MINT Lab
Machine Intelligence N Trust

Memory in LLM-based Multi-agent Systems: Mechanisms, Challenges, and Collective Intelligence

Shanglin Wu[1] **and Kai Shu**[1]
[1]Emory University

Memory plays a central role in transforming Large Language Model (LLM)-based agents from reactive predictors into consistent, context-aware collaborators. While LLM-based single-agent memory has been extensively studied, memory in LLM-based Multi-Agent Systems (LLM-MAS) lacks a systematic taxonomy and review. In multi-agent contexts, memory becomes a shared cognitive infrastructure enabling collective intelligence, long-term coordination, and team evolvement. This survey provides the first comprehensive review of memory in LLM-MAS, synthesizing research across memory architectures, management and operations, evaluation, and application, while formalizing key definitions and introducing the design space. Our survey reveals that memory in LLM-MAS is not a trivial extension of single-agent memory but a distinct research frontier—with new challenges in synchronization, access control, scalability, alignment, and safety. By consolidating diverse literature, we aim to provide a foundation for future research in building intelligent, memory-augmented multi-agent systems. In order to follow the latest works in this field, we create and maintain a repository at `https://github.com/ShanglinWu/LLM-MAS_Memory_Survey.git`.

1. Introduction

Recent advances in Large Language Model (LLM) agents have further revealed the remarkable understanding and reasoning capabilities of backbone models when equipped with additional modules such as memory, tools, and environment interaction (Luo et al., 2025; Masterman et al., 2024). Among the various modules that enhance LLM-based agents across large number of domains, memory stands out as the key component that transforms the original LLM into a "true agent"(Yu et al., 2025; Zhang et al., 2025d). It enables the agent to maintain long-term context, accumulate and utilize knowledge over time, and coordinate effectively with external sources. Furthermore, the emergence of LLM-based Multi-Agent Systems (LLM-MAS) where multiple agents collaborate and communicate has demonstrated early promising results compared to single-agent setups, as supported by academic research and industrial practices (Guo et al., 2024; Han et al., 2024). Exploring memory within LLM-MAS represents a crucial step toward achieving Artificial General Intelligence (AGI) (Hendrycks et al., 2025), where memory enables agents to characterize human's collaboration behaviors and evolve together in a team under real-world scenarios.

Why memory is crucial for LLM-MAS. LLM-MAS systems divide complex tasks among multiple agents, each assigned distinctive capabilities and roles. Recent work in simulation (Jinxin et al., 2023; Park et al., 2023), role-playing (Li et al., 2024), and reasoning (Du et al., 2023; Liang et al., 2024b) shows the potential of this approach. For these systems to succeed, memory is essential: it allows agents to recall prior subtasks, coordinate handoffs, maintain role histories, share relevant knowledge, and build on earlier steps rather than starting from scratch. Moreover, most MAS failures stem from poor system design or inter-agent misalignment (Cemri et al., 2025), better memory module designing could help mitigate those failures by providing consistent common knowledge and interaction history. From a cognitive science perspective, memory serves as the fundamental

Corresponding author(s): kai.shu@emory.edu

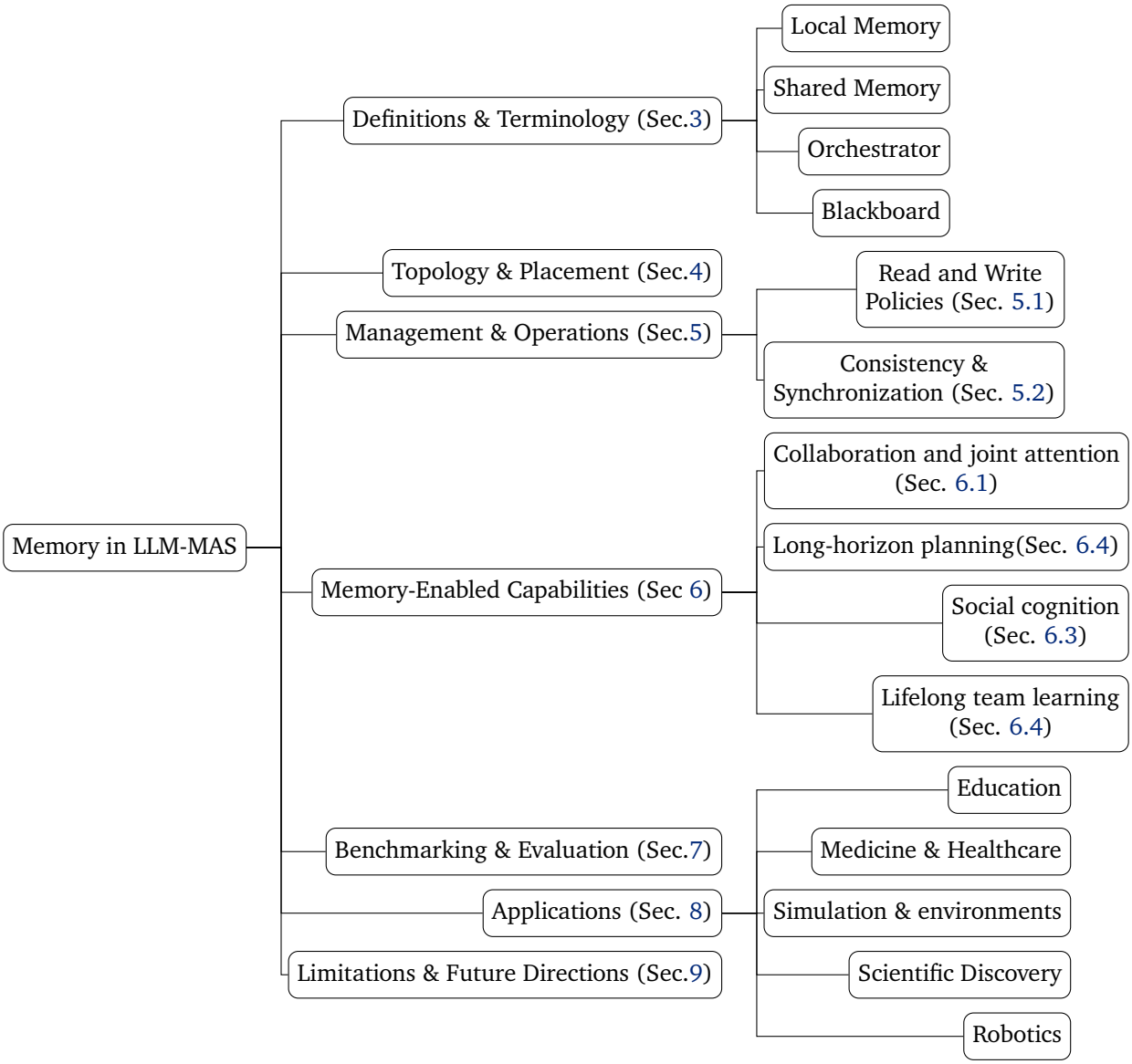

Figure 1 | Overview of the survey structure.

substrate enabling distributed cognition across agent teams, where collective intelligence emerges from the interaction between individual memory systems and shared representational states (Wu et al., 2025b). Just as human teams develop transactive memory systems, i.e., knowing "who knows what", LLM-based agents require similar meta-memory capabilities to efficiently allocate cognitive resources and avoid redundant processing. For instance, working memory constraints in individual agents necessitate external memory architectures that support both episodic recall of specific interactions and semantic abstraction of learned patterns. Theory of mind capabilities, essential for anticipating other agents' actions and maintaining coordination, depend critically on memory of past agent behaviors and established communication protocols (Spreng and Mar, 2012). Without persistent memory spanning multiple interaction cycles, agents may not develop the specialized expertise and mutual adaptation that characterizes effective human teams.

Significant differences between LLM-MAS memory and single agent memory. While recent work has increasingly explored memory mechanisms in single LLM agent, we still lack a thorough understanding of memory in LLM-MAS. Importantly, MAS memory is not just a simple extension of

single-agent memory. Rather, it marks a transition of cognition from individual-level to collection-level, with distinct motivations, architectures, and challenges. In single-agent settings, the primary motivation is to extend the agent's context window and enable learning from its own past trials, typically through a monolithic memory store local and accessible by the agent, with challenges centered on efficient memory operations, management, and interpretability (Zhang et al., 2025d). In contrast, MAS memory is driven by **collective intelligence**: shared memories can transform individual knowledge into a team knowledge base, allowing agents to achieve together what none could accomplish alone (Gao and Zhang, 2024). From the architecture perspective, MAS memory may be centralized, with a shared repository accessible by all agents, or distributed, with each agent maintaining its own memory and relying on synchronization or selective-sharing protocols. Such flexibility introduces challenges unique to multi-agent settings, including information asymmetry-different agents may have access to different knowledge and permissions (Rezazadeh et al., 2025)-as well as consistency and synchronization requirements, since agents must operate on a coherent and up-to-date view of important facts (Zhang et al., 2025d). Furthermore, MAS memory must support coordination across multiple contextual layers: an individual agent's local context, the team's joint context, and the evolving environment state (Han et al., 2024). Together, these distinctions highlight why memory in LLM-MAS requires dedicated study rather than being treated as a straightforward extension of single-agent memory.

Survey Scope. In this survey, we take a deep dive into existing research and discuss how memory functions within LLM-MAS. This perspective completes the broader roadmap toward building truly intelligent and cooperative agent systems. Specifically, we discuss several key aspects of LLM-MAS memory as follows:: (1) Definitions and Terminologies: we formalize define LLM-MAS memory and its related terminologies. (2) Memory Topology and Placement: we analyze how memory is distributed and where it is hosted, surfacing the trade-offs for scalability, robustness, and privacy. (3) Memory Management and Operations: we study the read and write policies, and memory synchronization and consistency in LLM-MAS. (4) Memory-Enabled Capabilities: we show how memory enable collaboration and joint attention, long-horizon planning and reuse, social cognition, and lifelong team learning. (5) Benchmarking and evaluation: emerging benchmarks and metrics to systematically evaluate the effectiveness of memory architectures in multi-agent tasks. (6) Applications: we review practical applications of memory in LLM-MAS across education, medicine and healthcare, simulation and environments, scientific discovery, and robotics. (7) Limitations and future directions: by concluding all the previous discussion, we point out the limitations of exited memory architectures, discuss open issues, and propose future directions for memory in LLM-MAS. The structure of this survey is shown in figure 1.

