Agentic-Like Memory with Category-Aware Prompts and LLM Reranking for Long-Term Conversational QA

Ningsen Wang
Computer Science and Technology
Fudan University
22307130058@m.fudan.edu.cn

Scale	F1 Score	BLEU-1	LLM Score
First two	41.50	36.63	56.22
All	41.39	36.42	57.73

Table 1: Metrics for first two conversations and all ten.

Abstract

Long-lived conversational agents must decide what to remember from months of dialogue while operating under tight context limits. Existing systems often rely on complex online memory controllers that interleave retrieval and generation, which makes them hard to reproduce and to attribute performance gains. We revisit this problem from a simpler angle and ask how far one can go by simulating an agentic memory layer entirely offline. We flatten LoCoMo conversations, slide fixed-size windows to form episodic memory chunks, encode them with an embedding model, and index them in FAISS. At query time, we perform question-type aware dense retrieval followed by LLM-based reranking and condition a QA model on a small set of selected windows under strict answer-format prompts. On LoCoMo10 our pipeline matches or surpasses several deployed memory systems on 3 out of 4 question categories. Our results suggest that a carefully engineered agentic-like pipeline with strong embeddings, simple windowing, and type-aware prompts already captures a large fraction of the benefits typically attributed to fully agentic memory controllers. Future work can based on this pipeline and explore how to further cut token scale. Code can be found here: https://github.com/Wangningsen/FDU-2025NLP-PJ.

1 Introduction

Large language models are increasingly deployed as long-lived conversational agents that accompany a user across months of interactions rather than a single session. In these settings, simple reuse of the current context window is not enough: the agent must selectively retain, organize, and exploit salient information from earlier conversations, while avoiding hallucinations and privacy leaks. Recent benchmarks such as LoCoMo(Maharana et al., 2024), a long conversational memory suite and adopted by multiple memory frameworks, highlight that even strong LLMs suffer substantial degradation when asked to recall or reason over events that lie far outside their immediate context.

To address these shortcomings, the community has begun to build explicit “memory layers” on top of foundation models. Systems such as mem0(Chhikara et al., 2025), Zep(Rasmussen et al., 2025), and related frameworks(Packer et al., 2023; Sarthi et al., 2024; Xu et al., 2025) treat memory as a first-class component. In parallel, Hu et al. (2025) have started to map this space, distinguishing short-term in-context memory from longer-term episodic and semantic stores, and contrasting retrieval-augmented generation (RAG) architectures with more agentic memory controllers. These trends suggest that long-term memory is becoming a core design axis for agent systems, not merely an implementation detail.

In this work, instead of streaming the conversations through an online agent that decides when to write memories, we simulate a memory layer offline with conversatinal "windows" that act as episodic memory units. These windows are embedded with a strong multilingual encoder, indexed with FAISS, and later retrieved and reranked for each question before a QA model produces the final answer. With this simulated memory-writing agent loop, this pipeline achieves competitive or superior performance to several dedicated memory systems on LoCoMo10, especially on multi-hop and temporal questions.

Our contributions are threefold. First, we present a LoCoMo-tailored agentic-like retrieval pipeline that combines conversation-aware windowing, dense retrieval, and LLM-based reranking with a type-aware prompting scheme that explicitly accounts for temporal and multi-hop structure. Second, we instantiate this pipeline with open-weight embedding models and an off-the-shelf QA LLM, and we provide a transparent description of all components, making the system easy to reproduce or adapt. Third, we evaluate the resulting system on LoCoMo10, comparing against a range of memory-enabled baselines and analyzing where more sophisticated agentic memory mechanisms may still be needed.

2 Methodology

2.1 Overview

Figure 1 shows our pipeline. The full pipeline can be summarized as a four-stage process:

Preprocessing and chunking. We flatten LoCoMo sessions into a single turn sequence per conversation, inject timestamp meta-turns, annotate turns that contain explicit or relative temporal expressions, and slide fixed-size windows (This is to simulate LLM context window. When we get enough text for a context window, we push it to memory store) over the resulting sequence to form memory chunks.

