Proceedings of CONF-CDS 2026 Symposium: Data-Centric AI Security: Securing Models, Learning Agents, and Autonomous Systems
DOI: 10.54254/2755-2721/2026.CH34641

A Survey of Large Language Models in Multi-Agent Systems: Advancing Coordination and Intelligence

Zhongzhi Gong

College of Life and Environmental Sciences,Wenzhou University, Wenzhou, China
yuquansong3@gmail.com

Abstract. Multi-agent systems(MASs) are more and more significant for solving complex real problems that demanded coordinated decision and interaction in dynamic environments. However, traditional ways, especially in reinforcement learning(RL), are not competent in the situation, including language grounding, multimodal perception, long-horizon planning, and cross-domain generalization. In contrast, large language models (LLMs) show a stronger and promising potential by equipping agents with strong reasoning, planning, and communication capabilities in its distinctive paradigm shift. In this survey,we will comprehensively demonstrate emerging LLM-based multi-agent systems in four key aspects:(1) communication and interaction, which governs how agents swaps information with coordinated reasoning;(2) system architecture and coordination, defining the way to assign and organise complex tasks;(3) evaluation and benchmarking, which is used to measure agents'abilities in different environments;(4) learning, evolution, characterizing how agents improve, adapt and generalize. Through these advanced procedure of refereed dimensions, we reveal how LLMs reshape MASs' designing space, which can realize the more flexible, extensible and human-aligned coordination strategies. Finally, we discuss open challenges including efficiency, reliability and standardized evaluation, and outline future research directions toward robust and autonomous multi-agent intelligence.

Keywords: Large language system, Multi-agent system, Agent communication

1. Introduction

LLMs are a significant force now in artificial intelligence (AI). They can do way more than just generate text. Actually at their core, these models are proficient at reasoning and planning.For example, they break down hard tasks, anticipate future states, and put together a sensible sequence of actions. Take the work of Wei et al., who came up with chain-of-thought prompting, and showed that if you reason step by step, you get much better results on arithmetic, commonsense, and symbolic tasks [1] . Then Yao et al. built on that and proposed ReAct—basically mixing reasoning with action, so models can adjust their plans on the fly while interacting with external environments [2] .

Modern large language models show great competence in dealing with human language. They can figure out what people imply in conversations, get rid of semantic confusion, and carry out multi-turn interactions smoothly—developing these functions is never an easy task. Apart from

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2026.CH34641

language skills, knowledge grounding and dynamic adaptation are where LLMs shine the most. Mass-scale pre-training lets them master general rules for cross-domain use, and in-context learning plus self-reflection further improves their inner operation. Plaat's research team classified core model functions into reasoning, acting and interaction not long ago. Their findings show that reflection, knowledge retrieval and tool use work together to make models more autonomous [3]. All in all, today's LLMs are far more than text handlers; they operate as full-scale reasoning systems to realize various intelligent behaviors in open environments.

Given how well LLMs work alone, it's natural to try them in MAS. In that setting, multiple agents have to coordinate to solve tasks that no single agent could handle. If you give each agent an LLM-based brain, then MASs can move away from rigid, pre-set rules toward more flexible and spontaneous collaboration. Agents can communicate in natural language, share bits of what they see to build a shared picture of the environment, and make joint plans over long time horizons. Several recent surveys say this area is growing really fast [4, 5] . Also, modern LLMs can handle multiple modalities—text, vision, audio—so agents can ground their interactions in rich sensory input. That's related to work on using LLMs in robotic autonomy and embodied AI [6, 7].

But moving from single-agent intelligence to multi-agent teamwork brings new challenges. For example, errors like hallucinations can spread from one agent to another [8, 9]; communication becomes both a coordination mechanism and a vulnerability [10, 11]; and the system-level behavior emerges from the complex interaction of several imperfect parts [12]. So we really need a systematic understanding of how LLMs change the design and evaluation of multi-agent systems.

This survey, then—we're taking the fast-growing literature on LLM-based MAS and sorting it into four buckets. First bucket: Communication. Second: System Architecture and Coordination. Third: Evaluation and Benchmarking. Fourth: Learning, Evolution, and Robustness.Four lenses, one unified view. That's the aim—to map out the design space for LLM-based multi-agent systems, as it actually stands.We will spotlight two things. One: the substantial potential of LLM-driven coordination. Two: the critical limits that still need fixing. And then we will lay out a roadmap—pointing toward more scalable, more reliable, truly autonomous multi-agent intelligence.

