G-Memory: Tracing Hierarchical Memory for Multi-Agent Systems

Guibin Zhang[*1], **Muxin Fu**[*2], **Kun Wang**[3†], **Guancheng Wan**[4], **Miao Yu**[5], **Shuicheng Yan**[1†]
[1]NUS, [2]Tongji University, [3]NTU, [4]WHU, [5]A*STAR
[*] Equal Contribution, [†] Corresponding author
wang.kun@ntu.edu.sg, yansc@comp.nus.edu.sg

Abstract

Large language model (LLM)-powered multi-agent systems (MAS) have demonstrated cognitive and execution capabilities that far exceed those of single LLM agents, yet their capacity for self-evolution remains hampered by underdeveloped memory architectures. Upon close inspection, we are alarmed to discover that prevailing MAS memory mechanisms (1) are overly simplistic, completely disregarding the nuanced inter-agent collaboration trajectories, and (2) lack cross-trial and agent-specific customization, in stark contrast to the expressive memory developed for single agents. To bridge this gap, we introduce **G-Memory**, a hierarchical, agentic memory system for MAS inspired by organizational memory theory [1], which manages the lengthy MAS interaction via a three-tier graph hierarchy: insight, query, and interaction graphs. Upon receiving a new user query, **G-Memory** performs bi-directional memory traversal to retrieve both *high-level, generalizable insights* that enable the system to leverage cross-trial knowledge, and *fine-grained, condensed interaction trajectories* that compactly encode prior collaboration experiences. Upon task execution, the entire hierarchy evolves by assimilating new collaborative trajectories, nurturing the progressive evolution of agent teams. Extensive experiments across five benchmarks, three LLM backbones, and three popular MAS frameworks demonstrate that **G-Memory improves success rates in embodied action and accuracy in knowledge QA by up to** 20.89% **and** 10.12%**,** respectively, without any modifications to the original frameworks. Our codes are available at https://github.com/bingreeky/GMemory.

1 Introduction

As Large Language Models (LLMs) continue to redefine the frontier of artificial intelligence, *LLM-driven agents* have exhibited unprecedented prowess in perception [2, 3, 4, 5], planning [6, 7, 8], reasoning [9, 10], and action [11, 12], which have catalyzed remarkable progress across diverse downstream domains, including code generation [13, 14], data analysis [15], embodied tasks [16] and autonomous driving [3, 17, 18]. Building upon the impressive competencies of single agents, LLM-based Multi-Agent Systems (MAS) have been demonstrated to push the boundaries of single model capacity [19, 20, 21]. Similar to collective intelligence arising from human social collaboration [22, 23, 24], MAS orchestrates multiple agents [25, 26, 27], whether through cooperation [28, 29, 30, 31] or competition [32, 33, 34], to transcend the cognitive and specialized limitations of solitary agents.

Self-Evolving Agents. What especially characterizes LLM agents is their *self-evolving capacity*, *i.e.*, the ability to continuously adapt and improve through interactions with the environment, as seen in prior works where such adaptability has led to two- to three-fold quantitative improvements [35]. The central driving force behind such self-evolving nature is **memory mechanism** of agents [36, 37], which parallels human abilities to accumulate knowledge, process past experiences, and retrieve

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

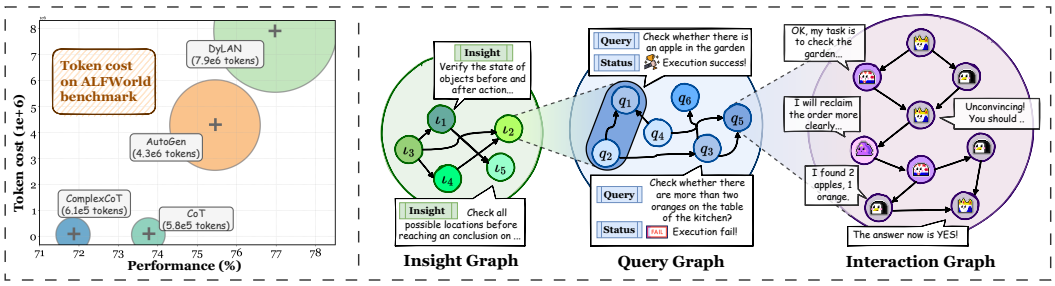


Figure 1: (***Left***) We report the token cost of several single-agent and MAS baselines on ALFWorld benchmark; (***Right***) The overview of `G-Memory`'s three-tier hierarchical memory architecture, encompassing the insight graph, query graph and interaction (utterance) graph.

relevant information. Previous successful memory mechanism designs, including both inside-trial memory (*i.e.*, context retained within solving one single query) and cross-trial memory (*i.e.*, experience accumulated across multiple tasks) [38], have empowered agents to excel in diverse applications such as personalized chat [36, 39, 40], recommendation [41], embodied action [42, 16], and social simulation [19, 43, 44], enabling them to evolve into experiential learners that effectively leverage past experiences and world knowledge.

Self-Evolving MAS. However, such self-evolving capacity remains largely absent in multi-agent systems. Most existing MAS are still constrained by manually defined workflows, such as the Standard Operating Procedures (SOP) in MetaGPT [21] and ChatDev [45], or rely on pre-defined communication topologies in MacNet [46] and AgentPrune [30]. More recent automated MASs, such as GPTSwarm [47], ADAS [48], AFlow [49], and MaAS [50] have made it to automatically optimize inter-agent topologies or prompts, which, nevertheless, ultimately yield giant and cumbersome MAS architectures, lacking the agility to self-adjust with accumulated collaboration experience.

Memory for MAS. The absence of the aforementioned self-evolving capacity is, in fact, rooted in the lack of memory mechanisms specifically tailored for MAS. One may challenge this claim from two perspectives: ❶ *Do existing MASs lack memory mechanisms altogether?* Not entirely. Classical MAS frameworks such as MetaGPT, ChatDev, and Exchange-of-Thought [51] incorporate memory-related designs. However, these are often limited to inside-trial memory [51], while cross-trial memory, if present, remains rudimentary—typically involving the transmission of overly condensed artifacts (*e.g.*, final solutions or execution results) [21, 45, 46], and failing to enable meaningful learning from collaborative experience. ❷ *Why not directly transfer existing single-agent memory mechanisms to MAS?* Unfortunately, such a transfer is far from straightforward. The inherent nature of MAS, *i.e.*, multi-turn orchestration across multiple agents [26, 27], leads to substantially longer task-solving trajectories compared to single-agent settings (up to 10× more tokens, as demonstrated by Figure 1 (*Left*)). This poses a significant challenge to traditional retrieval-based memory designs [36, 37, 16], as naive feeding of the entire long-context trajectory without proper abstraction from a collaborative perspective offers little benefit. Given the aforementioned challenges, a natural question arises:

How can we design a memory mechanism capable of storing, retrieving, and managing the lengthy interaction history of multi-agent systems, such that agent teams can benefit from concise and instructive experience and insights?

The Present Work: `G-Memory`. In response to the above question, we introduce a *Graph-based Agentic Memory Mechanism for LLM-based Multi-Agent Systems*, dubbed `G-Memory`, which manages the complex and lengthy interaction history of MAS through a three-tier hierarchical graph structure:

- ✱ **Insight Graph**, which abstracts generalizable insights from historical experience;
- ✱ **Query Graph**, which encodes meta-information of task queries and their connectivity;
- ✱ **Interaction Graph**, which stores fine-grained textual communication logs among agents.

Figure 1 (*Right*) visualizes these structures, and their formal definitions are placed in Section 3. When a new query arrives, `G-Memory` efficiently retrieves relevant query records by leveraging the topology of the query graph, and then traverses *upward* (*i.e.*, query→insight graph) to extract associated high-level insights and *downward* (*i.e.*, query→interaction graph) to identify core interaction subgraphs that are most pertinent to the task at hand, thereby mitigating information overload. Based on the

retrieved memory, `G-Memory` offers actionable guidance to the MAS, *e.g.*, division of labor, task decomposition, and lessons from past failures. Upon the completion of a task, all three levels of the memory hierarchy are updated in an agentic manner, with newly distilled insights, enriched query records, detailed MAS trajectories, and their level of detailed associations. Through this refinement, `G-Memory` functions as a plug-and-play module that can be seamlessly embedded into mainstream MAS frameworks, empowering evolving inter-agent collaboration and collective intelligence.

Our contributions are summarized as follows:

- ❶ **Bottleneck Identification.** We conduct a thorough review of existing multi-agent systems and identify a fundamental bottleneck in their self-evolving capabilities, which is largely attributed to the oversimplified memory architectures.
- ❷ **Practical Solution.** We propose `G-Memory`, a hierarchical agentic memory architecture for MAS, which models complex and prolonged inter-agent collaboration through a three-tier structure comprising insight, query, and interaction graphs.
- ❸ **Experimental Evaluation.** Extensive experiments across five benchmarks show that `G-Memory` is **(I)** ***high-performing***, improving state-of-the-art MAS by up to 20.89% and 10.12% on embodied action and knowledge QA tasks, respectively; and **(II)** ***resource-friendly***, maintaining comparable or even lower token usage than mainstream memory designs.

2 Related Works

Single-Agent Memory. Memory serves as a primary driving force for agents to accumulate experiences and explore the world through interactions with the environment [52, 53, 54, 55]. It plays a critical role in both *task-solving* and *social simulation* LLM agents, and this work primarily focuses on the former. Early research on agent memory was confined to simple inside-trial memory, mainly addressing limitations posed by the LLM context window in chatbot applications, including MemoryBank [36], ChatDB [39], MemoChat [40], and MemGPT [37], which typically adopt retrieval-augmented generation (RAG)-style, similarity-based chunk retrieval. Subsequent developments have progressed toward more cognitively inspired memory architectures, including (1) memory scope extended to cross-trial memory like ExpeL [42] and Synapse [56]; (2) application domains broadened to include computer control [56], embodied action [57], scientific discovery [58], coding and reasoning [59]; and (3) management techniques evolved from coarse-grained textual similarity toward more sophisticated abstraction and summarization of acquired knowledge and experiences [19], as seen in A-Mem [60], Mem0 [61] and MemInsight [62]. More discussions are in Appendix D.

Memory in Multi-agent System. However, the memory mechanisms tailored for MAS remain markedly underexplored. Some representative frameworks, such as LLM-Debate [20, 33] and Mixture-of-Agent [63], omit memory components altogether. Others merely adopt simplistic inside-trial memory schemes [46, 51]. Even in frameworks that attempt cross-trial memory [45], the memory is merely compressed as the final outcome artifacts, overlooking the nuanced agent interactions. Collectively, there is a pressing need for a principled memory architecture that can capture, organize, and retrieve the inherently intricate task-solving processes unique to MAS [38].

LLM-based Multi-Agent Systems. Our work focuses on *task-solving* MAS, which, unlike their single-agent counterparts, often lack the capacity for continual evolution through interaction with the environment [64, 65]. Early frameworks such as AutoGen [13], CAMEL [24], and AgentVerse [66] rely entirely on pre-defined workflows. More recent efforts [67, 68, 49, 48, 69, 31] introduce a degree of adaptivity by generating dynamic MAS in response to environmental feedback. However, such evolution is often *one-shot*: for example, AFlow [49] employs Monte Carlo Tree Search to construct a complex MAS tailored to a specific task domain, which yet lacks the capacity to evolve with increasing task exposure or transfer across domains [50, 70]. From this perspective, constructing MAS with genuine self-evolving capabilities remains an open and challenging research frontier.

3 Preliminary

In this section, we establish the notation and formalize key concepts of multi-agent systems and `G-Memory`'s hierarchical memory architecture.

Multi-agent System Formalization. Consider a multi-agent framework represented by a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $|\mathcal{V}| = N$ is the number of agents and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ defines their communication

channels. Each node $C_i \in \mathcal{V}$ corresponds to an individual agent described by the quadruple:

$$C_i = (\mathsf{Base}_i, \mathsf{Role}_i, \mathsf{Mem}_i, \mathsf{Plugin}_i), \tag{1}$$

where Base_i denotes the underlying large language model instance, Role_i specifies the agent's designated role or persona, Mem_i encapsulates its memory state, including past interactions or external knowledge stores, and Plugin_i is the set of auxiliary tools (*e.g.*, web-search engine).

Upon receiving a user query Q, the system evolves through T synchronous communication epochs. At each epoch t, we derive a topological ordering $\pi = [\pi_1, \ldots, \pi_N]$ of the nodes such that if there is an edge from π_j to π_k, then $j < k$, which guarantees that every agent processes its inputs only after all its predecessors have acted. For each agent C_i in π, its output at iteration t is computed as:

$$r_i^{(t)} = C_i\Big(P_{\text{sys}}^{(t)}, Q, \{r_j^{(t)} : C_j \in \mathcal{N}^-(C_i)\}\Big),$$

where: $r_i^{(t)}$ denotes the response generated by C_i (which may include reasoning steps, intermediate analyses, or final proposals), $P_{\text{sys}}^{(t)}$ comprises global instructions (including each agent's $\mathcal{R}_i$), $\mathcal{N}^-(C_i)$ is the set of in-neighbors of C_i, whose outputs serve as contextual inputs. After all agents have acted, a global aggregation operator $\mathcal{A}$ fuses the collection of responses into an interim solution $a^{(t)}$:

$$a^{(t)} = \mathcal{A}(r_1^{(t)}, \ldots, r_N^{(t)}).$$

Common implementations for $\mathcal{A}$ include majority voting schemes [47], hierarchical summarization via dedicated aggregator agents [13, 30], or simply adopting the final agent's output as the answer [46]. These epochs iterate for $t = \{1, \ldots, T\}$ until either a preset limit is reached or an early-stopping criterion is met [71], producing the final response $a^{(T)}$ to the query Q.

