuned models, the average ranking of other memory-augmented methods improves.

GPT-4o-mini).

4.2 Baseline Performance

The Full Context performance is consistent with findings from Maharana et al. (2024), showing that most long context LLMs can comprehend longer narratives but are prone to mistakes in Adversarial questions and can struggle with Multi-Hop queries when presented with extremely long contexts. This could be due to the "lost in the middle" effect, where models oversample from the beginning and end of their context window (Liu et al., 2023b).

4.3 RAG vs. A-Mem

Both performing retrieval over embedded and indexed utterances and using A-Mem can serve as the semantic memory or data source for long-context and knowledge-intensive QA. Importantly, RAG is much more efficient and scalable than A-Mem, not just in terms of tokens used at inference time, but also due to the nature of the information being stored. Each utterance is stored as a separate memory in A-Mem, and this requires two LLM calls per utterance to process the memory and generate all the required meta-data, in addition to embedding and indexing.

Due to the requirement for consistent structured output, A-Mem performs poorly on foundation models. RAG shows substantially better performance across all foundation models. This gap narrows for instruction-tuned models, which are able to take advantage of the full potential of A-Mem. While RAG still outperforms A-Mem in GPT-4o mini and all other instruction-tuned models, Llama 3B Instruct with A-Mem achieves an average F1 ranking of 4.0, compared to 4.5 for RAG.

4.4 Impact of Episodic Memory

The integration of Episodic Memory through ICL with RAG shows promising results. Rag+EpMem outperforms RAG on GPT-4o mini and Llama3.2 3B Instruct (respectively, 1.66 vs. 2.66 and 1.83 vs. 4.5 F1 ranking), while it achieves better results than all other approaches but slightly worse than RAG on Qwen2.5 7B Instruct. This suggests that incorporating episodic memories containing examples of model performance and reflections can provide valuable guidance for answering similar questions. Improvements can be seen in the adversarial and temporal reasoning categories for Llama and Qwen Instruct, and in multi-hop and temporal reasoning queries for GPT-4o mini. This suggests that the reflections on failures that required a specific type of reasoning were useful for the model to infer how to respond to new queries of the same category.

The review of the 199 episodic memories generated by GPT-4o-mini highlights different types of errors that the model identifies in its reflections. In 68 cases, the model correctly did not identify any error. In *Omission Errors*, 47% of mistakes, the model failed to include relevant context or information, but still gave a partially correct answer; for errors under *Misinterpretation of Temporal Context*, 27% of errors identified, the model did not correctly reason about dates; these correspond to almost all temporal questions in the sample. *Assumption and Overgeneralisation* errors, 11% of the total, include incorrect assumptions and broad conclusions drawn without evidence, usually about adversarial questions; in errors due to *Lack of Specificity*, the model did not provide detailed and precise information, but typically still had a partially correct answer; errors due to *Inclusion of Extraneous Information* included irrelevant details that might confuse the main point. We report an example of reflection on temporal reasoning in figure 3.

4.5 Impact of Procedural Memory

The addition of procedural memory to RAG and episodic memory shows negative results in the QA setting. This outcome can be interpreted as reflecting a mismatch between the nature of the task and our current implementation of procedural memory. Procedural memory is intended to encode strategies and behaviours and is likely to become more effective for tasks requiring planning steps, multi-step reasoning, or tool usage rather than QA over conversations. Additionally, different prompt optimisation techniques instead of the textual feedback inspired by LangMem might be better suited.

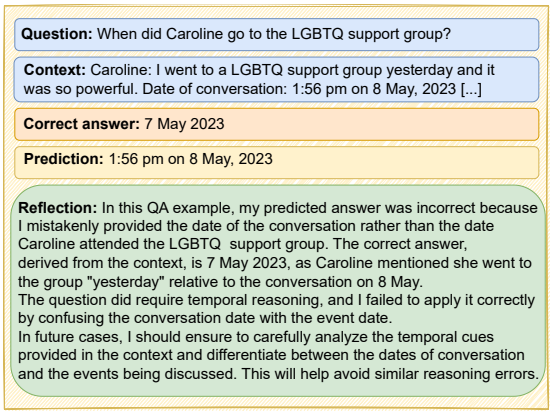

Figure 3: Example of episodic memory generated by GPT-4o mini

Our qualitative analysis of generated prompts reveals that newly generated prompts often focus too closely on specific examples instead of generalizing behavioural patterns and tend to repeat instructions (Appendix B.6.5). This over-specification affects instruction-following models, leading to decreased performance. This approach appears to suffer from overfitting to the training examples, creating prompts that are too narrow for the diverse reasoning required in our evaluation set. This suggests that the optimization procedure captured surface-level corrections (e.g., succeeding in adversarial questions) rather than abstracting broader strategies applicable across contexts.

