Tailoring Memory Granularity for Multi-Hop Reasoning over Long Contexts

Anonymous ACL submission

Abstract

Multi-hop reasoning over long contexts remains challenging, as it requires integrating relevant contexts scattered across distant sources while resisting semantic drift and noise from distracting content. While retrieval-augmented generation (RAG) has emerged as the prevailing solution, most RAG approaches encode and store context in monolithic memory representations, resulting in noisy retrieval and brittle reasoning. To overcome these limitations, we introduce TAG (Tailoring Memory Granularity), a framework that prestructures memory into diverse granularities and employs a reward-guided navigator to adaptively compose hybrid memory tailored to each query. The navigator is trained with a multi-objective Bradley–Terry loss that learns the relative utility of different memory types, enabling dynamic routing across granularities. This design allows RAG systems to balance fine-grained detail with high-level abstraction, yielding more reliable reasoning. Extensive experiments on long-context multi-hop question answering (QA) benchmarks show that TAG achieves state-of-the-art performance. With only 0.033% additional parameters, it remains lightweight, highlighting its practicality as a scalable and effective solution for enhancing language model agents in complex, real-world scenarios.

1 Introduction

Large Language Models (LLMs) are increasingly being endowed with agentic capabilities, enabling them to interact with environments, utilize tools, and pursue complex objectives autonomously (Wang et al., 2024c; Xi et al., 2025). A cornerstone of such sophisticated agency is the memory module, which controls how agents process, store, and retrieve past information to inform future actions and reasoning processes (Zhang et al., 2024; Lee et al., 2024; Diao et al., 2025).

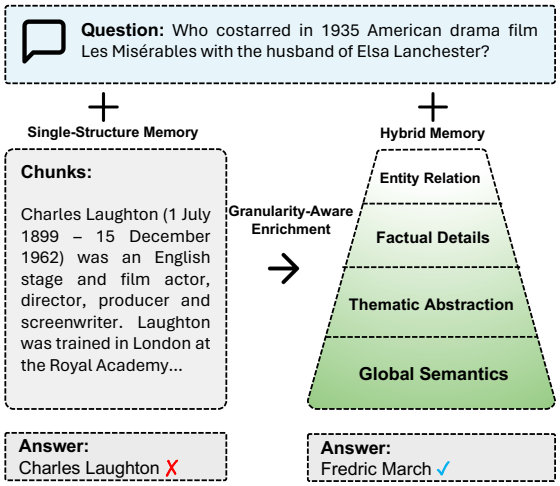

Figure 1: Hybrid memory system of LLM-based agents offers rich granular text information.

This is particularly critical for long-context reasoning tasks, where relevant information may be distributed across vast textual passages, demanding nuanced understanding at multiple memory granularities—from fine-grained factual details to global semantics (Li et al., 2024a; Lee et al., 2024).

Previous LLM-based agents often rely on singular memory structure, typically text chunks (Hu et al., 2024; Packer et al., 2023). While straightforward, such uni-modal memory can lead to inefficient retrieval and the inclusion of noisy or irrelevant information in the agent's context window, potentially impairing reasoning, a phenomenon referred to as being *"lost in the middle"* (Liu et al., 2023).

Cognitive Fit Theory, inspired by cognitive science (Vessey, 1991; Umanath and Vessey, 1994), suggests that human cognitive processes are optimized when the representation of information aligns with task requirements. For example, while segmented text maintains local context, knowledge triples prove superior for tasks that necessitate clearly defined relationships (Anokhin et al., 2024). As a result, recent progress has utilized LLMs to

create a range of knowledge structures (Li et al., 2023; Jain et al., 2024; Li et al., 2024b) and to implement a hybrid memory system (Zeng et al., 2024), aiming to find the best structural representations for a variety of tasks in complex real-world scenarios.

However, a critical challenge remains: different memory structures inherently convey different granularities of information, and existing approaches often fail to adaptively leverage the optimal blend of these varied structural representations based on the specific demands of an incoming query, as shown in Figure 1. In this work, we introduce TAG, a novel framework that integrates multiple, structurally diverse memory types and employs a reward-guided retrieval mechanism to adaptively build the optimal memory composition for each query. TAG constructs hybrid memory with stratified information granularity, including document chunks for local contextual details, atomic facts for precise factual units, knowledge triples for explicit entity relationships, and summaries for abstractive understanding.

The central component of TAG is a memory router module, based on a multi-objective reward model, which predicts the relative preference of each memory structure for a given query. To achieve this, we propose a multi-object Bradley-Terry loss, enabling the reward model to learn a memory blending coefficient that guides a weighted retrieval strategy. This allows our LLM agent to dynamically assemble a granularly balanced and task-relevant hybrid memory, enhancing retrieval-augmented generation. We evaluate TAG on three long-context multi-hop reasoning benchmarks. Our method achieves state-of-the-art performance and outperforms the uni-structure remarkably, by up to 7% on the HotPotQA benchmark. Additionally, our lightweight framework only adds 0.033% additional parameters, highlighting its practicality as a scalable and effective solution for enhancing LLM agents in complex, real-world scenarios.

2 Related Works

2.1 LLM-based Agents

The paradigm of LLMs has recently expanded from proficient text generators to the core reasoning engines of autonomous agents (Xi et al., 2023; Wang et al., 2023c). These LLM-based agents are designed to perceive their environment, maintain memory, plan, and act to achieve complex goals across diverse applications, including social simulation (Park et al., 2023), software development (Hong et al., 2023; Qian et al., 2023),role-playing (Wang et al., 2024e; Chen et al., 2024), and open-world gaming (Wang et al., 2023b; Wang et al.). A critical architectural shift involves augmenting LLMs with external modules, such as planning modules, tool-use libraries, and sophisticated memory systems (Yao et al., 2022; Shinn et al., 2023; Packer et al., 2023). Among these, the memory module is paramount, as it enables agents to learn from past experiences, retain knowledge over extended interactions, and ground their reasoning in relevant information. However, designing effective memory systems that balance richness, efficiency, and relevance remains a significant challenge (Zhang et al., 2024; Gutiérrez et al., 2024).