Embedding and indexing. For each chunk, we encode it with Qwen3-Embedding-8B(Zhang et al., 2025), normalize the vectors, and build a FAISS inner-product index, storing metadata such as *conv_id* and *turn spans*.

Question typing and retrieval. For each LoCoMo question, we map the official category to an internal question type, perform vector search over the index, and rerank candidate chunks with an LLM to assemble a small set of highly relevant memories.

Answer Generation. We call Claude Sonnet 4(Anthropic, 2025) on the question plus selected memories and enforce strict answer format constraints.

2.2 Preprocessing and chunking

LoCoMo provides conversations as per-session turn lists together with optional timestamps and annotated QA pairs. Rather than treating the entire dialogue as a static document, we view it as a long running interaction between two speakers that is gradually distilled into an external memory store. Concretely, we linearize each LoCoMo conversation into a sequence of turns and apply a sliding window over turns. Each window can be interpreted as a short multi turn episode that would fit into the model's native context. After this episode has "fallen out" of the immediate context, we commit it to long term memory by encoding the window into a single Qwen3 embedding and storing it in a Faiss index along with its conversation and turn span metadata.

In other words, our offline chunking and indexing procedure acts as an idealized simulation of an online agent that periodically compresses past context into compact memory entries while keeping only the most recent turns in the model's working context. At question answering time, the agent no longer has access to the raw dialogue history and must instead retrieve and integrate these stored memory windows to answer benchmark queries.

We first normalize each example into a single flattened turn sequence. For a given sample, we iterate over *session_1*, *session_2* ...,in temporal order and append their turns to a unified conversation. When a session timestamp is available, we insert a synthetic meta-turn describing the date and time, so that temporal reasoning can later resolve expressions such as "yesterday" or "last week" against an explicit anchor.

To support time-sensitive retrieval, we annotate each turn with a binary temporal flag, which is achieved by a regular-expression matcher scans for calendar dates and relative expressions, and any matched turn is marked as *has_datetime*. This flag is propagated to the chunk level and used downstream to bias retrieval for temporal questions. The flattened conversation is then segmented into overlapping "memory chunks" using a sliding-window scheme. We slide a window 64 turns with a stride of 32 turns. Each chunk stores the conversation identifier, a chunk identifier derived from its turn span, the inclusive-exclusive indices of its start and end turns, an aggregate temporal flag indicating whether contained turn mentions time, and the concatenated text of all turns in the window.

2.3 Embeddings and indexing

To enable efficient candidate selection from hundreds of chunks, we adopt a dense retrieval strategy. Each chunk is encoded with Qwen3-Embedding-8B(Zhang et al., 2025). Text is tokenized up to a fixed maximum length (8,192 tokens), passed