By synthesizing recent advances along these axes, our survey aims to provide the first comprehensive overview focused specifically on memory in LLM-MAS. We fill that gap by unifying findings from a range of systems, from multi-agent conversation frameworks to embodied task-solving agents, through the lens of implementation and exploitation of memory. We focus on LLM-MAS where agents primarily communicate via natural language and use LLMs for policy generation. By synthesizing memory-centric research with general LLM-MAS frameworks, we identify how memory architectures support agents in both shared-goal and competitive scenarios. This scope reflects the current wave of research that treats language agents as a new paradigm for AI coordination (Gao and Zhang, 2024; Rezazadeh et al., 2025). The common theme is that enabling collective cognition via memory which is a distinct challenge that sets multi-agent systems apart from single-agent ones. Overall, the contributions of this survey are:

- Review the state-of-the-art in memory for LLM-MAS, summarize core concepts, and provide a systematic taxonomy.

- Critically compare memory architectures used in recent systems, integrating theoretical perspectives with practical considerations.
- Compile an extensive collection of relevant literature with concise summaries to serve as a reference for researchers.
- Analyze key limitations of existing frameworks and propose comprehensive future directions.

By elucidating how memory can transform a collection of LLM agents into a coherent team, we hope to identify current challenges and inspire new research on LLM-MAS memory frameworks.

2. Related Areas

2.1. LLM-based Multi-agent Systems.

The fast growth of LLM-based agent research in recent years has spurred several surveys that provide overviews of this broad area. Guo et al. (Guo et al., 2024) present a comprehensive survey of progress and challenges in LLM-based multi-agents, covering aspects like agent communication and role-specific capabilities. They outline how multi-agent systems can leverage collective intelligence by specializing LLM agents for different roles and enabling them to interact, and they identify challenges such as coordination and scalability. Chen et al. (Chen et al., 2024b) survey recent advances in LLM-based MAS with an application-oriented taxonomy, examining how multi-agent frameworks are used for complex task solving and world simulations. Their work extends earlier frameworks and highlights new frontiers that fall outside prior taxonomies, given the rapid influx of papers. Han et al. (Han et al., 2024) focus on challenges and open problems in LLM-based multi-agent systems, emphasizing issues like reliable agent orchestration and the need for more rigorous evaluation methodologies. In addition, specialized surveys have looked at multi-agent collaboration from adjacent angles: for instance, Tran et. al. (Tran et al., 2025) discusses collaboration mechanism for LLM-MAS, and Yan et. al. (Yan et al., 2025) review LLM-MAS from a communication-centric perspective. Practically, many surveys have reviewed LLM-MAS under specific tasks or application domains such as software engineering (He et al., 2025), embodied AI (Feng et al., 2025), and chemical engineering (Rupprecht et al., 2025). These surveys collectively map out the LLM-MAS landscape, but memory is not their primary focus. They mention memory as one mechanism among others, e.g. to maintain context or enable self-evolution of agents, yet none provide a deep dive into memory architectures or systematically compare how different frameworks implement memory. Our work differs by honing in on memory as the lens for organizing and analyzing LLM-MAS literature, allowing us to bridge insights across papers that a general MAS survey might treat only in passing.

2.2. Memory for LLM Agents.

Very recently, a dedicated survey on memory mechanisms for LLM-based agents was published by Zhang et al. (Zhang et al., 2025d). This survey recognizes memory as the key to enabling long-term, complex interactions and attempts a first taxonomy of memory designs in LLM agents. Zhang et al. discuss what memory is in the context of LLM agents and why it is needed, then review various implementations of memory modules and how they are evaluated. They cover memory-augmented models, retrieval strategies, and applications where memory is crucial. While comprehensive, their survey looks at LLM agents in general (often single-agent settings) and does not specifically emphasize multi-agent scenarios. In contrast, our survey focus on multi-agent systems and the unique challenges and opportunities that arise when multiple LLM agents share information. We accept some concepts identified by Zhang et al., such as short-term vs long-term memory, but we contextualize them and propose new taxonomy for multi-agent frameworks. Another related thread is the broader literature

on knowledge integration and retrieval-augmented generation (RAG) for LLMs, surveyed by Mialon et al. (Mialon et al., 2023) and others, which covers how external information can serve as extended memory for LLMs. Our survey intersects with RAG when discussing memory retrieval strategies, but we will also review interactive and procedural memory, i.e., memories formed through the agents' own interactions and actions. Finally, earlier work in cognitive architectures and reinforcement learning (Ramani, 2019; Yoo and Collins, 2022) provide useful perspectives such as episodic memory and semantic memory distinctions, but they predate the advent of LLM-based agents and thus do not address the quirks of using LLMs with prompts and natural language traces as memory. In summary, our survey is the first to synthesize a view of memory in LLM-based multi-agent systems, differentiating itself from prior surveys by its dual focus: marrying the latest LLM agent developments with classic MAS considerations, and doing so through the unifying theme of memory.

3. Definitions and Terminologies

While prior studies on LLM-MAS have proposed various taxonomies and definitions of memory (Chen et al., 2024b; Han et al., 2024), most remain at a narrative level and lack continuity with the well-established frameworks of single-agent memory. In this section, we provide formal definitions of memory within LLM-MAS and clarify related terminologies, while also delineating the the design space of memory architectures in multi-agent settings.

Definitions of memory in LLM-MAS. The ability to self-evolve through interactions with the environment is a fundamental requirement for LLM-based agents. In multi-agent settings, however, interactions also occur between agents themselves, adding another layer of complexity and coordination. Thus, we define memory in LLM-MAS as the collection of information that agents retain from past interactions (with the environment and with each other) to inform future behavior. In single-agent settings, Zhang et al. formalize an agent's memory M as derived from its past observations and actions: narrowly from the history of the current trial, and broadly from multiple trials and outside knowledge (Zhang et al., 2025d). In our multi-agent context, this definition expands from individual to collective cognition. Formally, consider a team of N agents. At step t, the memory of MAS is the union of all agents' local memory and shared memory: $M^{MAS}(t) = \{M_1(t), M_2(t), \ldots, M_N(t)\} \cup M^{shared}(t)$. More specifically, we formally define and discuss memory-related terminologies:

- **Team:** A team is a set of agents designed to work together toward certain tasks or goals, cooperatively or competitively. We denote a team as $T = \{A_1, A_2, \ldots, A_N\}$. Agents in a team interact with each other and often have a shared objective in collaborative scenarios or opposing objectives in competitive scenarios.
- **Local Memory:** A local memory is the memory internal to a single agent. Let $E_t^j = \{\tau_t^1, \tau_t^2, \ldots, \tau_t^p\}$ denotes the collection of p trails for agent A_j, where $\tau_t^p = \{a_1^p, o_1^p, a_2^p, o_2^p \ldots a_t^p, o_t^p\}$ is the history of interacting with environment which consist of past actions and observations. Let $I_t^j = \{i_t^k | i_t^k = \{\epsilon_1^k, \epsilon_2^k, \ldots, \epsilon_t^k\}\}$ denotes history of interacting between A_j with other agents where i_t^k is the history between A_j and A_k and ϵ_t^k is their communication at step t. Let D_t^j denotes external knowledge. Each A_j maintains a local memory $M_i(t) = E_t^j \cup I_t^j \cup D_t^j$. Only A_j can read from or write to its local memory (Rezazadeh et al., 2025). Local memory is analogous to the single-agent memory module in prior works (Zhang et al., 2025d), but in an MAS each agent also has distinct inter-agent communication history. Moreover, in our context, short-term memory and long-term memory including *episodic memory* which refers to context-specific event traces, and *semantic memory* which refers to general facts or rules extracted from experience (Wu et al., 2025b), are both stored within an agent's local memory (Han et al., 2024).
- **Shared Memory:** Shared memory is a memory resource accessible to multiple agents in the

team. It serves as a common knowledge repository or global memory that the team of agents can collectively read from and contribute to. Shared memory M^{shared} could be a vector database, knowledge graph, or document that aggregates information from all agents. Unlike local memory, which is siloed per agent, shared memory breaks those silos, enabling what is essentially a team-wide memory. Shared memory can be centralized or distributed, but in all cases the key property is accessibility by multiple agents. We note that shared memory may be moderated by policies or permissions, especially in competitive scenarios (Rezazadeh et al., 2025).

- **Orchestrator:** An orchestrator is a special agent or module responsible for coordinating the multi-agent system. In some LLM-MAS architectures (Hadfield et al., 2025), one agent is designated as a leader or controller that manages the others. Formally, one can think of an orchestrator A_{orc} as having a high-level policy Π_{orc} that takes the global state and decides which agent should act next or how to merge information. The orchestrator often maintains an overview of the team's shared memory or state.
- **Blackboard:** The blackboard is a classical architecture for decentralized problem solving which was first proposed in 1980s (Hayes-Roth, 1985; Nii, 1986), which we adopt here to mean a shared workspace or memory board that all agents can read and write. In a blackboard system, agents post information onto a common board, and any agent can react to the current contents of the board. The blackboard effectively is a global memory store plus a coordination mechanism. Han et. al (Han and Zhang, 2025) describe a blackboard-based LLM-MAS where "agents with various roles can share all the information and others' messages during the whole problem-solving process", using the blackboard as the persistent state. The blackboard content determines which agent should act and when, and agents iterate until a solution consensus is reached on the board. In our terms, a blackboard is one implementation of shared memory with an associated control loop where there is a supervising logic that picks the next contributor based on the board's state.

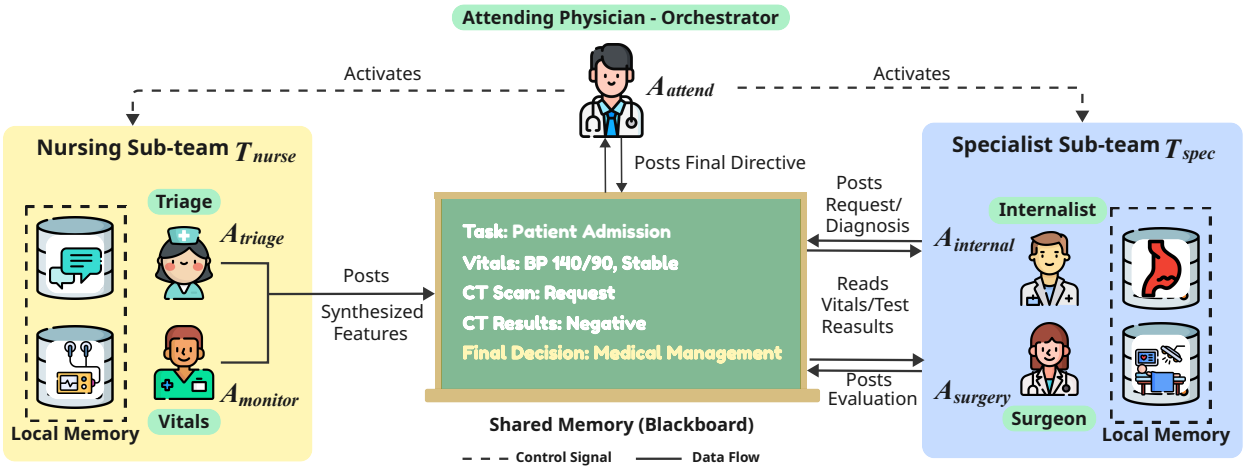

Figure 2 | A LLM-MAS for Collaborative Patient Diagnosis.