Memory Architecture. Our proposed `G-Memory` orchestrates and manages the memory of multi-agent systems via the following three hierarchical graph structures:

[✱] Interaction Graph (Utterance Graph). For query Q, let $\mathcal{G}_{\mathsf{inter}}^{(Q)} = (\mathcal{U}^{(Q)}, \mathcal{E}_{\mathsf{u}}^{(Q)})$ denote its interaction trajectory, where (i) nodes $\mathcal{U}^{(Q)} = \{u_i\}$ represent atomic utterances, with each $u_i \triangleq (\mathcal{A}_i, m_i)$ containing $\mathcal{A}_i \in \mathcal{V}$ (speaking agent), and m_i (textual content), (ii) Edges $\mathcal{E}_{\mathsf{u}}^{(Q)} \subseteq \mathcal{U}^{(Q)} \times \mathcal{U}^{(Q)}$ follow temporal relationships: $(u_j, u_k) \in \mathcal{E}_{\mathsf{u}}^{(Q)} \iff u_j$ is transmitted to and inspires u_k.

[✱] Query Graph. The query graph, storing previously tackled queries and metadata, is as follows:

$$\mathcal{G}_{\mathsf{query}} = (\mathcal{Q}, \mathcal{E}_{\mathsf{q}}) = \left(\{Q_i, \Psi_i, \mathcal{G}_{\mathsf{inter}}^{(Q_i)}\}_{i=1}^{|\mathcal{Q}|}, \mathcal{E}_{\mathsf{q}}\right), \tag{2}$$

where $\mathcal{Q} = \{q_i\}$ is the node set, node $q_i \triangleq (Q_i, \Psi_i, \mathcal{G}_{\mathsf{inter}}^{(Q_i)})$ is composed of the original query Q_i, task status $\Psi_i \in \{\mathsf{Failed}, \mathsf{Resolved}\}$, and its associated interaction graph $\mathcal{G}_{\mathsf{inter}}^{(Q_i)}$. The edges $\mathcal{E}_{\mathsf{q}} \subseteq \mathcal{Q} \times \mathcal{Q}$ encode semantic relationships between queries. The query graph enables retrieval beyond coarse metrics such as embedding similarity, with its meticulous topology.

[✱] Insight Graph. The highest-level insight graph is featured as follows:

$$\mathcal{G}_{\mathsf{insight}} = (\mathcal{I}, \mathcal{E}_{\mathsf{i}}) = \Big(\langle \underbrace{\kappa_k, \Omega_k}_{\iota_k} \rangle_{k=1}^{|\mathcal{I}|}, \mathcal{E}_{\mathsf{i}}\Big), \tag{3}$$

where the node set $\mathcal{I} = \{\iota_k\}$ represents distilled insights, each node ι_k is composed of the insight content κ_k and the set of supporting queries $\Omega_k \subseteq \mathcal{Q}$. The edges $\mathcal{E}_{\mathsf{i}} \subseteq \mathcal{I} \times \mathcal{I} \times \mathcal{Q}$ forming hyper-connections where (ι_m, ι_n, q_j) indicates insight ι_m contextualizes ι_n through query q_j.

4 G-Memory

This section outlines the management workflow of `G-Memory`, as illustrated in Figure 2. Specifically, upon the arrival of a new query Q, `G-Memory` first conducts coarse-grained retrieval to identify pertinent trajectory records (▷ Section 4.1). It then performs bi-directional hierarchical memory traversal: upward to retrieve collective cognitive insights, and downward to distill concrete procedural trajectories (▷ Section 4.2). After the memory-augmented MAS completes the query execution, the hierarchical memory architecture is jointly updated based on environmental feedback, thereby achieving the institutionalization of group knowledge (▷ Section 4.3).

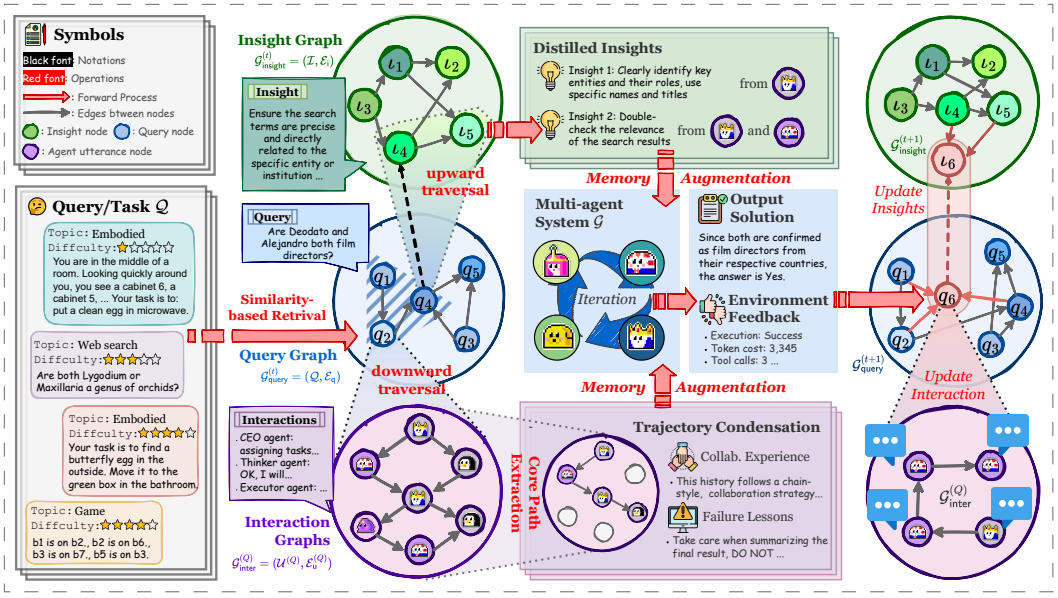


Figure 2: The overview of our proposed **G-Memory**.

4.1 Coarse-grained Memory Retrieval

As a plug-in designed for seamless integration into mainstream MAS, **G-Memory** is triggered when the MAS $\mathcal{G}$ encounters a new user query Q. As emphasized in organizational memory theory [1], efficient knowledge retrieval typically begins with broadly relevant schemas prior to more fine-grained access. Following this principle, **G-Memory** first performs a coarse-grained similarity-based retrieval over the query graph $\mathcal{G}_{\text{query}}$ to efficiently obtain a sketched set of queries $\mathcal{Q}^{\mathcal{S}}$:

$$\mathcal{Q}^{\mathcal{S}} = \underset{q_i \in \mathcal{Q} \text{ s.t. } |\mathcal{Q}^{\mathcal{S}}|=k}{\arg \text{top-k}} \left(\frac{\mathbf{v}(Q) \cdot \mathbf{v}(q_i)}{|\mathbf{v}(Q)|\,|\mathbf{v}(q_i)|} \right), \tag{4}$$

where $\mathbf{v}(\cdot)$ maps queries into fixed-length embeddings using models such as MiniLM [72]. While Equation (4) retrieves semantically similar historical queries, the similarity may be only superficial or noisy. Therefore, **G-Memory** further enlarges the relevant set via **hop expansion** on the query graph:

$$\tilde{\mathcal{Q}}^{\mathcal{S}} = \mathcal{Q}^{\mathcal{S}} \cup \left\{ Q_k \in \mathcal{Q} \mid \exists Q_j \in \mathcal{Q}^{\mathcal{S}},\ Q_k \in \mathcal{N}^{+}(Q_j) \cup \mathcal{N}^{-}(Q_j) \right\}, \tag{5}$$

where $\tilde{\mathcal{Q}}^{\mathcal{S}}$ is augmented with the 1-hop neighbors of $\mathcal{Q}^{\mathcal{S}}$ on the query graph $\mathcal{G}_{\text{query}}$, and $\mathcal{N}^{+}(\cdot)$ and $\mathcal{N}^{-}(\cdot)$ denote the out-neighborhood and in-neighborhood of node Q_j, respectively. However, it is suboptimal to directly feed these relevant records as input akin to certain single-agent memory systems [40, 37]. On one hand, the excessive context length may overwhelm the LLM; on the other hand, agents in MAS play distinct roles and should be assigned *specialized* memory tailored to their functions. To address this, the next section introduces a bi-directional processing scheme in **G-Memory** that operates over both abstract and fine-grained memory levels.

4.2 Bi-directional Memory Traversal

Subsequent to identifying the expanded set of relevant query nodes $\tilde{\mathcal{Q}}^{\mathcal{S}}$ within $\mathcal{G}_{\text{query}}$, **G-Memory** executes a **bi-directional memory traversal** to furnish multi-granularity memory support. Specifically, **G-Memory** first performs an *upward traversal* ($\mathcal{G}_{\text{query}} \rightarrow \mathcal{G}_{\text{insight}}$), retrieving insight nodes that may provide high-level guidance for the current task:

$$\mathcal{I}^{\mathcal{S}} = \Pi_{\mathcal{Q}\to\mathcal{I}}(\tilde{\mathcal{Q}}^{\mathcal{S}}),\ \Pi_{\mathcal{Q}\to\mathcal{I}}(\mathcal{S}_q) \triangleq \{\iota_k \in \mathcal{I} \mid \Omega_k \cap \mathcal{S}_q \neq \emptyset\}, \tag{6}$$

where $\Pi_{\mathcal{Q}\to\mathcal{I}}$ is a query-to-insight projector that identifies all the insight nodes whose supporting query sets intersect with the input query set, and the retrieved insights $\mathcal{I}^{\mathcal{S}}$ encapsulate distilled, generalized knowledge potentially relevant for orienting the MAS $\mathcal{G}$'s strategic approach to Q.

Beyond generalized insights, the fine-grained textual interaction history of the MAS is equally valuable, as it reveals the underlying reasoning patterns that led to successful or failed collaborations [67, 73, 74]. To utilize these concisely, in the downward traversal ($\mathcal{G}_{\text{query}} \rightarrow \mathcal{G}_{\text{interaction}}$),

`G-Memory` employs an LLM-facilitated graph sparsifier $\mathcal{S}_{\text{LLM}}(\cdot,\cdot)$ to extract the core subgraph that encapsulates essential inter-agent collaboration:

$$\{\hat{\mathcal{G}}_{\text{inter}}^{Q_i}\}_{i=1}^{|M|} = \Big\{\mathcal{S}_{\text{LLM}}(\mathcal{G}_{\text{inter}}^{(Q_j)}, Q) \mid q_j \in \underset{\{q'_k \in \tilde{\mathcal{Q}}^{\mathcal{S}}\} \text{ s.t. } |\cdot|=M}{\text{argtop-M}} \mathcal{R}_{\text{LLM}}(Q, q'_k)\Big\}, \tag{7}$$

where $\mathcal{R}_{\text{LLM}}(Q, q_j)$ rates the relevancy of historical queries w.r.t. Q, and the sparsifier $\mathcal{S}_{\text{LLM}}(\mathcal{G}_{\text{inter}}^{(Q_j)}, Q)$ constructs a sparsified graph $\hat{\mathcal{G}}_{\text{inter}}^{(Q_j)} = (\hat{\mathcal{U}}^{(Q_j)}, \hat{\mathcal{E}}_{\text{u}}^{(Q_j)})$ from the original $\mathcal{G}_{\text{inter}}^{(Q_j)}$ by identifying and retaining dialogue elements. Please refer to Appendix C for their implementations.

Upon completing the bi-directional traversal, we obtain both generalizable insights ($\mathcal{I}^{\mathcal{S}}$) and detailed collaborative trajectories ($\{\hat{\mathcal{G}}_{\text{inter}}^{Q_i}\}_{i=1}^{|M|}$). `G-Memory` then proceeds to provide specialized memory support for each agent $\mathcal{C} \in \mathcal{V}$ within the MAS $\mathcal{G}$.

$$\text{Mem}_i \leftarrow \Phi\left(\mathcal{I}^{\mathcal{S}}, \{\hat{\mathcal{G}}_{\text{inter}}^{Q_i}\}_{i=1}^{|M|}; \text{Role}_i, Q\right), \ \forall C_i = (\text{Base}_i, \text{Role}_i, \text{Mem}_i, \text{Plugin}_i) \in \mathcal{V}, \tag{8}$$

where the operator $\Phi(\cdot;\cdot)$ evaluates the utility and relevance of each insight $\iota_k \in \mathcal{I}^{\mathcal{S}}$ and sparsified interaction graph $\hat{\mathcal{G}}_{\text{inter}}^{(Q_j)}$ concerning the agent's specific role Role_i and the task Q (see Appendix C). Based on this evaluation, Φ intializes each agent's internal memory state Mem_i with filtered insights, interaction snippets, summaries thereof, equipping it with pertinent historical context before it participates in the subsequent reasoning epochs of the MAS. It is worth noting that `G-Memory` is invoked at the onset of solving query Q in our implementation. However, practitioners may flexibly configure more fine-grained invocation strategies, such as at the beginning of each MAS dialogue round or selectively for specific agents, based on their needs.

