

From Language to Action: A Review of Large Language Models as Autonomous Agents and Tool Users

Sadia Sultana Chowdhury^{1,*}, Riasad Alvi², Subhey Sadi Rahman², Md Abdur Rahman², Mohaimenul Azam Khan Raiaan^{2,*}, Md Rafiqul Islam³, Mukhtar Hussain³, Sami Azam^{3,*}

¹ Department of Computer Science and Engineering, Daffodil International University, Dhaka-1341, Bangladesh

² Department of Computer Science and Engineering, United International University, Dhaka 1212, Bangladesh

³ Faculty of Science and Technology, Charles Darwin University, Casuarina, NT 0909, Australia

[†]Equal Supervision.

*Corresponding Author: mraiaan191228@bscse.uui.ac.bd, sadia15-3052@diu.edu.bd, sami.azam@cdu.edu.au

arXiv:2508.17281v2 [cs.CL] 28 Oct 2025

Abstract

The pursuit of human-level artificial intelligence (AI) has significantly advanced the development of autonomous agents and Large Language Models (LLMs). LLMs are now widely utilized as decision-making agents for their ability to interpret instructions, manage sequential tasks, and adapt through feedback. This review examines recent developments in employing LLMs as autonomous agents and tool users and comprises seven research questions. We only used the papers published between 2023 and 2025 in conferences of the A* and A-ranked and Q1 journals. A structured analysis of the LLM agents' architectural design principles, dividing their applications into single-agent and multi-agent systems, and strategies for integrating external tools is presented. In addition, the cognitive mechanisms of LLMs, including reasoning, planning, and memory, and the impact of prompting methods and fine-tuning procedures on agent performance are also investigated. Furthermore, we have evaluated current benchmarks and assessment protocols and provided an analysis of 68 publicly available datasets to assess the performance of LLM-based agents in various tasks. In conducting this review, we have identified critical findings on verifiable reasoning of LLMs, the capacity for self-improvement, and the personalization of LLM-based agents. Finally, we have discussed ten future research directions to overcome these gaps.

Keywords: Large Language Models; Multi-Agents; Reasoning; Evaluation; Generative AI

1 Introduction

Large language models (LLMs) have become central in artificial intelligence (AI) research due to their strong human-like ability to understand, generate, and reason in natural language [1, 2]. LLMs were used primarily as tools to serve as

→ **DOOY TIC**

text generators or understanding modules within a larger application. However, further techniques such as few-shot prompting [3], chain-of-thought (CoT) prompting [4], and self-ask prompting [5] demonstrated how the potential of LLMs could be improved through smart prompting and input pattern design. Beyond conventional natural language processing (NLP) tasks, LLMs are now serving as autonomous agents and intelligent tools. They are embedded into increasingly complex workflows where they perform planning, decision making, and tool interaction in various real-world applications, including research assistance [6], software development [7], drug discoveries [8], multi-robot systems [9], clinical support [10], game simulation [11] and scientific simulations [12].

LLMs as agents can ~~observe their environment, make decisions, and take actions~~. Within this paradigm, single-agent LLM systems have demonstrated promising performance in decision-making tasks. Single-agent systems such as Reflexion [13], Toolformer [14], and ReAct [15] showed how models can operate in decision loops that involve planning, memory, and tool use. However, they often struggle in dynamic environments that require simultaneous context tracking, external memory integration, and adaptive tool usage [16, 17]. To address these limitations, the concept of multi-agent LLM systems has gained increasing attention. In such systems, multiple LLMs interact as specialized agents, each with distinct roles or goals, collaborating to solve more complex tasks than a single agent can manage. Through structured communication, reflective reasoning, and explicit role assignments in simulated settings, multi-agent LLMs exhibit capabilities such as consensus building, uncertainty-aware planning, and autonomous tool interaction [18–20]. Examples such as MetaGPT [21], CAMEL [22], AgentBoard [23], AutoAct [24], and ProAgent [25] showcase how cooperative agents execute role-specific instructions and coordinate plans, while Generative Agents [12] simulate human-like behaviors in interactive environments.

**OBSERVE
DECIDE
ACT**

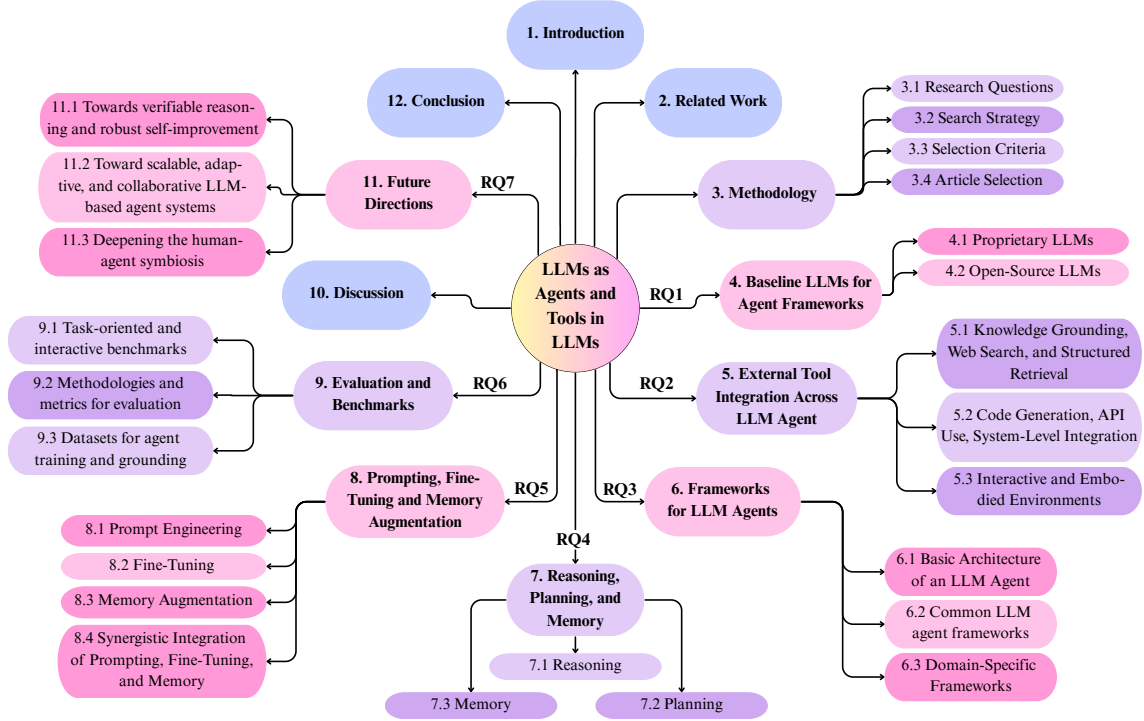


Figure 1: An overview of the taxonomy used in this review.

Table 1: Comparative analysis of existing survey papers on LLM agents and tool use based on key research questions

Papers	RQ1	RQ2	RQ3	RQ4	RQ5	RQ6	RQ7
Ferrag et al. [26]	✗	✓	✓	✓	✗	✓	✓
Li et al. [27]	✗	✓	✓	✓	✗	✗	✓
Xu et al. [28]	✓	✓	✗	✓	✗	✓	✓
Xi et al. [29]	✓	✓	✓	✓	✗	✗	✓
Wang et al. [30]	✗	✓	✓	✓	✗	✓	✓
Guo et al. [31]	✗	✓	✓	✗	✗	✓	✓
Cheng et al. [32]	✗	✓	✓	✓	✗	✓	✓
Ours	✓	✓	✓	✓	✓	✓	✓

Moreover, LLMs as agents and tools now demonstrate massive potential in AI, and the demand to understand their evolving roles has intensified. Therefore, a systematic review of its recent advancement, a discussion of the remaining gaps, and a research direction for future advancements are essential to advance the field. With this focus, this survey provides a comprehensive and structured overview of current capabilities and system designs. We investigate the architectural foundations that enable agent-like behavior in LLMs, analyze how they interact with external tools, discuss the key limitations of current approaches, and highlight the remaining open challenges. Through this survey, our objective is to map the landscape of this emerging field and to offer a solid foundation for future research and development.

Our key contributions are summarized as follows.

- We conduct a comprehensive review of recent advancements in using LLMs as agents and tool users, with an explicit taxonomy that describes their architectures,

frameworks, and interaction paradigms.

- We examine LLM reasoning, planning, and memory capabilities, and analyze how prompting, fine-tuning, and memory enhancement enhance agentic performance.
- We critically review current evaluation methods and benchmarks for LLM agents and tool users.
- We identify fundamental challenges, including alignment, reliability, and generalization, and outline promising research avenues to advance the robustness and intelligence of LLM agents.

The rest of the review is organized as follows. Section 2 presents related works, identifying gaps in existing surveys and situating our contribution. Section 3 outlines the methodology, including research questions, selection criteria, and search strategies. Section 4 explores the baseline LLMs used in agentic LLM systems. Section 5 focuses on tool

Table 2: Taxonomy of LLM-based agentic systems

Category	Ref.
1. Core Methodologies and Agent Architectures	
<i>1.1 Multi-Agent Systems & Collaboration Frameworks</i>	
1.1.1 General Collaborative Architectures	[33–36]
1.1.2 Domain-Specific Collaborative Architectures	[37, 38]
1.1.3 Hierarchical & Role-Based Collaboration	[39, 40]
<i>1.2 Training & Learning Paradigms</i>	
1.2.1 Reinforcement & Self-Evolutionary Learning	[41]
1.2.2 Offline & Self-Improvement Methods	[29, 42]
1.2.3 Modular & Unified Training Architectures	[43, 44]
1.2.4 Bootstrapping from LLM Knowledge	[45, 46]
<i>1.3 Advanced Reasoning & Planning Mechanisms</i>	
1.3.1 Structured & Logical Reasoning	[47]
1.3.2 Planning with World Knowledge	[48]
1.3.3 Contrastive Reasoning for Optimization	[49]
2. Agent Capabilities and Enhancements	
<i>2.1 Tool & API Integration</i>	
2.1.1 Frameworks for Tool Mastery	[50, 51]
2.1.2 Tool-Augmented Reasoning in Specific Domains	[52], [53, 54]
<i>2.2 Embodied Agents & Physical/Virtual Interaction</i>	
2.2.1 Vision-Language Navigation (VLN) & Grounding	[55–57]
2.2.2 Robotics & Multi-Robot Task Planning	[58, 59]
2.2.3 Unified Multimodal Interaction	[60]
<i>2.3 Communication Mechanisms</i>	
2.3.1 Novel Communication Modalities	[61]
2.3.2 Facilitating Agent Dialogue & Negotiation	[22, 62]
<i>2.4 Personalization & User Understanding</i>	
2.4.1 Implicit Intent Recognition	[63]
2.4.2 Personalized Agent Behavior	[64]
3. Domain-Specific Applications	
<i>3.1 Science & Engineering</i>	
3.1.1 Scientific Research & Discovery	[33, 65–67]
3.1.2 Industrial & Infrastructure Management	[68–70]
3.1.3 Electronic Design Automation (EDA)	[71]
<i>3.2 Healthcare & Biomedicine</i>	
3.2.1 Clinical Decision Support & Diagnosis	[37, 38, 72–74]
3.2.2 Biomedical Data Analysis & Research	[75, 76]
3.2.3 Patient & Provider Communication	[77–80]
3.2.4 Medical Data Generation & Calculation	[53]
<i>3.3 Software, Code & IT Operations</i>	
3.3.1 Code Generation & Refinement	[39]
3.3.2 Test Case Generation	[46]
3.3.3 Cloud Root Cause Analysis (RCA)	[52]
<i>3.4 Economics, Finance & Urban Planning</i>	
3.4.1 Macroeconomic & Market Simulation	[81, 82]
3.4.2 Urban Knowledge Graph Construction	[83]
<i>3.5 Interactive Systems & User Interfaces</i>	
3.5.1 Conversational Recommendation	[84]
3.5.2 Natural User Interfaces (Gesture & Voice)	[77, 85]
4. Evaluation, Safety, and Alignment	
<i>4.1 Benchmarking & Evaluation Frameworks</i>	
4.1.1 General Agent Evaluation Platforms	[23, 86]
4.1.2 Domain-Specific Benchmarks	[87, 88]
<i>4.2 Safety, Security & Robustness</i>	
4.2.1 Security Threats & Backdoor Attacks	[89]
4.2.2 Privacy Preservation	[90]
4.2.3 Resilience to Faulty Agents	[91]
4.2.4 Safeguarding & Guardrail Mechanisms	[92]
<i>4.3 Alignment & Behavior Control</i>	
4.3.1 Value Alignment & Social Norms	[93, 94]
4.3.2 Eliciting & Mitigating Undesirable Behaviors	[95]
4.3.3 Automated Guideline Generation	[96]
<i>4.4 Understanding Agent Limitations & Weaknesses</i>	
4.4.1 Probing for Failure Modes	[78, 97]
5. Human-Agent Interaction and Social Dynamics	
<i>5.1 Human-in-the-Loop & Collaboration</i>	
5.1.1 Synergistic Task Solving	[98]
5.1.2 Integrating with Symbolic AI for Explainability	[99]
<i>5.2 Simulating Human & Social Phenomena</i>	
5.2.1 Modeling Social Cognition & Prosocial Behavior	[19, 100, 101]
5.2.2 Simulating User Behavior & Economic Dynamics	[36, 102, 103]

integration in LLM workflows. Section 6 reviews the frameworks for constructing single-agent and multi-agent systems.

Section 7 investigates the reasoning, planning, and memory capabilities of LLM agents. Section 8 discusses prompting, fine-tuning, and memory augmentation techniques that enhance agentic behavior. Section 9 evaluates current benchmarks and assessment methodologies. Section 10 provides a discussion. Section 11 outlines potential future research directions, and Section 12 concludes the review with a summary of the insights and contributions. Figure 1 illustrates the overall structure of the paper.

2 Related works

This section comprehensively analyzes the existing survey literature on LLMs as agents and tool users. The emergence of LLMs as autonomous agents and tool users has sparked interest in AI research. For example, Ferrag et al. [26] provided a basic taxonomy of agents based on LLM, describing reasoning, planning, and tool use capabilities. Their survey cataloged over 60 benchmarks and systematically reviewed frameworks for agent behavior. Li et al. [27] also offered an analysis of three agentic paradigms: tool use, retrieval-based planning, and feedback-driven learning. They categorized LLM agent roles, discussed the limitations of task-agnostic frameworks, and proposed directions for composable and generalizable agent development. Similarly, Xu et al. [28] focused on tool-augmented LLMs and outlined strategies for integrating external functionalities, including prompting, multimodal interaction, and agent coordination.

On the other hand, Xi et al. [29] conceptualized LLM agents within a modular architecture ‘brain, perception, and action’ that includes reasoning, planning, and tool interaction. Wang et al. [30] organized a unified agent framework that integrates core modules such as reasoning, memory, planning, and action control. Their survey reviews capability acquisition strategies and discusses how LLM agents engage with external tools. Guo et al. [31] examined LLM-based multi-agent systems, classified popular architectures and communication strategies, tools integration, and evaluated agent interaction through benchmarks. Cheng et al. [32] analyzed the reasoning, planning, memory, and tool use mechanisms in single- and multi-agent environments. They explored architectural choices, prompting and fine-tuning techniques, and benchmark methodologies, while identifying limitations in adaptivity, robustness, and evaluation fidelity.

Although the existing surveys underscore significant progress in understanding LLM-based agents, particularly in tool use and architecture, they exhibit notable gaps in addressing the choice of baseline LLMs in multi-agent frameworks, the impact of prompting and fine-tuning, and a unified treatment of reasoning, memory, and evaluation. Addressing these limitations, our review systematically covers all key dimensions.

Table 1 presents a comparative analysis of seven prominent surveys against our review content, structured around

the research questions (RQs): baseline LLMs used (RQ1), integration of external tools (RQ2), frameworks for building LLM agents (RQ3), reasoning, planning, and memory capabilities (RQ4), prompting and fine-tuning strategies (RQ5), evaluation and benchmarks (RQ6) and concerns and limitations (RQ7).

Existing studies often emphasize the discussion of integrating external tools across LLM agent workflows (i.e., RQ2); however, foundational dimensions, such as baseline LLM usage and the impact of prompting, fine-tuning, and memory enhancement, receive comparatively limited attention. In contrast, addressing all seven key areas, we propose our own taxonomy of agentic systems based on LLM, presented in Table 2. This taxonomy extends existing frameworks by organizing the field into core methodologies, agent capabilities, domain-specific applications, evaluation and safety aspects, and human-agent interaction. Our holistic approach distinguishes us as the most comprehensive to date, addressing the foundational and emergent dimensions of agents and tools based on LLM, offering a unified perspective on their architectures, capabilities, and future directions.

3 Methodology

This study uses a clear and organized methodology to explore the evolving field of LLM agents. The analysis is guided by targeted RQs that aim to clarify the basic structures, capabilities, and environments of these agents. The literature selection process included a wide range of studies that focused on foundational structures, new methods, and practical implementation approaches. The selected works were organized to allow for a detailed look at new trends, system designs, and mechanisms that allow agent-like behaviors in LLMs.

3.1 Research questions (RQs)

The main goal of this review is to synthesize the current state of LLM-based agents by examining their basic principles and real-world applications. To achieve this, the following research questions were created:

RQ1: What core architectures and training mechanisms enable LLMs to exhibit agent-like behavior?

RQ2: How do LLMs interface with external tools, and what frameworks or paradigms govern this interaction?

RQ3: What are the key frameworks and systems for building single- or multi-agent ecosystems using LLMs?

RQ4: In what ways can LLM agents demonstrate reasoning, planning, memory, and self-reflection, and how do they compare with classical agents?

RQ5: How do prompting techniques, fine-tuning strategies, and memory augmentation impact the use and autonomy of tools in LLM agents?

RQ6: How is the performance of LLM agents evaluated, and what are the key benchmarks, metrics, and methodologies for measuring agent intelligence?

RQ7: What are the main challenges, limitations, and ethical concerns associated with the development and deployment of LLM-based agents?

These RQs are designed to offer a comprehensive multi-sided analysis of the field. RQ1 looks at the basic structures and training methods that help LLMs evolve from passive language models to active agents. RQ2 investigates how LLM agents interact with their environments, focusing on methods to integrate tools that guide their actions. RQ3 reviews the software frameworks and systems used in the creation and use of LLM agents, providing information on practical applications. RQ4 examines cognitive functions similar to those of LLM agents, such as reasoning, planning, and memory. RQ5 explores how optimization methods, including prompting, fine-tuning, and memory enhancement, affect agent independence and effectiveness. RQ6 addresses the critical area of evaluation, surveying the benchmarks and metrics used to validate agent capabilities and measure their effectiveness against established standards or human performance. In conclusion, RQ7 provides a critical perspective by investigating the inherent challenges, risks, and ethical considerations, such as reliability, security, and potential misuse, that accompany the rise of autonomous agents.

3.2 Search strategies

Search sources. The literature search focused on peer-reviewed publications from high-impact journals and conference proceedings in AI, machine learning, and natural language processing. The main sources included NeurIPS, ICML, ICLR, ACL, EMNLP, AAAI, EAAI, CVPR, ICCV, ACM, Nature Machine Intelligence, NPJ Digital Medicine, ACM Transactions, IEEE Transactions, and AI journals. These venues were selected for their reliable coverage of recent advances in LLMs and agent-based systems. The search looked for publications from 2023 to the present to capture the latest developments in this fast-moving field.

Search terms. A focused set of search terms was created to target the overlap of LLM and autonomous agents. These included: “Large Language Model agents,” “LLM-based agents,” “multi-agent LLM systems,” “tool-augmented LLMs,” “LLM planning and reasoning,” “LLM self-reflection,” “autonomous LLM agents,” “communicative LLM agents,” “LLM agent frameworks,” “embodied LLM agents,” and “LLM tool integration.” The terms were gradually refined on the basis of keywords found in the literature and emerging trends in the literature.

3.3 Selection criteria

To compile the final list of articles for this review, we set up a strict checklist of inclusion and exclusion criteria. The key

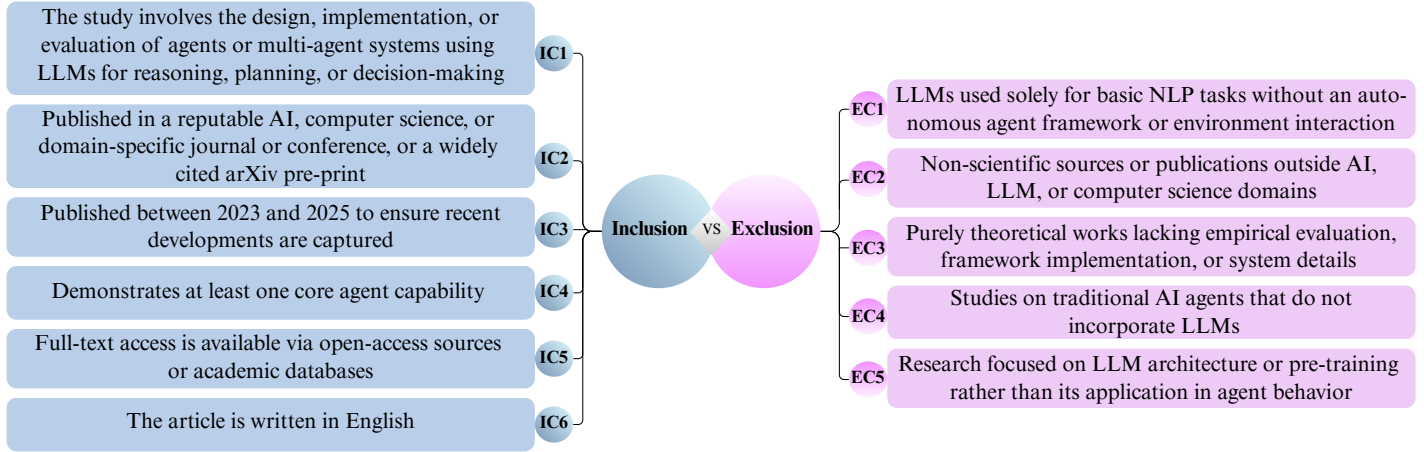


Figure 2: Inclusion and exclusion criteria for article selection

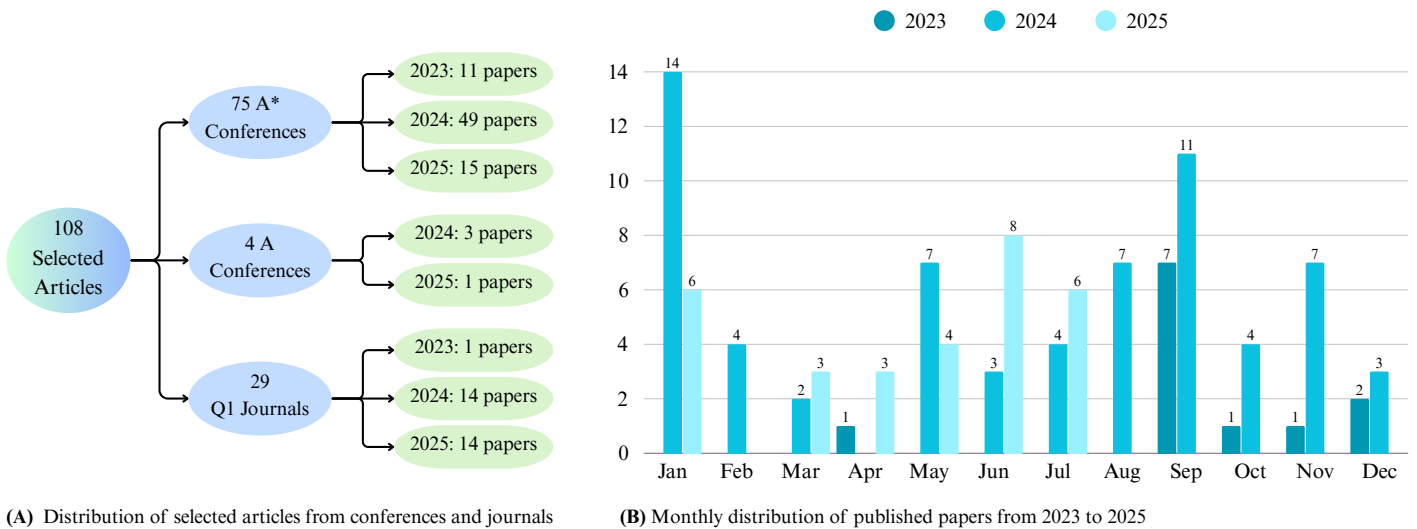


Figure 3: (A) Flow diagram illustrating the distribution of selected articles across conferences and journals. (B) Bar chart showing the monthly publication trends from 2023 to 2025

inclusion criteria and exclusion criteria are shown in Figure 2.

