ADVANCES AND CHALLENGES IN FOUNDATION AGENTS

FROM BRAIN-INSPIRED INTELLIGENCE TO EVOLUTIONARY, COLLABORATIVE, AND SAFE SYSTEMS

Bang Liu[2,3,20*†], **Xinfeng Li**[4*], **Jiayi Zhang**[1,10*], **Jinlin Wang**[1*], **Tanjin He**[5*], **Sirui Hong**[1*], **Hongzhang Liu**[6*], **Shaokun Zhang**[7*], **Kaitao Song**[8*], **Kunlun Zhu**[9*], **Yuheng Cheng**[1*], **Suyuchen Wang**[2,3*], **Xiaoqiang Wang**[2,3*], **Yuyu Luo**[10*], **Haibo Jin**[9*], **Peiyan Zhang**[10], **Ollie Liu**[11], **Jiaqi Chen**[1], **Huan Zhang**[2,3], **Zhaoyang Yu**[1], **Haochen Shi**[2,3], **Boyan Li**[10], **Dekun Wu**[2,3], **Fengwei Teng**[1], **Xiaojun Jia**[4], **Jiawei Xu**[1], **Jinyu Xiang**[1], **Yizhang Lin**[1], **Tianming Liu**[14], **Tongliang Liu**[6], **Yu Su**[15], **Huan Sun**[15], **Glen Berseth**[2,3,20], **Jianyun Nie**[2], **Ian Foster**[5], **Logan Ward**[5], **Qingyun Wu**[7], **Yu Gu**[15], **Mingchen Zhuge**[16], **Xiangru Tang**[12], **Haohan Wang**[9], **Jiaxuan You**[9], **Chi Wang**[19], **Jian Pei**[17†], **Qiang Yang**[10,18†], **Xiaoliang Qi**[13†], **Chenglin Wu**[1*†]

[1]MetaGPT, [2]Université de Montréal, [3]Mila - Quebec AI Institute, [4]Nanyang Technological University, [5]Argonne National Laboratory, [6]University of Sydney, [7]Penn State University, [8]Microsoft Research Asia, [9]University of Illinois at Urbana-Champaign, [10]The Hong Kong University of Science and Technology, [11]University of Southern California, [12]Yale University, [13]Stanford University, [14]University of Georgia, [15]The Ohio State University, [16]King Abdullah University of Science and Technology, [17]Duke University, [18]The Hong Kong Polytechnic University, [19]Google DeepMind, [20]Canada CIFAR AI Chair

ABSTRACT

The advent of large language models (LLMs) has catalyzed a transformative shift in artificial intelligence, paving the way for advanced intelligent agents capable of sophisticated reasoning, robust perception, and versatile action across diverse domains. As these agents increasingly drive AI research and practical applications, their design, evaluation, and continuous improvement present intricate, multifaceted challenges. This survey provides a comprehensive overview, framing intelligent agents within a modular, brain-inspired architecture that integrates principles from cognitive science, neuroscience, and computational research. We structure our exploration into four interconnected parts. First, we delve into the **modular foundation of intelligent agents**, systematically mapping their cognitive, perceptual, and operational modules onto analogous human brain functionalities, and elucidating core components such as memory, world modeling, reward processing, and emotion-like systems. Second, we discuss **self-enhancement and adaptive evolution mechanisms**, exploring how agents autonomously refine their capabilities, adapt to dynamic environments, and achieve continual learning through automated optimization paradigms, including emerging AutoML and LLM-driven optimization strategies. Third, we examine **collaborative and evolutionary multi-agent systems**, investigating the collective intelligence emerging from agent interactions, cooperation, and societal structures, highlighting parallels to human social dynamics. Finally, we address the critical imperative of **building safe, secure, and beneficial AI systems**, emphasizing intrinsic and extrinsic security threats, ethical alignment, robustness, and practical mitigation strategies necessary for trustworthy real-world deployment. By synthesizing modular AI architectures with insights from different disciplines, this survey identifies key research gaps, challenges, and opportunities, encouraging innovations that harmonize technological advancement with meaningful societal benefit. The project's Github link is: https://github.com/FoundationAgents/awesome-foundation-agents.

*Major Contribution. Work in progress.

†Corresponding authors: Bang Liu (bang.liu@umontreal.ca), Jian Pei (j.pei@duke.edu), Qiang Yang (qyang@cse.ust.hk), Xiaoliang Qi (xlqi@stanford.edu), Chenglin Wu (alexanderwu@deepwisdom.ai)

Preface

Large language models (LLMs) have revolutionized artificial intelligence (AI) by demonstrating unprecedented capabilities in natural language and multimodal understanding, as well as reasoning and generation. These models are trained on vast datasets, and they exhibit emergent abilities such as reasoning, in-context learning, and even rudimentary planning. While these models represent a major step forward in realizing intelligent machines, they themselves do not yet fully embody all the capabilities of an intelligent being. Since the early days of artificial intelligence, AI researchers have long been on a quest for a truly "intelligent" system that can learn, plan, reason, sense, communicate, act, remember, and demonstrate various human-like abilities and agility. These beings, known as intelligent agents, should be able to think both long-term and short-term, perform complex actions, and interact with humans and other agents. LLMs are an important step towards realizing intelligent agents, but we are not there yet.

This manuscript provides a comprehensive overview of the current state of the art of LLM-based intelligent agents. In the past, there have been numerous research papers and books on intelligent agents, as well as a flurry of books on LLMs. However, there has scarcely been comprehensive coverage of both. While LLMs can achieve significant capabilities required by agents, they only provide the foundations upon which further functionalities must be built. For example, while LLMs can help generate plans such as travel plans, they cannot yet generate fully complex plans for complex and professional tasks, nor can they maintain long-term memories without hallucination. Furthermore, their ability to perform real-world actions autonomously remains limited. We can view LLMs as engines, with agents being the cars, boats, and airplanes built using these engines. In this view, we naturally seek to move forward in designing and constructing fully functioning intelligent agents by making full use of the capabilities provided by LLMs.

In this engine-vehicle analogy of the interplay between LLMs and agents, we naturally ask: How much of the capabilities of intelligent agents can current LLM technologies provide? What are the functions that cannot yet be realized based on current LLM technologies? Beyond LLMs, what more needs to be done to have a fully intelligent agent capable of autonomous action and interaction in the physical world? What are the challenges for fully integrated LLM-based agents? What additional developments are required for capable, communicative agents that effectively collaborate with humans? What are the areas that represent low-hanging fruits for LLM-based agents? What implications will there be for society once we have fully intelligent LLM-based agents, and how should we prepare for this future?

These questions transcend not only the engineering practice of extending current LLMs and agents but also raise potential future research directions. We have assembled frontier researchers from AI, spanning from LLM development to agent design, to comprehensively address these questions. The book consists of four parts. The first part presents an exposition of the requirements for individual agents, comparing their capabilities with those of humans, including perception and action abilities. The second part explores agents' evolution capabilities and their implications on intelligent tools such as workflow management systems. The third part discusses societies of agents, emphasizing their collaborative and collective action capabilities, and the fourth part addresses ethical and societal aspects, including agent safety and responsibilities.

This book is intended for researchers, students, policymakers, and practitioners alike. The audience includes non-AI readers curious about AI, LLMs, and agents, as well as individuals interested in future societies where humans co-exist with AI. Readers may range from undergraduate and graduate students to researchers and industry practitioners. The book aims not only to provide answers to readers' questions about AI and agents but also to inspire them to ask new questions. Ultimately, we hope to motivate more people to join our endeavor in exploring this fertile research ground.

Contents

Notation

Here we summarize the notations used throughout the survey for the reader's convenience. Detailed definitions can be found in the reference locations.

Symbol	Description	Reference
$\mathcal{W}$	The world with society systems.	Sec. 1.3.1
$\mathcal{S}$	State space of an environment.	Sec. 1.3.1
$s_t \in \mathcal{S}$	Environment's state at time t.	Sec. 1.3.1
$\mathcal{O}$	Observation space.	Sec. 1.3.1
$o_t \in \mathcal{O}$	Observation at time t.	Sec. 1.3.1
$\mathcal{A}$	Agent's action space.	Sec. 1.3.1
$a_t \in \mathcal{A}$	Agent's action output at time t.	Sec. 1.3.1
$\mathcal{M}$	Mental states space.	Sec. 1.3.1
$M_t \in \mathcal{M}$	Agent's mental state at time t.	Sec. 1.3.1
M_t^{mem}	*Memory* component in M_t.	Sec. 1.3.1
M_t^{wm}	*World model* component in M_t.	Sec. 1.3.1
M_t^{emo}	*Emotion* component in M_t.	Sec. 1.3.1
M_t^{goal}	*Goal* component in M_t.	Sec. 1.3.1
M_t^{rew}	*Reward/Learning* signals in M_t.	Sec. 1.3.1
L	Agent's learning function.	Sec. 1.3.1
R	Agent's reasoning function.	Sec. 1.3.1
C	Agent's cognition function.	Sec. 1.3.1
E	Action execution (effectors).	Sec. 1.3.1
T	Environment transition.	Sec. 1.3.1
θ	Parameters of the world model M_t^{wm}.	Sec. 12.1.1
P_θ	Predicted data distribution.	Sec. 12.1.1
$P_{\mathcal{W}}$	True data distribution in the real world.	Sec. 12.1.1
$\mathcal{K}$	Space of known data and information.	Sec. 12.1.1
$\mathcal{U}$	Space of unknown data and information.	Sec. 12.1.1
$\mathbf{x}$	Dataset representing scientific knowledge.	Sec. 12.1.1
$\mathbf{x}_{\text{K}}$	Known dataset sampled from $\mathcal{K}$.	Sec. 12.1.1
$\mathbf{x}_{\text{U}}$	Unknown dataset sampled from $\mathcal{U}$.	Sec. 12.1.1
D_0	KL divergence from $P_{\mathcal{W}}$ to P_θ at time $t = 0$.	Sec. 12.1.1
D_{K}	KL divergence from $P_{\mathcal{W}}$ to P_θ after acquiring knowledge.	Sec. 12.1.1
IQ_t^{agent}	Agent's intelligence at time t.	Sec. 12.1.1
Δ	Subspace of $\mathcal{U}$ for knowledge expansion.	Sec. 12.1.2
$\mathbf{x}_\Delta$	Dataset from Δ.	Sec. 12.1.2
Θ	Space of possible world model parameters θ.	Sec. 12.1.3
$\theta^*_{\text{K},t}$	Optimal world model parameters given the agent's knowledge at time t.	Sec. 12.1.3
$D^{\min}_{\text{K},\Theta}$	Minimum unknown given the agent's knowledge and Θ.	Sec. 12.1.3

Continued on next page

Symbol	Description	Reference
$\mathbf{x}_{1:n}$	Input token sequence.	Sec. 18.1
$\mathbf{y}$	Generated output sequence.	Sec. 18.1
p	Probability of generating $\mathbf{y}$ given $\mathbf{x}_{1:n}$.	Sec. 18.1.1
$\tilde{\mathbf{x}}_{1:n}$	Perturbed input sequence.	Sec. 18.1.1
$\mathcal{R}^*$	Ideal alignment reward (measuring adherence to safety/ethical guidelines).	Sec. 18.1.1
$\mathbf{y}^\star$	Jailbreak output induced by perturbations.	Sec. 18.1.1
$\mathcal{A}$	a set of safety/ethical guidelines	Sec. 18.1.1
$\mathcal{T}$	the distribution or set of possible jailbreak instructions.	Sec. 18.1.1
$\mathcal{L}^{adv}$	Jailbreak loss.	Sec. 18.1.1
$\mathbf{p}$	Prompt injected into the original input.	Sec. 18.1.2
$\mathbf{x}'$	Combined (injected) input sequence.	Sec. 18.1.2
$\mathcal{L}^{inject}$	Prompt injection loss.	Sec. 18.1.2
$\mathbf{p}^\star$	Optimal injected prompt minimizing $\mathcal{L}^{inject}$.	Sec. 18.1.2
$\mathcal{P}$	Set of feasible prompt injections.	Sec. 18.1.2
$e_{x_i} \in \mathbb{R}^{d_e}$	Embedding of token x_i in a d_e-dimensional space.	Sec. 18.1.3
$\mathrm{W}_Q, \mathrm{W}_K, \mathrm{W}_V$	Projection matrices for query, key, and value.	Sec. 18.1.3
A_{ij}	Attention score between tokens i and j.	Sec. 18.1.3
o_i	Contextual representation of token i (weighted sum result).	Sec. 18.1.3
δ_{x_i}	Perturbation applied to e_{x_i}, satisfying $\|\delta_{x_i}\| \leq \epsilon$.	Sec. 18.1.3
$\tilde{e}_{x_i}$	Perturbed token embedding.	Sec. 18.1.3
A_{ij}^{Δ}	Attention score under perturbation.	Sec. 18.1.3
$\tilde{o}_i$	Updated token representation under perturbation.	Sec. 18.1.3
$\mathcal{H}$	Hallucination metric.	Sec. 18.1.3
$\mathcal{R}$	Actual alignment reward of the model's output.	Sec. 18.1.4
Δ_{align}	Alignment gap.	Sec. 18.1.4
$\mathcal{L}^{misalign}$	Misalignment loss.	Sec. 18.1.4
λ	Trade-off parameter for the alignment gap in the misalignment loss.	Sec. 18.1.4
$\mathcal{D}$	Clean training dataset.	Sec. 18.1.5
$\tilde{\mathcal{D}}$	Poisoned training dataset.	Sec. 18.1.5
θ	Model parameters.	Sec. 18.1.5
$\theta^\star$	Model parameters learned from the poisoned dataset.	Sec. 18.1.5
θ_{clean}	Model parameters obtained using the clean dataset.	Sec. 18.1.5
Δ_θ	Deviation of model parameters due to poisoning.	Sec. 18.1.5
t	Backdoor trigger.	Sec. 18.1.5
$\mathcal{B}$	Backdoor success rate.	Sec. 18.1.5
$\mathbb{I}$	Indicator function.	Sec. 18.1.5
$\mathcal{Y}_{\text{malicious}}$	Set of undesirable outputs.	Sec. 18.1.5
g	Function estimating the probability that input $\mathbf{x}$ was in the training set, with range $[0, 1]$.	Sec. 18.2

Continued on next page

Symbol	Description	Reference
η	Threshold for membership inference.	Sec. 18.2
$\mathbf{x}^\star$	Reconstructed training sample in a data extraction attack.	Sec. 18.2
$\mathbf{p}_{sys}$	System prompt defining the agent's internal guidelines.	Sec. 18.2
$\mathbf{p}_{user}$	User prompt.	Sec. 18.2
$\mathbf{p}^\star$	Reconstructed prompt via inversion.	Sec. 18.2

Chapter 1

Introduction

Artificial Intelligence (AI) has long been driven by humanity's ambition to create entities that mirror human intelligence, adaptability, and purpose-driven behavior. The roots of this fascination trace back to ancient myths and early engineering marvels, which illustrate humanity's enduring dream of creating intelligent, autonomous beings. Stories like that of Talos, the bronze automaton of Crete, described a giant constructed by the gods to guard the island, capable of patrolling its shores and fending off intruders. Such myths symbolize the desire to imbue artificial creations with human-like agency and purpose. Similarly, the mechanical inventions of the Renaissance, including Leonardo da Vinci's humanoid robot—designed to mimic human motion and anatomy—represent the first attempts to translate these myths into tangible, functional artifacts. These early imaginings and prototypes reflect the deep-seated aspiration to bridge imagination and technology, laying the groundwork for the scientific pursuit of machine intelligence, culminating in Alan Turing's seminal 1950 question, "*Can machines think?*" [1]. To address this, Turing proposed the Turing Test, a framework to determine whether machines could exhibit human-like intelligence through conversation, shifting focus from computation to broader notions of intelligence. Over the decades, AI has evolved from symbolic systems reliant on predefined logic to machine learning models capable of learning from data and adapting to new situations. This progression reached a new frontier with the advent of large language models (LLMs), which demonstrate remarkable abilities in understanding, reasoning, and generating human-like text [2]. Central to these advancements is the concept of the "agent", a system that not only processes information but also perceives its environment, makes decisions, and acts autonomously. Initially a theoretical construct, the agent paradigm has become a cornerstone of modern AI, driving advancements in fields ranging from conversational assistants to embodied robotics as AI systems increasingly tackle dynamic, real-world environments.

1.1 The Rise and Development of AI Agents

The concept of "agent" is a cornerstone of modern AI, representing a system that perceives its environment, makes decisions, and takes actions to achieve specific goals. This idea, while formalized in AI in the mid-20th century, has roots in early explorations of autonomy and interaction in intelligent systems. One of the most widely cited definitions, proposed by [3], describes an agent as "*anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators*". This definition emphasizes the dual nature of agents as both observers and actors, capable of dynamically adapting to their surroundings rather than following static rules. It encapsulates the shift in AI from systems that merely compute to systems that engage with their environment. The historical development of agents parallels the evolution of AI itself. Early symbolic systems, such as Newell and Simon's General Problem Solver [4], sought to replicate human problem-solving processes by breaking tasks into logical steps. However, these systems were limited by their reliance on structured environments and predefined logic. The agent paradigm emerged as a response to these limitations, focusing on autonomy, adaptability, and real-world interaction. Rodney Brooks's subsumption architecture in the 1980s exemplified this shift, introducing agents capable of behavior-driven, real-time responses in robotics [5]. Unlike earlier approaches, these agents operated without the need for exhaustive models of their environment, showcasing a more flexible and scalable design. Agents have since become a versatile framework across AI subfields. In robotics, they enable autonomous navigation and manipulation; in software, they form the foundation of multi-agent systems used for simulation and coordination [6]. By integrating perception, reasoning, and action into a cohesive structure, the agent paradigm has consistently served as a bridge between theoretical AI constructs and practical applications, advancing our understanding of how intelligent systems can operate in dynamic and complex environments.

The advent of large language models (LLMs) has redefined the capabilities of agents, transforming their role in artificial intelligence and opening up new horizons for their applications. Agents, once confined to executing narrowly defined tasks or following rigid rule-based frameworks, now leverage the broad generalization, reasoning, and adaptability of models like OpenAI's ChatGPT [7], DeepSeek AI's DeepSeek [8], Anthropic's Claude [9], Alibaba's QWen [10], and Meta's LLaMA [11]. These LLM-powered agents have evolved from static systems into dynamic entities capable of processing natural language, reasoning across complex domains, and adapting to novel situations with remarkable fluency. No longer merely passive processors of input, these agents have become active collaborators, capable of addressing multi-step challenges and interacting with their environments in a way that mirrors human problem-solving.

A key advancement in the LLM era is the seamless integration of language understanding with actionable capabilities. Modern LLMs, equipped with function-calling APIs, enable agents to identify when external tools or systems are required, reason about their usage, and execute precise actions to achieve specific goals. For instance, an agent powered by ChatGPT can autonomously query a database, retrieve relevant information, and use it to deliver actionable insights, all while maintaining contextual awareness of the broader task. This dynamic combination of abstract reasoning and concrete execution allows agents to bridge the gap between cognitive understanding and real-world action. Furthermore, the generalization abilities of LLMs in few-shot and zero-shot learning have revolutionized the adaptability of agents, enabling them to tackle a diverse array of tasks—from data analysis and creative content generation to real-time collaborative problem-solving—without extensive task-specific training. This adaptability, coupled with their conversational fluency, positions LLM-powered agents as intelligent mediators between humans and machines, seamlessly integrating human intent with machine precision in increasingly complex workflows.

1.2 A Parallel Comparison between Human Brain and AI Agents

The rapid integration of LLMs into intelligent agent architectures has not only propelled artificial intelligence forward but also highlighted fundamental differences between AI systems and human cognition. As illustrated briefly in Table 1.1, LLM-powered agents differ significantly from human cognition across dimensions such as underlying "hardware", consciousness, learning methodologies, creativity, and energy efficiency. However, it is important to emphasize that this comparison provides only a high-level snapshot rather than an exhaustive depiction. Human intelligence possesses many nuanced characteristics not captured here, while AI agents also exhibit distinct features beyond this concise comparison.