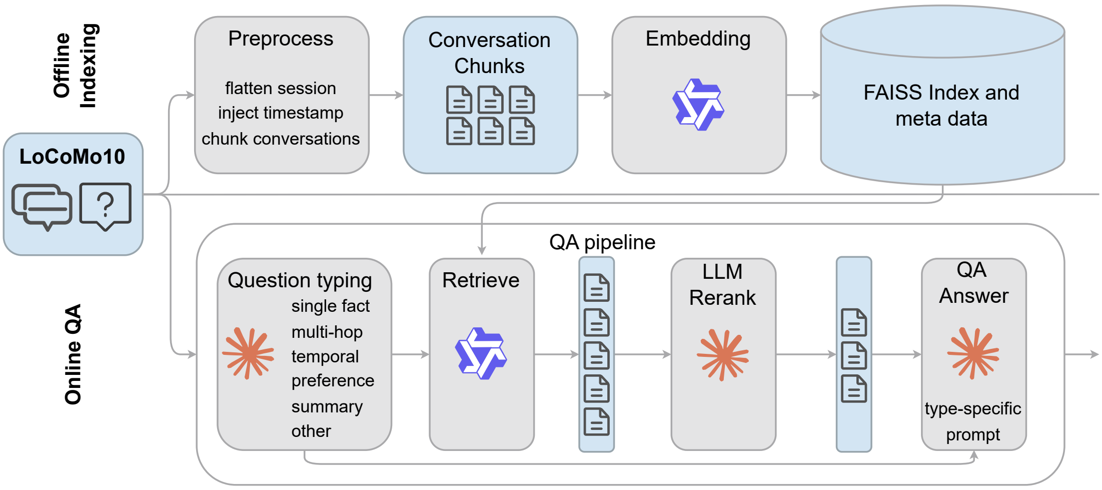

Figure 1: The whole pipeline.

through the transformer, and the final hidden states are mean-pooled to obtain a single vector representation. Embeddings are cast to 32-bit floating point and L2-normalized so that inner products correspond to cosine similarity.

All chunk embeddings for the corpus are then inserted into a FAISS inner-product index. We maintain a parallel metadata table that records the conversation ID, chunk ID, and turn boundaries for each vector. The resulting artefacts are a compact FAISS index, a NumPy array of all embeddings, and a JSON metadata file. At inference time, the retriever can either query FAISS directly or, when convenient, operate on the precomputed embedding matrix with in-memory similarity search, but in both cases the underlying representation and similarity function remain identical.

2.4 Question typing and retrieving

At query time the agent receives only the question and the external memory store. It first issues an embedding-based retrieval over all archived windows for the corresponding conversation, then uses a large language model to rerank and compress the retrieved snippets into a small working set, from which it generates the final answer. This mimics an agent that cannot replay the full dialogue, but has to reason over the memories it previously wrote.

LoCoMo annotates each question with a coarse category describing the required reasoning pattern. We map these numeric labels according to LoCoMo setting. The category label is essential for later LLM QA prompt. For further real application that there is no category annotation for each question, we also design a light-weight classifier that combines keyword rules and an auxiliary LLM call.

Retrieval proceeds in two stages. First, we embed the question with the same Qwen3 encoder used for chunks, normalize the vector, and query the FAISS index for the nearest neighbours in the corresponding conversation. The maximum number of raw candidates is conditioned on the question type: temporal questions draw from a larger pool (10) to ensure coverage of time-marked segments, whereas single-hop or open-domain questions use a smaller budget (8). Temporal questions further prioritize chunks whose *has_datetime* flag is set.

In the second stage, we refine the candidate set with an Claude Sonnet 4 as reranker. For each candidate we construct a short snippet by truncating the chunk text to a fixed character budget, and we present the snippets in indexed form to the reranker. The reranking prompt asks the model to specify the indices of the most relevant chunks for answering the question. When parsing succeeds, we retain at most a handful of chunks. Specifically, at most **2** for general questions and **4** for temporal questions, are treated as the final context.

2.5 Answer Generation

The final stage conditions a QA LLM on the question and the selected memory chunks. We call Claude Sonnet 4 for answer generation. Crucially, we impose category-specific prompt and strict output-format constraints to make automatic scoring meaningful. **For version 1 try, we did not include strict format constraint and get very high llm judge score, but extremely low BLEU-1**