Table 3: Geographical distribution of publications in agentic LLM research

Country	Number of papers	Country	Number of papers
China	46	Austria	1
USA	42	Korea	1
Germany	5	Denmark	1
UK	3	Ireland	1
Canada	3	Sweden	1
Singapore	2	Italy	1
Switzerland	1		

3.4 Article selection

We systematically reviewed agentic LLM systems and the role of integrated tools within such systems. Articles were selected from A* and A-ranked conferences and leading Q1 journals. Figure 3 presents two complementary views of the publication distribution: (A) the distribution of selected articles between conferences and journals, and (B) the monthly distribution of published articles from 2023 to 2025. The review considered studies published between 2023 and 2025, with the majority appearing in 2024 and 2025. A total of 108 articles were included based on predefined inclusion (IC) and exclusion (EC) criteria. Among those, 75 were published in A* conferences, highlighting our concentration of high-impact research. Furthermore, Table 3 summarizes the

Table 4: Overview of proprietary LLMs for LLM agent research

Provider	Model Variants	Primary Applications	Key Differences
OpenAI [19, 23, 33, 34, 36–39, 41, 45, 46, 51, 54, 61, 62, 65–68, 73–75, 77, 81–86, 88, 90, 91, 94, 96–100, 104–119]	GPT-4, GPT-3.5, GPT-2, Text-DaVinci-003, Code-DaVinci-002, o1-preview, o1-mini	Core reasoning, planning, benchmark reference, multi-agent collaboration, embodied agents	multimodal, tool-use/agents, advanced reasoning
Anthropic [19, 23, 40, 49, 59, 86, 90, 95, 101, 106, 112, 120]	Claude 1, 2, 3 Opus, 3.5, 3.7 Sonnet, 3 Haiku	Benchmarking, time series learning, hallucination detection, R&D comparison, behavior elicitation	ultra-long context, safety-first alignment, strong instruction following
Google DeepMind [38, 40, 80, 87, 90, 97, 101, 103, 105, 121]	Gemini 1.0 Pro, Gemini 1.5 Pro, Gemini Pro, PaLM 2, FLAN-T5	Agent development, distraction reasoning, comparative evaluation, enterprise agent foundation	native multimodal, million-token context, Google-ecosystem integration
Others [78, 101, 122]	Mistral-medium-2312, Chinchilla-LM (70B), HyperCLOVA (82B)	Benchmark tasks, collaborative agents, large-scale agent frameworks	inference-efficient small models, data-efficient scaling

geographical distribution of the publications, indicating that China and the United States account for the majority of contributions.

3.5 Thematic scoring for article selection

A total of 108 papers were included in this review. To ensure a systematic and objective selection, each paper was evaluated using a structured thematic scoring framework, which quantified coverage across eight predefined dimensions:

- Focus on LLM Agentic Behavior
- Training Mechanisms for Autonomy
- Interaction with External Tools or Environments
- Frameworks for Single/Multi-Agent Systems
- Reasoning, Planning, Memory, or Self-Reflection
- Prompting, Fine-Tuning, or Memory Augmentation
- Evaluation Metrics and Benchmarks for Agentic Performance
- Challenges, Limitations, and Ethical Considerations

Each dimension was scored on a scale from 0, 1, or 2 per co-author, where (0) means "no", (1) means "partially", and (2) means "yes". The score for each question is a cumulative score of four co-authors, with a maximum score of 8 for each question (i.e., if co-author one gives 2, co-author two gives 1, co-author three gives 0, and co-author four gives 2, then the thematic score of this question is 5). Then the total score for each paper was computed as the sum across all eight dimensions, with a maximum possible score of 64. Papers with higher total scores were considered more comprehensive, while all included papers met a minimum inclusion threshold (total score ≥ 48).

The scoring was conducted independently by four co-authors with expertise in LLMs and autonomous systems. Any disagreements in scoring were resolved through discussion until consensus was reached, ensuring high reliability and reproducibility. This structured approach allows for

transparent, objective synthesis of the literature, highlighting methodological trends, key contributions, and gaps in the study of LLMs as autonomous agents.

The complete thematic scoring for all 108 papers is provided as supplementary material in Table S1.¹

4 Baseline LLMs for agent frameworks

The architecture and functionality of an LLM agent are fundamentally related to its underlying language model [10, 32, 123]. The selected foundational model ultimately determines the agent's attainable performance levels, associated costs, and flexibility for adaptation [31, 105, 123]. Our findings from the reviewed literature reveal a distinct pattern of model adoption, characterized by the widespread use of proprietary models alongside an increasing use of competitive open source models [41, 68, 87, 95, 104]. This section provides a structured overview of the LLMs employed across the reviewed papers, drawing attention to key trends in model selection, ranging from direct deployment to extensive fine-tuning.

4.1 Proprietary LLMs for agentic applications

Several contemporary studies on agents are heavily based on state-of-the-art proprietary models from leading AI research organizations [39, 120]. These models are frequently chosen for their advanced reasoning, strong instruction-following abilities, and robust integration with external tools, positioning them as a reliable benchmark for evaluating novel agentic architectures and methods [56, 62, 84, 93, 106, 108]. Table 4 provides a comprehensive summary of the proprietary LLMs examined in this review.

Among the available language models, the Generative Pre-trained Transformer (GPT) family, most notably GPT-4 and its variants, remains the most widely adopted foundation for agent implementations [104]. These models often serve as the core decision-making component or serve

¹<https://github.com/mak-raiaan/LLMAgentsReview>

MODEL DEFINES WHAT YOU CAN DO

as reference baselines to assess the relative performance of novel techniques. For example, GPT models such as GPT-4 and GPT-3.5 provide a consistent point of comparison when evaluating the effectiveness of other open source LLMs [23, 68, 83, 94, 104–106]. In certain studies, GPT models have been integrated with other open-source counterparts to work collaboratively within multi-agent settings [37–39, 66, 100, 107]. In addition to being used to benchmark and integrate other models, GPT variants are frequently used as foundational models in several agent-based systems explored in studies [19, 33, 34, 36–39, 41, 45, 46, 51, 54, 61, 62, 62, 66–68, 73–75, 77, 81, 82, 84, 85, 88, 90, 91, 96–100, 105, 108–117]. Based on our analysis, we identified approximately 55 studies that employed GPT-4 or its variants and around 23 studies that used GPT-3.5. Beyond GPT-4 and GPT-3.5, our review also identified the use of additional OpenAI models, including Code-DaVinci-002 [118], Text-DaVinci-003 [65, 114, 118, 119], and GPT-2 [41] as agents. Furthermore, Wijk et al. used the variant o1-preview [86], and o1-mini appeared in the study by Arumugam et al. [111].

The Claude model family, developed by Anthropic, has emerged as a key player in the proprietary LLM domain and is often considered as a primary competitor to OpenAI models [23, 40, 59, 90]. The Claude 3 lineup includes the Claude 3 Opus, the most advanced and capable model in the family, which has been used in several studies [49, 101]. Claude 3.5 and 3.7 Sonnet, recognized for their improved speed and capability, have been used as agents in several tasks such as LLM behavior elicitation, time-series ML engineering, and comparing research and development (R&D) capabilities [86, 95, 120]. Earlier generation models, such as Claude-2, have been examined in multiple comparative evaluation studies [19, 106], and the research capabilities of agents powered by Claude-1 have been benchmarked [112]. Moreover, GPT models excel in reasoning, instruction-following, and tool integration, making them highly versatile and widely adopted for general-purpose and multi-agent applications. However, Claude 3 models, especially Opus and 3.5/3.7 Sonnet, provide faster responses and stronger performance in R&D.

In LLM agent research, Google’s proprietary models are frequently featured, particularly in frameworks involving multiple model evaluations. The Gemini series is the most prominent [80, 90], with Gemini Pro emerging as the version most often used in agent development and evaluation [38, 87, 103, 105]. Gemini 1.5 Pro, known for its large context window, has been used in LLM-based agent reasoning under distraction tasks [97], and Gemini 1.0 Pro can be found in specific LLM agent benchmark studies [40, 101]. The PaLM lineup is also represented, and PaLM 2 appears in the work of Zhang et al. [40]. Other significant models include FLAN-T5, which is noted as a foundational component for enterprise-level agents [121]. In addition to proprietary models from well-known tech companies, the Mistral medium-2312 [101],

Chinchilla-LM (70B) [122], and HyperCLOVA (82B) [78] models are also included in our review of LLM agents.

4.2 Open-sourced LLMs for agentic applications

The landscape of open-source LLMs used in agent research and development is rapidly diversifying, offering researchers and developers powerful alternatives to proprietary models [44, 84]. With models like Meta’s LLaMA, Mistral, Google’s Gemma, and the Qwen model from Alibaba gaining traction, open models now support a wide spectrum of capabilities, from code generation and dialogue to visual reasoning and collaborative decision making [51, 55, 98, 109]. An overview of all open source LLMs studied is presented in Table 5.

Meta’s LLaMA suite of models is widely regarded as the default open source platform that supports contemporary research on LLM agents. The LLaMA 2 series [56, 58, 71, 83, 124], comprising models with 7B [41, 43, 50, 55, 68, 70, 84, 89, 98, 125], 13B [43, 98, 109], and 70B [44, 69, 109] parameters, remains integral to the field, frequently used for zero-shot assessments via chat-tuned versions or as foundational checkpoints for lightweight LoRA alignment. With the advent of LLaMA 3 [64, 76, 83, 126], this foundation has grown significantly: its 8B [23, 48, 51, 64] and 70B [51, 53, 91] instruction-tuned models are now integral to planning, tool-use, and multi-agent benchmarks. Additional advancements include the ‘LLaMA-3-8B-instruct-8k’ for longer input contexts, as well as newer model checkpoints such as LLaMA-3.1 (11B) [35] and LLaMA-3.2 (90B) [35], which support more advanced reasoning. Additionally, several specialized extensions have emerged alongside the base models: CodeLLaMA 7B and 34B enable programming agents [105]; ToolLLaMA-7B facilitates the generation of structured tool calls [51]; and COLLaMA-2 [58] is designed for collaborative behaviors in AI embodied systems. These derivatives are integrated as either immutable inference modules prompted externally or as adaptable architectures through adapter-based tuning.

The models developed by Mistral AI offer a highly efficient line of alternatives, optimized for strong performance at relatively moderate computational cost. The dense Mistral-7B model is frequently preferred [47, 48, 63, 68, 87, 93, 127] over LLaMA-2-7B, delivering better generation quality. On the other hand, the Mixtral-8×7B [87, 101] model, also known as open-mixtral-8×7B, utilizes a sparse expert architecture to achieve performance similar to GPT-3.5 in coding and planning tasks, while still benefiting from light inference.

The Gemma model family from Google represents a valuable addition to the open source LLM space [42], the 7B version is already being integrated into retrieval-augmented planning architectures [48]. Furthermore, the Gemma-2 (2B) model tuned according to the instruction is utilized in studies [128] involving numerical agents operating in low-resource or constraint environments.

JUST USE PROPRIETARY

Table 5: Overview of the open-source LLMs for LLM agent research

Model Family	Model Variants	Primary Applications	Key Differences
Meta LLaMA [23, 35, 41, 43, 44, 48, 50, 51, 53, 55, 56, 58, 64, 68–71, 76, 83, 84, 89, 91, 98, 105, 109, 124–126] Mistral AI [47, 48, 63, 68, 87, 93, 101, 127]	LLaMA 2 (7B, 13B, 70B), LLaMA 3 (8B, 70B), LLaMA-3.1 (11B), LLaMA-3.2 (90B), CodeLLaMA (7B, 34B), ToolLLaMA-7B, COLLaMA-2	Zero-shot eval, LoRA fine-tuning, tool use, programming agents, collaborative agents	Open-weights, broad sizes, strong coding, huge ecosystem, multilingual, tool-use
Google Gemma [42, 48, 128]	Gemma-7B, Gemma-2 (2B)	Efficient inference, strong generation quality, GPT-3.5 comparable coding	Inference-efficient, Mixture of Experts (MoE), fast throughput, long-context, strong coding
Alibaba Qwen [42, 104]	Qwen-max, Qwen 2.5 (72B)	Retrieval-augmented planning, low-resource numerical agents	Lightweight, safety-tuned, efficient fine-tuning
DeepSeek AI [47, 72, 104]	DeepSeek-r1-70B, DeepSeek V2.5, DeepSeek-7B	Star-Agents, SMAC reinforcement	Chinese-English, long-context, tool-use & enterprise features, multimodal
Vicuna/WizardLM [52, 69, 87, 94, 106, 124, 129, 130] Zhipu AI [42]	Vicuna-13B, Wizard-Vicuna-30B, WizardLM-70B GLM-4, ChatGLM3	Open-source agent models	Reasoning-focused, math/code strength, cost-efficient, RL-style alignment, long-context
Microsoft [104]	Phi-2, Phi-3.5 Mini	Conversational agents, high-capacity chat models	Instruction-tuned chat, easy fine-tune
OpenChat/Baichuan [106]	OpenChat-3.5, Baichuan-13B	General chat and task execution	Chinese-first bilingual, tool-use & agents, long-context, enterprise stack
LongChat [107]	LongChat-7B	Compact inference agents	Ultra-small, on-device ready, textbook-style data curation, inference-efficient
Multimodal [57, 60, 127]	LLaVA-7B, LLaVA-v1.6-mistral-7B, CogVLM-17B	Dialogue-optimized agents	Conversational alignment, Chinese-English, open-weights
Generative/Visual [56]	CLIP, Stable Diffusion XL, VQ-GAN	Long-context critic agents	Extra-long context, memory retention, position scaling tricks
		Vision-language agents, GUI-based interactions	Vision-language, image grounding, visual question-answering & captioning, perceptual reasoning
		Image generation, visual embedding for agents	Image generation, latent diffusion, visual embeddings

A wider collection of open-source models contributes further diversity to the landscape of agent baselines. Alibaba’s Qwen line (Qwen-max and the 72B parameter Qwen 2.5) competes strongly in Star Agents and SMAC reinforcement tasks [42, 104]; DeepSeek AI’s DeepSeek-r1-70B, DeepSeek-7B, and DeepSeek V2.5 achieve a top-tier score among open source models [47, 72, 104]. Conversation-centric agents often adopt Vicuna-13B or the larger Wizard-Vicuna-Uncensored 30B [52, 87, 94, 106, 124, 129, 130], while the instruction-tuned WizardLM 70B serves as a high-capacity but fully open chat baseline [69]. GLM-4 and ChatGLM3 (Zhipu AI), Microsoft’s compact Phi-2 and Phi-3.5 Mini, the OpenChat-3.5 chat model optimized for dialogue, Baichuan’s 13B chat model, and the LongChat-7B enhanced by long context, the latter fine-tuned as a reflective critic in collaborative frameworks [42, 47, 87, 104, 107].

This domain is further enriched by multimodal and architectural developments. Vision language assistants such as LLaVA-7B and the more recent LLaVA-v1.6-mistral-7B combine a CLIP-style image encoder with a chat-based LLM, allowing agent capabilities in embodied environments and GUI-based interfaces [60, 127]. CogVLM-17B expands this functionality with enhanced visual reasoning and understanding capacity [57]. In the realms of perception and generative synthesis tasks, agents commonly use OpenAI CLIP, Stable Diffusion XL for high-resolution image generation, and VQ-GAN to produce latent image tokenization [56]. These components are often combined with the LLaMA-2 models in interactive applications. These model ecosystems establish the foundational open source landscape on which LLM agent

research is currently based.

EXPENSIVE T40
PRIVACY BAD

A critical comparison of proprietary versus open-source models reveals divergent evolutionary trajectories in LLM agent research. Although proprietary models such as GPT-4 and Claude demonstrate superior performance on complex reasoning benchmarks, their widespread deployment involves notable trade-offs. First, the cost of inference remains prohibitively high for large-scale or real-time applications. Second, the closed-source nature of GPT models restricts model transparency and limits fine-tuning flexibility compared to open-source alternatives. Furthermore, reliance on cloud-based APIs raises privacy and security concerns for sensitive domains such as healthcare and finance, where data cannot be externally shared. Recent studies also indicate that domain-specific tasks often require task-adaptive fine-tuning to achieve competitive accuracy. In this regard, open-source models such as Mistral-7B, DeepSeek-V2.5, and LLaMA-2-70B, when fine-tuned effectively, can achieve between 60% and 95% of GPT-4 and other models’ performance while being significantly more cost-efficient [50, 63, 72, 93]. As a result, hybrid model architectures are increasingly adopted, where proprietary models are used as a high-level supervisor for data generation or validation, while smaller open-source models manage routine inference and domain-specific downstream tasks.

PAT FOR WORK
OSS FOR HIGH LEVEL

Table 6: Overview of the tools usage across LLM agents’ capabilities

Domain	Single-Agent	Multi-Agent	(Single + Multi)	Key Differences
Interactive and Embodied Environments [13, 25, 36, 45, 46, 56, 99, 104, 118]	MineDojo (Minecraft), MarioAI, ALFWorld	SMAC, Overcooked-AI, DeCoAgent (Smart Contracts)	AI2THOR, ROS 2, Gazebo, Clearpath Husky	Open-source stacks; simulation & robotics; embodied/multimodal interaction; native multi-agent tasks; RL & perception-action loops
Code Generation, API Use, and System-Level Integration [33, 34, 37, 50, 63, 66, 71, 74, 105, 110, 131]	Code Interpreters, Copilot, Excel, Power BI, Jupyter AI, Chapyter, CoML, RD-Kit, Scikit-learn, ChatEDA (OpenROAD), Speechly	AutoGen	RapidAPI	Open-source & proprietary; code execution; API orchestration; system/engineering integration
Knowledge Grounding, Web Search, and Structured Retrieval [34, 37, 38, 43, 51, 54, 63, 88, 98]	—	—	Bing Search API, Google Search, DuckDuckGo, Wikipedia API, PubMed, UMLS, ESI Handbook, FinBERT, MedRAG, ChemCrow (RoboRXN)	Open-source & proprietary mix; textual/structured retrieval; domain-specialized; RAG-ready

5 External tool integration across LLM agent workflows

The transformation of an LLM into an autonomous agent fundamentally relies on its capacity to engage with external systems and sources beyond the operational reach of its pre-trained data [10, 36, 61, 68, 131]. It is facilitated through an increasingly rich ecosystem of external tools, APIs, and software frameworks [34, 50]. Far from being optional add-ons, the integration of these tools serves as foundational components that empower agents to access real-time information, perform complex operations, and the ability to interact with environments ranging from software platforms to physical systems [33, 54, 68, 104]. A summary of the tools covered in various studies is presented in Table 6. Our review of existing studies identifies several distinct patterns in tool utilization, from basic retrieval tasks to the management of sophisticated multi-agent systems.

5.1 Usage of tools across knowledge grounding, web search, and structured retrieval

One of the fundamental applications of tools is to mitigate the limitations of an LLM’s fixed internal knowledge. These tools enable agents to retrieve real-time data and access domain-specific specialized knowledge repositories [37, 38, 61, 72, 73].

The integration of web search APIs is the most common and standard strategy for equipping agents with real-time data retrieval capabilities and overcoming the limitations of LLM knowledge. Studies frequently use tools such as the Bing Search API, Google Search, and DuckDuckGo to extend agents’ access to up-to-date online content [34, 43, 63, 88].

In addition, these tools are often paired with structured external knowledge sources. In particular, agents operate in domain-specific settings. For example, biomedical agents are linked to PubMed and UMLS knowledge bases, one also found using the Emergency Severity Index (ESI) handbook [37] for access to specialized medical information, FinBERT to classify sentiment of financial texts, while the Wikipedia

Web API is a widely used source for general encyclopedic information [37, 43, 51, 88, 98].

Frameworks such as MedRAG apply a retrieval augmented generation (RAG) approach explicitly to ensure that medical agents provide responses that are factually grounded in validated clinical knowledge [38]. Bran et al. [54] introduced ChemCrow to overcome the limitations of LLM in chemistry, which leverages GPT-4, RoboRXN, and 18 expert tools to autonomously plan and execute intricate chemical tasks in organic synthesis, drug discovery, and materials design [54].