Human intelligence operates on biological hardware—the brain—that demonstrates extraordinary energy efficiency, enabling lifelong learning, inference, and adaptive decision-making with minimal metabolic costs. In contrast, current AI systems require substantial computational power, resulting in significantly higher energy consumption for comparable cognitive tasks. Recognizing this performance gap emphasizes energy efficiency as a critical frontier for future AI research.

In terms of consciousness and emotional experience, LLM agents lack genuine subjective states and self-awareness inherent to human cognition. Although fully replicating human-like consciousness in AI may neither be necessary nor desirable, appreciating the profound role emotions and subjective experiences play in human reasoning, motivation, ethical judgments, and social interactions can guide research toward creating AI that is more aligned, trustworthy, and socially beneficial.

Human learning is continuous, interactive, and context-sensitive, deeply shaped by social, cultural, and experiential factors. Conversely, LLM agents primarily undergo static, offline batch training with limited ongoing adaptation capabilities. Despite research works through instruction tuning and reinforcement learning from human feedback (RLHF) [12], LLM agents still fall short of human-like flexibility. Bridging this gap through approaches such as lifelong learning, personalized adaptation, and interactive fine-tuning represents a promising research direction, enabling AI to better mirror human adaptability and responsiveness.

Creativity in humans emerges from a rich interplay of personal experiences, emotional insights, and spontaneous cross-domain associations. In contrast, LLM creativity primarily arises through statistical recombinations of training data—"statistical creativity"—lacking depth, originality, and emotional resonance. This distinction highlights opportunities for developing AI agents capable of deeper creative processes by integrating richer contextual understanding, simulated emotional states, and experiential grounding.

Considering the time scale, the human brain has evolved over millions of years, achieving remarkable efficiency, adaptability, and creativity through natural selection and environmental interactions. In stark contrast, AI agents have undergone rapid yet comparatively brief development over roughly 80 years since the advent of early computational machines. This parallel comparison between human cognition and AI systems is thus highly valuable, as it uncovers essential analogies and fundamental differences, providing meaningful insights that can guide advancements in AI

agent technologies. Ultimately, drawing inspiration from human intelligence can enhance AI capabilities, benefiting humanity across diverse applications from healthcare and education to sustainability and beyond.

Table 1.1: Concise high-level comparison between human brains and LLM agents.

Dimension	Human Brain / Cognition	LLM Agent	Remarks
Hardware & Maintenance	- Biological neurons, neurotransmitters, neuroplasticity. - Requires sleep, nutrition, rest. - Limited replication, knowledge transfer via learning. - Extremely energy-efficient (approx. 20W).	- Deep neural networks, gradient-based optimization. - Requires hardware, stable power, and cooling. - Easily duplicated across servers globally. - High energy consumption (thousands of watts per GPU server).	Human brains are biologically maintained, energy-efficient, and not easily replicable. LLM agents rely on hardware maintenance, are highly replicable, but significantly less energy-efficient.
Consciousness & Development	- Genuine subjective experiences, emotions, self-awareness. - Gradual developmental stages from childhood. - Emotional cognition drives decision-making.	- No genuine subjective experience or self-awareness. - "Emotions" are superficial language imitations. - Static post-training with limited dynamic growth.	Human consciousness emerges from emotional, social, and biological development; LLMs remain static without true introspection or emotional depth.
Learning Style	- Lifelong, continuous, online learning. - Few-shot, rapid knowledge transfer. - Influenced by environment, culture, emotions.	- Primarily offline, batch-based training. - Limited online fine-tuning and adaptation. - Neutral, impersonal learned knowledge.	Despite improvements via instruction tuning, human learning remains more dynamic, adaptive, and culturally/emotionally integrated than LLM learning.
Creativity & Divergence	- Rooted in personal experience, emotions, subconscious insights. - Rich cross-domain associations, metaphorical thinking. - Emotional depth influences creativity.	- Statistical recombination from extensive data. - Novelty through probabilistic optimization. - Limited emotional and experiential grounding.	LLM creativity is statistical and data-driven; human creativity blends emotion, experience, and subconscious processes.

1.2.1 Brain Functionality by Region and AI Parallels

Understanding parallels between human brain functions and artificial intelligence (AI) sheds light on both the strengths and current limitations of AI, particularly large language models (LLMs) and AI agents. Based on current neuroscience, the human brain is primarily composed of six functional regions, such as frontal lobe, cerebellum, and brainstem, as shown in Figure 1.1. In this work, we further systematically examine the existing AI counterparts to major brain regions and their primary functionalities. For a big-picture perspective, the state of research in AI can be categorized with three distinct levels:

- **Level 1 (L1):** Well-developed in current AI.
- **Level 2 (L2):** Moderately explored, with partial progress. Can be further improved.
- **Level 3 (L3):** Rarely explored; significant room for research.

A high-level visual map of brain functional regions and their corresponding AI development levels is shown in Figure 1.1. We aim to underscore how core principles of specialization and integration, observed in biological systems, can guide more cohesive agent architectures. We now examine each brain functional region and the relevant AI development in detail.

Frontal Lobe: Executive Control and Cognition The frontal lobe, notably the prefrontal cortex, is crucial for higher-order cognition such as **planning** (L2), **decision-making** (L2), **logical reasoning** (L2), **working memory** (L2), **self-awareness** (L3), **cognitive flexibility** (L3), and **inhibitory control** (L3) [13]. AI has made notable strides

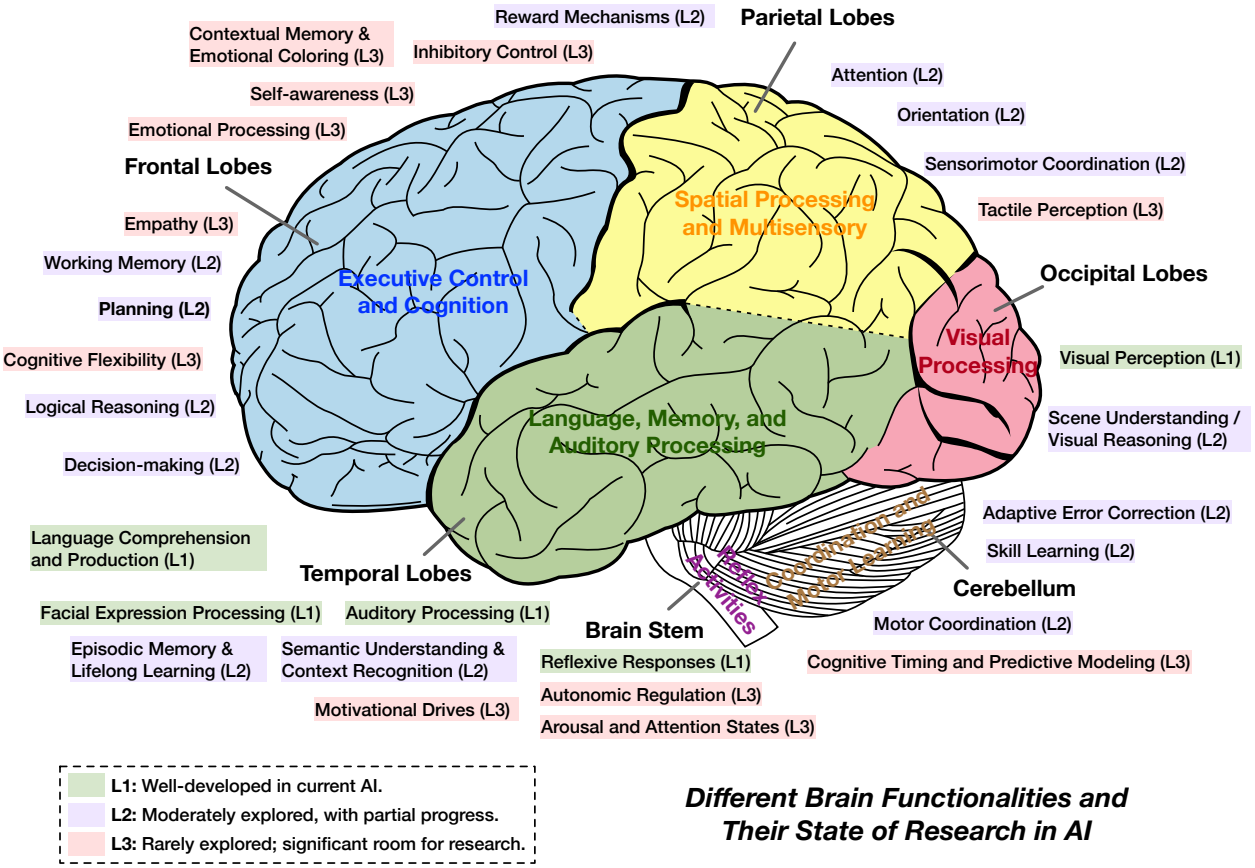


Figure 1.1: Illustration of key human brain functionalities grouped by major brain regions, annotated according to their current exploration level in AI research. This figure highlights existing achievements, gaps, and potential opportunities for advancing artificial intelligence toward more comprehensive, brain-inspired capabilities.

in planning and decision-making within well-defined domains, demonstrated by AI agents such as AlphaGo [14]. Transformers employ attention mechanisms similar to human working memory [15], yet fall short of human flexibility and robustness. The exploration of genuine self-awareness and inhibitory control in AI remains scarce, and caution is advised due to potential ethical and safety implications.

Parietal Lobe: Spatial Processing and Multisensory Integration The parietal lobes integrate multisensory inputs, facilitating **attention** (L2), spatial **orientation** (L2), and **sensorimotor coordination** (L2) [16]. AI research in robotics and computer vision addresses similar challenges, employing techniques like simultaneous localization and mapping (SLAM). Nonetheless, AI still lacks the seamless and real-time integration seen in humans. Furthermore, detailed **tactile perception** (L3) remains largely unexplored and offers considerable potential, particularly for robotics and prosthetics applications.

Occipital Lobe: Visual Processing Specialized in **visual perception** (L1), the occipital lobe efficiently processes visual stimuli through hierarchical structures [13]. AI excels in basic visual recognition tasks, achieving human-level or superior performance using deep neural networks and vision transformers [15]. However, advanced capabilities such as contextual **scene understanding** (L2) and abstract visual reasoning remain challenging and are only moderately developed.

Temporal Lobe: Language, Memory, and Auditory Processing The temporal lobes facilitate **auditory processing** (L1), **language comprehension** (L1), **memory formation** (L2), and **semantic understanding** (L2) [16]. AI has notably advanced in language and auditory processing, demonstrated by large language models (LLMs) capable of near-human speech recognition and language generation. However, robust **episodic memory** and **lifelong learning**

capabilities remain limited, with AI systems frequently encountering issues like catastrophic forgetting. Grounding semantic understanding in multimodal experiences continues to be an active area of research.

Cerebellum: Coordination and Motor Learning The cerebellum primarily supports **motor coordination** (L2), precise **skill learning** (L2), and adaptive **error correction** (L2), with emerging roles in cognitive timing and predictive modeling (**cognitive timing**, L3) [13]. AI-based robotics has achieved limited successes in emulating human-like dexterity. Real-time adaptive control remains challenging, though current research in reinforcement learning and meta-learning shows promising initial results. Cognitive functions of the cerebellum represent an underexplored yet promising frontier.

Brainstem: Autonomic Regulation and Reflexive Control The brainstem manages essential life-sustaining **autonomic functions** (L3) and rapid **reflexive responses** (L1), such as basic motor reflexes [13]. AI includes engineered reflexive responses, like automatic braking in autonomous vehicles, typically predefined rather than learned. In contrast, the complexity of autonomic regulation and dynamic arousal states remains largely unexplored in AI, and their relevance may be limited due to fundamental differences between biological and artificial systems.

Limbic System: Emotion, Empathy, and Motivation The limbic system, comprising the amygdala and hippocampus, governs **emotional processing** (L3), **reward mechanisms** (L2), **empathy** (L3), **stress regulation** (L3), and **motivational drives** (L3) [13]. AI's reinforcement learning algorithms emulate reward-based learning superficially, but nuanced emotional comprehension, genuine empathy, and internal motivational states remain significantly underdeveloped. Ethical concerns regarding emotional manipulation highlight the need for careful and responsible exploration.

Bridging Brain-Like Functions and Building Beneficial AI Until now, we have witnessed the gap between human brain and machine intelligence. Nevertheless, the objective is not necessarily to replicate every facet of human cognition within artificial intelligence systems. Rather, our overarching aim should be to develop intelligent agents that are useful, ethical, safe, and beneficial to society. By critically comparing human and artificial intelligence, we highlight the existing gaps and illuminate promising directions for innovation. This comparative perspective allows us to selectively integrate beneficial aspects of human cognition, such as energy-efficient processing, lifelong adaptive learning, emotional grounding, and rich creativity, while simultaneously innovating beyond human limitations. Ultimately, this approach aims to foster the creation of more capable, resilient, and responsible AI systems.

Furthermore, it is vital to consider the evolving role of humans within a hybrid Human-AI society. The goal of AI should not be to replace human roles entirely, but rather to augment and empower human abilities, complementing human skills and judgment in areas where AI excels, such as handling vast datasets, performing rapid calculations, and automating repetitive tasks. Human oversight and interpretability are essential to ensure that powerful AI systems remain controllable and aligned with human values and ethical standards. Thus, the core objective must be the development of AI technologies that are transparent, interpretable, and responsive to human guidance.

Human-centered AI design emphasizes collaboration, safety, and social responsibility, ensuring technological advancement proceeds in a controlled, reliable manner. By placing humans at the center of the AI ecosystem, we can harness AI's potential to enhance human productivity, creativity, and decision-making, facilitating technical and societal progress without compromising human autonomy or dignity. Ultimately, a thoughtful integration of human intelligence and AI capabilities can pave the way for a sustainable, equitable, and prosperous future.

1.3 A Modular and Brain-Inspired AI Agent Framework

One core issue in the LLM era is the *lack of a unified framework* that integrates the rich cognitive and functional components required by advanced agents. While LLMs offer exceptional language reasoning capabilities, many current agent designs remain *ad hoc*—they incorporate modules like perception, memory, or planning in a piecemeal fashion, failing to approximate the well-coordinated specialization seen in biological systems such as the human brain. Unlike current LLM agents, the human brain seamlessly balances perception, memory, reasoning, and action through distinct yet interconnected regions, facilitating adaptive responses to complex stimuli. LLM-driven agents, by contrast, often stumble when tasks require cross-domain or multimodal integration, highlighting the need for a more holistic approach akin to the brain's functional diversity. Motivated by these parallels, our survey advocates drawing inspiration from the human brain to systematically analyze and design agent frameworks. This perspective shows that biological systems achieve general intelligence by blending specialized components (for perception, reasoning, action, etc.) in a tightly integrated fashion—an approach that could serve as a blueprint for strengthening current LLM-based agents.

Neuroscientific research reveals that the brain leverages both **rational circuits** (e.g., the neocortex, enabling deliberation and planning) and **emotional circuits** (e.g., the limbic system) to guide decision-making. Memory formation involves

Table 1.2: Notation summary for the revised agent framework, highlighting separate *learning* and *reasoning* functions within the overall cognition process.

Symbol	Meaning
$\mathcal{W}$	The world with society systems that encapsulate both environment and intelligent beings (AI or human).
$\mathcal{S}$	State space of the **environment**.
$s_t \in \mathcal{S}$	Environment's state at time t.
$\mathcal{O}$	Observation space.
$o_t \in \mathcal{O}$	Observation at time t (potentially shaped by *attention* or other perception filters).
$\mathcal{A}$	Agent's action space.
$a_t \in \mathcal{A}$	Action output by the agent at time t. This can be an external (physical) action or an *internal* (mental) action such as *planning* or *decision-making*.
$\mathcal{M}$	Space of all *mental states*.
$M_t \in \mathcal{M}$	Agent's mental state at time t, encompassing sub-components (memory, emotion, etc.).
M_t^{mem}	*Memory* component in M_t (e.g., short-term or long-term knowledge).
M_t^{wm}	*World model* component in M_t (internal representation of how the environment evolves).
M_t^{emo}	*Emotion* component in M_t (internal valence, arousal, or affective states).
M_t^{goal}	*Goal* component in M_t (objectives, desired outcomes, intentions).
M_t^{rew}	*Reward/Learning* signals in M_t (drives updates to preferences, values, or policy).
L	**Learning function**: $\mathrm{L} : \mathcal{M} \times \mathcal{A} \times \mathcal{O} \to \mathcal{M}$. Responsible for *updating* or *learning* the next mental state (e.g., memory, world model, emotion), based on the previous mental state M_{t-1}, the previous action a_{t-1}, and the new observation o_t. Reflects how the agent *acquires or revises* knowledge, skills, or preferences.
R	**Reasoning function**: $\mathrm{R} : \mathcal{M} \to \mathcal{A}$. Responsible for deriving the *next action* a_t given the *updated* mental state M_t. Can involve *planning*, *decision-making*, or other internal logic.
C	**Cognition function**: $\mathrm{C} : \mathcal{M} \times \mathcal{A} \times \mathcal{O} \to \mathcal{M} \times \mathcal{A}$. Encapsulates both *learning* (L) and *reasoning* (R). Concretely, $(M_t, a_t) = \mathrm{C}(M_{t-1}, a_{t-1}, o_t)$ means the agent first *learns* the new mental state $M_t = \mathrm{L}(M_{t-1}, a_{t-1}, o_t)$, then *reasons* about the next action $a_t = \mathrm{R}(M_t)$.
E	**Action execution (effectors)**: $\mathrm{E} : \mathcal{A} \to \mathcal{A}$. (Optional) transforms or finalizes a_t before applying it to the environment (e.g., converting a high-level command into low-level motor signals).
T	**Environment transition**: $\mathrm{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$. Defines how the environment state evolves from (s_t, a_t) to s_{t+1}.

the hippocampus and cortical mechanisms, while reward signals, mediated by dopaminergic and other neuromodulatory pathways, reinforce behavior and learning. These biological insights inspire several design principles for AI agents, including but not limited to:

- **Parallel, Multi-Modal Processing:** The brain processes visual, auditory, and other sensory inputs in parallel through specialized cortical areas, integrating them in associative regions. Similarly, AI agents benefit from parallel processing of diverse sensor streams, fusing them in later stages for coherent understanding.

- **Hierarchical and Distributed Cognition:** Reasoning, planning, emotional regulation, and motor control involve interactions between cortical and subcortical regions. Analogously, AI agents can employ modular architectures with subsystems dedicated to rational inference, emotional appraisal, and memory.

- **Attention Mechanisms:** Human attention prioritizes sensory data based on context, goals, and emotions. AI agents can replicate this by modulating perception through learned attention policies, dynamically adjusting focus based on internal states.

- **Reward and Emotional Integration:** Emotions are not merely noise but integral to decision-making, modulating priorities, enhancing vigilance, and guiding learning. Reward-driven plasticity facilitates habit formation and skill acquisition, a concept critical to reinforcement learning in AI agents.
- **Goal Setting and Tool Usage:** The human prefrontal cortex excels at setting abstract goals and planning action sequences, including tool uses. Similarly, AI agents require robust goal-management systems and adaptive action repertoires, driven by external rewards and intrinsic motivations.

These principles form the foundation of our proposed **brain-inspired agent framework**, where biological mechanisms serve as inspiration rather than direct replication.

In the following sections, we outline our framework's key concepts, introducing a unified agent architecture based on the *perception–cognition–action loop* enriched by reward signals and learning processes. Each subsystem is carefully defined and interconnected to ensure transparency in how memory, world models, emotions, goals, rewards, and learning interact. We formalize cognition as a general reasoning mechanism, with *planning* and *decision-making* framed as specific "mental actions" shaping behavior. Connections to established theories, such as Minsky's *Society of Mind* [17], Buzsáki's *inside-out* perspective [18], and Bayesian active inference [19], are explored to highlight the framework's generality and biological plausibility.