4.6 Token Efficiency

A key advantage of memory and retrieval augmentation approaches lies in their substantial reduction in token usage. Full Context approaches consistently require between 23000 and 26000 tokens per query, RAG approaches with top-10 utterances and episodic or procedural memory use between 600 and 1500 tokens, A-Mem uses between 2300 and 3300 tokens per query – not including the ones needed to populate the memory. This represents a reduction of token consumption of more than 90% for RAG-based methods, with minimal or no degradation in overall performance. In particular, RAG+EpMem seems to offer the best balance of performance and efficiency for low-resource environments.

4.7 Model-Specific Observations

Different modes show varying responsiveness to memory augmentation. GPT-4o mini clearly shows the positive effect of RAG+EpMem, which leads to the best results on Single-Hop, Multi-Hop, and Temporal questions, with better results than the baseline of Adversarial questions. Additionally, these results confirm our conclusions that RAG+PromptOpt can lead to the over-generation of "No information avaliable" answers.

As table 3 shows, the instruction-tuned variant of Llama 3B performs the best with the full conversation being appended to the context, but it also shows dramatically improved performance with memory augmentation when compared to its base counterpart.

Qwen 7B models achieve the best performance on adversarial tasks, with significant improvements led by memory-augmented approaches. This might indicate that Qwen models have a strong inherent ability to handle adversarial examples and reply in a trustworthy way, but the lower performance in other categories shows that they tend to be "over-cautious" and generate "no information available" answers even for many queries from different categories, degrading overall performance and usability.

5 Conclusions and future work

This work provides an evaluation of memory augmentation strategies for long-context and knowledge-intensive tasks. We extend the prior research by comparing simple and complex implementations of semantic memory, and minimal implementations of episodic and procedural memory across multiple LLMs. By studying semantic, episodic, and procedural memory under a unified framework, we identify distinct trade-offs between performance, efficiency, and interpretability.

While previous research had approached the LoCoMo QA task by tailoring prompts to different reasoning categories (Maharana et al., 2024; Xu et al., 2025), we apply a consistent prompting strategy across all reasoning categories, making it possible to better isolate the effect of the memory mechanisms on performance.

Our findings confirm that while full-context prompting serves as a strong baseline, this approach suffers from inefficiency, poor scalability, limited interpretability, and vulnerability to context-length issues that can degrade performance. Most foundation and instruction-tuned models can benefit from simple RAG approaches, but instruction-tuned models can make use of their improved instruction-following and reasoning capabilities and also show strong performance with more complex approaches to semantic memory like A-Mem and episodic memory integration. In particular, GPT-4o mini shows the best results with RAG+EpMem, highlighting the potential of experiential learning with textual feedback.

For knowledge-intensive and long-context tasks, semantic memory is essential, but the best implementation should be determined based on use cases, scalability requirements, efficiency constraints, and data structure. Our findings show that, for less powerful models, complex approaches like A-Mem do not work at the best of their potential. Additionally, though more efficient than full-context prompting at inference time, A-Mem is more resource-intensive than RAG to construct, it remains effective for stable data that can benefit from clustering, but would become inefficient when memory requires frequent updates. Procedural memory, in this implementation, appears less suited to QA tasks but may prove valuable for planning or tool-use scenarios. In alignment with previous findings (Shinn et al., 2023; Zhao et al., 2024), episodic memory seems particularly valuable in enabling models to learn from both positive and negative experiences for complex QA tasks such as the adversarial one.

Taken together, our results show the impact of different types of memory-augmentation for knowledge-intensive tasks in smaller-scale LLMs. At the same time, they highlight directions for future research: validating results on real-world dialogue data, extending evaluation to multi-turn coherence, and developing selective, adaptive, metacognitively informed memory mechanisms. Future works should investigate agentic approaches to selecting experience examples, removing the need for fixed update schedules, and how the selection of positive or negative examples for episodic and procedural memory affects overall results. Comparing such non-parametric, memory-based adaptation with reinforcement learning methods may further illuminate how LLMs can learn not just from data, but from their own experiences.