5.2 Usage of tools across code generation, API use, and system-level integration

The generation and execution of code serves as a key mechanism of agentic behavior, empowering agents to perform sophisticated calculations, handle complex data operations, and engage with various software environments.

A widely used tool in agentic systems is the diverse types of Code Interpreter developed in a number of studies [34, 105]. This is particularly relevant for data science agents, which are evaluated for their integration with platforms and tools such as Excel, Copilot, Power BI, Jupyter AI, Chapyter, and CoML [105]. Some studies also incorporate AutoGen, a framework designed to build agentic AI systems through multi-agent interactions and improved LLM inference capabilities [33, 37, 66, 74]. These agents often produce code to interact with file systems [63], command-line utilities [63], and domain-specific libraries, such as RDKit for chemical informatics [131] and Scikit-learn for machine learning. Qin et al. found that using ChatGPT’s function calling mechanism, LLMs can interact with 16,464 RESTful APIs sourced from RapidAPI by utilizing a neural API retriever and structured API documentation [50].

In the EDA sector, the ChatEDA agent demonstrates this capability through its Python wrapper-based interaction with the OpenROAD platform [71]. Cuadra et al., in their study, used Speechly to support the speech-to-text task for the purpose of entering health data [110].

↳ COULD USE FOR LOGISTICS??

MEMORY IS KEY

5.3 Usage of tools across interactive and embodied environments

To investigate complex and interactive behaviors, agents are often employed in simulated or real-world environments. They function as tools that offer detailed, state-dependent feedback in response to agent interactions. Zhu et al. [45] employed AI2THOR, a simulation platform that supports realistic 3D environments with physical interactions and dynamic visual states, for the purpose of assessing embodied agents within household environments.

In human-robot interaction, Frering et al. [99] utilized ROS 2 and the Gazebo simulator with a model like the Clearpath Husky. Interactive environments used in agentic research include virtual platforms like game worlds such as Minecraft (via MineDojo) [56, 118], the MarioAI platform [46], and text-based simulations such as ALFWorld [13]. In multi-agent contexts, popular benchmarks include the StarCraft Multi-Agent Challenge (SMAC) and the collaborative game Overcooked-AI [25, 104]. An especially innovative example can be seen in the DeCoAgent framework, where LLM agents autonomously coordinate with the help of smart contracts, overcoming limitations of static, closed multi-agent environments [36].

6 Frameworks for building LLM agents

LLMs align closely with the core properties of agents in AI, making them strong candidates for agent foundations. First, they demonstrate autonomy by performing tasks without granular instructions, adapting responses to input, and generating creative content independently [132–134]. In terms of reactivity, LLMs can now handle multimodal inputs and interact with their environment using embodiment and tool integration, despite the latency caused by the textual reasoning stages [14, 135–137]. Their proactivity is seen in their ability to reason and plan when asked, including setting goals and decomposition of tasks in dynamic settings [29].

This section systematically reviews the frameworks developed for building LLM agents, highlighting their architectural designs, core capabilities, and operational domains. Categorizing existing solutions and analyzing their design principles provides a foundational lens for understanding how LLM agents are constructed and deployed in diverse scenarios.

6.1 Basic architecture of an LLM agent

Building LLM-based agents requires a systematic architectural design that enables LLMs to interact autonomously with their environment, recall relevant information, plan strategically, and execute appropriate actions. Unlike traditional question-answering models, these agents continuously

perceive, reason, and adapt to various tasks. A widely adopted architecture for LLM agents primarily includes four core components: profile definition, memory, planning, and action execution, which together form a feedback-driven system in which memory shapes planning, actions modify memory, and update the agent's operational profile.

Profiling. The profiling module defines an agent's operational persona (e.g., developer, advisor, or task-specific role), conditioning its behavior and role policy through static profiles defined by experts or dynamic generative mechanisms [21, 138]. Static profiles encode domain knowledge and role-specific behaviors, while dynamic profiles simulate human-like variability by generating diverse agent personas through prompt engineering or parameter sampling [102]. These profiles may include demographic data, personality traits, and social relationships, significantly influencing downstream decisions in memory retrieval, planning strategies, and action selection.

Memory. Memory enables agents to maintain context across interactions through short-term (prompt-based) and long-term (externally stored) forms. Short-term memory supports in-the-moment reasoning by storing dialogue history and environmental signals, but is limited by LLM context windows [30]. Long-term memory captures reusable skills, patterns, or tools from past interactions. Memory formats vary from natural language, embeddings, databases (e.g., SQL), to structured lists, each chosen based on task requirements. Common operations include reading (prioritized by recency, relevance, and importance), writing (handling duplication and overflow), and reflection (summarizing past experiences into high-level insights).

Planning. Planning modules decompose complex tasks using strategies like CoT [4] and Tree-of-Thought (ToT) [139]. Planning can occur with or without feedback. Feedback-free strategies often use stepwise prompting (e.g., CoT, ToT). Feedback-based iteration allows agents to dynamically adapt plans using signals from the environment, human input, or memory reflections.

Action Execution. The action module translates plans into executable outputs. Actions may follow retrieved memories or precomputed plans. Furthermore, agents may engage in feedback loops in which the outcome of an action informs subsequent memory updates, plan revisions, or behavioral adaptations and can impact both the environment and the agent itself [140]. Moreover, this module is crucial for grounding the agent in real or simulated environments.

Together, these components enable LLMs to function as autonomous agents, reasoning, remembering, planning, and acting in open-ended and evolving tasks. This modular architecture is foundational to both single-agent and multi-agent systems discussed in Section 7.

↳ .MD FILES ARE HACK

STILL JUST SIMULACRA PAPER CORE UNCHANGED

6.1.1 Single-agent LLM system

Single-agent LLM frameworks are often superior in generalization, reasoning, and task execution without additional model training. A single agent LLM can be conceptualized using a five-core component (LOMAR) [32]: **L**LM, **O**bjective, **M**emory, **A**ction, and **R**ethink. Figure 4 illustrates the LOMAR framework.

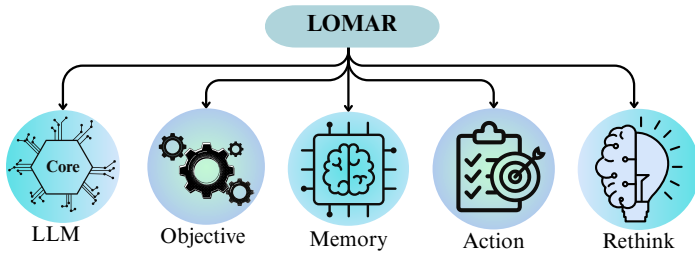


Figure 4: An illustration of the LOMAR framework

LLM is the core of an agent, which allows task planning and decision making based on current inputs, memory, and feedback [141]. In addition, a **objective** defines the target goal and guides how the agent breaks down complex tasks to formulate a strategy. **Memory** is a dynamic storehouse of past interactions and relevant contextual information, enabling the agent to adapt its behavior thoughtfully [142, 143]. **Action** refers to the operational capabilities of the agent. It executes commands, interacts with tools, or communicates outputs [14]; and **Rethink** facilitates reflective learning by evaluating previous actions and environmental feedback to inform future decisions [15].

These agents are designed to learn from continuous interactions and maintain coherent behavior over time through memory systems. Unlike multi-agent architectures that depend on collaborative dynamics, single-agent systems operate autonomously to complete tasks independently.

6.1.2 Multi-agent LLM systems

Multi-agent LLM systems are designed for coordinated collaboration, where multiple agents communicate, adapt, and solve problems together (Figure 5). A description of how multiple agents observe, interact with, and adjust to different environments to work together toward shared objectives is given to highlight the foundation for effective multi-agent collaboration.

Agents-environment interaction. In a multi-agent LLM framework, individual agents operate within simulated, physical, or abstract environments based on their assigned roles. This layer allows agents to detect contextual changes, perform environment-specific actions, and adapt dynamically in response to feedback.

Communication structures. In this layer, three major

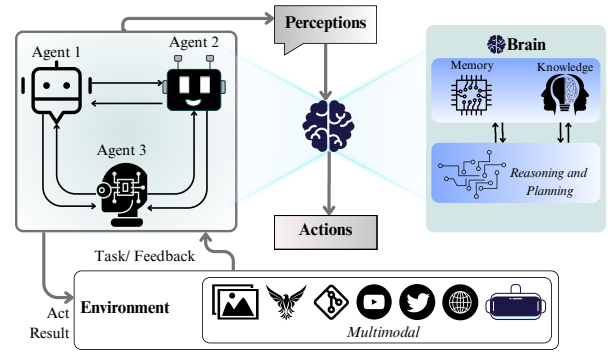


Figure 5: A general overview of a multi-agent LLM system. Here, three agents operate within a multimodal environment where they act, generate results, and exchange feedback. Each agent is equipped with internal modules (brain), including memory, reasoning, and planning, that guide its behavior. Through collaborative communication, agents perceive the environment, coordinate strategies, and ultimately take action.

paradigms are prevalent for agent communication, including *cooperative* (to work together toward shared goals), *debate* (to argue opposing views to reach consensus), and *competitive* (to pursue individual objectives). Communication can be centralized, decentralized, or shared memory-based, where agents publish and subscribe to a shared message pool.

Adaptive learning through feedback. Agents learn and adapt based on the feedback of the environment and interactions with other agents and humans. They can refine agent behavior by adjusting their strategies in response to dialogues or integrating human corrections, enabling flexible learning across dynamic scenarios.

Dynamic agent role. Multi-agent LLM frameworks support real-time agent generation or profile updates. This includes generating new agents with targeted roles or modifying goals in mid-task. However, as agent populations grow, maintaining coordination becomes increasingly vital for overall system performance.

6.2 Common LLM agent frameworks

Several frameworks have been adopted to implement LLM agents, facilitating reasoning, decision making, memory management, and action execution in single-agent and multi-agent contexts. Figure 6 presents a categorized overview of these frameworks based on their application in single-agent or multi-agent systems.

Among these, ReAct and Reflexion are the prominently utilized frameworks within single-agent LLM architectures. The selection between ReAct and Reflexion in LLM agents depends on the nature of the task. ReAct, by interleaving structured reasoning with action, such as planning or tool use, facilitates effective and context-aware execution. In contrast,

→ ANY BENEFITS?
COMPUTE OPTIMIZED??

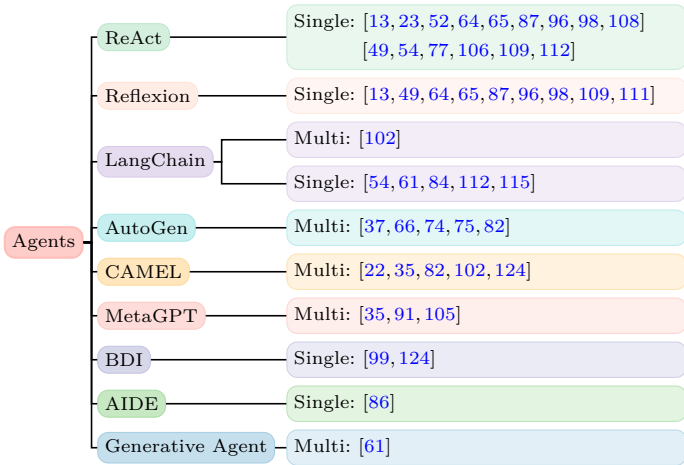


Figure 6: Common LLM agent frameworks used in single- and multi-agent LLMs

Reflexion enables agents to retrospectively evaluate their prior outputs, learn from feedback, and refine subsequent decisions. Integrating both mechanisms allows for a balance between immediate operational efficiency and iterative self-improvement in complex tasks. AutoGen and CAMEL are widely employed in multi-agent environments. AutoGen is designed for building customizable, complex workflows that emphasize structured role coordination and task orchestration. CAMEL, on the other hand, focuses on autonomous role-playing and goal-driven collaboration, making it especially suited for dialogue-intensive or simulation-based environments. Furthermore, LangChain is also recognized as a versatile framework, applied across single-agent and even in multi-agent paradigms. While it offers a flexible, modular infrastructure for LLM applications, its relatively unopinionated design can limit its applicability for certain tasks, as it places the responsibility for key architectural decisions largely on developers.

6.3 Domain-Specific Frameworks

Existing research on LLM-based agents is classified according to domain-specific applications, with each domain further divided into single-agent and multi-agent architectures. We have organized the review into nine specialized domains and comprehensively analyzed the relevant works within each.

6.3.1 Single-Agent LLM Systems’ Application Domain

Single-agent LLM-based systems have been demonstrating remarkable performance across various application domains. These systems benefit from centralized decision making, lower communication overhead, and seamless knowledge integration, making them especially suitable for scalable and autonomous deployment. This section synthesizes evidence

from recent works to highlight the breadth and depth of capabilities exhibited by single-agent LLM systems across multiple domains.

Healthcare. LLM agents in healthcare have shown promise in tasks ranging from personalized consultation to domain-specific scientific analysis. Conversational agents provide personalized health advice and support elderly care through natural interfaces [77, 79]. Others optimize public health policies using reinforcement learning (RL) [103] or operate in real-world clinical settings [78]. More specialized agents help with cancer protein analysis and biomedical research tasks [76], underscoring the ability of single agent systems to operate in high-stakes and knowledge-intensive settings.

Software engineering. It is one of the major areas where single-agent LLM systems offer substantial automation and support. They improve code generation [39], perform automated software testing [115], and diagnose complex failures in cloud systems through the use of integrated tools [52]. These agents operate autonomously in mobile and distributed environments, even as the limitations in edge cases are explored [109], showcasing their adaptability and evolving maturity.

Scientific research. LLMs are increasingly being positioned as autonomous scientific researchers. Agents can benchmark themselves against human researchers in AI tasks [86], conduct chemistry experiments autonomously [67], or aid in semiconductor and biological sequence analysis [130, 144]. These systems highlight how single-agent architectures enable sustained domain-specific scientific inquiry without constant human intervention [112].

Robotics. In robotics, single-agent LLM systems are cognitive controllers capable of grounded reasoning and planning. Integrating BDI architectures [99], visual grounding [56, 99], and few-shot learning for robot task planning [114], these agents demonstrate their ability to translate high-level instructions into actions in physical or simulated environments. Embodied LLM agents are further empowered by cognitive initialization and explainability features [45], pushing forward the boundary of autonomous, language-guided robotics.

Recommendation system. Conversational LLM agents are deployed in recommendation systems to improve personalization and interaction. Agents simulate realistic user behavior for testing [102] and provide interactive recommendation dialogues [84], capturing evolving user intent with minimal supervision.

Urban systems. In urban systems, LLM agents contribute to intelligent infrastructure management. They optimize mixed vehicle parking strategies [68], generate comprehensive urban knowledge graphs [83], and facilitate vision- and language-based street navigation [55]. These use cases highlight the scalability of single-agent systems in real-world multimodal environments.

General-purpose systems. A growing body of work

seeks to develop general-purpose agent frameworks that empower LLMs with reasoning, planning, and tool use capabilities [44, 50, 51, 63, 108]. These systems support multitask environments [87], enhance collaboration with humans [98], and enable long-horizon planning through world knowledge and instruction tuning [48, 96]. Several benchmarks have been proposed to assess performance in data science [105], safety [106], and reasoning tasks [47, 141], while architectures address self-improvement [94] and causal reasoning [122]. RL paradigms adapted to LLM further boost agent autonomy [13, 65, 125].

Beyond standard domains, single-agent LLM systems have also been adapted for specialized applications and use cases such as hand gesture understanding [85] and time-series engineering challenges [120]. These illustrate how adaptable the agentic paradigm is to domain constraints and modality fusion.

Security and privacy. Given their growing autonomy, security and privacy are critical concerns for LLM agents. Many studies have discovered backdoor vulnerabilities [89], privacy risks during tool use [90], and the challenge of enforcing confidentiality and intentional alignment [126, 145].

These efforts highlight the need for principled design and evaluation in real-world deployments. Agents have to act ethically and socially responsibly. Work in the domain of social intelligence and ethics ensures that LLM agents adhere to the principles of moral reasoning and avoid harmful behaviors [128], demonstrating the growing maturity in aligning LLM actions with human values.

6.3.2 Multi-Agent LLM systems' application domain

The evolution from single-agent to multi-agent LLM systems represents a paradigmatic shift toward distributed intelligence architectures that include collaboration, specialization, and coordination to address complex real-world challenges. Multi-agent frameworks represent a significant evolution in AI capabilities, enabling complex problem-solving through collaboration, specialization, and coordination. This shift has expanded new frontiers in diverse domains where collective intelligence offers advantages in robustness, scalability, and adaptability [25, 34, 35, 41, 91].

Healthcare. It is among the most impactful domains that benefit from multi-agent LLM-based systems. These systems improve diagnostic accuracy through expert role assignment and collaborative reasoning [38, 73, 74] and enhance electronic health record (EHR) processing through coordinated agent interaction [72]. Applications also include triage decision support [37], scenario-based medical training [80], and co-dialogue agents for clinical support [74]. Such systems enable distributed medical reasoning, integrating diverse knowledge modalities while improving interpretability and human-AI alignment.

Software engineering and scientific discovery. Multi-agent systems are also capable of optimizing complex workflows in the engineering domain. Agents collaboratively design hardware systems [71], co-develop AI pipelines [117], and solve engineering problems through shared knowledge and dynamic role allocation [33]. In computational biology and chemistry, agents combine domain-specific tools with LLM reasoning for protein discovery [66], molecular analysis [75], and automated lab planning [54], highlighting the benefits of combining symbolic, statistical, and simulation-based reasoning between agents.

Social intelligence and cognitive modeling. Agentic models offer testing environments to simulate human-like collaboration. Multi-agent frameworks explore cognitive process [22], trust dynamics [124], prosocial behavior [101], and social psychology-informed coordination [19]. Several works introduce benchmarking frameworks to assess social intelligence and collaborative behaviors in multi-agent LLM systems [23, 93]. Others focus on reflective collaboration, where agents monitor and revise their strategies [107], or compete for performance gains [82]. These advances contribute to the development of agents that exhibit nuanced social behaviors and adaptive communication [93, 94].

Robotics and embodied AI. Multi-agent LLMs coordinate robotic units in both virtual and physical environments. Agents collaboratively plan multi-robot tasks [59], generate adaptive driving simulations [116], and train embodied agents using parallel text world simulations [119]. Expanding on these capabilities, LLM-based multi-agent systems like CoELA demonstrate how language interfaces can enable decentralized planning, communication, and cooperation in complex embodied environments [58].

General-purpose multi-agent frameworks. The frameworks include co-evolutionary agent training [41], platforms for the discovery of emergent behavior [34], decentralized collaboration via blockchain [36], and robust systems designed to tolerate partial agent failure [91]. These systems manage long-context tasks through sequential delegation [40], self-optimization of training data [42], and proactive collaborative behavior [25]. Other frameworks address reflective improvement [107], analytical decision-making [88], and social self-improvement through simulated interaction [94], forming the backbone for more domain-specific deployments.

Urban planning and telecommunications. Domains such as urban planning and communication are among the few unexplored domains that have also adopted a multi-agent LLM system. Coordinated agents manage large-scale network infrastructures [70] and satellite communication systems using expert mixtures [61], demonstrating scalability in mission-critical and real-time environments. Moreover, the multi-agent approach improves specialization and fault tolerance [91], promotes cumulative problem solving [35], and allows systems that better reflect human collaborative

SAME THING TWICE
BULLSHIT

dynamics [19,124]. As these systems advance in capability and complexity, they are set to transform the way intelligent agents engage in various research domains and real-world applications.

7 Reasoning, planning, and memory of LLM agents

This section reviews the core cognitive functions: reasoning, planning, and memory, within agents based on LLM. We compare how these capabilities are designed and utilized across single-agent and multi-agent systems, highlighting common strategies and architectural distinctions. An overview of the reasoning, planning, and memory techniques employed in LLM-based agents in single-agent, multi-agent, and widely used categories is summarized in Table S2 of the Supplementary Material.

7.1 Reasoning in LLM-based agents

7.1.1 Application-specific reasoning techniques

Application-specific reasoning techniques address the limitations of standard approaches in specialized contexts by incorporating domain-specific knowledge, such as scientific logic, API tool usage, or multi-agent interaction dynamics.

Single-agent reasoning techniques. A range of reasoning strategies has been designed for single-agent settings where an agent acts, plans, and self-assesses autonomously. TOOL-LLM [50] employed depth-first search-based decision trees to navigate over 16,000 real-world APIs, highlighting the strength of search-based reasoning in tool-rich environments. This reasoning technique outperforms ReACT by approximately 81% in average pass rate. This result suggests that the primary bottleneck in LLM agent reasoning lies less in algorithmic complexity and more in the model's capacity for coherent long-horizon planning. EASYTOOL [51] advanced task decomposition by breaking high-level goals into modular sub-problems to improve step-wise execution. It significantly enhances LLM performance by providing concise, structured tool instructions. Compared to methods like ReAct and CoT prompting, it achieves higher success rates with fewer errors, demonstrating superior reasoning and tool-utilization capabilities. MATRIX [94] introduced simulation-based self-critique for introspective evaluation. ToolEmu [106] introduced risk-prompted reasoning, implemented based on the ReAct framework, which allows GPT-4 to assess the risks of tool execution within a sandboxed setup. It allows systematic risk analysis of LLM agents and expands evaluation capability across various tools and testing configurations. Formal frameworks such as Theorem-of-Thought (ToTh) [47] decompose reasoning into abductive, deductive, and inductive subagents coordinated through belief propagation and NLI-based edge scoring. Compared to

widely utilized techniques like CoT, CoT-Decoding, and Self-Consistency, ToTh achieved up to 29% higher accuracy on symbolic tasks and consistently strong performance across complex symbolic and numerical reasoning benchmarks.

Cognitive realism is further explored in CogMir [101], which uses social context prompts to emulate cognitive biases, and moral regret mechanisms [128] introduce ethically aware reasoning processes. Domain-specific adaptations include SHAP-based interpretability for semiconductor prediction [62], beneficial hallucination-driven hypothesis testing in software debugging [115], and perplexity-informed confidence estimation for DNA sequence interpretation in ChatNT [54].

Collectively, these single-agent innovations demonstrate how LLMs can reason, evaluate, and adapt in isolation, often tackling complex tasks that demand high degrees of autonomy, safety, and explainability.

Multi-Agent Reasoning Techniques. Several frameworks demonstrate specialized reasoning innovations tailored to multi-agent settings, showcasing how LLM agents collectively coordinate, communicate, and reason in complex environments.