2. Related work

Recent advances in LLM-based multi-agent systems (LLM-MAS) have spawned a rapidly expanding body of work—across quite a few research directions, we think. To give a structured overview, we organize the existing literature into four key aspects. First, communication and interaction. This one studies how agents exchange information and coordinate their reasoning. Second, system architecture and coordination—here the focus is on task decomposition and how you design the organization (importantly, not just roles but also workflows). Third, evaluation and benchmarking. The aim: to develop environments and metrics that can actually assess multi-agent capabilities. And fourth, learning, evolution, and robustness. This looks at how agents get better over time, and stay reliable when things are uncertain. While these four aspects are different in concept, they are also overlapping in practical applications and there are much research entailing various directions.Therefore, we classify these works through their key contribution. A summary of relevant work data is presented in Table 1.

DOI: 10.54254/2755-2721/2026.CH34641

Table 1. Details of multi-agent systems referred to in related work

Motivation	Reasearch Domain	Goals	Agent Number	Cooperation or Competition	Strength	Limitation
Communication and Interaction	CAMEL [13] Multi-agent communication	Role-playing agents for task completion	3(User, Assistant, Task Specifier)	Cooperation	Role-play, task decomposition	No tool execution, prone to deadlock
	AutoGen [14] LLM application framework	Enabling multi-agent conversations	2+ (e.g., Assistant + UserProxy + optional)	Both	Flexible, modular, tool execution	Complex, risk of lossing control
	LLM Harmony [15] Multi-agent problem solving	Role-play for reasoning tasks	2 (Expert + Evaluator)	Cooperation	Chain-of-thought, simple	Relies on base model, context limited
	ChatEval [16] LLM evaluation	Multi-agent debate for assessment	Multiple	Cooperative (with adversarial discussion)	Diverse roles, communication strategies, human-aligned	Multi-round, depends on base LLM
	network topology [17] Collective intelligence	Network topology effects on debate	25 (configurable network size)	Cooperative	Random networks save tokens, consensus measures uncertainty	Biased hubs affect performance, context limited
	Multi-llm debate [19] Theoretical analysis of debate	Mathematical framework and interventions	Multiple	Both	Theoretical framework, effective interventions	Requires embeddings, less effective for arithmetic
	MetaGPT [20] Multi-agent collaboration	Meta-programming with SOPs	5 (Product Manager, Architect, Project Manager, Engineer, QA Engineer)	Cooperative	SOP, structured communication, executable feedback	Fixed hierarchy, lacks flexibility
System Architecture and Coordination	Drop the Hierarchy and Roles [21] Self-organizing LLM agentsd	Outperform designed hierarchies	4+	Cooperative	Self-organization, no predefined roles, scales well	Sequential protocol high latency, requires strong models
	CausalPlan [22] Causality-driven planning	Efficient multi-agent collaboration	2(LLM agent + partner)	Cooperative	Causal reasoning reduces invalid actions	High compute overhead, relies on predefined features
Evaluation and Benchmarking	Mindagent [23] Emergent gaming interaction	Study emergent behavior in games	2 – 4 (configurable, centralized dispatch)	Cooperative (also human-AI collaboration)	Multi-agent coordination, human-in-the-loop, zero-shot planning, transferable to Minecraft	Weaker models perform poorly, context length limits long-horizon planning
	Discovery to Application Loop [24] LLM discovery–application loop	Assess loop closure capability	2 (Engineer + optional Scientist)	Cooperative	Discovery-application loop, knowledge consolidation	Low success rate (~25%), knowledge gap identification bottleneck

DOI: 10.54254/2755-2721/2026.CH34641

Table 1. (continued)

Evaluation and Benchmarking	Lemma [25] Language-conditioned multi-robot manipulation	Learn manipulation from language	2(two robots, centralized or decentralized)	Cooperative	Multi-robot manipulation, tool use, task allocation	Long-horizon difficulty, language understanding
	Teamcraft [26] Multi-modal multi-agent benchmark	Benchmark Minecraft agents	2 – 4	Cooperative	Multi-modal, large demonstrations, generalization splits	VLA models underperform, centralized beats decentralized
	CORAL [27] Autonomous multi-agent evolution	Open-ended discovery	1 – 4 (configurable, multi-agent coevolution)	Cooperative	Autonomous evolution, shared persistent memory, heartbeat reflection, multiagent exploration	Depends on strong models, may plateau in local optima
Learning, Evolution, and Robustness	EvoSkills [28] Co-evolutionary skill verification	Self-evolving agent skills	2 (Skill Generator + Surrogate Verifier)	Cooperative (coevolutionary)	Generates structured multifile skills, outperforms humancurated skills, modeltransferable	Relies on LLM capability, surrogate verifier may misalign with oracle
	LLM-TOC [29] Theory-of-mind adversarial curriculum	Multi-agent generalization	Multiple (student MARL agent + LLMgenerated adversary population)	Both	Gradient saliency feedback, high sample efficiency, strong zeroshot generalization	Depends on proprietary LLMs, requires environment with injectable policies