4.3 Hierarchy Memory Update

After completing memory augmentation for each agent, the system $\mathcal{G}$ is executed as outlined in Section 3, yielding a final solution $a^{(T)}$ and receiving environmental feedback, including execution status $\Psi_i \in \{\text{Failed}, \text{Resolved}\}$, token usage, and other performance metrics. Subsequently, `G-Memory` updates its hierarchical memory architecture to incorporate this new query. At the **interaction level**, `G-Memory` traces each agent's utterances to construct the interaction graph $\mathcal{G}_{\text{inter}}^{(Q)}$, which is then stored. At the **query level**, a new query node is instantiated and added to the query graph $\mathcal{Q}_{\text{query}}$:

$$\begin{gathered} q_{\text{new}} \leftarrow (Q, \Psi, \mathcal{G}_{\text{inter}}^{(Q)}),\ \mathcal{N}_{\text{conn}} \leftarrow \mathcal{Q}^{\mathcal{R}} \cup \Big(\bigcup_{\iota_k \in \mathcal{I}^{\mathcal{S}}} \Omega_k\Big), \\ \mathcal{E}_{\text{new}} \leftarrow \{(q_n, q_{\text{new}}) \mid q_n \in \mathcal{N}_{\text{conn}}\},\ \mathcal{G}_{\text{query}}^{\text{next}} \leftarrow (\mathcal{Q} \cup \{q_{\text{new}}\}, \mathcal{E}_{\text{q}} \cup \mathcal{E}_{\text{new}}), \end{gathered} \tag{9}$$

where edges are established between q_{new} and (ii) the set $\mathcal{Q}^{\mathcal{R}}$ containing the top-M relevant historical queries identified in Equation (7), and (ii) the set of queries $\bigcup_{\iota_k \in \mathcal{I}_{\text{ret}}} \Omega_k$ that support the insights $\mathcal{I}^{\mathcal{S}}$ utilized for solving Q. $\mathcal{G}_{\text{query}}^{\text{next}}$ denotes the updated query graph.

Finally, at the **insight level**, `G-Memory` integrates the learning from the completed query Q into the insight graph $\mathcal{G}_{\text{insight}} = (\mathcal{I}, \mathcal{E}_{\text{i}})$. First, possible new insights summarizing the experience are generated and structurally linked via a summarization function $\mathcal{J}(\cdot,\cdot)$ (see prompt in Appendix C) as follows:

$$\begin{gathered} \iota_{\text{new}} = (\mathcal{J}(\mathcal{G}_{\text{inter}}^{(Q)}, \Psi), \{q_{\text{new}}\}),\ \mathcal{E}_{\text{i, new}} \leftarrow \{(\iota_k, \iota_{\text{new}}, q_{\text{new}}) \mid \iota_k \in \mathcal{I}^{\mathcal{S}}\} \\ \mathcal{G}'_{\text{insight}} \leftarrow (\mathcal{I} \cup \{\iota_{\text{new}}\}, \mathcal{E}_{\text{i}} \cup \mathcal{E}_{\text{i, new}}) \end{gathered} \tag{10}$$

where edges are added to connect the previously utilized insights which inspires the completion of Q in Equation (6). Afterward, the supporting query sets (Ω_k) for the utilized insights ($\mathcal{I}^{\mathcal{S}}$) are updated to include q_{new}, reflecting their relevance to this successful (or failed) application:

$$\begin{gathered} \mathcal{I}^{\text{next}} \leftarrow (\mathcal{I} \setminus \mathcal{I}_{\text{ret}}) \cup \{(\kappa_k, \Omega_k \cup \{q_{\text{new}}\}) \mid \iota_k = (\kappa_k, \Omega_k) \in \mathcal{I}_{\text{ret}}\} \cup \{\iota_{\text{new}}\} \\ \mathcal{G}_{\text{insight}}^{\text{next}} \leftarrow (\mathcal{I}^{\text{next}}, \mathcal{E}_{\text{i}} \cup \mathcal{E}_{\text{i, new}}), \end{gathered} \tag{11}$$

where the final node set $\mathcal{I}^{\text{next}}$ incorporates the new insight and the updated versions of the utilized insights, and the resulting graph $\mathcal{G}_{\text{insight}}^{\text{next}}$ thus encapsulates the integrated knowledge. This continuous update cycle across all hierarchical levels enables `G-Memory` to learn and adaptively refine its collective memory based on ongoing experience.

Table 1: Performance comparison with single/multi-agent memory architectures on five benchmarks. The underlying LLM backbone is `GPT-4o-mini`. We highlight the best and second best results.

MAS	Memory	ALFWorld	SciWorld	PDDL	HotpotQA	FEVER	Avg.
AutoGen COLM 2024	No-memory	77.61 ↑0.00	54.49 ↑0.00	23.53 ↑0.00	28.57 ↑0.00	57.13 ↑0.00	48.27 ↑0.00
	Voyager	85.07 ↑7.46	62.36 ↑7.87	24.56 ↑1.03	32.32 ↑3.75	63.27 ↑6.14	53.52 ↑5.25
	MemoryBank	74.96 ↓2.65	53.11 ↓1.38	20.41 ↓3.12	33.67 ↑5.10	61.22 ↑4.09	48.67 ↑0.40
	Generative	86.36 ↑8.75	61.19 ↑6.70	25.53 ↑2.00	31.63 ↑3.06	60.20 ↑3.07	52.98 ↑4.71
	MetaGPT	81.34 ↑3.73	61.91 ↑7.42	21.63 ↓1.90	32.67 ↑4.10	62.67 ↑5.54	52.04 ↑3.77
	ChatDev	79.85 ↑2.24	50.96 ↓3.53	16.65 ↓6.88	24.49 ↓4.08	59.18 ↑2.05	46.23 ↓2.04
	MacNet	76.55 ↓1.06	55.44 ↑0.95	22.94 ↓0.59	28.36 ↓0.21	60.87 ↑3.74	48.83 ↑0.56
	`G-Memory` (Ours)	88.81 ↑11.20	67.40 ↑12.91	27.77 ↑4.24	35.67 ↑7.10	66.24 ↑9.11	57.18 ↑8.91
DyLAN COLM 2024	No-memory	56.72 ↑0.00	55.38 ↑0.00	11.62 ↑0.00	31.69 ↑0.00	60.20 ↑0.00	43.12 ↑0.00
	Voyager	66.42 ↑9.70	62.83 ↑7.45	15.10 ↑3.48	32.64 ↑0.95	62.24 ↑2.04	47.85 ↑4.73
	MemoryBank	55.22 ↓1.50	54.74 ↓0.64	8.08 ↓3.54	29.59 ↓2.10	59.13 ↓1.07	41.35 ↓1.77
	Generative	67.91 ↑11.19	64.16 ↑8.78	13.87 ↑2.25	29.29 ↓2.40	62.30 ↑2.10	47.51 ↑4.39
	MetaGPT-M	69.40 ↑12.68	62.37 ↑6.99	14.45 ↑2.83	32.34 ↑0.65	60.20 ↑0.00	47.75 ↑4.63
	ChatDev-M	46.27 ↓10.45	53.35 ↓2.03	10.75 ↓0.87	22.45 ↓9.24	58.33 ↓1.87	38.23 ↓4.89
	MacNet-M	53.44 ↓3.28	54.32 ↓1.06	12.11 ↑0.49	30.12 ↓1.57	61.10 ↑0.90	42.22 ↓0.90
	`G-Memory` (Ours)	70.90 ↑14.18	65.64 ↑10.26	18.95 ↑7.33	34.69 ↑3.00	64.22 ↑4.02	50.88 ↑7.76
MacNet ICLR 2025	No-memory	51.49 ↑0.00	57.53 ↑0.00	12.18 ↑0.00	28.57 ↑0.00	60.29 ↑0.00	42.01 ↑0.00
	Voyager	61.94 ↑10.45	64.53 ↑7.00	14.06 ↑1.88	32.65 ↑4.08	62.54 ↑2.25	47.14 ↑5.13
	MemoryBank	50.00 ↓1.49	60.15 ↑2.62	8.64 ↓3.54	33.67 ↑5.10	61.22 ↑0.93	42.74 ↑0.73
	Generative	62.69 ↑11.20	65.49 ↑7.96	7.92 ↓4.26	29.59 ↑1.02	63.27 ↑2.98	45.79 ↑3.78
	MetaGPT-M	63.70 ↑12.21	65.27 ↑7.74	16.03 ↑3.85	31.00 ↑2.43	59.33 ↓0.96	47.07 ↑5.06
	ChatDev-M	49.25 ↓2.24	56.58 ↓0.95	13.51 ↑1.33	29.00 ↑0.43	59.18 ↓1.11	41.50 ↓0.51
	MacNet-M	53.44 ↑1.95	56.14 ↓1.39	13.59 ↑1.41	27.89 ↓0.68	59.20 ↓1.09	42.05 ↑0.04
	`G-Memory` (Ours)	67.16 ↑15.67	68.11 ↑10.58	24.33 ↑12.15	35.69 ↑7.12	64.44 ↑4.15	51.95 ↑9.94

5 Experiment

In this section, we conduct extensive experiments to answer: (**RQ1**) How does `G-Memory` perform compared to existing single/multi-agent memory architectures? (**RQ2**) Does `G-Memory` incur excessive resource overhead? (**RQ3**) How sensitive is `G-Memory` to its key components and parameters?

5.1 Experiment Setup

Datasets and Benchmarks. To thoroughly evaluate the effectiveness of `G-Memory`, we adopt five widely-adopted benchmarks across three domains: **(1) Knowledge reasoning**, including HotpotQA [75] and FEVER [76]; **(2) Embodied action**, including ALFWorld [77] and SciWorld [78]; **(3) Game**, namely PDDL [79]. Details on these benchmarks are in Appendix A.1.

Baselines. We select four representative single-agent memory baselines, including non-memory, Voyager [16], MemoryBank [36], and Generative Agents [19], as well as three multi-agent memory implementations from MetaGPT [21], ChatDev [45], and MacNet [46], denoted as MetaGPT-M, ChatDev-M, and MacNet-M, respectively. Details are in Appendix A.2.

MAS and LLM Backbones. We select three representative multi-agent frameworks to integrate with `G-Memory` and the baselines, including **AutoGen** [13], **DyLAN** [71], and **MacNet** [46]. More details on the MAS setups are placed in Appendix A.3. For instantiating these MAS frameworks, we adopt two open-source LLMs, `Qwen-2.5-7b` and `Qwen-2.5-14b`, as well as one proprietary LLM, `gpt-4o-mini`. The deployment of `Qwen` series is via local instantiation using Ollama[1], and `GPT` models are accessed via OpenAI APIs.

Parameter Configurations. We implement the embedding function $\mathbf{v}(\cdot)$ in Equation (4) with ALL-MINILM-L6-V2 [80]. The number of the most relevant interaction graphs M in Equation (7) is set among $\{2, 3, 4, 5\}$, and the number of relevant queries k in Equation (4) is set among $\{1, 2\}$. The detailed ablation study on hyper-parameters is placed in Section 5.4.

5.2 Main Results (RQ1)

Tables 1, 2 and 3 comprehensively report the performance of different memory architectures across three LLM backbones and three MAS frameworks. We summarize the key observations as follows:

[1] http://github.com/ollama/ollama

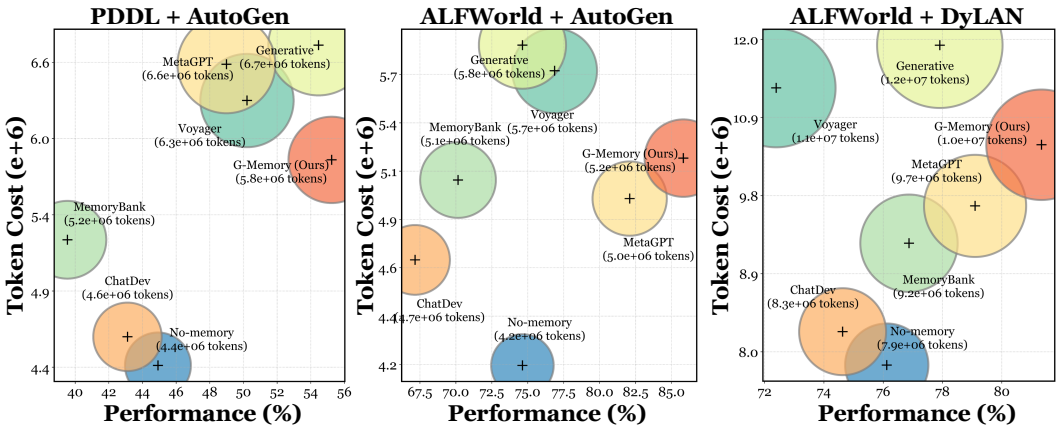


Figure 3: Cost analysis of `G-Memory`. We showcase the performance versus the overall system token cost when combined with different memory architectures.

