Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Remember What Matters: Utility-Aware Adaptive Memory Management for Long-Term LLM Agents

Mohamed Hechmi

mohamedamine.hechmi@supcom.tn

University of Carthage

Research Article

Keywords: Large Language Models, Conversational Mem- ory, Long-Term Question Answering, Information Retrieval, Hy- brid Retrieval, Cross-Encoder Reranking, Retrieval-Augmented Generation, Evidence Retrieval

Posted Date: November 26th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11870850/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Remember What Matters: Utility-Aware Adaptive Memory Management for Long-Term LLM Agents

Mohamed Amine Hechmi[1]
[1]Sup'Com, University of Carthage, Mediatron Lab, Tunisia

Abstract

Long-term conversational agents powered by large language models (LLMs) require effective mechanisms for retrieving relevant evidence from historical interactions. Although lexical retrieval, semantic similarity, and hybrid retrieval provide complementary capabilities, relying on a single ranking signal may fail to place the most informative conversational sessions near the top of the retrieved list. This challenge is particularly important in long-term question answering, where the correct response may depend on information distributed across multiple historical sessions.

This paper proposes *Remember What Matters* (RWM), a conversational memory retrieval framework that combines multi-source candidate generation with cross-encoder-based reranking. RWM constructs a candidate pool using hybrid retrieval, BM25, and semantic similarity, and then refines the ranking by combining cross-encoder relevance scores with a normalized hybrid-retrieval signal. The approach aims to improve evidence retrieval while limiting the computational cost of cross-encoder inference through candidate-pool restriction.

RWM is evaluated on the cleaned LongMemEval-S dataset, comprising 500 questions and 23,854 historical sessions. The selected configuration uses a candidate pool of 20 sessions and a fusion parameter of 0.20, determined through validation. On a held-out test set of 200 questions, RWM achieves an EvidenceRecall@5 of 0.9684, a Hit@5 of 1.000, a StrictRecall@5 of 0.930, and a mean reciprocal rank (MRR) of 0.9395. Compared with the Hybrid baseline, EvidenceRecall@5 improves by 1.17 percentage points. Bootstrap analysis yields a 95% confidence interval of [0.0025, 0.0225] for the difference in EvidenceRecall@5, indicating a positive improvement under the reported evaluation protocol.

These findings support the potential of combining complementary retrieval sources with cross-encoder reranking for conversational evidence retrieval. They do not, however, establish the effectiveness of a complete persistent-memory lifecycle policy. Further experiments across datasets and resource configurations are required to assess generalization, computational overhead, and the integration of retrieval with memory admission, updating, and retention.

Large Language Models, Conversational Memory, Long-Term Question Answering, Information Retrieval, Hybrid Retrieval, Cross-Encoder Reranking, Retrieval-Augmented Generation, Evidence Retrieval.

1 Introduction

Large language models (LLMs) have demonstrated strong capabilities in natural language understanding, reasoning, and instruction following. Their integration into autonomous and interactive agents has created opportunities for applications involving personalized assistance, multi-step reasoning, and sustained interactions. However, long-term interaction introduces an important challenge: an agent must retrieve relevant information from previous sessions when a current query depends on historical context. The information required to answer a question may have been introduced much earlier, expressed using different terminology, or distributed across several conversational sessions.

The Transformer architecture introduced by Vaswani et al [1] provides the foundation for many contemporary LLMs through its attention-based mechanism for processing contextual information. Nevertheless, the information available to a model during inference is constrained by its context window and computational resources. Supplying an entire interaction history for every query can increase inference costs and introduce irrelevant information without guaranteeing that the model will identify the correct evidence. Consequently, effective long-term conversational systems require mechanisms for locating and selecting historical information before it is passed to the language model.

Retrieval-augmented generation (RAG) combines language generation with the retrieval of information from an external knowledge source [2]. Within conversational systems, retrieval can be performed using lexical matching, semantic representations, or hybrid approaches that combine multiple signals. Lexical methods such as BM25 can identify exact terms and expressions, whereas semantic retrieval can identify related content even when the wording differs. Hybrid retrieval attempts to exploit the complementary strengths of these approaches. However, the initial retrieval ranking may still place less informative sessions above sessions that contain stronger evidence for the current query.

This limitation motivates the separation of candidate generation from final ranking. Candidate generation aims to retrieve a sufficiently diverse set of potentially relevant sessions, whereas reranking evaluates the relationship between the query and each candidate in greater detail. Cross-encoders provide one way to perform this second-stage assessment by processing query-candidate pairs jointly. However, scoring every session in a large conversational repository may be computationally expensive. Restricting cross-encoder inference to a candidate pool provides a practical compromise between retrieval quality and computational cost, although its effectiveness depends on whether the initial retrieval stage includes the relevant evidence.

Beyond ranking quality, long-term conversational memory raises broader lifecycle questions. An agent may need to decide which information to retain, how to handle outdated or contradictory statements, and when redundant information should be consolidated or removed. These problems are related to, but distinct from, the retrieval problem addressed in this work. A retrieval method can improve the ordering of available

evidence without necessarily solving persistent-memory admission, updating, conflict resolution, or eviction. Maintaining this distinction is essential when evaluating the scope and effectiveness of a memory architecture.

To address the retrieval challenge, this paper proposes *Remember What Matters* (RWM), a conversational memory retrieval framework that combines multi-source candidate generation with cross-encoder-based reranking. RWM constructs a candidate pool from three complementary retrieval sources: hybrid retrieval, BM25, and semantic similarity. The resulting candidates are scored using a pretrained cross-encoder, and the cross-encoder signal is combined with a normalized hybrid-retrieval feature to produce the final ranking. This design aims to improve the position of relevant historical sessions while limiting cross-encoder inference to a bounded set of candidates.

The framework is evaluated on the cleaned LongMemEval-S dataset, which contains 500 questions and 23,854 historical sessions. Candidate-pool size and score-fusion parameters are selected using validation data. The final evaluation compares RWM with established retrieval baselines using evidence-retrieval metrics, including EvidenceRecall@5, Hit@5, StrictRecall@5, and mean reciprocal rank (MRR). A bootstrap analysis is additionally used to estimate uncertainty in the observed difference between RWM and the Hybrid baseline.

The experimental results show that RWM achieves an EvidenceRecall@5 of 0.9684 on a held-out test set of 200 questions, compared with 0.9568 for Hybrid. RWM also achieves a Hit@5 of 1.000, a StrictRecall@5 of 0.930, and an MRR of 0.9395. The observed EvidenceRecall@5 improvement is approximately 1.17 percentage points, with a reported 95% bootstrap confidence interval of [0.0025, 0.0225] for the difference. These findings support the effectiveness of the proposed retrieval and reranking pipeline under the evaluated conditions. They do not establish that the method generalizes to all conversational-memory settings or that it independently solves the broader persistent-memory lifecycle problem.

The main contributions of this work are summarized as follows:

- We propose RWM, a conversational memory retrieval framework that combines hybrid retrieval, BM25, and semantic similarity for multi-source candidate generation, followed by cross-encoder-based reranking.
- We formulate the reranking stage as a score-fusion mechanism that combines query-conditioned cross-encoder relevance with a normalized hybrid-retrieval signal. The candidate-pool size and fusion parameter are selected using validation data.
- We evaluate RWM on the cleaned LongMemEval-S dataset using EvidenceRecall@5, Hit@5, StrictRecall@5, and MRR, and compare its performance with multiple retrieval baselines under the reported evaluation protocol.

- We analyze the difference between RWM and the Hybrid baseline using bootstrap confidence intervals, providing an estimate of uncertainty in the observed evidence-retrieval improvement.

The remainder of this paper is organized as follows. Section II reviews related work on conversational memory, information retrieval, and retrieval-augmented generation. Section III presents the RWM framework and its retrieval architecture. Section IV formulates the candidate-generation and reranking mechanisms. Section V describes the experimental setup and evaluation metrics. Section VI discusses the results and limitations, and Section VII concludes the paper.

2 Related Work

2.1 Retrieval-Augmented Generation and External Memory

Retrieval-augmented generation (RAG) combines parametric language models with non-parametric knowledge sources to improve access to information that may not be encoded in model parameters. Lewis et al [2] introduced a general RAG framework in which retrieved documents are incorporated into the generation process. This paradigm provides a foundation for knowledge-intensive question answering and enables language models to exploit external information at inference time.

Despite its effectiveness, conventional retrieval pipelines may struggle when the knowledge source consists of long conversational histories containing repeated observations, semantically similar sessions, and information distributed across multiple interactions. A retrieval method based on a single scoring strategy may fail to identify all relevant evidence. Moreover, the relevance of a retrieved passage depends not only on its similarity to the query but also on the retrieval objective and the context in which the information is used.

RWM investigates this problem in the context of long-term conversational memory. Its experimentally evaluated V9 retrieval component combines complementary candidate-generation strategies and subsequently reranks the resulting candidates using a cross-encoder. Rather than introducing a new language model or replacing the general RAG paradigm, RWM aims to improve the identification of relevant conversational evidence before it is supplied to a downstream language model.

The broader RWM architecture additionally considers memory selection and memory evolution as important aspects of long-term memory management. These components define the intended scope of the framework, whereas the current experimental evaluation focuses on retrieval and reranking.

2.2 Persistent and Hierarchical Memory for LLM Agents

Several studies have explored persistent memory mechanisms that extend LLM-based agents beyond individual interactions. Generative Agents, proposed by Park et al [3], employ a memory stream that records observations and supports retrieval and reflection. This architecture demonstrates how accumulated experiences can inform subsequent agent behavior, although its primary objective is the simulation of believable human behavior.

MemGPT, introduced by Packer et al [3], addresses context limitations through a hierarchical memory architecture inspired by operating systems. It manages information across different memory tiers and allows the agent to move information into and out of its active context. MemoryBank, proposed by Zhong et al [4], investigates long-term memory for conversational systems and incorporates a memory mechanism inspired by human forgetting.

These approaches establish important foundations for persistent storage, context management, and retrieval across extended interactions. RWM shares the broader objective of enabling language models to use information from previous conversations. Its current experimental contribution is more specifically concerned with retrieving relevant sessions from a large conversational history by combining multiple retrieval strategies with cross-encoder-based reranking.

The broader RWM design also considers how information could be selected for retention and subsequently revised as new interactions occur. These aspects are treated as architectural objectives rather than experimentally established advantages of the current V9 implementation.

2.3 Dynamic Memory Organization and Evolution

Recent work has investigated memory systems that organize and revise stored information instead of treating each memory as an isolated record. A-MEM, proposed by Xu et al [5], introduces an agentic memory mechanism inspired by the Zettelkasten method. It dynamically links memories, creates structured notes, and allows new information to influence the organization and contextual representation of existing memories.

Such approaches highlight the importance of memory organization in long-running LLM interactions. They also motivate the distinction between retrieving relevant information from an existing repository and modifying the repository itself as new information becomes available.

The broader RWM framework considers memory evolution as a complementary mechanism to retrieval. Conceptually, this component may support operations such as updating, consolidation, conflict handling, and controlled forgetting. However, the current V9 experiments do not establish the effectiveness of these operations. In particular, they

do not demonstrate that RWM outperforms existing dynamic memory systems in memory maintenance or temporal consistency.

Accordingly, the present study evaluates the retrieval component separately from the broader memory evolution concept. A systematic comparison with dynamic memory systems, including A-MEM, remains an important direction for future work.

2.4 Evaluation of Long-Term Conversational Retrieval

Evaluating long-term conversational memory requires distinguishing the ability to retrieve relevant evidence from the ability to generate a correct answer using that evidence. A system may retrieve useful conversational sessions without necessarily producing a correct final answer. Conversely, a fluent response does not guarantee that the information supporting it was retrieved from the appropriate historical interaction.

LongMemEval, introduced by Wu et al. (**wu2025longmemeval?**), provides an evaluation framework for long-term conversational memory. Its tasks cover information extraction, multi-session reasoning, temporal reasoning, knowledge updates, and abstention. These tasks provide a relevant setting for studying whether systems can retrieve and use information distributed across extended conversational histories.

MemoryAgentBench, introduced by Hu et al [6], evaluates memory agents through incremental multi-turn interactions and considers accurate retrieval, test-time learning, long-range understanding, and selective forgetting. It offers a complementary perspective on systems that must continuously process new information rather than operate exclusively on a fixed collection of conversations.

The current RWM evaluation focuses on evidence retrieval from the LongMemEval-S dataset. The V9 experiment uses a held-out test set of 200 questions and assesses retrieval through EvidenceRecall, Hit, StrictRecall, and mean reciprocal rank (MRR). These metrics characterize the quality of the retrieved candidates; they do not, by themselves, establish improvements in final answer accuracy, temporal consistency, selective forgetting, or memory-management efficiency.

Future evaluations should extend this analysis to answer-level performance, retrieval latency, token consumption, and the behavior of memory-selection and memory-evolution mechanisms.