2.1. Communication and interaction

A significant body of research has examined how LLM agents learn to interact and cooperate using natural language. One early representative, CAMEL, works by fixing agent roles and letting those agents talk under shared dialogue rules [13]. Dedicated initial prompts plus explicit termination conditions are what keep multi-turn interactions stable. That same design prevents two common issues: role drift and infinite loops. Another thing CAMEL shows is that generating simulated multi-agent interaction data can support scaling. Then AutoGen takes a different approach [14]. It turns agent communication into a programmable working mode. Developers can use unified interfaces to set up various interaction types—group chats, tool-augmented agents, and human-in-the-loop collaboration. This makes it easier to build complex multi-agent task execution frameworks. Further evidence comes from LLM Harmony [15]. Even a simple team of two agents, one expert and one evaluator, can achieve better reasoning results through repeated feedback. The advantage of specialized role design is clearly visible from this result.

Another study direction focuses on multi-agent debate and structured interaction mechanisms. For example, ChatEval introduces a similar framework which allows agents take on different roles and utilise various communication maneuvers to evaluate generated content, which improves the consistency between the evaluating result and human judgments [16]. Regan and colleagues investigated how communication structures influence system performance. Surprisingly, sparse and random communication schemes perform equally well as fully connected ones, yet with much less

DOI: 10.54254/2755-2721/2026.CH34641

resource consumption [17]. Then there are works that step back and offer analytical or theoretical views on communication. Guo et al., for instance, sort communication paradigms into three buckets —cooperation, debate, competition—and discuss their roles in problem solving and simulation [18]. A communication-centric survey (Anonymous, 2026) breaks interaction down further: system-level protocols on one hand, intra-agent strategies on the other. From a theoretical angle, Estornell and Liu model multi-agent debate using Bayesian inference. They identify specific failure modes—echo chambers, shared misconceptions—and propose interventions like diversity and quality pruning [19].

2.2. System architecture and coordination

Beyond communication, another obvious challenge: how to structure multi-agent systems when they face complex tasks. MetaGPT makes a seminal effort here. It introduces standardized operating procedures (SOPs)—inspired by software engineering, that is. The framework assigns agents to specialized roles (product manager, architect, engineer) and enforces structured outputs, especially in code generation. Then subsequent work keeps exploring the design space of coordination protocols. Consider Dochkina's large-scale empirical study. After thousands of runs, she finds what she calls the "endogeneity paradox": hybrid coordination schemes—fixed structure plus partial autonom—actually outperform both fully centralized and fully decentralized approaches [21]. Importantly, this finding undercuts the usual assumption that more autonomy always means better performance. It also drives home the point: careful protocol design really matters.

Another direction—causal reasoning integrated into coordination—is now drawing increasing notice. CausalPlan targets one recurring problem: LLM agents tend to produce actions that lack causal validity. Its solution? Learn structural causal models from interaction trajectories, then reweight or filter the generated actions using a causal action matrix [22]. That, in turn, improves decision consistency. And interestingly, it outperforms reinforcement learning baselines—without any extra fine-tuning. Overall, these studies show that structuring a effective MAS requires powerful agents as well as principled framework and coordination.

2.3. Evaluation and benchmarking

As LLM-based MAS become more complex,evaluating their ability also becomes more difficult. MindAgent structures a CuisineWorld benchmark based on dynamic environment to assess multi-agent collaboration. Simultaneously, MindAgent proposes Collaboration Efficiency Score (CoS) to quantify coordination quality and supporting fair comparisons among diverse agent setups. Rather than focusing merely on the result of task execution, SciCrafter (Zhou et al., 2026) conducted evaluation on the transitionfrom exploring knowledge to practicl application.It divides agent abilities in four stages: knowledge gap identification, conducting experimental exploration, consolidating acquired knowledge and deploying knowledge for application. This evaluation system can locate current frameworks' defects. And knowledge gap identification becomes a critical bottlenecke to xisting systems. LEMM—it targets language-conditioned multi-robot manipulation, specifically in tabletop settings. The benchmark gives you eight procedurally generated task types, each with strong temporal dependencies, plus 6,400 expert demonstrations and language instructions [25]. It actually solves three things at once: multi-agent coordination, task allocation, and language grounding. And it throws in a modular hierarchical planning baseline as well.Later, TeamCraft (2024) extends benchmarking to multi-modal multi-agent environments. It combines cross-modal

DOI: 10.54254/2755-2721/2026.CH34641

perception and interaction so that it can reflect the complexity of real-world scenarios successfully [26].