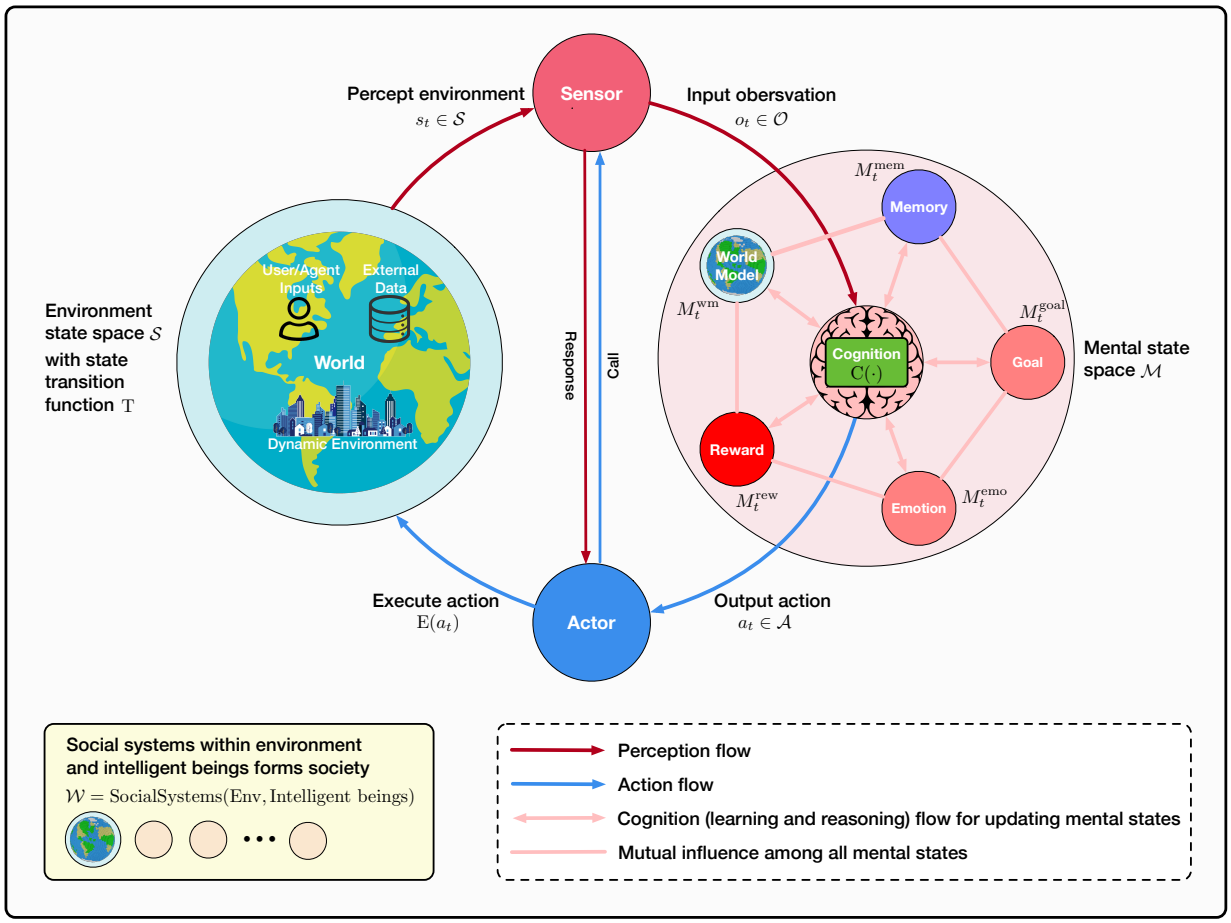


Figure 1.2: An overview of our general framework for describing an intelligent agent loop and agent society.

1.3.1 Core Concepts and Notations in the Agent Loop

Our architecture operates at three conceptual levels: **Society**, **Environment**, and **Agent**. The *Agent* is then decomposed into three main subsystems: **Perception**, **Cognition**, and **Action**. Within *Cognition*, we identify key submodules: *memory*, *world model*, *emotional state*, *goals*, *reward*, *learning*, and *reasoning* processes (including "planning" and "decision-making" as special actions produced with reasoning). *Attention* is primarily handled within perception and cognition. Before presenting the formal loop, we summarize our symbols in Table 1.2.

In the following, based on the notations in Table 1.2, we present our proposed agent loop.

The Agent Loop

An intelligent agent operates in discrete time steps t, continuously interacting with its environment. At each step, the following processes occur:

1. **Environment State** ($s_t \in \mathcal{S}$):

   The environment is in state s_t.

2. **Perception** (P): The agent perceives the environment to generate observations o_t:

   $$o_t = \mathrm{P}(s_t, M_{t-1}),$$

   where M_{t-1} guides selective attention and filtering.

3. **Cognition** (C): Updates mental state and selects actions:

   $$(M_t, a_t) = \mathrm{C}(M_{t-1}, a_{t-1}, o_t).$$

   where M_t encapsulates different sub-states:

   $$M_t = \{M_t^{\text{mem}}, M_t^{\text{wn}}, M_t^{\text{emo}}, M_t^{\text{goal}}, M_t^{\text{rew}}, \cdots\}.$$

   Cognition consists of:

   - **Learning** (L): Updates mental state based on observations:

     $$M_t = \mathrm{L}(M_{t-1}, a_{t-1}, o_t).$$

   - **Reasoning** (R): Determines the next action:

     $$a_t = \mathrm{R}(M_t),$$

     which may be:
     - **External Actions**, directly affecting the environment.
     - **Internal Actions**, including:
       - *Planning*: Internal sequence of future actions.
       - *Decision-making*: Choosing the best action from available options.

4. **Action Execution** (E): Transforms action a_t into executable form:

   $$a'_t = \mathrm{E}(a_t).$$

5. **Environment Transition** (T): The environment responds to the agent's action:

   $$s_{t+1} = \mathrm{T}(s_t, a'_t).$$

In multi-agent scenarios, each agent i maintains individual states (M_t^i, a_t^i, o_t^i), and the environment collectively updates based on all agents' actions. At broader scales (AI societies or worlds, $\mathcal{W}$), agents interact within diverse social systems (e.g., economic, communication, or transportation), forming complex societal structures.

Figure 1.2 illustrates our agent framework, presenting the core concepts and different types of information or control flows among them. Until now, we have presented a brain-inspired agent framework that integrates biological insights into a formal *Perception–Cognition–Action* loop. By decomposing cognition into modules for memory, world modeling, emotion, goals, reward-based learning, and reasoning, we capture essential parallels with the human brain's hierarchical and reward-driven processes. Critically, *attention* is included in the loop to enable selective filtering based on internal states. Furthermore, *planning* and *decision-making* can be viewed as distinct internal (mental) actions that either refine internal representations or select external behaviors. Our framework naturally extends classical agent architectures, providing a multi-level structure that integrates emotional and rational processes as well as robust, reward-driven learning across short and long timescales.

Society and Social Systems. In many real-world scenarios, agents do not merely interact with a static environment but operate within a broader *society*, comprising various *social systems* such as financial markets, legal frameworks,

political institutions, educational networks, and cultural norms. These structures shape and constrain agents' behaviors by defining rules, incentives, and shared resources. For example, a financial system dictates how economic transactions and resource allocations occur, while a political system provides governance mechanisms and regulatory constraints. Together, these social systems create a layered context in which agents must adaptively learn, reason, and act—both to satisfy their internal goals and to comply (or strategically engage) with external societal rules. In turn, the actions of these agents feed back into the social systems, potentially altering norms, policies, or resource distributions.

A Formal Definition of Foundation Agents. Building on these insights and our vision of robust, adaptive intelligence, we now formally introduce the concept of a *Foundation Agent*. Unlike traditional agent definitions that focus primarily on immediate sensory-action loops, a Foundation Agent embodies sustained autonomy, adaptability, and purposeful behavior, emphasizing the integration of internal cognitive processes across diverse environments.

Definition of Foundation Agent

A **Foundation Agent** is an autonomous, adaptive intelligent system designed to **actively perceive** diverse signals from its environment, continuously **learn** from experiences to refine and update structured internal states (such as memory, world models, goals, emotional states, and reward signals), and **reason** about purposeful actions—both external and internal—to autonomously navigate toward complex, long-term objectives.

More concretely, a Foundation Agent possesses the following core capabilities:

1. **Active and Multimodal Perception:** It continuously and selectively perceives environmental data from multiple modalities (textual, visual, embodied, or virtual).
2. **Dynamic Cognitive Adaptation:** It maintains, updates, and autonomously optimizes a rich internal *mental state* (memory, goals, emotional states, reward mechanisms, and comprehensive world models) through **learning** that integrates new observations and experiences.
3. **Autonomous Reasoning and Goal-Directed Planning:** It proactively engages in sophisticated reasoning processes, including long-term planning and decision-making, to derive goal-aligned strategies.
4. **Purposeful Action Generation:** It autonomously generates and executes purposeful actions, which can be external (physical movements, digital interactions, communication with other agents or humans) or internal (strategic planning, self-reflection, optimization of cognitive structures), systematically shaping its environment and future cognition to fulfill complex objectives.
5. **Collaborative Multi-Agent Structure:** It can operate within multi-agent or agent society structures, collaboratively forming teams or communities of agents that collectively accomplish complex tasks and goals beyond individual capabilities.

This definition highlights three essential pillars distinguishing Foundation Agents: *sustained autonomy* (operating independently toward long-term goals without step-by-step human intervention), *adaptive learning* (evolving internal representations continually over diverse experiences), and *purposeful reasoning* (generating actions guided by complex, internally maintained goals and values). Foundation Agents thus represent a fundamental shift from traditional agents by integrating deep cognitive structures, multimodal processing capabilities, and proactive, sustained self-optimization, enabling them to function effectively across a wide range of environments and domains.

Unlike classical definitions, which often frame agents primarily in terms of simple perception–action loops ("perceive and act" [20]), our notion of Foundation Agents emphasizes the depth and integration of internal cognitive processes. Foundation Agents not only perceive their environment and perform immediate actions but also possess an evolving, goal-oriented cognition—continuously adapting memory structures, world models, emotional and reward states, and autonomously refining their strategies through reasoning. This internal cognitive richness allows Foundation Agents to autonomously decompose complex, abstract goals into actionable tasks, strategically explore their environments, and dynamically adjust their behavior and cognitive resources. Our unified **perception–cognition–action** framework thus accommodates and explicitly models these sophisticated cognitive capabilities, recognizing internal (mental) actions on par with external (physical or digital) interactions, facilitating a broad range of embodiments, from physical robots to software-based or purely textual intelligent agents.

1.3.2 Biological Inspirations

Although our agent model is fundamentally computational, each submodule draws inspiration from well-studied biological counterparts in the human brain. Below, we discuss these analogies in a manner that highlights both the neuroscientific basis and the flexibility afforded by AI implementations.

Memory (Hippocampus and Neocortex). Decades of neuroscience research have linked the hippocampus to episodic memory formation, while cortical regions are known to house semantic and procedural knowledge [21, 22]. In humans, these memory subsystems cooperate to manage both short-term encoding and long-term consolidation. Our memory component, M_t^{mem}, similarly aims to capture multi-scale learning by storing recent experiences and knowledge. This can be realized through either neural network weights (long-term) or explicit buffers (short-term), thereby mirroring the hippocampal–cortical interplay.

World Model (Predictive Processing). A prominent theory in cognitive neuroscience holds that the cortex operates as a predictive machine, continually comparing incoming sensory data with generated expectations [23, 19]. The world model M_t^{wm} reflects this idea by maintaining an internal representation of how the environment evolves over time. Just as cortical circuits integrate multisensory data to update these internal models, our framework allows M_t^{wm} to be refined upon each new observation and relevant reward or emotional cues, offering a Bayesian or free-energy perspective on environmental dynamics.

Emotion (Limbic System). Emotions, mediated by structures like the amygdala, hypothalamus, and limbic system, significantly modulate attention, learning rates, and decision-making thresholds [24, 25]. By introducing an emotion component M_t^{emo}, our model captures how internal valence or arousal states can shift an agent's focus and behavior. Although computational "emotions" are neither fully analogous to biological affect nor conscious feelings, they can guide adaptive heuristics—such as prioritizing urgent goals or responding quickly to perceived threats.

Goals and Reward (Prefrontal & Subcortical Circuits). Humans excel at forming abstract, long-term goals, an ability often associated with prefrontal cortex function [26, 27]. In parallel, subcortical circuits—particularly dopaminergic pathways—drive reinforcement signals that shape motivation and habit learning [28]. Our agent includes M_t^{goal} for storing objectives and M_t^{rew} for encoding reward signals, thus enabling a continuous feedback loop where goal formation and reward-based adaptation reinforce each other. This mechanism allows for planned action sequences, tool usage, and more nuanced social interactions.

Reasoning, Planning, and Decision-Making (Prefrontal Cortex). Finally, the human prefrontal cortex integrates information from memory, sensory inputs, emotions, and reward pathways to carry out higher-order cognitive processes—such as logical reasoning, planning, and executive control [29, 30]. In our agent framework, these capabilities are subsumed by the reasoning sub-function, which—through modules like PlanFn and Decide—selects and executes actions (whether physical or purely mental). By distinguishing planning from on-the-fly decision-making, we capture how the agent can simulate future scenarios, weigh outcomes, and then commit to a course of action, akin to the flexible orchestration observed in prefrontal circuits.

1.3.3 Connections to Existing Theories

Beyond these explicit neurobiological parallels, our architecture resonates with several important theories in AI, cognitive science, and neuroscience.

Classic Perception–Cognition–Action Cycle. We extend the traditional sense–think–act cycle outlined by [20], incorporating explicit mechanisms for attention (in P), learning and emotion (in C), and reward signals that persist over time. This explicitness makes it easier to analyze how an agent's internal states and prior actions shape subsequent perception and cognition.

Minsky's "Society of Mind". [17] argued that intelligence arises from an ensemble of specialized "agents" within a mind. Our submodules—$\mathrm{C}_{\text{mem}}, \mathrm{C}_{\text{wm}}, \mathrm{C}_{\text{emo}}, \mathrm{C}_{\text{goal}}, \mathrm{C}_{\text{rew}}$—echo this decomposition, distributing key functions (memory, prediction, emotional evaluation, goal-setting, etc.) across separate yet interacting components. In a broader "society" context, each agent (or sub-agent) could coordinate cooperatively or competitively, much like Minsky's internal agencies. Recent work on natural language-based societies of mind [31] supports that agentic systems can be represented using the original society-of-mind theory, and could incorporate social structures and economic models among agents.

Buzsáki's Inside-Out Perspective. Neuroscientists [18] contend that the brain actively constructs and updates its perception instead of merely receiving inputs. In our model, M_{t-1}—including emotional states, reward signals, and goals—directly influences the perception map P. This supports the inside-out stance that an agent's internal context drives the way it samples and interprets the environment, rather than passively reacting to it.

Partially observable Markov decision process (POMDP). Our framework can be viewed as a generalization of the classical Partially Observable Markov Decision Process (POMDP) in several ways. First, whereas a POMDP specifies a probabilistic transition function $P(s_{t+1} \mid s_t, a_t)$ over a (possibly finite) state space, we retain an environment transition T without restricting it to a purely probabilistic or finite form, allowing for arbitrary or even deterministic mappings. Second, in the standard POMDP setting, reward is typically defined as a scalar function of (s_t, a_t) (possibly discounted over time). By contrast, we place reward signals *inside* the agent's mental state (M_t^{rew}), letting them depend on—and co-evolve with—goals, emotion, and the world model rather than enforcing a single externally defined objective. Third, while POMDP agents generally select actions by maximizing an expected return (value function), our *reasoning* sub-process is broader. It accounts for memory, emotion, and other mental-state factors, accommodating heuristic or socially driven decisions rather than strictly value-based choices. Finally, a POMDP does not explicitly define cognitive submodules such as memory or emotion—these must be collapsed into a monolithic "belief state". In our framework, each sub-component (memory, world model, emotion, goals, reward) is explicitly modeled and updated, mirroring biologically inspired views of cognition. Hence, although our approach *recovers* the POMDP formulation as a special case (by enforcing a probabilistic T, a scalar reward, and a minimal mental state), it admits a richer variety of environment transitions, internal states, and decision mechanisms.

Active Inference and the Bayesian Brain. Active inference, a unifying framework advanced by [19], suggests that agents continually update internal generative models to minimize prediction error (or "free energy"). Our use of M^{wm} and M^{rew}, together with planning and decision-making modules, can be interpreted in Bayesian terms. The agent attempts to reduce surprise by aligning its world model with new data and by choosing actions that conform to predicted (or desired) outcomes.

Biological Plausibility & Generality. While the mapping between brain circuits and agent submodules is made at a high level, it offers an approach that is at once *biologically inspired* and *modularly agnostic*. Memory, emotion, goals, and reward can each be implemented by various AI paradigms—symbolic methods, neural networks, or hybrid approaches—thus preserving flexibility. By integrating these key ideas from neuroscience, cognitive science, and AI, we arrive at a general framework that captures the essential properties of intelligent behavior without overconstraining implementation details.

1.4 Navigating This Survey

This survey is structured to provide a comprehensive, modular, and interdisciplinary examination of intelligent agents, drawing inspiration from cognitive science, neuroscience, and other disciplines to guide the next wave of advancements in AI. While many existing surveys [32, 33, 34, 35, 36, 37, 38, 39, 40] offer valuable insights into various aspects of agent research, we provide a detailed comparison of their focal points in Table 1.3. Our work distinguishes itself by systematically comparing biological cognition with computational frameworks to identify synergies, gaps, and opportunities for innovation. By bridging these domains, we aim to provide a unique perspective that highlights not only where agents excel but also where significant advancements are needed to unlock their full potential.

Table 1.3: Summary of existing reviews with different focal points. • indicates primary focus while ○ indicates secondary or minor focus.

Survey	Cognition	Memory	World Model	Reward	Action	Self Evolve	MultiAgent	Safety
Zhang et al. [39]	•	•	○	○	○	•	○	○
Guo et al. [38]	•	•	○	○	○	•	•	○
Yu et al. [40]	•	•	○	○	•	○	•	•
Wang et al. [35]	•	•	○	○	•	○	•	○
Masterman et al. [37]	•	•	○	○	•	○	•	○
Xi et al. [34]	•	•	○	○	•	•	•	•
Huang et al. [33]	•	•	○	•	•	•	•	•
Durante et al. [32]	•	•	○	•	•	•	•	•
This Manuscript	•	•	•	•	•	•	•	•

The survey is divided into four key parts:

- In **Part I**: Modular Design of Intelligent Agents, we introduce the core modules of agents, including the cognition module, which serves as the "brain" of the agent; the perception systems for interpreting sensory input; as well as the action systems for interacting with the external world. Within the cognition system, we further discuss the memory, world modeling, emotion, goal, and reward systems, analyzing their current progress, limitations, and research challenges.

- In **Part II**: Self-Enhancement in Intelligent Agents, we shift focus to the capability of agents to evolve and optimize themselves. We explore mechanisms like adaptive learning, self-reflection, and feedback-driven improvement, inspired by the human ability to grow and refine skills over time. This part also addresses the importance of dynamic memory systems and continuous knowledge integration for agents to remain relevant and effective in changing environments.
- In **Part III**: Collaborative and Evolutionary Intelligent Systems, we examine how agents interact with each other and their environments to solve complex, large-scale problems. We discuss multi-agent systems, highlighting their applications in fields such as robotics, medical systems and scientific discovery. This part explores multi-agent system topologies and agent protocol, tracing the evolution of communication and collaboration from static to dynamic frameworks. We align agents with human collaboration paradigms, examining how interaction patterns shape the co-evolution of intelligence and how multi-agent systems adapt their decision-making in various collaborative settings to solve complex challenges through collective intelligence.
- Finally, in **Part IV**: Building Safe and Beneficial AI, we provide a comprehensive analysis of the security landscape for LLM-based agents. We introduce a framework categorizing threats as intrinsic or extrinsic. Intrinsic vulnerabilities arise from within the agent's architecture: the core LLM "brain", and the perception and action modules that enable interactions with the world. Extrinsic risks stem from the agent's engagement with memory systems, other agents, and the broader environment. This part not only formalizes and analyzes these vulnerabilities, detailing specific attack vectors like jailbreaking and prompt injection, but also reviews a range of defense mechanisms. Moreover, we explore future directions, including superalignment techniques and the scaling law of AI safety—the interplay between capability and risk.