Example: Collaborative Patient Diagnosis. To illustrate these definitions, we consider a scenario where a patient presents with ambiguous abdominal trauma (as shown in figure 2). We define a team $T = \{A_{attend}, T_{nurse}, T_{spec}\}$. Here, the orchestrator is the **attending physician** A_{attend}, who manages two sub-teams: the **nursing team** T_{nurse} and the **specialist team** T_{spec}.

- **Initialization (Blackboard):** The hospital admission ("*Patient: Male, 45, acute abdominal pain*") is initialized in the Shared Memory M^{shared}, acting as the central electronic health record. The attending physician A_{attend} reads this state and activates the nursing sub-team T_{nurse} for triage.

- **[Phase 1] Data Collection (T_{nurse}).** The sub-team T_{nurse} consists of a Triage Agent A_{triage} and a Monitoring Agent $A_{monitor}$. A_{triage} conducts the patient interview. Its Local Memory M_{triage} stores the raw, unstructured dialogue transcript I^{raw} ("*Patient complains of pain after eating...*"). Simultaneously, $A_{monitor}$ tracks real-time sensor data in its local buffer. Instead of flooding the blackboard, T_{nurse} processes these local logs and writes only synthesized clinical features to M^{shared}: "*BP: 140/90, History: Post-prandial pain, Status: Stable*".
- **[Phase 2] Specialist Reasoning (T_{spec}).** Observing the updated M^{shared}, the orchestrator A_{attend} recognizes the complexity and summons T_{spec}, comprising an Internist $A_{internal}$ and a Surgeon $A_{surgery}$. $A_{internal}$ accesses its private domain knowledge base $D_{internal}$ which contains pharmaceutical protocols. It hypothesizes "*Acute Pancreatitis*" based on shared vitals and posts a request for a CT scan to M^{shared}. $A_{surgery}$ Accesses its private domain knowledge base $D_{surgery}$ which contains anatomical models. It evaluates the scan results in M^{shared} and rules out obstruction.
- **Synchronization & Consensus.** The specialists may initially post conflicting treatment options to M^{shared}. A_{attend} acts as the conflict resolver, reviewing the evidence chains from both $A_{internal}$ and $A_{surgery}$. A_{attend} synthesizes a final plan ("*Admit for medical management, Surgery on standby*") and updates the Shared Memory with the final directive, releasing the agents.

In this architecture, the Shared Memory acts as a synchronization point between distinct ontologies. $A_{internal}$ and $A_{surgery}$ do not need to access each other's private reasoning steps or the nurses' raw chat logs; they only interact with the crystallized medical facts (symptoms, test results, diagnoses) maintained by A_{attend} on the blackboard.

4. Memory Architecture: Topology and Placement

Memory in LLM-MAS can be structured and deployed in multiple ways, and these choices fundamentally shape how agents coordinate, share information, and maintain long-term state. The *topology* determines who can read or write which parts of memory; the *placement* specifies where that memory resides within the system; and both factors introduce practical engineering trade-offs. By examining these dimensions together, we can understand how different MAS designs balance scalability, robustness, privacy, and coordination efficiency, and why no single memory architecture fits all real-world settings. Table 1 summarizes representative LLM-MAS frameworks based on their memory topology and placement. Figure 3 demonstrates different memory topology and placement patterns in LLM-MAS.

4.1. Topology Patterns

An LLM-MAS memory can be organized in **local, shared, or hybrid topologies**, referring to who has read/write access to the memory. In a **local-only topology**, each agent maintains its own private memory store and does not directly access others' memories. Early frameworks like CAMEL (Li et al., 2023), AutoGen (Wu et al., 2024b), Chain-of-agents (Zhang et al., 2024b) mainly rely on local memory and inter-agent communication to exchange knowledge. This isolation avoids interference between agents but risks redundancy – the same fact may be stored by multiple agents – and inconsistency if one agent updates a belief that others don't witness. By contrast, a **shared topology** gives all agents access to a common memory repository. For example, the Memory Sharing framework (Gao and Zhang, 2024) provides a shared memory pool that every agent can read and write. This enables a form of team mind where knowledge is immediately global, supporting tight coordination and "joint attention". However, unstructured sharing can lead to a noisy commons: irrelevant or low-value details accumulate and distract agents. Moreover, naive shared memory overlooks access distinctions – every agent sees everything, raising privacy and role segregation issues. **Hybrid topology** seek

Table 1 | Mapping of representative LLM-MAS frameworks.

Frameworks	Topology			Memory Placement		
	Local	Shared	Hybrid	Per-agent	Orchestrator-level	External
Anokhin et al.; Yang et al. Mandi et al.; Zhang et al. Li et al.; Qian et al. Zhang et al.	✓			✓		
Trivedi et al.; Zhang et al.	✓			✓		✓
Wu et al.	✓			✓	✓	
Gao and Zhang		✓				✓
Hong et al.		✓		✓		✓
Han and Zhang		✓			✓	
Rezazadeh et al.; Zhang et al. Han et al.			✓			✓
Fourney et al.; Liu et al. Hu et al.; Zhu et al. Wan et al.			✓	✓	✓	
Liu et al.			✓		✓	✓
Hadfield et al.			✓	✓	✓	✓

a balance: agents have individual memory modules but also contribute to some shared stores. A common pattern is an orchestrator or coordinator agent that maintains a global memory, while worker agents keep local memories. For example, this is seen in LEGOMem (Han et al., 2025) and Antropic's multi-agent research system (Hadfield et al., 2025),where the orchestrator's memory holds the high-level team memory, and each specialist agent records details of its own task execution. Such hybrid setups reduce clutter (not every detail goes into the global store) and limit cross-talk (agents read the orchestrator's summary instead of each others' raw traces), while still enabling collective reasoning through the orchestrator.

4.2. Memory Placement

Orthogonal to topology is the question of **where the memory module is hosted**, that is, the physical or logical location of system's memory modules. Memory placement typically falls into three categories. First, in **per-agent hosting**, each agent maintains its own internal memory store tightly coupled to its architecture. For example, an agent may embed a private dialogue document (Light et al., 2023) or maintain a local knowledge graph (Anokhin et al., 2024) for long-term factual retention. This pattern is common in local-topology systems. Second, in **orchestrator-level hosting**, a central coordinator acts as the memory hub for the entire team. Although the orchestrator may not contain domain knowledge itself, it aggregates, summarizes, and indexes information on behalf of all agents, following a blackboard-style pattern. Frameworks such as PC-Agent (Liu et al., 2025a) adopt this design, where a manager agent maintains the evolving global task state and worker agents complete their role-specific sub-tasks. Finally, memory may reside in the **external hosting**. In this case, no single agent "owns" the memory. Instead, all agents query a shared database (Wang and Chen, 2025), a knowledge graph (Zhang et al., 2025a), or a document (Han and Zhang, 2025). Such external memory stores are highly persistent and can scale far beyond an LLM's context window, but introduce

additional latency, explicit query formulation, and the need for concurrency control when multiple agents read or write simultaneously.

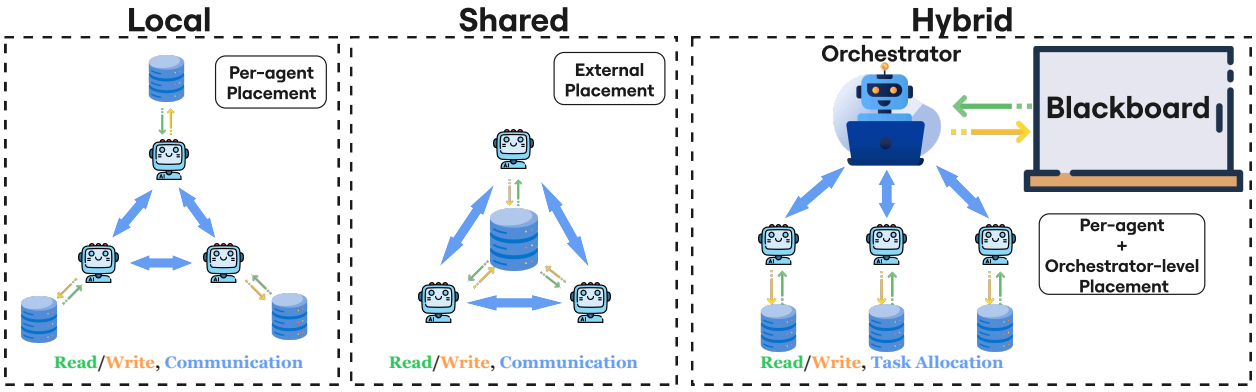

Figure 3 | Topology and placement of LLM-MAS memory.

4.3. Engineering Trade-offs

Different memory topologies introduce trade-offs in scalability, robustness, and privacy. A centralized shared memory is simple but risks becoming a throughput bottleneck and a single point of failure. Fully local memories scale well and avoid global crashes, but agents must either duplicate common knowledge or communicate frequently to stay synchronized. Shared memories also face write contention, requiring read/write policies or an orchestrator to serialize updates; hybrid designs alleviate conflicts by channeling modifications through designated roles. For robustness, distributed memory is advantageous: losing one agent does not erase the group's knowledge, whereas centralized memory requires replication or checkpointing to avoid catastrophic loss. Privacy concerns further shape design: shared memories expose all content unless access control is added, while local memories naturally restrict visibility but complicate selective sharing. Partitioned blackboards, role-scoped permissions, and private/public tiers are practical compromises. In short, no single topology fits all settings. Small cooperative teams may prefer centralized memory for efficiency, while larger or heterogeneous MAS need modular, role-aware memory layouts.

5. Memory Management and Operations

Effective memory management in LLM-MAS necessitates moving beyond simple storage and retrieval to address the intricacies of concurrent interaction and distributed decision-making. In this section, we examine memory read and write policies alongside consistency and synchronization protocols, reviewing recent works that target the unique challenges posed by LLM-MAS.

5.1. Read and Write Policies

In multi-agent settings, memory operations become significantly more complex than in single-agent systems because multiple agents may read or write concurrently. As a result, write policies must account not only for what information is important, but which agent is producing it, how trustworthy or role-relevant it is, and how it should be stored relative to other agents' contributions.