Limitations

Our study has several limitations that open up directions for future research. Firstly, the lack of comparable benchmarks in this specific domain makes LoCoMo the most appropriate choice for this study, but its synthetic nature might not capture the noise and ambiguity of real-world conversations. We think our work establishes baseline comparisons that future research can build upon with additional datasets Additionally, we don't study multi-turn QA directly, but it's worth noting that the dataset includes several question categories that test similar skills: multi-hop questions require connecting information across multiple conversation turns; temporal questions require understanding the chronological flow of conversations; adversarial questions test whether systems can distinguish between information that exists versus doesn't exist in the conversation history.

Third, our experiments focus on small open-weight models ($\leq$7B parameters) and on an efficient commercial reasoning model (GPT-4o mini). This choice reflects realistic deployment constraints and enables reproducibility, and the results on GPT-4o mini validate our assumptions on the positive impact of episodic memory, the issues with procedural memory, and the validity of retrieval methods for efficiency purposes. Nonetheless, much larger models may exhibit different memory behaviours.

Finally, our implementations of episodic and procedural memory approaches are deliberately minimal, to isolate their effects, and we store memories only from the first sample out of 10 in the dataset. This artificial constraint limits both the benefits and complexities of a system where memories accumulate over time and throughout conversations. Our study did not address long-term memory management issues such as mechanisms for forgetting, updating, or consolidating memories and handling contradictions. These limitations suggest directions for future research to develop more robust, efficient, and reliable memory-augmented conversational systems.

Ethical Considerations

All external datasets and models used in this work (e.g., LoCoMo benchmark, A-Mem, open-weight LLMs) are publicly available on the web under research-compatible licenses. Our use of these artifacts is consistent with their intended purposes, as specified by their authors.

Acknowledgments

We used GitHub Copilot for code debugging support. All outputs were reviewed and edited by the authors.

References

Akari Asai, Zexuan Zhong, Danqi Chen, Pang Wei Koh, Luke Zettlemoyer, Hannaneh Hajishirzi, and Wen-tau Yih. 2024. Reliable, adaptable, and attributable language models with retrieval. *arXiv preprint arXiv:2403.03187*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv (Cornell University)*.

Kostas Hatalis, Despina Christou, Joshua Myers, Steven Jones, Keith Lambert, Adam Amos-Binks, Zohreh Dannenhauer, and Dustin Dannenhauer. 2023. Memory matters: The need to improve long-term memory in llm-agents. In *Proceedings of the AAAI Symposium Series*, volume 2, pages 277–280.

Samuel GB Johnson, Amir-Hossein Karimi, Yoshua Bengio, Nick Chater, Tobias Gerstenberg, Kate Larson, Sydney Levine, Melanie Mitchell, Iyad Rahwan, Bernhard Schölkopf, and 1 others. 2024. Imagining and building wise machines: The centrality of ai metacognition. *arXiv preprint arXiv:2411.02478*.

Savya Khosla, Zhen Zhu, and Yifei He. 2023. Survey on memory-augmented neural networks: Cognitive insights to ai applications. *arXiv preprint arXiv:2312.06141*.

LangChain. 2024. Langmem. https://langchain-ai.github.io/langmem/. Accessed: 2025-05-16.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Bang Liu, Xinfeng Li, Jiayi Zhang, Jinlin Wang, Tanjin He, Sirui Hong, Hongzhang Liu, Shaokun Zhang, Kaitao Song, Kunlun Zhu, and 1 others. 2025. Advances and challenges in foundation agents: From brain-inspired intelligence to evolutionary, collaborative, and safe systems. *arXiv preprint arXiv:2504.01990*.

Lei Liu, Xiaoyan Yang, Yue Shen, Binbin Hu, Zhiqiang Zhang, Jinjie Gu, and Guannan Zhang. 2023a. Think-in-memory: Recalling and post-thinking enable llms with long-term memory. *arXiv preprint arXiv:2311.08719*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023b. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. 2024. Evaluating very long-term conversational memory of llm agents. *arXiv preprint arXiv:2402.17753*.