2.5 Research Gap and Positioning of RWM

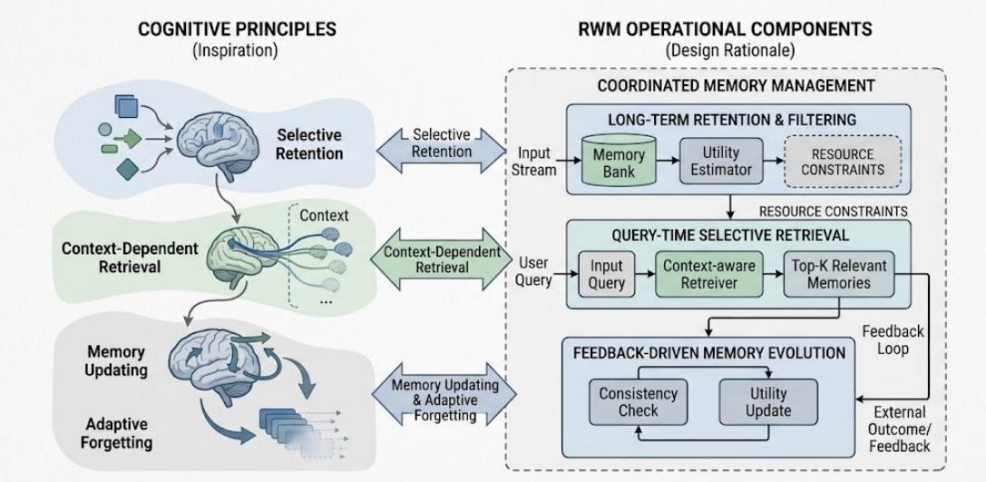
The reviewed literature demonstrates substantial progress in retrieval-augmented generation, hierarchical context management, persistent conversational memory, dynamic memory organization, and long-term memory evaluation. Consequently, RWM is not positioned as the first architecture to introduce persistent memory, hybrid retrieval, memory updating, or forgetting.

Instead, RWM investigates how complementary retrieval strategies and cross-encoder reranking can improve the identification of relevant evidence in long conversational histories. Its broader architectural motivation is to integrate this retrieval capability into a memory framework that also considers memory selection and evolution.

The experimentally evaluated V9 component combines Hybrid, BM25, and semantic similarity retrieval to construct a candidate pool, applies cross-encoder-based relevance estimation, and fuses the resulting scores with a normalized Hybrid signal before selecting the highest-ranked sessions. The resulting retrieval performance is compared with a Hybrid baseline under the reported experimental protocol.

The central empirical question addressed in this study is whether this retrieval and reranking strategy improves conversational evidence retrieval relative to the selected baseline. The broader hypothesis that coordinating retrieval with memory selection and memory evolution can improve long-term recall, consistency, and resource efficiency remains to be investigated through additional experiments, ablations, and comparisons under controlled resource constraints.

3 Cognitive Inspiration and Design Rationale



Cognitive inspiration and computational design rationale of the RWM framework. Principles associated with selective retention, context-dependent retrieval, memory updating, and adaptive forgetting motivate the proposed memory-selection, adaptive-retrieval, and memory-evolution mechanisms. The V9 experiments evaluate the retrieval and reranking component, whereas the complete memory lifecycle remains to be validated. The mapping represents a conceptual analogy rather than a biological model of human memory.

3.1 Cognitive Motivation

The Remember What Matters (RWM) framework draws conceptual inspiration from functional principles associated with human memory, particularly selective retention,

context-dependent retrieval, memory updating, and adaptive forgetting. Human memory does not operate as a passive, verbatim repository of all past experiences. Instead, the accessibility and use of past information depend on multiple interacting processes that influence how experiences are encoded, retained, retrieved, and modified over time.

These principles provide a conceptual motivation for designing memory mechanisms for large language model (LLM)-based agents. During long-running interactions, conversational histories may grow substantially and contain repeated, outdated, or context-dependent information. Retrieving an appropriate historical interaction is therefore distinct from simply storing the complete interaction history or selecting the most recent observation.

RWM addresses this challenge at the architectural level by distinguishing persistent memory, query-dependent retrieval, and memory evolution. The experimentally evaluated V9 implementation focuses on the retrieval component, while the other mechanisms describe the broader design objectives of the framework. The cognitive analogy motivates these design choices but does not imply that the computational mechanisms reproduce biological memory.

3.2 Functional Principles and Computational Analogies

3.2.0.1Selective retention.

The principle of selective retention motivates the idea that not every conversational observation should necessarily be preserved in long-term memory. In a computational memory system, retention decisions may depend on estimated future usefulness, confidence, provenance, redundancy, temporal validity, and storage cost.

Let $\mathcal{C}_t$ denote candidate memories extracted at time t, and let $\mathcal{M}_t$ denote the persistent memory repository. A general memory-selection mechanism would determine which candidates should be added to or used to update the repository, subject to the available storage resources.

This mechanism is part of the broader RWM design rationale. The current V9 retrieval experiments do not evaluate a memory-admission policy or demonstrate improvements in storage efficiency.

3.2.0.2Context-dependent retrieval.

Human recollection depends partly on retrieval cues and the context in which information is needed. Analogously, a conversational memory system should select historical information according to the current query rather than relying exclusively on recency or a fixed ordering of stored observations.

For a query q_t and a repository $\mathcal{M}_t$, a retrieval mechanism aims to identify a subset of historical records that provides relevant evidence for the current task. Different retrieval strategies may identify different candidates because they rely on different signals, including lexical matching and semantic similarity.

The V9 implementation operationalizes this principle by combining Hybrid retrieval, BM25 retrieval, and semantic similarity retrieval. Their candidate sets are aggregated, and a cross-encoder subsequently evaluates query–session pairs. A score-fusion stage combines the cross-encoder signal with a normalized Hybrid signal before producing the final ranking.

This component provides the principal experimentally evaluated connection between the cognitive motivation of RWM and its implemented computational mechanism.

3.2.0.3Memory updating and temporal consistency.

In human memory, past experiences may be reinterpreted as new information becomes available. An analogous requirement arises in conversational systems when a user corrects a previous statement, changes a preference, or provides information whose validity is restricted to a particular period.

A persistent memory system may therefore require operations for updating, merging, linking, or marking records as superseded or uncertain. Such operations should consider provenance, timestamps, and temporal scope rather than assuming that every new statement automatically replaces an older one.

These operations are conceptual components of the broader RWM architecture. They are not evaluated by the current V9 retrieval experiment, which does not establish the correctness of memory updates or the resolution of contradictory records.

3.2.0.4Adaptive forgetting and resource management.

The principle of adaptive forgetting motivates the management of stored information when memory resources are limited. In a computational system, obsolete or redundant records may be candidates for archival or removal, provided that the decision does not eliminate information needed for subsequent tasks.

A complete evaluation of such a mechanism would need to distinguish removal from active memory, archival for historical retrieval, and the determination that a fact is no longer valid. It would also need to quantify resource savings and the risk of losing information required by future queries.

In RWM, adaptive forgetting belongs to the broader memory-management vision. The current experiments do not implement or validate a forgetting policy, and no claim of reduced storage consumption or improved selective forgetting is made on the basis of V9 results.

3.3 From Cognitive Principles to the RWM Architecture

The preceding principles motivate three complementary mechanisms within the broader RWM design.

First, *memory selection* concerns deciding which information should be retained in persistent conversational memory. It is motivated by the need to control redundancy and preserve information that may remain useful across interactions.

Second, *adaptive retrieval* concerns selecting historical information in response to a particular query. The V9 implementation instantiates this component through multi-source candidate generation, cross-encoder reranking, score fusion, and top-k selection.

Third, *memory evolution* concerns revising the persistent repository as new interactions and reliable evidence become available. Potential operations include consolidation, updating, conflict handling, and controlled forgetting. These operations remain part of the broader architectural proposal and require dedicated implementation and evaluation.

The architecture also distinguishes two resource constraints. A persistent-memory budget limits the cost of maintaining stored information, whereas a context budget limits the amount of retrieved information supplied to an LLM invocation. These constraints are conceptually distinct, but the current V9 experiment does not evaluate a joint optimization of persistent storage and context-token consumption.

The distinction between the overall architecture and the experimentally evaluated retrieval component is essential. The V9 results provide evidence about retrieval quality under the reported protocol; they do not establish the effectiveness of the complete memory lifecycle.

3.4 Research Hypothesis and Scope of Contribution

The design rationale of RWM motivates the hypothesis that combining complementary retrieval strategies with cross-encoder reranking can improve the identification of relevant evidence in long conversational histories.

The current experimental contribution evaluates this hypothesis using a held-out test set of 200 questions from LongMemEval-S. RWM achieves an EvidenceRecall@5 of 0.9684, compared with 0.9568 for the Hybrid baseline. It also obtains a Hit@5 of 1.000, a StrictRecall@5 of 0.930, and an MRR of 0.9395 under the reported protocol.

These results support a retrieval-focused claim: the evaluated RWM pipeline improves the measured evidence-retrieval performance relative to the selected Hybrid baseline on this test set. They do not establish that RWM solves the general problem of LLM memory, improves final answer accuracy, or provides more efficient persistent storage.

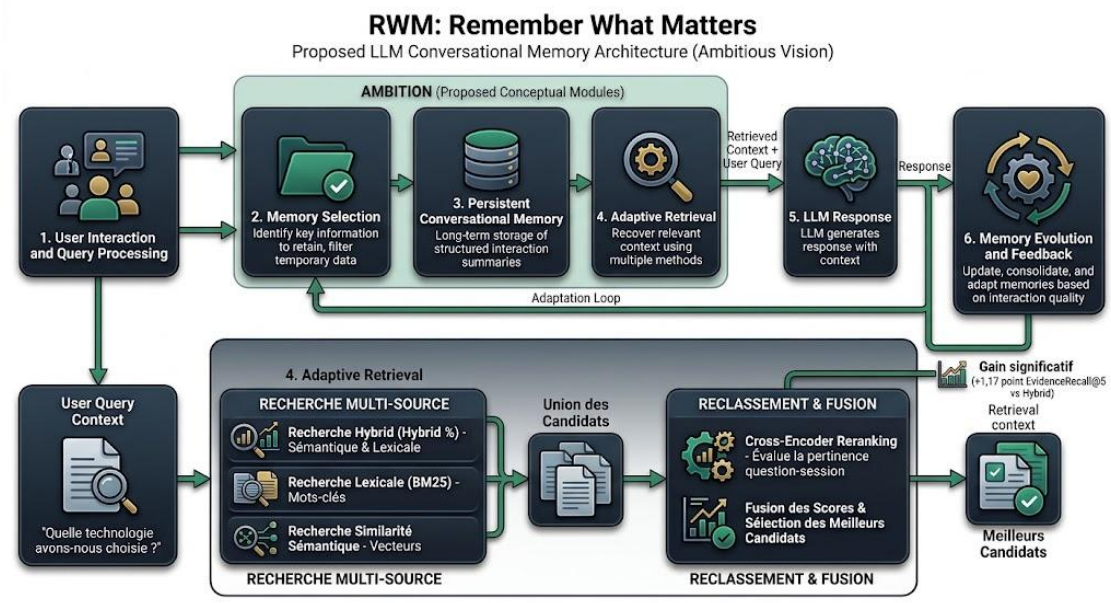
The broader hypothesis is that integrating retrieval with explicit memory selection and memory evolution may provide further benefits in long-running interactions. Evaluating this hypothesis will require dedicated experiments on memory retention, update correctness, temporal consistency, selective forgetting, answer-level accuracy, and resource consumption.

Figure 1 summarizes the conceptual relationship between the cognitive principles motivating RWM and its proposed computational mechanisms. It represents a design rationale rather than a claim of biological equivalence or experimental validation of every architectural component.