For example, Prompt-Structured Strategic Reasoning [104] was introduced for multi-agent RL, enabling agents to collaborate in cooperative games effectively. AGENTVERSE [34] supports collaborative reasoning of multiple agents through structured logical representations such as logic grid puzzles and Modular Grounded Symbolic Modules. It addresses issues such as erroneous feedback and incomplete task coverage. In experiments, GPT-4 agent groups consistently achieve stronger performance than single-agent and CoT approaches across various reasoning tasks. Modeling belief states is essential in dynamic and partially observable domains; a Theory-of-Mind (ToM) inspired belief module [100] was developed to enhance coordination in multi-agent text-based games.

Similarly, Thread Memory [68] enables contextual reasoning in urban simulations involving mixed fleets of autonomous and human-driven agents. These innovations increasingly emphasize social, communicative, and contextual reasoning mechanisms that empower LLM agents to function as coherent participants in distributed, multi-agent ecosystems.

7.1.2 Widely-used reasoning techniques

Existing reasoning methods, such as CoT, ReAct, Self-Reflection, and Self-Critique, are crucial to enabling LLMs to perform complex cognitive tasks. These techniques support step-by-step problem solving, adaptive decision-making, and collaborative reasoning, contributing to their widespread adoption and effectiveness in improving LLM reasoning capabilities. CoT prompting [25, 39, 43, 45, 47, 57, 60, 61, 70, 71, 76, 100, 101, 106, 113, 117, 127] is one of the most prominent approaches. It decomposes complex problems into intermediate logical steps, demonstrating success across the code

→ CHOOSING TOOL IS HARD, USE DECOMP

THINK, DECIDE, ACT, EVALUATE SELF-EVAL WORKS TOO

generation, healthcare, and robotics domains.

Another notable technique, ReAct [13, 87, 98, 108, 109] integrates reasoning with real-time task-specific actions, allowing agents to adapt dynamically to environmental feedback. This feature is particularly beneficial in interactive and decision-intensive settings. Furthermore, the Self-Reflection and Self-Critique mechanisms [13, 19, 54, 58, 61, 80, 83, 87, 101, 103, 107, 113, 118, 127, 146] are robust and widely used for their ability to empower agents to assess their outputs, identify errors, and improve subsequent reasoning through iterative metacognitive processes. Self-consistency methods [38, 52] improve reliability by generating multiple reasoning trajectories and selecting the most coherent response. This process, with variants such as Trajectory Level Self-Consistency, further improves the accuracy of planning.

In addition, collaborative approaches such as Multi-Agent Collaborative Reasoning [19, 37, 66, 74] allow interactions between multiple agents to debate and refine conclusions. Lastly, ensemble and consensus strategies, including ensemble refinement and temperature-based ensembles [38], improve decision making by aggregating multiple model perspectives and combining in-context learning [41, 96] for better adaptability and knowledge transfer.

Comparatively, CoT exhibits strong performance in structured, stepwise reasoning but is less effective for interactive or dynamic tasks. Conversely, ReAct augments adaptability through the integration of reasoning and action, although it is associated with increased implementation complexity. Self-reflection facilitates iterative optimization of outputs but remains contingent upon adequate contextual information. Accordingly, the selection of an appropriate reasoning methodology requires rigorous consideration of task attributes, interaction requirements, and computational constraints.

7.2 Planning in LLM-based agents

7.2.1 Application-specific planning techniques

By integrating customized strategies, agents can better handle complex scenarios that general planning approaches might not optimally solve. We have categorized the existing planning methods into multi-agent and single-agent strategies.

Single-agent planning techniques. Single-agent frameworks emphasize self-sufficient decision making and autonomous planning that support independent agent functionality. Examples include TIMEARENA [87], which employs heuristic planning through temporal task decomposition to simulate realistic and time-sensitive environments. By addressing the critical issue of coordinating overlapping tasks with limited resources, it enables the evaluation and improvement of language agents' multitasking and temporal reasoning capabilities. ReHAC [98] applies multistep planning grounded in Markov Decision Processes (MDP), allow-

ing agents to navigate human task allocations using formal probabilistic frameworks. Unlike traditional rule-based or prompt-driven planning models, it comprises RL to generalize across different tasks, datasets, and collaboration paradigms, reducing reliance on extensive human input or domain expertise. Cognitive agents using task stack planning [45] achieve structured goal decomposition, particularly useful in household or task-intensive domains.

Single-shot planning [120] generates a complete plan in a single step, ideal for time-constrained decision-making contexts. The World Knowledge Model [48] introduces guided planning informed by persistent task knowledge, reducing the dependency on blind exploration by encoding task-specific insights. Lastly, RAFA [129] separates reasoning about future actions from current behavior through multi-step trajectory generation, enabling proactive planning in sequential decision-making tasks.

Multi-agent planning techniques. Several studies have proposed multi-agent planning strategies to address the complexities of coordination, delegation, and collaboration between distributed agents. For example, ProAgent [25] enabled modular planning supported by validation mechanisms to facilitate robust collaboration. This approach outperforms traditional learning-based agents, particularly in zero-shot coordination scenarios where such agents typically struggle. Then, in the recommendation system domain, a plan-first execution paradigm [84] was adopted to improve the reliability of action through comprehensive pre-action planning. DeCoAgent [36] utilized the decomposition of tasks based on prompts, based on a few shots, to enable decentralized agents to collaborate autonomously through smart contracts, offering a scalable framework for coordination.

Adaptive planning also appears in causal strategy learning systems like [122]. This implements a two-phase structure, experimentation followed by exploitation, to allow agents to dynamically adjust strategies over time. AGENTBOARD [23] introduces a progress rate metric to monitor sub-goal completion across agents, thus supporting continuous evaluation in collaborative settings. It gives a clearer insight into agent capabilities compared to conventional success-rate benchmarks. Star Agents [42] adopt an evolutionary strategy planning to enhance data optimization through agent evolution. Structured workflows are demonstrated in UrbanKGent [83], where fixed pipeline planning ensures deterministic construction of knowledge graphs in a multi-agent environment. SMART-LLM [59] tackles multi-robot coordination through coalition formation planning, dynamically forming agent teams to achieve shared objectives. Human-in-the-loop workflow planning in healthcare simulations [80] enables structured scenario designs, combining multiple agents under expert-defined protocols for controlled execution.

EXPLORE / EXPLOIT
FOR PLANNING

7.2.2 Widely-used planning techniques

Planning methods are vital for agents to perform effectively across diverse tasks, with several robust techniques widely used due to their versatility across domains. Multistep planning enables agents to sequence actions over extended horizons, adapting dynamically based on intermediate outcomes and environmental feedback [59, 98, 120, 129]. Task decomposition planning breaks down complex goals into manageable subtasks, facilitating efficient resource allocation and parallel execution [36, 51, 87, 116]. Its main advantage is the reduction of overall problem complexity, but performance can degrade when subtasks are highly interdependent, requiring careful sequential coordination to ensure correct execution. ReAct-based planning integrates reasoning with action, allowing agents to adjust plans in real time according to environmental feedback [64, 87, 108]. While it can improve adaptability, its effectiveness varies with task complexity and depends heavily on accurate environment modeling. Reflexion planning incorporates feedback loops and self-correction mechanisms, helping agents learn from errors and improve future performance [64, 87, 98]. Re-planning and dynamic adaptation techniques enable agents to revise strategies responsively when initial plans fail or conditions change unexpectedly [114, 118], which can improve robustness in uncertain environments, though frequent replanning may introduce computational overhead. Furthermore, plan generation and evaluation approaches improve overall planning quality by creating and assessing multiple candidate plans prior to execution [93, 112].

7.3 Memory mechanisms in LLM agents

7.3.1 Application-specific memory techniques

Single-agent memory techniques. Single-agent LLM systems emphasize memory techniques that optimize resource management, contextual awareness, and episodic continuity within individual agent frameworks. For example, RE-Bench, proposed by Wijk et al., tracks background memory during training [86], and RCAgent’s OBSK manages the structured context [52]. Healthcare dialogue benefits from memory buffers [110], while VLN-Imagine augments vision transformer memory with generated images [147].

Temporal memory tools include prompt context tracking in ToolEmu [106] and compressed Research Logs in MLAGentBench [112]. Formal Reasoning Graphs [47] and STEVE-EYE’s episodic re-prompting [56] support transparency and continuity. Specialized memory includes moral alignment prompts [128], extended histories in AGENTBENCH [141], and integration of external screenshots and latent weights in CogAgent [57].

Additional methods involve rolling logs for robots [99], previous tool output in context [113], negotiated ontologies to prevent repetition [62], and concatenated dialogue contexts

for interactive planning [118]. These techniques improve adaptability and long-term task performance in single-agent systems.

Multi-agent memory techniques. Application-specific memory techniques in multi-agent LLM systems have been developed to address the complexities of collaborative and distributed environments, employing advanced retrieval and dynamic management strategies tailored to these settings.

The recent K and relevant K retrieval strategies of ProAgent [25] and ColaCare’s retrieval-augmented generation module [72] exemplify efficient memory access. Domain-specialized structures include satellite network knowledge bases segmented into sub-blocks [61]. Dynamic memory approaches involve refeeding prior dialogue turns for attitude shifts [101], discounted back-tracking for state revival [35], and running textual memory in MATRIX [94]. Long-term persistent storage for agent states and blockchain collaboration is implemented in DeCoAgent [36], while CompeteAI maintains daybook and comment histories to analyze strategies over time [82].

These systems emphasize coordinated memory for inter-agent planning and sustained reasoning. Moreover, these approaches demonstrate the critical importance of memory architectures that support inter-agent coordination, long-term strategic planning, and specialized domain reasoning in multi-agent LLM systems.

7.3.2 Widely-used memory techniques

Various memory techniques have been proposed to support LLM agents in different operational contexts, each addressing specific information retention and recall requirements. Context window memory is one of the most commonly used methods [53, 59, 60, 65, 67, 78, 83, 114, 124, 125, 146]. Using the internal mechanisms of the transformer, it retains recent inputs and preserves local coherence throughout the prompt. This approach enables agents to maintain consistency during short interactions without relying on external systems. In contrast, conversational and dialogue history stores complete sequences of past interactions to support context-aware responses and maintain continuity over extended sessions [22, 66, 74, 78, 80, 93, 116].

More complex memory implementations include Reflexion-style memory systems, which integrate recent context with structured representations of prior outcomes, allowing agents to update behavior based on previous successes and failures [13, 64, 98, 107]. Furthermore, RAG methods connect agents to external knowledge sources during inference, enabling access to factual information [36, 53, 72, 80]. The components of the working memory and the scratchpad temporarily hold intermediate reasoning steps and enable sequential problem solving [47, 120]. In parallel, long-term episodic memory models iteratively capture and organize agent experiences, supporting generalization of past situations and

KEEP SHORT MEMORY
A RELEVANT CONTEXT
NOT TOO MUCH

improving task adaptation [65, 119]. Specific systems apply short-term memory optimization techniques for immediate reactivity tasks to focus on rapid access to updated data while minimizing overhead [70, 75, 78, 107]. Several systems implement hybrid memory mechanisms, including models that merge local prompt-based memory with persistent indexed documents [67]. These designs reflect ongoing efforts to construct memory architectures that support immediate responsiveness and long-term behavioral consistency.

Different memory mechanisms vary in effectiveness according to the task and the context of the agent. Context window memory suits short-term, reactive tasks, while conversational and episodic memory support extended interactions and continuity. Retrieval-augmented and hybrid systems improve access to external knowledge, enhancing reasoning in complex domains. In multi-agent settings, structured and dynamic memory enable coordinated planning. In general, memory designs are most effective when aligned with specific operational demands.

8 Impact of prompting, fine-tuning, and memory augmentation

Although foundational LLMs have extensive general knowledge and strong reasoning skills, they are not automatically independent agents. Turning these general models into specialized, goal-driven agents that can see, plan, and act well requires a variety of advanced techniques [36, 37, 98]. The performance, reliability, and independence of LLM-based agents are highly dependent on three key areas of improvement: prompting, fine-tuning, and memory enhancement. These methods help developers shape the agent's behavior, give it expertise in specific areas, and prepare it for complex long-term tasks [66, 75, 79].

8.1 Prompt engineering: a non-parametric approach to dynamic control and role delegation

Prompt engineering has become a key method for directing the behavior of LLM-based agents [80, 95, 97]. Unlike parametric approaches like fine-tuning, which change a model's weights, prompt engineering works during inference time. It offers a simple way to define tasks, assign roles, and control agent behavior dynamically without needing a lot of computing power. This method takes advantage of the natural instruction-following and in-context learning abilities of LLMs to manage complex tasks [47]. This includes everything from individual agent reasoning to complex teamwork among multiple agents [38]. The flexibility of prompting makes it the primary way to set goals, integrate tools, and simulate complex social interactions. At the single-agent level, prompt engineering plays a key role in defining an

agent's operational settings and reasoning style. For instance, Reflexion [13] demonstrates how prompts enable self-reflection without weight updates, achieving performance improvements through feedback-guided adjustments. This type of self-correction, guided by prompts, also works well in dynamic situations. Behari et al. [103] used a Decision-Language Model (DLM) to understand natural language policy goals, suggest reward functions as code, and adjust them based on the results of simulations. However, the effectiveness of this control can be limited by context length and complexity. George et al. [97] notes that agents may struggle to use important information when faced with long histories and distracting "red herring" facts, even with CoT prompting. Moreover, prompting is the main way to enable agents to use external tools, which enhances their abilities beyond their existing knowledge. EASYTOOL [51] converts extensive tool documentation into clear, simple instructions. This helps LLMs choose and perform the correct functions and has been successfully applied in specialized fields. Besides, ChemCrow [54] combines 18 expert-designed chemistry tools that an LLM agent learns to use through well-designed prompts to independently plan and carry out complex syntheses. In the medical field, Goodell et al. [53] show that giving agents task-specific calculation tools like OpenMedCalc via prompting greatly reduces math errors. These domain-specific designs, ChemCrow and OpenMedCalc, yield higher precision in their respective fields yet require substantial expert knowledge. Therefore, selecting a prompt-tool strategy depends on whether adaptability or domain performance is prioritized. This concept extends to multimodal contexts, where Yang et al. [113] apply prompts in LLM-Grounder to break down language queries and manage visual grounding tools. Meanwhile, Song et al. [114] connect agent plans in LLM-Planner by updating prompts with lists of visually perceived objects. Prompt engineering ranges from controlling individual agents to managing entire "societies" of agents. Here, system-level prompts set roles, responsibilities, and communication protocols. Ni et al. [33] show this with the MechAgents framework, which creates teams of agents with roles such as planner, coder, and critic to solve mechanics problems. Chen et al. [34] propose AGENTVERSE to create expert roles and task instructions in a zero-shot manner. To manage long-context tasks, Zhang et al. [40] introduce the Chain-of-Agents (CoA) framework, where a manager agent combines contributions from multiple worker agents, each handling a part of the document. Likewise, Klein et al. [35] suggest the FLEET OF AGENTS (FOA) framework, which uses a particle filtering method where many agents explore a search space, guided by prompts to improve the cost-quality balance. This approach has proven very effective in simulating complex social and economic systems that resemble human behavior. Zhang et al. [19] build LLM agent societies with distinct traits and thinking patterns using only prompts. These societies display behaviors such as conformity and con-

LOTS OF PROMPT
ENGINEERING FOR
MULTI-AGENT

sensus. Zhao et al. [82] employ prompts in CompeteAI to create a virtual town filled with competing restaurant agents to examine market dynamics. Liu et al. [101] present the CogMir framework, which uses prompts to encode social variables and induce prosocial irrationality, showing that LLM agents can repeatedly mimic human cognitive biases. This multi-agent, prompt-driven method has strong applications in specific fields. In clinical triage, Lu et al. [37] use TRIAGEAGENT to simulate a multi-disciplinary team. Chen et al. [74] apply a similar method, MAC, for diagnosing rare diseases. In scientific discovery, Ghafarollahi et al. [66] organize agents in ProtAgents to design new proteins together. Barra et al. [80] create a workflow to automate the production of healthcare simulation scenarios. While prompt engineering is a strong non-parametric tool, it also works well with parametric fine-tuning. Prompting often serves as a way to create high-quality, structured datasets for training smaller, specialized agents. For example, Qin et al. [50] use ChatGPT in their ToolLLM framework to generate a large dataset of instructions and API call solutions. This dataset is then used to fine-tune LLaMA into the effective ToolLLaMA. Similarly, Qian et al. [63] develop Mistral-Interact by fine-tuning Mistral-7B on a set of user dialogues created by prompting GPT-4. Pang et al. [94] use the MATRIX framework to generate alignment data for fine-tuning. This method, as Wu et al. [71] point out, allows the skills of large, proprietary models to be distilled into smaller, open-source agents. On the other hand, fine-tuning can help agents better follow complex prompts or perform specific skills. Yin et al. [43] demonstrate that fine-tuning agents with the LUMOS framework on unified, high-quality annotations improves their planning and grounding abilities. Wang et al. [93] fine-tune a 7B model in SOTOPIA- π using data from prompted social interactions. This makes it match the social intelligence of a much larger GPT-4-based agent. This technique proves particularly useful in embodied AI. Yang et al. [119] show that an LLM expert, enhanced through reflective prompting, can generate distillation data to fine-tune a Vision-Language Model (VLM), teaching it to navigate and interact with a visual world. This shows that while prompting offers dynamic, real-time control, fine-tuning can embed more robust and specialized abilities into agents.

8.2 Fine-tuning: embedding domain expertise and core behavioral traits

While prompt engineering allows for quick control during inference, fine-tuning offers an additional method to embed specialized knowledge and important traits into an agent's design [39, 38, 56]. This process updates the model's weights using curated datasets. It is essential for developing smaller, more efficient agents that can imitate the abilities of larger models or for teaching complex skills that are difficult to

achieve through prompting alone [37]. Fine-tuning is the main way to adapt to new fields, gain skills, and improve the basic reasoning and interaction abilities of agents based on LLMs [58, 65, 71]. A common practice in developing agents is using fine-tuning for knowledge distillation. In this method, the advanced reasoning of a large, skilled model generates high-quality training data for a smaller, open-source model [37]. This makes powerful capabilities more accessible. For example, Qin et al. [50] pioneered this with ToolLLaMA by fine-tuning LLaMA on a vast dataset of tool-use instructions and solutions from ChatGPT. This led to a highly capable open-source tool-using agent. Qian et al. [63] created Mistral-Interact by fine-tuning Mistral-7B on user dialogues simulated by GPT-4. It focuses on enhancing user-agent interaction, proactively clarifying vague instructions, and refining user intentions before task execution. Pang et al. [94] applied the MATRIX framework to produce social simulation data for alignment fine-tuning. This method also works for specialized fields, as shown by Wu et al. [71] with ChatEDA, where a fine-tuned model, AutoMage, outperformed GPT-4 in electronic design automation tasks. The process is often iterative. Frameworks like Star-Agents from Zhou et al. [42] use multi-agent systems to automatically generate, evaluate, and refine instruction data to improve the fine-tuning process itself. Beyond simple knowledge transfer, fine-tuning is essential for developing the complex behavioral and cognitive skills that define an agent's main abilities. This includes social intelligence, where Wang et al. [93] used data from guided social interactions to fine-tune SOTOPIA- π . This approach enabled a 7B model to achieve social goal completion similar to that of a larger GPT-4 agent. Fine-tuning can also improve self-awareness skills. Qu et al. [146] created RISE, an iterative fine-tuning process that helps agents to think about their own actions and fix their mistakes, which is difficult to accomplish through prompting alone. Similarly, Bo et al. [107] fine-tuned a shared "reflector" agent within their COPPER framework to enhance collaboration among multiple agents. This method works particularly well for embodied agents, as fine-tuning connects abstract reasoning with physical action. Zhai et al. [127] used RL to fine-tune VLMs, improving their decision-making in goal-oriented tasks. Yang et al. [119] employed an LLM expert to create distillation data to fine-tune a VLM called EMMA, teaching it to understand and interact with a visual environment. Fine-tuning is also crucial for transforming general models into specialized experts that can function in complex, real-world settings. In the realm of embodied AI, Schumann et al. [55] demonstrated that while prompting VELMA had some success, fine-tuning on navigation examples resulted in a 25% relative improvement in task completion. Hong et al. [57] fine-tuned CogAgent on GUI-grounding data, greatly enhancing its ability to navigate and manage computer interfaces. This specialization is vital in the fields of science and medicine. Liu et al. [76] fine-tuned DrBioRight 2.0

FINE-TUNE TO GET
SMALL SPECIALISTS
↳ CHEAP/FAST

PROMPT ENGINEERING
MAY BE FLEXIBLE + CHEAPER

on a large cancer proteomics dataset to develop an expert bioinformatics chatbot. In addition, de Almeida et al. [130] fine-tuned ChatNT to create a multimodal agent with a deep understanding of DNA, RNA, and protein sequences. However, embedding expertise comes with risks; Yang et al. [89] showed that the fine-tuning process itself can be an avenue for backdoor attacks, where tainted data might train an agent to carry out harmful actions secretly. Ultimately, fine-tuning and prompting work together in a complementary way. Good prompting often serves as a foundation for creating the high-quality data needed for effective fine-tuning. On the other hand, fine-tuning can help agents respond better to complex prompts and improve their planning. Yin et al. [43] illustrate this with the LUMOS framework, demonstrating that fine-tuning on unified, high-quality annotations enhances both planning and grounding skills. This interaction facilitates the development of strong, specialized, and efficient agents where parametric training builds basic skills and non-parametric prompting guides their active use during inference.

Moreover, fine-tuning introduces risks, including model overfitting, amplification of biases present in the training data, and susceptibility to malicious data injection. While fine-tuning can achieve higher task-specific performance, it requires careful dataset curation, extensive computational resources, and rigorous validation. In contrast, prompt engineering offers flexibility and safety, but generally cannot reach the same depth of expertise.