On the write side, MAS rarely benefit from a uniform curation or summarization rule. Different agents may write at different abstraction levels depending on their roles: an orchestrator may store

high-level task summaries, while worker agents write fine-grained procedural traces; a critic or evaluator agent may write observations with higher importance weight (Nii, 1986; Rezazadeh et al., 2025). This leads to role-conditioned summarization, where the system applies more aggressive compression to low-level agent traces but retains detailed logs from agents whose information is central to coordination. In shared or hybrid memories, concurrent writes require conflict-resolution rules (Fourney et al., 2024): the system must decide whose update takes precedence and whether to merge competing entries. Some MAS further support multi-resolution summaries (Liu et al., 2025b; Zhang et al., 2025b), where the memory keeps both a global summary and agent-specific fine-grained logs, promoting or demoting entries based on team importance scores or task relevance.

On the read side, multi-agent retrieval requires considering both who is asking and how many agents are querying at once. Instead of a single retrieval policy, MAS often adopt role-adaptive retrieval (Wan et al., 2025): an orchestrator retrieves high-level global state, planners retrieve strategic memories, and worker agents retrieve localized or skill-specific entries (Hong et al., 2023). When many agents query simultaneously, systems must prioritize retrieval to avoid contention or prompt overflow in shared memories. In hybrid systems, agents may read from their own local memories while only requesting global memories when necessary, reducing latency and cross-talk. Additionally, MAS require explicit access-ordering policies: agents may issue conflict read/write, so the system needs locking, versioning, or orchestrator-mediated serialization to maintain a consistent team state (Hadfield et al., 2025; Nii, 1986). Some frameworks explore learning these policies—training or optimizing agents to decide when to write, what granularity to write at, or how aggressively to retrieve—yielding more efficient memory use in large teams (Yang et al., 2025; Zhou et al., 2025).

5.2. Consistency and Synchronization

In LLM-MAS context, different agents may maintaine compatible views of key facts. Whenever memory is shared, multiple agents may read and write concurrently, so naïve single-agent assumptions i.e., the agent always sees the latest state break down. MAS therefore need explicit protocols for when and how agents can update shared state, and how those updates propagate to others.

The simplest MAS-specific mechanism is serialized turns: agents act one at a time in a fixed order, each reading the latest memory and then writing its updates. Many dialogue-style and debate systems explicitly or implicitly use this turn-taking pattern, which avoids simultaneous writes (Liang et al., 2024a; Qian et al., 2024a). However, once agents are allowed to act in parallel, stronger synchronization primitives are needed. Task-level updates ensure that logically coupled changes are applied atomically from the perspective of other agents (Hong et al., 2023; Trivedi et al., 2024). Furthermore, MAS must also keep agents aligned with the evolving shared state. Publish/subscribe patterns are particularly natural here: agents subscribe to topics according to its role such as "global plan" or "Probability Theories" (Han and Zhang, 2025; Han et al., 2025). When an orchestrator updates a plan, all subscribed workers are notified and can refresh their local view, reducing the risk that they continue reasoning based on stale information. Collaborative Memory (Rezazadeh et al., 2025) highlight that, without such sync mechanisms, loosely-coupled agents may update knowledge that others remain unaware of, leading to divergent beliefs and coordination failures.

Conflict resolution is also a genuinely multi-agent concern. Different agents can arrive at incompatible conclusions about the same fact and attempt to write conflicting entries. A simple last-writer-wins policy is often unsafe if a less competent agent overwrites a better assessment. MAS therefore benefit from role or confidence-aware schemes: an orchestrator, a voting mechanism, or a trust model can arbitrate which entry becomes authoritative (Han and Zhang, 2025). Provenance is crucial for this: memory entries should track metadata such as author agent, timestamp, confidence, and evidence (Han et al., 2025; Rezazadeh et al., 2025). When conflicts arise, the system can inspect provenance to

decide which update to keep or to trigger a reconciliation step. Provenance also supports explainability in MAS: an agent can justify a decision by referencing who wrote it and why.

6. Memory-Enabled Capabilities

Designing effective memory for LLM-MAS is not just an engineering pursuit it directly expands the capabilities of those systems. By shifting from purely individual cognition to collective cognition, memory allows an ensemble of agents to achieve tasks and behaviors that would be impossible without it. In this section, we discuss several high-level capabilities unlocked by memory in LLM-MAS: (1) improved collaboration through shared situational awareness and joint attention, (2) the ability to tackle long-horizon tasks via planning and reuse of knowledge, (3) emergent social intelligence like tracking others' knowledge and commitments, and (4) lifelong learning as a team, accumulating experience over multiple problems.

6.1. Collaboration and joint attention

In LLM-MAS the transition from individual performance to team capability is mediated fundamentally by memory. Collaboration, defined as the coordinated exertion of effort by multiple agents toward a shared objective, requires a mechanism to synchronize the internal states of distinct agents. In human cognition, this is achieved through joint attention—the shared focus of two individuals on an object (Dunham, 1995). In multi-agent systems, memory serves as the functional equivalent of this joint attention, acting as the shared cognitive substrate upon which collaboration is built.

The Mechanism of Joint Attention. The primary challenge in multi-agent collaboration is the "Context Fragmentation" problem (Krishnan, 2025). In local-only topology where agents possess only private memory, no single agent holds the complete state of the collaborative effort. This fragmentation necessitates continuous, high-volume communication to maintain alignment, scaling poorly as the number of agents N increases, with communication overhead approaching $O(N^2)$. Shared and hybrid topology resolve this by externalizing the joint context. Shared memory functions as a persistent "Team Mind". When an agent updates the shared memory, it effectively shifts the attention of all other "subscribing" agents to this new state (Hong et al., 2023). This mechanism mimics human "common ground" theory (Stalnaker, 2002), where collaboration is predicated on mutual knowledge, i.e., "I know that you know that I know". In LLM-MAS, the shared memory stores this mutual knowledge explicitly (Heller et al., 2025). In hybrid topology, collaboration is further refined by separating global context from local context. Usually, the orchestrator agent maintains a high-level memory of the overall plan, while worker agents maintain granular memories of their specific sub-tasks. The joint attention is established at the intersection of these memories: the orchestrator publishes a task to the shared space, and the worker ingests it into its private space. This tiered memory structure allows for **Selective Joint Attention**, where agents can collaborate on specific modules without being overwhelmed by the cognitive load of the entire system state. This addresses the "distraction" problem in massive shared memories, where irrelevant updates dilute the agent's focus.

Situational Awareness. Effective collaboration requires Situational Awareness (SA)—the perception of environmental elements, the comprehension of their meaning, and the projection of their status (Salmon et al., 2010). In LLM-MAS, SA is a function of memory retrieval capabilities. An agent's "awareness" is defined by what it can retrieve from the collective store. Information Asymmetry (Clarkson et al., 2007) is a critical barrier to collaboration. If agent A possesses critical information in its local memory that agent B lacks, agent B's actions may become misaligned with the goal. Memory architectures mitigate this through synchronization. By implementing a Publish-Subscribe pattern (Chen and Shiu, 2025), agents can push critical updates from local memory to shared memory and

reducing asymmetry. For example, in a medical diagnosis MAS (Tang et al., 2024), a Radiology Agent might identify an anomaly. If this remains in local memory, the Clinical History Agent proceeds on false assumptions. By writing findings to the shared patient record, the system forces a synchronization of states. Table 2 provides an analysis of the impact of memory on agent collaboration in LLM-MAS.

Table 2 | Comparative Analysis of Memory-Enabled Collaboration

Feature	**Local-Only Topology**	**Shared/Hybrid Topology**	**Collaboration Impact**
Information Flow	Pair-wise message passing.	Broadcasting via Shared State.	Shared memory reduces latency and ensures all agents have equal access to critical updates.
Asynchrony	Blocking: Agent A waits for Agent B.	Non-blocking: Agent A writes, Agent B reads when ready.	Decouples agent workflows, allowing parallel processing and higher throughput.
Join/Leave	New agents require full context dumps.	New agents "Onboard" by reading Shared Memory.	Enables dynamic scaling of teams; agents can be added mid-task without disrupting the workflow.

6.2. Long-horizon planning and knowledge reuse

Complex real-world tasks usually require long-horizon planning (Erdogan et al., 2025), where actions are coherent across large amount of steps. Memory is the mechanism that extends the agent's temporal horizon, enabling the persistence of intent and the reuse of acquired knowledge.

Hierarchical Planning. Long-horizon tasks cannot be solved by a single, monolithic plan. They require decomposition into sub-goals, tasks, and atomic actions. Memory enables this through hierarchical representation (Zhang et al., 2025c). The orchestrator agent utilizes a high-level memory structure to store the strategic plan—the dependency graph of major milestones. This memory is relatively stable and abstract. Crucially, memory allows for plan fluidity. In a static system, a plan is generated once and followed blindly. In a memory-augmented system, the orchestrator agent monitors the feedback stored in shared memory by workers. If a worker reports a failure, the orchestrator retrieves the strategic plan and performs re-planning. It modifies the dependency graph in memory, pruning the failed path and instantiating a new strategy. This Read-Modify-Write loop on the plan itself is what constitutes "adaptive planning" (Lee et al., 2019).

Episodic to Semantic Consolidation for Knowledge Reuse. A hallmark of intelligent systems is the ability to do more with less over time (Gregor and Benbasat, 1999). This is achieved through knowledge reuse (Da Silva et al., 2018)—the application of previously learned solutions to new, similar problems. This capability relies on the transformation of memory from episodic (what happened) to semantic (how things work) (Wu et al., 2025b). In systems like Voyager (Wang et al., 2023) or MetaGPT Hong et al. (2023), when an agent solves a novel problem, the interaction trace is initially stored in shared episodic memory. A background process analyzes these traces. It identifies the successful pattern and abstracts it into a generalizable skill or rule. This new skill is then written to the shared semantic memory. Future LLM-MAS, when faced with a similar task, should query the semantic memory before attempting generation. This retrieval-augmented planning has profound implications (Lee et al., 2024):

- Efficiency: Retrieving a solution is computationally cheaper than generating a new one.

- Reliability: Retrieved solutions that have been verified by past execution and interactions, whereas generated solutions are stochastic and less reliable.
- Compound Innovation: Agents can build complex plans by composing simple, retrieved skills. The memory acts as a "scaffolding", allowing the agent to reach higher levels of complexity without managing the low-level details of every sub-component.

6.3. Social cognition and Theory of Mind

Social cognition involves the processes used to understand and interact with other social agents (Sun, 2006). A key component is Theory of Mind (ToM)—the cognitive capacity to attribute mental states to oneself and others, and to understand that others have beliefs different from one's own (Carlson et al., 2013). In LLM-MAS, where agents must coordinate or compete, memory is the basement that makes ToM possible. This capacity is essential for agents to anticipate actions, resolve conflicts, and engage in sophisticated social planning within the multi-agent environment.