4 Proposed Approach

4.1 System Model

4.1.1 System Overview

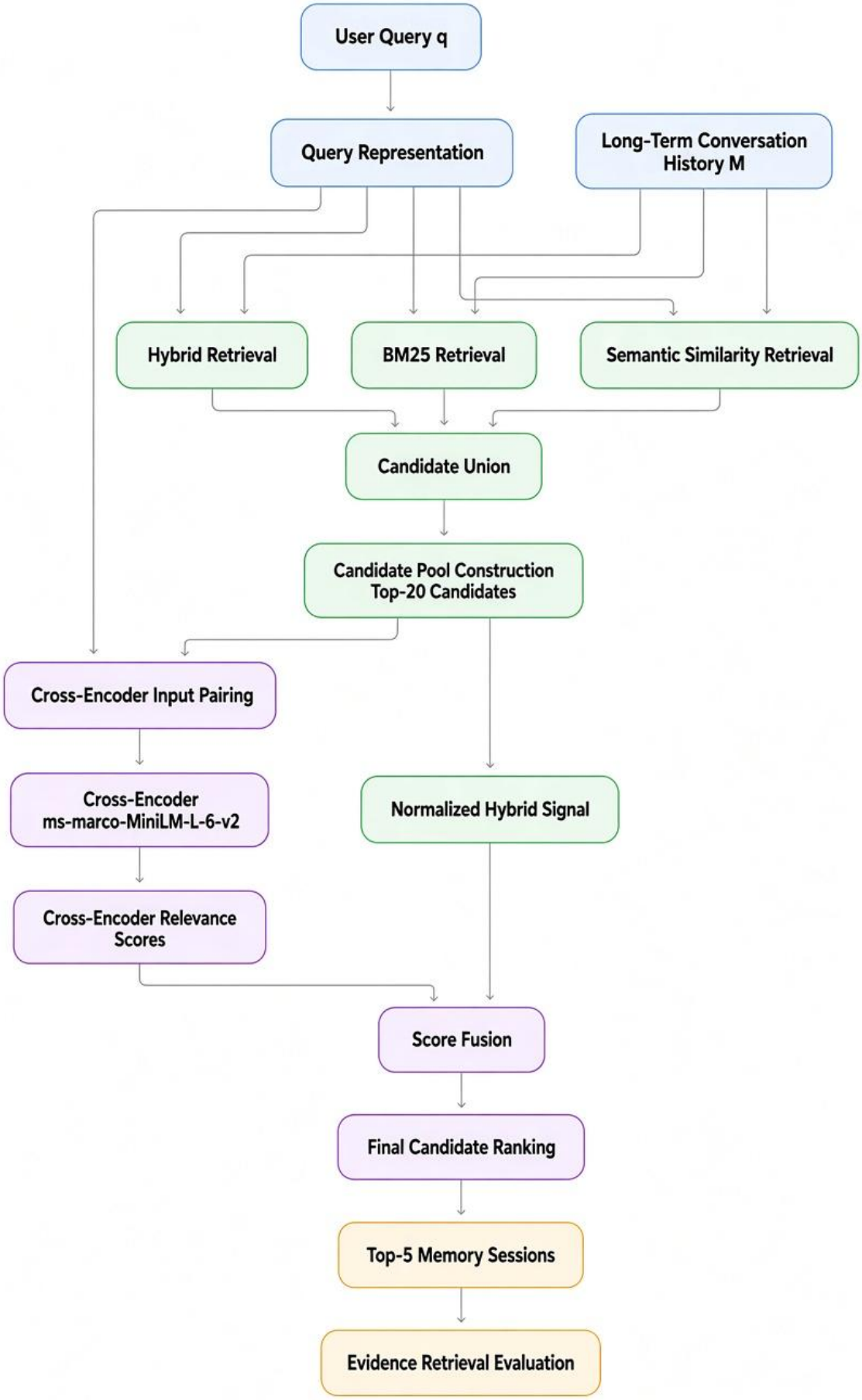


System model of Remember What Matters (RWM), illustrating multi-source candidate generation, candidate aggregation, cross-encoder relevance estimation, score fusion, and final conversational evidence retrieval.

We consider a long-term conversational question-answering system in which a pretrained large language model (LLM) interacts with a user over multiple sessions. As interactions accumulate, the system maintains a repository of historical conversational sessions that may contain information relevant to future queries. The main challenge is to identify the most useful historical evidence for a new query without requiring the downstream LLM to process the entire conversation history.

Remember What Matters (RWM) is designed as a query-dependent conversational memory retrieval and reranking framework. Its primary objective is to improve the selection and ordering of historical sessions by combining heterogeneous retrieval sources with a cross-encoder-based relevance estimation stage. Instead of relying exclusively on lexical matching or semantic similarity, RWM aggregates candidates from multiple retrieval

mechanisms and subsequently refines their ranking using query–session interaction features.


User Query q
Query Representation
Long-Term Conversation History M
Hybrid Retrieval
BM25 Retrieval
Semantic Similarity Retrieval
Candidate Union
Candidate Pool Construction Top-20 Candidates
Cross-Encoder Input Pairing
Cross-Encoder ms-marco-MiniLM-L-6-v2
Cross-Encoder Relevance Scores
Normalized Hybrid Signal
Score Fusion
Final Candidate Ranking
Top-5 Memory Sessions
Evidence Retrieval Evaluation

Flowchart of the Proposed Remember What Matters (RWM) Retrieval Pipeline

The overall system can be represented by the tuple

$$\mathcal{S}_{\mathrm{RWM}} = (\mathcal{H}, \mathcal{Q}, \mathcal{R}, f_{\mathrm{CE}}, \Phi, \alpha, K),$$

where $\mathcal{H}$ is the historical-session repository, $\mathcal{Q}$ is the query space, $\mathcal{R}$ denotes the set of candidate-retrieval mechanisms, f_{CE} is the cross-encoder scoring function, Φ is the score-fusion function, α is its configuration parameter, and K is the maximum candidate-pool size.

The system operates through the following logical stages:

$$\begin{aligned} q &\rightarrow \text{Candidate Generation} \\ &\rightarrow \text{Candidate Aggregation} \\ &\rightarrow \text{Cross-Encoder Scoring} \\ &\rightarrow \text{Score Fusion} \\ &\rightarrow \text{Final Ranking.} \end{aligned}$$

The output is a ranked list of historical sessions that can be evaluated against the benchmark's reference evidence or supplied as contextual information to a downstream LLM. The current experimental implementation evaluates the retrieval and ranking stages rather than the correctness of generated answers.

4.1.2 Historical Conversation Repository

Let the repository of historical conversational sessions be defined as

$$\mathcal{H} = \{h_1, h_2, \dots, h_M\},$$

where M denotes the number of available sessions and h_i represents the textual content of session i. Each session may contain one or more conversational turns, including user questions and assistant responses.

For a given query q, the system searches this repository to identify sessions that contain evidence relevant to the current information need. The retrieval process is query-dependent: a session may be relevant to one query and irrelevant to another. Consequently, relevance is represented as a relation between a query and a historical session rather than as a permanent property of the session.

We express this relation as

$$\mathrm{Rel}: \mathcal{Q} \times \mathcal{H} \rightarrow \mathbb{R},$$

where $\mathrm{Rel}(q, h_i)$ denotes the relevance of session h_i to query q. This quantity is not assumed to be directly observable. RWM estimates it using the outputs of its retrieval and reranking components.

Let $\mathcal{G}(q) \subseteq \mathcal{H}$ denote the set of reference sessions or evidence units associated with query q under the benchmark annotations. The retrieval objective is to place sessions containing relevant evidence near the beginning of the ranked list.

Importantly, the repository is treated as an available collection of historical sessions during retrieval. The current experimental model does not introduce a learned policy that determines which new interactions should be stored, which existing sessions should be deleted, or how contradictory memories should be reconciled.

4.1.3 Query Representation and Retrieval Sources

Given an incoming query $q \in \mathcal{Q}$, RWM invokes three complementary retrieval sources:

$$\mathcal{J} = \{\mathrm{hyb}, \mathrm{BM25}, \mathrm{sim}\},$$

where hyb denotes hybrid retrieval, $\mathrm{BM25}$ denotes lexical retrieval, and sim denotes semantic-similarity retrieval.

Each source produces a ranked list of historical sessions:

$$\mathcal{L}_j(q) = \left[h_{j,1}, h_{j,2}, \ldots, h_{j,n_j}\right], \qquad j \in \mathcal{J},$$

where n_j is the number of candidates returned by source j.

The retrieval sources provide complementary signals. BM25 estimates lexical relevance based on query-term matching. Semantic similarity estimates the relationship between query and session representations. Hybrid retrieval combines the matching signals available in its implementation. These mechanisms can retrieve different sessions because lexical overlap, semantic correspondence, and combined relevance do not necessarily identify the same candidates.

The system does not assume that these retrieval sources are statistically independent or that each source always contributes unique relevant evidence. Their usefulness depends on the query, the repository, and the retrieval configurations.

4.1.4 Candidate Aggregation and Pool Construction

The candidate-generation stage combines the outputs of the retrieval sources into a unified candidate set. The initial union is defined as

$$\mathcal{C}(q) = \bigcup_{j \in \mathcal{J}} \mathcal{L}_j(q).$$

Since a historical session may appear in several source lists, the union is defined over unique session identifiers. Duplicate removal avoids treating repeated occurrences of the same session as separate evidence items.

The aggregated candidate set is then restricted to a maximum size K:

$$\mathcal{C}_K(q) = \mathrm{Select}_K(\mathcal{C}(q)),$$

subject to

$$|\mathcal{C}_K(q)| \leq \min(K, |\mathcal{C}(q)|).$$

Here, Select_K represents the candidate-aggregation and truncation procedure implemented in RWM. In the evaluated configuration, the maximum candidate-pool size is $K = 20$.

The candidate pool serves two purposes. First, it combines the outputs of heterogeneous retrieval sources. Second, it limits the number of query–session pairs processed by the more computationally demanding cross-encoder.

Candidate generation also introduces an important limitation. If a relevant session is absent from the aggregated candidate set, subsequent reranking cannot recover it. Therefore, the quality of the final ranking depends both on the initial candidate coverage and on the ability of the reranker to order the retrieved candidates correctly.

4.1.5 Cross-Encoder-Based Relevance Estimation

After candidate-pool construction, RWM evaluates each candidate jointly with the query. Let $h_i \in \mathcal{C}_K(q)$. Its cross-encoder score is defined as

$$e(q, h_i) = f_{\mathrm{CE}}(q, h_i),$$

where f_{CE} is the pretrained model `cross-encoder/ms-marco-MiniLM-L-6-v2`.

The cross-encoder jointly processes the query and candidate text to estimate their relevance. This differs from a bi-encoder approach in which query and session embeddings are computed independently and compared using a similarity function. Joint processing provides a query-conditioned relevance signal, although its inference cost is incurred for each candidate pair.

For a candidate pool containing K_q sessions, where

$$K_q = |\mathcal{C}_K(q)|,$$

the number of cross-encoder evaluations per query is

$$N_{\mathrm{CE}}(q) = K_q \leq K.$$

The cross-encoder therefore refines the ordering within the candidate pool instead of searching the complete historical repository. Its role is to provide a more detailed relevance estimate for the candidates already identified by the retrieval sources.

The cross-encoder output is treated as a model score. Unless the implementation explicitly calibrates the output, it should not be interpreted as a probability that a session contains the required evidence.

4.1.6 Hybrid Signal and Score Fusion

In addition to the cross-encoder score, RWM uses a normalized feature derived from hybrid retrieval. For a candidate session h_i, let

$$e_i = e(q, h_i)$$

denote its cross-encoder score, and let

$$\tilde{h}_i = \mathrm{Norm}(h_{\mathrm{raw}}(q, h_i))$$

denote the normalized hybrid-retrieval feature. Here, $h_{\mathrm{raw}}(q, h_i)$ represents the underlying hybrid signal and $\mathrm{Norm}(\cdot)$ denotes the normalization procedure used in the implementation.

The final relevance score is formulated as

$$s_{\mathrm{RWM}}(q, h_i) = \Phi\big(e_i, \tilde{h}_i; \alpha\big),$$

where Φ is the implemented fusion function and α is its configuration parameter. The selected validation configuration uses

$$K^* = 20, \qquad \alpha^* = 0.20.$$

The purpose of fusion is to combine the query-conditioned cross-encoder signal with information provided by hybrid retrieval. The contribution of each signal depends on the implemented fusion equation and the scales of the input features. Accordingly, the normalization rule, treatment of missing hybrid scores, and exact mathematical definition of Φ must be consistent with the Python implementation.

The parameter α is not meaningful independently of this definition. It should not automatically be interpreted as the weight assigned to either component unless the implemented equation establishes that interpretation.

4.1.7 Final Ranking and Context Selection

After score fusion, candidates are sorted in descending order of their final scores. The RWM ranking for query q is

$$\mathcal{L}_{\mathrm{RWM}}(q) = \mathrm{SortDescending}\{(h_i, s_{\mathrm{RWM}}(q, h_i)) : h_i \in \mathcal{C}_K(q)\}.$$

The top-k sessions are selected as

$$\mathcal{R}_k(q) = \mathrm{TopK}_k(\mathcal{L}_{\mathrm{RWM}}(q)),$$

where $k = 5$ in the reported retrieval evaluation.

When RWM is integrated into a question-answering application, the selected sessions can be converted into a contextual input for the downstream LLM. The resulting answer can be represented abstractly as

$$\hat{y} = g_{\mathrm{LLM}}(q, \mathrm{Context}(\mathcal{R}_k(q))),$$

where g_{LLM} denotes the answer-generation model and $\mathrm{Context}(\cdot)$ denotes the procedure that formats the retrieved sessions as context.

Equation [eq:rwm_downstream_answer] describes the intended system integration. The current retrieval experiments evaluate the quality of the selected evidence and do not, by themselves, establish improvements in answer correctness or generation quality.

4.1.8 Optimization Objective and Evaluation

The central retrieval objective is to rank relevant historical evidence as highly as possible. Let $\mathcal{G}(q)$ denote the reference evidence for query q, and let m be a metric defined by the benchmark evaluation protocol. The general objective over N_q evaluation queries can be expressed as

$$J_{\mathrm{retrieval}} = \frac{1}{N_q} \sum_{i=1}^{N_q} m\left(\mathcal{R}_5(q_i), \mathcal{G}(q_i)\right).$$

The evaluation uses EvidenceRecall@5, Hit@5, StrictRecall@5, and mean reciprocal rank (MRR). These metrics characterize evidence retrieval coverage, successful retrieval under the benchmark's hit criterion, strict evidence-recall performance, and the rank of the first relevant result, respectively. Their precise definitions follow the benchmark annotations and the evaluation code.