2.4. Learning, evolution, and robustness

CORAL, one recent study, looks at how multi-agent systems can get better over time and still hold up in complex environments. The authors propose an asynchronous evolutionary framework. Agents iteratively refine solutions by sharing persistent memory—past attempts and learned skills—and that design supports open-ended discovery [27]. Compared to static search methods, it performs clearly better, though it also brings new issues like redundancy and coordination overhead. EvoSkills, for its part, concentrates on skill acquisition. It co-evolves a skill generator together with a verifier, so reusable skill modules can be generated automatically [28]. The framework does show strong performance plus transferability. But at the same time, it comes with a high computational price, and it's hard to verify things reliably when you lack ground-truth signals. Meanwhile, robustness and security have become serious worries for the field. LLM-TOC sees multi-agent generalization as a two-level optimization task. In that view, LLMs produce adversarial policies that reveal where the system is weak [29]. Gradient saliency feedback is used here to look for problems in how agents behave. Once those weak points are found, we can adjust the system to make it more resistant to errors. That said, this method isn't perfect. It relies a lot on very powerful proprietary models, which is definitely a drawback in practice. Taking a broader view, these studies show that pushing for ongoing improvement has to go together with stability, safety, and the ability to resist adversarial attacks—none of which can be ignored.

3. Methodological advances

Large language models, when integrated into multi-agent systems, are changing the old paradigms in a fundamental way—and they open up design possibilities that simply did not exist before. This section looks at recent methodological advances. We focus on four things. One is inter-agent communication and interaction. System design and coordination—that's another part. Then we need good ways to test performance and compare results—that is evaluation and benchmarking. The last piece is adaptive learning, evolution, and staying robust over time. We do not treat these as separate topics. Instead, we highlight the tricky trade-offs, the main limitations of current methods, and the new design principles that are starting to emerge—all of which matter for the field.

3.1. Structured communication vs. emergent interaction

A central methodological question in LLM-MAS is whether agent communication should be explicitly structured or allowed to emerge through free-form interaction.

Structured communication frameworks, such as CAMEL and AutoGen, impose predefined roles, dialogue formats, and termination conditions [13, 14]. These designs improve stability, interpretability, and controllability, preventing issues such as role drift, infinite loops, and incoherent reasoning. Similarly, debate-based methods like ChatEval introduce explicit communication protocols (e.g., turn-taking, summarization), which help guide agents toward consensus [16]. However, structured approaches often limit flexibility and scalability. They require manual prompt engineering, fixed communication patterns, and task-specific design, making them brittle in open-ended environments. In contrast, more emergent interaction paradigms, such as those explored in LLM Harmony or decentralized communication settings (e.g., Regan et al., 2024), allow agents to

DOI: 10.54254/2755-2721/2026.CH34641

dynamically adapt their communication strategies [15, 17]. These methods better capture the fluidity of real-world collaboration and can generalize across tasks without extensive redesign.

The trade-off lies between control and adaptability. Structured protocols provide reliability and ease of debugging, while emergent communication enables flexibility but introduces risks of misalignment, ambiguity, and error propagation. A key open direction is developing adaptive communication protocols that can dynamically adjust structure based on task complexity and agent capability.

3.2. Predefined workflows vs. dynamic coordination

Another major methodological axis concerns how multi-agent systems are organized to solve complex tasks, ranging from rigid workflows to fully autonomous coordination.

Workflow-driven systems such as MetaGPT adopt predefined pipelines inspired by human organizations, where agents assume specialized roles and follow standardized operating procedures [20]. This approach significantly improves task decomposition, modularity, and reproducibility, particularly for structured tasks like software development. It also reduces coordination overhead by constraining agent behavior within well-defined boundaries. In contrast, recent work highlights the benefits of dynamic coordination mechanisms. For instance, Dochkina shows that hybrid protocols —combining fixed ordering with partial autonomy—outperform both fully centralized and fully decentralized systems [21]. Similarly, CausalPlan introduces causal reasoning to guide action selection, enabling agents to dynamically adapt decisions based on inferred dependencies rather than fixed workflows [22].

The key limitation of predefined workflows is their lack of adaptability: they struggle in environments with high uncertainty or evolving task structures. On the other hand, dynamic coordination introduces challenges in consistency, convergence, and coordination cost, as agents may generate conflicting plans or redundant actions.

This tension suggests that effective MAS design should balance structure and flexibility, potentially through hierarchical or hybrid architectures where high-level planning is structured, while low-level execution remains adaptive.

3.3. Static benchmarks vs. process-oriented evaluation

Evaluation methodologies for LLM-MAS can be broadly divided into static, outcome-based benchmarks and process-oriented, capability-driven evaluation.

Traditional benchmarks, such as MindAgent and TeamCraft, focus on task completion metrics within predefined environments [23, 26]. These approaches enable standardized comparison and reproducibility, making them essential for tracking progress. However, they often reduce complex multi-agent behavior to single scalar metrics, failing to capture intermediate reasoning steps, coordination dynamics, and failure modes.