Takeaway ❶: `G-Memory` consistently improves performance across all task domains and MAS frameworks. As shown in Table 2, when integrated with AutoGen and MacNet (powered by `Qwen-2.5-7b`), **`G-Memory`** surpasses the best-performing single-/multi-agent memory baselines by an average of 6.8% and 5.5%, respectively. With the more capable `Qwen-2.5-14b`, the improvement is even more pronounced: in Table 3, **`G-Memory`** boosts MacNet's performance on ALFWorld from 58.21% to 79.10%, achieving a substantial 20.89% gain.

Takeaway ❷: Multi-agent systems demand specialized memory designs. A thorough examination of existing baselines reveals a surprising insight: most memory mechanisms fail to consistently benefit MAS settings. In Table 2, baselines such as Voyager and MemoryBank degrade AutoGen's performance on PDDL by as much as 4.17% and 1.34%, respectively. We attribute this to the inability of these methods to provide agent role-specific memory support, which is essential in the PDDL strategic game tasks, where effective division of labor is critical to success. Even MAS-oriented designs, such as ChatDev-M, result in a 2.32% performance drop when applied to MacNet+SciWorld. We attribute this to ChatDev-M's narrow memory scope—storing only the execution results of past queries, which provides limited utility in embodied action environments. These findings highlight the necessity of **`G-Memory`**'s core characteristics: role-specific memory cues, abstracted high-level insights, and trajectory condensation—all of which are critical for effective memory in MAS.

5.3 Cost Analysis (RQ2)

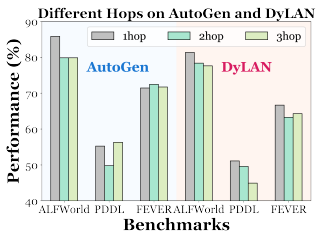
To evaluate the efficiency of **`G-Memory`** in terms of token consumption, we visualize the performance versus token cost trade-off across various settings, as shown in Figures 3 and 7. Our findings are:

Takeaway ❸: `G-Memory` achieves high-performing collective memory without excessive token consumption. As depicted in Figure 3, **`G-Memory`** consistently delivers the highest performance improvement ($10.32\% \uparrow$ over no-memory setting on PDDL+AutoGen) while maintaining a modest increase in token consumption (only 1.4×10^6). In contrast, MetaGPT-M incurred an additional 2.2×10^6 tokens for a mere 4.07% gain. This clearly demonstrates the token-efficiency of **`G-Memory`**.

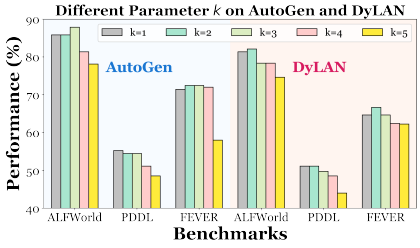
5.4 Framework Analysis (RQ3)

Sensitivity Analysis. Regarding the hop expansion, as shown in Figure 4a, 1-hop expansion consistently yields the best or near-best performance across tasks, with peak accuracies of 85.82% (ALFWorld), 55.24% (PDDL) in AutoGen. In contrast, 2-hop and 3-hop settings often degrade performance, *e.g.*, PDDL drops to 49.79% (2-hop). This suggests that excessive hop expansion may introduce irrelevant insights during memory upward traversal, impairing task-specific reasoning. Similarly, Figure 4b shows that the optimal k is among $\{1, 2\}$. Larger k values (*e.g.*, k=5) can significantly degrade the system performance, *e.g.*, $7.71\% \downarrow$ on ALFWorld+AutoGen and $2.5\% \downarrow$ on FEVER+DyLAN, indicating that retrieving more queries may introduce task-irrelevant noise. Collectively, we employ 1-hop expansion and $k \in \{1, 2\}$ throughout the experiments.

Ablation Study. Figure 4c presents an ablation of **`G-Memory`** by isolating the impact of the high-level insight module ($\mathcal{I}^{\mathcal{S}}$ in Equation (6)) and fine-grained interactions ($\{\hat{\mathcal{G}}_{\text{inter}}^{Q_i}\}_{i=1}^{|M|}$ in Equation (7)). As shown, removing either part leads to a consistent performance drop. When only fine-grained interactions are enabled, the average scores drop by $4.47\% \downarrow$ for AutoGen and $3.82\% \downarrow$ for DyLAN



(a) Sensitivity analysis on #hop.



(b) Sensitivity analysis on parameter k.

MAS	Inter.	Insi.	PDDL	FEVER
AutoGen	✔	○	54.46	63.27
	○	✔	50.00	68.77
	✔	✔	55.24	71.43
DyLAN	✔	○	48.75	61.39
	○	✔	46.69	64.31
	✔	✔	51.12	66.66

(c) Ablation study on two variants of `G-Memory`.

Figure 4: (a) Sensitivity analysis of the hop expansion in Equation (5); (b) Sensitivity analysis of the number of selected queries k in Equation (4); (c) We study two variants of `G-Memory`: merely providing high-level insights (*i.e.*, the insights $\mathcal{I}^{\mathcal{S}}$ in Equation (6)) or fine-grained interactions (*i.e.*, the core trajectories in Equation (7)). All the experiments here are done with `Qwen-2.5-14b`.

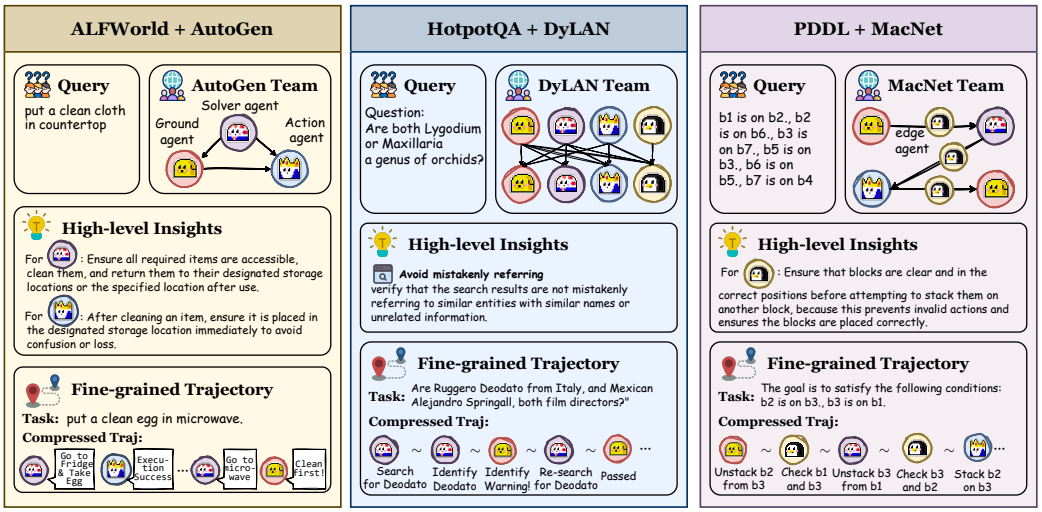


Figure 5: Case study of `G-Memory`.

compared to the full method. Conversely, enabling only insights leads to smaller drops of 3.95% and 3.39%. This indicates that while both components are contributive, interactions offer a slightly greater impact, likely due to their preserving more fine-grained, dialogue-level contextual grounding.

5.5 Case Study

Figure 5 illustrates concrete memory cues provided by `G-Memory` across diverse tasks. For example, in the ALFWorld+AutoGen setting, given the task query "put a clean cloth in countertop", `G-Memory` successfully retrieves a highly analogous historical query, "put a clean egg in microwave"—both requiring the object to be in a clean state. Alongside this, `G-Memory` surfaces a critical trajectory segment where the solver agent attempts to place the egg in the microwave before cleaning, prompting the ground agent to intervene. This collaborative trajectory offers actionable guidance for the current task. Moreover, the high-level insights retrieved by `G-Memory` prove equally valuable for task execution. In the context of HotpotQA's web search task, `G-Memory` retrieves an insight warning against "mistakenly referring", which helps prevent agents from incorrectly answering based on similarly named individuals. Overall, `G-Memory` provides effective multi-level memory support across varied domains, including embodied action, knowledge reasoning, and game environments.

6 Conclusion & Limitation

In this paper, we conduct a thorough examination of existing memory architectures designed for multi-agent systems (MAS) and identify that their overly simplified designs fundamentally hinder the systems' capacity for self-evolution. To bridge this gap, we propose `G-Memory`, a hierarchical memory framework that organizes the complex and extended interaction trajectories of MAS into a three-tier graph hierarchy: the *insight*, *query*, and *interaction* graphs. `G-Memory` provides each agent with customized and hierarchical memory cues, ranging from abstract, generalizable insights

to fine-grained, task-critical collaborative segments, and dynamically evolves its knowledge base across episodes. Extensive experiments demonstrate that `G-Memory` can be seamlessly integrated into state-of-the-art MAS frameworks, significantly enhancing their self-evolution capability, *e.g.*, up to $20.89\% \uparrow$ improvement on embodied action tasks. **Limitations:** Although `G-Memory` has been evaluated across three domains and five benchmarks, further validation on more diverse tasks (*e.g.*, medical QA) would strengthen its soundness, which we leave for future work.

Acknowledgment

This research is supported in part by NUS Start-up Grant A-0010106-00-00.

References

[1] James P Walsh and Gerardo Rivera Ungson. Organizational memory. Academy of management review, 16(1):57–91, 1991.

[2] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, et al. Palm-e: An embodied multimodal language model. 2023.

[3] Shihao Wang, Zhiding Yu, Xiaohui Jiang, Shiyi Lan, Min Shi, Nadine Chang, Jan Kautz, Ying Li, and Jose M Alvarez. Omnidrive: A holistic llm-agent framework for autonomous driving with 3d perception, reasoning and planning. arXiv preprint arXiv:2405.01533, 2024.

[4] Sipeng Zheng, Jiazheng Liu, Yicheng Feng, and Zongqing Lu. Steve-eye: Equipping llm-based embodied agents with visual perception in open worlds. arXiv preprint arXiv:2310.13255, 2023.

[5] Yuxi Wei, Zi Wang, Yifan Lu, Chenxin Xu, Changxing Liu, Hao Zhao, Siheng Chen, and Yanfeng Wang. Editable scene simulation for autonomous driving via collaborative llm-agents. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 15077–15087, 2024.

[6] Yuqi Zhu, Shuofei Qiao, Yixin Ou, Shumin Deng, Shiwei Lyu, Yue Shen, Lei Liang, Jinjie Gu, Huajun Chen, and Ningyu Zhang. Knowagent: Knowledge-augmented planning for llm-based agents. arXiv preprint arXiv:2403.03101, 2024.

[7] Lutfi Eren Erdogan, Nicholas Lee, Sehoon Kim, Suhong Moon, Hiroki Furuta, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. Plan-and-act: Improving planning of agents for long-horizon tasks. arXiv preprint arXiv:2503.09572, 2025.

[8] Xu Huang, Weiwen Liu, Xiaolong Chen, Xingmei Wang, Hao Wang, Defu Lian, Yasheng Wang, Ruiming Tang, and Enhong Chen. Understanding the planning of llm agents: A survey. arXiv preprint arXiv:2402.02716, 2024.

[9] Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. Agent q: Advanced reasoning and learning for autonomous ai agents. arXiv preprint arXiv:2408.07199, 2024.

[10] Tula Masterman, Sandi Besen, Mason Sawtell, and Alex Chao. The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey. arXiv preprint arXiv:2404.11584, 2024.

[11] Manling Li, Shiyu Zhao, Qineng Wang, Kangrui Wang, Yu Zhou, Sanjana Srivastava, Cem Gokmen, Tony Lee, Erran Li Li, Ruohan Zhang, et al. Embodied agent interface: Benchmarking llms for embodied decision making. Advances in Neural Information Processing Systems, 37:100428–100534, 2024.

[12] Yijun Yang, Tianyi Zhou, Kanxue Li, Dapeng Tao, Lusong Li, Li Shen, Xiaodong He, Jing Jiang, and Yuhui Shi. Embodied multi-modal agent trained by an llm from a parallel textworld. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 26275–26285, 2024.

[13] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation framework, August 01, 2023 2023.

[14] Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y. Wu, Y. K. Li, Fuli Luo, Yingfei Xiong, and Wenfeng Liang. Deepseek-coder: When the large language model meets programming – the rise of code intelligence, 2024.

[15] Sirui Hong, Yizhang Lin, Bang Liu, Bangbang Liu, Binhao Wu, Ceyao Zhang, Chenxing Wei, Danyang Li, Jiaqi Chen, Jiayi Zhang, et al. Data interpreter: An llm agent for data science. arXiv preprint arXiv:2402.18679, 2024.

[16] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An Open-Ended Embodied Agent with Large Language Models. arXiv e-prints, page arXiv:2305.16291, May 2023.

[17] Long Chen, Oleg Sinavski, Jan Hünermann, Alice Karnsund, Andrew James Willmott, Danny Birch, Daniel Maund, and Jamie Shotton. Driving with llms: Fusing object-level vector modality for explainable autonomous driving. In 2024 IEEE International Conference on Robotics and Automation (ICRA), pages 14093–14100. IEEE, 2024.

[18] Yuan Sun, Navid Salami Pargoo, Peter Jin, and Jorge Ortiz. Optimizing autonomous driving for safety: A human-centric approach with llm-enhanced rlhf. In Companion of the 2024 on ACM International Joint Conference on Pervasive and Ubiquitous Computing, pages 76–80, 2024.