The retrieval configuration is selected using validation data. Let $\mathcal{D}_{\mathrm{val}}$ denote the validation set and Θ the set of candidate configurations. Parameter selection can be represented as

$$(\alpha^*, K^*) = \underset{(\alpha,K)\in\Theta}{\arg\max}\ M_{\mathrm{val}}(\alpha, K),$$

where M_{val} denotes the metric used for configuration selection. The precise metric and search grid must correspond to the validation implementation. After selection, the configuration is fixed for evaluation on the held-out test set.

This procedure separates parameter selection from final testing and reduces the risk of choosing a configuration specifically to maximize performance on the test questions.

4.1.9 Design Rationale and System Boundaries

The RWM architecture separates candidate discovery from candidate ranking. Multi-source retrieval is responsible for identifying a sufficiently broad set of potentially relevant historical sessions, whereas the cross-encoder and score-fusion stages refine the order of that set. This separation provides a practical way to combine complementary retrieval signals while limiting the number of expensive relevance evaluations.

The system nevertheless has several boundaries. Its final ranking is constrained by candidate coverage; its computational cost depends on retrieval and cross-encoder

inference; and its evidence-retrieval performance does not directly establish the quality of the downstream generated answer.

Furthermore, the current implementation does not experimentally validate a complete memory-lifecycle mechanism. In particular, it does not establish learned policies for memory admission, retention, eviction, consolidation, contradiction resolution, or feedback-driven memory evolution. These mechanisms belong to the broader RWM research direction and require explicit algorithms and separate evaluation.

Consequently, the experimentally supported contribution of RWM is the combination of multi-source candidate generation, cross-encoder relevance estimation, and implementation-specific score fusion for conversational evidence retrieval. The broader objective of managing long-term conversational memory remains an architectural direction beyond the scope of the current retrieval experiments.

4.2 Analytical System

The evaluated Remember What Matters (RWM) system is formulated as a multi-stage retrieval and ranking framework for conversational memory. Given a user query and a repository of historical conversational sessions, the system identifies a restricted set of potentially relevant sessions and refines their ordering through query-conditioned relevance estimation and score fusion. The analytical formulation consists of candidate generation, candidate aggregation, cross-encoder scoring, score fusion, final ranking, and validation-based parameter selection.

Let $\mathcal{H} = \{h_1, h_2, \dots, h_M\}$ denote the repository of M historical conversational sessions, and let q denote an incoming query. The objective is to construct a ranked list of historical sessions whose content provides relevant evidence for the current query.

4.2.1 Problem Formulation

The retrieval problem is represented by a scoring function

$$s: \mathcal{Q} \times \mathcal{H} \to \mathbb{R},$$

where $\mathcal{Q}$ denotes the query space and $s(q, h_i)$ estimates the relevance of historical session h_i to query q.

For each query, the retrieval system produces a ranking

$$\pi_q = \operatorname{argsort}_{h_i \in \mathcal{H}}[-s(q, h_i)],$$

where π_q orders sessions by decreasing relevance score. Directly applying an expensive relevance model to every session can be computationally demanding when the historical repository is large. RWM therefore adopts a candidate-generation stage followed by a more detailed reranking stage.

Let $\mathcal{G}(q) \subseteq \mathcal{H}$ denote the candidate set generated for query q. The ranking problem is then restricted to

$$\pi_q^{(K)} = \text{argsort}_{h_i \in \mathcal{C}_K(q)}[-s(q, h_i)],$$

where $\mathcal{C}_K(q)$ is the candidate pool, with a maximum configured size of K sessions.

4.2.2 Multi-Source Candidate Generation

RWM combines three candidate-generation sources: hybrid retrieval, BM25, and semantic similarity. Let

$$\mathcal{J} = \{\text{hyb}, \text{BM25}, \text{sim}\}$$

denote the set of retrieval sources. For each source $j \in \mathcal{J}$, the retrieval procedure produces an ordered list

$$\mathcal{L}_j(q) = \left[h_{j,1}, h_{j,2}, \dots, h_{j,n_j}\right],$$

where n_j is the number of candidates returned by source j.

The initial candidate set is defined as the union of the source-specific lists:

$$\mathcal{C}(q) = \bigcup_{j \in \mathcal{J}} \mathcal{L}_j(q).$$

Since the same historical session may be retrieved by several sources, the union is interpreted as a set of unique sessions. Duplicate removal prevents a session from occupying multiple candidate positions solely because it was returned by different retrieval methods.

The final candidate pool is constructed using the candidate-aggregation and truncation procedure implemented in the system:

$$\mathcal{C}_K(q) = \text{Select}_K(\mathcal{C}(q)),$$

subject to

$$|\mathcal{C}_K(q)| \leq \min(K, |\mathcal{C}(q)|).$$

Here, $\text{Select}_K(\cdot)$ denotes the implemented candidate-selection rule. The selected configuration uses $K = 20$. The exact ordering used during truncation must be specified according to the implementation; the union operation alone does not determine which candidates are retained when the number of unique candidates exceeds K.

The use of multiple retrieval sources is intended to improve candidate coverage by combining different relevance signals. Lexical retrieval can identify exact terms and named entities, semantic similarity can identify candidates expressed using related language, and hybrid retrieval can combine complementary matching signals. However, the benefit of source aggregation depends on the quality and overlap of the generated lists. A relevant

session omitted by all candidate generators cannot be recovered by the subsequent reranking stage.

4.2.3 Cross-Encoder Relevance Estimation

After candidate generation, each retained session is evaluated jointly with the query by a pretrained cross-encoder. For a query q and candidate session $h_i \in \mathcal{C}_K(q)$, the relevance score is defined as

$$e_i(q) = f_{\mathrm{CE}}(q, h_i),$$

where f_{CE} denotes the cross-encoder scoring function.

The implementation uses the model `cross-encoder/ms-marco-MiniLM-L-6-v2`. Unlike a retrieval method that compares independently generated query and session embeddings, the cross-encoder jointly processes the query–candidate pair. Its output therefore provides a query-conditioned relevance signal that can refine the ordering of the retrieved candidates.

For a candidate pool containing $K_q = |\mathcal{C}_K(q)|$ sessions, the cross-encoder evaluates the pairs

$$\mathcal{P}(q) = \{(q, h_i) : h_i \in \mathcal{C}_K(q)\}.$$

The number of cross-encoder evaluations per query is consequently

$$N_{\mathrm{CE}}(q) = K_q \leq K.$$

Across a set of N queries, the total number of candidate-pair evaluations is

$$N_{\mathrm{CE,total}} = \sum_{i=1}^{N} K_{q_i} \leq NK.$$

For $N = 500$ and $K = 20$, this gives an upper bound of $10{,}000$ candidate-pair evaluations if every query is processed once and all candidate pools contain 20 sessions. This is a theoretical bound derived from the configured pool size, not a measured inference-time result. It also does not include the computational cost of the candidate-generation methods.

4.2.4 Score Fusion

The cross-encoder score is combined with a normalized hybrid-retrieval feature. Let

$$e_i = e(q, h_i) \qquad \text{and} \qquad \tilde{h}_i = \tilde{h}(q, h_i)$$

denote the cross-encoder score and normalized hybrid-retrieval feature, respectively, for candidate h_i.

The final RWM score is expressed generally as

$$s_{\mathrm{RWM}}(q, h_i) = \Phi\big(e_i, \tilde{h}_i; \alpha\big),$$

where Φ is the fusion function implemented in the code and α is its configuration parameter.

The reported validation configuration uses

$$\alpha^* = 0.20, \qquad K^* = 20.$$

The fusion function must be defined consistently with the implementation, including the score-normalization procedure, the contribution of each input signal, and the treatment of missing source scores. The value of α alone is not sufficient to reproduce the ranking. In particular, a weighted-sum equation should only be reported if the implementation actually uses that operation.

The fusion stage provides a mechanism for combining the cross-encoder relevance estimate with the hybrid-retrieval signal. Its effectiveness depends on the relative scales of the input features and on the validation procedure used to select the configuration.

4.2.5 Final Ranking and Evidence Retrieval

Once the fused score has been computed, the candidate sessions are sorted in descending order:

$$\pi_q^{\mathrm{RWM}} = \mathrm{argsort}_{h_i \in \mathcal{C}_K(q)}[-s_{\mathrm{RWM}}(q, h_i)].$$

The top-k retrieved sessions are given by

$$\mathcal{R}_k(q) = \mathrm{TopK}\big(\pi_q^{\mathrm{RWM}}\big),$$

where $k = 5$ for the reported retrieval evaluation. The resulting set $\mathcal{R}_5(q)$ contains the five highest-ranked sessions, or fewer when the candidate pool contains fewer than five unique sessions.

Let $\mathcal{G}_q$ denote the reference evidence associated with query q. The retrieval objective is to maximize the amount of relevant evidence recovered near the top of the ranking. More generally, the ranking quality can be represented by

$$\mathcal{J}_{\mathrm{retrieval}} = \frac{1}{N}\sum_{i=1}^{N} m\left(\mathcal{R}_5(q_i), \mathcal{G}_{q_i}\right),$$

where $m(\cdot,\cdot)$ denotes an evaluation function defined by the benchmark protocol. The reported evaluation uses EvidenceRecall@5, Hit@5, StrictRecall@5, and mean reciprocal rank (MRR), with their precise definitions determined by the evaluation implementation and benchmark annotations.

4.2.6 Validation-Based Parameter Selection

The candidate-pool size and fusion parameter are selected using validation data rather than the held-out test set. Let $\mathcal{D}_{\text{val}}$ denote the validation dataset and let Θ denote the set of candidate configurations examined during validation.

For a selected validation metric M_{val}, the parameter-selection procedure can be written as

$$(\alpha^*, K^*) = \arg\max_{(\alpha,K)\in\Theta} M_{\text{val}}(\alpha, K).$$

This expression describes the general selection principle. The precise objective, candidate grid, and tie-breaking procedure must match the implemented validation code. The reported configuration is $\alpha^* = 0.20$ and $K^* = 20$.

After configuration selection, the parameters are fixed and the system is evaluated on the held-out test set. This separation reduces the risk of overestimating performance through repeated tuning on the final evaluation questions. The test set is used for final assessment rather than for choosing the ranking configuration.

4.2.7 Computational Considerations

The retrieval pipeline has two principal computational components: candidate generation and cross-encoder reranking. Let $T_{\text{gen}}(q)$ denote the time required to generate and aggregate candidates for query q, and let $t_{\text{CE}}(q, h_i)$ denote the time required to score an individual query–candidate pair.

The total retrieval time for one query can be represented as

$$T_{\text{RWM}}(q) = T_{\text{gen}}(q) + \sum_{h_i\in\mathcal{C}_K(q)} t_{\text{CE}}\,(q, h_i) + T_{\text{rank}}(q),$$

where $T_{\text{rank}}(q)$ denotes the time required for score fusion and final sorting. This expression describes the components of the runtime; it does not establish a measured latency value.

The candidate-pool restriction bounds the number of cross-encoder evaluations per query by K. Nevertheless, the actual runtime depends on the retrieval implementation, sequence lengths, batch size, hardware, software versions, and model-inference settings. Since measured component-wise latency is not specified in the current experimental results, no quantitative speedup or end-to-end latency improvement is claimed.

4.2.8 Analytical Scope

The formulation above describes the retrieval and reranking mechanism evaluated in the current experiments. It does not define a complete persistent-memory management algorithm. In particular, memory admission, utility-based retention, eviction, consolidation, contradiction resolution, and feedback-driven memory evolution are not represented as validated optimization components in the present pipeline.

These mechanisms may be incorporated into a broader RWM architecture, but doing so requires explicit state representations, update rules, objective functions, and separate experimental evaluation. The analytical contribution assessed here is limited to the integration of multi-source candidate generation, cross-encoder relevance estimation, and implementation-dependent score fusion for conversational evidence retrieval.

4.3 Proposed Algorithms

The experimentally evaluated Remember What Matters (RWM) algorithm consists of three principal operations: multi-source candidate generation, cross-encoder-based reranking, and final evidence ranking. The algorithm combines complementary retrieval signals while limiting expensive cross-encoder inference to a bounded candidate pool.

Algorithm 1: Multi-Source Candidate Generation

For an input query q, three retrieval mechanisms independently produce ranked lists: hybrid retrieval, BM25, and semantic similarity. The candidate lists are merged, duplicate sessions are removed, and the resulting candidates are restricted to the configured pool size.

Let $\mathcal{C}_K(q)$ denote the final candidate pool. In the selected configuration, $K = 20$. Candidate generation aims to improve evidence coverage by combining different retrieval signals. It cannot guarantee that all relevant evidence will be retrieved, because relevant sessions omitted by every source cannot be recovered during reranking.

Algorithm 2: Cross-Encoder Reranking

For each candidate session $h_i \in \mathcal{C}_K(q)$, the cross-encoder computes a relevance score

$$e_i = f_{\mathrm{CE}}(q, h_i).$$

RWM combines this score with a normalized hybrid-retrieval feature according to the fusion function implemented by the system:

$$s_i = \Phi(e_i, \tilde{h}_i; \alpha).$$

The parameter α controls the fusion, and the selected validation configuration uses $\alpha = 0.20$. The implementation's exact fusion equation and normalization must be reported to make the algorithm fully reproducible.