2.2 Memory Structures

Early LLM-based agents typically rely on vector databases storing raw text chunks (Lewis et al., 2020), which often contain substantial noise and make it difficult to retrieve truly relevant information (Liu et al., 2023). To overcome these limitations, semi-structured forms, such as summaries (Xu et al., 2023) and atomic facts (Li et al., 2024a; Min et al., 2023), condense information while retaining key content. Highly structured memories, including knowledge graphs (Baek et al., 2023; Sun et al., 2024) and triples (Anokhin et al., 2024), offer higher precision but may lose contextual nuance. This trade-off between precision and coverage has motivated hybrid memory systems that combine multiple representations (Zeng et al., 2024). Recent work has also examined structure and granularity in retrieval. RAPTOR (Wang et al., 2024d) builds hierarchical summaries through recursive abstractive processing, while Dense X Retrieval (DxR) (Wang et al., 2023a) analyzes retrieval at document, passage, and proposition levels to study granularity effects. Both methods, however, operate within a single memory structure and lack mechanisms to dynamically integrate heterogeneous representations. In contrast, TAG introduces multiple prestructured memory types and a reward-guided router that adaptively composes hybrid memories tailored to each query.

3 Method

Figure 2 illustrates the overview of our method. Section 3.1 describes the generation of each memory structure. Section 3.2 explains the pipeline of

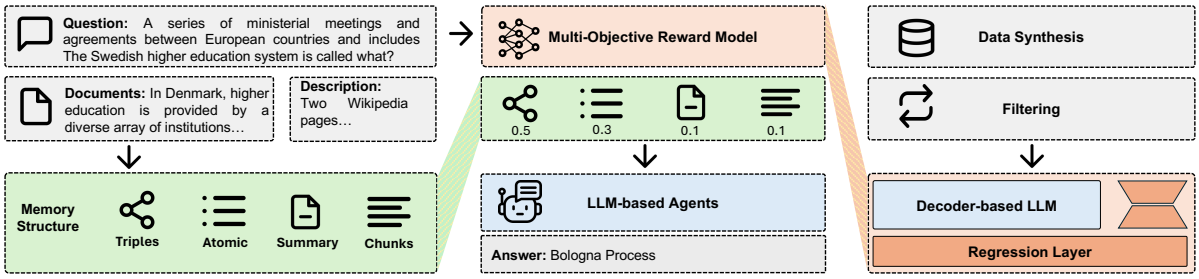

Figure 2: Overview of the augmented hybrid memory retrieval via reward-guided structuring. For each question and its corresponding document, raw information is transformed into various structural memories. The router then determines the optimal allocation strategy for the hybrid memory and orchestrates the retrieval of the most relevant memories to support precise and contextually enriched responses.

our augmented retrieval process.

3.1 Memory Generation

Structural memory generation equips agents with the ability to transform raw textual documents D_q into structured representations M_q, thereby enhancing the storage, retrieval, and reasoning capabilities of LLM-based agents. Following prior work (Zeng et al., 2024), we build our memory system using four representative structures with increasing levels of abstraction and granularity: knowledge triples (T_q), atomic facts (A_q), summaries (S_q), and document chunks (C_q).

Knowledge Triples T_q encodes semantic relationships between entities in the form (h, r, t), where h, r, and t denote the *head*, *relation*, and *tail*, respectively. We follow prior work (Anokhin et al., 2024; Fang et al., 2024; Zeng et al., 2024) to generate such triples using a prompt-conditioned LLM. For example, from a document mentioning *Bologna Process*, we may extract: (*Bologna Process*, *under*, *Lisbon Recognition Convention*), or (*Bologna Process*, *named after*, *University of Bologna*). The prompt for generating triples is shown in Figure 7a.

Atomic Facts A_q are concise, standalone declarative statements that convey a single factual assertion, or verifiable pieces of information extracted from the source document D_q (Min et al., 2023; Li et al., 2024a). Each atomic fact is designed to represent a minimal unit of knowledge, facilitating precise retrieval and reasoning. For instance, from a document discussing the Bologna Process, we might extract: *The Bologna Process was opened to other countries in the European Cultural Convention of the Council of Europe.* Unlike knowledge triples, which strictly represent explicit semantic relationships between entities, atomic facts can express more abstract content, such as implicit relationships and conditions that are difficult to succinctly encode in a triple structure. The prompt for generating atomic facts is shown in Figure 8.

Summaries S_q are concise, high-level representations of documents D_q, capturing essential information while omitting extraneous details. This approach ensures that the summaries retain both global semantics and critical details pertinent to downstream tasks (Lee et al., 2024). The prompt for generating summaries is shown in Figure 7b.

Chunks C_q denote contiguous segments of text derived from document D_q, designed to preserve local coherence and facilitate efficient processing. Following typical chunking methods that employ fixed-length segmentation (Gao et al., 2023; Zeng et al., 2024), the chunked memory is represented as $C_q(D_q) = \{c_1, c_2, \ldots, c_i\}$, where each c_j is a chunk with at most L length.

3.2 Augmented Memory Composition via Reward-Guided Retrieval

After building up the hybrid memory for a given question q, we propose an augmented retrieval mechanism that adaptively fuses multiple structural memories. This process is guided by a multi-objective reward model trained to infer an optimal allocation for each memory type. The inference pipeline is illustrated in Figure 2. The design and training of the reward model are detailed in Section 4. Specifically, for a given question q associated with corresponding hybrid memory, we denote the full hybrid memory M_q as:

$$M_q = \{C_q, T_q, A_q, S_q\}. \quad (1)$$

The reward model R maps the input question q to a 4-dimensional weight vector:

$$w_q = R(q) = [w_C, w_T, w_A, w_S], \quad (2)$$

where each w_i denotes the importance weight of the corresponding memory type (Chunks, Triples, Atomic facts, and Summaries, respectively). To ensure these weights reflect probabilities summing to 1, we apply a softmax normalization:

$$w_i' = \frac{\exp(w_i/\tau)}{\sum_{j\in\{C,T,A,S\}} \exp(w_j/\tau)}, \quad (3)$$

where $i \in \{C, T, A, S\}$ and τ denotes a temperature parameter adjusting the distribution sharpness. Next, we quantize these normalized weights into discrete retrieval counts for the memory sets. Given a desired total retrieval budget K, we compute the quantized counts as:

$$n_i = \lfloor w_i' \cdot K \rfloor, \quad \text{for each } i \in C, T, A, S, \quad (4)$$

where $\lfloor \cdot \rfloor$ denotes the floor function. Typically, this results in $\sum_i n_i \leq K$, leaving a small remainder $R = K - \sum_i n_i$. To allocate this remainder, we distribute the leftover retrieval counts to memory types based on the largest fractional components from the product $w_i' \cdot K$. During retrieval:

$$\hat{M}_q = \cup_{M\in M_q} \text{Retrieve}(q, M, n_M), \quad (5)$$

where n_M denotes the quantized retrieval count for memory type M. The retrieval function $\text{Retrieve}(q, M, n_M)$ samples the top-n_M items from each memory type based on semantic similarity to q. Specifically, we utilize Qwen series embedding models for semantic retrieval (details in Appendix B).