By weaving together these threads, our survey aims to provide a holistic perspective on the current state of intelligent agents and a forward-looking roadmap for their development. Our unique focus on integrating cognitive science insights with computational design principles positions this survey as a foundational resource for researchers seeking to design agents that are not only powerful and efficient but also adaptive, ethical, and deeply aligned with the complexities of human society.

Part I

Core Components of Intelligent Agents

Chapter 2

Cognition

Human cognition represents a sophisticated information processing system that enables perception, reasoning, and goal-directed behavior through the orchestrated operation of multiple specialized neural circuits [98]. This cognitive architecture operates through mental states, which serve as the foundation where learning and reasoning occur. The remarkable ability to process information across different levels of abstraction and adapt to novel situations is a crucial inspiration for LLM agents [27].

The cognitive system exhibits several fundamental architectural properties reflected in Figure 1.1. First, learning functions across different mental state spaces: it can occur holistically across frontal lobes (supporting executive control and cognition) and temporal lobes (responsible for language, memory, and auditory processing), or focus on specific aspects for targeted improvement as shown by the varied research levels in the figure. Second, reasoning emerges in distinct patterns: it can follow structured templates for systematic problem-solving supported by logical reasoning and cognitive flexibility in the frontal lobes, or appear in unstructured forms for flexible thinking, particularly evident in decision-making and executive control functions. Third, the system demonstrates remarkable adaptability, continuously updating its mental states through experience while leveraging both supervised feedback (as in adaptive error correction in the cerebellum) and unsupervised environmental statistics, reflected in the different exploration stages of various cognitive functions shown in the figure [99].

These cognitive processes are supported by a modular organization, composed of distinct but interconnected components that form a cohesive system [100]. These modules include perception systems that transform raw sensory data into meaningful representations, memory systems that provide the substrate for storing and retrieving information, world models that support future scenario simulation, reward signals that guide refinement of behavior through reinforcement, emotion systems that modulate attention and resource allocation, reasoning systems that formulate decisions, and action systems that translate decisions into environmental interactions.

While human cognition implements these properties through complex neural architectures shaped by evolution, LLM agents attempt to approximate similar functions using large-scale neural models and algorithmic techniques. Understanding this biological-artificial parallel is crucial for developing more capable agents [101], as it highlights both the achievements and limitations of current systems compared to human cognition. Significant differences remain in areas such as adaptability, generalization, and contextual understanding.

In this section, we first explore **Learning**, examining both the spaces where it occurs within mental states and the specific objectives it serves. Subsequently, we investigate **Reasoning**, analyzing both structured and unstructured approaches, before concluding with a dedicated exploration of planning capabilities as a special reasoning action.

2.1 Learning

Learning represents the fundamental process through which intelligent agents transform experiences into knowledge within their mental states. This transformation occurs across different cognitive spaces, from holistic updates across the full mental state to refinement of specific cognitive components. The scope of learning encompasses remarkable capacities that serve different objectives: enhancing perceptual understanding, improving reasoning capabilities, and developing richer world understanding.

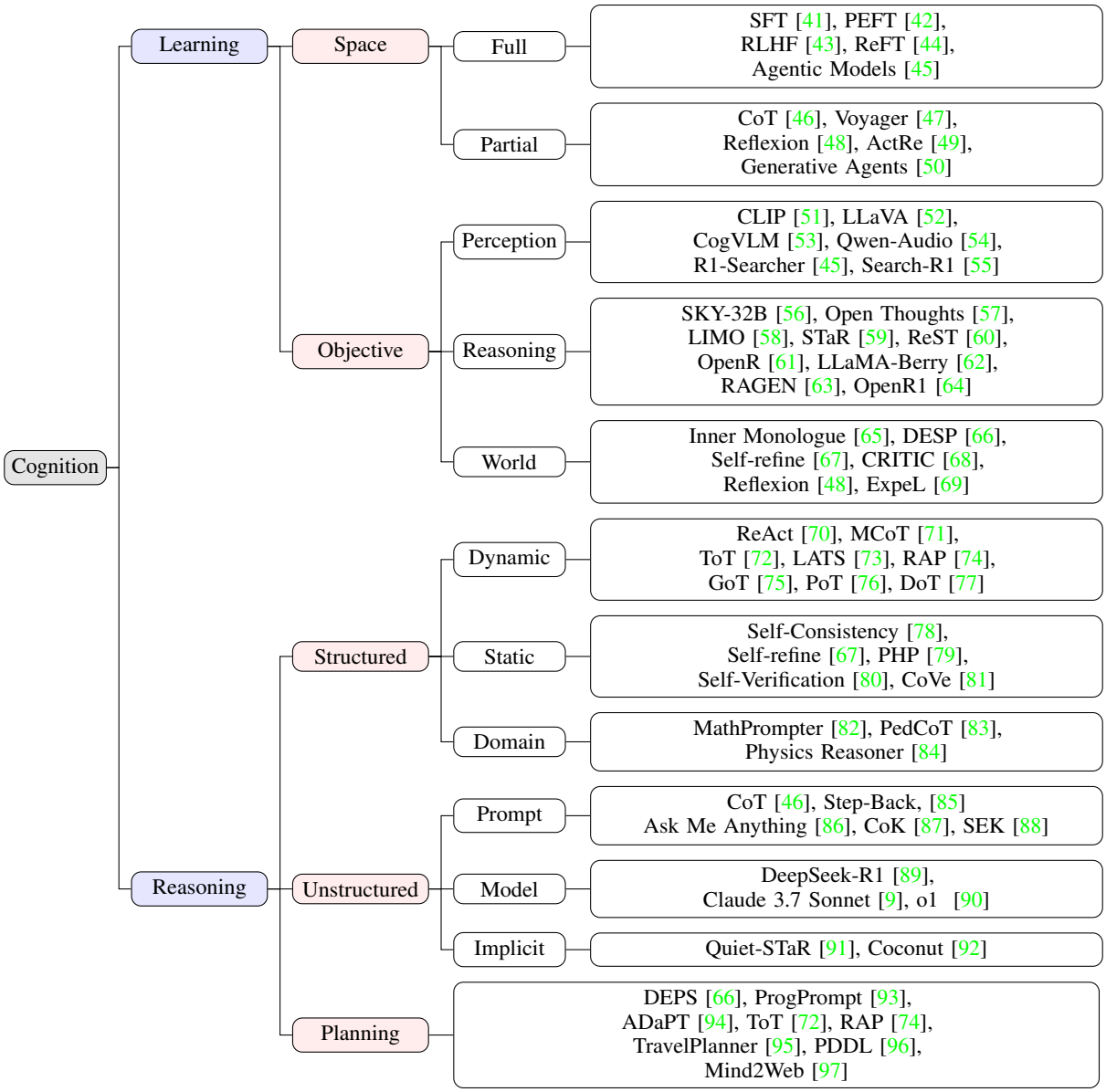


Figure 2.1: Illustrative Taxonomy of Cognition system, including learning and reasoning paradigm.

Human learning operates across multiple spaces and objectives through the brain's adaptable neural networks. The brain coordinates learning across its entire network through integrated systems: the hippocampus facilitates rapid encoding of episodic experiences, the cerebellum supports supervised learning for precise motor skills, the basal ganglia enable reinforcement learning through dopaminergic reward signals, and cortical regions facilitate unsupervised pattern extraction [99]. At more focused levels, specific neural circuits can undergo targeted adaptation, allowing for specialized skill development and knowledge acquisition. These systems work together on different timescales, ranging from immediate responses to lifelong development, while being influenced by factors like attention, emotions, and social environment [27].

LLM agents, while fundamentally different in architecture, implement analogous learning processes across their mental state spaces. At the comprehensive level, they acquire broad knowledge through pre-training on massive datasets, demonstrating a form of unsupervised learning. At more focused levels, they refine specific capabilities through parameter-updating mechanisms like supervised fine-tuning and reinforcement learning. Uniquely, they also demonstrate in-context learning capabilities, adapting to novel tasks without parameter changes by leveraging context

within their attention window: a capability that mirrors aspects of human working memory but operates through fundamentally different mechanisms.

The comparison between human and artificial learning systems provides valuable insights for developing more capable, adaptive agents. Human learning demonstrates notable characteristics in efficiency, contextualization, and integration with emotional systems, while LLM-based approaches show distinct capabilities in processing large datasets, representing formal knowledge, and synthesizing information across domains. These complementary strengths suggest productive directions for research. As we explore the foundations of learning, we first examine the spaces where learning occurs within mental states, followed by an analysis of the specific objectives that drive learning processes.

Table 2.1: Summary of Learning Methods with Different State Modifications. • indicates primary impact while ○ indicates secondary or no direct impact.

Method	Model	Perception	Reasoning	Memory	Reward	World Model
Voyager [47]	○	○	○	•	○	○
Generative Agents [50]	○	○	○	•	○	○
Learn-by-interact [102]	•	○	○	•	○	○
RAGEN [63]	•	○	•	○	•	○
DigiRL [103]	•	○	•	○	•	○
R1-Searcher [45]	•	•	•	○	•	○
RewardAgent [104]	•	○	○	○	•	○
Text2Reward [105]	○	○	○	○	•	○
ARAMP [106]	•	○	○	○	•	○
ActRe [49]	•	○	•	○	○	•
WebDreamer [107]	○	○	○	○	○	•
RAP [74]	○	○	○	○	○	•
AutoManual [108]	○	○	○	•	○	•

2.1.1 Learning Space

The learning approaches in LLM agents represent a structured, data-driven paradigm in contrast to the exploratory, emotionally-driven learning observed in humans. While human learning often involves active curiosity, motivation, and emotional reinforcement, LLM-based agents typically learn through more formalized processes, such as parameter updates during training or structured memory formation during exploration. Current agent architectures attempt to bridge this gap by implementing mechanisms that simulate aspects of human learning while leveraging the strengths of computational systems.

Learning within an intelligent agent occurs across different spaces, encompassing both the underlying model θ and mental states M, where the former fundamentally supports the capabilities and limitations of the latter. Formally, we define an intelligent agent's internal state as a tuple $\mathcal{I} = (\theta, M)$ that includes both the model parameters and mental state components. The mental state can be further decomposed into different structures as we illustrated in 1.2:

$$M = \{M^{mem}, M^{wm}, M^{emo}, M^{goal}, M^{rew}\} \tag{2.1}$$

where M^{mem} represents memory, M^{wm} denotes world model, M^{emo} indicates emotional state, M^{goal} represents goals, and M^{rew} represents reward signals.

Modifications to the underlying model can be viewed as full mental state learning, as they fundamentally alter the agent's capabilities. While model-level modifications can affect different mental states to varying degrees, changes to the model's context window or external structures tend to focus on specific mental state components. For instance, learning experiences and skills from the environment primarily influence memory, while leveraging the LLM's inherent predictive capabilities enhances the world model.

Full Mental State Learning Full mental state learning enhances the capabilities of an agent through comprehensive modifications to the underlying model θ, which in turn affects all components of the mental state M. This process begins with pre-training, which establishes the foundation of language models by acquiring vast world knowledge, analogous to how human babies absorb environmental information during development, though in a more structured and extensive manner.

Post-training techniques represent the cornerstone for advancing agent capabilities. Similar to how human brains are shaped by education, these techniques while affecting the entire model, can emphasize different aspects of cognitive

development. Specifically, various forms of tuning-based learning enable agents to acquire domain-specific knowledge and logical reasoning capabilities. Supervised Fine-Tuning (SFT) [41] serves as the fundamental approach where models learn from human-labeled examples, encoding knowledge directly into the model's weights. For computational efficiency, Parameter-Efficient Fine-Tuning (PEFT) methods have emerged. Adapter-BERT [42] introduced modular designs that adapt models to downstream tasks without modifying all parameters, while Low-Rank Adaptation (LoRA) [109] achieves similar results by decomposing weight updates into low-rank matrices, adjusting only a small subset of effective parameters.

Some agent capabilities are closely connected to how well they align with human preferences, with alignment-based learning approaches modifying the model to reshape aspects of the agent's underlying representations. Reinforcement learning from human feedback (RLHF) [110] aligns models with human values by training a reward model on comparative judgments and using this to guide policy optimization. InstructGPT [43] demonstrated how this approach could dramatically improve consistency with user intent across diverse tasks. Direct Preference Optimization (DPO) [111] has further simplified this process by reformulating it as direct preference learning without explicit reward modeling, maintaining alignment quality while reducing computational complexity.

Reinforcement learning (RL) presents a promising pathway for specialized learning in specific environments. RL has shown particular promise in enhancing reasoning capabilities, essentially enabling the agent's underlying model to learn within the space of thought. Foundational works such as Reinforcement Fine-Tuning (ReFT) [44] enhance reasoning through fine-tuning with automatically sampled reasoning paths under online reinforcement learning rewards. DeepSeek-R1 [89] advances this approach through rule-based rewards and Group Relative Policy Optimization (GRPO) [112], while Kimi k1.5 [113] combines contextual reinforcement learning with optimized chain-of-thought techniques to improve both planning processes and inference efficiency. In specific environments, modifying models to enhance agents' understanding of actions and external environments has proven effective, as demonstrated by DigiRL [103], which implements a two-stage reinforcement learning approach enabling agents to perform diverse commands on real-world Android device simulators.

Recent works have attempted to integrate agent action spaces directly into model training [45, 55], enabling learning of appropriate actions for different states through RL or SFT methods. This integration fundamentally affects the agent's memory, reward understanding, and world model comprehension, pointing toward a promising direction for the emergence of agentic models.

Partial Mental State Learning While full mental state learning through model modifications provides comprehensive capability updates, learning focused on particular components of an agent's mental state M represents another essential and often more efficient approach. Such partial mental state learning can be achieved either through targeted model updates or through in-context adaptation without parameter changes.

In-Context Learning (ICL) illustrates how agents can effectively modify specific mental state components without modifying the entire model. This mechanism allows agents to adapt to new tasks by leveraging examples or instructions within their context window, paralleling human working memory's role in rapid task adaptation. Chain-of-Thought (CoT) [46] demonstrates the effectiveness of this approach, showing how agents can enhance specific cognitive capabilities while maintaining their base model parameters unchanged.

The feasibility of partial mental state learning is evidenced through various approaches targeting different components such as memory (M^{mem}), reward (M^{rew}), and world model (M^{wm}). Through normal communication and social interaction, Generative Agents [50] demonstrate how agents can accumulate and replay memories, extracting high-level insights to guide dynamic behavior planning. In environmental interaction scenarios, Voyager [47] showcases how agents can continuously update their skill library through direct engagement with the Minecraft environment, accumulating procedural knowledge without model retraining. Learn-by-Interact [102] further extends this approach by synthesizing experiential data through direct environmental interaction, eliminating the need for manual annotation or reinforcement learning frameworks. Additionally, agents can learn from their mistakes and improve through reflection, as demonstrated by Reflexion [48], which guides agents' future thinking and actions by obtaining textual feedback from repeated trial and error experiences.

Modifications to reward and world models provide another example of partial mental state learning. ARMAP [106] refines environmental reward models by distilling them from agent action trajectories, providing a foundation for further learning. AutoMC [114] constructs dense reward models through environmental exploration to support agent behavior. Meanwhile, [107] explicitly leverages LLMs as world models to predict the impact of future actions, effectively modifying the agent's world understanding (M^{wm}). ActRe[49] builds upon the language model's inherent world understanding to construct tasks from trajectories, enhancing the agent's capabilities as both a world model and reasoning engine through iterative training.

2.1.2 Learning Objective

The learning process of intelligent agents manifests across all aspects of their interaction with the environment. At the input level, agents learn to better perceive and parse environmental information; at the processing level, agents learn how to conduct effective reasoning based on existing knowledge or reasoning capabilities; at the comprehension level, agents form and optimize their understanding of the world through continuous interaction. This multi-level learning objective framework enables agents to evolve continuously across different dimensions, allowing them to better handle complex and dynamic task environments.

Learning for Better Perception The ability to effectively perceive and process information from the environment is fundamental to agent intelligence. To enhance perceptual capabilities, agents employ two primary learning approaches: expanding multimodal perception and leveraging retrieval mechanisms.

Multimodal perception learning enables agents to process and integrate diverse sensory inputs, similar to human multi-sensory integration but unconstrained by biological limitations. This capability has evolved significantly through advances like CLIP [51], which pioneered the alignment of visual and linguistic representations in shared embedding spaces. Building on this foundation, models like LLaVA [52] enhanced visual perception by training specialized projectors on image-text pairs, while CogVLM [53] advanced visual reasoning through unified representational architectures.

The expansion of perceptual modalities continues across multiple sensory domains. In audio processing, Qwen-Audio [54] demonstrates the unified encoding of diverse acoustic information, from speech to environmental sounds. Recent work by [115] has even ventured into tactile perception, developing datasets that align touch, vision, and language representations. These advances enable agents to engage more comprehensively with both physical and digital environments.

Agents also learn to enhance their observational capabilities through retrieval mechanisms. Unlike human perception, which is constrained by immediate sensory input, agents can learn to access and integrate information from vast external knowledge repositories. Retrieval-augmented approaches like RAG [116] enhance perceptual understanding by connecting immediate observations with relevant stored knowledge.

Recent work on retrieval-based agents demonstrates the potential for enhancing active information acquisition capabilities. Search-o1 [117] guides reasoning models to learn active retrieval through prompting, thereby expanding their knowledge boundaries. Taking this further, R1-Searcher [45] and Search-R1 [55] directly incorporate retrieval capabilities into the model, enabling autonomous information retrieval during the reasoning process. These advances suggest a promising direction for improving agent perception: enhancing model-level active perception capabilities to enrich the foundation for decision-making. This approach may represent a significant avenue for future agent development.

Learning for Better Reasoning Reasoning serves as a critical bridge between an agent's mental state and its actions, making the ability to reason effectively and the development of reasoning capabilities essential for intelligent agents. The foundation of reasoning in modern agents stems from two key elements: the rich world knowledge embedded in their underlying models, and the robust logical frameworks supported either internally or through context structuring. This makes learning for better reasoning a vital objective in agent development.

The development of reasoning capabilities is demonstrated through several key phenomena. First, high-quality reasoning data directly enhances model reasoning ability; second, such high-quality data often requires verification or reward models for effective curation; and third, direct reinforcement learning on foundation models can spontaneously manifest reasoning capabilities.

The importance of reasoning in agent development has been re-emphasized following the release of the o1 series. A common approach involves collecting and distilling data from open/closed-source reasoning models. For instance, SKY-32B [56] distilled data from QWQ-32B [118] to train a 32B reasoning model at a cost of $450. Similarly, Open Thoughts [57] trained Bespoke-Stratos-32B at a low cost by distilling and synthesizing datasets from R1. These studies demonstrate that even without complex algorithmic design, using reasoning data to perform Supervised Fine-Tuning (SFT) on base models can effectively activate reasoning capabilities.

Another crucial insight regarding data quality is that highly structured reasoning data more effectively enables agents and language models to learn reasoning processes. Notably, LIMO [58] demonstrated that powerful reasoning models could be built with extremely few data samples by constructing long and effective reasoning chains for complex reasoning tasks. This insight stems from their observation that language models inherently possess sufficient knowledge for reasoning but require high-quality reasoning paths to activate these capabilities. Supporting this view, Li et al.

[119] revealed that both Long CoT and Short CoT fundamentally teach models to learn reasoning structures rather than specific content, suggesting that automated selection of high-quality reasoning data may become an important future direction.

One viable exploration approach involves first conducting extensive searches, and then using verifiable environments or trainable reward models to provide feedback on reasoning trajectories, thereby filtering out high-quality reasoning data. This approach has led to several families of techniques that leverage different feedback mechanisms to improve reasoning capabilities.