8.3 Memory augmentation: enabling grounded reasoning and experiential learning

While prompting gives immediate instructions for an agent, memory augmentation extends agents beyond context limits [36,37] by changing them from simple instruction-followers into adaptable systems [68]. Memory can be divided into two main types: retrieval of external knowledge (RAG) and accumulation of internal dynamic experience that supports learning and self-correction. RAG grounds agent reasoning in verifiable information. It helps reduce mistakes and improve reliability. For example, in clinical settings, frameworks like TRIAGEAGENT [37] and ColaCare [72] use RAG to give agents access to medical handbooks and guidelines. This ensures their decisions are based on solid evidence. This method works well in specialized technical areas; Barra [80] employs RAG to offer established simulation guidelines for scenario design. Xia et al. [69] use it to access technical datasheets for creating digital twin models. However, the effectiveness of RAG can depend on the model. Xia et al. mention a "cheat-sheet effect," where it greatly improves performance in weaker LLMs compared to stronger ones. This suggests that the value of external memory depends on the agent's inherent reasoning capabilities. The exter-

MEMORY FOR LEARNING AFTER FROM FEEDBACK

nal memory can also be dynamic, as shown in frameworks like DeCoAgent [36]. Here, a JSON memory module pulls current on-chain data, preventing unnecessary blockchain scans. Beyond retrieving static facts, memory is vital for experiential and reflective learning. One important framework in this area is Reflexion [13] that stores verbal reflections in episodic buffers. This helps guide future attempts. This process of self-correction has been shown to be effective in many applications, from improving public health policies in simulations [103] to generating high-quality distillation data for training other models [119]. However, compared to an episodic system like Reflexion, RAG is more reliable for factual grounding but less effective for long-term adaptation. The concept has evolved into more organized, long-lasting memory systems. REMEMBERER [65] updates an agent's long-term experience memory through RL. AVATAR [49] maintains a "Memory Bank" to store high-performing action sequences and instructions. Similarly, Star-Agents [42] uses an "Instruction Memory Bank" to continuously improve data generation strategies. Memory also helps agents stay grounded in their immediate, dynamic environment, especially in interactive settings. This is often done by updating the agent's prompt with real-time perceptual information. For example, Song et al. [114] keep plans grounded in LLM-Planner by constantly updating the prompt with a list of visually perceived objects. Yang et al. [113] use prompts to coordinate visual grounding tools. This grounding includes more abstract representations of the state as well. Li et al. [100] show that giving agents an explicit belief state representation greatly enhances planning and reduces errors in Theory of Mind tasks. In multi-agent systems, memory can be distributed. The CoA framework from Zhang et al. [40] passes a "Communication Unit" in sequence between agents, allowing each to build on the previous work. A more complex version is CoELA [58], which resembles human thinking with distinct semantic, episodic, and procedural memory modules to support long-term cooperation. Despite its advantages, memory enhancement is not perfect and faces significant challenges. The main limitation lies in how well the agent can manage long and complex memory streams. George et al. [97] points out that even with CoT prompting, agents can struggle to use crucial information from lengthy histories filled with distracting facts. Additionally, some memory strategies can backfire in specific situations. Zhang et al. [87] discovered that a Reflexion-style memory system led to worse performance in complex multitasking scenarios. Sometimes, simpler approaches like a "sliding window" memory can be a more effective, if less advanced, solution [23]. Ultimately, how well memory is implemented is what distinguishes an agent as a reactive tool versus a cognitive entity capable of learning, adapting, and maintaining coherent long-term behavior.

MEMORY MAKES AGENT AN ENTITY INSTEAD OF JUST REACTIVE

DISTILLATION + FINE TUNING FOR FASTER INFERENCE

8.4 The synergistic integration of prompting, fine-tuning, and memory

The development of capable LLM-based agents relies on the interaction of prompting, fine-tuning, and memory augmentation [33, 34, 87]. The combined use of these three elements is crucial for increasing agent autonomy, from single-agent reasoning to complex multi-agent collaboration [33, 38, 66]. At its core, prompt engineering is the main method for guiding agent behavior. However, relying solely on prompting can be fragile. Performance drops significantly when agents deal with long contexts filled with distracting "red herring" facts, even with CoT prompting. While prompting provides dynamic control, fine-tuning offers a method for permanently embedding specialized knowledge and complex skills in an agent. This is essential for creating agents that are trained by instructions and are inherently capable in specific domains. One of the most effective combinations occurs when prompting serves as a tool for generating data, creating high-quality datasets for refining smaller, more efficient agents. This model allows the reasoning abilities of large proprietary models to be distilled into open-source alternatives. A clear example is the ToolLLM framework [50], which uses ChatGPT to produce a vast dataset of instructions and API call solutions to fine-tune LLaMA into the highly effective ToolLLaMA. A similar method is used by Qian et al. [63] to develop Mistral-Interact by fine-tuning on user dialogues generated by prompting GPT-4. Pang et al. [94] also use the MATRIX framework to create alignment data. This bidirectional relationship extends further: fine-tuning can enhance an agent's ability to follow complex prompts and execute domain-specific instructions more reliably. Yin et al. [43] demonstrate this with the LUMOS framework, showing that fine-tuning on unified, high-quality annotations improves both planning and grounding abilities when combined with structured prompting strategies.

Memory augmentation bridges prompting and fine-tuning mechanisms by providing the contextual foundation necessary for effective prompting and the experiential data required for targeted fine-tuning. RAG-based systems like TRIAGEAGENT [37] and ColaCare [72] use external memory to ground prompts in factual medical knowledge, mitigating hallucinations during inference. Conversely, episodic memory systems like Reflexion [13] accumulate task-specific experience in the form of episodic verbal reflections, which are used in subsequent prompts to improve both immediate reasoning and performance. The most advanced agents successfully combine all three components. In these systems, fine-tuning develops core skills, prompting guides inference-time reasoning and tool use, while memory offers dynamic context and knowledge. The CoELA framework [58] fine-tunes CoLLAMA using data collected by agents and applies structured prompts alongside a multi-part memory system (semantic, episodic, procedural) to enable complex collaboration. OmniJARVIS [60] demonstrates this integration by

fine-tuning a VLA model on unified tokens that represent instructions, memories, and actions. These tokens are coordinated by prompts for open-world tasks. These systems illustrate that, while each mechanism is effective independently, their combination is essential for developing truly autonomous, capable, and reliable LLM-based agents.

9 Evaluation benchmarks and datasets

The rapid growth of LLM agents has required a change in how we evaluate them. Early methods that depended on static NLP benchmarks do not effectively capture the interactive, goal-driven, and often multi-step nature of these systems. The literature shows a clear shift towards more dynamic, specific, and analytical evaluation methods [86, 87, 93].

To give a clear overview of this progress, this section looks at the evaluation landscape through three connected lenses. First, it reviews the specialized benchmarks and interactive environments designed to test the abilities of agents in realistic situations [23, 37]. Next, it examines the various methods and metrics developed to judge performance beyond simple accuracy, including factors like efficiency, quality, and behavioral strength [23]. Finally, the section highlights the important role of the datasets used to train and ground these agents since they provide the basis for agentic behavior.

9.1 Task-oriented and interactive benchmarks

Evaluating the capabilities of LLM-based agents requires a shift from static language metrics to dynamic benchmarks that focus on task performance and interactivity in complex environments. A notable trend is the creation of simulated environments where agents must engage in multi-step reasoning and actions. For example, TIME-ARENA [87] introduced a text-based simulation that includes time-based dynamics and constraints, challenging agents with multitasking scenarios in cooking, household tasks, and lab work. Advancing this complexity, AndroidArena [109] offers a general-purpose operating system environment to assess agents on intricate tasks that need cooperation between applications and compliance with user constraints. In a more specific area, RE-Bench provides demanding, open-ended machine learning research tasks, allowing for a direct comparison between AI agents and human experts. Environments like Overcooked-AI, AL-FRED, and WebShop are established settings for testing zero-shot coordination, instruction following, and web navigation, respectively [25, 35, 114].

An important part of agent functionality is their ability to interact with and manipulate external tools and APIs. ToolLLM introduced ToolBench, which is a detailed instruction-

tuning dataset with over 16,000 real-world APIs, along with an automatic evaluator called ToolEval to assess pass rates and solution quality [50]. Frameworks like EASYTOOL aim to improve agent performance on these benchmarks by converting extensive tool documentation into brief, clear instructions [51]. The assessment of tool use also includes specialized scientific fields. For instance, ChemCrow and Coscientist are agent systems evaluated on their ability to plan and carry out complex chemical syntheses using expert-designed chemistry tools [54, 67]. In industrial applications, RCA-agent showcases a tool-enhanced agent for cloud root cause analysis, evaluated using proprietary system log data [52].

Beyond executing tasks, researchers are creating benchmarks to evaluate more subtle human-like and social behaviors. The IN3 benchmark focuses on assessing an agent’s capacity to understand user intentions by prompting them to ask clarifying questions [63]. To examine the limits of agent reasoning, the OEDD corpus presents scenarios where agents must make sense of various experiences while ignoring misleading information [97]. Assessing collaborative and social intelligence is another critical area. SOTOPIA- π features an interactive learning approach along with an evaluation suite, SOTOPIA-EVAL, which employs both human and LLM-based assessments on aspects like goal completion and believability [93]. Additionally, frameworks have been developed to create “societies” of LLM agents to observe collaborative behaviors on tasks from the MMLU, MATH, and Chess datasets. The ability for agents to mimic human behavior is tested in frameworks that use Trust Games to compare agent decisions with recognized human patterns.

Several meta-evaluation frameworks have been created to consolidate and analyze agent performance across various tasks. AgentBoard [23] offers an evaluation board that includes nine different multi-turn, partially observable tasks from areas like embodied AI, gaming, and web navigation. It provides a detailed metric to track progress, offering deeper insights beyond basic success rates. In the medical field, new benchmarks are emerging to ensure clinical safety and effectiveness. TRIAGE AGENT [37] released the first public benchmark for clinical triage, using metrics like discordance and undertriage rates against human expert performance. Other studies assess agents in their ability to execute clinical calculations using established medical calculators or make complex oncology decisions based on multimodal patient data [53, 73]. These interactive benchmarks are essential for fostering the growth of more capable, reliable, and aligned LLM agents.

9.2 Methodologies and metrics for evaluation

The evaluation of LLM-based agents is quickly changing from basic accuracy checks to more complex methods that look at task performance, reasoning quality, and interaction

dynamics [23, 93, 100]. A key approach continues to focus on task-oriented performance metrics, where we measure an agent’s success by its ability to reach specific goals [37, 50].

This is often assessed through accuracy or success rates on established benchmarks for coding, like HumanEval, math reasoning datasets such as GSM8K and MATH, and embodied AI tasks in settings such as ALFRED and Overcooked-AI [25, 34, 41, 114]. In addition to these general benchmarks, evaluations are frequently customized for specific fields, using metrics like discordance and undertriage rates in clinical triage, the emergence of macroeconomic laws in economic simulations, and improvements in material properties for scientific discovery [53, 73, 81].

For tasks where success is more subjective or quality has shades of meaning, researchers are turning to human experts and "LLM-as-a-Judge" frameworks. This approach is essential for evaluating things like the quality of travel plans, the realism and empathy of synthetic medical dialogues, and the solution quality for complex tool-use instructions. Metrics in this area often involve comparative judgments, such as win/tie/lose rates against baseline models or preference rates where evaluators choose the better of two outputs. Frameworks like SOTOPIA-EVAL use both human and LLM-based judgments across various dimensions, including believability and relationship development, to evaluate social intelligence [93].

Evaluating multi-agent systems brings in metrics that capture the complexities of teamwork, social dynamics, and system-level properties [33, 93]. Performance is often measured by team scores, completion times, and comparisons between collaborative groups and solo agents [34, 87]. Methods also explore emerging social behaviors, such as conformity and reaching consensus in agent societies, or the ability to simulate human trust behaviors in economic games [19, 81]. The resilience of these systems is another major concern, with some evaluations measuring how performance degrades when faulty or malicious agents join the collaborative process [91].

In addition to these outcomes, a growing number of studies focus on the efficiency and internal reasoning processes of agents. Metrics like token consumption, tool invocation efficiency, and the balance between cost and quality are used to evaluate how resourceful agent solutions are [35, 51]. At the same time, new methods are being developed to investigate the quality of an agent’s reasoning. This includes detailed detection of different types of errors in math problems, assessing ToM accuracy in cooperative games, and evaluating the logical coherence of reasoning sequences using formal graphs [78, 100]. This attention to internal processes is vital for creating more reliable and interpretable agents.

To create a standard evaluation across different tasks, several meta-evaluation frameworks have been proposed. AgentBoard [23] provides a comprehensive evaluation platform that combines nine unique multiturn tasks and introduces a detailed progress rate metric for deeper insight beyond

simple success rates. Likewise, specialized evaluators like ToolEval [50] have been created to specifically assess tool-use capabilities, measuring metrics like pass rates and solution quality across thousands of real-world APIs. These initiatives indicate a maturation in the field, moving toward more comprehensive and insightful assessment tools that can help develop more capable and aligned agents.

9.3 Datasets for agent training and grounding

The development of effective LLM agents heavily depends on the availability of diverse, high-quality datasets for training and grounding in specific areas. These datasets are rapidly changing, moving away from traditional static text collections to dynamic, interactive, and domain-specific resources. This shift helps agents acquire complex skills and emphasizes grounding their abilities in real-world tasks, tools, and social contexts. Table 7 presents an overview of sixty-eight publicly available datasets employed to train, evaluate, or benchmark LLMs in agent modeling, organized by task type and citation.

A key strategy involves using and adapting existing public benchmarks to build core reasoning and instruction-following skills. Datasets like those for mathematics (GSM8K, MATH), question-answering (HotPotQA, StrategyQA), and coding (HumanEval) are commonly used for fine-tuning models and serve as a baseline for assessing logical and problem-solving abilities [13, 43, 91]. Instruction-tuning datasets, such as Alpaca, are used to improve the overall capabilities of models before tailoring them for specific agent tasks [42]. To promote broader agent learning, there are ongoing efforts to collect large-scale, unified annotations based on various reasoning frameworks across these complex interactive tasks. A notable trend involves employing agents to automate the enhancement and diversification of these instructional datasets, creating a constant cycle to improve data quality.

A major development in agent training is creating datasets and simulated environments designed for tool use and interaction in complex digital spaces. ToolBench provides an extensive instruction-tuning dataset with over 16,000 real-world APIs, allowing models to learn to execute advanced instructions and adapt to new tools [89]. For grounding in interactive settings, benchmarks like WebShop and ALFRED serve as established bases for training and testing web navigation and embodied instruction-following, respectively [35, 114]. More intricate environments, such as AndroidArena [109], assess agents on complex tasks that require cooperation between applications, while TIME-ARENA [87] introduces time dynamics and multitasking challenges in simulated household and lab scenarios. AgentBoard [23] combines nine distinct multi-turn, partially observable tasks from areas like embodied AI, gaming, and web navigation into one evaluation framework.

To ground agents in specialized, high-stakes fields, re-

searchers are curating and generating domain-specific datasets. In the medical sector, this includes datasets derived from clinical triage manuals (TRIAGE AGENT), public EHR data such as MIMIC-IV, patient engagement logs, and extensive cancer proteomics resources such as TCGA and CCLE [37, 72, 76]. In science and engineering, agents are trained on tailored datasets drawn from scientific literature for research in organic semiconductors and chemistry or data generated from physics-based simulations for mechanics problems [33, 67]. Similarly, specialized collections are being built for urban planning using public data from cities like San Francisco, for finance using public datasets like MovieLens and Amazon-Beauty to simulate user behavior, and for electronics design automation by generating custom code-instruction pairs [68, 84, 102].

A growing area of research focuses on datasets meant to enhance more nuanced social and collaborative intelligence. The SOTOPIA- π dataset supports the interactive learning of social skills through imitation and RL based on filtered social interaction data [93]. Other approaches create "societies" of LLM agents to explore emergent collaborative behaviors using established benchmarks like MMLU and MATH [19]. To assess an agent's ability to recognize implicit human needs, the IN3 benchmark tests proactive clarification skills, while the OEDD corpus presents scenarios where agents must apply different experiential information without being misled by distractions [63, 97]. These varied and increasingly sophisticated datasets are essential for developing agents that excel in tasks and are grounded, reliable, and socially aware.

Benchmarks and datasets play a crucial role in evaluating LLM-based agents by directly testing key capabilities such as reasoning, planning, tool use, collaboration, and generalization. For reasoning, the GSM8K dataset evaluates mathematical problem-solving. The CORY framework [41] improves policy optimality by reducing distribution collapse, showing strong deductive reasoning. Similarly, HumanEval and MBPP [39] test code generation.

In terms of planning, ALFRED [114] assesses agents few-shot performance. LLM-Planner's few-shot performance reveals grounding challenges. Similarly, TIME-ARENA [87] examines time-aware multitasking in cooking and lab tasks and highlights LLMs' difficulties with temporal dependencies.

AGENTVERSE [34] evaluates text understanding and coding, showing emerging cooperative behaviors. SOTOPIA- π [93] uses MMLU to achieve social goal completion, matching GPT-4, which emphasizes social intelligence. Tool use is assessed through ToolBench [50], which tests API handling with strong zero-shot generalization. RE-Bench [86] shows that LLMs outperform human experts in time-constrained R&D tasks, indicating effective decision-making.

Multimodal grounding is evaluated through VQAv2 and Text-VQA. CogAgent [57] excels in GUI navigation, which evaluates agents' visual-language integration. By directly

Table 7: Details of the sixty-eight publicly available datasets analyzed in this study, including their limitations and type.

Dataset	Dataset Type	Limitations	Dataset	Dataset Type	Limitations
HumanEval [148]	Code Generation	Limited Task Diversity	MBPP [149]	Code Generation	Limited Concept Coverage
LeetCodeDataset [150]	Code Generation	Narrow Problem Coverage	CoNaLa [151]	Code Generation	Single Answer Bias
RepoBench-P [152]	Code Completion	Limited Language Diversity	Evol-CodeAlpaca [153]	Code Instruction	Limited Domain Coverage
InterCode [154]	Interactive Coding	Limited Language Coverage	GSM8K [155]	Math Reasoning	Limited Problem Variety
M GSM [156]	Math Reasoning	Limited Problemset Coverage	Math-Shepherd [157]	Math Reasoning	Narrow Problem Scope
MultiArith [158]	Math Reasoning	Limited Problem Complexity	MATH [159]	Math Reasoning	Lacks Algorithmic Reasoning Problems
TabMWP [160]	Math Word Problem	Limited Problem Diversity	MoST [161]	Multi-Step Reasoning	Inherent Annotation Noise
STaRK [162]	Reasoning	Limited Human-Query Variety	BridgeData [163]	Reasoning	Limited Task Diversity
Aci-bench [164]	Agent Interaction	Synthetic Data Reliance	HotpotQA [165]	Multi-hop QA	Limited Domain Coverage
MuSiQue [166]	Multi-hop QA	Restricted Question Diversity	StrategyQA [167]	Multi-step QA	Limited Factual Diversity
ArxivQA [168]	Scientific QA	Limited Disciplinary Diversity	QASPER [169]	Scientific QA	Limited Domain Coverage
NarrativeQA [170]	Long-form QA	Limited Training Data	FEVER [171]	Fact Verification	Limited Evidence Scope
QMSum [172]	Query Summarization	Limited Domain Coverage	WikiHow [173]	Summarization	Limited Abstraction
GovReport [174]	Summarization	Limited Domain Representation	BookSum [175]	Long Summarization	Limited Coverage Of Diverse Genres
Stanford Alpaca [176]	Instruction Tuning	Limited Contextual Variation.	OpenAssistant [177]	Instruction Tuning	Limited Domain Coverage
OpenOrca [178]	Instruction Tuning	Limited Domain Coverage	WildChat [179]	Instruction Tuning	Limited Real-World Diversity.
IN3 [63]	Instruction Tuning	Limited Task Diversity	FineWeb [180]	Pre-training Corpus	Limited Content Diversity
RefinedWeb [181]	Pre-training Corpus	Contains Residual Noise	RedPajama [182]	Web Text	Benchmark Contamination Risk
LLMsecEval [183]	Code Security	Limited CWE Coverage	PKU-SafeRLHF [184]	Safety	Limited Scale And Harm Granularity
HarmfulQA [185]	Safety	Synthetic Prompt Bias	HH-RLHF [186]	Alignment	Limited Generalization Scope
BeaverTails [187]	Alignment	Narrow Harm Category	LLM Attacks [188]	Adversarial	Limited Attack Diversity
R2R [189]	Embodied AI	Limited Context Diversity	REVERIE [190]	Embodied AI	Sparse Object Annotations
Touchdown [191]	Embodied AI	Narrow Urban Context	ALFRED [192]	Instruction	Limited Environment Diversity
ALFWorld [193]	Instruction	Limited Task Variety	TravelPlanner [194]	Planning	Low Real-World Complexity.
BlocksWorld [195]	Planning	Lacks Real-World Variability	HaGRID [196]	Gesture Recognition	Lacks Dynamic Gestures
EgoGesture [197]	Gesture Recognition	Limited Gesture Diversity	CogScene [198]	3D Scene Understanding	Lacks Scenario Diversity
WIDER FACE [199]	Face Detection	Demographic Bias	CelebA [200]	Face Attributes	Limited Attribute Diversity
CDSL [201]	Clinical Data	Lacks Temporal Coverage	CCLE [202]	Genomics	Limited Tissue Representation
APARENT2 [130]	Genomics	Limited Task Complexity	Saluki [203]	Genomics	Limited Domain Coverage
SKEMPI [204]	Protein Binding Affinity	Lacks Structural Diversity	PubMed [205]	Scientific Abstracts	Possesses Inconsistent Indexing
GAIA [206]	General AI Assistant	Lacks Language Diversity	BIG-bench [207]	General Evaluation	Limited Task Diversity
MMLU [208]	General Evaluation	Data Contains Cultural Bias	CAMEL [22]	Multi-Agent Convo.	Limited Temporal Coverage
SOTOPIA-PI [93]	Social Simulation	Limited Safety Dimensions	ToolBench [209]	Tool Use	Exhibits Tool Specific Bias
Mind2Web [210]	Web Agent	Evaluation Bias And Limited Scope	RoboNet [211]	Trajectory	Limited Task Modalities

examining the strengths of LLM, these benchmarks also uncover limitations, such as long-term focus issues [40] and social biases [101]. Although datasets in LLM-based autonomous agent research have significantly advanced areas such as reinforcement learning, code generation, healthcare,

and urban planning, they still present major limitations that hinder generalization and real-world applicability. Many existing datasets are built within simulated or synthetic environments, including ALFRED [114], Overcooked AI [25], and Sociodojo [88]. These settings often fail to capture the

inherent noise, unpredictability, and multimodal complexity of real environments, leading to overfitting and poor performance [48, 58, 109]. Robotics and navigation datasets, for instance Street View [55] and AndroidArena [109] often lack diversity in user behavior, scene variation, and environmental dynamics, resulting in biased models with limited robustness [64, 113]. Standard benchmarks, including HumanEval, MBPP, GSM8K, MATH, and WebShop, remain task-specific with limited cross-domain and multilingual coverage [13, 39, 41, 212].