4 Navigator Training

A central component of the TAG framework is the memory navigator, which determines the optimal composition of memory structures for each input query. However, the absence of annotated datasets specifying preferred memory allocations presents a significant challenge for supervised training. To overcome this, we propose a weakly supervised training pipeline that enables the learning of a robust multi-objective reward model navigator.

4.1 Data Construction

Currently, there are no datasets available for optimal memory structure allocation, making it challenging to guide advanced language models in directly crafting allocation strategies for various memory structures. Building on previous research (Li et al., 2024b), we initially apply in-context learning to derive the most appropriate memory structure from four types for each question, complemented by essential document content. We then refine and balance the dataset by selecting 200 examples for each memory structure across all datasets. Each example is labeled with a preferred structure t_w, while other structures are marked as less relevant, resulting in a multi-pairwise preference sample:

$$D_{\text{synthetic}} = \{q^{(k)}, C^{(k)}, t_w^{(k)}, \}_{k=1}^{N}, \quad (6)$$

where $q^{(k)}$ denotes the query, $C^{(k)}$ is the associated document content, and $t_w^{(k)}$ are the preferred structure types, respectively.

4.2 Model Architecture

Drawing inspiration from typical multi-objective reward model design (Wang et al., 2024b, 2023d, 2024a), we leverage a pre-trained decoder-based LLM as the feature extractor. To repurpose this backbone for preference modeling, we freeze the original language modeling head and instead append a lightweight regression head tailored to the memory structure ranking task. To enable efficient fine-tuning, we adopt Parameter-Efficient Fine-Tuning (PEFT) via Low-Rank Adaptation (LoRA) (Hu et al., 2022), which introduces trainable low-rank updates into selected layers of the LLM while keeping the majority of parameters frozen. This design allows the router to adapt quickly to the ranking task with minimal computational overhead. During the training phase of the router, only the LoRA parameters and the regression head are updated; the pre-trained LLM backbone remains frozen. For inference, the process involves two stages: first, the LoRA adapters and the regression head are activated to predict the memory allocation weights based on the input query (concatenated with specific instructions and system prompts tailored for weight prediction). Subsequently, these components are deactivated, and the LLM agent's original output head is used for generating the final answer, now informed by the retrieved memory. The input query is again concatenated with different instructions and system prompts suitable for answer generation.

4.3 Bradley-Terry Loss for Multi-Score Regression

The reward model R maps an input query to a four-dimensional score vector $\mathbf{s} = [s_1, s_2, s_3, s_4]$, where each component corresponds to the predicted

Memory Structure	HotPotQA		2WikiQA		MuSiQue		Average	
	EM	F1	EM	F1	EM	F1	EM	F1
Uni-memory								
Chunks	52.50	69.12	35.50	47.44	16.50	33.02	34.83	49.86
Triples	29.00	44.89	29.00	39.85	9.00	19.76	22.33	34.83
Atomic Facts	37.50	50.05	27.00	38.14	12.50	23.61	25.67	37.27
Summaries	51.50	68.88	38.50	48.25	22.00	36.39	37.33	51.17
Hybrid-memory								
`Random`	53.00	71.17	37.00	48.66	20.50	36.66	36.83	52.16
`Best@1`	47.00	61.07	34.50	45.51	14.50	29.09	32.00	45.22
`Best@2`	48.00	65.42	36.00	47.01	15.50	30.88	33.83	47.77
`Best@3`	53.00	70.28	35.00	46.00	15.00	30.12	33.00	46.47
`Equal`	**58.00**	73.61	40.00	51.01	16.50	32.66	38.83	52.43
`TAG`	**58.00**	**75.73**	**40.50**	**51.22**	**22.50**	**37.92**	**40.33**	**54.96**

Table 1: Performance of single (uni) and hybrid memory structures using single-step retrieval across three datasets. Random refers to randomly retrieved memory. Best@k selects memory from the Top-K most suitable memory structures. Equal retrieves an equal number of items from each of the four available memory structures. The best scores are bolded.

utility of one memory type. Unlike traditional classification objectives, we aim to learn *relative* contributions, as multiple memory types may offer complementary benefits. To this end, we adopt the Bradley-Terry (BT) loss (Bradley and Terry, 1952), a principled probabilistic framework for modeling pairwise preferences. For a given pair (i, j), the BT model defines the probability that memory type i is preferred over j as:

$$P(i \succ j) = \frac{\exp(s_i)}{\exp(s_i) + \exp(s_j)} = \sigma(s_i - s_j), \tag{7}$$

where $\sigma(\cdot)$ is the sigmoid function. Given a training instance with a preferred type g and all other types $j \neq g$, we define the loss as the average negative log-likelihood across all pairwise comparisons:

$$\mathcal{L}_{\text{BT}}(\mathbf{s}, g) = \frac{1}{K-1} \sum_{j \neq g} -\log\left(\sigma(s_g - s_j)\right), \tag{8}$$

where $K = 4$ is the total number of structure types. This loss navigates the model to assign higher scores to the preferred structure relative to the others while encouraging a *relative ranking* rather than an absolute classification. Section 7.2 conduct ablation study on different loss functions.

5 Experimental Results

5.1 Experimental Setting

Dataset Following prior work (Gutiérrez et al., 2024), we evaluate our method on three challenging long-context Multi-hop question answering (QA) datasets: HotPotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020), and MuSiQue (Trivedi et al., 2022). To reduce cost during memory construction, we adopt the evaluation set used in Zeng et al. (2024). Additional details about these datasets are provided in Appendix C.

Evaluation To evaluate QA performance, we follow previous work (Li et al., 2024a) and use standard metrics such as Exact Match (EM) score and F1 score for all the datasets.