The candidates are sorted in descending order of s_i. This stage refines the order of the retrieved sessions without requiring the cross-encoder to score every session in the complete repository.

Integrated RWM Procedure

Algorithm [alg:rwm] summarizes the evaluated retrieval pipeline.

Query q, session repository $\mathcal{H}$, candidate-pool size K, fusion parameter α Ranked candidate list $\mathcal{L}_{\mathrm{RWM}}(q)$

Retrieve ranked candidates using hybrid retrieval Retrieve ranked candidates using BM25 Retrieve ranked candidates using semantic similarity Merge the candidate lists and remove duplicate sessions Construct candidate pool $\mathcal{C}_K(q)$ using the configured aggregation rule

Compute cross-encoder score $e_i = f_{\mathrm{CE}}(q, h_i)$ Obtain the normalized hybrid feature $\tilde{h}_i$ Compute fused score $s_i = \Phi(e_i, \tilde{h}_i; \alpha)$

Sort candidates in descending order of s_i ranked candidate list $\mathcal{L}_{\mathrm{RWM}}(q)$

The selected parameters are $K = 20$ and $\alpha = 0.20$, determined using the validation procedure. The algorithm describes the implementation responsible for the reported retrieval results. It does not include persistent-memory admission, eviction, consolidation, or online feedback learning, because the current experiment does not independently establish the effectiveness of those operations.

4.4 Computational Complexity

Let N denote the number of stored sessions, K the number of candidates submitted to the cross-encoder, and d the dimension of the vector representations used by semantic retrieval. The computational cost of RWM consists of candidate generation, candidate aggregation, cross-encoder inference, and final sorting.

For dense vector representations, exhaustive query-to-session similarity computation has time complexity $O(Nd)$. This bound applies to exhaustive semantic scoring; indexed retrieval may have a different cost depending on the index and search configuration. BM25 and hybrid retrieval have additional costs determined by their indexing structures and implementation.

If the retrieval sources return a total of L candidate entries before duplicate removal, candidate merging and deduplication can be implemented in expected $O(L)$ time using a hash-based representation, excluding the cost of any additional candidate-aggregation operations. Sorting the final K candidates requires $O(K\log K)$ time.

Let C_{CE} denote the cost of one cross-encoder inference for a query-session pair under the configured sequence lengths and hardware. Reranking K candidates requires approximately

$$O(KC_{\mathrm{CE}})$$

in inference-cost notation. The value of C_{CE} depends on model architecture, input length, batching, and hardware; it should be measured rather than treated as a constant independent of these factors.

The overall cost can therefore be expressed as

$$T_{\mathrm{RWM}} = T_{\mathrm{retrieval}} + T_{\mathrm{aggregation}} + O(KC_{\mathrm{CE}}) + O(K\log K),$$

where $T_{\mathrm{retrieval}}$ includes the cost of the configured hybrid, BM25, and semantic retrieval systems. This expression separates the cost of searching the repository from the additional cost of reranking a bounded candidate pool.

The principal computational trade-off is between candidate coverage and reranking cost. Increasing K can expose the cross-encoder to additional potentially relevant sessions, but also increases inference time. A smaller pool reduces reranking cost but may exclude relevant evidence before the cross-encoder is applied. The selected pool size should therefore be justified by validation performance and runtime measurements.

This analysis concerns retrieval and reranking only. It does not establish the complexity of a complete persistent-memory management system involving online memory updates, conflict detection, consolidation, or eviction.

4.5 Design and Evaluation Considerations

The RWM evaluation investigates whether multi-source candidate generation followed by cross-encoder reranking improves conversational evidence retrieval compared with established retrieval baselines. The experiments use the LongMemEval-S cleaned dataset and evaluate the ranked historical sessions against the dataset's supporting evidence.

The dataset contains 500 questions and 23,854 historical sessions. The evaluation pipeline successfully extracted the session text for all questions. Across the complete experiment, 22,932 query-session pairs were scored by the cross-encoder. The reported final test comparison uses 200 questions. The remaining data are used according to the experiment's data-partition protocol, including validation-based parameter selection. The final manuscript should document the exact partition sizes and ensure that no test labels or supporting answers were used to select the hyperparameters.

The selected validation configuration uses a candidate-pool size of $K = 20$ and a fusion parameter of $\alpha = 0.20$. The evaluated baselines include Hybrid, BM25, semantic Similarity, Recency, reciprocal-rank fusion (RRF), maximal marginal relevance (MMR), and earlier RWM variants. Comparisons with Hybrid should use the matched candidate-pool protocol wherever applicable, so that observed differences can be attributed more clearly to reranking rather than to an unequal candidate budget.

Performance is assessed using EvidenceRecall@5, Hit@5, StrictRecall@5, and mean reciprocal rank (MRR). EvidenceRecall@5 measures the amount of supporting evidence recovered within the top five results under the dataset's evidence-matching protocol. Hit@5 measures whether at least one required evidence item is found within the top five. StrictRecall@5 applies the experiment's stricter evidence-retrieval criterion, while MRR measures the reciprocal rank of the first qualifying result. The exact operational definitions and aggregation procedures for these metrics should be specified in the evaluation protocol.

On the 200-question test set, RWM obtains EvidenceRecall@5 of 0.9684, compared with 0.9568 for Hybrid. RWM achieves Hit@5 of 1.000, compared with 0.995 for Hybrid; StrictRecall@5 of 0.930, compared with 0.910; and MRR of 0.9395, compared with 0.9368.

The absolute improvement in EvidenceRecall@5 over Hybrid is approximately 0.0117, equivalent to 1.17 percentage points. A bootstrap analysis of the difference reports a mean improvement of approximately 0.0117 and a 95% confidence interval of $[0.0025, 0.0225]$. Because this interval is above zero under the reported bootstrap procedure, the result supports a positive difference on this test sample. It does not establish universal superiority across datasets, model configurations, or interaction distributions.

The reported results should be interpreted in light of the candidate-generation stage. Reranking can improve the ordering of retrieved sessions, but it cannot recover supporting evidence that is absent from the candidate pool. Consequently, future evaluation should report candidate-pool evidence coverage separately from final top-k ranking performance. This distinction helps determine whether further improvements should target retrieval diversity, candidate-pool size, score fusion, or the cross-encoder itself.

For a reproducible comparison, all methods should use the same dataset partitions and evidence-matching protocol. Hyperparameters must be selected using validation data, and test results should be used only for final reporting. Where stochastic components are present, repeated runs or paired statistical comparisons should be considered. The bootstrap resampling unit, number of resamples, and confidence-interval construction should also be reported.

The current experiment validates a retrieval and reranking contribution. It does not, by itself, establish that RWM reduces persistent-memory interference or optimally manages memory admission, retention, conflict resolution, and eviction. Those broader claims require experiments that explicitly implement and evaluate the corresponding operations under matched storage and context budgets.

The central empirical finding is that RWM improves the measured retrieval metrics over the Hybrid baseline on the reported test set, with a positive bootstrap confidence interval for EvidenceRecall@5. Further work should test the robustness of this improvement on additional datasets, alternative query distributions, different candidate-pool sizes, and different cross-encoder configurations. It should also measure end-to-end latency and computational overhead to establish the practical trade-off between retrieval quality and inference cost.

5 Results and Interpretation

5.1 Experimental Environment

The experimental environment is designed to evaluate Remember What Matters (RWM) for conversational memory retrieval. The experiments focus on two main stages: multi-source

candidate generation and cross-encoder-based reranking. The objective is to assess whether reranking a restricted candidate pool improves the retrieval of relevant historical conversational sessions compared with conventional retrieval methods. The evaluation focuses on evidence retrieval rather than answer generation or the complete lifecycle of persistent conversational memory.

The experiments are implemented in Python and executed in the Google Colab environment. Python is used to implement the retrieval pipeline, candidate aggregation, cross-encoder scoring, score fusion, and evaluation procedures. The cross-encoder model `cross-encoder/ms-marco-MiniLM-L-6-v2` is used for candidate reranking. Google Colab provides the computational environment for executing the experiments and processing the dataset. The software versions, hardware specifications, and accelerator configuration should be reported according to the actual execution environment to ensure reproducibility.

The experiments use a cleaned version of the LongMemEval-S dataset containing 500 questions and 23,854 historical conversational sessions. Each evaluation instance consists of a query, its available historical sessions, and the corresponding reference evidence specified by the benchmark annotations. The dataset can be represented as

$$\mathcal{D} = \{(q_i, \mathcal{H}_i, \mathcal{G}_i)\}_{i=1}^{N},$$

where q_i denotes the current question, $\mathcal{H}_i$ denotes the historical sessions available for retrieval, $\mathcal{G}_i$ denotes the associated reference evidence, and $N = 500$ is the total number of questions. A held-out test set of 200 questions is used for final evaluation, while configuration parameters are selected using validation data.

RWM follows a two-stage retrieval pipeline. First, candidate sessions are collected from three retrieval sources: hybrid retrieval, BM25, and semantic similarity. Let $\mathcal{C}_{\mathrm{hyb}}(q)$, $\mathcal{C}_{\mathrm{BM25}}(q)$, and $\mathcal{C}_{\mathrm{sim}}(q)$ denote the candidate sets generated for query q. These sets are aggregated and restricted to a candidate pool according to the implemented retrieval procedure:

$$\mathcal{C}_K(q) = \mathrm{TopK}\big(\mathcal{C}_{\mathrm{hyb}}(q) \cup \mathcal{C}_{\mathrm{BM25}}(q) \cup \mathcal{C}_{\mathrm{sim}}(q)\big),$$

where $K = 20$ is the selected candidate-pool size. The TopK operation follows the candidate ordering and aggregation procedure implemented in the code.

Each candidate is then scored using the pretrained cross-encoder `cross-encoder/ms-marco-MiniLM-L-6-v2`, which jointly processes the query and candidate text to estimate their relevance. The cross-encoder signal is combined with a normalized hybrid-retrieval feature to obtain the final candidate ranking. The fusion parameter is set to $\alpha = 0.20$, selected using validation data. The exact fusion equation, score normalization, and role of α are defined by the implemented scoring function and must be reported consistently with the code to ensure reproducibility.

The resulting ranked sessions are evaluated at a cutoff of $k = 5$ using EvidenceRecall@5, Hit@5, StrictRecall@5, and mean reciprocal rank (MRR). These metrics characterize evidence retrieval coverage, successful retrieval, performance under the benchmark's strict evidence-recall criterion, and the ranking position of the first relevant result, respectively. Metric computation follows the benchmark annotations and the evaluation implementation.

The evaluated baseline methods include Recency, semantic Similarity, BM25, Hybrid, Reciprocal Rank Fusion (RRF), and Maximal Marginal Relevance (MMR). Earlier RWM variants are also included where applicable and are distinguished from the final configuration. Comparisons should use the same evaluation questions and compatible evidence annotations. In particular, the RWM–Hybrid comparison provides a reference for evaluating the contribution of the proposed retrieval and reranking procedure relative to hybrid retrieval. Any comparison involving different candidate pools or evaluation protocols must be identified separately.

The evaluation proceeds by generating candidates, selecting the configuration using validation data, and applying the selected configuration to the held-out test set. The candidate-pool size and fusion parameter are not selected using test-set outcomes. Retrieval rankings and question-level evaluation outcomes are used to calculate aggregate metrics. Bootstrap analysis is used to estimate uncertainty in the EvidenceRecall@5 difference between RWM and the Hybrid baseline.

The experimental scope is limited to the retrieval and reranking of historical conversational evidence. Memory admission, persistent-memory eviction, memory consolidation, contradiction resolution, and feedback-driven utility adaptation are not evaluated as components of this pipeline. Similarly, the experiments do not establish end-to-end answer-generation latency, persistent-memory growth, or online adaptation performance. These aspects require separate experimental protocols and remain outside the scope of the current evaluation.

5.2 Simulation Parameters

The RWM retrieval pipeline is evaluated using the cleaned LongMemEval-S dataset. The experimental configuration includes the dataset settings, candidate-generation strategy, candidate-pool size, cross-encoder model, fusion parameter, and evaluation cutoff. Table 1 summarizes the principal parameters used in the implementation.

Simulation parameters of the RWM retrieval pipeline.

Parameter	**Symbol**	**Setting**
Dataset	–	Cleaned LongMemEval-S
Total number of questions	N	500
Number of historical sessions	–	23,854
Test-set size	N_{test}	200

Parameter	Symbol	Setting
Candidate-generation methods	–	Hybrid, BM25, Similarity
Candidate aggregation	–	Union of retrieved candidates
Candidate-pool size	K	20
Cross-encoder model	–	`ms-marco-MiniLM-L-6-v2`
Fusion parameter	α	0.20
Parameter selection	–	Validation-based
Final ranking cutoff	k	5
Evaluation metrics	–	EvidenceRecall@5, Hit@5, StrictRecall@5, MRR

The candidate-generation stage combines the results obtained from Hybrid retrieval, BM25, and semantic similarity retrieval. The resulting candidate union is restricted to a pool of $K = 20$ candidates before cross-encoder scoring. The cross-encoder, `ms-marco-MiniLM-L-6-v2`, evaluates the query–candidate pairs to support the final ranking process.