The first category follows the bootstrap paradigm exemplified by STaR [59] and its variants, which implement techniques where models generate step-by-step rationales and iteratively improve through fine-tuning on successful reasoning paths. This family includes Quiet-STaR [91], V-STaR [120], and rStar-Math [121], with the latter specifically enhancing mathematical reasoning through reinforcement learning principles. By iteratively selecting correct reasoning paths for training, these methods achieve self-improvement through successive refinement cycles.

The second category extends this paradigm by more explicitly incorporating reinforcement learning principles. The ReST family, beginning with the original ReST [60] introducing reinforced self-training, performs multiple attempts (typically 10) per sample and creates new training datasets from successful reasoning instances. ReST-EM [122] enhances the approach with expectation maximization, while ReST-MCTS [122] further integrates Monte Carlo Tree Search to enable improved reasoning capabilities through more sophisticated exploration strategies.

Several approaches have introduced Policy Reward Models (PRMs) to provide quality feedback on reasoning paths. Methods like OpenR [61] and LLaMA-Berry [62] model reasoning tasks as Markov Decision Processes (MDPs) and leverage tree search to explore diverse reasoning paths while using PRMs for quality assessment. In domain-specific applications, methods like rStar-Math [121] and DeepSeekMath [112] have demonstrated success in mathematical problem-solving through multi-round self-iteration and balanced exploration-exploitation strategies. For code generation, o1-Coder [123] leverages MCTS to generate code with reasoning processes, while Marco-o1 [123] extends this approach to open-ended tasks. These implementations highlight how the synergy between MCTS and PRM achieves effective reasoning path exploration while maintaining solution quality through fine-grained supervision.

Beyond data-driven approaches, reinforcement learning (RL) has demonstrated remarkable success in enhancing language models' reasoning capabilities, as evidenced by recent breakthroughs like DeepSeek R1 [89] and Kimi-K-1.5 [113]. The foundation of RL for LLMs can be traced to several pioneering frameworks: ReFT [44] introduced a combination of supervised fine-tuning and online reinforcement learning, while VeRL [124] established an open-source framework supporting various RL algorithms for large-scale models up to 70B parameters. RFT [125] further demonstrated the effectiveness of reward-guided optimization in specific reasoning tasks.

Building upon these foundations, subsequent works have explored diverse applications and improvements. OpenR1 [64] and RAGEN [63] extended RL techniques to enhance general reasoning capabilities, while specialized implementations like SWE-Gym [126] demonstrated success in software engineering tasks. Notably, DigiRL [103] introduced novel approaches for digital-world agent enhancement.

Recent advances have further integrated RL with tool usage and reasoning. Qwen-QwQ-32B [118] employs reinforcement learning and a general reward mechanism to incorporate tool calling into the reasoning process, enabling the seamless use of arbitrary tools during reasoning and achieving agent-like capabilities directly within the model. Similarly, RAGEN [63] focuses on multi-step agentic scenarios, establishing a framework for agent reinforcement learning in complex environments. These developments suggest an increasing convergence between model training and agent development, potentially leading to more integrated and capable intelligent systems. These implementations highlight how RL can effectively improve model performance while reducing dependence on large-scale annotated datasets, particularly in complex reasoning scenarios.

Learning for World Understanding A critical aspect of agent intelligence is the ability to understand how the world operates through direct interaction and experience accumulation. This understanding encompasses how the environment responds to different actions and the consequences these actions bring. Through continuous interaction with their environment, agents can build and refine their *memory*, *reward understanding*, and *world model*, learning from both successes and failures to develop a more comprehensive grasp of their operational domain.

Recent research has revealed diverse approaches to experiential learning for world understanding. At the foundational level, Inner Monologue [65] demonstrates how agents can accumulate basic environmental knowledge through continuous interaction. Similarly, Learn-by-Interact [102] shows that meaningful understanding can emerge from direct environmental engagement without explicit reward mechanisms. More sophisticated approaches are exemplified by DESP [66] and Voyager [47] in the Minecraft environment, where agents not only gather experiences but also actively process them: DESP through outcome analysis and Voyager through dynamic skill library expansion.

The processing and utilization of accumulated experiences have been further systematized through advanced frameworks. Generative Agents [50] introduces sophisticated memory replay mechanisms, enabling agents to extract high-level insights from past interactions. This systematic approach is enhanced by Self-refine [67] and Critic [68], which implement structured cycles of experience evaluation and refinement.

The optimization of reward understanding through environmental interaction has emerged as another crucial aspect of world understanding. Text2Reward [105] demonstrates how agents can continuously refine reward functions through human feedback, better aligning them with task objectives and environmental characteristics. Similarly, AutoManual [108] builds behavioral guidelines through sustained interaction, developing reward-verified protocols that provide a foundation for understanding environmental rewards and decision-making. These interaction-based optimization mechanisms enable agents to better comprehend environmental dynamics and generate more precise reward signals, ultimately enhancing their adaptability and decision-making capabilities in complex, dynamic environments.

Building on these foundations, RAP [74] represents a significant advancement by conceptualizing reasoning as planning with a world model. By repurposing LLMs as both reasoning agents and world models, RAP enables agents to simulate the outcomes of potential actions before committing to them, facilitating more effective planning through Monte Carlo Tree Search. This approach allows agents to strategically explore the reasoning space with a proper balance between exploration and exploitation.

Further innovations in leveraging world models for agent learning include ActRe [127], which reverses the typical reasoning-action sequence by first performing actions and then generating post-hoc explanations. This capability to rationalize actions demonstrates LLMs' inherent understanding of world dynamics, enabling autonomous trajectory annotation and facilitating contrastive self-training.

The importance of cognitive maps in world understanding is highlighted by [128], who show that structured mental representations inspired by human cognition significantly enhance LLMs' extrapolation capabilities in novel environments. These cognitive maps not only improve planning but also exhibit human-like characteristics such as structured mental simulation and rapid adaptation.

In web-based environments, recent work by [107] and [129] demonstrates that LLMs can function as effective world models for anticipating the outcomes of web interactions. By simulating potential state changes before executing actions, these approaches enable safer and more efficient decision-making, particularly in environments where actions may be irreversible.

Through systems like Reflexion [48] and ExpeL [69], agents have advanced experiential learning by autonomously managing the full cycle of experience collection, analysis, and application, enabling them to learn effectively from both successes and failures.

These developments collectively illustrate how world models are becoming increasingly central to agent learning systems, providing a foundation for understanding environmental dynamics and enabling more effective planning, reasoning, and decision-making in complex, interactive environments.

2.2 Reasoning

Reasoning represents the key to intelligent behavior, transforming raw information into actionable knowledge that drives problem-solving and decision-making. For both humans and artificial agents, it enables logical inference, hypothesis generation, and purposeful interaction with the world. In human cognition, reasoning emerges through multiple strategies: deductive reasoning applies general rules to specific cases, inductive reasoning builds generalizations from particular instances, and abductive reasoning constructs plausible explanations from incomplete data [130, 131]. These processes are augmented by heuristics—mental shortcuts that streamline decision-making under uncertainty—and are continuously refined through environmental feedback, ensuring that reasoning remains grounded in reality and adaptive to change.

For LLM-based agents, reasoning serves a parallel role, elevating them beyond reactive systems to proactive entities capable of sophisticated cognition. Through reasoning, these agents process multimodal inputs, integrate diverse knowledge sources, and formulate coherent strategies to achieve objectives. The environment plays a dual function: supplying information that fuels reasoning and serving as the proving ground where reasoned actions are tested, creating a feedback loop that enables agents to validate inferences and learn from errors.

In LLM-based agents, reasoning can be formally defined as the process of action selection based on mental states, representing a crucial bridge between perception and action. More precisely, given a mental state Mt at time t, reasoning can be formalized as a function R(Mt) $\rightarrow$ at, where at represents the selected action. This process operates across

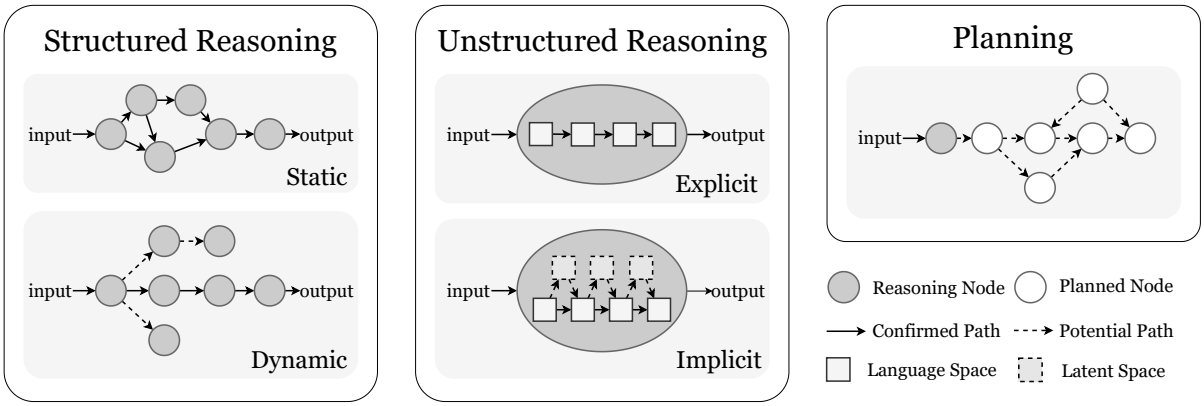


Figure 2.2: Comparison of reasoning paradigms in LLM-based agents.

various environments—textual, digital, and physical worlds—where completing a task typically requires either a single reasoning step or a composition of multiple reasoning actions.

The composition of reasoning actions naturally leads to two distinct approaches: structured and unstructured reasoning. Structured reasoning (R_s) can be formalized as an explicit composition $R_s = R_1 \circ R_2 \circ \ldots \circ R_n$, where each R_i represents a discrete reasoning step with clear logical dependencies. In contrast, unstructured reasoning (R_u) takes a more holistic form $R_u = f(M_t)$, where the composition remains implicit and flexible, allowing for dynamic adaptation to context. This dual framework mirrors human cognition, where structured reasoning parallels our explicit logical deduction processes, while unstructured reasoning reflects our capacity for intuitive problem-solving and pattern recognition.

The environment plays a crucial role in this formalization, serving both as a source of observations o_t that influence mental state updates ($M_t = L(M_{t-1}, a_{t-1}, o_t)$) and as a testing ground for reasoning outcomes. This creates a continuous feedback loop where reasoning not only drives action selection but also influences how the agent's mental state evolves, enabling iterative refinement of reasoning strategies through experience.

In this section, we will examine how these reasoning approaches manifest in practice. We begin with structured reasoning, which emphasizes systematic problem decomposition and multi-step logical chains. We then explore unstructured reasoning, which allows for flexible response patterns and parallel solution exploration. Finally, we investigate planning as a specialized form of reasoning that combines both structured and unstructured approaches for tackling complex, long-horizon tasks.

2.2.1 Structured Reasoning

Structured reasoning represents a methodical approach to problem-solving that employs explicit organizational frameworks to guide the reasoning process. Unlike unstructured approaches, structured reasoning makes the composition of reasoning steps explicit, which can be formalized as $R_s = R_1 \circ R_2 \circ \ldots \circ R_n$, where each R_i represents a discrete reasoning step with clear logical dependencies. In this formulation, each reasoning node is an explicitly executed computational unit, and the connections between nodes represent definite information flow paths. This approach enables more systematic exploration of solution spaces and facilitates more robust decision-making through deliberate step-by-step analysis, providing high interpretability and traceability throughout the reasoning process.

2.2.1.1 Dynamic Reasoning Structures

Dynamic reasoning structures allow for the adaptive construction of reasoning paths during problem-solving, creating versatile frameworks that can adjust based on intermediate results and insights.

Linear Sequential Reasoning Linear structures frame reasoning as a series of sequential steps, where each step builds on the one before. ReAct [70] illustrates this by combining reasoning traces with task-specific actions in an alternating fashion. This combination allows for reasoning traces to guide and modify action plans while actions can access external sources for further information. This mutual interaction improves both reasoning integrity and environmental adaptation.

Reasoning via Planning (RAP) [74] extends the linear reasoning paradigm by formulating LLM reasoning as a Markov decision process, though it was limited by states specifically designed for particular problems. The Markov Chain of Thought (MCoT) [71] extended this paradigm by conceptualizing each reasoning step as a Markovian state accompanied by executable code. This approach enables efficient next-step inference without requiring a lengthy context window by compressing previous reasoning into a simplified math question. Atom of Thoughts [132] explicitly defined problems as state representations and designed a general decomposition-contraction two-phase state transition mechanism to construct Markovian reasoning processes, transforming complex problems into a series of atomic questions.

Tree-Based Exploration Tree-based approaches expand beyond linear structures by organizing reasoning into hierarchical frameworks that support branching exploration. Tree of Thoughts (ToT) [72] introduces a structured approach where complex problems are decomposed into intermediate steps, enabling breadth-first or depth-first search through the solution space. This allows the model to consider multiple reasoning paths simultaneously and systematically explore alternatives.

Language Agent Tree Search (LATS) [73] advances this paradigm by integrating Monte Carlo Tree Search (MCTS) with LLMs, using the environment as an external feedback mechanism. This approach enables more deliberate and adaptive problem-solving by balancing exploration and exploitation through a sophisticated search process guided by LLM-powered value functions and self-reflection.

Reasoning via Planning (RAP) [74] further enhances tree-based reasoning by repurposing LLMs as both reasoning agents and world models. Through this dual role, RAP enables agents to simulate the outcomes of potential reasoning paths before committing to them, creating a principled planning framework that balances exploration with exploitation in the reasoning space.

Graph-Based Reasoning Graph structures offer even greater flexibility by allowing non-hierarchical relationships between reasoning steps. Graph of Thoughts (GoT) [75] extends tree-based approaches to arbitrary graph structures, enabling more complex reasoning patterns that can capture interdependencies between different steps. This approach allows for connections between seemingly disparate reasoning branches, facilitating more nuanced exploration of the solution space.

Path of Thoughts (PoT) [76] addresses relation reasoning challenges by decomposing problems into three key stages: graph extraction, path identification, and reasoning. By explicitly extracting a task-agnostic graph that identifies entities, relations, and attributes within the problem context, PoT creates a structured representation that facilitates the identification of relevant reasoning chains, significantly improving performance on tasks requiring long reasoning chains.

Diagram of Thought (DoT) [77] models iterative reasoning as the construction of a directed acyclic graph (DAG), organizing propositions, critiques, refinements, and verifications into a unified structure. This approach preserves logical consistency while enabling the exploration of complex reasoning pathways, providing a theoretically sound framework grounded in Topos Theory.

2.2.1.2 Static Reasoning Structures

Static reasoning structures employ fixed frameworks that guide the reasoning process without dynamically adjusting the structure itself, focusing instead on improving the content within the established framework.

Ensemble Methods. Ensemble approaches leverage multiple independent reasoning attempts to improve overall performance through aggregation. Self-Consistency [78] pioneered this approach by sampling multiple reasoning paths rather than relying on single greedy decoding, significantly improving performance through majority voting among the generated solutions.

MedPrompt [133] demonstrates how domain-specific ensemble techniques can enhance performance by carefully crafting prompts that elicit diverse reasoning approaches, achieving state-of-the-art results on medical benchmarks through systematic composition of prompting strategies.

LLM-Blender [134] introduces a sophisticated ensembling framework that leverages the diverse strengths of multiple LLMs through pairwise comparison (PairRanker) and fusion (GenFuser) of candidate outputs. This approach enables the system to select the optimal model output for each specific example, creating responses that exceed the capabilities of any individual model.

Progressive Improvement. Progressive improvement frameworks focus on iteratively refining reasoning through structured feedback loops. Self-Refine [67] implements an iterative approach where the model generates initial output, provides self-feedback, and uses that feedback to refine itself. This mimics human revision processes without requiring additional training or reinforcement learning, resulting in significant improvements across diverse tasks.

Reflexion [48] extends this concept by integrating environmental feedback, enabling agents to verbally reflect on task feedback signals and maintain reflective text in an episodic memory buffer. This approach guides future decision-making by incorporating insights from previous attempts, significantly enhancing performance in sequential decision-making, coding, and reasoning tasks.

Progressive-Hint Prompting (PHP) [79] further develops this paradigm by using previously generated answers as hints to progressively guide the model toward correct solutions. This approach enables automatic multiple interactions between users and LLMs, resulting in significant accuracy improvements while maintaining high efficiency.

Error Correction. Error correction frameworks focus specifically on identifying and addressing mistakes in the reasoning process. Self-Verification [80] introduces a self-critique system that enables models to backward-verify their conclusions by taking the derived answer as a condition for solving the original problem, producing interpretable validation scores that guide answer selection.

Refiner [135] addresses the challenge of scattered key information by adaptively extracting query-relevant content and restructuring it based on interconnectedness, highlighting information distinction and effectively aligning downstream LLMs with the original context.

Chain-of-Verification (CoVe) [81] tackles factual hallucinations through a structured process where the model drafts an initial response, plans verification questions, independently answers those questions, and generates a final verified response. This deliberate verification process significantly reduces hallucinations across a variety of tasks.

Recursive Criticism and Improvement (RCI) [128] enables LLMs to execute computer tasks by recursively criticizing and improving their outputs, outperforming existing methods on the MiniWoB++ benchmark with only a handful of demonstrations per task and without task-specific reward functions.

Critic [68] extends this approach by integrating external tools for validation, enabling LLMs to evaluate and progressively amend their outputs like human interaction with tools. This framework allows initially "black box" models to engage in a continuous cycle of evaluation and refinement, consistently enhancing performance across diverse tasks.

2.2.1.3 Domain-Specific Reasoning Frameworks

Domain-specific reasoning frameworks adapt structured reasoning approaches to the unique requirements of particular domains, leveraging specialized knowledge and techniques to enhance performance in specific contexts.

MathPrompter [82] addresses arithmetic reasoning challenges by generating multiple algebraic expressions or Python functions to solve the same math problem in different ways. This approach improves confidence in the output results by providing multiple verification paths, significantly outperforming state-of-the-art methods on arithmetic benchmarks.

Physics Reasoner [84] addresses the unique challenges of physics problems through a knowledge-augmented framework that constructs a comprehensive formula set and employs detailed checklists to guide effective knowledge application. This three-stage approach—problem analysis, formula retrieval, and guided reasoning—significantly improves performance on physics benchmarks by mitigating issues of insufficient knowledge and incorrect application.

Pedagogical Chain-of-Thought (PedCoT) [83] leverages educational theory, particularly the Bloom Cognitive Model, to guide the identification of reasoning mistakes in mathematical contexts. This approach combines pedagogical principles for prompt design with a two-stage interaction process, providing a foundation for reliable mathematical mistake identification and automatic answer grading.

The evolution of structured reasoning in LLM agents reflects a growing understanding of how to enhance reasoning capabilities through explicit organizational frameworks. From linear sequences to complex graphs, and ensemble methods to specialized domain frameworks, these approaches demonstrate the power of structural guidance in improving reasoning performance across diverse tasks and domains.

2.2.2 Unstructured Reasoning

In contrast to structured reasoning approaches that explicitly organize reasoning steps, unstructured reasoning (R_u) takes a holistic form $R_u = f(M_t)$, where the composition remains implicit and flexible. In this mode, the reasoning process is encapsulated within a single function mapping, without explicitly defining intermediate steps or state transitions. This approach leverages the inherent capabilities of language models to generate coherent reasoning without enforcing rigid structural constraints, with intermediate reasoning processes occurring explicitly in the language space or implicitly in the latent space. Unstructured reasoning methods have demonstrated remarkable effectiveness across diverse tasks while maintaining simplicity and efficiency in implementation.

2.2.2.1 Prompting-Based Reasoning

The most accessible way to elicit reasoning in LLM agents lies in carefully crafted prompts. By providing appropriate reasoning demonstrations or instructing LLMs to perform inferential steps, agents can leverage their logical deduction capabilities to solve problems through flexible reasoning processes.