Baseline We compare our method against several hybrid and adaptive memory selection strategies. `Random` follows Zeng et al. (2024) by retrieving a randomly selected mix of memory items. `Best@k` selects items from the top k most suitable memory structures. For instance, `Best@1` (an alternative to StructRAG (Li et al., 2024b)) uses only the most preferred structure, while `Best@2` and `Best@3` use the top two or three preferred structures, respectively. Equal retrieves an equal number of items from each of the four available memory structures.

Implementation Details We build our framework upon the Qwen2 (Bai et al., 2023) series (Appendix B), using default hyperparameter configurations. Specifically, Qwen2.5-7B-Instruct serves as the backbone for both multi-objective reward modeling (see Section 4 for details) and LLM-agent inference. To validate the robustness of our approach, we further conduct experiments based on Llama-

Memory Structure	HotPotQA		2WikiQA		MuSiQue		Average	
	EM	F1	EM	F1	EM	F1	EM	F1
TAG	**58.00**	**75.73**	**40.50**	**51.22**	**22.50**	**37.92**	**40.33**	**54.96**
W/o Chunks	51.00	68.53	38.50	49.16	19.50	34.94	36.33	50.88
W/o Triples	56.50	72.48	37.00	49.51	22.50	37.70	38.67	53.23
W/o Atomic Facts	55.00	72.99	40.50	51.63	20.00	33.76	38.50	52.79
W/o Summaries	51.50	71.01	36.50	48.54	15.00	31.73	34.33	50.43

Table 2: Performance of removing each memory structure compared to keeping full memory.

3.1-8B-Instruct, under identical datasets and hyperparameter settings (Results are provided in Table 8). We use Qwen2.5-72B-Instruct for synthetic data generation and serve the model via API using `vllm`[1]. We present the details of reward model training in Appendix D.

5.2 Main Results

We report the primary results of TAG under the typical single-step retrieval setting (Rubin et al., 2022). The statistics of retrieved memory units is illustrated in Figure 6. As shown in Table 1, TAG consistently outperforms both single-memory and hybrid memory baselines. TAG achieves the highest performance across all datasets, surpassing the best single-memory by a considerable margin. On HotPotQA, TAG achieves an F1 score of 75.73, outperforming the strongest single memory (Chunks, 69.12) by 6.61. For 2WikiQA and MuSiQue, TAG achieves 51.22 and 37.92 F1 score, respectively, setting a new state-of-the-art performance, demonstrating its superior ability to retrieve task-aligned memory with rich granularity that can enhance the long-context reasoning ability of LLM-based agents.

5.3 Ablation Study

To better understand the contribution of each memory structure in our framework, we conduct an ablation study by systematically removing one structure at a time. The results are shown in Table 2. We observe that the TAG consistently outperforms all ablated variants across datasets, highlighting the complementary strengths of the four memory structures. Removing chunks and summaries leads to large performance drops, with an average F1 score decrease of 4.53 for summaries. This suggests that chunks and summaries, which offer high-level context and thematic overviews that aid global reasoning, are especially critical for grounding fine-grained reasoning in contextually rich passages. Removing triples and atomic facts causes a moderate drop in performance. Triples encode explicit entity-relation-entity structures that benefit factoid-style reasoning but may lack flexibility in capturing implicit connections.

[1] https://pypi.org/project/vllm/

Components	Parameters	Additional
Base Model	7.62B	100%
LoRA Block	2.52M	0.033%
Regression Head	0.014M	0.00019%

Table 3: Comparison of parameter size for router components relative to the base model.

Dataset	Predict Weights (s)	Generation (s)
2WikiQA	0.0451	0.4719
HotpotQA	0.0466	0.4582
MuSiQue	0.0468	0.4857

Table 4: Average per-call latency for weights prediction and answer generation across three datasets.

6 Latency Analysis

We assess the efficiency of the proposed reward-guided memory router by analyzing its parameter overhead and inference latency. As described in Section 4.2, the router comprises a LoRA-adapted LLM and a lightweight regression head that predicts optimal memory composition for each query. Table 3 reports the parameter size of these components. The LoRA module introduces only 0.033% additional parameters relative to the base LLM, while the regression head contributes a negligible 0.00019%. This minimal overhead highlights the scalability and practicality of our router for integration into large-scale agent systems. To evaluate runtime efficiency, we measure the average latency introduced by the router on three multi-hop QA

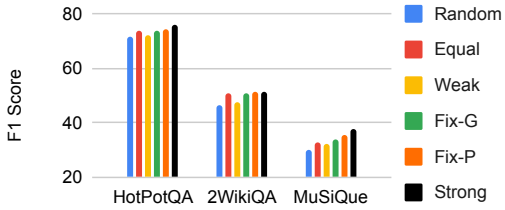

Figure 3: F1 score of different routing Strategies.

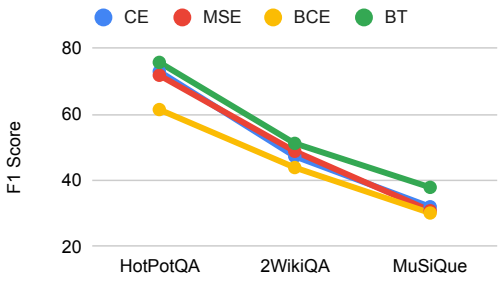

Figure 4: F1 score of different training loss functions.

benchmarks. As shown in Table 4, the memory routing step requires only 0.0451 to 0.0468 seconds per query, which is marginal compared to the total answer generation time (approximately 0.46 to 0.49 seconds). These results demonstrate that our reward-guided routing mechanism significantly improves the flexibility and task adaptiveness of memory retrieval, while incurring minimal computational cost, making it a highly viable solution for real-world deployment.

7 Analysis of Navigating Strategy

In this section, we aim to answer two questions: *1) How does the quality of the router impact the performance? 2) Is it necessary to employ a multi-objective reward model for memory routing?*

7.1 Routing Strategy

To assess the impact of the reward model's quality on TAG, we conducted a comparative analysis using routers of varying routing capabilities. We evaluated four distinct router configurations: (1) *Strong Router*: trained on the complete synthetic dataset (2400 examples), representing our best-performing reward model. (2) *Weak Router*: trained on a randomly selected 50% subset of the training data (1200 examples), simulating a less informed model. (3) *Average Router*: a heuristic baseline that assigns equal importance ($w_i = 0.25$) to all four memory structures. (4) *Bad Router*: a baseline that assigns random weights to memory structures. (5) *Fix-G (global)*: a single global weight vector $w = [w_C, w_T, w_A, w_S]$ applied to all queries, where each w_i is proportional to the single-structure F1 score of the corresponding memory type reported in Table 1, reflecting their relative importance. (6) *Fix-P (per-dataset)*: similar to (5), but a separate w is computed for each dataset (HotpotQA, 2Wiki, MuSiQue) using their respective single-structure F1 scores from Table 1.