[19] Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior, April 01, 2023 2023.

[20] Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. CoRR, abs/2305.14325, 2023.

[21] Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, and Chenglin Wu. Metagpt: Meta programming for multi-agent collaborative framework, August 01, 2023 2023.

[22] Marvin Minsky. Society of mind. Simon and Schuster, 1988.

[23] Push Singh. Examining the society of mind. Comput. Artif. Intell., 22(6):521–543, 2003.

[24] Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: communicative agents for "mind" exploration of large language model society. In NeurIPS, 2023.

[25] Zhenhailong Wang, Shaoguang Mao, Wenshan Wu, Tao Ge, Furu Wei, and Heng Ji. Unleashing cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration, July 01, 2023 2023. work in progress.

[26] Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. Large language model based multi-agents: A survey of progress and challenges. CoRR, abs/2402.01680, 2024.

[27] Pouya Pezeshkpour, Eser Kandogan, Nikita Bhutani, Sajjadur Rahman, Tom Mitchell, and Estevam Hruschka. Reasoning capacity in multi-agent systems: Limitations, challenges and human-centered solutions. CoRR, abs/2402.01108, 2024.

[28] Giorgio Piatti, Zhijing Jin, Max Kleiman-Weiner, Bernhard Schölkopf, Mrinmaya Sachan, and Rada Mihalcea. Cooperate or collapse: Emergence of sustainability behaviors in a society of llm agents. arXiv preprint arXiv:2404.16698, 2024.

[29] Rafael Pina, Varuna De Silva, and Corentin Artaud. Discovering causality for efficient cooperation in multi-agent environments. CoRR, abs/2306.11846, 2023.

[30] Guibin Zhang, Yanwei Yue, Zhixun Li, Sukwon Yun, Guancheng Wan, Kun Wang, Dawei Cheng, Jeffrey Xu Yu, and Tianlong Chen. Cut the crap: An economical communication pipeline for llm-based multi-agent systems. arXiv preprint arXiv:2410.02506, 2024.

[31] Yanwei Yue, Guibin Zhang, Boyang Liu, Guancheng Wan, Kun Wang, Dawei Cheng, and Yiyan Qi. Masrouter: Learning to route llms for multi-agent systems. arXiv preprint arXiv:2502.11133, 2025.

[32] Qinlin Zhao, Jindong Wang, Yixuan Zhang, Yiqiao Jin, Kaijie Zhu, Hao Chen, and Xing Xie. Competeai: Understanding the competition behaviors in large language model-based agents. arXiv preprint arXiv:2310.17512, 2023.

[33] Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. Encouraging divergent thinking in large language models through multi-agent debate. CoRR, abs/2305.19118, 2023.

[34] Wei Wang, Dan Zhang, Tao Feng, Boyan Wang, and Jie Tang. Battleagentbench: A benchmark for evaluating cooperation and competition capabilities of language models in multi-agent systems. arXiv preprint arXiv:2408.15971, 2024.

[35] Chuanyang Zheng, Zhengying Liu, Enze Xie, Zhenguo Li, and Yu Li. Progressive-hint prompting improves reasoning in large language models, April 01, 2023 2023. Tech Report.

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

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

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

[39] Chenxu Hu, Jie Fu, Chenzhuang Du, Simian Luo, Junbo Zhao, and Hang Zhao. Chatdb: Augmenting llms with databases as their symbolic memory. arXiv preprint arXiv:2306.03901, 2023.

[40] Junru Lu, Siyu An, Mingbao Lin, Gabriele Pergola, Yulan He, Di Yin, Xing Sun, and Yunsheng Wu. Memochat: Tuning llms to use memos for consistent long-range open-domain conversation. arXiv preprint arXiv:2308.08239, 2023.

[41] Yancheng Wang, Ziyan Jiang, Zheng Chen, Fan Yang, Yingxue Zhou, Eunah Cho, Xing Fan, Xiaojiang Huang, Yanbin Lu, and Yingzhen Yang. Recmind: Large language model powered agent for recommendation. arXiv preprint arXiv:2308.14296, 2023.

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

[43] Yuan Li, Yixuan Zhang, and Lichao Sun. Metaagents: Simulating interactions of human behaviors for llm-based task-oriented coordination via collaborative generative agents. arXiv preprint arXiv:2310.06500, 2023.

[44] Chen Gao, Xiaochong Lan, Zhihong Lu, Jinzhu Mao, Jinghua Piao, Huandong Wang, Depeng Jin, and Yong Li. S3: Social-network simulation system with large language model-empowered agents. arXiv preprint arXiv:2307.14984, 2023.

[45] Chen Qian, Xin Cong, Cheng Yang, Weize Chen, Yusheng Su, Juyuan Xu, Zhiyuan Liu, and Maosong Sun. Communicative agents for software development, July 01, 2023 2023. 25 pages, 9 figures, 2 tables.

[46] Chen Qian, Zihao Xie, Yifei Wang, Wei Liu, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, Zhiyuan Liu, and Maosong Sun. Scaling large-language-model-based multi-agent collaboration. arXiv preprint arXiv:2406.07155, 2024.

[47] Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. Gptswarm: Language agents as optimizable graphs. In Forty-first International Conference on Machine Learning, 2024.

[48] Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. arXiv preprint arXiv:2408.08435, 2024.

[49] Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. AFlow: Automating Agentic Workflow Generation, October 2024. arXiv:2410.10762.

[50] Guibin Zhang, Luyang Niu, Junfeng Fang, Kun Wang, Lei Bai, and Xiang Wang. Multi-agent architecture search via agentic supernet. arXiv preprint arXiv:2502.04180, 2025.

[51] Zhangyue Yin, Qiushi Sun, Cheng Chang, Qipeng Guo, Junqi Dai, Xuan-Jing Huang, and Xipeng Qiu. Exchange-of-thought: Enhancing large language model capabilities through cross-model communication. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 15135–15153, 2023.

[52] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji-Rong Wen. A survey on large language model based autonomous agents. Front. Comput. Sci., 18, 2024.

[53] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, Zhangyue Yin, Shihan Dou, Rongxiang Weng, Wensen Cheng, Qi Zhang, Wenjuan Qin, Yongyan Zheng, Xipeng Qiu, Xuanjing Huan, and Tao Gui. The rise and potential of large language model based agents: A survey. arxiv preprint, abs/2309.07864, 2023.

[54] Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. Large language models empowered agent-based modeling and simulation: A survey and perspectives. CoRR, abs/2312.11970, 2023.

[55] Xinyi Li, Sai Wang, Siqi Zeng, Yu Wu, and Yi Yang. A survey on llm-based multi-agent systems: workflow, infrastructure, and challenges. Vicinagearth, 1(1):9, 2024.

[56] Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. Synapse: Trajectory-as-exemplar prompting with memory for computer control. arXiv preprint arXiv:2306.07863, 2023.

[57] Xizhou Zhu, Yuntao Chen, Hao Tian, Chenxin Tao, Weijie Su, Chenyu Yang, Gao Huang, Bin Li, Lewei Lu, Xiaogang Wang, et al. Ghost in the minecraft: Generally capable agents for open-world environments via large language models with text-based knowledge and memory. arXiv preprint arXiv:2305.17144, 2023.

[58] Xiangru Tang, Tianyu Hu, Muyang Ye, Yanjun Shao, Xunjian Yin, Siru Ouyang, Wangchunshu Zhou, Pan Lu, Zhuosheng Zhang, Yilun Zhao, et al. Chemagent: Self-updating library in large language models improves chemical reasoning. arXiv preprint arXiv:2501.06590, 2025.

[59] Noah Shinn, Beck Labash, and Ashwin Gopinath. Reflexion: an autonomous agent with dynamic memory and self-reflection. arXiv preprint, abs/2303.11366, 2023.

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

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

[62] Rana Salama, Jason Cai, Michelle Yuan, Anna Currey, Monica Sunkara, Yi Zhang, and Yassine Benajiba. Meminsight: Autonomous memory augmentation for llm agents. arXiv preprint arXiv:2503.21760, 2025.

[63] Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. Mixture-of-agents enhances large language model capabilities. arXiv preprint arXiv:2406.04692, 2024.

[64] Wangchunshu Zhou, Yixin Ou, Shengwei Ding, Long Li, Jialong Wu, Tiannan Wang, Jiamin Chen, Shuai Wang, Xiaohua Xu, Ningyu Zhang, et al. Symbolic learning enables self-evolving agents. arXiv preprint arXiv:2406.18532, 2024.

[65] Xuechen Liang, Meiling Tao, Yinghui Xia, Tianyu Shi, Jun Wang, and JingSong Yang. Self-evolving agents with reflective and memory-augmented abilities. arXiv preprint arXiv:2409.00872, 2024.

[66] Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chen Qian, Chi-Min Chan, Yujia Qin, Yaxi Lu, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors in agents, 2023.

[67] Yue Hu, Yuzhu Cai, Yaxin Du, Xinyu Zhu, Xiangrui Liu, Zijie Yu, Yuchen Hou, Shuo Tang, and Siheng Chen. Self-evolving multi-agent collaboration networks for software development. arXiv preprint arXiv:2410.16946, 2024.

[68] Guibin Zhang, Yanwei Yue, Xiangguo Sun, Guancheng Wan, Miao Yu, Junfeng Fang, Kun Wang, Tianlong Chen, and Dawei Cheng. G-designer: Architecting multi-agent communication topologies via graph neural networks. arXiv preprint arXiv:2410.11782, 2024.

[69] Siyu Yuan, Kaitao Song, Jiangjie Chen, Xu Tan, Dongsheng Li, and Deqing Yang. Evoagent: Towards automatic multi-agent generation via evolutionary algorithms. arXiv preprint arXiv:2406.14228, 2024.

[70] Guibin Zhang, Kaijie Chen, Guancheng Wan, Heng Chang, Hong Cheng, Kun Wang, Shuyue Hu, and Lei Bai. Evoflow: Evolving diverse agentic workflows on the fly. arXiv preprint arXiv:2502.07373, 2025.

[71] Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. Dynamic llm-agent network: An llm-agent collaboration framework with agent team optimization. CoRR, abs/2310.02170, 2023.

[72] Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia. Microsoft academic graph: When experts are not enough. Quantitative Science Studies, 1(1):396–413, 2020.

[73] Wanjia Zhao, Mert Yuksekgonul, Shirley Wu, and James Zou. Sirius: Self-improving multi-agent systems via bootstrapped reasoning. arXiv preprint arXiv:2502.04780, 2025.

[74] Heng Zhou, Hejia Geng, Xiangyuan Xue, Zhenfei Yin, and Lei Bai. Reso: A reward-driven self-organizing llm-based multi-agent system for reasoning tasks. arXiv preprint arXiv:2503.02390, 2025.

[75] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. arXiv preprint arXiv:1809.09600, 2018.

[76] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. Fever: a large-scale dataset for fact extraction and verification. arXiv preprint arXiv:1803.05355, 2018.

[77] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. arXiv preprint arXiv:2010.03768, 2020.

[78] Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. Scienceworld: Is your agent smarter than a 5th grader? arXiv preprint arXiv:2203.07540, 2022.

[79] Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. Agentboard: An analytical evaluation board of multi-turn llm agents. arXiv preprint arXiv:2401.13178, 2024.

[80] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. Advances in Neural Information Processing Systems, 33:5776–5788, 2020.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our main claim that multi-agent systems require more dedicated memory architectures is both methodologically and empirically supported.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please refer to Section 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: This paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have thoroughly provided the relevant information (see Appendix A and section 5.1).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We have provided the relevant codes with clear instructions in an anonymous link (See Abstract). All the datasets we used are publicly available and clearly cited.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We have described the dataset setups and evaluation metrics in Appendix A. All the hyperparameters are described in Section 5.1. There are no explicit dataset splits.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The experimental results are the average of three runs to avoid the random bias.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Please refer to Section 5.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: This paper conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Please refer to Section 6.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The information has been provided in Appendix A.1.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

Impact Statement

G-Memory introduces a structured, hierarchical memory architecture for multi-agent systems (MAS), enabling large language model (LLM)-based agents to store, recall, and reason over past experiences with enhanced task generalization and cooperation efficiency. The broader impacts of this work include advancing the development of scalable and adaptive collective intelligence, with potential applications in long-term robotic planning, real-world decision-making systems, and collaborative AI assistants. However, if the underlying language model is compromised or adversarially manipulated, the memory mechanisms could amplify incorrect reasoning. We urge responsible deployment of this architecture with appropriate safeguards, including continual validation, adversarial robustness checks, and alignment with human values.

A Experimental Details

A.1 Dataset Descriptions

In this section, we describe the datasets used in our experiments:

- **ALFWorld** [77] (available at https://alfworld.github.io/, MIT license) is a text-based embodied environment featuring household tasks, where agents navigate and interact with objects via natural language commands.
- **ScienceWorld** [78] (available at https://github.com/allenai/ScienceWorld, Apache-2.0 license) is another text-based embodied environment designed for interactive science tasks. Agents must navigate rooms and conduct experiments, testing their ability to perform procedural reasoning and scientific exploration.
- **PDDL** is a game dataset from AgentBoard [79] (available at https://github.com/hkust-nlp/AgentBoard, Custom properties), comprising a variety of strategic games where agents use PDDL expressions to complete complex tasks.
- **HotpotQA** [75] (available at https://hotpotqa.github.io/, CC BY-SA 4.0 License) is a multi-hop question answering dataset with strong supervision on supporting facts. It evaluates the agent's ability to retrieve and synthesize information, especially through web search tools, for explainable reasoning.
- **FEVER** [76] (available at https://fever.ai/dataset/fever.html, Creative Commons Attribution-ShareAlike License) is a knowledge-intensive dataset focused on fact verification. Agents must validate claims using web search APIs, making it a benchmark for evidence-based reasoning.