The fusion parameter is set to $\alpha = 0.20$, based on the validation procedure. This value is an experimental configuration choice rather than a universally optimal setting. Its interpretation depends on the precise fusion function implemented in the code, including the score-normalization procedure and the contribution of each scoring component.

The final ranked list is evaluated at a cutoff of $k = 5$ using EvidenceRecall@5, Hit@5, StrictRecall@5, and mean reciprocal rank (MRR). These metrics characterize evidence coverage, successful retrieval, strict evidence-recall performance, and the ranking position of the first relevant result, respectively. Their exact computation follows the evaluation implementation and the evidence annotations provided by the benchmark.

The configuration separates validation-based parameter selection from final test-set evaluation. The reported setup focuses on candidate retrieval and reranking; it does not specify a persistent-memory budget, an eviction policy, or measured end-to-end latency.

5.3 Results and Interpretation

This section evaluates the retrieval performance of the proposed Remember What Matters (RWM) pipeline. The analysis first examines the held-out test-set performance of RWM against the Hybrid baseline. It then compares RWM with a broader set of retrieval methods and earlier RWM variants, investigates the effect of the candidate-pool size, and quantifies the uncertainty of the observed RWM–Hybrid difference through bootstrap analysis. Finally, the results are interpreted in relation to the retrieval mechanism, the preliminary evaluation, and the limitations of the current experimental scope. Unless otherwise specified, all retrieval metrics are computed at a ranking cutoff of $k = 5$.

5.3.1 Quantitative Results on the Held-Out Test Set

The final evaluation investigates whether multi-source candidate generation combined with cross-encoder-based reranking improves conversational evidence retrieval. The experiments use the cleaned LongMemEval-S dataset, which contains 500 questions and 23,854 historical conversational sessions. The selected RWM configuration uses a candidate pool of $K = 20$ sessions and a fusion parameter of $\alpha = 0.20$, selected using validation data. Final performance is reported on a held-out test set of $N_{\text{test}} = 200$ questions.

Four complementary metrics are considered: EvidenceRecall, Hit, StrictRecall, and mean reciprocal rank (MRR). EvidenceRecall measures the extent to which relevant evidence is retrieved, whereas Hit indicates whether relevant evidence is found within the top-k results. StrictRecall evaluates retrieval under the stricter relevance criterion used in the experimental implementation. MRR measures the reciprocal rank of the first relevant result, averaged across the evaluated queries.

Table 2 presents the test-set results for RWM and Hybrid.

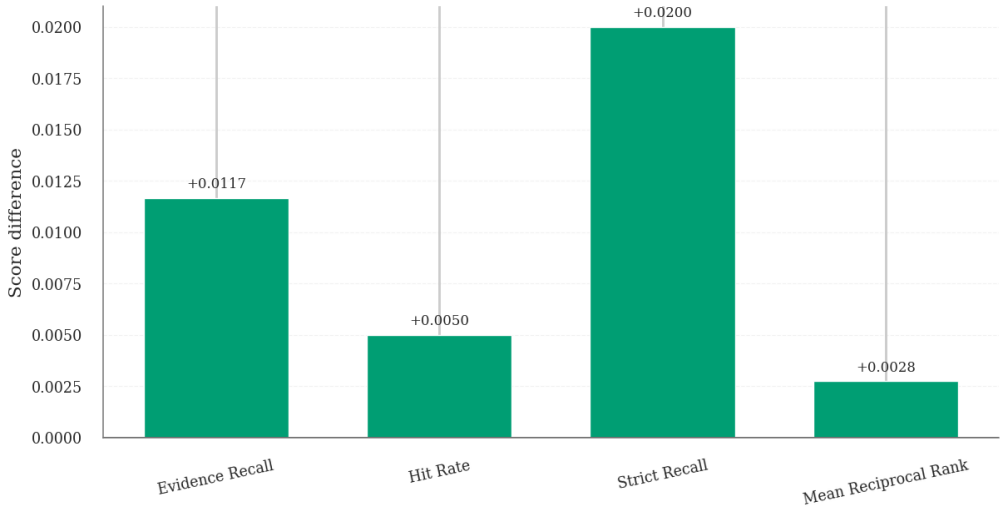
RWM and Hybrid performance on the held-out LongMemEval-S test set ($N_{\text{test}} = 200$, $k = 5$).

Metric	RWM	Hybrid
EvidenceRecall	**0.9684**	0.9568
Hit	**1.0000**	0.9950
StrictRecall	**0.9300**	0.9100
MRR	**0.9395**	0.9368

RWM obtains an EvidenceRecall of 0.9684, compared with 0.9568 for Hybrid, corresponding to an absolute improvement of approximately 1.17 percentage points. RWM also achieves a Hit value of 1.0000, compared with 0.9950 for Hybrid. StrictRecall increases from 0.9100 to 0.9300, an improvement of 2.00 percentage points, while MRR increases from 0.9368 to 0.9395.

These results indicate that RWM performs better than Hybrid on all four reported metrics under the evaluated test protocol. The largest absolute differences occur in EvidenceRecall and StrictRecall, whereas the difference in MRR is comparatively small. The Hit value of 1.0000 indicates that every test query satisfies the Hit criterion at the top-five cutoff under the implementation used in this experiment. It does not, by itself, imply that every relevant evidence item has been retrieved.

Figure 4 provides a visual comparison of the two methods across the four metrics.



Comparison of RWM and Hybrid on the held-out LongMemEval-S test set using EvidenceRecall, Hit, StrictRecall, and MRR at $k = 5$. Higher values indicate better performance for the reported metrics.

5.3.2 Comparison with Multiple Retrieval Baselines

To assess the effectiveness of the proposed Remember What Matters (RWM) retrieval pipeline beyond the direct comparison with Hybrid, we evaluate its performance against a broader set of retrieval strategies. The comparison includes Recency, semantic Similarity, BM25, Hybrid retrieval, Reciprocal Rank Fusion (RRF), Maximal Marginal Relevance (MMR), and three earlier RWM implementations: RWM (original), RWM_v2, and RWM_v3. These methods represent different retrieval principles, ranging from temporal prioritisation and lexical matching to semantic retrieval, ranking fusion, and successive versions of the proposed approach.

Table [tab:rwm_all_methods] summarises the available evaluation results using EvidenceRecall, Hit, StrictRecall, and mean reciprocal rank (MRR), all computed at $k = 5$. Considering several complementary metrics provides a broader assessment than relying on a single measure. EvidenceRecall characterises the recovery of relevant evidence, Hit indicates whether relevant evidence is retrieved within the top-five results, StrictRecall reflects performance under the strict relevance criterion implemented in the evaluation, and MRR captures how highly the first relevant result is ranked.

Method	EvidenceRecall	Hit	StrictRecall	MRR
RWM (new)	**0.9684**	**1.000**	**0.930**	**0.9395**
Hybrid	0.9568	0.995	0.910	0.9368
RWM_v3	0.9466	0.985	0.895	0.9265
RWM_v2	0.9463	0.980	0.895	0.9271
RRF	0.9433	0.980	0.890	0.9222

Method	EvidenceRecall	Hit	StrictRecall	MRR
BM25	0.9178	0.980	0.830	0.9234
RWM (original)	0.8968	0.975	0.805	0.8746
Similarity	0.8873	0.955	0.810	0.8656
MMR	0.7252	0.925	0.525	0.8549
Recency	0.1663	0.295	0.070	0.1756

The results in Table [tab:rwm_all_methods] show that the new RWM configuration achieves the highest reported value for each of the four metrics. Its EvidenceRecall reaches 0.9684, compared with 0.9568 for Hybrid, 0.9466 for RWM_v3, and 0.9463 for RWM_v2. This corresponds to an absolute EvidenceRecall improvement of approximately 1.17 percentage points over Hybrid and approximately 2.18 percentage points over RWM_v3. The result suggests that the latest configuration improves the recovery of relevant evidence relative to the methods included in this comparison.

The same pattern is observed for StrictRecall. RWM obtains 0.930, compared with 0.910 for Hybrid and 0.895 for both RWM_v2 and RWM_v3. This difference indicates that the advantage of the new configuration is not limited to the broader EvidenceRecall measure; it is also present under the stricter relevance criterion used in the evaluation. RWM additionally reaches a Hit value of 1.000, meaning that every evaluated query satisfies the implemented Hit criterion within the top-five results. This should not be interpreted as evidence that every relevant item is retrieved, since Hit and EvidenceRecall measure different aspects of retrieval.

The MRR values provide further information about ranking quality. RWM obtains an MRR of 0.9395, compared with 0.9368 for Hybrid, 0.9271 for RWM_v2, and 0.9265 for RWM_v3. The difference between RWM and Hybrid is relatively small for this metric, suggesting that both methods already tend to place a relevant result near the top of the ranking. The stronger differences in EvidenceRecall and StrictRecall indicate that the advantage of RWM is more pronounced in evidence recovery than in the reciprocal-rank measure.

The comparison also reveals substantial differences between the retrieval strategies. BM25 achieves an EvidenceRecall of 0.9178, while semantic Similarity reaches 0.8873. RRF obtains 0.9433, placing it among the stronger baselines in the reported comparison. By contrast, MMR records an EvidenceRecall of 0.7252 and a StrictRecall of 0.525. Recency performs substantially worse, with an EvidenceRecall of 0.1663 and an MRR of 0.1756. These results suggest that temporal proximity alone is insufficient for this conversational evidence retrieval task. However, the comparison does not establish that one general retrieval principle is universally superior, since performance depends on the dataset, relevance definitions, implementation, and evaluation protocol.

Figure 5 complements Table [tab:rwm_all_methods] by providing a visual comparison of the evaluated methods across the four metrics. It should make the relative proximity of RWM, Hybrid, RWM_v2, RWM_v3, and RRF easier to inspect, while also showing the larger

performance gap between these methods and Recency. The figure distinguishes the new RWM configuration from the original implementation, making the progression across the reported variants explicit.

Comparison of RWM and baseline retrieval strategies using EvidenceRecall, Hit, StrictRecall, and MRR at $k = 5$. The new RWM configuration and the original RWM implementation are shown as separate methods.

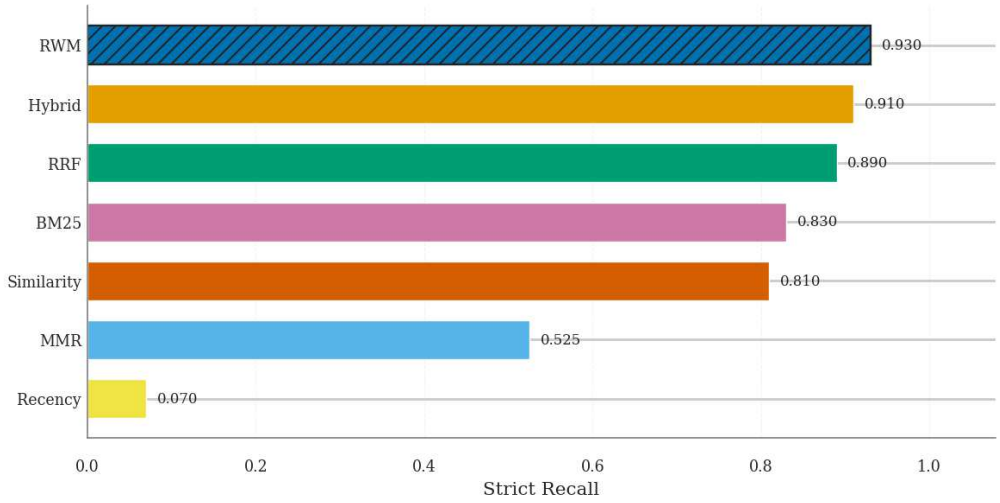
Taken together, the table and figure indicate that the latest RWM configuration achieves the strongest performance among the methods listed in the available comparison. Nevertheless, the interpretation of this result depends on the comparability of the underlying experiments. In particular, the methods should be considered a strict head-to-head comparison only if they were evaluated using the same test queries, relevance annotations, metric definitions, and candidate-generation constraints. Subject to this qualification, the results provide empirical support for further investigating multi-source retrieval and query-conditioned reranking as components of conversational evidence retrieval.

5.3.3 Effect of Candidate-Pool Size

The candidate-pool size K determines how many candidate sessions are passed from the candidate-generation stage to the cross-encoder reranking stage. This parameter is important because the reranker cannot recover relevant evidence that was excluded during candidate generation. Increasing K may therefore improve evidence coverage by allowing more candidate sessions to be considered. However, a larger candidate pool also increases the number of candidate pairs that must be scored by the cross-encoder.