Figure 3 demonstrates that the effectiveness of TAG heavily depends on the router's quality. The *Strong Router* achieves the highest F1 across all datasets, while the *Weak Router* can underperform the *Average Router*, indicating that insufficiently trained reward models may misallocate memory. Notably, both fixed-weight schemes (*Fix-G* and *Fix-P*) lag behind the query-adaptive *Strong Router*, underscoring the necessity of *per-query* routing over corpus-level weighting. Complete results are reported in Table 7.

7.2 Loss Function Design

As discussed in Section 4.3, our objective is to model the relative contributions of multiple memory structures to a given question, rather than selecting a single best structure. To this end, the model outputs a four-dimensional score vector, where each dimension reflects the predicted utility of a corresponding memory structure. We compare our proposed multi-object Bradley-Terry (BT) loss against three alternative loss functions: Cross-Entropy (CE)(Shannon, 1948), Mean Squared Error (MSE)(Bishop, 2006), and Binary Cross-Entropy (BCE)(Cox, 1958) (see AppendixD for implementation details). Each loss function introduces distinct inductive biases. CE treats the task as multi-class classification, assuming a single optimal structure. MSE models the task as regression, minimizing the squared L2 distance between predicted scores and the preference labels. BCE allows independent estimation of each structure's relevance, making it more flexible for multi-label scenarios. In contrast, our multi-object BT loss is explicitly designed to encourage pairwise ranking consistency. It encourages the model to score the preferred structure higher than the others, aligning directly with our goal of capturing relative usefulness. As illustrated in Figure 4, the BT loss achieves the highest average F1 score among all the loss functions, indicating that its ranking-based formulation is more effective for modeling nuanced structural preferences. Full results are reported in Table 7.

8 Case Study

To better understand the importance of hybrid memory composition, we analyze a failure case involving multi-hop reasoning from HotPotQA dataset. The query asks: *"Who costarred in the 1935 American drama film Les Misérables with the husband of Elsa Lanchester?"*.

Mix Memory

Triples:
• Charles Laughton; nationality; English
...
Atomic Facts:
• Charles Laughton was trained at the Royal Academy of Dramatic Art in London.
• Fredric March and Charles Laughton star in Les Misérables.→`Factual Details`
...
Chunks:
• Les Misérables is a 1935 American drama film starring Fredric March and Charles Laughton...→`Global Semantics`
...
Answer: Fredric March ✓

Single chunk structure memory

Chunks:
• Les Misérables is a 1935 American drama film starring Fredric March and Charles Laughton... →`Global Semantics`
• Charles Laughton...lived and worked with Elsa Lanchester... →`Noise Content`
...
Answer: Charles Laughton ✗

When relying on a single memory type such as document chunks, the agent retrieves one passage chunk including statement that Les Misérables stars Charles Laughton and Fredric March, and another chunk describe biographical information including that Charles Laughton was the husband of Elsa Lanchester. However, without structured representations to guide relevance, the agent selects Charles Laughton as the answer—incorrectly identifying Lanchester's husband instead of his co-star. This mistake highlights the "semantic drift" problem (Spataru et al., 2024) typical of long chunks, where textual noise or proximity bias misleads the model. Under Hybrid memory configuration, the agent correctly identifies the answer as **Fredric March**. This memory context explicitly links Elsa Lanchester to her husband, Charles Laughton, and co-lists both Laughton and March in the cast of *Les Misérables*. The model successfully performs two-hop reasoning: first inferring that Charles Laughton is Lanchester's husband, and then identifying Fredric March as Laughton's co-star.

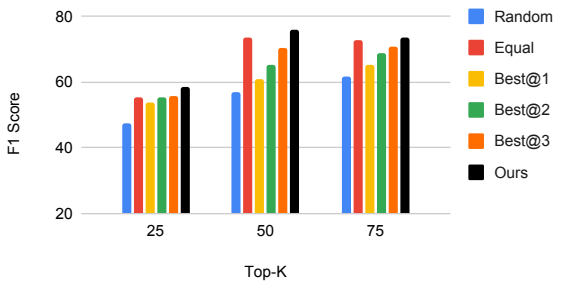

Figure 5: F1 score of different routing strategies under varying Top-K.

9 Hyperparameter Sensitivity

To further validate the robustness of `TAG` framework, we analyze its sensitivity to the hyperparameter Top-K, which controls the number of retrieved memory units during inference. We evaluate performance across three values of $K \in 25, 50, 75$ using the HotPotQA dataset. As shown in Figure 5, `TAG` achieves optimal performance at K = 50, with an F1 score of 75.73, outperforming other configurations. Overall, these results underscore the importance of balancing retrieval breadth and precision. While larger K values provide more opportunities to gather relevant facts, they may also introduce noise. Our adaptive routing mechanism helps mitigate this trade-off by learning to allocate retrieval resources more effectively across different memory types. The full results is provided in Table 7.

10 Conclusion

This work introduces `TAG`, a novel framework designed to overcome the limitations of uni-structural memory in large language model agents for complex, long-context reasoning. By integrating diverse memory structures and employing a reward-guided retrieval mechanism trained with a multi-object Bradley-Terry loss, `TAG` adaptively composes an optimal memory set tailored to each query. Our extensive experiments underscore the critical role of adaptive, granular memory composition in enhancing the long-context reasoning capabilities of LLM agents.

Limitations

Our experiments are confined to open-source models, which might not be representative of the broader landscape of LLMs, particularly those that are closed-source and potentially optimized for proprietary datasets.

Ethics Statement

We have not identified any ethical concerns directly related to this study.

References

Petr Anokhin, Nikita Semenov, Artyom Sorokin, Dmitry Evseev, Mikhail Burtsev, and Evgeny Burnaev. 2024. Arigraph: Learning knowledge graph world models with episodic memory for llm agents. *arXiv preprint arXiv:2407.04363*.