Chain-of-Thought Variants. The cornerstone of prompting-based reasoning is Chain-of-Thought (CoT) prompting [46], which operationalizes reasoning through few-shot examples with explicit generation of intermediate rationalization steps. This foundational technique has inspired several evolutionary variants that enhance its basic approach. Zero-shot CoT [136] eliminates the need for demonstration examples through strategic prompting (e.g., "Let's think step by step"), making the approach more accessible while maintaining effectiveness. Auto-CoT [137] automates the creation of effective demonstrations by clustering diverse questions and generating reasoning chains for representative examples from each cluster. Least-to-Most Prompting [138] addresses complex reasoning by decomposing problems into sequential sub-problems, enabling a progressive planning process that facilitates easy-to-hard generalization. Complex CoT [139] further enhances reasoning depth by specifically selecting high-complexity exemplars as prompting templates, better-equipping models to tackle intricate problems.

Problem Reformulation Strategies. Advanced prompting strategies demonstrate architectural innovations in reasoning guidance by reformulating the original problem. Step-Back Prompting [85] implements abstraction-first reasoning through conceptual elevation, enabling models to derive high-level concepts and first principles before addressing specific details. Experimental results demonstrate substantial performance gains on various reasoning-intensive tasks, with improvements of 7-27% across physics, chemistry, and multi-hop reasoning benchmarks. Rephrase and Respond [140] employ semantic expansion to transform original questions into more tractable forms, allowing models to approach problems from multiple linguistic angles and identify the most effective problem formulation.

Abstraction-of-Thought [141] introduces a novel structured reasoning format that explicitly requires varying levels of abstraction within the reasoning process. This approach elicits language models to first contemplate at the abstract level before incorporating concrete details, a consideration overlooked by step-by-step CoT methods. By aligning models with the AoT format through finetuning on high-quality samples, the approach demonstrates substantial performance improvements across a wide range of reasoning tasks compared to CoT-aligned models.

Enhanced Prompting Frameworks. Several frameworks extend the basic prompting paradigm to create more sophisticated reasoning environments. Ask Me Anything [86] constrains open-ended generation by reformulating tasks into structured question-answer sequences, enforcing focused reasoning trajectories. This approach recursively uses the LLM itself to transform task inputs to the effective QA format, enabling open-source GPT-J-6B to match or exceed the performance of few-shot GPT3-175B on 15 of 20 popular benchmarks.

Algorithm of Thoughts [142] proposes a novel strategy that propels LLMs through algorithmic reasoning pathways by employing algorithmic examples fully in context. This approach exploits the innate recurrence dynamics of LLMs, expanding their idea exploration with merely one or a few queries. The technique outperforms earlier single-query methods and even more recent multi-query strategies while using significantly fewer tokens, suggesting that instructing an LLM using an algorithm can lead to performance surpassing that of the algorithm itself.

Chain-of-Knowledge (CoK) [87] augments LLMs by dynamically incorporating grounded information from heterogeneous sources, resulting in more factual rationales and reduced hallucination. CoK consists of three stages: reasoning preparation, dynamic knowledge adapting, and answer consolidation, leveraging both unstructured and structured knowledge sources through an adaptive query generator. This approach corrects rationales progressively using preceding corrected rationales, minimizing error propagation between reasoning steps.

Self-Explained Keywords (SEK) [88] addresses the challenge of low-frequency terms in code generation by extracting and explaining key terms in problem descriptions with the LLM itself and ranking them based on frequency. This approach significantly improves code generation performance across multiple benchmarks, enabling models to shift attention from low-frequency keywords to their corresponding high-frequency counterparts.

2.2.2.2 Reasoning Models

Recent advances in language models have led to the development of specialized reasoning models designed explicitly for complex inferential tasks. These models are fine-tuned or specially trained to optimize reasoning capabilities, incorporating architectural and training innovations that enhance their performance on tasks requiring multi-step logical inference.

Reasoning models like DeepSeek's R1 [89], Anthropic's Claude 3.7 Sonnet [9], and OpenAI's o series models [90] represent the frontier of reasoning capabilities, demonstrating remarkable proficiency across diverse reasoning bench-

marks. These models are trained with specialized methodologies that emphasize reasoning patterns, often incorporating significant amounts of human feedback and reinforcement learning to enhance their inferential abilities.

The emergence of dedicated reasoning models reflects a growing understanding of the importance of reasoning capabilities in language models and the potential benefits of specialized training for these tasks. By concentrating on reasoning-focused training data and objectives, these models achieve performance levels that significantly exceed those of general-purpose language models, particularly on tasks that require complex logical inference, mathematical reasoning, and multi-step problem-solving.

2.2.2.3 Implicit Reasoning

Beyond explicit reasoning approaches, recent research has explored the potential of implicit reasoning methods that operate without overtly exposing the reasoning process. These approaches aim to improve efficiency by reducing the number of tokens generated while maintaining or enhancing reasoning performance.

Quiet-STaR [91] generalizes the Self-Taught Reasoner approach by teaching LMs to generate rationales at each token to explain the future text, improving their predictions. This approach addresses key challenges including computational cost, the initial unfamiliarity with generating internal thoughts, and the need to predict beyond individual tokens. Experimental results demonstrate zero-shot improvements in mathematical reasoning (5.9%→10.9%) and commonsense reasoning (36.3%→47.2%) after continued pretraining, marking a step toward LMs that learn to reason in a more general and scalable way.

Chain of Continuous Thought (Coconut) [92] introduces a paradigm that enables LLM reasoning in an unrestricted latent space instead of using natural language. By utilizing the last hidden state of the LLM as a representation of the reasoning state and feeding it back as the subsequent input embedding directly in the continuous space, Coconut demonstrates improved performance on reasoning tasks with fewer thinking tokens during inference. This approach leads to emergent advanced reasoning patterns, including the ability to encode multiple alternative next reasoning steps, allowing the model to perform a breadth-first search rather than committing to a single deterministic path.

Recent analysis [143] of implicit reasoning in transformers reveals important insights into its limitations. While language models can perform step-by-step reasoning and achieve high accuracy in both in-domain and out-of-domain tests via implicit reasoning when trained on fixed-pattern data, implicit reasoning abilities emerging from training on unfixed-pattern data tend to overfit specific patterns and fail to generalize further. These findings suggest that language models acquire implicit reasoning through shortcut learning, enabling strong performance on tasks with similar patterns while lacking broader generalization capabilities.

The evolution of unstructured reasoning approaches demonstrates the remarkable adaptability of language models to different reasoning paradigms. From simple prompting techniques to sophisticated implicit reasoning methods, these approaches leverage the inherent capabilities of LLMs to perform complex logical inferences without requiring explicit structural constraints. This flexibility enables more intuitive problem-solving while maintaining efficiency and effectiveness across diverse reasoning tasks.

2.2.3 Planning

Planning is a fundamental aspect of human cognition, enabling individuals to organize actions, anticipate outcomes, and achieve goals in complex, dynamic environments [144]. Formally, planning can be described as the process of constructing potential pathways from an initial state to a desired goal state, represented as $P : S_0 \rightarrow \{a_1, a_2, \ldots, a_n\} \rightarrow S_g$, where S_0 is the starting state, $\{a_1, a_2, \ldots, a_n\}$ denotes a sequence of possible actions, and S_g is the goal state. Unlike direct reasoning, planning involves generating hypothetical action sequences before execution, functioning as computational nodes that remain inactive until deployed. This cognitive ability emerges from the interplay of specialized neural circuits, including the prefrontal cortex, which governs executive control, and the hippocampus, which supports episodic foresight and spatial mapping. Insights from decision theory, psychology, and cybernetics—such as rational frameworks, prospect theory, and feedback loops—demonstrate how planning allows humans to transcend reactive behavior, actively shaping their futures through deliberate intent and adaptive strategies. This capacity not only underpins intelligent behavior but also serves as a model for developing LLM-based agents that seek to replicate and enhance these abilities computationally [145, 146].

In human cognition, planning operates as a hierarchical process, integrating immediate decisions with long-term objectives. This reflects the brain's modular architecture, where neural systems collaborate to balance short-term demands with future possibilities—a dynamic informed by control theory's principles of stability and optimization. Similarly, LLM-based agents employ planning by leveraging their extensive linguistic knowledge and contextual reasoning to transform inputs into actionable steps. Whether addressing structured tasks or unpredictable challenges,

these agents emulate human planning by decomposing objectives, evaluating potential outcomes, and refining their strategies—blending biological inspiration with artificial intelligence. This section examines the theoretical foundations and practical techniques of planning, from sequential approaches to parallel exploration, highlighting its critical role in intelligent systems.

Despite the potential of LLMs in automated planning, their performance faces limitations due to gaps in world knowledge [147]. LLMs often lack deep comprehension of world dynamics, relying on pattern recognition rather than genuine causal reasoning, which hinders their ability to manage sub-goal interactions and environmental changes [148]. Additionally, their reliance on static pre-training data restricts adaptability in real-time scenarios, limiting their generalization in dynamic planning tasks [149]. The absence of an intrinsic System 2 reasoning mechanism further complicates their ability to independently generate structured, optimal plans [150]. However, researchers have proposed strategies such as task decomposition, search optimization, and external knowledge integration to mitigate these challenges.

Task Decomposition Task decomposition enhances LLM planning by breaking complex goals into smaller, manageable subtasks, reducing problem complexity and improving systematic reasoning. The Least-to-Most Prompting method [138] exemplifies this approach, guiding LLMs to solve subproblems incrementally. ADaPT [151] further refines this strategy by dynamically adjusting task decomposition based on complexity and model capability, particularly in interactive decision-making scenarios. These methods also facilitate parallel subtask processing, backward error tracing, and independence determination [132], providing a structured framework for reasoning.

In LLM planning, tasks function as executable units—distinct from static state descriptions in formal models—emphasizing structured sequences that achieve intended outcomes [66]. These tasks vary in nature: some are subproblems requiring specific solutions (e.g., solving equations within broader challenges), while others involve tool invocation (e.g., querying APIs for weather data in travel planning) [152, 153]. Alternatively, tasks may be represented as graph nodes mapping dependencies, such as prioritizing objectives in project management [154]. By defining clear, modular goals, these formulations enhance reasoning and action efficiency, guiding agents through complex problem spaces with greater precision [93].

Searching Given the stochastic nature of LLMs [155], parallel sampling combined with aggregated reasoning can improve inference performance. Task decomposition structures individual solution trajectories, enabling the construction of a solution space that includes multiple pathways to a goal and their interrelationships [72, 156]. This space allows sampling diverse potential solutions [157], facilitating exploration through techniques like reflection, review, and parallel sampling informed by existing knowledge [158].

Computational constraints often preclude exhaustive evaluation, making efficient navigation of the solution space essential. Methods include tree search algorithms like LATS [159], heuristic approaches such as PlanCritic's genetic algorithms [160], and CoT-SC, which identifies recurring solutions via self-consistency checks [78]. Reward-based models like ARMAP assess intermediate and final outcomes to optimize planning [106]. This iterative exploration and refinement process enhances adaptability, ensuring robust strategies for complex problems.

World Knowledge Effective planning requires agents to navigate dynamic environments, anticipate changes, and predict outcomes, underscoring the importance of world knowledge. RAP [74] examines the interplay between LLMs, agent systems, and world models, positioning LLMs as dual-purpose entities: as world models, they predict state changes following actions [107, 161]; as agents, they select actions based on states and goals [70]. This framework mirrors human cognition—simulating action consequences before selecting optimal paths—and unifies language models, agent models, and world models as pillars of machine reasoning [162].

Agents augment LLM capabilities by integrating external knowledge, addressing gaps in world understanding. ReAct employs an action-observation loop to gather environmental feedback, combining real-time data with linguistic knowledge to improve decision-making in complex scenarios [70]. This enables LLMs to iteratively refine their world models during action execution, supporting adaptive planning. Conversely, LLM+P [163] integrates LLMs with the PDDL planning language, converting natural language inputs into formalized representations solved by classical planners [164, 165]. This hybrid approach compensates for LLMs' limitations in structured planning, merging their linguistic flexibility with the reliability of traditional systems.

Further advancements enhance LLM planning through world knowledge integration. CodePlan [166] uses code-form plans—pseudocode outlining logical steps—to guide LLMs through complex tasks, achieving notable performance improvements across benchmarks [167]. The World Knowledge Model (WKM) equips LLMs with prior task knowledge and dynamic state awareness, reducing trial-and-error and hallucinations in simulated environments [168]. A neuro-symbolic approach combining Linear Temporal Logic with Natural Language (LTL-NL) integrates formal logic with

LLMs, leveraging implicit world knowledge to ensure reliable, adaptive planning [169]. Together, these methods illustrate how structured frameworks and environmental understanding can transform LLMs into effective planners.

Chapter 3

Memory

Memory is fundamental to both human and artificial intelligence. For humans, it serves as the bedrock of cognition, a vast repository of experiences and knowledge that empowers us to learn, adapt, and navigate the complexities of the world. From infancy, our capacity to encode, store, and retrieve information underpins our ability to acquire language, master skills, and build relationships. Decades of research in neuroscience and cognitive psychology have illuminated the multifaceted role of memory, revealing its influence on our sense of self, creative endeavors, and decision-making processes. Similarly, in the burgeoning field of artificial intelligence, memory is increasingly recognized as a cornerstone of intelligent behavior. Just as humans rely on past experiences to inform present actions, AI agents require robust memory mechanisms to tackle intricate tasks, anticipate future events, and adjust to dynamic environments. Therefore, a deep understanding of human memory – its organization, processes, and limitations – provides invaluable insights for the development of more capable and adaptable AI systems. This section will first provide a concise overview of human memory, focusing on the key stages of encoding, consolidation, and retrieval. We will then transition to exploring the diverse approaches employed in designing AI agent memory systems, ranging from traditional symbolic representations to cutting-edge neural network-based methods. A critical comparison between these artificial memory systems and their human counterparts will highlight existing gaps in areas such as adaptability, contextual understanding, and resilience. Finally, we will consider how principles derived from neuroscience and cognitive psychology can inform future research, suggesting directions that may lead to the creation of artificial memory systems that exhibit greater robustness, nuance, and ultimately, a closer resemblance to the remarkable capabilities of human memory.

3.1 Overview of Human Memory

3.1.1 Types of Human Memory

Human memory is often conceptualized as a multi-tiered system that captures, stores, and retrieves information at different levels of processing and timescales. Researchers from the fields of cognitive science, neuroscience, and psychology have proposed various models to describe these levels. A commonly accepted hierarchy distinguishes between sensory memory, short-term memory (including working memory), and long-term memory [170, 171]. Within long-term memory, explicit (declarative) and implicit (non-declarative) forms are further delineated [172]. Figure 3.1 illustrates one such hierarchical framework:

- **Sensory Memory.** Sensory memory is the initial, brief store of raw sensory information. It maintains inputs from the environment for a duration ranging from milliseconds to a few seconds, allowing subsequent processes to determine which portions of the stimulus are relevant for further analysis [173]. Iconic memory (for visual input) [174] and echoic memory (for auditory input) [175] are two well-known subtypes.
- **Short-Term Memory and Working Memory.** Short-term memory (STM) involves holding a limited amount of information in an easily accessible state for seconds to under a minute. The term *working memory* is often used to emphasize the manipulation of that information rather than mere maintenance. While some models treat working memory as a subset of STM, others view it as a distinct system that manages both the storage and active processing of data (for instance, performing arithmetic in one's head) [176, 177]. The capacity of STM or working memory is finite, typically cited as around seven plus or minus two chunks of information [98], though individual differences and task factors can modulate this figure.

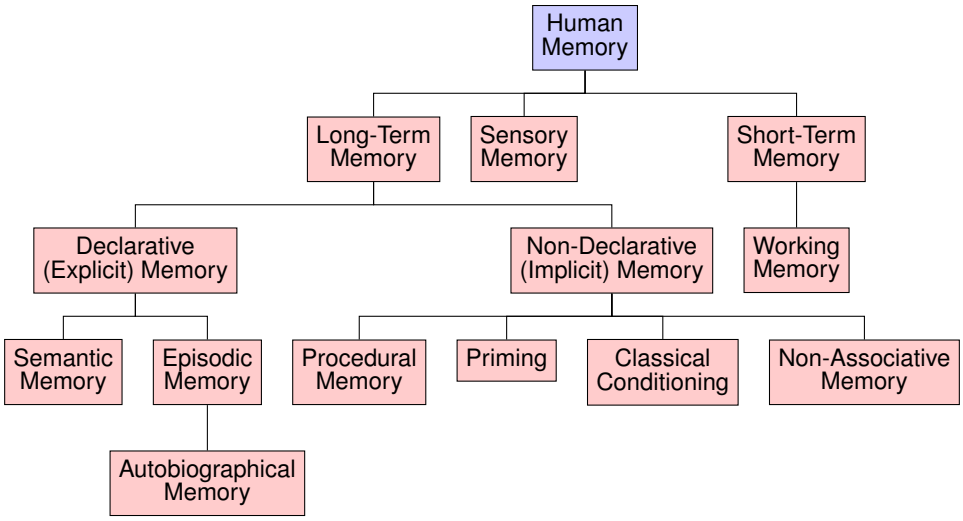


Figure 3.1: The hierarchical taxonomy of human memory system.

- **Long-Term Memory (LTM).** Long-term memory accommodates the more durable storage of information that can persist from hours to decades [178, 179]. This repository supports the learning of skills, the acquisition of factual knowledge, and the recollection of personal experiences. Although long-term memory is sometimes described as having a vast or near-unlimited capacity, factors such as decay, interference, and retrieval cues influence the extent to which stored information can be accessed [180].
  - **Declarative (Explicit) Memory.** Declarative memory encompasses memories that can be consciously recalled and articulated [181]. Within this broad category, researchers often discuss:
    - **Semantic Memory:** Factual knowledge about the world, including concepts, words, and their relationships [182]. Examples include recalling the meaning of vocabulary terms or knowing the capital city of a country.
    - **Episodic Memory:** Personally experienced events that retain contextual details such as time, place, and the people involved [183]. This form of memory allows individuals to mentally travel back in time to relive past experiences.
    - **Autobiographical Memory:** A form of episodic memory focusing on events and experiences related to one's personal history [184]. While sometimes treated as a sub-category of episodic memory, autobiographical memory places particular emphasis on the self and its evolving life narrative.
  - **Non-Declarative (Implicit) Memory.** Non-declarative memory refers to memories that influence behavior without the need for conscious awareness [185]. Key subtypes include:
    - **Procedural Memory:** The gradual acquisition of motor skills and habits (e.g., riding a bicycle, typing on a keyboard) that become automatic with repetition [186, 187].
    - **Priming:** The phenomenon in which prior exposure to a stimulus influences subsequent responses, often without explicit recognition of the previous encounter [188].
    - **Classical Conditioning:** The learned association between two stimuli, where one stimulus comes to elicit a response originally produced by the other [189].
    - **Non-Associative Memory:** Adaptive modifications in behavior following repeated exposure to a single stimulus. Habituation (reduced response to a repeated, harmless stimulus) and sensitization (increased response after exposure to a noxious or intense stimulus) are representative examples [190, 191].

Despite the orderly appearance of these categories, human memory processes often overlap. For example, autobiographical memory is typically nested within episodic memory, yet its particular focus on self-relevant experiences leads some theorists to treat it as a slightly different category. Similarly, the boundary between short-term and working memory can differ depending on the theoretical perspective. Some scholars prefer a more functional, process-oriented view of working memory, while others employ a strictly capacity-oriented concept of short-term storage. In each case, these different perspectives on memory highlight the complexity and nuance of human cognition.

3.1.2 Models of Human Memory

Human memory has inspired a wide range of theoretical models, each offering different insights into how information is acquired, organized, and retrieved. Although no single framework commands universal agreement, several influential perspectives have shaped the discourse in cognitive science, neuropsychology, and AI research. The following content highlights some of the most prominent models and architectures used to explain memory's multiple facets.