Modeling "Others" via Local Memory. In a local-only topology, each agent maintains a private log of its past interactions. To function effectively, agent A must maintain a model of agent B within its own memory. This model is constructed from the history of messages received from B, i.e., I_t^j in Section 3. By analyzing this history, agent A can infer agent B's Knowledge State. For example, if agent A sent a specific file to agent B at step $t-k$, agent A's memory records that "agent B has the file." This prevents agent A from redundantly re-sending the file. This Recursive Belief Modeling (Moreno et al., 2021), i.e., I believe that you believe..., reduces communication overhead and prevents Message Storms. Furthermore, agents can model capabilities. For instance, if agent B successfully completed a "Python Coding" task in the past, agent A will learn to route future coding requests to agent B. Conversely, if agent B failed, agent A adjusts its model to "agent B is unreliable at Python". This dynamic role adaptation (Li et al., 2015) allows the team to self-optimize. The "social network" of the agent team is effectively a graph stored in the distributed memories of the agents, where edge weights represent trust and capability assessments.

Social Dynamics in Simulation. In simulation environments like Generative Agents (Park et al., 2023) or AgentSociety (Piao et al., 2025) , memory is the engine of personality and social realism. Agents utilize Associative Memory Retrieval to simulate human-like recall. When agent A sees agent B, it queries its memory for past interactions with B. If the retrieved memories are positive, agent A adopts a cooperative stance. If they are negative, it adopts an antagonistic or avoidant stance. This capability allows for the emergence of relationships. A "friendship" in an LLM-MAS is algorithmically defined as a cluster of positive episodic memories linking two agent IDs (Chen et al., 2020). These systems also utilize memory to encode Social Norms (Hechter and Opp, 2001). Norms are essentially semantic memories regarding acceptable behavior in the group, for example, "Agents must queue for the coffee machine". When an agent violates a norm and is corrected, this event is stored. The aggregation of such memories across the population constitutes culture of the simulation. New agents "socialize" by accessing this shared cultural memory or by interacting with agents who enforce it based on their own memories.

Strategic Behavior in Competitive Environments. In competitive or adversarial LLM-MAS, such as game theory simulations (Malinovskiy, 2025) or debate frameworks (Hu et al., 2025c; Liang et al., 2024a), memory enables deception, bluffing, and counter-strategy. To deceive and persuade effectively, an agent must maintain two parallel memory tracks: (1) True State: The actual facts. (2) Public State: The false narrative presented to the opponent (e.g., "I acted confident"). This dual-tracking capability is computationally demanding but essential. If an agent "forgets" its own statement, the deception collapses. Memory also enables opponent modeling (Jing et al., 2025). Agents analyze the historical traces of their opponents to identify patterns. This modeling, based on

longitudinal memory data, allows agents to exploit sub-optimal behaviors in others. In benchmarks like Avalonbench (Light et al., 2023), the win rate of an agent is directly correlated with the depth and accuracy of its memory regarding opponent history.

6.4. Lifelong team learning

Lifelong learning in single agents focuses on preventing catastrophic forgetting (Zheng et al., 2025). In LLM-MAS, lifelong team learning is the capability of the system to improve its performance, efficiency, and knowledge base over an indefinite number of interaction cycles, effectively evolving a "team culture" or "hive mind" that transcends individual agent instantiations.

The Accumulation of Semantic Assets. The primary mechanism of lifelong learning is the continuous consolidation of episodic experience into semantic assets (Tarale et al., 2025). As the team operates, the shared memory grows by: (1) SOP Refinement: The team can "reflect" on a completed project. If a particular workflow step caused delays, the orchestrator agent can update the Standard Operating Procedure (SOP) stored in memory (Wagner, 2014). Future instantiations of the team, reading this updated SOP, will operate more efficiently. (2) Domain Adaptation: A generalist agent team deployed to a medical domain will initially struggle with terminology. Over time, as it processes medical texts and receives feedback, it populates its semantic memory with domain-specific mappings.

Table 3 | Evolution of LLM-MAS: Mapping memory architecture sophistication to corresponding system behaviors and learning capabilities.

Stage	Memory Capability	System Behavior
Transient	Context Window Only	Reactive. No learning. Repeats errors.
Episodic	Interaction Logs (Local/Shared)	Reflective. Can backtrack and explain past actions.
Semantic	Skill/Rule Extraction	Adaptive. Reuses solutions. Faster execution on repeat tasks.
Lifelong	Automated Pruning & SOP Evolution	Evolving. Self-optimizing workflows. Inter-generational transfer.

Inter-Generational Knowledge Transfer. In many MAS deployments, agents are ephemeral—spun up for a task and terminated upon completion (Jonnakuti, 2023). Memory provides the Continuity mechanism by: (1) Onboarding: A new agent instance acts as a "new hire". Instead of starting blank, it reads the team manual and project history from the shared memory. It instantly inherits the wisdom of its predecessors. (2) Model-Invariant Knowledge Persistence: The memory serves as a bridge between model generations. A knowledge base built by GPT-3.5 agents can be utilized by GPT-4 agents. The reasoning trails and interaction history stored in memory survive the upgrade of the models, allowing for a stable evolutionary path for the system. In table 3, we categorize the development of LLM-MAS into distinct stages based on their memory infrastructure, outline this evolutionary trajectory, and illustrate how the shift from transient contexts to persistent, lifelong memory fundamentally alters system behavior from reactive to self-optimizing.

7. Benchmarking and Evaluation

Evaluating memory in LLM-MAS requires metrics and benchmarks that capture both how well agents remember, retrieve, and update memory, also how memory affects coordinated downstream behavior. As recent works show, memory failures remain a dominant bottleneck—affecting task success rates, long-horizon planning, and inter-agent communication quality (Cemri et al., 2025).

Current evaluations span both task-level performance and memory-specific capabilities. This section synthesizes the emerging landscape of benchmarks and metrics, highlighting both what the community can measure today and the critical gaps that must be addressed to assess memory in LLM-MAS.

7.1. Task-Level Evaluation

Task-level evaluation assesses memory's impact on downstream performance across diverse multi-agent scenarios. For example, OfficeBench (Han et al., 2025) evaluate procedural memory through 300 office automation tasks and MultiAgentBench (Zhu et al., 2025b) establishes comprehensive evaluation across research collaboration, Minecraft building, and adversarial games, implementing memory stores with task-specific success metrics:

$$Success\ Rate = \frac{Correct\ Task\ Completions}{Total\ Tasks} \times 100\%$$

While other benchmarks evaluate Deduction Accuracy (Han et al., 2025), Trajectory Efficiency Score (TES), and Incremental TES (ITES) (Sun et al., 2025) for more fine-grained evaluation.

7.2. Memory-Specific Evaluation

Memory-specific benchmarks directly target retrieval, retention, and consistency capabilities. For instance, MemoryAgentBench (Hu et al., 2025b) establishes the most comprehensive framework through 17 datasets spanning four core competencies: Accurate Retrieval (RULER-QA, NIAH-MQ), Test-Time Learning, Long-Range Understanding, and Conflict Resolution. Key metrics include:
Dynamic Memory Recall Probability:

$$P_{recall}(t,r) = exp(-\lambda(n) \times t) \times r, \lambda(n) = \lambda_0 \times (1 - \beta \times tanh(\gamma \times n)$$

where $\lambda(n)$ represents adaptive decay modulated by recall frequency n, and r applies cosine similarity between query and memory embeddings.
Memory Write Accuracy:

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N} Score_{LLM}(response_i, reference_i)$$

Inter-Agent Knowledge Consistency:

$$IAKC = \frac{1}{|A|}\sum_{i\in A}\sum_{j\in A, j\neq i} sim(K_i, K_j)$$

Other benchmarks such as LoCoMo (Maharana et al., 2024) targets conversational memory across 9,209 tokens/300 turns with 7,512 questions, LongMemEval (Wu et al., 2024a) tests five core abilities through 500 questions, MemTrack (Deshpande et al., 2025) evaluates cross-platform memory through Linear/Slack/Git integration, and MemBench (Tan et al., 2025) distinguishes memory levels (factual vs reflective) and interaction scenarios, evaluating effectiveness, efficiency, and capacity through:

$$MQI = \omega_1 \times Accuracy + \omega_2 \times Efficiency + \omega_3 \times Capacity$$

Where $Efficiency = 1/(latency \times cost)$ and $Capacity = max\_memory\_size \times retention\_rate$. These benchmarks complete the evaluation landscape for memory in LLM-based agents.

While specialized memory benchmarks and general multi-agent benchmarks exist independently, a comprehensive benchmark that specifically evaluates memory in multi-agent contexts is still lacking.

Current memory benchmarks focus on single agents, missing critical multi-agent dimensions. General multi-agent benchmarks lack memory-specific metrics for retention decay, retrieval precision, and temporal consistency. The field urgently needs unified benchmarks measuring both task performance and memory efficiency, evaluating cross-agent knowledge consistency, and stress-testing memory under realistic multi-agent coordination scenarios. Adapting existing frameworks represents a promising starting point for comprehensive LLM-MAS memory evaluation.

8. Application of Memory in LLM-MAS Systems

In this section, we examine how memory-augmented LLM-MAS are applied across a wide range of real-world domains. We focus on several representative areas: Education, Medicine and Healthcare, Simulation and Environments, and Robotics. For each, we discuss both established systems and emerging research prototypes, highlighting how memory integration enhances capabilities. In addition, we reference relevant single-agent frameworks that inspire and inform the design of memory architectures within multi-agent contexts.

Education. LLM-based multi-agent systems are increasingly applied in education, where memory enables sustained learner modeling, personalized feedback, and long-horizon pedagogical planning. MAS frameworks such as EduAgent (Xu et al., 2024) and CGMI (Jinxin et al., 2023) simulate teacher–student interactions, using persistent memory to track dialogue histories, prior assignments, and evolving learning trajectories. Classroom Simulacra (Xu et al., 2025) similarly leverages long-term contextual memory to model realistic classroom dynamics over extended periods. On the learner side, systems like PEERS (Arana et al., 2025) build multi-agent cognitive profiles by storing errors, misconceptions, and engagement patterns. Memory further underpins adaptive tutoring systems such as Goal-oriented Learning Agents and OATutor (Pardos et al., 2023), enabling continuity across sessions and more reliable knowledge tracing. Domain-specific educational agents—including MACM for mathematical reasoning (Lei et al., 2024), ProtChat for protein analysis (Huang et al., 2024), and systems for medical or legal training (Sun et al., 2024; Wei et al., 2024) show how memory-rich MAS support specialized, multi-step reasoning. Overall, LLM-MAS excel in education because memory allows agents to maintain long-term student models, coordinate pedagogical roles, and deliver personalized, context-aware learning experiences that evolve over time.