Existing benchmarks and datasets inadequately measure agentic reasoning by testing performance in simplified, static environments that fail to reflect real-world complexity. They often prioritize subjective, language-level evaluation over assessing an agent’s objective, action-level impact on achieving goals [95]. This reliance on metrics like final success rate offers few insights into the actual reasoning process [23] and neglects crucial challenges like navigating dynamic action spaces [109]. Furthermore, these benchmarks often overlook critical real-world constraints, such as temporal dynamics, which are essential for assessing an agent’s ability to plan and multitask efficiently [87].

10 Discussion

Our review explored the rapid use of LLMs as agents and tools for complex autonomous tasks. This study presents a comprehensive examination of existing LLM-based frameworks and discusses cognitive and operational components critical to agentic intelligence. By analyzing a broad spectrum of prior work, we identified how these systems are architected, the capabilities that underpin their autonomy, and the current limitations constraining their scalability. We categorized the most widely adopted methods and their implementation within single-agent and multi-agent systems across various application domains. In general, this review offers a coherent foundation for understanding the developmental trajectory of LLM agents and tools to guide future advancements in this rapidly evolving field.

Baseline LLMs. Recent studies addressing (RQ1) reveal that GPT-4 is currently the dominant foundational model, cited in fifty-five studies, followed by GPT-3.5 in 23 studies, establishing it as a performance benchmark despite cost and access restrictions. GPT variants fulfill three roles: first, as gold-standard ablation references; second, as multi-agent collaborators with open models; and third, as primary reasoning modules within agent stacks.

In addition to OpenAI models, Anthropic’s Claude 3 series is a prominent proprietary alternative [23, 40, 59, 90]. Google’s Gemini excels at very-long-context reasoning for document-scale planning. Meanwhile, the open-source ecosystem is rapidly advancing, reducing the performance gap. Models such as LLaMA-2/3 [56, 58, 71, 83, 124], Mistral [47, 48, 63, 68, 87, 93, 127], Gemini [80, 90], Qwen, DeepSeek,

Vicuna, WizardLM, and GLM-4 are increasingly adopted for coding, planning, and dialogue. Specialized derivatives like CodeLLaMA and ToolLLaMA demonstrate, task-specific pre-training can produce domain expertise without complete retraining. For constrained environments, Phi-2, Phi-3.5 Mini [104], Gemini-2B, and Mistral-7B, are more efficient. Lastly, multimodality through CLIP, Stable Diffusion XL, and VQ-GAN enables LLM agents beyond text to vision and interface automation.

External tool integration in LLMs. We found that integrating external tools (RQ2) is a primary driver of LLM autonomy. LLMs paired with rich computational resources, web APIs, knowledge graphs, code execution, RESTful service libraries [50], and high-fidelity simulators elevate agent capability. Real-time data retrieval from domain-specific sources strengthens RAG accuracy while speech recognition tools, such as Speechly, enable real-world interaction. Furthermore, environments such as AI2-THOR, ALFWorld, and SMAC, connected through platforms like ROS 2 and Gazebo, demonstrate that LLMs can perceive state, reason, and act in dynamic and interactive settings [13, 42, 99, 104, 118]. Collectively, these patterns strongly suggest that external tool access is an essential foundational mechanism for accessing up-to-date knowledge, enabling perception, action, and reasoning in modern LLM agents.

Frameworks for LLM agents. Our analysis of LLM agent frameworks (RQ3) reveals single-agent scenario is dominated by ReAct and Reflection, prioritizing streamlined reasoning-action integration and iterative self-improvement. ReAct’s thought-action sequences enable dynamic task adaptation, while Reflexion refines performance through memory-based learning and feedback [13]. These frameworks excel at tasks that require adaptability, minimal infrastructure, and self-guided correction.

Multi-agent implementations favor frameworks such as AutoGen and CAMEL, effective for interaction, role differentiation, and collective planning [213]. AutoGen’s conversation programming supports asynchronous, modular communication workflows. CAMEL’s role-conditioned framework enables agents to solve tasks collaboratively through structured, multi-turn dialogues. These capabilities are critical for multistep reasoning, distributed system design, or collaborative creativity, where agents operate semi-independently while maintaining group objectives. LangChain is compatible in single and multi-agent setups with chain reasoning steps, memory access, and tool use [35, 112]. MetaGPT [21] integrates software engineering conventions among less prevalent frameworks, while AIDE [86] and BDI [99] focus on interpretability and logical behaviors to enhance explainability and work in rule-constrained domains.

The trend is shifting from instruction-bound architectures to adaptive, role-aware, and coordination-driven systems. The widespread use of ReAct, Reflexion, LangChain, and AutoGen indicates a convergence on reasoning, memory

integration, and collaborative execution as core capacities.

Reasoning, planning, and memory. Integration of reasoning, planning and memory mechanisms (*RQ4*) shaped the development of LLM-agents. During analysis, we found single-agent frameworks like TOOLLLM [50], EASYTOOL [51], and ToTh [47] emphasize autonomy, safety, and introspective capability. In contrast, multi-agent frameworks like AGENTVERSE [34], Thread Memory [68], and ToM [100] prioritize coordination, belief modeling, and contextual reasoning. AGENTVERSE implements collaborative reasoning using structured symbolic modules; Thread Memory preserves context in multi-turn dialogue and interaction; ToM enhances coordination by modeling belief states in multi-agent settings. CoT, ReAct, and Self-Reflection appear in both. CoT is effective in code generation and robotics [39, 100], where ReAct [64, 87] and Reflexion [87, 98] integrate feedback-aware reasoning, contributing to real-time adaptability and iterative self-improvement.

Single agent planning strategies, such as ReHAC [98] and RAFA [129], emphasize proactive multistep forecasting and heuristic goal decomposition. Multi-agent systems such as DeCoAgent [36] and SMART-LLM [59] address coordination through prompt-based decomposition and coalition formation, respectively. In particular, adaptive and evolution-based planning in Star-Agents [42] and UrbanKGent [83] reflect increasing interest in long-horizon self-organizing workflows.

We observe a clear divide between localized and distributed memory. Single-agent models optimize within episode retention using episodic reprompting [56], compressed logs [112], or symbolic graphs [47], supporting isolated task continuity. Multi-agent models like ProAgent [25] provide shared task memory and thread-level history, while MATRIX [94] generates diverse interaction data as a multi-agent simulation environment, rather than managing shared memory. Context windows [65, 125] and dialogue history [22, 93] underpin both categories, while hybrid memory systems [67, 214] signal convergence toward unified architectures that bridge short/long-term retention.

Prompting, fine-tuning, and memory augmentation.

Our findings showed that agent autonomy is enabled by prompt engineering, fine-tuning, and memory augmentation (*RQ5*). Prompt engineering, a non-parametric technique, allows flexible, inference-time control over agent behavior, as seen in MechAgents [33] and AGENTVERSE [34], which used structured prompts to simulate complex interactions such as expert collaboration, role play, trust modeling, and cognitive biases. However, in special domains where prompting is insufficient, fine-tuning is employed as a parametric approach to internalize domain expertise and task-specific skills, as seen in ToolLLM [50] for tool use specialization, RISE for self-correction abilities, and SOTOPIA- π [93] for modeling social intelligence. Likewise, memory augmentation through external RAG and internal episodic memory is

essential to ground agents in real-world contexts and support long-term reasoning. For example, TRIAGEAGENT [37] utilizes medical knowledge retrieval to mitigate hallucination, while Reflexion [13] enables agents to adapt based on accumulated experience. Therefore, the architectural and behavioral autonomy of modern LLM agents is largely scaffolded by design choices rather than innate model capabilities.

Evaluation and benchmarks. Our analysis indicated that evaluating LLM-based agents (*RQ6*) requires a methodological change, from static NLP benchmarks to dynamic process-oriented evaluations. We also found that a robust evaluation framework should incorporate final outputs and the full agentic process. For example, AGENTBOARD [23] introduces "progress rate" metrics that capture incremental task completion across intermediate steps. Similarly, RE-Bench [86] aligns agent performance with human expert benchmarks on complex research problems, reinforcing the importance of human-level baselines in agent evaluation.

In addition to this, controlled simulations reveal handling unreliable or adversarial collaborators [91], highlighting resilience as a critical component of agent intelligence. Domain-specific evaluations also emerged as essential for assessing practicality. For example, ChemCrow [54] benchmarks autonomous chemical synthesis against metrics grounded in scientific outputs, while tool-integrated agents in healthcare significantly reduce clinical calculation errors [53].

Our analysis points to the growing recognition of failure analysis and safety evaluations as a key to agent benchmarking. Adversarial testing techniques are being developed to expose vulnerabilities, including backdoor attacks [89], while new approaches such as PrivacyAsst [90] aim to maintain data confidentiality and trustworthiness during agent deployment. The evaluation state for LLM agents is rapidly evolving towards more comprehensive, interactive, and context-aware approaches. These emerging strategies emphasize whether an agent can succeed and how it reasons, interacts, adapts, and safeguards users along the way.

11 Future directions

LLM-based agents demonstrate significant potential to automate complex tasks, yet their current limitations (see Figure 7) highlight the need for systemic research. To build agents that are reliable, adaptive, and ethically aligned, challenges such as unverifiable reasoning, limited contextual collaboration, vulnerability to adversarial attacks, and inadequate personalization must be addressed.

In this section, we outline the key areas where future work and critical research is needed (*RQ7*). Table S3, in the Supplementary Material, provides an organized summary of the existing research gaps and potential research pathways discussed in this section.

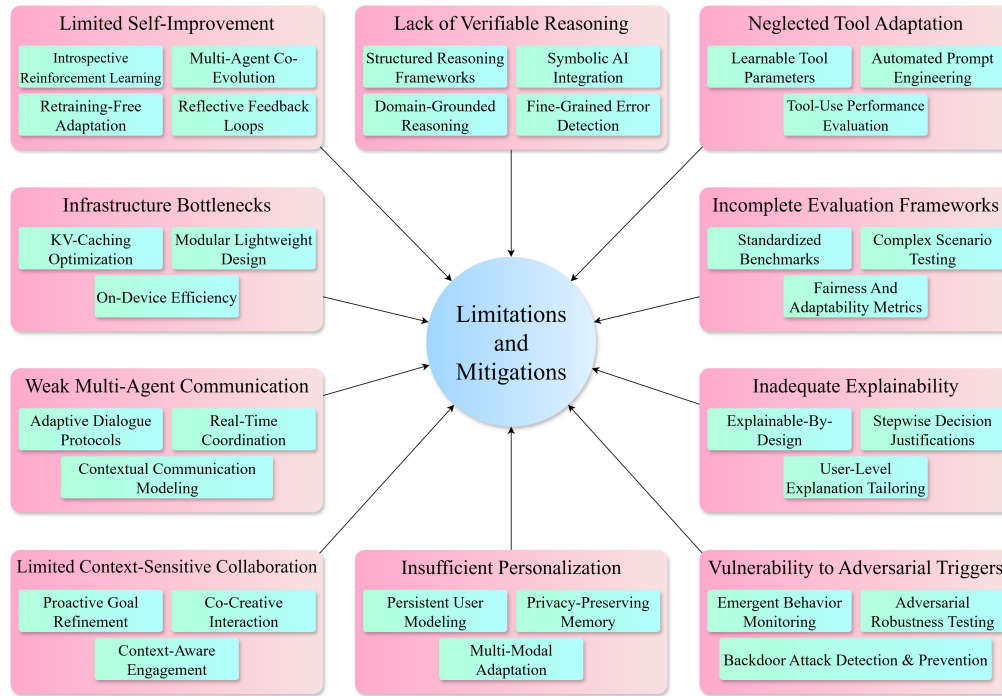


Figure 7: Limitations in the current landscape of LLM-based agents and corresponding mitigation strategies

11.1 Towards verifiable reasoning and robust self-improvement

Although current agents based on LLMs show impressive abilities, their reasoning processes frequently remain unclear, and their capacity to learn from failure is limited. This lack of clarity and inflexibility limits their usage. Future work should develop agents with clear, verifiable reasoning methods and strong, efficient self-improvement cycles.

One major trend in improving reasoning is shifting from unstructured, free-form "CoT" to more organized and logically sound frameworks. Some methods ensure logical consistency from the start. For example, the ToTh model's reasoning is a multi-agent collaboration that simulates abductive, deductive, and inductive inference, assembling outputs into a formal reasoning graph, validating for coherence [47]. In future works, a similar approach could assist in detecting financial fraud or security breaches by producing a reasoning graph that will provide a transparent, stepwise logic trail that authorities can trace and validate. MechAgents [33] utilizes a team of agents for complex engineering challenges and self-correcting physics-based simulation code. This verifies the reasoning of the agent against established scientific models. Building on this strategy, future work may explore the potential drug-drug interactions by modeling their combined molecular effects. Combining LLMs with traditional symbolic AI approaches, such as Belief-Desire-Intention agents, enables verifiable decision-making and improves safety [99]. An essential aspect of verifiability is identifying and address-

ing flaws in the reasoning process. FG-PRM research [212] reliably tackles this by creating a classification of errors in mathematical reasoning and training a reward model to identify and penalize these errors.

Alongside verifiable reasoning is the goal of strong self-improvement, where agents learn from experience without needing retraining. Through strategies like Reflexion, agents can reflect on their performance and learn from mistakes, using verbal RL, creating a memory of these reflections that guides future attempts [13]. Studies such as RISE employ recursive introspection to fine-tune models through an ongoing process of self-correction, to improve their output over time [29]. Subsequent work could apply these strategies to develop an agent for biomedical image segmentation that self-improves iteratively, minimizes reliance on continuous large-scale human annotation, and adapts to patient-specific image differences. Multi-agent dynamics also serves as a robust method for promoting self-improvement. In the CORY framework, a model is split into pioneer and observer agents that co-evolve through a cooperative RL process. This improves resilience and avoids policy failures common in traditional RL fine-tuning [41]. Other frameworks, such as COPPER [107], feature a special reflector agent that provides tailored, constructive feedback to other agents, improving teamwork through a systematic self-improvement loop. Some studies, such as AgentOptimizer [108], suggest treating agent tools as learnable parameters, enabling task adaptation through tool improvement. Likewise, AVATAR [49]

VERIFIABLE REASONING
IS HARD → NEED FRAMEWORKS

automates the optimization of prompts for tool use by employing a comparator module to reason over both positive and negative examples. Integrating structured reasoning, introspective learning, and adaptive capabilities is driving LLM agents toward more reliability, transparency, and continual improvement.

11.2 Towards scalable, adaptive, and collaborative LLM-based agent systems

Future research should enhance LLM agents' scalability, adaptability, and real-time operational capacity, for deployment in multimodal and domain-specific environments [72, 77, 85, 102]. Beyond internal agent reasoning and learning mechanisms, future work should address infrastructure constraints, memory usage, and latency, while enabling efficient inference through architectural techniques such as KV caching and optimized decoders [68, 85]. These improvements are vital for low-resource or time-sensitive scenarios, such as on-device processing or streaming interaction.

Another key priority lies in enhancing multi-agent collaboration and the development of perceptually aware communication. The absence of adaptive communication protocols and contextual modeling in frameworks such as AGENT-VERSE [34] limits their ability in dynamic multi-agent environments [86]. Future agent systems should adopt human-inspired, socially grounded dialogue strategies to enhance collaboration capability, contextual awareness and intent inference among agents [38]. Such improvements will enable the deployment of lightweight agents on edge hardware for traffic systems, warehouses, and drones, facilitating inter-agent communication and collaboration.

Furthermore, as the range of agent applications expands, extending to gesture-based interfaces, EHR prediction, collaborative games, continuous learning, coevolutionary training, and prompt refinement methods become essential [72]. Simultaneously, development of negotiation-heavy simulations like Lewis-style coordination games or Diplomacy can facilitate deeper understanding of agent reasoning under critical settings [86, 93].

Likewise, ensuring robustness and interpretability remains a primary concern. Beyond introspective reasoning and verification, transparent evaluation pipelines are needed to track agent behavior over time, particularly under unpredictability and emergence. The acceleration of open-source advancements, led by models such as LLaMA-2-70B, provides an accessible, scalable, reproducible foundation, enabling researchers to collectively develop safe, effective, socially aligned agentic systems [43, 95].

11.3 Deepening the human-agent symbiosis: personalization, proactivity, and trust

Future research should enhance autonomy, alignment, and practical deployability of LLM-based agents, for complex, dynamic, open-ended, and high-stakes environments. Increased operational freedom produces emergent effects, from strategic cooperative gains to unsafe shortcut-seeking actions. Addressing this duality requires robust alignment strategies to mitigate adverse behaviors while preserving constructive autonomy to operate efficiently and effectively.

Simultaneously, the strategic focus is shifting toward human-centered augmentation, emphasizing collaboration, reliability, and personalization. Realizing this calls for advances in intent recognition and co-creative interaction, where agents can engage in context-sensitive conversations that anticipate user needs and refine task goals interactively, an approach illustrated by studies like "Tell Me More" [63] and ReHAC [98].

Long-term personalization represents a critical research frontier in which future agents should build persistent, evolving user models over time while addressing memory and privacy, as stated in studies like PrivacyAsst [90]. Achieving such personalization requires integrating multimodal capabilities while aligning interface modalities to user-specific preferences. Recently, personalized LLM-based agents have introduced complex ethical and societal risks, anthropomorphism, and over-trust. Their humanlike interactions create deception and encourage vulnerable populations, such as children or the elderly, to disclose sensitive information [215]. Since LLM-based agents lack moral agency, attributing accountability is conceptually flawed [216]. Beyond interface risks, agent architectures suffer from jailbreaking [217], and environmental signal-triggered backdoor attacks [89]. Jailbreaking refers to deliberate overriding of built-in safety measures of LLMs [218], and since agents need to handle multi-round dialogues and multiple sources of information, they become more prone to jailbreaking attacks. Effective human-agent collaboration requires both resilient safeguard frameworks against vulnerabilities and transparent, explainable reasoning [99].

Continuous model optimization across collaborative decoding, lightweight modular execution strategies, and efficient caching mechanisms drives real-world scalability and enables a new class of agents that go beyond technical proficiency to ethically aligned, emotionally aware, privacy-conscious, and deeply integrated in complex workflows.

12 Conclusion

In this article, we present a comprehensive overview of LLM-based agents and the integration of tools within these systems. We examined how prompt engineering, fine-tuning, memory enhancement, and tool use contribute to the building of

LLM agents by addressing seven focused research questions. We observed that single-agent systems prioritize autonomy and introspective decision making. However, multi-agent systems focus on coordination, role distribution, and collaborative planning. Crucially, this distinction becomes domain sensitive, with multi-agent configurations demonstrating pronounced advantages in areas requiring social intelligence, cooperative problem solving, and high-stakes decision support, such as healthcare, scientific research, and complex engineering tasks. Integrating external tools, real-time data sources, and multimodal systems has become essential to enable LLM agents to perform tasks beyond the limitations of pre-trained models. In parallel, the evaluation of these agents is changing from static accuracy-based benchmarks to dynamic process-oriented methods that account for reasoning quality, adaptability, and task completion in real-world settings.

We have also examined critical limitations and safety concerns as LLM agents are deployed in sensitive environments. These include risks related to security, performance limitations, adaptability in dynamic or personalized settings, and challenges in trust, explainability, and agent co-evolution. Addressing these challenges will ensure reliability, trust, and efficiency in future applications.

In our view, future work should focus on two critical goals: making agent reasoning transparent and verifiable and developing reliable methods for self-improvement without compromising safety. These capabilities are critical in high-risk environments, where errors can have serious consequences. Ensuring that LLM agents are trustworthy, resilient, and aligned with domain-specific requirements will be central to their responsible deployment across disciplines.

Declarations

Conflict of interests: On behalf of all authors, the corresponding author states that there is no conflict of interest.

Funding: No external funding is available for this research.

Data availability statement: Not applicable.

Ethics approval and consent to participate. Not applicable.

Informed consents: Not applicable.

Author Contributions: *Conceptualization* Sadia Sultana Chowa, Riasad Alvi, Subhey Sadi Rahman, Mohaimenul Azam Khan Raiaan; *Methodology:* Sadia Sultana Chowa, Riasad Alvi, Subhey Sadi Rahman, *Resources and Literature Review:* Sadia Sultana Chowa, Riasad Alvi, Subhey Sadi Rahman, *Writing – Original Draft Preparation:* Sadia Sultana Chowa, Riasad Alvi, Subhey Sadi Rahman, Md Abdur Rahman, Mohaimenul Azam Khan Raiaan;

Validation: Sami Azam, Mohaimenul Azam Khan Raiaan, Md Abdur Rahman, Md Rafiqul Islam;

Formal Analysis: Sadia Sultana Chowa, Sami Azam, Mohaimenul Azam Khan Raiaan, Md Abdur Rahman;

Writing – Reviewing and Finalization: Mohaimenul Azam Khan Raiaan, Sami Azam, Md Rafiqul Islam, Mukhtar Husain;

Project Supervision: Mohaimenul Azam Khan Raiaan, Sami Azam;

References

- [1] P. Kumar, “Large language models (llms): survey, technical frameworks, and future challenges,” *Artificial Intelligence Review*, vol. 57, no. 10, p. 260, 2024.
- [2] X. Wang, H. Jiang, Y. Yu, J. Yu, Y. Lin *et al.*, “Building intelligence identification system via large language model watermarking: a survey and beyond,” *Artificial Intelligence Review*, vol. 58, no. 8, p. 249, 2025.
- [3] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan *et al.*, “Language models are few-shot learners,” *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.
- [4] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter *et al.*, “Chain-of-thought prompting elicits reasoning in large language models,” in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, ser. NIPS ’22. Red Hook, NY, USA: Curran Associates Inc., 2022.
- [5] O. Press, M. Zhang, S. Min, L. Schmidt, N. Smith, and M. Lewis, “Measuring and narrowing the compositionality gap in language models,” in *Findings of the Association for Computational Linguistics: EMNLP 2023*. Association for Computational Linguistics, 2023, pp. 5687–5711.
- [6] S. Schmidgall, Y. Su, Z. Wang, X. Sun, J. Wu *et al.*, “Agent laboratory: Using llm agents as research assistants,” *arXiv preprint arXiv:2501.04227*, 2025.
- [7] C. Qian, W. Liu, H. Liu, N. Chen, Y. Dang *et al.*, “Chatdev: Communicative agents for software development,” *arXiv preprint arXiv:2307.07924v5*, 2023.
- [8] J. Gottweis, W.-H. Weng, A. Daryin, T. Tu, A. Palepu *et al.*, “Towards an ai co-scientist,” *arXiv preprint arXiv:2502.18864*, 2025.
- [9] J. Chen, C. Yu, X. Zhou, T. Xu, Y. Mu *et al.*, “Emos: Embodiment-aware heterogeneous multi-robot operating system with llm agents,” *arXiv preprint arXiv:2410.22662v2*, 2025.