In contrast, process-oriented frameworks like SciCrafter decompose agent performance into multiple stages (e.g., knowledge discovery, consolidation, application), offering a more fine-grained understanding of system capabilities [24]. This paradigm shifts evaluation from "what outcome is achieved" to "how the outcome is achieved," revealing bottlenecks that are otherwise hidden in aggregate metrics. Nevertheless, process-based evaluation introduces higher complexity and subjectivity. Defining meaningful intermediate metrics is challenging, and the evaluation process itself can become computationally expensive and difficult to standardize. Moreover, the growing use of LLMs as evaluators raises concerns about bias, consistency, and alignment with human judgment.

DOI: 10.54254/2755-2721/2026.CH34641

The key challenge is to design evaluation frameworks that are both scalable and diagnostically informative, combining the reproducibility of static benchmarks with the insight of process-level analysis.

3.4. Isolated optimization vs. open-ended evolution

A defining shift in LLM-MAS is the move from static, task-specific optimization toward open-ended learning and evolution.

Traditional approaches treat each task independently, optimizing agent behavior within a fixed setting. While this paradigm ensures efficiency and stability, it limits the ability of agents to accumulate knowledge and generalize across tasks. In contrast, evolutionary frameworks such as CORAL and EvoSkills enable agents to iteratively refine solutions, share knowledge, and acquire reusable skills. These methods support continual improvement and transferability, bringing MAS closer to open-ended intelligence [27, 28]. However, open-ended evolution introduces significant challenges. First, it is often computationally expensive, requiring repeated interactions and evaluations. Second, coordination becomes more difficult as agents evolve asynchronously, leading to redundancy, conflicts, and instability. Third, the absence of reliable verification mechanisms can result in error accumulation, especially when surrogate evaluators are imperfect.

These issues are further compounded by concerns of robustness and security. As shown in LLM-TOC, adversarial strategies can exploit vulnerabilities in agent behavior, and the use of natural language as a control interface expands the attack surface [29]. Ensuring robustness in evolving systems requires not only better training mechanisms but also principled safeguards against adversarial manipulation and cascading failures.

4. Challenges and open problems

Existing research on large language model (LLM)-based multi-agent systems (LLM-MAS) has constructed a comprehensive research system covering benchmark construction, collaborative frameworks, task decomposition, interactive programming, autonomous evolution, and security evaluation. Dedicated benchmarks such as LEMMA and TeamCraft have characterized multi-agent collaboration from multiple dimensions, while frameworks including CAMEL, MetaGPT, AutoGen, and CORAL have explored agent collaboration, task execution, and iterative evolution [13, 14, 25, 27]. However, from the perspectives of agent capability, system collaboration, evaluation mechanism, practical deployment, and security robustness, current studies still face prominent bottlenecks and unresolved open problems, which are summarized as the following core challenges:

4.1. Fundamental bottlenecks in multi-agent collaboration efficiency and generalization

Multi-agent coordination and cross-scene generalization remain the most critical core challenges for LLM-MAS. First, collaboration efficiency and spatial reasoning capabilities are severely insufficient: TeamCraft experiments reveal that existing models perform poorly in spatial reasoning and coordinated interaction [26], and LEMMA only solves partial problems of task allocation and language grounding without fundamentally breaking through complex temporal dependency collaboration [13]. Second, cross-environment and distribution-shift generalization is fragile: Reinforcement Fine-Tuning (RFT) shows sharp performance degradation under observation space changes, action interface differences, or background knowledge mismatches [30]; LLM-TOC generates static robust policies without online adaptive capabilities during testing [29], and

DOI: 10.54254/2755-2721/2026.CH34641

AgentTuning exhibits significantly diminished effectiveness in ultra-complex scenarios such as Minecraft [31]. Third, collaboration scale and scalability are extremely limited: Classical frameworks like CAMEL and LLM Harmony only support fixed two-agent role-playing [13, 15], MindAgent is constrained to 2–4 agents, and none can adapt to large-scale agent populations or dynamic team composition [23]. In addition, existing surveys (e.g., Sun et al., Han et al.) only conceptualize collaboration paradigms and communication architectures but fail to propose practical algorithmic solutions for credit assignment, scalability, and coordination conflicts, and overemphasize communication while ignoring key modules such as agent memory and perception [32, 33].