Medicine and Healthcare In medicine, LLM-MAS function as AI analogues of multidisciplinary clinical teams, where memory is the central mechanism enabling longitudinal patient understanding and coordinated reasoning across specialist agents. Systems such as CARE-AD (Li et al., 2025), MedAgents (Tang et al., 2024), and other multi-specialist frameworks (SHENG, 2025) synchronize radiology, genetics, and clinical-history agents through a shared or hybrid patient record, allowing each agent to reason over a unified case profile. Memory ensures that past symptoms, lab results, and intermediate findings are preserved across rounds of communication, preventing information loss as the agents critique and refine each other's conclusions. Architectures often adopt a blackboard-style shared memory or iterative summarization, as in MedAgents(Tang et al., 2024) where a report-assistant agent compresses multi-agent conversations into persistent context for the next round. Medical LLM-MAS also face strict requirements for consistency, auditability, and privacy: memory must be role-filtered , versioned, and explainable to support clinical trust. Benchmarks like MedAgentBench (Jiang et al., 2025) test multi-agent reasoning over EHRs, emphasizing the need to integrate memory with structured medical workflows. Memory ultimately allows medical LLM-MAS to deliver coherent, temporally grounded reasoning-much like human clinical teams building a shared mental model of the patient over time.

Simulation Environments. Simulation environments are a natural fit for LLM-MAS, where

memory provides the backbone for agents to exhibit coherent personalities, relationships, and world models over long horizons. Generative Agents (Park et al., 2023) pioneered this area by equipping 25 simulated characters with episodic and reflective memory modules, enabling believable daily behavior. Subsequent systems—including large communities such as Artificial Leviathan (Dai et al., 2024) and AgentSociety (Piao et al., 2025) use memory to encode personal histories, social interactions, locations, and goals. Memory retrieval strategies such as recency, importance filtering (Park et al., 2023), embedding-based search (Hong and He, 2025), allow agents to act based on past events rather than reset each timestep. Because simulated societies often use decentralized memory—each agent storing its own journal—communication, while the environment itself provides persistent world state. Whether modeling cooperation, social norms, or emergent phenomena, memory is the mechanism that allows simulated agents to maintain continuity, pursue long-term objectives, and form stable relationships that evolve over time.

Scientific Discovery. At the mean time, LLM-MAS are also being applied to assist in scientific research, effectively acting as AI co-researchers that can hypothesize, experiment, analyze data, and write reports. Unlike single-agent research assistants, the frontier is teams of specialized agents that together mimic the scientific method (Ren et al., 2025). For example, Google's AI Co-Scientist (Gottweis et al., 2025) uses a team of agents for automated scientific discovery. It separates roles such as literature review agent and experimental designer. The agents work in parallel and exchange findings via a shared memory, enabling a "generate, debate, and evolve" workflow reminiscent of human research labs. Virsci (Su et al., 2025), where one agent proposes a novel idea, another acts as a critic or peer-reviewer to identify flaws, and others incorporate external knowledge. Moreover, ChemCrow (Bran et al., 2023) used a multi-step agent approach for chemistry and Coscientist (Boiko et al., 2023) defined a multi-agent autonomous lab for chemical experiments with different agents. Scientific discovery necessitates collective and long-term intelligence, making persistent memory essential for LLM-MAS to function as a unified "lab notebook" that integrates specialized knowledge and prevents redundant experimentation(Hadfield et al., 2025). To manage scale, systems often adopt hierarchical patterns, such as orchestrator agents that merge researchers' reports into a global log, often supplemented by pruning strategies to maintain long-term coherence.

Robotics. In robotics and embodied AI, LLM-MAS improve control and coordination by distributing perception, planning, and motor reasoning across specialized agents, with memory enabling persistence across sensor updates and multi-step tasks. MALMM (Singh et al., 2024) exemplifies this trend by using three GPT-4 agents—one for high-level planning, one for vision, and one for motion control—whose shared memory allows them to refine each other's outputs and outperform single-agent baselines in manipulation tasks. Multi-robot systems such as GenRobotics (Wu et al., 2025a) or Emos (Chen et al., 2024a) similarly rely on hybrid memory architectures where each robot maintains local perceptual memory while sharing summarized world-state updates for coordination. Memory allows robots to maintain temporal continuity, remember prior failures, refine long-horizon plans, and exchange situational awareness.

9. Limitations and Future Directions

The research coverage on memory for LLM-MAS, while growing, is still in its early stages and exhibits clear limitations. Many multi-agent frameworks lack dedicated persistent memory, relying only on prompt-level context or ad-hoc history sharing; as Rezazadeh et al. (Rezazadeh et al., 2025) note, most either omit persistent memory entirely or assume a fully shared store. This is a crucial limitation: agents in such systems cannot retain knowledge beyond the immediate context window or they risk overwriting each other's memory without control. Even specialized MAS memory designs remain point solutions that do not yet form a coherent design space. Technically, current methods struggle

with scalability and storage overhead, i.e., memory bloat, slow retrieval, lack of principled pruning, redundancy and knowledge silos, and the absence of standardized protocols or middleware for memory representation, retrieval, access control, and provenance. Each system tends to implement its own bespoke memory layer, making it hard to reuse components, compare approaches, or guarantee security and privacy. Addressing these limitations opens several concrete research avenues:

- **Cross-agent pruning, compression, and forgetting.** Future research must move beyond local context management to develop pruning and summarization mechanisms that operate across multiple agents. This involves sophisticated algorithms capable of deduplicating shared experiences and merging overlapping interaction traces from different agents into a unified state. We need coordinated forgetting protocols (Salwa et al., 2025) that can identify and discard noise while preserving team-critical information, thereby maintaining the collective knowledge base's relevance while reducing storage costs and retrieval latency.
- **Cost control and memory test-time scaling.** As agent teams grow, the economic and computational costs of memory operations escalate (Qian et al., 2024b). Future work should explore token-efficient and latency-aware memory policies that optimize the trade-off between performance and resource consumption. This includes the development of scalable lifelong memory stores and multi-resolution summarization techniques, where the system dynamically adjusts the granularity of stored information based on its query frequency. Additionally, adaptive write-frequency strategies are needed to maintain memory usefulness under the pressure of large teams or long-running tasks without overwhelming the underlying storage infrastructure.
- **Benchmarks and metrics for collaborative memory.** There is an urgent need to establish standardized benchmarks that explicitly test multi-agent memory performance rather than just single-agent retrieval. These benchmarks must evaluate specific LLM-MAS dimensions, such as cross-agent consistency, retention capabilities during long-horizon collaboration, and retrieval precision during complex, coordinated planning tasks. Metrics should also measure the "time-to-synchronization" to understand how quickly a new memory entry propagates across the agent team.
- **Memory-driven alignment and safety.** Finally, persistent shared memory should be investigated as a substrate for encoding ethical norms, safety constraints, and historical failure modes (Anwar et al., 2024). By maintaining shared "risk logs," alignment charters, and immutable audit trails, agents can learn from the collective mistakes of the team and avoid repeating unsafe behaviors. Future work should strictly study how such alignment memories are written, retrieved, and protected—potentially via provenance tracking, cryptographic guarantees, or trust-weighted update policies—to ensure that the moral compass of the multi-agent system remains intact during autonomous operation.

10. Conclusion

Memory is the foundation that transforms a collection of LLMs into a coherent multi-agent system capable of long-term reasoning, coordination, and collective intelligence. By reviewing memory architectures, operational mechanisms, synchronization strategies, benchmarks, and applications across diverse domains, this survey shows that memory in LLM-based multi-agent systems is not a trivial extension of single-agent memory but a distinct research frontier with unique challenges in consistency, access control, scalability, and safety. As multi-agent systems increasingly undertake complex real-world tasks, the need for robust, structured, and interoperable memory grows more urgent. We hope this survey provide both a conceptual foundation and a roadmap for future work toward reliable, memory-centric multi-agent intelligence.

References

P. Anokhin, N. Semenov, A. Sorokin, D. Evseev, A. Kravchenko, M. Burtsev, and E. Burnaev. Arigraph: Learning knowledge graph world models with episodic memory for llm agents. *arXiv preprint arXiv:2407.04363*, 2024.

U. Anwar, A. Saparov, J. Rando, D. Paleka, M. Turpin, P. Hase, E. S. Lubana, E. Jenner, S. Casper, O. Sourbut, et al. Foundational challenges in assuring alignment and safety of large language models. *arXiv preprint arXiv:2404.09932*, 2024.

J. M. Arana, K. A. M. Carandang, E. R. Casin, C. Alis, D. S. Tan, E. F. Legara, and C. Monterola. Foundations of PEERS: Assessing LLM role performance in educational simulations. In J. Zhao, M. Wang, and Z. Liu, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 908–918, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-254-1. doi: 10.18653/v1/2025.acl-srw.66. URL https://aclanthology.org/2025.acl-srw.66/.

D. A. Boiko, R. MacKnight, B. Kline, and G. Gomes. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578, 2023.

A. M. Bran, S. Cox, O. Schilter, C. Baldassari, A. D. White, and P. Schwaller. Chemcrow: Augmenting large-language models with chemistry tools. *arXiv preprint arXiv:2304.05376*, 2023.

S. M. Carlson, M. A. Koenig, and M. B. Harms. Theory of mind. *Wiley Interdisciplinary Reviews: Cognitive Science*, 4(4):391–402, 2013.

M. Cemri, M. Z. Pan, S. Yang, L. A. Agrawal, B. Chopra, R. Tiwari, K. Keutzer, A. Parameswaran, D. Klein, K. Ramchandran, et al. Why do multi-agent llm systems fail? *arXiv preprint arXiv:2503.13657*, 2025.

C. H. Chen and M. F. Shiu. Agentflow: Resilient adaptive cloud-edge framework for multi-agent coordination. *arXiv preprint arXiv:2505.07603*, 2025.

J. Chen, C. Yu, X. Zhou, T. Xu, Y. Mu, M. Hu, W. Shao, Y. Wang, G. Li, and L. Shao. Emos: Embodiment-aware heterogeneous multi-robot operating system with llm agents. *arXiv preprint arXiv:2410.22662*, 2024a.

S. Chen, Y. Liu, W. Han, W. Zhang, and T. Liu. A survey on llm-based multi-agent system: Recent advances and new frontiers in application. *arXiv preprint arXiv:2412.17481*, 2024b.

Y. Chen, J. Liu, H. Zhao, and H. Su. Social structure emergence: A multi-agent reinforcement learning framework for relationship building. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1807–1809, 2020.

G. Clarkson, T. E. Jacobsen, and A. L. Batcheller. Information asymmetry and information sharing. *Government Information Quarterly*, 24(4):827–839, 2007.

F. L. Da Silva, M. E. Taylor, and A. H. R. Costa. Autonomously reusing knowledge in multiagent reinforcement learning. In *IJCAI*, pages 5487–5493, 2018.