Ali Modarressi, Ayyoob Imani, Mohsen Fayyaz, and Hinrich Schütze. 2023. Ret-llm: Towards a general read-write memory for large language models. *arXiv preprint arXiv:2305.14322*.

Yifu Qiu, Zheng Zhao, Yftah Ziser, Anna Korhonen, Edoardo M Ponti, and Shay B Cohen. 2023. Are large language models temporally grounded? *arXiv preprint arXiv:2311.08398*.

Aymeric Roucher, Albert Villanova del Moral, Thomas Wolf, Leandro von Werra, and Erik Kaunismäki. 2025. 'smolagents': a smol library to build great agentic systems. https://github.com/huggingface/smolagents.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. *Preprint*, arXiv:2305.15294.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652.

Yuqing Wang and Yun Zhao. 2023. Tram: Benchmarking temporal reasoning for large language models. *arXiv preprint arXiv:2310.00835*.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. 2025. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*.

Zeyu Zhang, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Quanyu Dai, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. 2024. A survey on the memory mechanism of large language model based agents. *arXiv preprint arXiv:2404.13501*.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. 2024. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19632–19642.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19724–19731.

A Hyperparameter Settings

For all answer-generation LLM calls and A-Mem we set the temperature to 0.5, to balance model creativity and accuracy. An higher temperature of 0.7 is used for reflection and optimized prompt generation steps, while lower (0.4) temperature is used for the prompt classification step, as the model should stick to generating a list of prompt names from the given one.

For RAG we decided to retrieve the top-10 most relevant utterances to both provide the model with enough relevant information and keep the context short, without introducing noise through the retrieval of less relevant utterances. Table 4 shows the results with k set to 5, 10, and 20 for Llama 3.2 3B. K also aligns with the number of memory notes being retrieved by A-Mem.

Top-k	Category					Average	
	S-Hop	M-Hop	Temp.	Open	Adv.	F1 Rank	Tokens
5	13.45	**11.13**	7.36	13.03	**4.48**	**1.6**	384.90
10	11.99	10.64	7.28	**13.85**	2.91	2.2	658.11
20	**14.74**	8.64	**7.53**	12.78	2.47	2.2	1196.08

Table 4: **Performance comparison of RAG-Utterances with different** $top - k$. F1 scores, rankings and tokens per query across five categories: Single-Hop (S-Hop), Multi-Hop (M-Hop), Temporal (Temp.), Open-Domain (Open), and Adversarial (Adv.). Results from Llama 3.2 3B, best in bold.

B Prompts Used

B.1 Full Context and LoCoMo Summaries

```
Based on the given conversations, write a short
↪  answer for the following question in a few
↪  words. Do not write complete and lengthy
↪  sentences. Answer with exact words from the
↪  conversations whenever possible.

Below is a conversation between two people: {name
↪  of speaker 1} and {name of speaker 2}. The
↪  conversation takes place over multiple days
↪  and the date of each conversation is written
↪  at the beginning of the conversation.

{full conversation transcript or session
↪  summaries}

If the answer to the question requires you to do
↪  temporal reasoning, use DATE of CONVERSATION
↪  to answer with an approximate date. If the
↪  question cannot be answered, write 'No
↪  information available.
Question: {question} Answer:
```

B.2 RAG

```
Based on the given conversations, write a short
↪ answer for the following question in a few
↪ words. Do not write complete and lengthy
↪ sentences. Answer with exact words from the
↪ conversations whenever possible.

Below are retrieved snippets from a conversation
↪ between two people: {name of speaker 1} and
↪ {name of speaker 2}.

{top-10 retrieved conversations snippets in the
↪ format
"speaker: utterance, date of conversation: day
↪ month year"}

If the answer to the question requires you to do
↪ temporal reasoning, use DATE of CONVERSATION
↪ to answer with an approximate date. If the
↪ question cannot be answered, write 'No
↪ information available.
Question: {question} Answer:
```

B.3 A-Mem

```
Based on the context: {top-10 retrieved memories
↪ containing talk start time, speaker and
↪ utterance, memory context, memory keywords,
↪ memory tags}, write an answer in the form of
↪ a short phrase for the following question.
↪ Answer with exact words from the context
↪ whenever possible. If the answer to the
↪ question requires you to do temporal
↪ reasoning, use DATE of CONVERSATION to answer
↪ with an approximate date. If the question
↪ cannot be answered, write 'No information
↪ available'.

Question: {question} Short answer:
```