Method	Single Hop			Multi-Hop			Open Domain			Temporal		
	F1 ↑	BLEU-1 ↑	LLM ↑	F1 ↑	BLEU-1 ↑	LLM ↑	F1 ↑	BLEU-1 ↑	LLM ↑	F1 ↑	BLEU-1 ↑	LLM ↑
LoCoMo	25.02	19.75	–	12.04	11.16	–	40.36	29.05	–	18.41	14.77	–
ReadAgent	9.15	6.48	–	5.31	5.12	–	9.67	7.66	–	12.60	8.87	–
MemoryBank	5.00	4.77	–	5.56	5.94	–	6.61	5.16	–	9.68	6.99	–
MemGPT	26.65	17.72	–	9.15	7.44	–	41.04	34.34	–	25.52	19.44	–
A-Mem	27.02	20.09	–	12.14	12.00	–	44.65	37.06	–	45.85	36.67	–
A-Mem*	20.76	14.90	39.79 ± 0.38	9.22	8.81	18.85 ± 0.31	33.34	27.58	54.05 ± 0.22	35.40	31.08	49.91 ± 0.31
LangMem	35.51	26.86	62.23 ± 0.75	26.04	22.32	47.92 ± 0.47	40.91	33.63	71.12 ± 0.20	30.75	25.84	23.43 ± 0.39
Zep	35.74	23.30	61.70 ± 0.32	19.37	14.82	41.35 ± 0.48	**49.56**	38.92	**76.60 ± 0.13**	42.00	34.53	49.31 ± 0.50
OpenAI	34.30	23.72	63.79 ± 0.46	20.09	15.42	42.92 ± 0.63	39.31	31.16	62.29 ± 0.12	14.04	11.25	21.71 ± 0.20
mem0	38.72	27.13	**67.13 ± 0.65**	28.64	21.58	**51.15 ± 0.31**	47.65	38.72	72.93 ± 0.11	48.93	40.51	55.51 ± 0.34
memg	38.09	26.03	65.71 ± 0.45	24.32	18.82	47.19 ± 0.67	49.27	**40.30**	75.71 ± 0.21	**51.55**	40.28	58.13 ± 0.44
Ours	**40.05**	**31.78**	61.35	**36.84**	**30.84**	46.73	20.04	17.70	37.50	46.02	**42.25**	**63.02**

Table 2: Performance comparison of memory-enabled systems across different question types in the LOCOMO dataset. Evaluation metrics include F1 score, BLEU-1 , and LLM-as-a-Judge score, with higher values indicating better performance. **Bold** denotes the best performance for each metric across all methods. (↑) represents higher score is better.

and F1 score. The model is then instructed that its entire reply must be a single short answer string without any reasoning. The detail prompts are listed in appendix A.2.

3 Experiments

3.1 Experiment Setup

We evaluate our system on the LoCoMo10(Maharana et al., 2024). We skip all category 5 questions to align with mem0(Chhikara et al., 2025) evaluation recipe. We choose Qwen3-Embedding-8B(Zhang et al., 2025) as embedding model and Claude Sonnet 4(Anthropic, 2025) for answer generation. We run Qwen-3-Embedding-8B with a A100-80G GPU in one local A100 node, and call Claude Sonnet 4 via AWS Bedrock. Claude is restricted to generate at most 64 tokens with temperature 0.0. For evaluation, we follow mem0 style. Specifically, we call Qwen-plus from dashscope for LLM judge score. We deliberately use a Qwen-based judge rather than a Claude-family model, since prior work on LLM-as-a-judge has reported **self-preference bias: models tend to assign higher scores to outputs that match their own generation and stylistic distribution**(Wataoka et al., 2024; Panickssery et al., 2024). Using a different model family for evaluation helps mitigate this bias and yields more robust scores across systems. We report BLEU-1, F1 score and LLM judge score.

3.2 Result

Table 1 reports the average performance of our pipeline on the first two conversations and all conversations in LoCoMo10, while Table 2 compares our system against previously reported memory-enabled baselines. We follow the official evaluation and report F1 score, BLEU-1, and an LLM-as-a-judge score for each question type.