Jihun Baek, Soyeong Lee, Sungmin Kim, Kangwook Lee, and Sungju Kim. 2023. Knowledge graph-enhanced large language models for instruction following. *arXiv preprint arXiv:2310.04172*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer, New York.

Ralph Allan Bradley and Milton E. Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Jiangjie Chen, Xintao Wang, Rui Xu, Siyu Yuan, Yikai Zhang, Wei Shi, Jian Xie, Shuang Li, Ruihan Yang, Tinghui Zhu, et al. 2024. From persona to personalization: A survey on role-playing language agents. *arXiv preprint arXiv:2404.18231*.

David R. Cox. 1958. The regression analysis of binary sequences. *Journal of the Royal Statistical Society: Series B (Methodological)*, 20(2):215–232.

Xingjian Diao, Chunhui Zhang, Weiyi Wu, Zhongyu Ouyang, Peijun Qing, Ming Cheng, Soroush Vosoughi, and Jiang Gui. 2025. Temporal working memory: Query-guided segment refinement for enhanced multimodal understanding. *arXiv preprint arXiv:2502.06020*.

Jinyuan Fang, Zaiqiao Meng, and Craig MacDonald. 2024. TRACE the evidence: Constructing knowledge-grounded reasoning chains for retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 8472–8494, Miami, Florida, USA. Association for Computational Linguistics.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

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

Sirui Hong, Mingchen Zheng, Jiahui Chen, Yuheng Su, Chen Wang, Ceyao Cui, Weize Liu, Yuxuan Cheng, Jian Zhou, Jinspac Fu, et al. 2023. Metagpt: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00366*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Mengkang Hu, Tianxing Chen, Qiguang Chen, Yao Mu, Wenqi Shao, and Ping Luo. 2024. Hiagent: Hierarchical working memory management for solving long-horizon agent tasks with large language model. *arXiv preprint arXiv:2408.09559*.

Priya Jain et al. 2024. Knowledge structure generation and utilization by llms: Emerging methods. *arXiv preprint arXiv:2402.03546*.

Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John F. Canny, and Ian Fischer. 2024. A human-inspired reading agent with gist memory of very long contexts. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

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

Shilong Li, Yancheng He, Hangyu Guo, Xingyuan Bu, Ge Bai, Jie Liu, Jiaheng Liu, Xingwei Qu, Yangguang Li, Wanli Ouyang, et al. 2024a. Graphreader: Building graph-based agent to enhance long-context abilities of large language models. *arXiv preprint arXiv:2406.14550*.

Xiang Li et al. 2023. Structured knowledge generation by large language models. *arXiv preprint arXiv:2310.08547*.

Zhuoqun Li, Xuanang Chen, Haiyang Yu, Hongyu Lin, Yaojie Lu, Qiaoyu Tang, Fei Huang, Xianpei Han, Le Sun, and Yongbin Li. 2024b. Structrag: Boosting knowledge intensive reasoning of llms via inference-time hybrid information structurization.

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

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G Patil, Ion Stoica, and Joseph E Gonzalez. 2023. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22.

Chen Qian, Xin Cong, Cheng Yang, Weize Chen, Yusheng Su, Juyuan Wu, Zhiyuan Liu, and Weize Liu. 2023. Chatdev: Communicative agents for software development. *arXiv preprint arXiv:2307.07924*.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2671.

Claude E. Shannon. 1948. A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423.

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

Ava Spataru, Eric Hambro, Elena Voita, and Nicola Cancedda. 2024. Know when to stop: A study of semantic drift in text generation. *arXiv preprint arXiv:2404.05411*.

Zhen Sun, Runjin Wang, Siyuan Chen, Kexuan Wang, Kaisheng Feng, Chonggang Wang, Shimin Shi, Yejin Zhang, Xin Huang, Yu Zhang, et al. 2024. Structgpt: A general framework for large language model to reason over structured data. *arXiv preprint arXiv:2402.00818*.

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

Narayan S Umanath and Iris Vessey. 1994. Multiattribute data presentation and human judgment: A cognitive fit perspective. *Decision Sciences*, 25(5-6):795–824.

Iris Vessey. 1991. Cognitive fit: A theory-based analysis of the graphs versus tables literature. *Decision Sciences*, 22(2):219–240.

Alex Wang, Jiahui Gao, Jing Ma, and Kelvin Guu. 2023a. Dense x retrieval: What retrieval granularity should we use? *arXiv preprint arXiv:2312.06648*.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023b. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.

Haoxiang Wang, Yong Lin, Wei Xiong, Rui Yang, Shizhe Diao, Shuang Qiu, Han Zhao, and Tong Zhang. 2024a. Arithmetic control of llms for diverse user preferences: Directional preference alignment with multi-objective rewards. In *Association for Computational Linguistics*.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. 2024b. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. *arXiv preprint arXiv:2406.12845*.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024c. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Luo, Wayne Xin Tian, Miaohua Niu, Le Wu, Keyu Wang, Shucheng Wang, and Lining Wang. 2023c. A survey on large language model based autonomous agents. *arXiv preprint arXiv:2308.11432*.

Sarthak Wang, Nicholas Roberts, and Dragomir Radev. 2024d. Raptor: Recursive abstractive processing for tree-organized retrieval. *arXiv preprint arXiv:2401.18059*.

Xintao Wang, Yunze Xiao, Jen-tse Huang, Siyu Yuan, Rui Xu, Haoran Guo, Quan Tu, Yaying Fei, Ziang Leng, Wei Wang, et al. 2024e. Incharacter: Evaluating personality fidelity in role-playing agents through psychological interviews. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1840–1873.

Zhilin Wang, Yi Dong, Jiaqi Zeng, Virginia Adams, Makesh Narsimhan Sreedhar, Daniel Egert, Olivier Delalleau, Jane Polak Scowcroft, Neel Kant, Aidan Swope, and Oleksii Kuchaiev. 2023d. Helpsteer: Multi-attribute helpfulness dataset for steerlm. *Preprint*, arXiv:2311.09528.

Zihao Wang, Shaofei Cai, Anji Liu, Xiaojian Ma, and Yitao Liang. Jarvis-1: Open-world multi-task agents with memory-augmented multimodal language models. In *Second Agent Learning in Open-Endedness Workshop*.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. 2023. The rise and potential of large language model based agents: A survey. *arXiv preprint arXiv:2309.07864*.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. 2025. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101.