- [10] X. Chen, J. Xiang, S. Lu, Y. Liu, M. He, and D. Shi, “Evaluating large language models and agents in healthcare: key challenges in clinical applications,” *Intelligent Medicine*, vol. 5, no. 2, pp. 151–163, 2025.
- [11] S. Mao, Y. Cai, Y. Xia, W. Wu, X. Wang *et al.*, “Alympics: Llm agents meet game theory – exploring strategic decision-making with ai agents,” *arXiv preprint arXiv:2311.03220*, 2023.
- [12] 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*, 2023, pp. 1–22.
- [13] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao, “Reflexion: Language agents with verbal reinforcement learning,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 8634–8652, 2023.
- [14] T. Schick, J. Dwivedi-Yu, R. Dessi, R. Raileanu, M. Lomeli *et al.*, “Toolformer: Language models can teach themselves to use tools,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 68 539–68 551, 2023.
- [15] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran *et al.*, “React: Synergizing reasoning and acting in language models,” in *International Conference on Learning Representations (ICLR)*, 2023.
- [16] C. Sun, S. Huang, and D. Pompili, “Llm-based multi-agent decision-making: Challenges and future directions,” *IEEE Robotics and Automation Letters*, vol. 10, no. 6, pp. 5681–5688, 2025.
- [17] J. Gao, Y. Zhang, Y. Chen, Y. Dong, Y. Chen *et al.*, “Agent-in-the-loop to distill expert knowledge into artificial intelligence models: a survey,” *Artificial Intelligence Review*, vol. 58, no. 9, p. 266, 2025.
- [18] Y. Talebirad and A. Nadiri, “Multi-agent collaboration: Harnessing the power of intelligent llm agents,” *arXiv preprint arXiv:2306.03314*, 2023.
- [19] J. Zhang, X. Xu, N. Zhang, R. Liu, B. Hooi, and S. Deng, “Exploring collaboration mechanisms for LLM agents: A social psychology view,” in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2024, pp. 14 544–14 607.
- [20] S. Han, Q. Zhang, Y. Yao, W. Jin, and Z. Xu, “Llm multi-agent systems: Challenges and open problems,” *arXiv preprint arXiv:2402.03578v2*, 2025.
- [21] S. Hong, M. Zhuge, J. Chen, X. Zheng, Y. Cheng *et al.*, “Metagpt: Meta programming for a multi-agent collaborative framework,” in *The Twelfth International Conference on Learning Representations*, 2023.
- [22] 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*, vol. 36, pp. 51 991–52 008, 2023.
- [23] M. Chang, J. Zhang, Z. Zhu, C. Yang, Y. Yang, Y. Jin, Z. Lan, L. Kong, and J. He, “Agentboard: An analytical evaluation board of multi-turn llm agents,” *Advances in neural information processing systems*, vol. 37, pp. 74 325–74 362, 2024.
- [24] S. Qiao, N. Zhang, R. Fang, Y. Luo, W. Zhou *et al.*, “Autoact: Automatic agent learning from scratch for qa via self-planning,” 2024.
- [25] C. Zhang, K. Yang, S. Hu, Z. Wang, G. Li *et al.*, “Proagent: building proactive cooperative agents with large language models,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 16, 2024, pp. 17 591–17 599.
- [26] M. A. Ferrag, N. Tihanyi, and M. Debbah, “From llm reasoning to autonomous ai agents: A comprehensive review,” *arXiv preprint arXiv:2504.19678*, 2025.
- [27] X. Li, “A review of prominent paradigms for llm-based agents: Tool use, planning (including rag), and feedback learning,” in *Proceedings of the 31st International Conference on Computational Linguistics*, 2025, pp. 9760–9779.
- [28] W. Xu, C. Huang, S. Gao *et al.*, “Llm-based agents for tool learning: A survey,” *Data Science and Engineering*, pp. 1–31, 2025.
- [29] Z. Xi, W. Chen, X. Guo *et al.*, “The rise and potential of large language model based agents: a survey,” *Science China Information Sciences*, vol. 68, p. 121101, 2025.
- [30] L. Wang, C. Ma, X. Feng *et al.*, “A survey on large language model based autonomous agents,” *Frontiers of Computer Science*, vol. 18, p. 186345, 2024.
- [31] T. Guo, X. Chen, Y. Wang, R. Chang, S. Pei *et al.*, “Large language model based multi-agents: A survey of progress and challenges,” *arXiv preprint arXiv:2402.01680*, 2024.
- [32] Y. Cheng, C. Zhang, Z. Zhang, X. Meng, S. Hong *et al.*, “Exploring large language model based intelligent agents: Definitions, methods, and prospects,” *arXiv preprint arXiv:2401.03428*, 2024.

- [33] B. Ni and M. J. Buehler, “Mechagents: Large language model multi-agent collaborations can solve mechanics problems, generate new data, and integrate knowledge,” *Extreme Mechanics Letters*, vol. 67, p. 102131, 2024.
- [34] W. Chen, Y. Su, J. Zuo, C. Yang, C. Yuan *et al.*, “Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors in agents,” 2024.
- [35] L. H. Klein, N. Potamitis, R. Aydin, R. West, C. Gulcehre, and A. Arora, “Fleet of agents: Coordinated problem solving with large language models,” in *Forty-second International Conference on Machine Learning*, 2025.
- [36] A. Jin, Y. Ye, B. Lee, and Y. Qiao, “Decoagent: Large language model empowered decentralized autonomous collaboration agents based on smart contracts,” *IEEE Access*, 2024.
- [37] M. Lu, B. Ho, D. Ren, and X. Wang, “Triageagent: Towards better multi-agents collaborations for large language model-based clinical triage,” in *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024, pp. 5747–5764.
- [38] Y. Kim, C. Park, H. Jeong, Y. S. Chan, X. Xu *et al.*, “Mdagents: An adaptive collaboration of llms for medical decision-making,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 79 410–79 452, 2024.
- [39] X. Bai, S. Huang, C. Wei, and R. Wang, “Collaboration between intelligent agents and large language models: A novel approach for enhancing code generation capability,” *Expert Systems with Applications*, vol. 269, p. 126357, 2025.
- [40] 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*, vol. 37, pp. 132 208–132 237, 2024.
- [41] H. Ma, T. Hu, Z. Pu, L. Boyin, X. Ai *et al.*, “Coevolving with the other you: Fine-tuning LLM with sequential cooperative multi-agent reinforcement learning,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 15 497–15 525, 2024.
- [42] H. Zhou, Y. Tang, H. Qin, Y. Yang, R. Jin *et al.*, “Star-agents: Automatic data optimization with llm agents for instruction tuning,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 4575–4597, 2024.
- [43] D. Yin, F. Brahman, A. Ravichander, K. Chandu, K.-W. Chang *et al.*, “Agent lumos: Unified and modular training for open-source language agents,” in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2024, pp. 12 380–12 403.
- [44] Y. Xu, H. Su, C. Xing, B. Mi, Q. Liu *et al.*, “Lemur: Harmonizing natural language and code for language agents,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [45] F. Zhu and R. Simmons, “Bootstrapping cognitive agents with a large language model,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 1, 2024, pp. 655–663.
- [46] S. Jorgensen, G. Nadizar, G. Pietropolli, L. Manzoni, E. Medvet *et al.*, “Large language model-based test case generation for gp agents,” in *Proceedings of the genetic and evolutionary computation conference*, 2024, pp. 914–923.
- [47] S. Abdaljalil, H. Kurban, K. Qaraqe, and E. Serpedin, “Theorem-of-thought: A multi-agent framework for abductive, deductive, and inductive reasoning in language models,” in *Proceedings of the 3rd Workshop on Towards Knowledgeable Foundation Models (KnowFM)*. Vienna, Austria: Association for Computational Linguistics, 2025, pp. 111–119.
- [48] S. Qiao, R. Fang, N. Zhang, Y. Zhu, X. Chen *et al.*, “Agent planning with world knowledge model,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 114 843–114 871, 2024.
- [49] S. Wu, S. Zhao, Q. Huang, K. Huang, M. Yasunaga *et al.*, “Avatar: Optimizing llm agents for tool usage via contrastive reasoning,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 25 981–26 010, 2024.
- [50] Y. Qin, S. Liang, Y. Ye, K. Zhu, L. Yan *et al.*, “Tool-LLM: Facilitating large language models to master 16000+ real-world APIs,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [51] S. Yuan, K. Song, J. Chen, X. Tan, Y. Shen *et al.*, “EASYTOOL: Enhancing LLM-based agents with concise tool instruction,” in *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Association for Computational Linguistics, 2025, pp. 951–972.
- [52] Z. Wang, Z. Liu, Y. Zhang, A. Zhong, J. Wang *et al.*, “Rcagent: Cloud root cause analysis by autonomous agents with tool-augmented large language models,” in

Proceedings of the 33rd ACM International Conference on Information and Knowledge Management, 2024, pp. 4966–4974.

- [53] A. J. Goodell, S. N. Chu, D. Rouholiman, and L. F. Chu, “Large language model agents can use tools to perform clinical calculations,” *npj Digital Medicine*, vol. 8, no. 1, p. 163, 2025.
- [54] A. M. Bran, S. Cox, O. Schilter, C. Baldassari, A. D. White, and P. Schwaller, “Augmenting large language models with chemistry tools,” *Nature Machine Intelligence*, vol. 6, no. 5, pp. 525–535, 2024.
- [55] R. Schumann, W. Zhu, W. Feng, T.-J. Fu, S. Riezler, and W. Y. Wang, “Velma: Verbalization embodiment of llm agents for vision and language navigation in street view,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 17, 2024, pp. 18924–18933.
- [56] S. Zheng, jiazheng liu, Y. Feng, and Z. Lu, “Steve-eye: Equipping LLM-based embodied agents with visual perception in open worlds,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [57] W. Hong, W. Wang, Q. Lv, J. Xu, W. Yu *et al.*, “Cogagent: A visual language model for gui agents,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 14281–14290.
- [58] H. Zhang, W. Du, J. Shan, Q. Zhou, Y. Du *et al.*, “Building cooperative embodied agents modularly with large language models,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [59] S. S. Kannan, V. L. Venkatesh, and B.-C. Min, “Smart-llm: Smart multi-agent robot task planning using large language models,” in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 12140–12147.
- [60] Z. Wang, S. Cai, Z. Mu, H. Lin, C. Zhang *et al.*, “Omnijarvis: Unified vision-language-action tokenization enables open-world instruction following agents,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 73278–73308, 2024.
- [61] R. Zhang, H. Du, Y. Liu, D. Niyato, J. Kang *et al.*, “Generative ai agents with large language model for satellite networks via a mixture of experts transmission,” *IEEE Journal on Selected Areas in Communications*, 2024.
- [62] S. Zhang, Y. Dong, Y. Zhang, T. R. Payne, and J. Zhang, “Large language model assisted multi-agent dialogue for ontology alignment,” in *Proceedings of the*

23rd International Conference on Autonomous Agents and Multiagent Systems, 2024, pp. 2594–2596.

- [63] C. Qian, B. He, Z. Zhuang, J. Deng, Y. Qin *et al.*, “Tell me more! towards implicit user intention understanding of language model driven agents,” in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2024, pp. 1088–1113.
- [64] H. Singh, N. Verma, Y. Wang, M. Bharadwaj, H. Fashandi *et al.*, “Personal large language model agents: A case study on tailored travel planning,” in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, 2024, pp. 486–514.
- [65] D. Zhang, L. Chen, S. Zhang, H. Xu, Z. Zhao, and K. Yu, “Large language models are semi-parametric reinforcement learning agents,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 78227–78239, 2023.
- [66] A. Ghafarollahi and M. J. Buehler, “Protagents: protein discovery via large language model multi-agent collaborations combining physics and machine learning,” *Digital Discovery*, vol. 3, no. 7, pp. 1389–1409, 2024.
- [67] D. A. Boiko, R. MacKnight, B. Kline, and G. Gomes, “Autonomous chemical research with large language models,” *Nature*, vol. 624, no. 7992, pp. 570–578, 2023.
- [68] Y. Jin and J. Ma, “Large language model as parking planning agent in the context of mixed period of autonomous vehicles and human-driven vehicles,” *Sustainable Cities and Society*, vol. 117, p. 105940, 2024.
- [69] Y. Xia, Z. Xiao, N. Jazdi, and M. Weyrich, “Generation of asset administration shell with large language model agents: Toward semantic interoperability in digital twins in the context of industry 4.0,” *IEEE Access*, vol. 12, pp. 84863–84877, 2024.
- [70] M. Xu, D. Niyato, J. Kang, Z. Xiong, S. Mao *et al.*, “When large language model agents meet 6g networks: Perception, grounding, and alignment,” *IEEE Wireless Communications*, 2024.
- [71] H. Wu, Z. He, X. Zhang, X. Yao, S. Zheng *et al.*, “Chateda: A large language model powered autonomous agent for eda,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 43, no. 10, pp. 3184–3197, 2024.

- [72] Z. Wang, Y. Zhu, H. Zhao, X. Zheng, D. Sui *et al.*, “Colacare: Enhancing electronic health record modeling through large language model-driven multi-agent collaboration,” in *Proceedings of the ACM on Web Conference 2025*, 2025, pp. 2250–2261.
- [73] D. Ferber, O. S. El Nahhas, G. Wölflein, I. C. Wiest, J. Clusmann *et al.*, “Development and validation of an autonomous artificial intelligence agent for clinical decision-making in oncology,” *Nature cancer*, pp. 1–13, 2025.
- [74] X. Chen, H. Yi, M. You, W. Liu, L. Wang *et al.*, “Enhancing diagnostic capability with multi-agents conversational large language models,” *NPJ digital medicine*, vol. 8, no. 1, p. 159, 2025.
- [75] 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*, vol. 65, no. 1, pp. 62–70, 2024.
- [76] W. Liu, J. Li, Y. Tang, Y. Zhao, C. Liu *et al.*, “Dr-bioright 2.0: an llm-powered bioinformatics chatbot for large-scale cancer functional proteomics analysis,” *Nature communications*, vol. 16, no. 1, p. 2256, 2025.
- [77] Z. Yang, X. Xu, B. Yao, E. Rogers, S. Zhang *et al.*, “Talk2care: An llm-based voice assistant for communication between healthcare providers and older adults,” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 8, no. 2, pp. 1–35, 2024.
- [78] E. Jo, D. A. Epstein, H. Jung, and Y.-H. Kim, “Understanding the benefits and challenges of deploying conversational ai leveraging large language models for public health intervention,” in *Proceedings of the 2023 CHI conference on human factors in computing systems*, 2023, pp. 1–16.
- [79] M. Abbasian, I. Azimi, A. M. Rahmani, and R. Jain, “Conversational health agents: a personalized large language model-powered agent framework,” *JAMIA Open*, vol. 8, no. 4, p. ooaf067, 2025.
- [80] F. L. Barra, G. Rodella, A. Costa, A. Scalogna, L. Carenzo *et al.*, “From prompt to platform: an agentic ai workflow for healthcare simulation scenario design,” *Advances in Simulation*, vol. 10, no. 1, p. 29, 2025.
- [81] N. Li, C. Gao, M. Li, Y. Li, and Q. Liao, “EconAgent: Large language model-empowered agents for simulating macroeconomic activities,” in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2024, pp. 15 523–15 536.
- [82] Q. Zhao, J. Wang, Y. Zhang, Y. Jin, K. Zhu *et al.*, “CompeteAI: Understanding the competition dynamics of large language model-based agents,” in *Proceedings of the 41st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 235. PMLR, 2024, pp. 61 092–61 107.
- [83] Y. Ning and H. Liu, “Urbankgent: A unified large language model agent framework for urban knowledge graph construction,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 123 127–123 154, 2024.
- [84] X. Huang, J. Lian, Y. Lei, J. Yao, D. Lian, and X. Xie, “Recommender ai agent: Integrating large language models for interactive recommendations,” *ACM Transactions on Information Systems*, vol. 43, no. 4, pp. 1–33, 2025.
- [85] X. Zeng, X. Wang, T. Zhang, C. Yu, S. Zhao, and Y. Chen, “Gesturegpt: Toward zero-shot free-form hand gesture understanding with large language model agents,” *Proceedings of the ACM on Human-Computer Interaction*, vol. 8, no. ISS, pp. 462–499, 2024.
- [86] H. Wijk, T. Lin, J. Becker, S. Jawhar, N. Parikh *et al.*, “RE-bench: Evaluating frontier AI r&d capabilities of language model agents against human experts,” in *Forty-second International Conference on Machine Learning*, 2025.
- [87] Y. Zhang, S. Yuan, C. Hu, K. Richardson, Y. Xiao, and J. Chen, “TimeArena: Shaping efficient multitasking language agents in a time-aware simulation,” in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2024, pp. 3894–3916.
- [88] J. Cheng and P. Chin, “Sociodojo: Building lifelong analytical agents with real-world text and time series,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [89] W. Yang, X. Bi, Y. Lin, S. Chen, J. Zhou, and X. Sun, “Watch out for your agents! investigating backdoor threats to llm-based agents,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 100 938–100 964, 2024.
- [90] X. Zhang, H. Xu, Z. Ba, Z. Wang, Y. Hong *et al.*, “Privacyasst: Safeguarding user privacy in tool-using large language model agents,” *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 6, pp. 5242–5258, 2024.
- [91] J. tse Huang, J. Zhou, T. Jin, X. Zhou, Z. Chen *et al.*, “On the resilience of LLM-based multi-agent collaboration with faulty agents,” in *Forty-second International Conference on Machine Learning*, 2025.

- [92] Z. Xiang, L. Zheng, Y. Li, J. Hong, Q. Li *et al.*, “Guardagent: Safeguard LLM agents via knowledge-enabled reasoning,” in *ICML 2025 Workshop on Computer Use Agents*, 2025.
- [93] R. Wang, H. Yu, W. Zhang, Z. Qi, M. Sap *et al.*, “SOTOPIA- π : Interactive learning of socially intelligent language agents,” in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2024, pp. 12912–12940.
- [94] X. Pang, S. Tang, R. Ye, Y. Xiong, B. Zhang *et al.*, “Self-alignment of large language models via monopolylogue-based social scene simulation,” in *Proceedings of the 41st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 235. PMLR, 2024, pp. 39416–39447.
- [95] X. L. Li, N. Chowdhury, D. D. Johnson, T. Hashimoto, P. Liang *et al.*, “Eliciting language model behaviors with investigator agents,” in *Forty-second International Conference on Machine Learning*, 2025.
- [96] Y. Fu, D.-K. Kim, J. Kim, S. Sohn, L. Logeswaran *et al.*, “Autoguide: Automated generation and selection of context-aware guidelines for large language model agents,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 119919–119948, 2024.
- [97] S. George, C. Sypherd, and D. Cashman, “Probing the capacity of language model agents to operationalize disparate experiential context despite distraction,” in *Findings of the Association for Computational Linguistics: EMNLP 2024*. Miami, Florida, USA: Association for Computational Linguistics, 2024, pp. 15447–15459.
- [98] X. Feng, Z.-Y. Chen, Y. Qin, Y. Lin, X. Chen *et al.*, “Large language model-based human-agent collaboration for complex task solving,” in *Findings of the Association for Computational Linguistics: EMNLP 2024*. Association for Computational Linguistics, 2024, pp. 1336–1357.
- [99] L. Frering, G. Steinbauer-Wagner, and A. Holzinger, “Integrating belief-desire-intention agents with large language models for reliable human–robot interaction and explainable artificial intelligence,” *Engineering Applications of Artificial Intelligence*, vol. 141, p. 109771, 2025.
- [100] H. Li, Y. Chong, S. Stepputtis, J. Campbell, D. Hughes *et al.*, “Theory of mind for multi-agent collaboration via large language models,” in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2023, pp. 180–192.
- [101] X. Liu, J. ZHANG, H. Shang, S. Guo, Y. Chengxu, and Q. Zhu, “Exploring prosocial irrationality for LLM agents: A social cognition view,” in *The Thirteenth International Conference on Learning Representations*, 2025.
- [102] L. Wang, J. Zhang, H. Yang, Z.-Y. Chen, J. Tang *et al.*, “User behavior simulation with large language model-based agents,” *ACM Transactions on Information Systems*, vol. 43, no. 2, pp. 1–37, 2025.
- [103] N. Behari, E. Zhang, Y. Zhao, A. Taneja, D. Nagaraj, and M. Tambe, “A decision-language model (dlm) for dynamic restless multi-armed bandit tasks in public health,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 3964–4002, 2024.
- [104] Z. Li, R. Zhang, Z. Wang, Z. Xie, and Y. Song, “Llm-guided decision-making toolkit for multi-agent reinforcement learning,” *Neurocomputing*, vol. 638, p. 130105, 2025.
- [105] Y. Zhang, Q. Jiang, X. XingyuHan, N. Chen, Y. Yang, and K. Ren, “Benchmarking data science agents,” in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2024, pp. 5677–5700.
- [106] Y. Ruan, H. Dong, A. Wang, S. Pitis, Y. Zhou *et al.*, “Identifying the risks of LM agents with an LM-emulated sandbox,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [107] X. Bo, Z. Zhang, Q. Dai, X. Feng, L. Wang *et al.*, “Reflective multi-agent collaboration based on large language models,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 138595–138631, 2024.
- [108] S. Zhang, J. Zhang, J. Liu, L. Song, C. Wang *et al.*, “Offline training of language model agents with functions as learnable weights,” in *Proceedings of the 41st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 235. PMLR, 21–27 Jul 2024, pp. 60315–60335.
- [109] M. Xing, R. Zhang, H. Xue, Q. Chen, F. Yang, and Z. Xiao, “Understanding the weakness of large language model agents within a complex android environment,” in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 6061–6072.
- [110] A. Cuadra, J. Breuch, S. Estrada, D. Ihim, I. Hung *et al.*, “Digital forms for all: A holistic multimodal large language model agent for health data entry,” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 8, no. 2, pp. 1–39, 2024.