Overall, our static pipeline achieves strong performance on factoid-style questions. For **single-hop** questions, our method attains the best F1 (40.05) and BLEU-1 (31.78). On **multi-hop** questions, the gains are more pronounced. Our approach reaches 36.84 F1 and 30.84 BLEU-1, substantially outperforming the best baseline (mem0, 28.64 / 21.58), indicating that the combination of Qwen3-based dense retrieval and LLM-based reranking is particularly effective. For **temporal** questions, our system achieves the best BLEU-1 (42.25) and the highest LLM-judge score (63.02), while remaining close to mem0 (48.93 F1) in terms of F1. This aligns with our design choice of temporal-aware retrieval and explicit prompting to normalize relative time expressions into calendar references. In contrast, our method underperforms on **open-domain** questions, with 20.04 F1, 17.70 BLEU-1, and a 37.50 LLM-judge score. This is largely expected: our QA prompt is explicitly conservative and encourages the model to answer *"I am not sure"* when the memories do not contain a clear supporting span, whereas many agentic systems are optimized to provide longer, more speculative responses that align more closely with the short reference answers under F1, BLEU-1, and LLM-based grading. In other words, our pipeline trades open-domain coverage for stricter faithfulness to the retrieved memory chunks.

Across both the first-two-conversation and full-conversation settings (Table 1), we observe consistent trends: the proposed pipeline is particularly effective on single-hop, multi-hop, and temporal questions, while remaining deliberately conservative on open-domain queries.

However, we have to acknowledge that this performance gain is very possible for stronger LLM (Claude Sonnet 4 vs ChatGPT 4o mini). Also, even without accurate token counting, I am afraid we have used up more tokens than mem0 method because we directly attach corresponding memory chunk for answer generation. This is a key limitation for our method.

References

Anthropic. 2025. Model card and evaluations for claude models. Technical report, Anthropic.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. Mem0: Building production-ready ai agents with scalable long-term memory. *Preprint*, arXiv:2504.19413.

Yuyang Hu, Shichun Liu, Yanwei Yue, Guibin Zhang, Boyang Liu, Fangyi Zhu, Jiahang Lin, Honglin Guo, Shihan Dou, Zhiheng Xi, Senjie Jin, Jiejun Tan, Yanbin Yin, Jiongnan Liu, Zeyu Zhang, Zhongxiang Sun, Yutao Zhu, Hao Sun, Boci Peng, and 28 others. 2025. Memory in the age of ai agents. *Preprint*, arXiv:2512.13564.

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*.

Charles Packer, Vivian Fang, Shishir_G Patil, Kevin Lin, Sarah Wooders, and Joseph_E Gonzalez. 2023. Memgpt: Towards llms as operating systems.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. 2024. LLM evaluators recognize and favor their own generations. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. 2025. Zep: A temporal knowledge graph architecture for agent memory. *Preprint*, arXiv:2501.13956.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D Manning. 2024. Raptor: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations*.

Koki Wataoka, Tsubasa Takahashi, and Ryokan Ri. 2024. Self-preference bias in LLM-as-a-judge. In *Neurips Safe Generative AI Workshop 2024*.

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*.

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

A Appendix

A.1 Case Study

In this subsection we provide one case to directly show how our pipeline work.

Table 3 shows the case question, answer and corresponding conversation.

Table 4 shows that the question's embedding retrieves 10 candidates, including the chunk (chunk 4) that contains the corresponding conversation:

Table 5 shows that LLM reranks and keeps chunk 4 as the final memory.

Eventually, model generate correct answer "7 May 2023".

A.2 Prompt

We show all prompts in Table 6, Table 8, Table 7 and Table 9.

Table 3: Case question, answer and corresponding conversation.

```
Question: When did Caroline go to the LGBTQ support group?
Answer: 7 May 2023
Conversation: Session 1 on 1:56 pm on 8 May, 2023:
Caroline: Hey Mel! Good to see you! How have you been?
Melanie: Hey Caroline! Good to see you! I’m swamped with the kids and work. What’s up with you?
    Anything new?
Caroline: I went to a LGBTQ support group yesterday and it was so powerful.
```