Yikang Xu, Shuyuan Li, J D Choi, and M Coyle. 2023. Memorysandbox: A unified framework for diverse agent memory. *arXiv preprint arXiv:2312.10272*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Ruihong Zeng, Jinyuan Fang, Siwei Liu, and Zaiqiao Meng. 2024. On the structural memory of llm agents. *arXiv preprint arXiv:2412.15266*.

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

A Loss Function

We consider the following three alternatives to the Bradley-Terry (BT) loss:

A.1 Cross-Entropy Loss (CE)

Cross-Entropy Loss (CE): This loss treats the task as a standard multi-class classification problem. The model outputs a score vector $\mathbf{s} = [s_1, s_2, s_3, s_4] \in \mathbb{R}^4$, and the loss encourages the score corresponding to the gold method g to be highest. The softmax function is applied to the scores to produce a probability distribution:

$$p_i = \frac{\exp(s_i)}{\sum_{j=1}^{4} \exp(s_j)}, \tag{9}$$

and the loss is defined as:

$$\mathcal{L}_{\text{CE}} = -\log(p_g) = -\log\left(\frac{\exp(s_g)}{\sum_{j=1}^{4} \exp(s_j)}\right). \tag{10}$$

While effective for hard classification, this loss enforces mutual exclusivity among the methods and discourages overlapping or partial contributions.

A.2 Mean Squared Error (MSE)

Mean Squared Error (MSE) Loss: This regression-based loss minimizes the squared distance between the predicted score vector $\mathbf{s}$ and the one-hot ground-truth label vector $\mathbf{y} = [y_1, y_2, y_3, y_4]$, where $y_g = 1$ and $y_j = 0$ for $j \neq g$. The loss is defined as:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{4}\sum_{i=1}^{4}(s_i - y_i)^2. \tag{11}$$

This formulation supports soft contributions but lacks a mechanism to enforce comparative ranking or preference ordering.

A.3 Binary Cross-Entropy(BCE)

Binary Cross-Entropy with Logits Loss (BCE): This loss models each score independently using binary classification. The raw scores $\mathbf{s}$ are passed through the sigmoid function $\sigma(s_i) = 1/(1+e^{-s_i})$, and the loss compares each prediction to its corresponding binary label $y_i \in \{0, 1\}$:

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{4}\sum_{i=1}^{4}\left[y_i \log \sigma(s_i) + (1 - y_i)\log(1 - \sigma(s_i))\right] \tag{12}$$

Unlike CE, BCE does not assume exclusivity and can assign high confidence to multiple methods. It also provides a natural extension path to soft supervision with fractional labels.

B Open-sourced models

We follow previous works (Li et al., 2024b) and use the Qwen2 series model in our work, as shown in Table 5.

Section	textbfModel Name
Retriever	`gte-Qwen2-1.5B-instruct`
Agent Backbone	`Qwen2.5-7B-Instruct`
Reward Model	`Qwen2.5-7B-Instruct`
Data Synthesis	`Qwen2.5-72B-Instruct`

Table 5: Open-sourced models used this work.

C Datasets

We evaluate our method on three challenging English multi-hop QA datasets, adapted for long-context reasoning by utilizing full Wikipedia passages. HotpotQA features 2-hop questions authored by native speakers, derived from two related Wikipedia paragraphs. 2WikiMultihopQA consists of questions requiring up to 5 reasoning hops, which are synthesized using manually designed templates to ensure true multi-hop reasoning and prevent shortcut solutions. Questions in MuSiQue are composed from simpler questions to involve up to 4 reasoning hops. They are subsequently paraphrased by human annotators to enhance linguistic naturalness and guard against superficial shortcuts. For our long-context setting, we used the complete Wikipedia passages from which the original supporting and distracting paragraphs were sourced. The statistical information of datasets is provided in Table 6.

Dataset	Avg. # Tokens	# Samples
HotpotQA	1,362	200
2WikiMultihopQA	985	200
MuSiQue	2,558	200

Table 6: The statistic and example of datasets.

D Reward Model training

We construct a dataset comprising 2,400 examples. We adopt a standard LoRA setup with rank 8 on the `q_proj` and `v_proj` layers of the decoder-based backbone LLM. We training the model in 1 NVIDIA RTX A6000 GPU using a learning rate of 2×10^{-5}, 3 training epochs, and a batch size of 8.

E Prompt

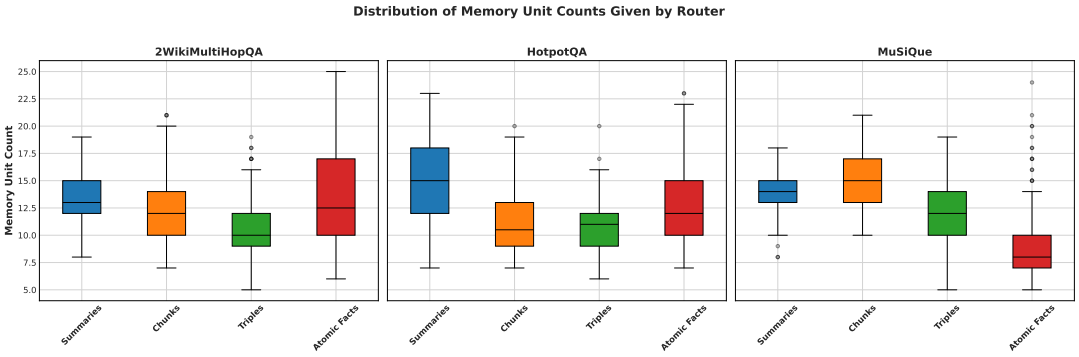

Figure 6: Distribution of memory unit counts given by router.

Memory	Top-k = 25		Top-k = 50		Top-k = 75	
	EM	F1	EM	F1	EM	F1
Random	35.00	47.43	53.00	71.17	52.50	69.13
Best@1	36.50	53.93	47.00	61.07	35.50	55.31
Best@2	37.50	55.17	37.50	50.05	45.50	60.43
Best@3	38.00	55.93	53.00	70.28	51.50	68.88
Equal	40.00	55.48	**58.00**	73.61	54.50	72.63
TAG	44.00	58.34	**58.00**	**75.73**	**55.50**	**73.71**

Loss Function	HotPotQA	2WikiQA	MuSiQue
CE	72.99	47.29	32.04
MSE	71.83	48.83	30.84
BCE	61.45	43.91	30.16
BT	**75.73**	**51.22**	**37.92**

Router	HotPotQA	2WikiQA	MuSiQue
Random	71.54	46.21	30.22
Equal	73.61	51.01	32.66
Weak	72.35	47.38	32.06
Fix-G	73.92	50.58	33.68
Fix-P	74.22	**51.36**	35.26
Strong	**75.73**	51.22	**37.92**

Table 7: (Top) Hybrid memory performance across different Top-k values. (Middle) F1 score of different training losses across three datasets. (Bottom) F1 score of different router types across three datasets.