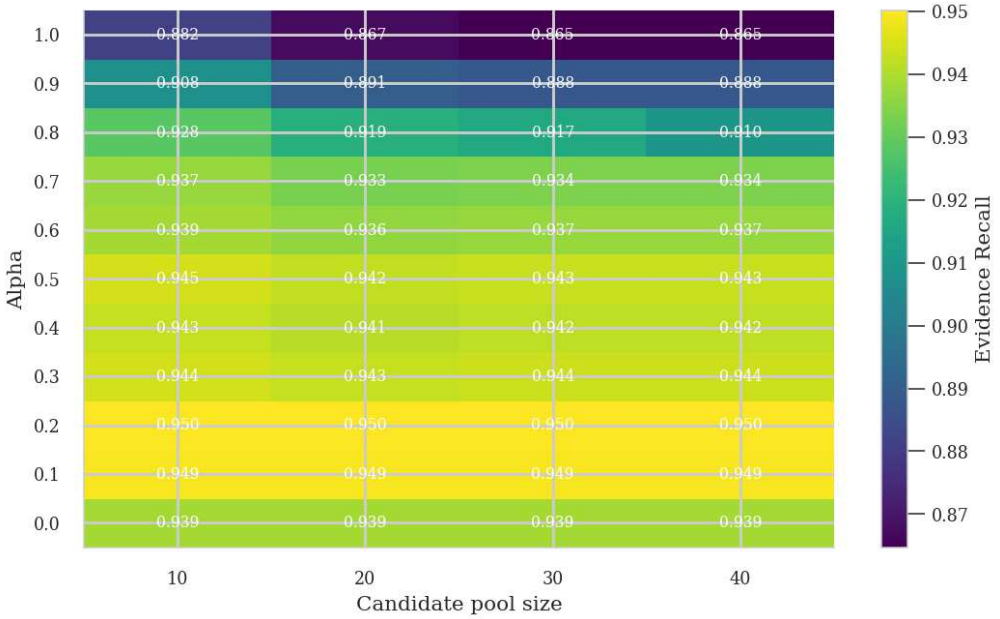
The choice of K consequently involves a trade-off between retrieval quality and computational requirements. A small pool reduces the number of candidates to be processed but may exclude relevant sessions. A larger pool can improve the opportunity to retrieve relevant evidence, although the additional candidates may provide diminishing returns. The purpose of this analysis is to examine how the retrieval metrics vary with K and to assess the suitability of the selected configuration, $K = 20$.

Figures 6–11 present the available candidate-pool analysis. Specifically, Figure 6 reports StrictRecall, Figure 7 presents the candidate-pool-size analysis, Figure 8 shows Recall, Figure 9 reports EvidenceRecall, Figure 10 presents MRR, and Figure 11 reports the Hit rate. Examining these metrics jointly is important because changes in candidate-pool size may affect evidence coverage and ranking quality in different ways.



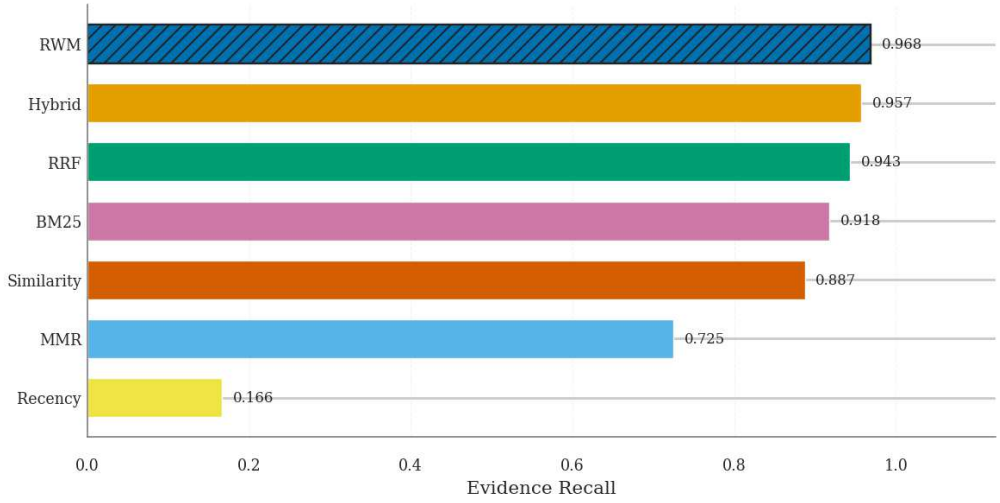
StrictRecall as a function of candidate-pool size K for the evaluated methods.

Figure 6 focuses on StrictRecall, which provides a stricter view of retrieval performance than a metric that only checks whether at least one relevant result appears in the top five. The evolution of this metric across different values of K helps determine whether increasing the candidate pool improves the retrieval of evidence satisfying the strict relevance criterion. If StrictRecall increases with K, the additional candidates may be helping the pipeline retain or recover more relevant evidence. If the curve becomes approximately flat, further increases in the candidate pool may provide limited additional benefit under this metric.



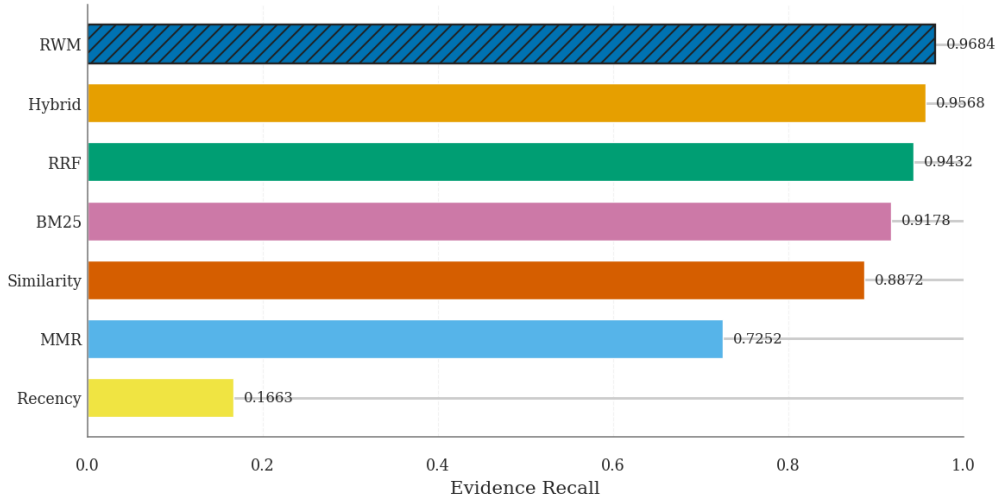
Candidate-pool-size analysis for the evaluated configurations.

Figure 7 should be interpreted alongside the retrieval-quality curves to clarify the role of K in the experimental configuration. The number of candidates determines the size of the set submitted to the reranker, whereas the quality metrics indicate whether considering additional candidates improves the final ranking. The candidate count alone is not a retrieval performance measure; its value lies in helping relate the pool size to the quality obtained from the subsequent reranking stage.



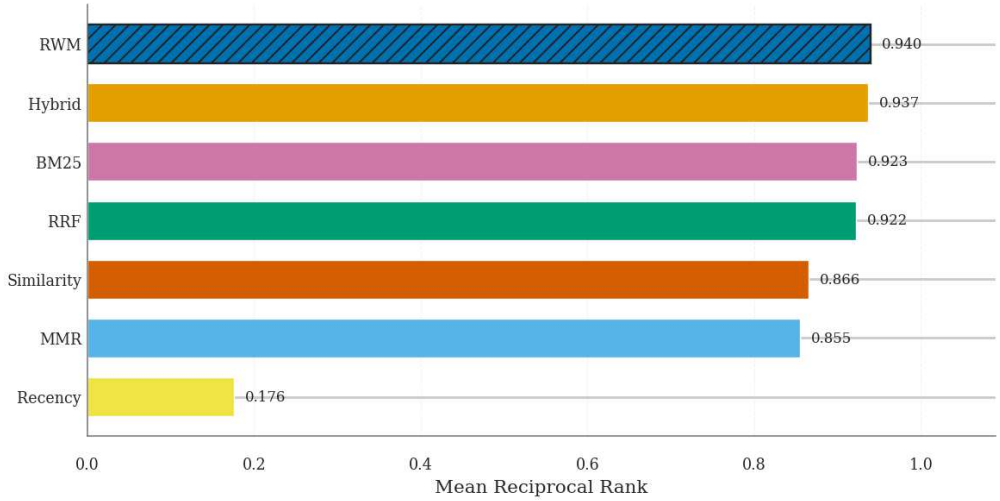
Recall as a function of candidate-pool size K for the evaluated methods.

As shown in Figure 8, Recall provides a complementary view of the effect of candidate-pool size on the retrieval of relevant items. Its evolution can help distinguish between a configuration that retrieves at least some relevant evidence and one that recovers a larger proportion of the relevant evidence. The precise interpretation depends on the definition of Recall used in the implementation and should remain consistent with the metric reported in the experimental results.



EvidenceRecall as a function of candidate-pool size K for the evaluated methods.

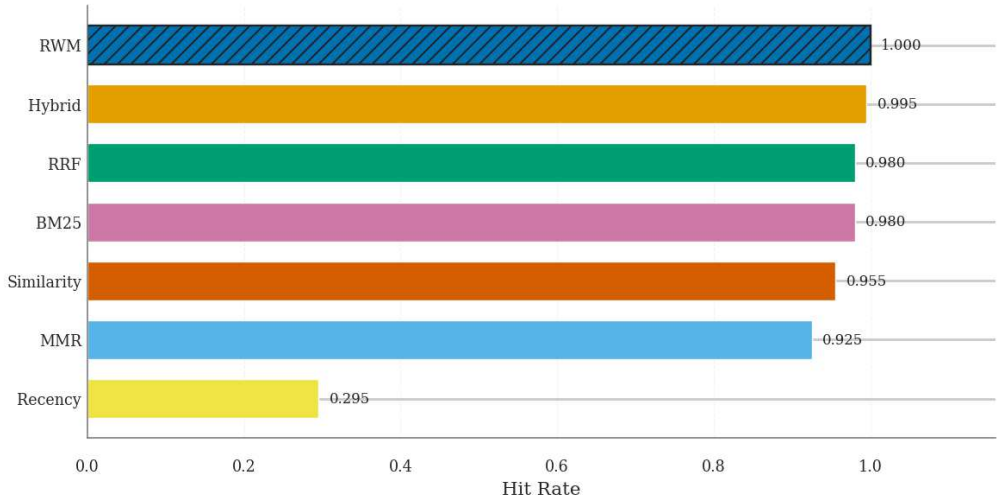
Figure 9 is particularly relevant to the objective of RWM, since EvidenceRecall directly evaluates the recovery of relevant conversational evidence according to the experimental definition. An increase in EvidenceRecall as K grows would indicate that the larger candidate pool enables the pipeline to retrieve additional relevant evidence. Conversely, a plateau would suggest that increasing the number of candidates does not substantially improve evidence recovery over the examined range of K. The plotted values, rather than the selected configuration alone, must determine which pattern is observed.



Mean reciprocal rank (MRR) as a function of candidate-pool size K for the evaluated methods.

Figure 10 examines the effect of candidate-pool size on the ordering of relevant results. Unlike a coverage-oriented metric, MRR gives greater importance to the rank of the first relevant result. Consequently, an increase in EvidenceRecall does not necessarily imply a corresponding increase in MRR: additional candidates may improve evidence coverage without moving the first relevant result closer to the top of the final ranking. Considering MRR alongside EvidenceRecall therefore helps distinguish improvements in evidence recovery from improvements in the position of the first relevant candidate.

Finally, Figure 11 reports Hit rate across the candidate-pool settings. This metric indicates the proportion of queries for which the implemented relevance criterion is satisfied within the top-five results. If Hit approaches its maximum at relatively small values of K, increasing the candidate pool may have limited effect on this particular metric, even if EvidenceRecall or StrictRecall continues to change. Such a pattern would indicate that retrieving at least one relevant result and recovering all relevant evidence are distinct objectives.



Hit rate as a function of candidate-pool size K for the evaluated methods.

The six figures should therefore be interpreted together rather than individually. StrictRecall and EvidenceRecall characterise evidence retrieval under their respective definitions, Recall provides an additional coverage measure, MRR evaluates the rank of the first relevant result, and Hit measures whether the top-five criterion is satisfied. Their combined behaviour offers a more complete view of how candidate-pool size affects the retrieval pipeline.

The selected value $K = 20$ was chosen using validation data. Its suitability should be assessed by examining whether it provides strong retrieval performance without an unnecessary increase in the number of candidates scored. If the observed curves show that performance gains become small beyond $K = 20$, this would support the choice as a practical operating point for the evaluated setup. However, the existence of such a plateau must be confirmed from the plotted values. Moreover, because execution time, cross-encoder inference cost, and memory consumption were not directly measured, the analysis does not establish that $K = 20$ is computationally optimal. A dedicated runtime and resource-consumption study would be required to quantify that trade-off.