- [111] D. Arumugam and T. L. Griffiths, “Toward efficient exploration by large language model agents,” in *The Exploration in AI Today Workshop at ICML 2025*, 2025.
- [112] Q. Huang, J. Vora, P. Liang, and J. Leskovec, “Benchmarking large language models as ai research agents,” in *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023.
- [113] J. Yang, X. Chen, S. Qian, N. Madaan, M. Iyengar *et al.*, “Llm-grounder: Open-vocabulary 3d visual grounding with large language model as an agent,” in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 7694–7701.
- [114] C. H. Song, J. Wu, C. Washington, B. M. Sadler, W.-L. Chao, and Y. Su, “Llm-planner: Few-shot grounded planning for embodied agents with large language models,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 2998–3009.
- [115] R. Feldt, S. Kang, J. Yoon, and S. Yoo, “Towards autonomous testing agents via conversational large language models,” in *2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 2023, pp. 1688–1693.
- [116] Y. Wei, Z. Wang, Y. Lu, C. Xu, C. Liu *et al.*, “Editable scene simulation for autonomous driving via collaborative llm-agents,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 15 077–15 087.
- [117] X. Xue, Z. Lu, D. Huang, Z. Wang, W. Ouyang, and L. Bai, “Comfybench: Benchmarking llm-based agents in comfyui for autonomously designing collaborative ai systems,” in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 24 614–24 624.
- [118] Z. Wang, S. Cai, G. Chen, A. Liu, X. Ma *et al.*, “Describe, explain, plan and select: interactive planning with large language models enables open-world multi-task agents,” in *Proceedings of the 37th International Conference on Neural Information Processing Systems*, ser. NIPS ’23. Red Hook, NY, USA: Curran Associates Inc., 2023.
- [119] Y. Yang, T. Zhou, K. Li, D. Tao, L. Li *et al.*, “Embodied multi-modal agent trained by an llm from a parallel textworld,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024, pp. 26 275–26 285.
- [120] Y. Cai, X. Li, M. Goswami, M. Wiliński, G. Welter, and A. Dubrawski, “LLM agents struggle at time series machine learning engineering,” in *1st ICML Workshop on Foundation Models for Structured Data*, 2025.
- [121] V. Muthusamy, Y. Rizk, K. Kate, P. Venkateswaran *et al.*, “Towards large language model-based personal agents in the enterprise: Current trends and open problems,” in *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023, pp. 6909–6921.
- [122] A. Lampinen, S. Chan, I. Dasgupta, A. Nam, and J. Wang, “Passive learning of active causal strategies in agents and language models,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 1283–1297, 2023.
- [123] Y. Su, D. Yang, S. Yao, and T. Yu, “Language agents: Foundations, prospects, and risks,” in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*. Miami, Florida, USA: Association for Computational Linguistics, 2024, pp. 17–24.
- [124] C. Xie, C. Chen, F. Jia, Z. Ye, S. Lai *et al.*, “Can large language model agents simulate human trust behavior?” *Advances in neural information processing systems*, vol. 37, pp. 15 674–15 729, 2024.
- [125] J.-C. Pang, S.-H. Yang, K. Li, J. Zhang, X.-H. Chen *et al.*, “Kalm: Knowledgeable agents by offline reinforcement learning from large language model rollouts,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 126 620–126 652, 2024.
- [126] N. Hemken, S. Koneru, F. Jacob, H. Hartenstein, and J. Niehues, “Can a large language model keep my secrets? a study on LLM-controlled agents,” in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*. Association for Computational Linguistics, 2025, pp. 746–759.
- [127] S. Zhai, H. Bai, Z. Lin, J. Pan, P. Tong *et al.*, “Fine-tuning large vision-language models as decision-making agents via reinforcement learning,” *Advances in neural information processing systems*, vol. 37, pp. 110 935–110 971, 2024.
- [128] E. Tennant, S. Hailes, and M. Musolesi, “Moral alignment for LLM agents,” in *The Thirteenth International Conference on Learning Representations*, 2025.
- [129] Z. Liu, H. Hu, S. Zhang, H. Guo, S. Ke *et al.*, “Reason for future, act for now: A principled architecture for autonomous LLM agents,” in *Proceedings of the 41st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 235. PMLR, 2024, pp. 31 186–31 261.
- [130] B. P. de Almeida, G. Richard, H. Dalla-Torre, C. Blum, L. Hexemer *et al.*, “A multimodal conversational agent

- for dna, rna and protein tasks,” *Nature Machine Intelligence*, pp. 1–14, 2025.
- [131] M. C. Ramos, C. J. Collison, and A. D. White, “A review of large language models and autonomous agents in chemistry,” *Chemical science*, 2025.
- [132] R. Liu, R. Yang, C. Jia, G. Zhang, D. Yang, and S. Vosoughi, “Training socially aligned language models on simulated social interactions,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [133] Z. Wang, G. Zhang, K. Yang, N. Shi, W. Zhou *et al.*, “Interactive natural language processing,” *arXiv preprint arXiv:2305.13246*, 2023.
- [134] G. Franceschelli and M. Musolesi, “On the creativity of large language models,” *AI & society*, vol. 40, no. 5, pp. 3785–3795, 2025.
- [135] Y. Qin, S. Hu, Y. Lin, W. Chen, N. Ding *et al.*, “Tool learning with foundation models,” *ACM Computing Surveys*, vol. 57, no. 4, pp. 1–40, 2024.
- [136] D. Driess, F. Xia, M. S. M. Sajjadi, C. Lynch, A. Chowdhery *et al.*, “Palm-e: An embodied multimodal language model,” *arXiv preprint arXiv:2303.03378*, 2023.
- [137] J. Kang, R. Laroche, X. Yuan, A. Trischler, X. Liu, and J. Fu, “Think before you act: Decision transformers with working memory,” *arXiv preprint arXiv:2305.16338v3*, 2024.
- [138] Y. Dong, X. Jiang, Z. Jin, and G. Li, “Self-collaboration code generation via chatgpt,” *ACM Transactions on Software Engineering and Methodology*, vol. 33, no. 7, pp. 1–38, 2024.
- [139] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths *et al.*, “Tree of thoughts: Deliberate problem solving with large language models,” *Advances in neural information processing systems*, vol. 36, pp. 11 809–11 822, 2023.
- [140] J. Luo, W. Zhang, Y. Yuan, Y. Zhao, J. Yang *et al.*, “Large language model agent: A survey on methodology, applications and challenges,” *arXiv preprint arXiv:2503.21460*, 2025.
- [141] X. Liu, H. Yu, H. Zhang, Y. Xu, X. Lei *et al.*, “Agent-bench: Evaluating LLMs as agents,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [142] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao *et al.*, “Voyager: An open-ended embodied agent with large language models,” *arXiv preprint arXiv:2305.16291*, 2023.
- [143] W. Xu, K. Mei, H. Gao, J. Tan, Z. Liang, and Y. Zhang, “A-mem: Agentic memory for llm agents,” *arXiv preprint arXiv:2502.12110*, 2025.
- [144] Q. Zhang, Y. Hu, J. Yan, H. Zhang, X. Xie *et al.*, “Large-language-model-based ai agent for organic semiconductor device research,” *Advanced Materials*, vol. 36, no. 32, p. 2405163, 2024.
- [145] M. A. Khan, M. Amani, S. Das, B. Ghosh, Q. Wu *et al.*, “In agents we trust, but who do agents trust? latent source preferences steer LLM generations,” in *ICML 2025 Workshop on Reliable and Responsible Foundation Models*, 2025.
- [146] Y. Qu, T. Zhang, N. Garg, and A. Kumar, “Recursive introspection: Teaching language model agents how to self-improve,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 55 249–55 285, 2024.
- [147] A. Perincherry, J. Krantz, and S. Lee, “Do visual imaginations improve vision-and-language navigation agents?” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2025, pp. 3846–3855.
- [148] M. Chen, J. Tworek, H. Jun, Q. Yuan *et al.*, “Evaluating large language models trained on code,” *arXiv preprint arXiv:2107.03374*, 2021.
- [149] J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski *et al.*, “Program synthesis with large language models,” *arXiv preprint arXiv:2108.07732*, 2021.
- [150] Y. Xia, W. Shen, Y. Wang, J. K. Liu, H. Sun *et al.*, “Leetcodedataset: A temporal dataset for robust evaluation and efficient training of code llms,” *arXiv preprint arXiv:2504.14655*, 2025.
- [151] P. Yin, B. Deng, E. Chen, B. Vasilescu, and G. Neubig, “Learning to mine aligned code and natural language pairs from stack overflow,” in *2018 IEEE/ACM 15th international conference on mining software repositories (MSR)*. IEEE, 2018, pp. 476–486.
- [152] T. Liu, C. Xu, and J. McAuley, “Repobench: Benchmarking repository-level code auto-completion systems,” *arXiv preprint arXiv:2306.03091*, 2023.
- [153] Z. Luo, C. Xu, P. Zhao, Q. Sun, X. Geng *et al.*, “Wizardcoder: Empowering code large language models with evol-instruct,” *arXiv preprint arXiv:2306.08568*, 2023.
- [154] J. Yang, A. Prabhakar, K. Narasimhan, and S. Yao, “Intercode: Standardizing and benchmarking interactive coding with execution feedback,” *Advances in*

- Neural Information Processing Systems*, vol. 36, pp. 23 826–23 854, 2023.
- [155] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun *et al.*, “Training verifiers to solve math word problems,” *arXiv preprint arXiv:2110.14168*, 2021.
- [156] F. Shi, M. Suzgun, M. Freitag, X. Wang, S. Srivats *et al.*, “Language models are multilingual chain-of-thought reasoners,” *arXiv preprint arXiv:2210.03057*, 2022.
- [157] P. Wang, L. Li, Z. Shao, R. Xu, D. Dai *et al.*, “Math-shepherd: Verify and reinforce llms step-by-step without human annotations,” *arXiv preprint arXiv:2312.08935v3*, 2023.
- [158] “Multiarith dataset,” *Kaggle dataset*, 2023, accessed: 2025-08-22. [Online]. Available: <https://www.kaggle.com/datasets/dschettler8845/multiarith-dataset>
- [159] Saxton, Grefenstette, Hill, and Kohli, “Analysing mathematical reasoning abilities of neural models,” *arXiv:1904.01557*, 2019.
- [160] P. Lu, L. Qiu, K.-W. Chang, Y. N. Wu, S.-C. Zhu *et al.*, “Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning,” in *International Conference on Learning Representations (ICLR)*, 2023.
- [161] L. Codeca and J. Härrri, “Monaco sumo traffic (most) scenario: A 3d mobility scenario for cooperative its,” *EPiC Series in Engineering*, vol. 2, pp. 43–55, 2018.
- [162] S. Wu, S. Zhao, M. Yasunaga, K. Huang, K. Cao *et al.*, “Stark: Benchmarking llm retrieval on textual and relational knowledge bases,” in *NeurIPS Datasets and Benchmarks Track*, 2024.
- [163] H. R. Walke, K. Black, T. Z. Zhao, Q. Vuong, C. Zheng *et al.*, “Bridgedata v2: A dataset for robot learning at scale,” in *Conference on Robot Learning*. PMLR, 2023, pp. 1723–1736.
- [164] W.-w. Yim, Y. Fu, A. Ben Abacha, N. Snider, T. Lin, and M. Yetisgen, “aci-bench-corpus.zip,” Figshare. Dataset, 2023. [Online]. Available: <https://doi.org/10.6084/m9.figshare.22494601.v1>
- [165] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. Cohen *et al.*, “HotpotQA: A dataset for diverse, explainable multi-hop question answering,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018, pp. 2369–2380.
- [166] H. Trivedi, N. Balasubramanian, T. Khot, and A. Sabharwal, “Musique: Multihop questions via single-hop question composition,” *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 539–554, 2022.
- [167] M. Geva, D. Khashabi, E. Segal, T. Khot, D. Roth, and J. Berant, “Did Aristotle Use a Laptop? A Question Answering Benchmark with Implicit Reasoning Strategies,” *Transactions of the Association for Computational Linguistics (TACL)*, 2021.
- [168] L. Li, Y. Wang, R. Xu, P. Wang, X. Feng *et al.*, “Multimodal ArXiv: A dataset for improving scientific comprehension of large vision-language models,” in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2024, pp. 14 369–14 387.
- [169] P. Dasigi, K. Lo, I. Beltagy, A. Cohan, N. A. Smith, and M. Gardner, “A dataset of information-seeking questions and answers anchored in research papers,” in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2021.
- [170] T. Kočiský, J. Schwarz, P. Blunsom, C. Dyer, K. M. Hermann, G. Melis, and E. Grefenstette, “The NarrativeQA reading comprehension challenge,” *Transactions of the Association for Computational Linguistics*, vol. 6, pp. 317–328, 2018.
- [171] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal, “FEVER: a large-scale dataset for fact extraction and VERification,” in *NAACL-HLT*, 2018.
- [172] M. Zhong, D. Yin, T. Yu, A. Zaidi, M. Mutuma *et al.*, “QMSum: A New Benchmark for Query-based Multi-domain Meeting Summarization,” in *North American Association for Computational Linguistics (NAACL)*, 2021.
- [173] M. Koupaee and W. Y. Wang, “Wikihow: A large scale text summarization dataset,” *arXiv preprint arXiv:1810.09305*, 2018.
- [174] L. Huang, S. Cao, N. Parulian, H. Ji, and L. Wang, “Efficient attentions for long document summarization,” in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2021, pp. 1419–1436.
- [175] W. Kryściński, N. Rajani, D. Agarwal, C. Xiong, and D. Radev, “Booksum: A collection of datasets for long-form narrative summarization,” *arXiv preprint arXiv:2105.08209*, 2021.

- [176] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto, “Stanford alpaca: An instruction-following llama model,” https://github.com/tatsu-lab/stanford_alpaca, 2023.
- [177] A. Köpf, Y. Kilcher, D. Von Rütte, S. Anagnostidis, Z. R. Tam *et al.*, “Openassistant conversations-democratizing large language model alignment,” *Advances in neural information processing systems*, vol. 36, pp. 47 669–47 681, 2023.
- [178] W. Lian, B. Goodson, E. Pentland, A. Cook, C. Vong, and "Teknium", “Openorca: An open dataset of gpt augmented flan reasoning traces,” *HuggingFace repository*, 2023. [Online]. Available: <https://huggingface.co/datasets/Open-Orca/OpenOrca>
- [179] W. Zhao, X. Ren, J. Hessel, C. Cardie, Y. Choi, and Y. Deng, “Wildchat: 1m chatGPT interaction logs in the wild,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [180] G. Penedo, H. Kydlíček, L. B. Allal, A. Lozhkov, M. Mitchell *et al.*, “The fineweb datasets: Decanting the web for the finest text data at scale,” in *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.
- [181] G. Penedo, Q. Malartic, D. Hesslow, R. Cojocaru, A. Cappelli *et al.*, “The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only,” *arXiv preprint arXiv:2306.01116*, 2023.
- [182] M. Weber, D. Y. Fu, Q. Anthony, Y. Oren, S. Adams *et al.*, “Redpajama: an open dataset for training large language models,” *NeurIPS Datasets and Benchmarks Track*, 2024.
- [183] C. Tony, M. Mutas, N. Díaz Ferreyra, and R. Scandariato, “Llmseceval: A dataset of natural language prompts for security evaluations,” in *2023 IEEE/ACM 20th International Conference on Mining Software Repositories (MSR)*, 2023.
- [184] J. Ji, D. Hong, B. Zhang, B. Chen, J. Dai *et al.*, “Pku-saferlhf: Towards multi-level safety alignment for llms with human preference,” *arXiv preprint arXiv:2406.15513*, 2024.
- [185] R. Bhardwaj and S. Poria, “Red-teaming large language models using chain of utterances for safety-alignment,” *arXiv preprint arXiv:2308.09662*, 2023.
- [186] Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen *et al.*, “Training a helpful and harmless assistant with reinforcement learning from human feedback,” *arXiv preprint arXiv:2204.05862*, 2022.
- [187] J. Ji, M. Liu, J. Dai, X. Pan, C. Zhang *et al.*, “Beaver-tails: Towards improved safety alignment of llm via a human-preference dataset,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 24 678–24 704, 2023.
- [188] A. Zou, Z. Wang, N. Carlini, M. Nasr, J. Z. Kolter, and M. Fredrikson, “Universal and transferable adversarial attacks on aligned language models,” *arXiv preprint arXiv:2307.15043*, 2023.
- [189] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson *et al.*, “Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [190] Y. Qi, Q. Wu, P. Anderson, M. Liu, C. Shen, and A. van den Hengel, “Reverie: Remote embodied visual referring expression in real indoor environments,” *arXiv preprint arXiv:1904.10151v2*, vol. 2, 2020.
- [191] H. Chen, A. Suhr, D. Misra, N. Snavely, and Y. Artzi, “Touchdown: Natural language navigation and spatial reasoning in visual street environments,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 538–12 547.
- [192] M. Shridhar, J. Thomason, D. Gordon, Y. Bisk, W. Han *et al.*, “ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 740–10 749.
- [193] M. Shridhar, X. Yuan, M.-A. Côté, Y. Bisk, A. Trischler, and M. Hausknecht, “ALFWorld: Aligning Text and Embodied Environments for Interactive Learning,” in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.
- [194] J. Xie, K. Zhang, J. Chen, T. Zhu, R. Lou *et al.*, “Travelplanner: A benchmark for real-world planning with language agents,” *arXiv preprint arXiv: 2402.01622*, 2024.
- [195] K. Valmeekam, M. Marquez, A. Olmo, S. Sreedharan, and S. Kambhampati, “Planbench: An extensible benchmark for evaluating large language models on planning and reasoning about change,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 38 975–38 987, 2023.
- [196] A. Kapitanov, K. Kvanchiani, A. Nagaev, R. Kraynov, and A. Makhliarchuk, “Hagrid – hand gesture recognition image dataset,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, January 2024, pp. 4572–4581.

- [197] Y. Zhang, C. Cao, J. Cheng, and H. Lu, “Egogesture: A new dataset and benchmark for egocentric hand gesture recognition,” *IEEE Transactions on Multimedia*, vol. 20, no. 5, pp. 1038–1050, 2018.
- [198] I.-Y. Jeong and J. Park, “Cochlscene: Acquisition of acoustic scene data using crowdsourcing,” in *2022 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2022, pp. 17–21.
- [199] S. Yang, P. Luo, C.-C. Loy, and X. Tang, “Wider face: A face detection benchmark,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5525–5533.
- [200] Z. Liu, P. Luo, X. Wang, and X. Tang, “Deep learning face attributes in the wild,” in *Proceedings of International Conference on Computer Vision (ICCV)*, 2015, pp. 3730–3738.
- [201] A. Ritore, A. M. Oprescu, A. Estirado Bronchalo, and M. A. Armengol de la Hoz, “Covid data for shared learning (cdsl): A comprehensive, multimodal covid-19 dataset from hm hospitales (version 1.0.0),” *PhysioNet*, 2024.
- [202] M. Ghandi, F. W. Huang, J. Jané-Valbuena, G. V. Kryukov, T. R. Golub *et al.*, “Next-generation characterization of the cancer cell line encyclopedia,” *Nature*, vol. 569, pp. 503–508, 2019.
- [203] V. Agarwal and D. Kelley, “Saluki: Predicting mrna half-life from mrna sequence,” Data set, Zenodo, 2022. [Online]. Available: <https://doi.org/10.5281/zenodo.6326409>
- [204] J. Jankauskaitė, B. Jiménez-García, J. Dapkūnas, J. Fernández-Recio, and I. H. Moal, “Skempi 2.0: an updated benchmark of changes in protein–protein binding energy, kinetics and thermodynamics upon mutation,” *Bioinformatics*, vol. 35, no. 3, pp. 462–469, 2019.
- [205] U.S. National Library of Medicine, “Pubmed baseline repository,” https://www.nlm.nih.gov/databases/download/pubmed_medline.html, 2023, courtesy of the National Library of Medicine.
- [206] G. Mialon, C. Fourrier, T. Wolf, Y. LeCun, and T. Scialom, “Gaia: a benchmark for general ai assistants,” in *The Twelfth International Conference on Learning Representations*, 2023.
- [207] A. Srivastava, A. Rastogi, A. Rao, A. A. M. Shoeb, A. Abid *et al.*, “Beyond the imitation game: Quantifying and extrapolating the capabilities of language models,” *Transactions on machine learning research*, 2023.
- [208] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika *et al.*, “Measuring massive multitask language understanding,” *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.
- [209] Z. Guo, S. Cheng, H. Wang, S. Liang, Y. Qin *et al.*, “Stabletoolbench: Towards stable large-scale benchmarking on tool learning of large language models,” *arXiv preprint arXiv:2403.07714*, 2024.
- [210] X. Deng, Y. Gu, B. Zheng, S. Chen, S. Stevens *et al.*, “Mind2web: Towards a generalist agent for the web,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 28 091–28 114, 2023.
- [211] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher *et al.*, “Robonet: Large-scale multi-robot learning,” *arXiv preprint arXiv:1910.11215*, 2019.
- [212] R. Li, Z. Luo, and X. Du, “Fg-prm: Fine-grained hallucination detection and mitigation in language model mathematical reasoning,” *arXiv preprint arXiv:2410.06304*, 2024.
- [213] Q. Wu, G. Bansal, J. Zhang, Y. Wu, B. Li *et al.*, “Autogen: Enabling next-gen LLM applications via multi-agent conversations,” in *First Conference on Language Modeling*, 2024.
- [214] C. Gao, X. Lan, N. Li, Y. Yuan, J. Ding *et al.*, “Large language models empowered agent-based modeling and simulation: A survey and perspectives,” *Humanities and Social Sciences Communications*, vol. 11, no. 1, pp. 1–24, 2024.
- [215] S. Peter, K. Riemer, and J. D. West, “The benefits and dangers of anthropomorphic conversational agents,” *Proceedings of the National Academy of Sciences*, vol. 122, no. 22, p. e2415898122, 2025.
- [216] R. Meier, “Balancing minds and data: The privacy dilemma of llms and anthropomorphism in llms,” *Journal of Social Computing*, vol. 6, no. 3, pp. 173–183, 2025.
- [217] X. Shen, Z. Chen, M. Backes, Y. Shen, and Y. Zhang, “"do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models,” in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, 2024, pp. 1671–1685.
- [218] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright *et al.*, “Training language models to follow instructions with human feedback,” *Advances in neural information processing systems*, vol. 35, pp. 27 730–27 744, 2022.