Table 4: Retrieved Chunks.

```
Chunk 1:
"Melanie: Thanks, Caroline! I’m really lucky to have my family; they bring so much joy and love.\n[
    Session 11 on 2:24 pm on 14 August, 2023]\nMelanie: Hey Caroline! Last night was amazing! We
    celebrated my daughter’s birthday with a concert surrounded by music, joy and the warm summer
    breeze. Seeing my..."

Chunk 2:
"Caroline: Seeing my mentee’s face light up when they saw the support was the best! Such a special
    moment.\nMelanie: Wow, Caroline! They must have felt so appreciated. It’s awesome to see the
    difference we can make in each other’s lives. Any other exciting LGBTQ advocacy stuff coming up
    ?\nCaroline: Yay..."

Chunk 3:
"Melanie: That’s great, Caroline! Loving the inclusivity and support. Anything you’re excited for in
    the adoption process?\nCaroline: I’m thrilled to make a family for kids who need one. It’ll be
    tough as a single parent, but I’m up for the challenge!\nMelanie: You’re doing something amazing
    ! Creating..."

Chunk 4:
"[Session 1 on 1:56 pm on 8 May, 2023]\nCaroline: Hey Mel! Good to see you! How have you been?\
    nMelanie: Hey Caroline! Good to see you! I’m swamped with the kids & work. What’s up with you?
    Anything new?\nCaroline: I went to a LGBTQ support group yesterday and it was so powerful.\
    nMelanie: Wow, that’s co..."

(6 more chunks)...
```

Table 5: Final Chunk.

```
Chunk 4:
"[Session 1 on 1:56 pm on 8 May, 2023]\nCaroline: Hey Mel! Good to see you! How have you been?\
    nMelanie: Hey Caroline! Good to see you! I’m swamped with the kids & work. What’s up with you?
    Anything new?\nCaroline: I went to a LGBTQ support group yesterday and it was so powerful.\
    nMelanie: Wow, that’s co..."
```

Table 6: Prompt for question type: temporal

```
You are a careful assistant that answers questions based ONLY on the provided memory snippets from a long conversation.\n
If the memories are insufficient or conflicting, you must say you are not sure and explain briefly.\n
Do not invent facts that are not supported by the memories.\n
Question type: Temporal

Memory: {Memory blocks}
Question: {Question}

Pay careful attention to temporal expressions. If the question asks for ’when’, ’first’, ’last’, ’before’, ’after’, or a duration, you must compare the times of relevant events explicitly when deciding the answer.\n
If a memory uses relative time like ’yesterday’, ’last week’, ’last year’, ’next month’, convert it into an explicit calendar date, month+year, or year, using the session date and time given in the memories.\n

Formatting rules for your final answer:\n
1. Your ENTIRE reply must be ONLY the final answer string.\n
   - Do NOT show your reasoning.\n
   - Do NOT repeat the question.\n
   - Do NOT add any explanation, apology, or extra sentences.\n
   - Do NOT prefix with ’Answer:’ or similar.\n
2. The answer must be a single short phrase that directly answers the question.\n
   - If multiple items are needed, separate them with commas and a space.\n
   - Examples of valid answer styles:\n
       8 May 2025\n
       June 2012\n
       2018\n
       The week before 13 July 2004\n
       Transgender woman\n
       Psychology, counseling certification\n
       pottery, camping, painting, boxing\n
3. When the memories contain a short span that answers the question,
copy that span verbatim or with minimal changes, instead of paraphrasing.\n

For temporal questions, prefer explicit calendar references over relative words.\n
For example, if a memory says ’last year’ in a session dated 2023,
answer ’2022’. If it says ’last week’ in a session dated 9 June 2023,
you may answer ’The week before 9 June 2023’.\n
```