Evaluation Metrics. We use *exact match* accuracy for FEVER and HotpotQA. For ScienceWorld and PDDL, we report the *progress rate*, and for ALFWorld, we use the *success rate* as the evaluation metric.

A.2 Baseline Setup

In this section, we provide detailed descriptions of each baseline used in our comparison:

- **Voyager**: The Voyager memory is derived from the Voyager agent [16], where an embodied agent continuously interacts with the Minecraft environment and creates new artifacts. Memory serves as the core driver of the agent's evolution. As Voyager's memory design is tailored for a single-agent setting, we adapt it to the multi-agent scenario by implementing agent-specific history retrieval based on each agent's visible dialogue context. Other single-agent memory designs are adapted in a similar manner.
- **MemoryBank**: MemoryBank [36] mimics anthropomorphic memory behaviors by selectively preserving and forgetting information. It incorporates a memory updating mechanism inspired by the Ebbinghaus Forgetting Curve, allowing the agent to reinforce or discard memory based on temporal decay and the relative importance of stored information.
- **Generative**: This memory baseline is based on [19], which includes both raw observational memory and high-level reflective memory. The latter captures abstract thoughts generated by the agent through reflection, providing a more structured and conceptualized representation of experience.
- **MetaGPT-M**: The memory design originates from MetaGPT [21], focusing solely on *inside-trial* memory—information stored internally during the resolution of a single task by multiple agents.
- **ChatDev-M**: This memory design is adapted from ChatDev [45], which incorporates both *inside-trial* and *cross-trial* memory. The inside-trial memory is passed from the central or initiating agent at the beginning of each round to provide guidance based on prior interactions. The cross-trial memory is relatively simple, storing past solutions to previous queries for future retrieval. However, in our task, it does not effectively manage the information-rich inter-agent collaboration.
- **MacNet-M**: This memory design is adopted from MacNet [46], where the *inside-trial* memory consists solely of the final answers generated in the previous round. All non-artifact dialogue contexts, *i.e.*, the interaction trajectories among agents, are entirely discarded.

A.3 Multi-agent System Setup

In this section, we detail the setups of our three adopted MAS frameworks, AutoGen, DyLAN and MacNet:

A.3.1 AutoGen

AutoGen [13] is a popular multi-agent orchestration framework, to coordinate interactions among specialized agents for problem-solving tasks. Specifically, we utilize their A3 : Decision Making structure, which is composed of: (1) a **Solver Agent**, responsible for generating solutions, initialized with the system prompt "You are a smart agent designed to solve problems."; (2) a **Ground Truth Agent**, which critically evaluates the solver's output and identifies potential errors based on a reference standard; and (3) an **Executor Agent**, tasked with translating validated solutions into executable commands. This modular design enables transparent, verifiable, and actionable multi-agent collaboration.

A.3.2 DyLAN

DyLAN [71] is a debate-style framework similar to LLM-Debate, but incorporates a more efficient agent-wise early stopping mechanism during multi-turn interactions. DyLAN utilizes an agent selection algorithm based on an unsupervised metric, namely the *Agent Importance Score*, which identifies the most contributive agents through a preliminary trial tailored to the specific task. In our implementation of DyLAN, three agents engage in the debate, while an additional ranker agent evaluates their relative importance.

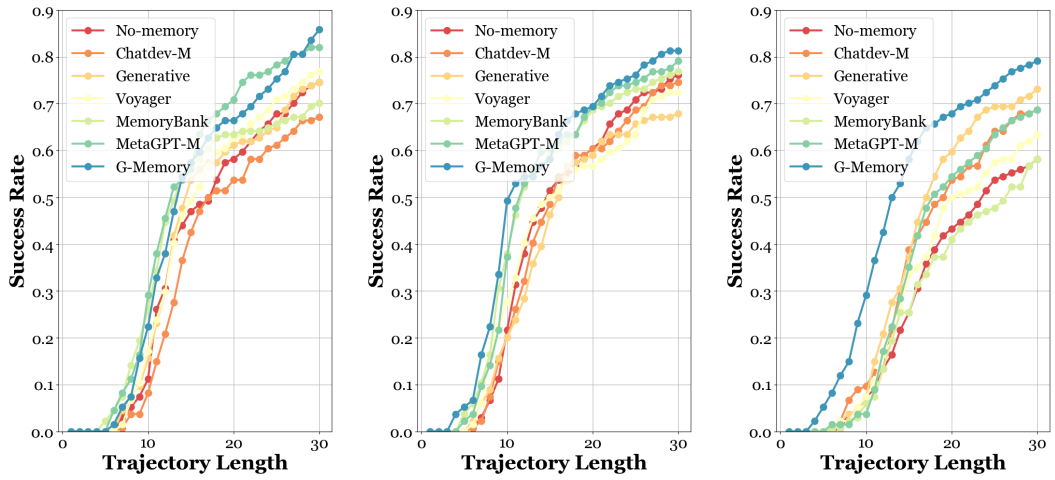
A.4 MacNet

MacNet [46] is a representative work that explores decentralized and scalable multi-agent systems. Its key feature lies in the absence of a central agent; instead, it introduces *edge agents*, which are invoked between agent interactions to provide actionable instructions to the next agent based on the previous agent's outputs. In our implementation, we adopt the random graph topology from MacNet, shown to be robust across diverse scenarios, and employ five agents in addition to the edge agents.

B Additional Experiment Results

B.1 RQ1 Results

Tables 2 and 3 present additional experimental results using `Qwen-2.5-7b` and `Qwen-2.5-14b` as the LLM backbones. Appendix B.1 illustrates the success rate curves on ALFWorld as the number of trials increases, comparing different MAS frameworks combined with various memory architectures. As shown in Figures 6b and 6c, **`G-Memory`** consistently enables MAS frameworks to achieve success with fewer trials and leads to higher final performance ceilings.



(a) The performance trajectory of AutoGen on ALFWorld.

(b) The performance trajectory of DyLAN on ALFWorld.

(c) The performance trajectory of MacNet on ALFWorld.

B.2 RQ2 Results

Figure 7 provides additional comparisons of token cost across various benchmarks and MAS frameworks when combined with different memory architectures. Overall, **`G-Memory`** incurs only a marginal or no increase in token cost compared to classical baselines such as Generative and MetaGPT-M, while consistently delivering the most significant performance improvements.

B.3 Latency Results

As shown in Table 4, **`G-Memory`** incurs only moderate inference overhead even when delivering the most significant performance gains. For instance, on AutoGen+ALFWorld, it increases latency by merely 9% compared to the no-memory baseline; (2) in some cases, **`G-Memory`** even improves efficiency, *e.g.*, on DyLAN+ALFWorld, it leads to a 20% speed-up, as helpful memory cues enable the multi-agent system to reach correct actions more quickly and terminate the interaction earlier.

Table 2: Performance comparison with single/multi-agent memory architectures on five benchmarks. The underlying LLM backbone is `Qwen-2.5-7b`. We highlight the best and second best results.

MAS	Memory	ALFWorld	SciWorld	PDDL	HotpotQA	FEVER	Avg.
Vanilla LLM	No-memory	37.31$_{\uparrow 0.00}$	23.49$_{\uparrow 0.00}$	10.86$_{\uparrow 0.00}$	20.26$_{\uparrow 0.00}$	48.17$_{\uparrow 0.00}$	28.02$_{\uparrow 0.00}$
	Voyager	38.19$_{\uparrow 0.88}$	24.11$_{\uparrow 0.62}$	12.14$_{\uparrow 1.28}$	19.12$_{\downarrow 1.14}$	49.68$_{\uparrow 1.51}$	28.65$_{\uparrow 0.63}$
	MemoryBank	40.30$_{\uparrow 2.99}$	21.64$_{\downarrow 1.85}$	14.36$_{\uparrow 3.50}$	18.79$_{\downarrow 1.47}$	47.66$_{\downarrow 0.51}$	28.55$_{\uparrow 0.53}$
	Generative	39.16$_{\uparrow 1.85}$	26.10$_{\uparrow 2.61}$	11.37$_{\uparrow 0.51}$	23.48$_{\uparrow 3.22}$	52.50$_{\uparrow 4.33}$	30.52$_{\uparrow 2.50}$
AutoGen COLM 2024	No-memory	52.99$_{\uparrow 0.00}$	30.27$_{\uparrow 0.00}$	16.17$_{\uparrow 0.00}$	33.33$_{\uparrow 0.00}$	58.74$_{\uparrow 0.00}$	38.30$_{\uparrow 0.00}$
	Voyager	55.22$_{\uparrow 2.23}$	26.70$_{\downarrow 3.57}$	12.00$_{\downarrow 4.17}$	34.29$_{\uparrow 0.96}$	52.44$_{\downarrow 6.30}$	36.13$_{\downarrow 2.17}$
	MemoryBank	53.37$_{\uparrow 0.38}$	27.33$_{\downarrow 2.94}$	14.83$_{\downarrow 1.34}$	32.67$_{\downarrow 0.66}$	59.45$_{\uparrow 0.71}$	37.53$_{\downarrow 0.77}$
	Generative	62.69$_{\uparrow 9.70}$	31.45$_{\uparrow 1.18}$	17.88$_{\uparrow 1.71}$	34.17$_{\uparrow 0.84}$	61.25$_{\uparrow 2.51}$	41.49$_{\uparrow 3.19}$
	MetaGPT-M	55.52$_{\uparrow 2.53}$	32.44$_{\uparrow 2.17}$	17.04$_{\uparrow 0.87}$	35.36$_{\uparrow 2.03}$	63.33$_{\uparrow 4.59}$	40.74$_{\uparrow 2.44}$
	ChatDev-M	46.27$_{\downarrow 6.72}$	28.67$_{\downarrow 1.60}$	13.42$_{\downarrow 2.75}$	31.11$_{\downarrow 2.22}$	61.32$_{\uparrow 2.58}$	36.16$_{\downarrow 2.14}$
	MacNet-M	53.18$_{\uparrow 0.19}$	31.10$_{\uparrow 0.83}$	16.89$_{\uparrow 0.72}$	34.29$_{\uparrow 0.96}$	58.43$_{\downarrow 0.31}$	38.78$_{\uparrow 0.48}$
	`G-Memory` (Ours)	67.91$_{\uparrow 14.92}$	34.89$_{\uparrow 4.62}$	21.01$_{\uparrow 4.84}$	37.34$_{\uparrow 4.01}$	64.34$_{\uparrow 5.60}$	45.10$_{\uparrow 6.80}$
DyLAN COLM 2024	No-memory	41.34$_{\uparrow 0.00}$	29.84$_{\uparrow 0.00}$	13.56$_{\uparrow 0.00}$	24.29$_{\uparrow 0.00}$	56.23$_{\uparrow 0.00}$	33.05$_{\uparrow 0.00}$
	Voyager	51.49$_{\uparrow 10.15}$	26.66$_{\downarrow 3.18}$	10.62$_{\downarrow 2.94}$	26.23$_{\uparrow 1.94}$	55.39$_{\downarrow 0.84}$	34.08$_{\uparrow 1.03}$
	MemoryBank	46.46$_{\uparrow 5.12}$	26.99$_{\downarrow 2.85}$	14.10$_{\uparrow 0.54}$	22.44$_{\downarrow 1.85}$	59.21$_{\uparrow 2.98}$	33.84$_{\uparrow 0.79}$
	Generative	48.52$_{\uparrow 7.18}$	31.55$_{\uparrow 1.71}$	16.31$_{\uparrow 2.75}$	26.54$_{\uparrow 2.25}$	50.19$_{\downarrow 6.04}$	34.62$_{\uparrow 1.57}$
	MetaGPT-M	42.54$_{\uparrow 1.20}$	30.93$_{\uparrow 1.09}$	14.47$_{\uparrow 0.91}$	19.33$_{\downarrow 4.96}$	57.22$_{\uparrow 0.99}$	32.90$_{\downarrow 0.15}$
	ChatDev-M	39.85$_{\downarrow 1.49}$	28.25$_{\downarrow 1.59}$	7.14$_{\downarrow 6.42}$	17.32$_{\downarrow 6.97}$	50.67$_{\downarrow 5.56}$	28.65$_{\downarrow 4.41}$
	MacNet-M	42.48$_{\uparrow 1.14}$	28.22$_{\downarrow 1.62}$	14.23$_{\uparrow 0.67}$	25.12$_{\uparrow 0.83}$	55.34$_{\downarrow 0.89}$	33.08$_{\uparrow 0.03}$
	`G-Memory` (Ours)	52.99$_{\uparrow 11.65}$	33.81$_{\uparrow 3.97}$	20.71$_{\uparrow 7.15}$	29.33$_{\uparrow 5.04}$	63.67$_{\uparrow 7.44}$	40.10$_{\uparrow 7.05}$
MacNet ICLR 2025	No-memory	44.03$_{\uparrow 0.00}$	28.76$_{\uparrow 0.00}$	13.36$_{\uparrow 0.00}$	22.24$_{\uparrow 0.00}$	55.12$_{\uparrow 0.00}$	32.70$_{\uparrow 0.00}$
	Voyager	47.01$_{\uparrow 2.98}$	28.88$_{\uparrow 0.12}$	11.36$_{\downarrow 2.00}$	25.67$_{\uparrow 3.43}$	58.78$_{\uparrow 3.66}$	34.34$_{\uparrow 1.64}$
	MemoryBank	52.24$_{\uparrow 8.21}$	27.86$_{\downarrow 0.90}$	13.33$_{\downarrow 0.03}$	23.97$_{\uparrow 1.73}$	54.18$_{\downarrow 0.94}$	34.32$_{\uparrow 1.61}$
	Generative	48.51$_{\uparrow 4.48}$	31.05$_{\uparrow 2.29}$	14.04$_{\uparrow 0.68}$	24.49$_{\uparrow 2.25}$	56.08$_{\uparrow 0.96}$	34.83$_{\uparrow 2.13}$
	MetaGPT-M	52.99$_{\uparrow 8.96}$	29.87$_{\uparrow 1.11}$	16.58$_{\uparrow 3.22}$	25.51$_{\uparrow 3.27}$	53.88$_{\downarrow 1.24}$	35.77$_{\uparrow 3.06}$
	ChatDev-M	44.78$_{\uparrow 0.75}$	26.44$_{\downarrow 2.32}$	10.19$_{\downarrow 3.17}$	16.32$_{\downarrow 5.92}$	56.02$_{\uparrow 0.90}$	30.75$_{\downarrow 1.95}$
	MacNet-M	43.55$_{\downarrow 0.48}$	30.11$_{\uparrow 1.35}$	12.91$_{\downarrow 0.45}$	21.77$_{\downarrow 0.47}$	50.71$_{\downarrow 4.41}$	31.81$_{\downarrow 0.89}$
	`G-Memory` (Ours)	54.48$_{\uparrow 10.45}$	32.23$_{\uparrow 3.47}$	17.48$_{\uparrow 4.12}$	27.53$_{\uparrow 5.29}$	59.14$_{\uparrow 4.02}$	38.17$_{\uparrow 5.47}$