4.2. Adaptation dilemmas in task decomposition, interaction paradigms and complex task execution

Decomposing complex tasks into executable workflows and supporting flexible agent interaction is still plagued by scenario limitations and poor scalability. On the one hand, task decomposition and standardized workflows lack cross-scene generalization: MetaGPT introduces software engineering-oriented SOPs to optimize code generation, but its mechanism is highly scenario-specific and cannot be extended to general complex tasks [20]; SciCrafter identifies knowledge gap identification as the primary bottleneck in the scientific discovery-to-application loop, with huge room for improvement in knowledge consolidation and experimental discovery, and the task system is limited to Minecraft redstone circuits with weak universality [24]. On the other hand, agent interaction programming relies on manual design and has narrow application boundaries: AutoGen supports flexible conversation patterns but requires developers to define workflows through natural language and programming languages, with high customization costs [14]; LLM Harmony manually crafts personas and prompts for specific tasks, with a fixed turn-based two-agent interaction mode that cannot handle dynamic complex tasks [15]. Meanwhile, AgentInstruct in AgentTuning has limited task diversity, restricting the breadth of learnable agentic behaviors and failing to support open-ended complex task execution [31].

4.3. Inherent defects in evaluation benchmarks, metrics and capability assessment

The evaluation system of LLM-MAS is incomplete, with unobjective metrics and narrow application scenarios, which cannot fully and accurately measure agent capabilities. First, specialized benchmarks have single-dimensional coverage: LEMMA and TeamCraft only target tabletop robot manipulation and Minecraft open worlds respectively, lacking unified evaluation standards for multi-modal002C cross-domain, and large-scale collaborative scenarios [13, 26]. Second, hallucination detection and traditional evaluation metrics are invalid: ROUGE, a widely used hallucination metric, has a weak correlation with human judgment, and the length-based heuristic only applies to short-form QA without semantic interpretability [34]; current evaluation practices have fundamental flaws without verification in long-form generation or dialogue scenarios. Third, LLM-as-judge systems have strong subjectivity and high deployment costs: The survey of LLM judges has subjective evaluation criteria, shallow exploration of multi-modal judges, and expensive multi-round voting strategies that are difficult to scale. In addition, most surveys (Li et al., Chen et al.) only classify architectures and applications taxonomically, lack empirical comparative analysis of method performance, and fail to form a quantifiable, comprehensive evaluation system [4, 35].

DOI: 10.54254/2755-2721/2026.CH34641

4.4. Accessibility problems of proprietary model dependence, data generation and open-source ecology

Many advanced LLM-MAS frameworks are restricted by proprietary models and high resource costs, hindering open research and large-scale deployment. First, over-reliance on closed-source LLMs leads to upper capability limits and poor accessibility: AgentTuning depends on GPT-4 for data generation [31], LLM-TOC requires proprietary LLMs for code generation and causal diagnosis [29], and CORAL relies on Claude Opus 4.6, resulting in high costs, uncontrollable model behavior, and inaccessibility for academic research [27]. Second, data generation and evolutionary training have high resource overheads: EvoSkills' skill co-evolution process is computationally expensive [28], and AgentTuning's relaxed filtering thresholds lead to potentially suboptimal trajectories. Third, multi-agent autonomous evolution has redundancy and conflict issues: CORAL suffers from redundant information and interactive conflicts in asynchronous evolution, and its heartbeat hyperparameters require manual tuning without adaptive adjustment mechanisms; EvoSkills can only learn task-specific skills without cross-task reuse, severely limiting the sustainability of agent capability improvement.

4.5. Security, privacy risks and hallucination vulnerabilities

Security, privacy, and factual reliability have become hidden dangers restricting the practical deployment of LLM-MAS, with no effective general defense mechanisms. First, LLM security and privacy research stays at conceptual classification: Existing surveys only categorize vulnerability detection, malicious abuse, and model weaknesses but do not propose new defense strategies, and emerging jailbreak attacks are only briefly mentioned with vague responses to hybrid security scenarios [36]. Second, system-level platform vulnerabilities have poor universality and lack automated detection tools: Research on LLM system vulnerabilities (e.g., cross-session file leakage, indirect prompt injection) is only specific to GPT-4 and cannot be generalized to other systems, with no automated vulnerability scanning tools [37]. Third, hallucination and truthfulness detection have severe application limitations: The method of using LLM internal states to judge truth requires white-box access to models and is invalid for black-box APIs; its template-based dataset cannot reflect the ambiguity of natural language [38], and there is a lack of robust detection mechanisms for long-text and multi-turn dialogue [34].

4.6. Shortcomings in multimodal fusion, perception and autonomous cognitive capabilities

Multi-modal integration and agent autonomous cognitive evolution are still in the preliminary stage, with obvious capability boundaries. On the one hand, multi-modal LLM support is limited and alignment efficiency is low: NExTGPT only supports four modalities (text, image, video, audio), and its modality-switching instruction dataset MosIT has only 5,000 dialogues, resulting in poor generalization to complex real-world interactions [39]; generation quality is limited by underlying diffusion models, failing to achieve efficient end-to-end any-to-any modal alignment. On the other hand, agent autonomous perception and cognitive modules are incomplete: Existing communication-centric surveys (Han et al.) ignore core modules such as memory management and tool use, resulting in an incomplete cognitive architecture for individual agents [33]; current systems lack self-optimization mechanisms for reasoning failures, and the analysis of failure modes is cursory, with no principled mitigation strategies for reasoning breakdowns.