5.3.4 Statistical Analysis of the RWM–Hybrid Difference

A bootstrap analysis was conducted to quantify the uncertainty in the observed EvidenceRecall difference between RWM and Hybrid on the 200-question held-out test set. The estimated mean difference is 0.0116667, equivalent to approximately 1.17 percentage points. The reported 95% bootstrap confidence interval is

$$\mathrm{CI}_{95\%} = [0.0025,\ 0.0225].$$

The confidence interval lies entirely above zero. Under the reported bootstrap procedure, this indicates that the estimated EvidenceRecall difference is positive. The interval corresponds to an improvement ranging from approximately 0.25 to 2.25 percentage points. This result strengthens the evidence for an improvement on the evaluated test set,

but it should not be interpreted as a guarantee of the same effect on other datasets or evaluation settings.

The interpretation also depends on the bootstrap implementation, including the sampling unit and resampling procedure. Here, the analysis is described as question-level resampling; this description should be retained only if it matches the actual implementation.

Figure 12 should display the point estimate and its confidence interval relative to a zero-difference reference line. The corresponding figure file was not specified in the supplied code, so its path must be added before compilation.

Bootstrap estimate of the EvidenceRecall difference between RWM and Hybrid on the held-out test set. The plot shows the estimated difference and its 95% confidence interval, together with a zero-difference reference line.

5.3.5 Interpretation of the RWM Retrieval Mechanism

The evaluated RWM pipeline combines multi-source candidate generation with cross-encoder-based reranking. Candidate sessions are collected from Hybrid, BM25, and semantic-similarity retrieval, after which the candidate pool is restricted to $K = 20$ sessions. The candidates are scored using the pretrained cross-encoder `cross-encoder/ms-marco-MiniLM-L-6-v2`. A fusion mechanism combines the cross-encoder relevance score with a normalised Hybrid retrieval feature, using $\alpha = 0.20$ as selected on validation data.

The observed performance is consistent with the hypothesis that complementary candidate-generation sources and query-conditioned reranking can improve the ordering of conversational evidence. The reranker can compare each candidate in the context of the current query rather than relying exclusively on the initial ranking produced by an individual retrieval method.

Nevertheless, the results do not establish the independent contribution of every retrieval source. Since the pipeline combines multiple components, controlled ablation experiments would be needed to determine how much of the improvement comes from candidate aggregation, cross-encoder scoring, or score fusion. Likewise, the selected value of α is the best value among those considered during validation, not proof of a globally optimal parameter.

The candidate-pool-size analysis can provide additional insight into the relationship between candidate coverage and reranking. If the reported curves flatten beyond $K = 20$, the observation would be consistent with diminishing retrieval gains from adding more candidates. It would not, on its own, establish that $K = 20$ minimises latency or computational cost, since these quantities were not directly measured.

5.3.6 Evolution from the Preliminary RWM Evaluation

An earlier, preliminary evaluation compared Recency, Similarity, Hybrid, and an initial RWM strategy on the 500-question cleaned LongMemEval-S dataset. In that experiment, the initial RWM obtained a Recall of 0.8839, an MRR of 0.8489, and a StrictRecall of 0.8020. Similarity achieved 0.8702, 0.8406, and 0.7860, respectively, while Hybrid achieved 0.8601, 0.8361, and 0.7640. Recency performed substantially worse, with corresponding values of 0.1358, 0.1047, and 0.0560.

These preliminary results motivated further development of the retrieval pipeline, including multi-source candidate generation and cross-encoder reranking. However, the preliminary evaluation and the latest held-out test evaluation represent distinct experimental stages. Their metrics should not be combined into a single ranking or interpreted as a direct, controlled estimate of improvement unless the dataset split, metric definitions, candidate-generation protocol, and evaluation procedure are confirmed to be comparable.

The latest results are therefore reported separately in Tables 2 and [tab:rwm_all_methods], with the direct RWM–Hybrid comparison illustrated in Figure 4.

5.3.7 Implications and Limitations

The results support the effectiveness of the evaluated RWM retrieval configuration on the selected LongMemEval-S test set. In particular, RWM exceeds Hybrid on the four reported metrics, and the bootstrap confidence interval for the EvidenceRecall difference lies above zero under the reported resampling procedure. The broader comparison also places the new RWM configuration ahead of the listed methods in the available all-methods results.

Several limitations should be acknowledged.

First, the experiments evaluate retrieval quality rather than end-to-end question-answering accuracy. Successful retrieval of relevant evidence does not necessarily guarantee that a downstream language model will produce a correct answer.

Second, the reranking stage is constrained by the candidate-generation stage. Relevant evidence omitted from the candidate pool cannot be recovered by reranking alone.

Third, the candidate pool is limited by the number of sessions, not by an explicitly controlled token budget. Consequently, the results do not establish that all methods operate under equal token-level context constraints.

Fourth, retrieval latency, cross-encoder inference time, computational resource consumption, and storage overhead were not directly evaluated. The candidate-pool-size results should therefore not be presented as direct measurements of efficiency.

Finally, the current experiments do not evaluate the complete persistent-memory lifecycle. Memory admission, updating, consolidation, eviction, temporal-conflict resolution, and feedback-driven memory evolution remain outside the scope of the reported retrieval

evaluation. These components require additional experiments before claims can be made about RWM as a complete adaptive memory system.

5.3.8 Summary

On the held-out test set of 200 questions, the evaluated RWM configuration achieves an EvidenceRecall of 0.9684, a Hit of 1.0000, a StrictRecall of 0.9300, and an MRR of 0.9395. Compared with Hybrid, EvidenceRecall increases by approximately 1.17 percentage points, StrictRecall by 2.00 percentage points, and MRR by approximately 0.00275. The reported 95% bootstrap confidence interval for the EvidenceRecall difference is $[0.0025,\ 0.0225]$.

Taken together, these results provide evidence that multi-source candidate generation and cross-encoder-based reranking improve conversational evidence retrieval under the evaluated experimental conditions. Further work is required to verify the contribution of individual components, assess computational efficiency, and evaluate the broader memory-management mechanisms needed for a complete adaptive persistent-memory system.

6 Conclusion and Future Work

This paper presented *Remember What Matters* (RWM), a conversational memory retrieval framework designed to improve the identification and ranking of relevant evidence in long-term interactions with large language models (LLMs). The proposed approach combines multi-source candidate generation using Hybrid retrieval, BM25, and semantic similarity with cross-encoder-based reranking. By restricting cross-encoder inference to a bounded candidate pool, RWM aims to improve retrieval quality while limiting the number of query–candidate pairs that require detailed relevance scoring.

The framework was evaluated on a cleaned version of the LongMemEval-S dataset containing 500 questions and 23,854 historical conversational sessions. The selected configuration used a candidate pool of 20 sessions and a fusion parameter of $\alpha = 0.20$, determined through validation. On a held-out test set of 200 questions, RWM achieved an EvidenceRecall@5 of 0.9684, a Hit@5 of 1.0000, a StrictRecall@5 of 0.9300, and a mean reciprocal rank (MRR) of 0.9395.

Compared with the Hybrid baseline, RWM improved EvidenceRecall@5 from 0.9568 to 0.9684, corresponding to an absolute gain of approximately 1.17 percentage points. StrictRecall@5 increased from 0.9100 to 0.9300, while MRR improved from 0.9368 to 0.9395. The reported bootstrap analysis estimated an EvidenceRecall@5 difference of 0.0117, with a 95% confidence interval of $[0.0025,\ 0.0225]$. This interval supports a positive difference under the reported evaluation protocol. The results from the broader retrieval comparison also place the new RWM configuration ahead of the evaluated baselines on the reported evidence-retrieval metrics.

These findings support the effectiveness of combining complementary candidate-generation methods with cross-encoder reranking for conversational evidence retrieval under the evaluated conditions. However, the magnitude and generality of the improvement remain dependent on the dataset, candidate-generation procedure, evaluation criteria, and test protocol. The current results do not establish that RWM will achieve the same gains across other benchmarks or deployment environments.

An important distinction is that the experiments validate the retrieval and reranking components rather than the complete memory-management framework originally envisioned for RWM. Persistent-memory admission and eviction, redundancy-aware consolidation, explicit contradiction resolution, temporal-validity management, and feedback-driven utility adaptation have not been jointly validated by the reported experiments. Moreover, the evaluation focuses on evidence retrieval rather than end-to-end answer correctness, persistent-storage consumption, or measured response latency. These capabilities should therefore be considered extensions of the current implementation rather than established experimental outcomes.

Future work will address four main directions. First, component-level ablation studies will investigate the individual contributions of Hybrid retrieval, BM25, semantic similarity, and cross-encoder reranking. Sensitivity analyses will examine the effect of candidate-pool size and the score-fusion parameter on retrieval quality and computational cost. Second, evaluation will be extended to additional conversational-memory benchmarks and, where appropriate, end-to-end question-answering tasks to assess generalization beyond the current test set.

Third, the retrieval pipeline will be evaluated under matched context-token budgets and with explicit measurements of retrieval latency, cross-encoder inference cost, and memory consumption. These experiments will help determine whether improvements in evidence retrieval can be obtained within comparable computational and context constraints. Statistical evaluation will also be strengthened through clearly specified paired resampling procedures and confidence intervals across relevant metrics.

Finally, future development will investigate the integration of retrieval with a complete persistent-memory lifecycle. This extension will consider memory admission, updating, consolidation, eviction, temporal validity, and the handling of contradictory or evolving information. Sequential interaction experiments will be needed to determine whether feedback-driven adaptation and utility-aware retention provide measurable benefits beyond the current retrieval-and-reranking pipeline.

Ultimately, RWM aims to provide a foundation for more effective and resource-aware conversational memory in long-running LLM agents. The present results establish a promising retrieval component, while further controlled experiments are required to determine whether integrating retrieval with persistent-memory management can deliver broader improvements in evidence quality, temporal consistency, computational efficiency, and long-term interaction performance.

99 F. I. M. Craik and R. S. Lockhart, “Levels of Processing: A Framework for Memory Research,” *Journal of Verbal Learning and Verbal Behavior*, vol. 11, no. 6, pp. 671–684, 1972, doi: 10.1016/S0022-5371(72)80001-X.

E. Tulving, *Elements of Episodic Memory*. Oxford, U.K.: Clarendon Press, 1983.

J. L. McGaugh, “Memory—A Century of Consolidation,” *Science*, vol. 287, no. 5451, pp. 248–251, 2000, doi: 10.1126/science.287.5451.248.

J. R. Anderson and L. J. Schooler, “Reflections of the Environment in Memory,” *Psychological Science*, vol. 2, no. 6, pp. 396–408, 1991, doi: 10.1111/j.1467-9280.1991.tb00174.x.

J. S. Park, J. O’Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein, “Generative Agents: Interactive Simulacra of Human Behavior,” in *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST)*, pp. 1–22, 2023, doi: 10.1145/3586183.3606763.

C. Packer, S. Wooders, K. Lin, V. Fang, S. G. Patil, I. Stoica, and J. E. Gonzalez, “MemGPT: Towards LLMs as Operating Systems,” arXiv preprint arXiv:2310.08560, 2023, doi: 10.48550/arXiv.2310.08560.

D. Wu, H. Wang, W. Yu, Y. Zhang, K.-W. Chang, and D. Yu, “LongMemEval: Benchmarking Chat Assistants on Long-Term Interactive Memory,” in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025.

Y. Wu, S. Liang, C. Zhang, Y. Wang, Y. Zhang, H. Guo, R. Tang, and Y. Liu, “From Human Memory to AI Memory: A Survey on Memory Mechanisms in the Era of LLMs,” arXiv preprint arXiv:2504.15965, 2025.

Y. Hu et al., “Memory in the Age of AI Agents,” arXiv preprint arXiv:2512.13564, 2025.

N. Shinn, B. Labash, and A. Gopinath, “Reflexion: Language Agents with Verbal Reinforcement Learning,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

G. Wang et al., “Voyager: An Open-Ended Embodied Agent with Large Language Models,” arXiv preprint arXiv:2305.16291, 2023.

P. Lewis et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 9459–9474, 2020.

K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang, “REALM: Retrieval-Augmented Language Model Pre-Training,” in *Proceedings of the 37th International Conference on Machine Learning (ICML)*, vol. 119, pp. 3929–3938, 2020.

V. Karpukhin et al., “Dense Passage Retrieval for Open-Domain Question Answering,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6769–6781, 2020.

S. Borgeaud et al., “Improving Language Models by Retrieving from Trillions of Tokens,” in *Proceedings of the 39th International Conference on Machine Learning (ICML)*, vol. 162, pp. 2206–2240, 2022.

Y. Gao et al., “Retrieval-Augmented Generation for Large Language Models: A Survey,” arXiv preprint arXiv:2312.10997, 2023.

N. Reimers and I. Gurevych, “Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks,” in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992, 2019.

N. Muennighoff et al., “MTEB: Massive Text Embedding Benchmark,” in *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pp. 2014–2037, 2023.

N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang, “Lost in the Middle: How Language Models Use Long Contexts,” *Transactions of the Association for Computational Linguistics*, vol. 12, pp. 157–173, 2024, doi: 10.1162/tacl_a_00638.

A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, “Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection,” in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

P. Sarthi, S. Abdullah, A. Tuli, S. Khanna, A. Goldie, and C. D. Manning, “RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval,” in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

A. Maharana et al., “Evaluating Very Long-Term Conversational Memory of Large Language Models,” in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.