G. Dai, W. Zhang, J. Li, S. Yang, S. Rao, A. Caetano, M. Sra, et al. Artificial leviathan: Exploring social evolution of llm agents through the lens of hobbesian social contract theory. *arXiv preprint arXiv:2406.14373*, 2024.

D. Deshpande, V. Gangal, H. Mehta, A. Kannappan, R. Qian, and P. Wang. Memtrack: Evaluating long-term memory and state tracking in multi-platform dynamic agent environments. *arXiv preprint arXiv:2510.01353*, 2025.

Y. Du, S. Li, A. Torralba, J. B. Tenenbaum, and I. Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*, 2023.

P. J. Dunham. Joint attention its origins and role. *Development*, 1995.

L. E. Erdogan, N. Lee, S. Kim, S. Moon, H. Furuta, G. Anumanchipalli, K. Keutzer, and A. Gholami. Plan-and-act: Improving planning of agents for long-horizon tasks. *arXiv preprint arXiv:2503.09572*, 2025.

Z. Feng, R. Xue, L. Yuan, Y. Yu, N. Ding, M. Liu, B. Gao, J. Sun, X. Zheng, and G. Wang. Multi-agent embodied ai: Advances and future directions. *arXiv preprint arXiv:2505.05108*, 2025.

A. Fourney, G. Bansal, H. Mozannar, C. Tan, E. Salinas, F. Niedtner, G. Proebsting, G. Bassman, J. Gerrits, J. Alber, et al. Magentic-one: A generalist multi-agent system for solving complex tasks. *arXiv preprint arXiv:2411.04468*, 2024.

H. Gao and Y. Zhang. Memory sharing for large language model based agents. *arXiv preprint arXiv:2404.09982*, 2024.

J. Gottweis, W.-H. Weng, A. Daryin, T. Tu, A. Palepu, P. Sirkovic, A. Myaskovsky, F. Weissenberger, K. Rong, R. Tanno, et al. Towards an ai co-scientist. *arXiv preprint arXiv:2502.18864*, 2025.

S. Gregor and I. Benbasat. Explanations from intelligent systems: Theoretical foundations and implications for practice. *MIS quarterly*, pages 497–530, 1999.

T. Guo, X. Chen, Y. Wang, R. Chang, S. Pei, N. V. Chawla, O. Wiest, and X. Zhang. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*, 2024.

J. Hadfield, B. Zhang, K. Lien, F. Scholz, J. Fox, and D. Ford. How we built our multi-agent research system, 2025. URL https://www.anthropic.com/engineering/multi-agent-research-system. Engineering article.

B. Han and S. Zhang. Exploring advanced llm multi-agent systems based on blackboard architecture. *arXiv preprint arXiv:2507.01701*, 2025.

D. Han, C. Couturier, D. M. Diaz, X. Zhang, V. Rühle, and S. Rajmohan. Legomem: Modular procedural memory for multi-agent llm systems for workflow automation. *arXiv preprint arXiv:2510.04851*, 2025.

S. Han, Q. Zhang, Y. Yao, W. Jin, and Z. Xu. Llm multi-agent systems: Challenges and open problems. *arXiv preprint arXiv:2402.03578*, 2024.

B. Hayes-Roth. A blackboard architecture for control. *Artificial intelligence*, 26(3):251–321, 1985.

J. He, C. Treude, and D. Lo. Llm-based multi-agent systems for software engineering: Literature review, vision, and the road ahead. *ACM Transactions on Software Engineering and Methodology*, 34 (5):1–30, 2025.

M. Hechter and K.-D. Opp. Social norms. 2001.

S. Heller, M. Ibrahim, D. A. Selby, and S. Vollmer. Finding common ground: Using large language models to detect agreement in multi-agent decision conferences. *arXiv preprint arXiv:2507.08440*, 2025.

D. Hendrycks, D. Song, C. Szegedy, H. Lee, Y. Gal, E. Brynjolfsson, S. Li, A. Zou, L. Levine, B. Han, et al. A definition of agi. *arXiv preprint arXiv:2510.18212*, 2025.

C. Hong and Q. He. Enhancing memory retrieval in generative agents through llm-trained cross attention networks. *Frontiers in Psychology*, 16:1591618, 2025.

S. Hong, M. Zhuge, J. Chen, X. Zheng, Y. Cheng, J. Wang, C. Zhang, Z. Wang, S. K. S. Yau, Z. Lin, et al. Metagpt: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*, 2023.

M. Hu, Y. Zhou, W. Fan, Y. Nie, B. Xia, T. Sun, Z. Ye, Z. Jin, Y. Li, Q. Chen, et al. Owl: Optimized workforce learning for general multi-agent assistance in real-world task automation. *arXiv preprint arXiv:2505.23885*, 2025a.

Y. Hu, Y. Wang, and J. McAuley. Evaluating memory in llm agents via incremental multi-turn interactions. *arXiv preprint arXiv:2507.05257*, 2025b.

Z. Hu, H. P. Chan, J. Li, and Y. Yin. Debate-to-write: A persona-driven multi-agent framework for diverse argument generation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 4689–4703, 2025c.

H. Huang, X. Shi, H. Lei, F. Hu, and Y. Cai. Protchat: An ai multi-agent for automated protein analysis leveraging gpt-4 and protein language model. *Journal of Chemical Information and Modeling*, 65 (1):62–70, 2024.

Y. Jiang, K. C. Black, G. Geng, D. Park, J. Zou, A. Y. Ng, and J. H. Chen. Medagentbench: A realistic virtual ehr environment to benchmark medical llm agents. *arXiv preprint arXiv:2501.14654*, 2025.

Y. Jing, K. Li, B. Liu, H. Fu, Q. Fu, J. Xing, and J. Cheng. An open-ended learning framework for opponent modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 23222–23230, 2025.

S. Jinxin, Z. Jiabao, W. Yilei, W. Xingjiao, L. Jiawen, and H. Liang. Cgmi: Configurable general multi-agent interaction framework. *arXiv preprint arXiv:2308.12503*, 2023.

S. Jonnakuti. Multi-agent systems in the cloud: Coordinating autonomous ai agents for enterprise processes. 2023.

N. Krishnan. Advancing multi-agent systems through model context protocol: Architecture, implementation, and applications. *arXiv preprint arXiv:2504.21030*, 2025.

H. Lee, M. Kim, H. Kim, H. Kim, and H.-J. Lee. Integration and boost of a read-modify-write module in phase change memory system. *IEEE Transactions on Computers*, 68(12):1772–1784, 2019.

M. Lee, S. An, and M.-S. Kim. Planrag: A plan-then-retrieval augmented generation for generative large language models as decision makers. *arXiv preprint arXiv:2406.12430*, 2024.

B. Lei, Y. Zhang, S. Zuo, A. Payani, and C. Ding. Macm: Utilizing a multi-agent system for condition mining in solving complex mathematical problems. *Advances in Neural Information Processing Systems*, 37:53418–53437, 2024.

G. Li, H. Hammoud, H. Itani, D. Khizbullin, and B. Ghanem. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36: 51991–52008, 2023.

R. Li, X. Wang, D. Berlowitz, J. Mez, H. Lin, and H. Yu. Care-ad: a multi-agent large language model framework for alzheimer's disease prediction using longitudinal clinical notes. *npj Digital Medicine*, 8(1):541, 2025.

Y. Li, K. P. Tee, W. L. Chan, R. Yan, Y. Chua, and D. K. Limbu. Continuous role adaptation for human–robot shared control. *IEEE Transactions on Robotics*, 31(3):672–681, 2015.

Y. Li, H. Wen, W. Wang, X. Li, Y. Yuan, G. Liu, J. Liu, W. Xu, X. Wang, Y. Sun, et al. Personal llm agents: Insights and survey about the capability, efficiency and security. *arXiv preprint arXiv:2401.05459*, 2024.

T. Liang, Z. He, W. Jiao, X. Wang, Y. Wang, R. Wang, Y. Yang, S. Shi, and Z. Tu. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pages 17889–17904, 2024a.

T. Liang, Z. He, W. Jiao, X. Wang, Y. Wang, R. Wang, Y. Yang, S. Shi, and Z. Tu. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pages 17889–17904, 2024b.

J. Light, M. Cai, S. Shen, and Z. Hu. Avalonbench: Evaluating llms playing the game of avalon. *arXiv preprint arXiv:2310.05036*, 2023.

H. Liu, X. Zhang, H. Xu, Y. Wanyan, J. Wang, M. Yan, J. Zhang, C. Yuan, C. Xu, W. Hu, et al. Pc-agent: A hierarchical multi-agent collaboration framework for complex task automation on pc. *arXiv preprint arXiv:2502.14282*, 2025a.

J. Liu, Z. Kong, C. Yang, F. Yang, T. Li, P. Dong, J. Nanjekye, H. Tang, G. Yuan, W. Niu, et al. Rcr-router: Efficient role-aware context routing for multi-agent llm systems with structured memory. *arXiv preprint arXiv:2508.04903*, 2025b.

J. Luo, W. Zhang, Y. Yuan, Y. Zhao, J. Yang, Y. Gu, B. Wu, B. Chen, Z. Qiao, Q. Long, et al. Large language model agent: A survey on methodology, applications and challenges. *arXiv preprint arXiv:2503.21460*, 2025.

A. Maharana, D.-H. Lee, S. Tulyakov, M. Bansal, F. Barbieri, and Y. Fang. Evaluating very long-term conversational memory of llm agents. *arXiv preprint arXiv:2402.17753*, 2024.

P. Malinovskiy. Advanced game-theoretic frameworks for multi-agent ai challenges: A 2025 outlook. *arXiv preprint arXiv:2506.17348*, 2025.

Z. Mandi, S. Jain, and S. Song. Roco: Dialectic multi-robot collaboration with large language models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 286–299. IEEE, 2024.

T. Masterman, S. Besen, M. Sawtell, and A. Chao. The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey. *arXiv preprint arXiv:2404.11584*, 2024.

G. Mialon, R. Dessì, M. Lomeli, C. Nalmpantis, R. Pasunuru, R. Raileanu, B. Rozière, T. Schick, J. Dwivedi-Yu, A. Celikyilmaz, et al. Augmented language models: a survey. *arXiv preprint arXiv:2302.07842*, 2023.

P. Moreno, E. Hughes, K. R. McKee, B. A. Pires, and T. Weber. Neural recursive belief states in multi-agent reinforcement learning. *arXiv preprint arXiv:2102.02274*, 2021.

H. P. Nii. Blackboard systems, knowledge systems laboratory report no. ksl 86-18. *Knowledge Systems Laboratory, Department of Medical and Computer Science, Stanford University*, 1986.