B.4 EpMem: Episodic Memory

B.4.1 Reflection Step

```
Reflect on your performance in the following QA
↪ example. Focus specifically on:
- Whether your answer was correct or not, and
↪ why.
- If the question required temporal reasoning,
↪ how well you handled it.
- If the question had no answer in the context,
↪ whether you correctly identified that.
- What reasoning errors (if any) occurred, and
↪ how to avoid them in future similar cases.
Provide a short reflection in a few sentences.

Question: {question}
Context: {top-10 retrieved utterances}
Correct Answer: {label}
Predicted Answer: {prediction}
Reflection:
```

B.4.2 QA Step

```
You are an AI assistant that answers questions
↪ based on a given conversation. Use the
↪ current context and optionally refer to past
↪ examples and reflections to help you reason,
↪ but focus on the current question.
Based on the given conversation snippets, write a
↪ short answer for the following question in a
↪ few words. Do not write complete and lengthy
↪ sentences. Answer with exact words from the
↪ conversations whenever possible. Use the
↪ following examples to answer correctly.

Examples: {top-3 most similar episodic memories
↪ with reflections}

Below are retrieved snippets from a conversation
↪ between two people.
{top-10 retrieved utterances}

Question: {query}

Now, answer the current question using the
↪ conversation context above. Refer to the past
↪ examples and reflections only if they help
↪ clarify your reasoning. If the answer to the
↪ question requires you to do temporal
↪ reasoning, use DATE of CONVERSATION to answer
↪ with an approximate date. If the question
↪ cannot be answered, write 'No information
↪ available'. Answer:
```

B.5 Examples of Reflection

```
"question": "Who supports Caroline when she has a
↪ negative experience?",
"context": {top-10 retrieved utterances},
"answer": "Her mentors, family, and friends"
"prediction": " Melanie, friends, others. Note:
↪ The answer should be a few words as
↪ requested.",
"f1_score": 0.333,
"reflection": " I incorrectly identified Melanie
↪ as the primary support person for Caroline.
↪ While Melanie does offer support, the context
↪ of the conversations reveals that Caroline's
↪ support system is diverse and includes her
↪ mentors, family, and friends. I struggled
↪ with temporal reasoning in this case, as the
↪ question requires me to analyze the
↪ conversations across different dates to
↪ identify patterns and relationships. In
↪ future, I will be more careful to consider
↪ the broader context and not rely solely on
↪ individual conversations. Also, I should have
↪ noticed that the question was asking about a
↪ specific scenario (negative experience)
↪ rather than general support. This would have
↪ led me to focus on conversations that mention
↪ Caroline's negative experiences, such as her
↪ tough breakup and the challenges she faced."

"question": "Would Caroline likely have Dr. Seuss
↪ books on her bookshelf?"
"context": "Caroline: I've got lots of kids'
↪ books - classics, stories from different
↪ cultures, educational books, all of that.
↪ What's a favorite book you remember from your
↪ childhood?"...{other retrieved snippets}
"answer": "Yes, since she collects classic
↪ children's books",
```

```
"prediction": "No information available.",
"f1_score": 0,
"reflection": "In this case, I didn't correctly
↪ identify that Caroline's love of reading and
↪ collecting classic children's books would
↪ imply that she would likely have Dr. Seuss
↪ books on her bookshelf. I should have used
↪ the context of their conversation to infer
↪ this, rather than relying on my initial
↪ instinct. I will make sure to pay closer
↪ attention to the conversation flow and use
↪ the context to guide my answers in the
↪ future."
```

B.6 PromptOpt: Procedural Memory

B.6.1 Original Prompts to Optimize

```
{
    "name": "task",
    "prompt": "Based on the above conversations,
    ↪ write a short answer for the following
    ↪ question in a few words. Do not write
    ↪ complete and lengthy sentences. Answer
    ↪ with exact words from the conversations
    ↪ whenever possible.",
},
{
    "name": "intro",
    "prompt": "Below are retrieved snippets from
    ↪ a conversation between two people. \n",
},
{
    "name": "rules",
    "prompt": "If the answer to the question
    ↪ requires you to do temporal reasoning,
    ↪ use DATE of CONVERSATION to answer with
    ↪ an approximate date. If the question
    ↪ cannot be answered, write 'No
    ↪ information available'.",
}
```