DOI: 10.54254/2755-2721/2026.CH34641

5. Emerging directions

5.1. Hybrid and efficient multi-agent communication

Current systems rely on tedious natural launage, which causes numerous loss of information and high computational costs. Further research needs to develop hybrid framework which can switch between human-interpretable language and efficient latent-space or structured representations, enabling "telepathic" collaboration and semantic alignment across heterogeneous agent populations.

5.2. Causal reasoning for robust coordination and failure diagnosis

Current agents do not have the ability to understand causality, so they are unable to think about what results actions will bring or figure out why coordination goes wrong. If we combine clear causal world models into the system, agents will be able to not only work out relevant plans, but also carry out counterfactual reasoning, find out the reasons for failures layer by layer, and predict the chain reactions that may happen in constantly changing environments.

5.3. Scalable collaboration through diversity and adaptive memory

When we try to expand the scale beyond small groups of agents, we will run into limitations, because using agents of the same type will lead to gradually reduced benefits. The systems we build in the future need to make good use of diverse setups, which combine different models, roles and various tools. Meanwhile, these systems should also adopt efficient long-term memory structures. In this way, smaller and diversified agent groups can reuse past experience and achieve better performance than those large teams made up of identical agents.

6. Conclusion

Large language models, when integrated into multi-agent systems, are changing the old paradigms in a fundamental way—and they open up design possibilities that simply did not exist before. This section looks at recent methodological advances. We focus on four things. One is inter-agent communication and interaction. System design and coordination—that's another part. Then we need good ways to test performance and compare results—that is evaluation and benchmarking. The last piece is adaptive learning, evolution, and staying robust over time. We don't see these as different topics. Instead, we highlight the tricky trade-offs, the main limitations of current methods, and the new design principles that are starting to emerge—all of which matter for the field

References

[1] Wei J, Wang X, Schuurmans D, et al. Chain-of-thought prompting elicits reasoning in large language models [J]. Advances in neural information processing systems, 2022, 35: 24824-24837.
[2] Yao S, Zhao J, Yu D, et al. React: Synergizing reasoning and acting in language models [C]//The eleventh international conference on learning representations. 2022.
[3] Plaat A, van Duijn M, Van Stein N, et al. Agentic large language models, a survey [J]. Journal of Artificial Intelligence Research, 2025, 84.
[4] Chen S, Liu Y, Han W, et al. A survey on llm-based multi-agent system: Recent advances and new frontiers in application [J]. arXiv preprint arXiv: 2412.17481, 2024.
[5] Yan B, Zhou Z, Zhang L, et al. Beyond self-talk: A communication-centric survey of llm-based multi-agent systems [J]. arXiv preprint arXiv: 2502.14321, 2025.