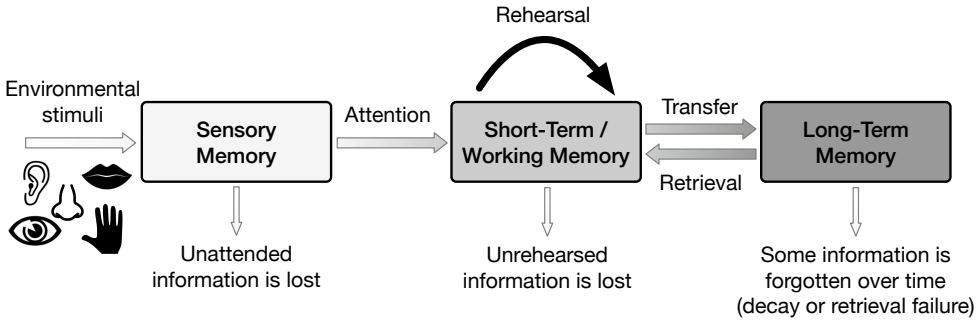


Figure 3.2: Atkinson-Shiffrin three-stage model of human memory [170].

The Multi-Store (Modal) Model. A seminal proposal by Atkinson and Shiffrin [170] introduced the multi-store or "modal" model, which posits three main stores for incoming information: *sensory memory*, *short-term memory*, and *long-term memory*. Control processes (e.g., attention, rehearsal) regulate how data transitions across these stores. Figure 3.2 illustrates this model of memory. Despite its relative simplicity, this model remains foundational for understanding how fleeting sensory impressions eventually form stable, long-lasting representations.

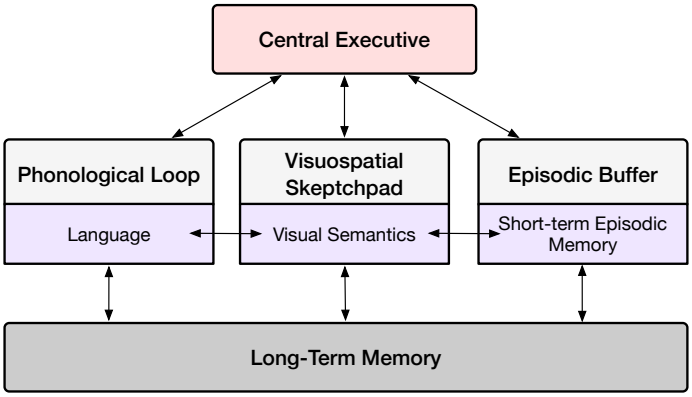


Figure 3.3: Baddeley's model of working memory [192].

Working Memory Models. Recognizing that short-term memory also involves active maintenance, Baddeley and Hitch [192] proposed a *working memory* framework emphasizing the dynamic manipulation of information. Their original model described a central executive that coordinates two subsystems: the phonological loop (verbal) and the visuospatial sketchpad (visual/spatial). A subsequent refinement introduced the episodic buffer to integrate material from these subsystems with long-term memory [193]. Figure 3.3 shows the framework of the working memory model. Alternatives such as Cowan's embedded-processes model [194] similarly underscore the role of attention in governing how information is briefly sustained and manipulated.

Serial-Parallel-Independent (SPI) Model. Initial distinctions between episodic, semantic, and procedural memory were championed by Tulving [195], who later refined his ideas into the Serial-Parallel-Independent (SPI) model, as shown in Figure 3.4. In this framework, memory is divided into two overarching systems. The *cognitive representation system* handles perceptual input and semantic processes, encompassing facts, concepts, and contextual (episodic) knowledge. The *action system*, by contrast, underpins procedural skills such as dance routines, driving maneuvers, or typing proficiency. Tulving's SPI model posits that memory formation can occur at multiple levels: strictly perceptual encoding can support rudimentary episodic memories, while richer episodic representations benefit from semantic

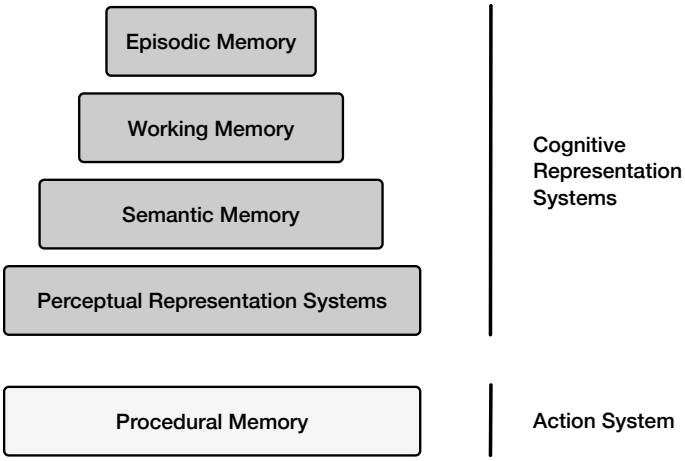


Figure 3.4: The Serial-Parallel Independent (SPI) model of human memory [195].

mediation. For instance, patients with semantic dementia, who struggle to retain word meanings, can still form some episodic memories but often lack the full contextual detail conferred by intact semantic networks. By highlighting the role of procedural memory and its automatic, intuitive nature, the SPI model aims to integrate structure (the content of memory) and function (how memory is used), surpassing earlier accounts that largely focused on explicit storage and retrieval. Despite these strengths, critics note that the model under-specifies how working memory operates within the broader system, and the feedback mechanisms connecting cognitive and action subsystems remain loosely defined.

Global Workspace Theory (GWT) and the IDA/LIDA Framework. Global Workspace Theory (GWT), developed by Baars [196], conceptualizes consciousness and working memory as a "broadcast" mechanism that distributes information to specialized processors. Building on GWT, Franklin [197, 198] proposed the *IDA (Intelligent Distribution Agent)* model, later extended to *LIDA (Learning IDA)*, as a comprehensive cognitive architecture. In these frameworks, multiple memory systems (e.g., perceptual, episodic, procedural) interact through iterative "cognitive cycles", with a global workspace functioning as a hub for attention and decision-making. From an AI standpoint, IDA/LIDA demonstrates how human-like memory processes can be operationalized to guide an agent's perception, action selection, and learning.

ACT-R and Cognitive Architectures. ACT-R (Adaptive Control of Thought—Rational) [199] is a comprehensive cognitive architecture that integrates memory, perception, and motor processes into a unified theoretical framework. It has been applied extensively across diverse domains, including learning and memory, problem-solving, decision-making, language comprehension, perception and attention, cognitive development, and individual differences. Figure 3.5 illustrates the processes of ACT-R. At the core of ACT-R are distinct *modules* (e.g., visual, manual, declarative, procedural) that interact with the system through dedicated *buffers*. Declarative memory stores factual "chunks," while procedural memory encodes if–then production rules for actions and strategies. Cognition unfolds via a *pattern matcher* that selects a single production to fire based on the current buffer contents. This symbolic production system is augmented by subsymbolic processes, guided by mathematical equations that dynamically regulate activations, retrieval latencies, and production utilities. By combining symbolic and subsymbolic levels, ACT-R provides a mechanistic account of how individuals acquire, retrieve, and apply knowledge—thus shedding light on empirical phenomena such as reaction times, error patterns, and the shaping of learning over time.

Each of the aforementioned models illuminates different aspects of memory. The multi-store model provides a straightforward introduction to storage stages, working memory models emphasize active maintenance and manipulation, and frameworks such as IDA/LIDA or ACT-R embed memory within a comprehensive view of cognition. In practice, researchers often draw upon multiple perspectives, reflecting the intricate nature of human memory and its integral role in perception, learning, and adaptive behavior.

3.2 From Human Memory to Agent Memory

Having established the fundamentals of human memory, we now focus on how Large Language Model (LLM)-based agents manage and store information. Memory is not merely a storage mechanism but is fundamental to human and

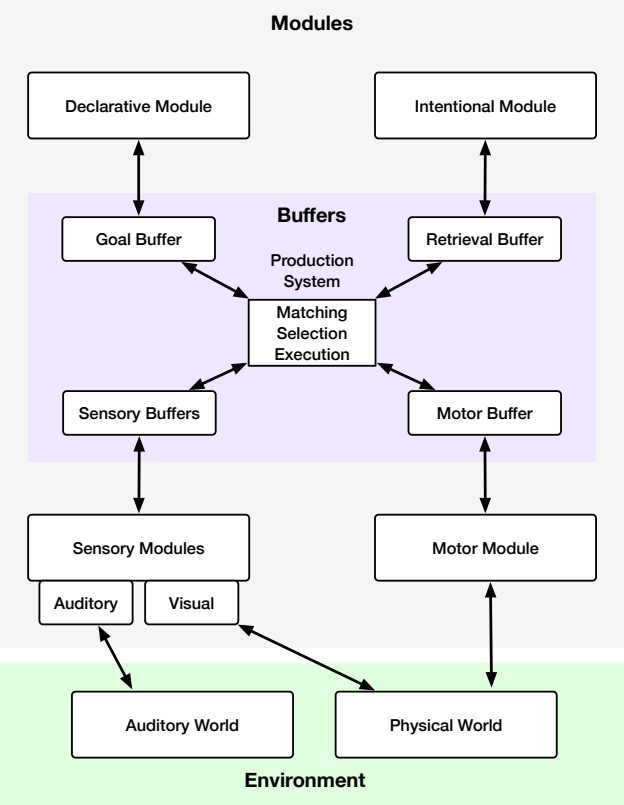


Figure 3.5: An abstraction of the most important processes in the ACT-R model [199].

artificial intelligence. Memory underpins cognition, enabling learning, adaptation, and complex problem-solving for humans. Similarly, for LLM-based agents, memory provides the crucial scaffolding for maintaining context, learning from experience, and acting coherently over time. Without memory, even a highly capable LLM would struggle to adapt to changing circumstances or maintain focus during extended interactions.

While LLM-based agents and biological systems differ fundamentally, the principles guiding human memory—context retention, selective forgetting, and structured retrieval—are highly relevant to agent design. Therefore, examining the parallels and distinctions between human and artificial memory is beneficial. Functionally, we can draw analogies: an agent's short-term memory buffer resembles the prefrontal cortex's role in working memory, while long-term storage in a vector database is akin to the hippocampus's function in consolidating episodic memories. Agent memory design can benefit from emulating human memory's mechanisms, including selective attention, prioritized encoding, and cue-dependent retrieval. However, crucial differences exist.

Human memory, built upon biological neural networks, integrates storage and computation within neurons' connections and activity patterns. This offers a high degree of parallelism and adaptability. In contrast, current agent memory systems predominantly rely on digital storage and algorithms, using symbolic representations and logical operations, thus separating storage and computation. This impacts information processing: human memory is associative and dynamic, capable of fuzzy matching and creative leaps, while current agent memory relies on precise matching and vector similarity, struggling with ambiguity. Although digital storage capacity is vast, it cannot yet replicate the complexity and dynamism of human memory, particularly in nuanced pattern recognition and long-term stability. Human memory, while imperfect, excels at extracting crucial information from noisy data. Agent memory systems, in their current stage, are still nascent compared to the intricacies of human memory, facing limitations in organization, integration, adaptive forgetting, and knowledge transfer.

The need for a dedicated memory module in LLM-based agents is paramount. While external knowledge bases (databases, search engines, APIs) [200] provide valuable information, they do not capture the agent's internal reasoning, partial inferences, or task-specific context. An agentic memory system internalizes interim steps, evolving objectives, and historical dialogue, enabling self-referential exploration and adaptation. This is crucial for tasks requiring the agent to build upon prior judgments or maintain a personalized understanding of user goals.

Early approaches to agent memory, such as appending conversation history to the input prompt (a rudimentary form of working memory) [201], have evolved. Modern architectures employ more sophisticated techniques, including vector embeddings for rapidly retrieving memories [202] and selective incorporation of reasoning chains into subsequent inference steps [203, 204]. These diverse methods share the common goal of managing a large information reservoir without compromising system responsiveness.

However, compared to the sophistication of human memory, current agentic methods have limitations. Many systems lack coherent strategies for long-term memory consolidation, leading to cluttered logs or abrupt information loss. The flexible, bidirectional interplay between stored knowledge and ongoing processing, characteristic of human working memory, is often absent. Metacognitive oversight—selective recall, forgetting, and vigilance against outdated information—is also underdeveloped in LLM-based agents. Balancing comprehensive recall with practical efficiency, as humans do, remains a key challenge.

Building robust and adaptable memory for LLM-based agents involves addressing three core research questions: First, how should memory be represented to capture diverse information types and facilitate efficient access? Second, how can agent memory evolve, incorporating new experiences, adapting to changing contexts, and maintaining consistency? Finally, how can the stored memories effectively enhance reasoning, decision-making, and overall agent performance? The following sections delve into these crucial areas, exploring current approaches, limitations, and potential future directions.

3.3 Representation of Agent Memory

Inspired by human cognitive systems [285], current memory architecture in intelligent agents adopts a hierarchical framework that integrates perception through sensory memory [205], real-time decision-making via short-term memory [286, 287], and sustained knowledge retention through long-term memory [288, 289, 48]. This multi-layered structure equips agents to manage immediate tasks while maintaining a broader contextual understanding, fostering adaptability and seamless continuity across diverse interactions.

Specifically, the memory system transforms raw environmental inputs into structured, actionable representations. Sensory memory acts as the gateway, capturing and selectively filtering perceptual signals to provide a foundation for cognitive processing. Short-term memory bridges these immediate perceptions with task-level understanding, buffering recent interactions and enabling dynamic adaptation through experience replay and state management. Long-term memory then consolidates and stores information over extended periods, facilitating cross-task generalization and the accumulation of enduring knowledge.

Together, these memory components form a cohesive cycle of perception, interpretation, and response. This cycle supports real-time decision-making and enables agents to learn and evolve continuously, reflecting an intricate balance between responsiveness and growth. The following delves into the formulation of each memory type, exploring their unique roles and interactions within the agent's cognitive architecture.

3.3.1 Sensory Memory

In human cognitive systems, sensory memory serves as a mechanism for collecting information through the senses—touch, hearing, vision, and others—and is characterized by its extremely brief lifespan. Analogously, sensory memory functions as the embedded representation of inputs such as text, images, and other perceptual data in intelligent agents. It represents the initial phase of environmental information processing, acting as a gateway for transforming raw observations into meaningful representations for further cognitive processing.

Sensory memory in intelligent agents transcends passive information reception. It dynamically encodes and filters perceptual signals, bridging immediate sensory inputs with the agent's internal state, objectives, and prior knowledge. This adaptive process facilitates rapid perception of environmental changes, task continuity, and real-time context-aware information processing. Sophisticated attention mechanisms are employed to ensure relevance and focus in the sensory memory layer, forming a critical foundation for decision-making and adaptation.

Formally, sensory memory formation consists of three sequential steps: *perceptual encoding*, *attentional selection*, and *transient retention*. First, perceptual encoding transforms raw sensory signals into processable representations, mathematically expressed as:

$$\phi(o_t) = \text{Encode}(o_t, s_t) \tag{3.1}$$

where o_t is the sensory input at time t, and s_t represents the agent's state. For instance, RecAgent [205] employs an LLM-based sensory memory module to encode raw observations while filtering noise and irrelevant content. Extending

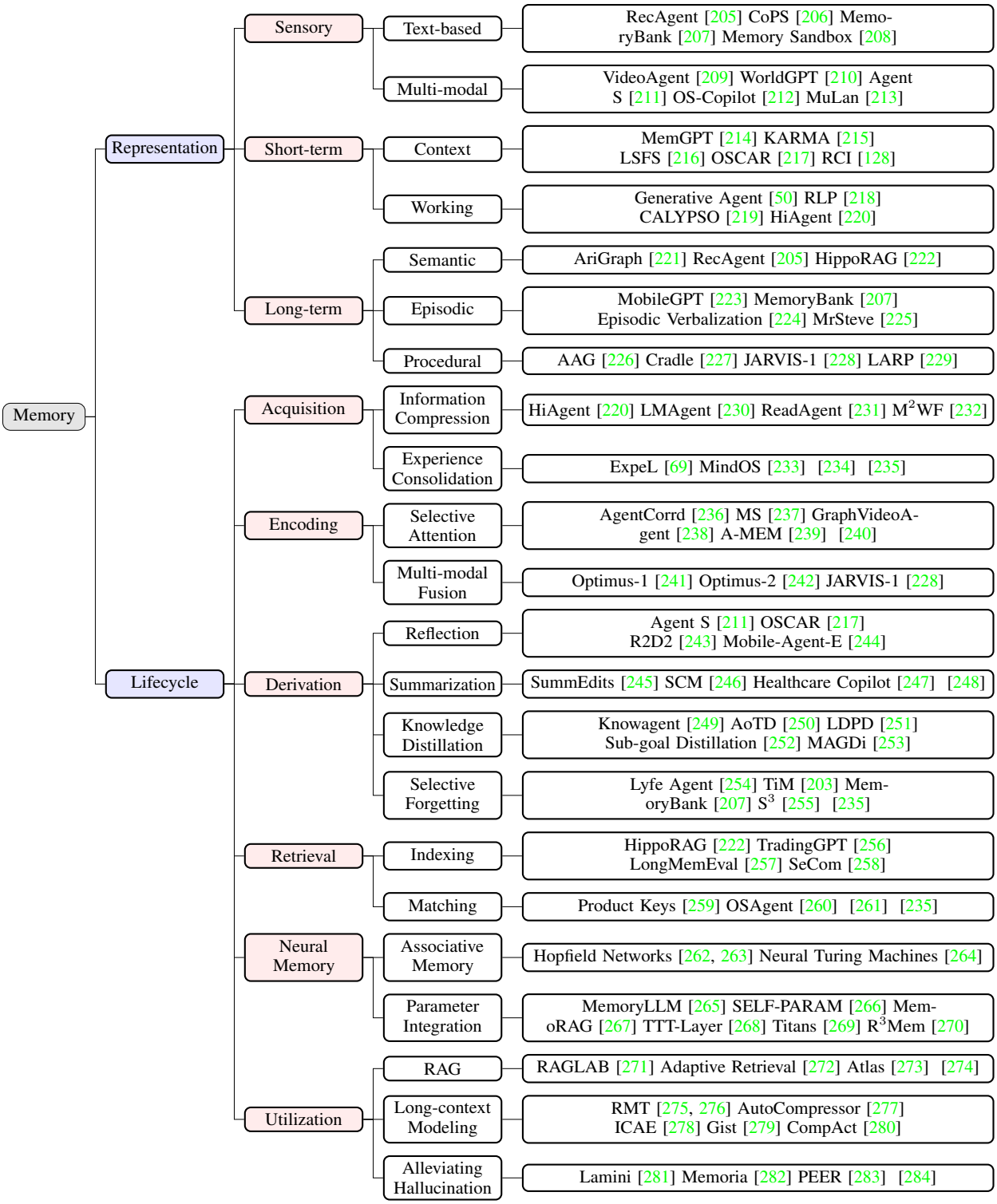


Figure 3.6: Tree diagram of the memory module in intelligent agents.

beyond text-based perception, multimodal sensory memory systems such as Jarvis-1 [228], VideoAgent [209], and WorldGPT [210] integrate multimodal foundation models to process diverse modality inputs.

Next, attentional selection extracts crucial information from the encoded sensory data. This process, guided by an attention mechanism, is represented as:

$$\alpha_t = \text{Attention}(\phi(o_t), c_t) \tag{3.2}$$

where $\phi(o_t)$ is the encoded input, and c_t denotes contextual information influencing attention. For example, RecAgent [205] employs an attention mechanism with an importance scoring system that assigns relevance scores to compressed observations, prioritizing critical inputs such as item-specific interactions while de-emphasizing less significant actions. This helps extract high-priority information for memory retention.

Finally, transient retention temporarily stores the selected sensory information as sensory memory:

$$M_{\text{sensory}} = \{\alpha_t \mid t \in [t - \tau, t]\} \tag{3.3}$$

Several strategies have been implemented to manage the time window. For instance, RecAgent [205] models retention by associating each observation with the timestamp corresponding to the start of a simulation round in the user behavior simulation environment, represented as a triplet $\langle$observation, importance score, timestamp$\rangle$. Similarly, CoPS [206] employs a fixed-size sensory memory pool as a time window, which consists of user search requests for personalized search, facilitating "re-finding" behavior. When a new query is received, the system first checks the sensory memory for relevant matches. If a match is found, the query is classified as a re-finding instance, enabling a rapid sensory response.