Z. A. Pardos, M. Tang, I. Anastasopoulos, S. K. Sheel, and E. Zhang. Oatutor: An open-source adaptive tutoring system and curated content library for learning sciences research. In *Proceedings of the 2023 chi conference on human factors in computing systems*, pages 1–17, 2023.

J. S. Park, J. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22, 2023.

J. Piao, Y. Yan, J. Zhang, N. Li, J. Yan, X. Lan, Z. Lu, Z. Zheng, J. Y. Wang, D. Zhou, et al. Agentsociety: Large-scale simulation of llm-driven generative agents advances understanding of human behaviors and society. *arXiv preprint arXiv:2502.08691*, 2025.

C. Qian, W. Liu, H. Liu, N. Chen, Y. Dang, J. Li, C. Yang, W. Chen, Y. Su, X. Cong, et al. Chatdev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15174–15186, 2024a.

C. Qian, Z. Xie, Y. Wang, W. Liu, K. Zhu, H. Xia, Y. Dang, Z. Du, W. Chen, C. Yang, et al. Scaling large language model-based multi-agent collaboration. *arXiv preprint arXiv:2406.07155*, 2024b.

D. Ramani. A short survey on memory based reinforcement learning. *arXiv preprint arXiv:1904.06736*, 2019.

S. Ren, P. Jian, Z. Ren, C. Leng, C. Xie, and J. Zhang. Towards scientific intelligence: A survey of llm-based scientific agents. *arXiv preprint arXiv:2503.24047*, 2025.

A. Rezazadeh, Z. Li, A. Lou, Y. Zhao, W. Wei, and Y. Bao. Collaborative memory: Multi-user memory sharing in llm agents with dynamic access control. *arXiv preprint arXiv:2505.18279*, 2025.

S. Rupprecht, Q. Gao, T. Karia, and A. M. Schweidtmann. Multi-agent systems for chemical engineering: A review and perspective. *arXiv preprint arXiv:2508.07880*, 2025.

P. M. Salmon, N. A. Stanton, G. H. Walker, D. P. Jenkins, and L. Rafferty. Is it really better to share? distributed situation awareness and its implications for collaborative system design. *Theoretical Issues in Ergonomics Science*, 11(1-2):58–83, 2010.

H. Salwa, N. Burhan, and E. Rahel. Continual learning: Overcoming catastrophic forgetting for adaptive ai systems. *Authorea Preprints*, 2025.

R. SHENG. A survey of llm-based multi-agent systems in medicine. *Authorea Preprints*, 2025.

H. Singh, R. J. Das, M. Han, P. Nakov, and I. Laptev. Malmm: Multi-agent large language models for zero-shot robotics manipulation. *arXiv preprint arXiv:2411.17636*, 2024.

R. N. Spreng and R. A. Mar. I remember you: a role for memory in social cognition and the functional neuroanatomy of their interaction. *Brain research*, 1428:43–50, 2012.

R. Stalnaker. Common ground. *Linguistics and philosophy*, 25(5/6):701–721, 2002.

H. Su, R. Chen, S. Tang, Z. Yin, X. Zheng, J. Li, B. Qi, Q. Wu, H. Li, W. Ouyang, P. Torr, B. Zhou, and N. Dong. Many heads are better than one: Improved scientific idea generation by a llm-based multi-agent system, 2025. URL https://arxiv.org/abs/2410.09403.

H. Sun, S. Zhang, L. Niu, L. Ren, H. Xu, H. Fu, F. Zhao, C. Yuan, and X. Wang. Collab-overcooked: Benchmarking and evaluating large language models as collaborative agents. *arXiv preprint arXiv:2502.20073*, 2025.

J. Sun, C. Dai, Z. Luo, Y. Chang, and Y. Li. Lawluo: A multi-agent collaborative framework for multi-round chinese legal consultation. *arXiv preprint arXiv:2407.16252*, 2024.

R. Sun. *Cognition and multi-agent interaction: From cognitive modeling to social simulation*. Cambridge University Press, 2006.

H. Tan, Z. Zhang, C. Ma, X. Chen, Q. Dai, and Z. Dong. Membench: Towards more comprehensive evaluation on the memory of llm-based agents. *arXiv preprint arXiv:2506.21605*, 2025.

X. Tang, A. Zou, Z. Zhang, Z. Li, Y. Zhao, X. Zhang, A. Cohan, and M. Gerstein. MedAgents: Large language models as collaborators for zero-shot medical reasoning. In L.-W. Ku, A. Martins, and V. Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 599–621, Bangkok, Thailand, Aug. 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.33. URL https://aclanthology.org/2024.findings-acl.33/.

P. Tarale, E. Rietman, and H. T. Siegelmann. Distributed multi-agent lifelong learning. *Transactions on Machine Learning Research*, 2025.

K.-T. Tran, D. Dao, M.-D. Nguyen, Q.-V. Pham, B. O'Sullivan, and H. D. Nguyen. Multi-agent collaboration mechanisms: A survey of llms. *arXiv preprint arXiv:2501.06322*, 2025.

H. Trivedi, T. Khot, M. Hartmann, R. Manku, V. Dong, E. Li, S. Gupta, A. Sabharwal, and N. Balasubramanian. Appworld: A controllable world of apps and people for benchmarking interactive coding agents. *arXiv preprint arXiv:2407.18901*, 2024.

A. R. Wagner. Sop: A model of automatic memory processing in animal behavior. In *Information processing in animals*, pages 5–47. Psychology Press, 2014.

Z. Wan, Y. Li, X. Wen, Y. Song, H. Wang, L. Yang, M. Schmidt, J. Wang, W. Zhang, S. Hu, et al. Rema: Learning to meta-think for llms with multi-agent reinforcement learning. *arXiv preprint arXiv:2503.09501*, 2025.

G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Y. Wang and X. Chen. Mirix: Multi-agent memory system for llm-based agents. *arXiv preprint arXiv:2507.07957*, 2025.

H. Wei, J. Qiu, H. Yu, and W. Yuan. Medco: Medical education copilots based on a multi-agent framework. In *European Conference on Computer Vision*, pages 119–135. Springer, 2024.

D. Wu, H. Wang, W. Yu, Y. Zhang, K.-W. Chang, and D. Yu. Longmemeval: Benchmarking chat assistants on long-term interactive memory. *arXiv preprint arXiv:2410.10813*, 2024a.

D. Wu, X. Wei, G. Chen, H. Shen, X. Wang, W. Li, and B. Jin. Generative multi-agent collaboration in embodied ai: A systematic review. *arXiv preprint arXiv:2502.11518*, 2025a.

Q. Wu, G. Bansal, J. Zhang, Y. Wu, B. Li, E. Zhu, L. Jiang, X. Zhang, S. Zhang, J. Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024b.

Y. Wu, S. Liang, C. Zhang, Y. Wang, Y. Zhang, H. Guo, R. Tang, and Y. Liu. From human memory to ai memory: A survey on memory mechanisms in the era of llms. *arXiv preprint arXiv:2504.15965*, 2025b.

S. Xu, X. Zhang, and L. Qin. Eduagent: Generative student agents in learning. *arXiv preprint arXiv:2404.07963*, 2024.

S. Xu, H.-N. Wen, H. Pan, D. Dominguez, D. Hu, and X. Zhang. Classroom simulacra: Building contextual student generative agents in online education for learning behavioral simulation. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–26, 2025.

B. Yan, Z. Zhou, L. Zhang, L. Zhang, Z. Zhou, D. Miao, Z. Li, C. Li, and X. Zhang. Beyond self-talk: A communication-centric survey of llm-based multi-agent systems. *arXiv preprint arXiv:2502.14321*, 2025.

Y. Yang, H. Chai, S. Shao, Y. Song, S. Qi, R. Rui, and W. Zhang. Agentnet: Decentralized evolutionary coordination for llm-based multi-agent systems. *arXiv preprint arXiv:2504.00587*, 2025.

A. H. Yoo and A. G. Collins. How working memory and reinforcement learning are intertwined: A cognitive, neural, and computational perspective. *Journal of cognitive neuroscience*, 34(4):551–568, 2022.

C. Yu, Z. Cheng, H. Cui, Y. Gao, Z. Luo, Y. Wang, H. Zheng, and Y. Zhao. A survey on agent workflow–status and future. In *2025 8th International Conference on Artificial Intelligence and Big Data (ICAIBD)*, pages 770–781. IEEE, 2025.

C. Zhang, K. Yang, S. Hu, Z. Wang, G. Li, Y. Sun, C. Zhang, Z. Zhang, A. Liu, S.-C. Zhu, et al. Proagent: building proactive cooperative agents with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17591–17599, 2024a.

G. Zhang, M. Fu, G. Wan, M. Yu, K. Wang, and S. Yan. G-memory: Tracing hierarchical memory for multi-agent systems. *arXiv preprint arXiv:2506.07398*, 2025a.

G. Zhang, L. Niu, J. Fang, K. Wang, L. Bai, and X. Wang. Multi-agent architecture search via agentic supernet. *arXiv preprint arXiv:2502.04180*, 2025b.

W. Zhang, C. Cui, Y. Zhao, R. Hu, Y. Liu, Y. Zhou, and B. An. Agentorchestra: A hierarchical multi-agent framework for general-purpose task solving. *arXiv e-prints*, pages arXiv–2506, 2025c.

Y. Zhang, R. Sun, Y. Chen, T. Pfister, R. Zhang, and S. Arik. Chain of agents: Large language models collaborating on long-context tasks. *Advances in Neural Information Processing Systems*, 37: 132208–132237, 2024b.

Z. Zhang, Q. Dai, X. Bo, C. Ma, R. Li, X. Chen, J. Zhu, Z. Dong, and J.-R. Wen. A survey on the memory mechanism of large language model-based agents. *ACM Transactions on Information Systems*, 43 (6):1–47, 2025d.

J. Zheng, C. Shi, X. Cai, Q. Li, D. Zhang, C. Li, D. Yu, and Q. Ma. Lifelong learning of large language model based agents: A roadmap. *arXiv preprint arXiv:2501.07278*, 2025.

H. Zhou, X. Wan, R. Sun, H. Palangi, S. Iqbal, I. Vulić, A. Korhonen, and S. Ö. Arık. Multi-agent design: Optimizing agents with better prompts and topologies. *arXiv preprint arXiv:2502.02533*, 2025.

D. Zhu, R. Meng, J. Chen, S. Li, T. Pfister, and J. Yoon. Doclens: A tool-augmented multi-agent framework for long visual document understanding. *arXiv preprint arXiv:2511.11552*, 2025a.

K. Zhu, H. Du, Z. Hong, X. Yang, S. Guo, D. Z. Wang, Z. Wang, C. Qian, R. Tang, H. Ji, et al. Multiagentbench: Evaluating the collaboration and competition of llm agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8580–8622, 2025b.