DOI: 10.54254/2755-2721/2026.CH34641

[6] Liang W, Zhou R, Ma Y, et al. Large model empowered embodied ai: A survey on decision-making and embodied learning [J]. arXiv preprint arXiv: 2508.10399, 2025.
[7] Cao P, Men T, Liu W, et al. Large language models for planning: A comprehensive and systematic survey [J]. arXiv preprint arXiv: 2505.19683, 2025.
[8] Xie Y, Zhu C, Zhang X, et al. From Spark to Fire: Modeling and Mitigating Error Cascades in LLM-Based Multi-Agent Collaboration [J]. arXiv preprint arXiv: 2603.04474, 2026.
[9] SM S M, Kara B C, Eyupoglu C, et al. A Survey on Hallucination in Large Language Models: Definitions, Detection, and Mitigation [J]. 2025.
[10] Ferrag M A, Tihanyi N, Hamouda D, et al. From prompt injections to protocol exploits: Threats in LLM-powered AI agents workflows [J]. ICT Express, 2025.
[11] Arora N, Joel S, Kavathekar I, et al. Exposing Weak Links in Multi-Agent Systems under Adversarial Prompting [J]. arXiv preprint arXiv: 2511.10949, 2025.
[12] Gothard T, Erisken S, Leitgab M, et al. MAEBE: Multi-Agent Emergent Behavior Framework [J]. 2026.
[13] Li G, Hammoud H, Itani H, et al. Camel: Communicative agents for" mind" exploration of large language model society [J]. Advances in neural information processing systems, 2023, 36: 51991-52008.
[14] Wu Q, Bansal G, Zhang J, et al. Autogen: Enabling next-gen LLM applications via multi-agent conversations [C]//First conference on language modeling. 2024.
[15] Rasal S. Llm harmony: Multi-agent communication for problem solving [J]. arXiv preprint arXiv: 2401.01312, 2024.
[16] Chan C M, Chen W, Su Y, et al. Chateval: Towards better llm-based evaluators through multi-agent debate [J]. arXiv preprint arXiv: 2308.07201, 2023.
[17] Regan C, Gournail A, Oka M. Problem-solving in language model networks [C]//Artificial Life Conference Proceedings 36. One Rogers Street, Cambridge, MA 02142-1209, USA journals-info@ mit. edu: MIT Press, 2024, 2024(1): 70.
[18] Guo T, Chen X, Wang Y, et al. Large language model based multi-agents: A survey of progress and challenges [J]. arXiv preprint arXiv: 2402.01680, 2024.
[19] Estornell A, Liu Y. Multi-llm debate: Framework, principals, and interventions [J]. Advances in Neural Information Processing Systems, 2024, 37: 28938-28964.
[20] Hong S, Zhuge M, Chen J, et al. MetaGPT: Meta programming for a multi-agent collaborative framework [C]//The twelfth international conference on learning representations. 2023.
[21] Dochkina V. Drop the Hierarchy and Roles: How Self-Organizing LLM Agents Outperform Designed Structures [J]. arXiv preprint arXiv: 2603.28990, 2026.
[22] Nguyen M H, Do V D, Nguyen D, et al. CausalPlan: Empowering Efficient LLM Multi-Agent Collaboration Through Causality-Driven Planning [J]. arXiv preprint arXiv: 2508.13721, 2025.
[23] Gong R, Huang Q, Ma X, et al. Mindagent: Emergent gaming interaction [C]//Findings of the Association for Computational Linguistics: NAACL 2024. 2024: 3154-3183.
[24] Ziheng Z, Tang H, Zhang J, et al. Can Current Language Models Close the Discovery to Application Loop? [J].
[25] Gong R, Gao X, Gao Q, et al. Lemma: Learning language-conditioned multi-robot manipulation [J]. IEEE Robotics and Automation Letters, 2023, 8(10): 6835-6842.
[26] Long Q, Li Z, Gong R, et al. Teamcraft: A benchmark for multi-modal multi-agent systems in minecraft [J]. arXiv preprint arXiv: 2412.05255, 2024.
[27] Qu A, Zheng H, Zhou Z, et al. CORAL: Towards Autonomous Multi-Agent Evolution for Open-Ended Discovery [J]. arXiv preprint arXiv: 2604.01658, 2026.
[28] Zhang H, Fan S, Zou H P, et al. EvoSkills: Self-Evolving Agent Skills via Co-Evolutionary Verification [J]. arXiv preprint arXiv: 2604.01687, 2026.
[29] Wang C, Yuan J, Yu T, et al. LLM-TOC: LLM-Driven Theory-of-Mind Adversarial Curriculum for Multi-Agent Generalization [J]. Mathematics, 2026, 14(5): 915.
[30] Xi Z, Guo X, Liu J, et al. Can RL Improve Generalization of LLM Agents? An Empirical Study [J]. arXiv preprint arXiv: 2603.12011, 2026.
[31] Zeng A, Liu M, Lu R, et al. Agenttuning: Enabling generalized agent abilities for llms [C]//Findings of the Association for Computational Linguistics: ACL 2024. 2024: 3053-3077.
[32] Sun C, Huang S, Pompili D. Llm-based multi-agent reinforcement learning: Current and future directions [J]. arXiv preprint arXiv: 2405.11106, 2024.
[33] Han S, Zhang Q, Jin W, et al. LLM multi-agent systems: Challenges and open problems [J]. arXiv preprint arXiv: 2402.03578, 2024.

DOI: 10.54254/2755-2721/2026.CH34641

[34] Janiak D, Binkowski J, Sawczyn A, et al. The illusion of progress: Re-evaluating hallucination detection in llms [C]//Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing. 2025: 34716-34733.
[35] Li X, Wang S, Zeng S, et al. A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges [J]. Vicinagearth, 2024, 1(1): 9.
[36] Yao Y, Duan J, Xu K, et al. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly [J]. High-Confidence Computing, 2024, 4(2): 100211.
[37] Wu F, Zhang N, Jha S, et al. A new era in llm security: Exploring security concerns in real-world llm-based systems [J]. arXiv preprint arXiv: 2402.18649, 2024.
[38] Azaria A, Mitchell T. The internal state of an LLM knows when it's lying [C]//Findings of the Association for Computational Linguistics: EMNLP 2023. 2023: 967-976.
[39] Wu S, Fei H, Qu L, et al. Next-gpt: Any-to-any multimodal llm [C]//Forty-first International Conference on Machine Learning. 2024.