B.4 Case Study

B.4.1 Case Study on Insight Graphs

Figure 8 visualizes the high-level insights summarized by `G-Memory` on the ALFWorld benchmark across different MAS frameworks and LLM backbones. Given that ALFWorld naturally consists of diverse task categories, we further examine how insight nodes corresponding to different task types are interconnected. Overall, we observe dense intra-category connections among insights derived from similar tasks, while also noting the emergence of meaningful inter-category links, reflecting transferable patterns across task domains.

B.4.2 Case Study on Query Graphs

Figures 9 to 11 visualize the query graphs constructed by `G-Memory` on the ALFWorld, PDDL, and SciWorld benchmarks. Recall that a directed edge between two query nodes indicates that the historical trajectory of one query offers useful guidance for the execution of another. We observe emergent clustering patterns, where groups of semantically similar queries form densely connected subgraphs, while sparser inter-cluster edges capture cross-task inspirations. These patterns demonstrate `G-Memory`'s ability to effectively organize and relate collaborative experiences through structured memory reasoning.

Table 3: Performance comparison with single/multi-agent memory architectures on five benchmarks. The underlying LLM backbone is `Qwen-2.5-14b`. We highlight the best and second best results.

MAS	Memory	ALFWorld	SciWorld	PDDL	HotpotQA	FEVER	Avg.
AutoGen COLM 2024	No-memory	74.63 ↑0.00	46.84 ↑0.00	44.92 ↑0.00	24.49 ↑0.00	63.27 ↑0.00	50.83 ↑0.00
	Voyager	76.87 ↑2.24	59.00 ↑12.16	50.21 ↑5.29	31.33 ↑6.84	61.22 ↓2.05	55.73 ↑4.90
	MemoryBank	70.15 ↓4.48	54.18 ↑7.34	39.54 ↓5.38	32.65 ↑8.16	64.29 ↑1.02	52.16 ↑1.33
	Generative	74.63 ↑0.00	57.37 ↑10.53	54.46 ↑9.54	33.21 ↑8.72	63.27 ↑0.00	56.59 ↑5.76
	MetaGPT-M	82.09 ↑7.46	58.86 ↑12.02	48.99 ↑4.07	31.63 ↑7.14	62.27 ↓1.00	56.77 ↑5.94
	ChatDev-M	67.16 ↓7.47	40.69 ↓6.15	43.11 ↓1.81	31.77 ↑7.28	61.28 ↓1.99	48.80 ↓2.03
	MacNet-M	73.65 ↓0.98	42.14 ↓4.70	45.94 ↑1.02	26.72 ↑2.23	64.69 ↑1.42	50.63 ↓0.20
	`G-Memory` (Ours)	85.82 ↑11.19	60.62 ↑13.78	55.24 ↑10.32	34.61 ↑10.12	71.43 ↑8.16	61.54 ↑10.71
DyLAN COLM 2024	No-memory	76.12 ↑0.00	53.24 ↑0.00	41.83 ↑0.00	30.61 ↑0.00	63.34 ↑0.00	53.03 ↑0.00
	Voyager	72.39 ↓3.73	58.93 ↑5.69	48.54 ↑6.71	30.71 ↑0.10	65.31 ↑1.97	55.18 ↑2.15
	MemoryBank	76.87 ↑0.75	57.92 ↑4.68	39.65 ↓2.18	29.59 ↓1.02	63.25 ↓0.09	53.46 ↑0.43
	Generative	77.91 ↑1.79	61.52 ↑8.28	46.69 ↑4.86	31.33 ↑0.72	61.39 ↓1.95	55.77 ↑2.74
	MetaGPT-M	79.10 ↑2.98	61.29 ↑8.05	49.75 ↑7.92	28.61 ↓2.00	64.11 ↑0.77	56.57 ↑3.54
	ChatDev-M	74.63 ↓1.49	54.03 ↑0.79	44.44 ↑2.61	30.67 ↑0.06	62.25 ↓1.09	53.20 ↑0.18
	MacNet-M	72.77 ↓3.35	52.22 ↓1.02	42.98 ↑1.15	29.22 ↓1.39	62.69 ↓0.65	51.98 ↓1.05
	`G-Memory` (Ours)	81.34 ↑5.22	64.68 ↑11.44	51.12 ↑9.29	34.63 ↑4.02	66.66 ↑3.32	59.69 ↑6.66
MacNet ICLR 2025	No-memory	58.21 ↑0.00	52.21 ↑0.00	41.74 ↑0.00	28.60 ↑0.00	64.65 ↑0.00	49.08 ↑0.00
	Voyager	63.43 ↑5.22	60.24 ↑8.03	43.95 ↑2.21	29.67 ↑1.07	62.24 ↓2.41	51.91 ↑2.82
	MemoryBank	62.21 ↑4.00	55.52 ↑3.31	38.26 ↓3.48	26.53 ↓2.07	65.22 ↑0.57	49.55 ↑0.47
	Generative	73.13 ↑14.92	60.83 ↑8.62	44.00 ↑2.26	30.53 ↑1.93	65.31 ↑0.66	54.76 ↑5.68
	MetaGPT-M	70.43 ↑12.22	59.70 ↑7.49	42.34 ↑0.60	26.26 ↓2.34	66.33 ↑1.68	53.01 ↑3.93
	ChatDev-M	68.66 ↑10.45	45.98 ↓6.23	42.19 ↑0.45	29.49 ↑0.89	59.18 ↓5.47	49.10 ↑0.02
	MacNet-M	60.45 ↑2.24	51.14 ↓1.07	39.22 ↓2.52	28.77 ↑0.17	62.42 ↓2.23	48.40 ↓0.68
	`G-Memory` (Ours)	79.10 ↑20.89	61.74 ↑9.53	45.76 ↑4.02	32.33 ↑3.73	70.33 ↑5.68	57.85 ↑8.77

Table 4: Performance (%) and latency (s) comparison of different memory mechanisms on AutoGen and DyLAN frameworks, along with ALFWorld and SciWorld benchmarks.

Method	AutoGen				DyLAN			
	ALFWorld		SciWorld		ALFWorld		SciWorld	
	Perf.	Lat.	Perf.	Lat.	Perf.	Lat.	Perf.	Lat.
No-memory	77.61	19204	54.59	16953	56.72	33520	55.38	32408
Voyager	85.07	21754	62.36	16650	66.42	29628	62.83	31633
MemoryBank	74.96	15492	53.11	10104	55.22	31813	54.74	32925
Generative	86.36	20682	61.19	16674	67.91	31010	64.16	34038
MetaGPT-M	81.34	16021	61.91	15853	69.40	22936	62.37	32049
ChatDev-M	79.85	22347	50.96	12904	46.27	29739	53.35	33111
MacNet-M	76.55	21089	55.44	16882	53.44	24991	54.32	34815
G-Memory (Ours)	**88.81**	21113	**67.40**	17326	**70.90**	26726	**65.64**	33447

C Prompt Set

Query Relevance Filtration

```
task_relevency_system_prompt = """You are an agent designed to score the relevance
      between two pieces of text."""
task_relevency_user_prompt = """You will be given a successful case where you
     successfully complete the task. Then you will be given an ongoing task. Do
     not summarize these two cases, but rather evaluate how relevant and helpful
     the successful case is for the ongoing task, on a scale of 1-10.
Success Case:
{trajectory}
Ongoing task:
{query_scenario}
```

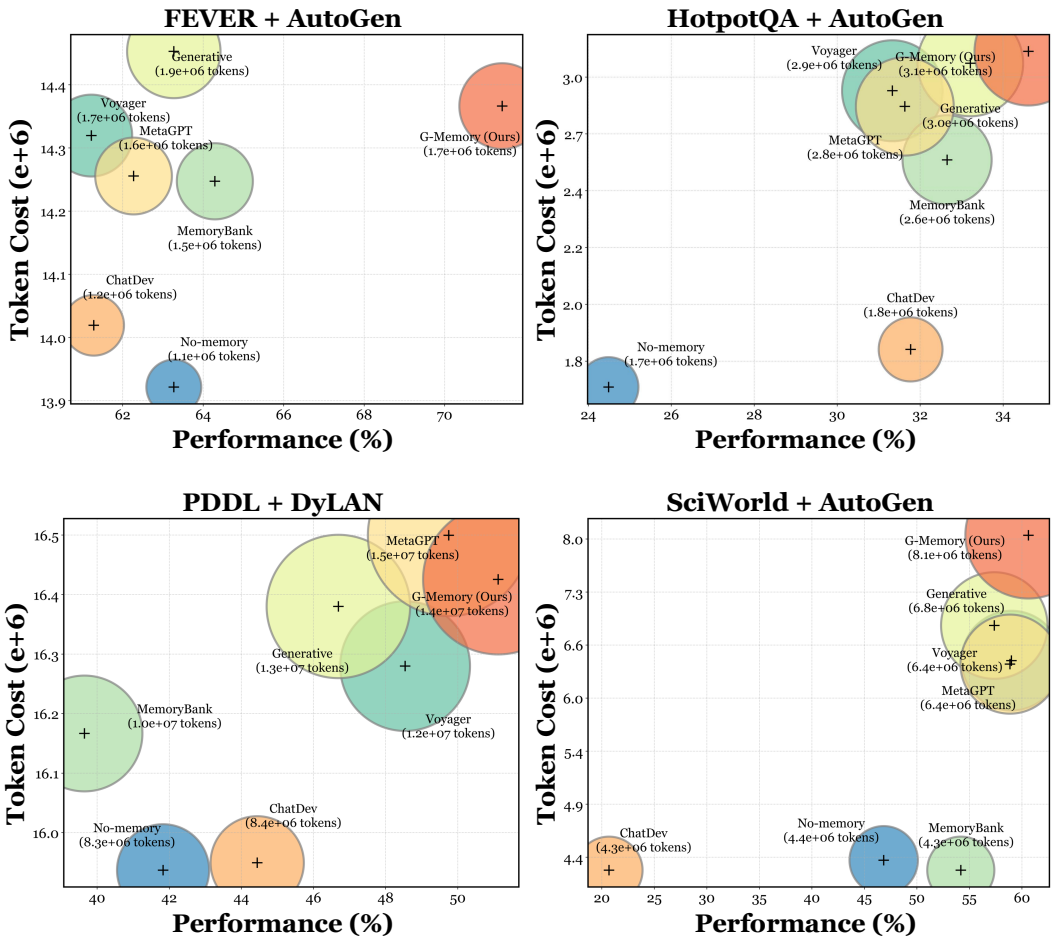
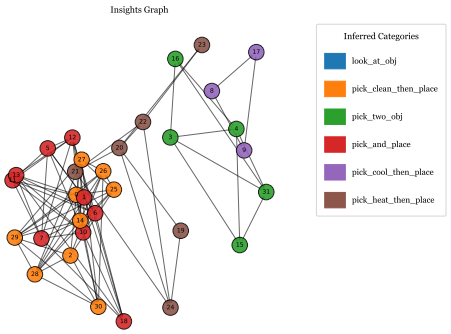


Figure 7: Cost analysis of `G-Memory`. We showcase the performance versus the overall system token cost when combined with different memory architectures.