B.6.2 Classification Step

```
You always return JSON output. Analyze the
↪ following trajectories and decide which
↪ prompts ought to be updated to improve the
↪ performance on future trajectories:

{batch of 5 LLM trajectories, containing query,
↪ predicted answer, correct answer, and F1}

Below are the prompts being optimized:
↪ {dictionary containing name of prompt: prompt
↪ content}

Return one JSON dictionary in the format
↪ {"which": [...]}, listing the names of
↪ prompts that need updates. The names must be
↪ in {prompt_names}. Do not return any
↪ explanations or reasoning.
```

B.6.3 Optimization Step

```
You are helping an AI assistant learn by
↪ optimizing its prompt. You always return JSON
↪ output.

## Background

Below is the current prompt: {prompt}

## Instructions

The developer provided these instructions
↪ regarding when/how to update:

<update_instructions>Do not make the prompts
↪ specific about any particular people or
↪ events mentioned in any question or
↪ conversation.<update_instructions>

## Session Data
Analyze the session(s) (and any user feedback)
↪ below:

<trajectories>{batch of 5 LLM trajectories,
↪ containing query, predicted answer, correct
↪ answer, and F1}<trajectories>

## Instructions

1. Reflect on the agent's performance on the
↪ given session(s) and identify any real
↪ failure modes (e.g., style mismatch, unclear
↪ or incomplete instructions, flawed reasoning,
↪ etc.).
2. Recommend the minimal changes necessary to
↪ address any real failures. If the prompt
↪ performs perfectly, simply respond rewriting
↪ the original prompt without making any
↪ changes.
3. DO NOT use any tags like <current_prompt>,
↪ <current_prompt> or <trajectories> in your
↪ response.
4. Be brief and concise. Avoid unnecessary
↪ verbosity.
IFF changes are warranted, focus on actionable
↪ edits. Be concrete. Edits should be
↪ appropriate for the identified failure modes.
↪ For example, consider clarifying the style or
↪ decision boundaries, or adding or modifying
↪ explicit instructions for conditionals,
↪ rules, or logic fixes; or provide
↪ step-by-step reasoning guidelines for
↪ multi-step logic problems if the model is
↪ failing to reason appropriately.
ONLY return JSON in the following format:
↪ {"reasoning": "<reasoning>",
↪ "updated_prompt": "<updated_prompt>"}.
```

B.6.4 QA Step

```
{(optimized) task prompt}

{(optimized) intro prompt}
{top-10 retrieved utterances}

{(optimized) rules prompt}
Question: {query} Answer:
```

B.6.5 Example of Optimized Prompts

The following prompt is the result obtained after optimization with Qwen2.5 7B Instruct:

```
Based on the above conversations, write a short
↪  answer for the following question in a few
↪  words. Use exact words from the conversations
↪  whenever possible. If no exact words are
↪  available, provide a concise summary using
↪  only the information from the conversations,
↪  but avoid adding extra details or
↪  interpretations. Always confirm with 'No
↪  information available' if the required
↪  information is not present in the
↪  conversations. Only use 'No information
↪  available' when no relevant information is
↪  present.

Below are retrieved snippets from a conversation
↪  between two people.
{top-10 retrieved utterances}

Provide direct answers based on the given
↪  information only. If the answer can be
↪  extracted verbatim, do so. If no information
↪  is available, write 'No information
↪  available'. Avoid adding extra context,
↪  personal speculations, or any assumptions
↪  and ensure the answer is accurate. Always
↪  refer back to the provided snippets for
↪  answers. Only provide 'No information
↪  available' unless explicitly asked to infer
↪  or provide additional context.
Question: {query} Answer:
```