Memory Structure	HotPotQA		2WikiQA		MuSiQue		Average	
	EM	F1	EM	F1	EM	F1	EM	F1
Summary	52.00	69.07	34.00	46.24	16.50	32.85	34.17	49.39
Chunks	52.00	68.48	33.50	**47.90**	17.00	32.30	34.17	49.56
Triples	33.50	45.75	24.50	36.98	7.50	17.35	21.83	33.36
Atomic Facts	38.50	51.12	29.50	43.10	10.00	18.04	26.00	37.42
Average	52.00	70.24	34.00	47.19	11.00	26.17	32.33	47.87
TAG	**53.50**	**71.01**	**35.00**	47.27	**17.50**	**32.42**	**35.33**	**50.23**

Table 8: Results of TAG and individual memory structures on Llama-3.1-8B-Instruct across three multi-hop QA benchmarks.

You are now an intelligent assistant tasked with meticulously extracting both key elements and triples from a long text.

1. Key Elements: The essential nouns (e.g., characters, times, events, places, numbers), verbs (e.g., actions), and adjectives (e.g., states, feelings) that are pivotal to the text's narrative.
2. Triples: Structured triplets in the format of "subject, relation, object". Each triple should represent a clear and concise fact, relation, or interaction within the observation. You should aim for simplicity and clarity, ensuring that each triplet has no more than 7 words.

Requirements:
#####
1. Ensure that all identified key elements are reflected within the corresponding atomic facts.
2. You should extract key elements and atomic facts comprehensively, especially those that are important and potentially query-worthy and do not leave out details.
3. Whenever applicable, replace pronouns with their specific noun counterparts (e.g., change I, He, She to actual names).
4. Ensure that the key elements and triples you extract are presented in the same language as the original text (e.g., English or Chinese).
5. Avoid Redundant Triples: Do not include irrelevant information like the current location of the agent (e.g., "you, are in, location") or placeholder entities such as "none."
6. Your answer format for each line should be: [Serial Number], [Atomic Facts], [List of Key Elements, separated with '|']
#####

Example:
#####
User: One day, a father and his little son .....
#
Assistant:
1. Father, went to, home | father | went to | home
2. Son, went to, home | son | went to | home
3. Father, accompanied by, son | father | accompanied by | son
4. ...
#####
#
Please strictly follow the above format. Let's begin.

Context:
{context}

(a) Prompt for generating knowledge triples.

You are a helpful assistant responsible for generating a comprehensive summary of the data provided below.
Given one or two atomic facts, and its original descriptions, all related to the atomic facts.
Please concatenate all of these into a single, comprehensive description. Make sure to include information collected from all the descriptions.
If the provided descriptions are contradictory, please resolve the contradictions and provide a single, coherent summary.
Make sure it is written in third person, and include the names so we have the full context.

#######
-Data-
Atomic facts:
{elements}

Original Description List:
{description_list}
#######

Output:

(b) Prompt for generating summaries.

Figure 7: Prompts for generating knowledge triples and summaries.

You are now an intelligent assistant tasked with meticulously extracting both key elements and atomic facts from a conversation history..

1. Key Elements: The essential nouns (e.g., characters, times, events, places, numbers), verbs (e.g., actions), and adjectives (e.g., states, feelings) that are pivotal to the text's narrative.
2. Atomic Facts: The smallest, indivisible facts, presented as concise sentences. These include propositions, theories, existences, concepts, and implicit elements like logic, causality, event sequences, interpersonal relationships, timelines, etc.

Requirements: #####
1. Ensure that all the atomic facts contain full and complete information, reflecting the entire context of the sentence without omitting any key details.
2. Ensure that all identified key elements are reflected within the corresponding atomic facts.
3. You should extract key elements and atomic facts comprehensively, especially those that are important and potentially query-worthy and do not leave out details.
4. Whenever applicable, replace pronouns with their specific noun counterparts (e.g., change I, He, She to actual names).
5. Ensure that the key elements and atomic facts you extract are presented in the same language as the original text (e.g., English or Chinese).
6. You should output a total of key elements and atomic facts that do not exceed 1024 tokens.
7. Your answer format for each line should be: [Serial Number], [Atomic Facts], [List of Key Elements, separated with '|']
#####

Example:
#####
Conversation:
1. Caroline said, "Woohoo Melanie! I passed the adoption agency interviews last Friday! I'm so excited and thankful. This is a big move towards my goal of having a family."
2. Melanie said, "Congrats, Caroline! Adoption sounds awesome. These figurines I bought yesterday remind me of family love. Tell me, what's your vision for the future?"
and shared a photo of a couple of wooden dolls sitting on top of a table.

Atomic Facts and Key Elements:
1. Caroline passed the adoption agency interviews last Friday. | Caroline | adoption agency interviews | last Friday
2. Caroline is excited and thankful for passing the adoption agency interviews. | Caroline | excited | thankful | adoption agency interviews
3. Passing the adoption agency interviews is a big move towards Caroline's goal of having a family. | Caroline | adoption agency interviews | goal | having a family
4. Melanie congratulated Caroline on passing the adoption agency interviews. | Melanie | Caroline | adoption agency interviews | Congratulations
5. Melanie thinks that adoption sounds awesome. | Melanie | Adoption | awesome
6. Melanie bought figurines yesterday. | Melanie | figurines | yesterday
7. The figurines Melanie bought remind her of family love. | Melanie | figurines | family love
8. Melanie asked Caroline about her vision for the future. | Melanie | Caroline | vision for the future
9. Melanie shared a photo of wooden dolls sitting on a table. | Melanie | wooden dolls | table | photo
#####
#
Please strictly follow the above format. Let's begin.

Conversation:
{conversation}

Atomic Facts and Key Elements:

Figure 8: Prompt for generating atomic facts.