Table 7: Prompt for question type: single fact

```
You are a careful assistant that answers questions based ONLY on the provided memory snippets from a long conversation.\n
If the memories are insufficient or conflicting, you must say you are not sure and explain briefly.\n
Do not invent facts that are not supported by the memories.\n

Question type: Single Fact

Memory: {Memory blocks}
Question: {Question}

Formatting rules for your final answer:\n
1. Your ENTIRE reply must be ONLY the final answer string.\n
   - Do NOT show your reasoning.\n
   - Do NOT repeat the question.\n
   - Do NOT add any explanation, apology, or extra sentences.\n
   - Do NOT prefix with 'Answer:' or similar.\n
2. The answer must be a single short phrase that directly answers the question.\n
   - If multiple items are needed, separate them with commas and a space.\n
   - Examples of valid answer styles:\n
       8 May 2025\n
       June 2012\n
       2018\n
       The week before 13 July 2004\n
       Transgender woman\n
       Psychology, counseling certification\n
       pottery, camping, painting, boxing\n
3. When the memories contain a short span that answers the question,
copy that span verbatim or with minimal changes, instead of paraphrasing.\n

For temporal questions, prefer explicit calendar references over relative words.\n
For example, if a memory says 'last year' in a session dated 2023,
answer '2022'. If it says 'last week' in a session dated 9 June 2023,
you may answer 'The week before 9 June 2023'.\n
```

Table 8: Prompt for question type: multi hop

```
You are a careful assistant that answers questions based ONLY on the provided memory snippets from a long conversation.\n
If the memories are insufficient or conflicting, you must say you are not sure and explain briefly.\n
Do not invent facts that are not supported by the memories.\n

Question type: Multi Hop

Memory: {Memory blocks}
Question: {Question}

This question likely requires combining multiple pieces of evidence.
Use all relevant memories together when deciding the answer.\n

Formatting rules for your final answer:\n
1. Your ENTIRE reply must be ONLY the final answer string.\n
   - Do NOT show your reasoning.\n
   - Do NOT repeat the question.\n
   - Do NOT add any explanation, apology, or extra sentences.\n
   - Do NOT prefix with 'Answer:' or similar.\n
2. The answer must be a single short phrase that directly answers the question.\n
   - If multiple items are needed, separate them with commas and a space.\n
   - Examples of valid answer styles:\n
       8 May 2025\n
       June 2012\n
       2018\n
       The week before 13 July 2004\n
       Transgender woman\n
       Psychology, counseling certification\n
       pottery, camping, painting, boxing\n
3. When the memories contain a short span that answers the question,
copy that span verbatim or with minimal changes, instead of paraphrasing.\n

For temporal questions, prefer explicit calendar references over relative words.\n
For example, if a memory says 'last year' in a session dated 2023,
answer '2022'. If it says 'last week' in a session dated 9 June 2023,
you may answer 'The week before 9 June 2023'.\n
```

Table 9: Prompt for question type: other

```
You are a careful assistant that answers questions based ONLY on the provided memory snippets from a long conversation.\n
If the memories are insufficient or conflicting, you must say you are not sure and explain briefly.\n
Do not invent facts that are not supported by the memories.\n

Question type: Other

Memory: {Memory blocks}
Question: {Question}

The question may be unanswerable from the memories alone.
If the memories do not clearly contain the required information,
explicitly say that the question cannot be answered from the
given memories.\n

Formatting rules for your final answer:\n
1. Your ENTIRE reply must be ONLY the final answer string.\n
   - Do NOT show your reasoning.\n
   - Do NOT repeat the question.\n
   - Do NOT add any explanation, apology, or extra sentences.\n
   - Do NOT prefix with 'Answer:' or similar.\n
2. The answer must be a single short phrase that directly answers the question.\n
   - If multiple items are needed, separate them with commas and a space.\n
   - Examples of valid answer styles:\n
       8 May 2025\n
       June 2012\n
       2018\n
       The week before 13 July 2004\n
       Transgender woman\n
       Psychology, counseling certification\n
       pottery, camping, painting, boxing\n
3. When the memories contain a short span that answers the question,
copy that span verbatim or with minimal changes, instead of paraphrasing.\n

For hypothetical or likelihood questions, answer with a short phrase like 'Likely yes', 'Likely no', or another short phrase that best matches the memories.
Still do NOT add any explanation.\n
```