C Full Results Table

Model	Approach	Category					Average	
		Single-Hop	Multi-Hop	Temp.	Open Dom.	Adv.	F1 Rank.	Tokens
Llama 3B Inst.	Full Context	**30.07**	**25.96**	**11.39**	**39.36**	26.90	**1.00**	23265.98
	RAG	5.96	8.73	4.29	5.18	17.26	4.50	658.11
	A-Mem	3.55	4.31	5.12	4.04	50.56	4.00	2480.37
	RAG+PromptOpt	10.51	10.16	6.88	8.31	33.08	2.83	821.42
	RAG+EpMem	19.96	11.80	5.59	13.33	67.71	1.83	1305.82
	RAG+PromptOpt+EpMem	17.40	9.78	3.85	10.78	**72.18**	2.83	1377.53
Mistral 7B Inst.	Full Context	17.90	15.01	11.03	30.00	39.46	2.83	26585.00
	RAG	**28.26**	**33.96**	**12.19**	**38.05**	32.73	**1.50**	752.86
	A-Mem	11.11	11.88	7.97	20.80	**51.93**	3.00	2932.01
	RAG+PromptOpt	14.01	5.81	7.69	13.50	19.79	4.83	933.66
	RAG+EpMem	23.81	20.13	7.82	22.48	46.62	2.66	1364.25
	RAG+PromptOpt+EpMem	22.94	21.52	11.55	23.72	44.84	2.16	1261.84
Qwen 7B Inst.	Full Context	10.61	3.39	3.72	8.24	78.02	4.50	23376.87
	RAG	**21.72**	8.05	4.89	**14.51**	94.84	**2.00**	695.74
	A-Mem	12.73	**8.67**	**6.28**	13.86	90.53	**2.00**	3307.27
	RAG+PromptOpt	7.99	1.97	4.00	9.11	73.18	4.66	1024.88
	RAG+EpMem	17.51	6.59	5.35	10.21	**95.24**	**2.00**	1452.88
	RAG+PromptOpt+EpMem	17.64	6.35	5.33	9.96	94.48	2.66	1453.28
GPT-4o mini	Full Context	31.68	20.40	12.04	**56.40**	52.13	2.66	23132.49
	RAG	29.98	30.06	10.13	49.47	84.46	2.66	649.17
	A-Mem	24.83	26.02	7.61	37.48	64.79	3.83	3514.45
	RAG+PromptOpt	15.58	13.22	5.96	22.98	**92.98**	4.00	668.90
	RAG+EpMem	**31.77**	**40.39**	**12.51**	51.78	77.69	**1.66**	969.26
	RAG+PromptOpt+EpMem	30.79	41.44	5.86	51.71	80.53	2.66	972.93
Llama 3B	Full Context	**14.72**	9.78	7.57	**14.74**	0.67	**2.00**	23265.98
	RAG	11.99	**10.64**	7.28	13.85	2.91	2.16	658.11
	A-Mem	5.49	4.11	**8.44**	6.51	**11.36**	3.33	2309.74
	RAG+PromptOpt	11.47	5.83	6.53	9.74	6.26	3.16	657.42
	RAG+EpMem	9.15	7.21	4.57	7.57	10.27	3.50	1243.96
	RAG+PromptOpt+EpMem	9.72	7.77	4.44	7.64	8.74	3.33	1232.17
Mistral 7B	Full Context	**10.74**	5.61	5.89	10.92	2.69	2.16	26585
	RAG	9.15	**7.75**	**6.13**	**11.83**	6.28	**1.50**	752.86
	A-Mem	1.17	1.3	0.80	2.93	0.08	6.00	2224.66
	RAG+PromptOpt	8.67	5.35	5.98	8.72	7.39	2.33	1019.7
	RAG+EpMem	6.17	3.17	3.94	5.04	**16.04**	2.83	1382.18
	RAG+PromptOpt+EpMem	5.06	2.86	2.39	4.44	8.02	3.66	1485.13
Qwen 7B	Full Context	16.96	12.32	8.17	28.43	5.17	2.33	23376.87
	RAG	**21.50**	**23.14**	**9.48**	**31.32**	44.84	**1.16**	695.74
	A-Mem	10.93	9.30	5.90	14.33	**66.14**	3.16	2460.99
	RAG+PromptOpt	12.30	8.29	7.36	12.98	7.02	3.50	886.8
	RAG+EpMem	15.85	7.32	5.93	12.72	44.36	3.83	1304.66
	RAG+PromptOpt+EpMem	15.46	7.42	6.03	12.53	47.61	3.50	1402.7

Table 5: **Performance comparison of memory augmentation approaches across various language models and reasoning categories.** We report average F1 scores, rankings and average number of tokens used per query.