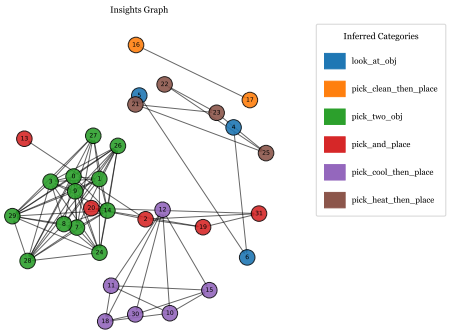
```
Score: """
```

Graph Sparsifier

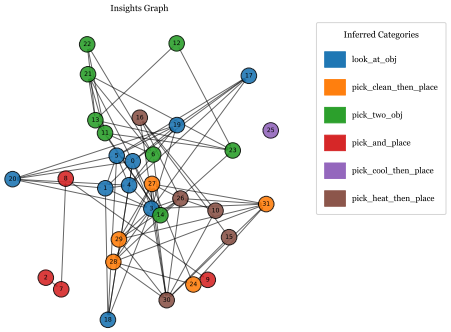
```
extract_true_traj_system_prompt = """You are an agent skilled at extracting key
    points.
Given a task and a successful execution trajectory, your job is to identify the
    critical steps needed to complete the task while filtering out less important
     steps."""

extract_true_traj_user_prompt = """
Note:
- Strictly follow the original trajectory; absolutely no steps that are not in the
     trajectory should be added.
- Even in a successful trajectory, there may be some incorrect steps. Pay
    attention to actions that correspond to "Nothing happens" observations, as
    these actions are likely incorrect. Filter out these actions for me.
- You need to ensure that each step is at the finest granularity.
- You should strictly follow the output format in the example.

## Here is the task:
### Task
{task}
```



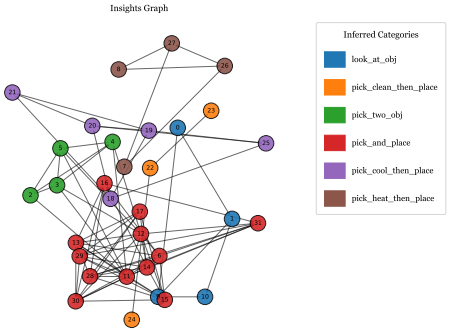
(a) Insight graph on `gpt-4o-mini` +MacNet+ALFWorld.



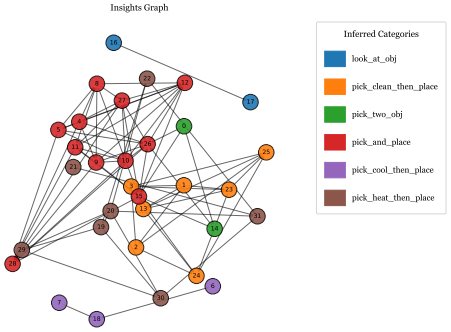
(b) Insight graph on `gpt-4o-mini` +DyLAN+ALFWorld.



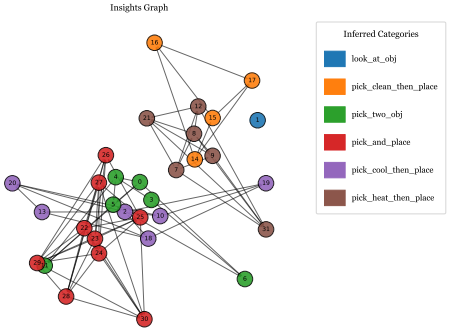
(c) Insight graph on `Qwen-7b` +MacNet+ALFWorld.



(d) Insight graph on `Qwen-7b` +DyLAN+ALFWorld.



(e) Insight graph on `Qwen-14b` +AutoGen+ALFWorld.



(f) Insight graph on `Qwen-14b` +DyLAN+ALFWorld.

Figure 8: Visualizations of insight graphs across different LLM backbones, MAS, and benchmarks.

```
### Trajectory
{trajectory}

### Output
"""
```

The prompt below is partially adapted from [42]. We would like to express our sincere gratitude for their valuable implementation.

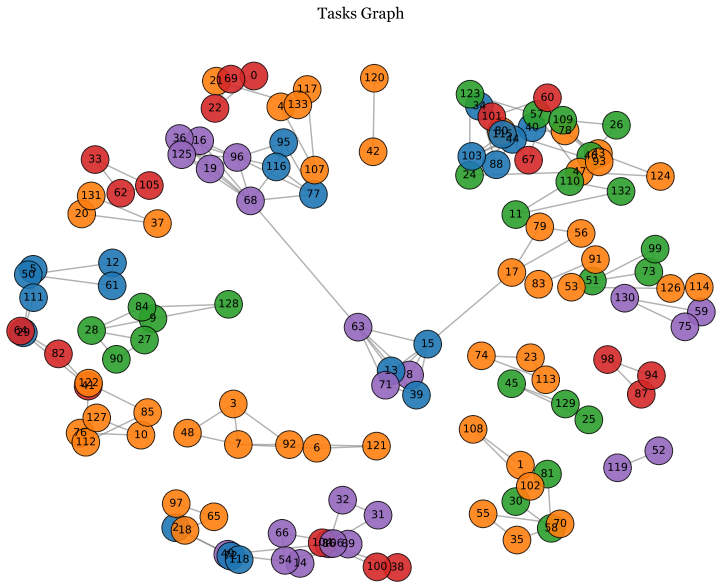


Figure 9: Query graph optimized from ALFWorld dataset.

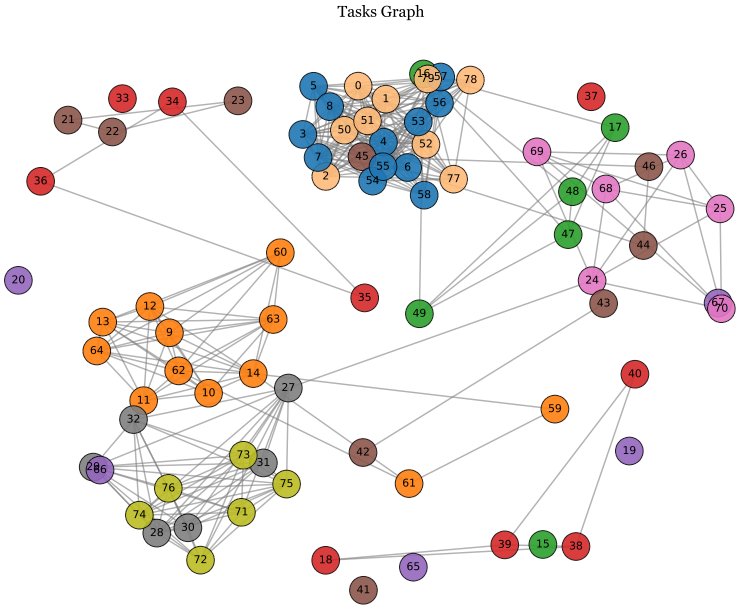


Figure 10: Query graph optimized from SciWorld dataset.

Inisght Summarization Function

```
learn_lessons_system_prompt_compare = """
You are an analysis-driven agent focused on learning from experience. You will be
     provided with:
- A failed trajectory and its outcome,
- A successful trajectory completing a similar task.

Your task is to analyze both trajectories and generate clear, actionable insights.
      Your insights should highlight what the failed trajectory missed and how the
      successful one addressed or avoided these pitfalls.

## Requirements:
- All insights must be derived directly from contrasting the two trajectories.
```

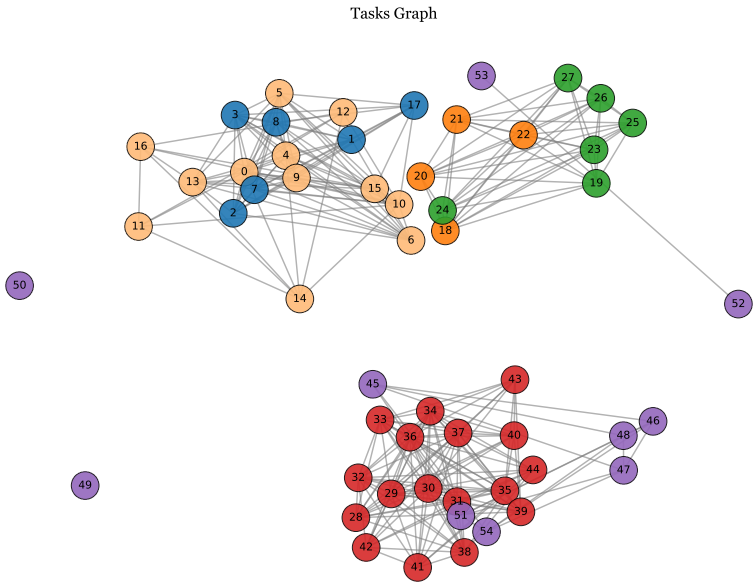


Figure 11: Query graph optimized from PDDL dataset.

```
- Do not speculate or introduce steps not supported by the successful example.
- Focus on **concrete behavioral or strategic differences** between the two cases.
- Keep each insight concise and impactful.

Output Format:
- Start immediately with a numbered list.
- No introduction or explanation.
- Use this exact format:
1. Insight 1
2. Insight 2
3. Insight 3
...
"""

learn_lessons_user_prompt_compare = """
## Successful trajectory
{true_traj}

## Failed trajectory
### trajectory
{false_traj}

Your output:
"""

learn_lessons_system_prompt_all_succ = """
You are an analysis-driven agent focused on learning from success. You will be
     provided with a set of successful trajectories that completed a similar task.

Your goal is to analyze these successful examples and extract clear, actionable
     insights that capture what contributed to their success. These insights will
     serve as guidance for future agents working on similar tasks.

## Requirements:
- All insights must be grounded in patterns or strategies observed across the
     successful trajectories.
- Do not speculate or introduce steps not reflected in the provided examples.
- Focus on common behaviors, strategies, or decisions that consistently led to
     positive outcomes.
- Keep each insight concise, specific, and impactful.

Output Format:
- Start immediately with a numbered list.
- No introduction or explanation.
- Use this exact format:
1. Insight 1
```

```
2. Insight 2
3. Insight 3
...
"""


learn_lessons_user_prompt_all_succ = """
## Successful trajectorys
{true_trajs}

Your output:
"""

# merge rules prompt
merge_rules_system_prompt = """You are an agent skilled at summarizing and
     distilling insights. You are given a list of insights that were previously
     extracted from similar tasks. These insights may contain redundancy or
     overlap.

Your job is to **merge and consolidate similar insights**, and output a refined
     version that is **clear, actionable, and concise**.

NOTE:
- All merged insights **must be based strictly on the given inputs**. You are **
     not allowed to make up** or infer any new information.
- The output should be easy to read and follow.

Output Format:
- Start your response directly with the numbered list, no preamble or explanations
     .
- Each insight should be a short sentence.
- Use the following format exactly:
1. Insight 1
2. Insight 2
3. Insight 3
...
"""

merge_rules_user_prompt = """
## Here are the current insights that need to be merged:
{current_rules}

## Please consolidate and rewrite them into **no more than {limited_number}
     refined insights**.

As the summarizing agent, remove redundancies, combine similar ideas, and ensure
     clarity.

Your output:
"""
```

Customizing Memory for Agents

```
project_insights_system_prompt: str = """
You are a thoughtful and context-aware agent. You will be provided with a
     successfully executed trajectory, a specific agent **role**, and a set of **
     general insights** applicable across all roles.
Your task is to **adapt these general insights** into **personalized insights**
     that are specifically tailored to the given role and its trajectory. These
     personalized insights should help the agent improve future performance by
     aligning with their unique background, responsibilities, and perspective.
Make sure your output reflects an understanding of the role's context and promotes
      actionable, role-relevant advice.

NOTE - Your output must strictly follow the format below:
1. Insight 1
2. Insight 2
3. Insight 3
...
"""

project_insights_user_prompt: str = """
### Trajectory
{trajectory}

### Agent's Role:
```

```
{role}

### General Insights:
{insights}

### Your Output (Personalized Insights for This Role):
"""
```

D Discussion with Related Works

In this section, we further discuss the relationship between `G-Memory` and several recent agent memory frameworks. For **A-Mem** [60], while both A-Mem and `G-Memory` aim to enhance the memory capabilities of LLM agents, they differ in two key aspects. First, A-Mem is tailored for single-agent scenarios, whereas `G-Memory` is designed for processing MAS's lengthy and nuanced interaction trajectory. Second, A-Mem emphasizes atomic memory construction for chatbot-style interactions, while `G-Memory` focuses on distilling reusable strategies from collaborative task execution, where fine-grained atomicity is neither required nor beneficial. For **Mem0** [61], although it also employs a graph-based structure, it remains within the chatbot paradigm. Its graph is closer to a knowledge graph, where nodes represent factual entities and edges represent relations, fundamentally differing from `G-Memory`'s agent-centric memory graphs that encode trajectories, decisions, and coordination patterns across agents.