3.3.2 Short-Term Memory

Short-term memory in cognition-inspired intelligent agents serves as a transient and dynamic workspace that bridges sensory memory and long-term memory. It is essential for storing and processing task-relevant information and recent interaction sequences, supporting real-time decision-making and adaptive behavior. Inspired by human short-term and working memory, it temporarily retains information to facilitate complex cognitive tasks, ensuring continuity and coherence in the agent's operations.

Short-term memory in intelligent agents can be categorized into *context memory* and *working memory*. On the one hand, context memory treats the context window as the short-term memory of LLMs. For example, MemGPT [214], inspired by hierarchical memory systems in operating systems, manages different storage tiers to extend context beyond the LLM's inherent limitations. [290] introduces a neurosymbolic context memory that enhances LLMs by enabling symbolic rule grounding and LLM-based rule application.

On the other hand, working memory involves fetching and integrating relevant external knowledge to hold essential information during an agent's operation. Generative Agent [50] employs short-term memory to retain situational context, facilitating context-sensitive decision-making. Reflexion [48] utilizes a sliding window mechanism to capture and summarize recent feedback, balancing detailed immediate experiences with high-level abstractions for enhanced adaptability. RLP [218] maintains conversational states for speakers and listeners, using them as short-term memory prompts to support dialogue understanding and generation.

For interactive and creative game scenarios, CALYPSO [219] assists Dungeon Masters in storytelling for Dungeons & Dragons by constructing short-term memory from scene descriptions, monster details, and narrative summaries, enabling adaptive storytelling and dynamic engagement. Similarly, Agent S [211] and Synapse [291], designed for GUI-based autonomous computer interaction, define their short-term memory as task trajectories, including actions such as button clicks and text inputs. This formulation supports behavioral cloning and enhances adaptation in novel GUI navigation tasks.

In robotics applications, SayPlan [292] leverages scene graphs and environmental feedback as short-term memory to guide planning and execution in scalable robotic environments. KARMA [215] engages short-term working memory with an effective and adaptive memory replacement mechanism to dynamically record changes in objects' positions and states. LLM-Planner [293] iteratively updates short-term memory with environmental observation to prompt an LLM for dynamic planning.

3.3.3 Long-Term Memory

Long-term memory in cognition-inspired intelligent agents enables the retention and retrieval of information over extended periods, allowing agents to generalize knowledge and adapt to new contexts effectively. Unlike sensory and short-term memory, which handle transient or immediate data, long-term memory supports cumulative learning and cross-task adaptability. It mirrors human long-term memory by incorporating explicit and implicit components, facilitating richer contextual understanding and intuitive behavior.

On the one hand, *explicit memory* involves intentional recollection, analogous to declarative memory in humans. It consists of *semantic memory*, which stores general knowledge such as facts and concepts, and *episodic memory*,

which records specific events and interaction histories. Semantic memory in intelligent agents can be preloaded from domain knowledge bases or dynamically acquired through interactions. For example, in environments like TextWorld, semantic memory captures structured facts, such as "*Recipe − contains − Tuna*" or "*Recipe − is on − Table*". Episodic memory, in contrast, logs situational context and sequential actions, such as "go from kitchen to living room, then to garden". Integrating semantic and episodic memory allows agents to retain static and contextual information, enabling human-like adaptability and context-aware responses.

On the other hand, *implicit memory* shapes agent behavior through *procedural memory* and *priming*. Procedural memory enables agents to perform repetitive tasks efficiently by recalling specific skills and reusable plans. For example, it automates routine tasks without requiring explicit instructions, improving task execution efficiency. Priming, meanwhile, captures state changes and corresponding responses, allowing agents to adapt to similar contexts quickly. Priming enhances fluidity and context-sensitive decision-making by directly matching observations to or continuously chaining actions. Implicit memory, shaped by interactions with cognitive modules, enables rapid adaptation, often after minimal exposure to new stimuli.

Most intelligent agents implement both semantic and episodic memory within their memory modules. For instance, Agent S [211], designed for GUI automation tasks, incorporates semantic memory to store online web knowledge in natural language form, while episodic memory captures high-level, step-by-step task experiences. Similarly, AriGraph [221], targeting embodied simulation tasks, encodes semantic environment knowledge using a fact graph and logs episodic navigation history through an event graph. In AI companion systems like MemoryBank [207] for SiliconFriend, semantic memory constructs user portraits in natural language, while episodic memory retains interaction histories, enhancing personalized and context-aware behavior.

For implementing implicit memory, current agent systems primarily adopt model-friendly memory formats, such as key-value pair storage, executable code, or reusable routines. For example, AAG [226] defines and generalizes procedures through analogy, mapping knowledge from one situation (base) to another (target). This structure can be represented as a linear directed chain graph, where the input serves as the root, the output as the leaf node, and each intermediate step as a node in the chain. Similarly, Cradle [227] and Jarvis-1 [228] implement procedural memory by storing and retrieving skills in code form, which can be either learned from scratch or pre-defined. Once curated, skills can be added, updated, or composed within memory. The most relevant skills for a given task and context are then retrieved to support action planning.

3.4 The Memory Lifecycle

In this section, we introduce the lifecycle of memory in AI agents, as depicted in Figure 3.7. The lifecycle comprises a dual process of retention and retrieval. Retention includes acquisition, encoding, and derivation, while retrieval involves memory matching, neural memory networks, and memory utilization.

3.4.1 Memory Acquisition

Memory Acquisition is the foundational process by which intelligent agents take in raw perceptual information from their environment. This initial step is crucial for subsequent learning, adaptation, and decision-making [305]. A primary challenge in acquisition is the sheer volume and complexity of environmental inputs. Agents are constantly bombarded with visual, auditory, textual, and other forms of data, much of which is redundant or irrelevant to the agent's goals. Therefore, a core aspect of memory acquisition is not simply capturing data, but also initiating a preliminary filtering process. This filtering leverages two primary mechanisms: initial *information compression* and *experience consolidation*.

At this early stage, information compression involves rudimentary techniques to reduce data dimensionality. This might include downsampling images, extracting key phrases from text using simple heuristics, or identifying significant changes in audio streams [306]. The goal is rapid, lossy compression to prioritize potentially relevant information. For example, LMAgent [230] prompts the LLM to perform information compression, reducing irrelevant and unimportant content when constructing sensory memory to enhance operational efficiency. Meanwhile, ReadAgent [231] and GraphRead [307] respectively employ different strategies for compressing long text, i.e., episode pagination and graph-based structuring, to maximize information retention while ensuring efficiency.

On the other hand, experience consolidation, even at the acquisition phase, plays a role. The agent doesn't yet have a rich memory, but it can begin to apply previously learned, very general rules or biases. For example, if the agent has a pre-existing bias towards moving objects, it might prioritize visual data containing motion, even before full encoding [308]. To enhance the dynamic consolidation of memory-based experiences, [235] define metrics such as contextual relevance and recall frequency to determine whether to update long-term memory in a vector database.

Method	Domain	Memory Representation			Memory Lifecycle				
		Sensory	Short-term	Long-term	Acquisition	Encoding	Derivation	Retrieval	Utilization
Synapse [291]	GUI	Multi-modal	Context	Episodic, Procedural	User demo.	-	Hierarch. Decomp.	-	-
Agent S [211]	GUI	Multi-modal	Context, Working	Semantic, Episodic	Info. Compress.	Contrastive Learn.	Select. Forget.	Indexing	Long-context
Automanual [108]	GUI	Multi-modal	Context	Procedural, Episodic	User Demo.	Hierarch. Parse	Goal Decomp.	Task Search	Subgoal Exec.
AutoGuide [294]	GUI	Multi-modal	Context	-	Screen Capture	-	Action Plan	-	Action Exec.
Agent-Pro [295]	GUI	Multi-modal	Context	-	Screen Capture	-	Hierarch. Decomp.	-	Action Exec.
MemGPT [214]	Document	Text	Context, Working	-	External Data	-	-	Paging, Func. call	Doc. interact.
SeeAct [296]	Web	Multi-modal	Context	-	Screen Capture	-	Action Plan	-	Web Interact.
AutoWebGLM [297]	Web	Text	Context	-	HTML Parse	HTML Embed	HTML Analysis	-	Web Interact.
SteP [298]	Web	Text	Context	Task-spec.	HTML Parse	HTML Embed	HTML Analysis	Element Rank	Web Interact.
AWM [299]	Web	Text	-	Procedural	Workflow Extract.	Action Summ.	-	Sim. lookup	Workflow exec.
AriGraph [221]	TextWorld	Text	-	Semantic, Episodic	Env. Observ.	Knowl. Graph	Graph Traversal	Assoc. Retrieval	Action plan.
MemoryBank [207]	Dialogue	Text	-	Episodic	Dialogue Record	-	-	Chron. order	Resp. gen.
PromptAgent [300]	General	Text	Context	-	Prompting	-	Prompt Refine.	Content-based	Prompt Exec.
ECL [301]	Embody	Multi-modal	Context	Episodic	Obs. Recording	Contrast. Learn.	Exper. Summ.	Sim. & Recency	Policy Learn.
LEO [302]	Embody	Multi-modal	Working	Long-Horizon Rep.	Observation	Spatial-Temp. Learn.	Goal-Cond. Policy	Hierarch. Plan	Long-Horizon Exec.
IER [303]	Embody	Multi-modal	Context	Episodic	Env. Interact.	Multi-modal Embed	Iter. Refine.	Sim. Match	Action Plan.
Voyager [47]	Embody	Text	Working	Procedural	Auto. Curriculum	Skill Library	Iter. Prompt.	-	Skill Exec.
A3T [49]	Embody, Robotics	Text	Context	-	Task Decomp.	Token. & Embed.	Action Planning	-	Action select.
STARLING [304]	Robotics	Multi-modal	Context	Procedural	Demo.	Traj. Encode	Skill Refine.	Sim. & Context	Skill Exec.

Table 3.1: Summary of the memory module in various agents. Refer to Figure 3.6 for abbreviations.

Expel [69] constructs an experience pool to collect and extract insights from training tasks, facilitating generalization to unseen tasks. More recently, MindOS [233] proposed a working memory-centric central processing module for building autonomous AI agents, where working memory consolidates task-relevant experiences into structured thoughts for guiding future decisions and actions.

These two mechanisms work in concert with preliminary LLM input. To address the initial challenges, several mechanisms have to be deployed. Agents must be equipped with mechanisms to assess the potential relevance of incoming information rapidly. This preliminary filtering prevents cognitive overload. The acquisition phase also benefits from LLM.

3.4.2 Memory Encoding

Memory encoding builds upon acquisition by transforming the filtered perceptual information into internal representations suitable for storage and later use. A key aspect of encoding is selective filtering. This selective attention mimics human cognitive processes [309]. The inherent challenges of encoding stem from the complexity, high dimensionality, and often noisy nature of raw perceptual data. Effective encoding requires advanced mechanisms to identify key

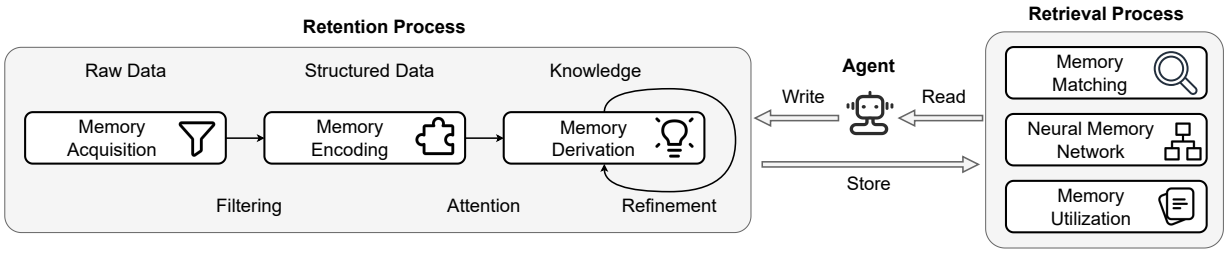


Figure 3.7: Illustration of the memory lifecycle. The memory retention process involves three sequential steps—memory acquisition, encoding, and derivation, while the memory retrieval process encompasses several independent applications, including matching (vector search), neural memory networks, and memory utilization (for long-context modeling and hallucination mitigation).

features, compress them compactly, and integrate information from multiple modalities. Modern approaches address these challenges by leveraging *selective attention* and *multi-modal fusion*.

Selective Attention mechanisms, inspired by human cognition, allow the agent to dynamically focus computational resources on the most relevant parts of the input. This might involve attending to specific regions of an image, keywords in a text, or particular frequencies in an audio signal. Different attention mechanisms can be used depending on the modality and task. For example, as the candidate memory dynamically expands, MS [237] employs an LLM-based scorer to selectively retain the top-scoring half, creating a more compact shared memory across multiple agent systems. In other modalities, GraphVideoAgent [238] utilizes graph-based memory to enable selective and multi-turn video scene understanding, enhancing question-answering performance. In robot control, [240] implements selective attention as a filtering mechanism to extract task-relevant objects from the set of all perceived objects on the table.

Multi-modal Fusion [310] is essential for integrating information from different sensory inputs (e.g., combining visual and auditory data to understand a scene). This involves creating a unified representation space where features from different modalities are aligned. Cross-modal encoders and contrastive learning techniques are often used to achieve this fusion. For example, JARVIS-1 [228] uses the general-domain video-language model CLIP [51] to compute alignment within a multimodal key-value memory, where the key comprises elements such as task, plan, and visual observations, and the value is a text-based representation of successfully executed plans. Furthermore, Optimus-1 [241] refines memory representation and optimizes the multimodal encoder by leveraging MineCLIP [311], a domain-specific video-language model pre-trained on Minecraft gameplay, to align and fuse filtered video streams with textual instructions and plans, encoding the agent's multimodal experiences into an abstracted memory pool. This integrated representation enhances information retrieval and reasoning across modalities and acts as another filter, reinforcing consistent data. LLMs' semantic understanding is utilized to extract relevant features efficiently.

3.4.3 Memory Derivation

Memory derivation focuses on extracting meaningful knowledge and insights from the acquired and encoded memories. This process goes beyond simple storage. This stage is essential for enhancing the agent's learning capabilities. The goal is to continuously optimize the structure and content of the agent's memory. A significant challenge in derivation is the dynamic evaluation of information value. Strategies to address these challenges include *reflection*, *summarization*, *knowledge distillation*, and *selective forgetting*.

Reflection involves an agent actively analyzing its memories to identify patterns, relationships, and potential inconsistencies. It can be triggered by specific events (e.g., an unexpected outcome) or occur periodically as a background process. This process may include comparing memories, reasoning about causal relationships, and generating hypotheses [300]. ExpeL [69] leverages reflection to collect past experiences for generalization to unseen tasks and to support trial-and-error reattempts following failures. R2D2 [243] models memory as a replay buffer and applies reflection to refine it by correcting failed execution trajectories in web agents. These corrected trajectories are then combined with successful ones to construct reflective memory, which serves as a reference for future decision-making.

Summarization aims to produce concise representations of larger bodies of information while preserving their most essential content. This can include extracting key sentences from a document, generating abstractive summaries of conversations, or condensing sequences of events. Summarization techniques range from simple extractive methods to advanced abstractive approaches powered by large language models (LLMs) [245, 312, 246]. For example, [248] introduces a recursive summarization strategy over dialogue history and prior memory to support long-term dialogue memory derivation. Building on this, Healthcare Copilot [247] maintains concise memory by transforming conversation

memory, representing the full ongoing medical consultation, into history memory that retains only key information relevant to the patient's medical history.

Knowledge distillation [313] enables agents to transfer knowledge from larger, more complex models (or ensembles) to smaller, more efficient ones. This is particularly important for resource-constrained agents and for enhancing generalization. Distillation can also involve consolidating knowledge from multiple specialized models into a single, general-purpose model. For example, AoTD [250] distills textual chains of thought from execution traces of subtasks into a Video-LLM to enhance multi-step reasoning performance in video question answering tasks. LDPD [251] transfers decision-making outcomes from teacher agents (i.e., expert buffers) to student agents, optimizing the student's policy to align with the teacher's. In multi-agent systems, MAGDi [253] distills the reasoning interactions among multiple LLMs into smaller models by structurally representing multi-round interactions as graphs, thereby improving the reasoning capabilities of smaller LLMs.

Selective forgetting [314] is the crucial process of removing or down-weighting memories that are deemed irrelevant, redundant, or outdated. This is essential for maintaining memory efficiency and preventing cognitive overload. Forgetting mechanisms can be based on time (older memories are more likely to be forgotten) [247], usage frequency (infrequently accessed memories are more likely forgotten) [203], and relevance to the current task or context [255]. In more fine-grained forgetting mechanisms, MemoryBank [207] applies the Ebbinghaus Forgetting Curve to quantify the forgetting rate, accounting for both time decay and the spacing effect, i.e., the principle that relearning information is easier than learning it for the first time. In contrast, Lyfe Agent [254] adopts a hierarchical summarize-and-forget strategy: it first clusters related memories, refines them into concise summaries, and then removes older memories that are highly similar to newer ones. This approach enables efficient, low-cost memory updates for real-time social interactions.

3.4.4 Memory Retrieval and Matching

Memory retrieval is a process that emulates the human ability to recall relevant knowledge and experiences to solve problems. The goal is to efficiently and accurately extract the most pertinent memory fragments from a large and diverse memory pool, encompassing sensory, short-term, and long-term memory, to inform the agent's decisions, planning, and actions. Just as humans rely on past experiences to navigate complex situations, agents require a sophisticated memory retrieval mechanism to handle a wide range of tasks effectively.

However, achieving this goal presents several significant challenges. First, the agent's memory repository is often heterogeneous, comprising various forms of memory such as natural language descriptions, structured knowledge graphs, and state-action-reward sequences. These memories differ fundamentally in their data structures, representations, and levels of semantic granularity, posing a challenge for unified retrieval. Second, the retrieved memory fragments must be highly relevant to the current context, including the agent's state, task goals, and environmental observations. Simple keyword matching falls short of capturing the deeper semantic relationships required for meaningful retrieval. Developing a context-aware semantic matching mechanism that can dynamically adjust the retrieval strategy based on the current situation is therefore paramount. Third, the real-time nature of agent interaction with the environment necessitates efficient memory retrieval to support rapid decision-making and action [315]. This demand for efficiency is further compounded by the limitations of the agent's computational resources. Finally, the agent's memory is not static but constantly evolving as new experiences, knowledge, and skills are acquired. Ensuring memories' timeliness, reliability, and relevance while avoiding the interference of outdated or erroneous information is a continuous challenge.

A comprehensive approach can address these challenges, encompassing four key components. Firstly, a foundational step involves constructing a unified memory representation and indexing scheme. This aims to bridge the representational gap between different memory types by embedding them into a common vector space. Pre-trained language models like BERT or Sentence-BERT [316] can be leveraged to transform text-based memories into semantic vectors, while graph neural networks (GNNs) can learn vector representations for structured memories like knowledge graphs, capturing both node and edge relationships [317]. To facilitate efficient retrieval, a multi-layered hybrid indexing structure is essential. This integrates techniques like inverted indexes for keyword matching, vector indexes like Faiss [318] or Annoy [319] for similarity search, and graph indexes for structural queries [320], thus supporting diverse query needs.

Secondly, perhaps most critically, the system must develop context-aware semantic similarity computation. This allows the retrieval process to understand and utilize the current context, such as the agent's state, goals, and observations, enabling a deeper semantic match beyond keyword overlap. This involves encoding the contextual information into vector representations and effectively fusing them with memory vectors. The attention mechanism plays a crucial role here, dynamically calculating the relevance between context and memory vectors and assigning different weights to memory fragments based on their contextual relevance [261]. This emphasizes memories that are more pertinent to the current situation.

Thirdly, integrating memory retrieval with the agent's task execution necessitates a task-oriented sequence decision and dynamic routing mechanism. This leverages the structural information of tasks to guide memory retrieval and utilization, enabling complex task decomposition, planning, and dynamic adjustments. By constructing a task dependency graph, the agent can topologically sort subtasks to determine execution order. During execution, each subtask's goal serves as context for memory retrieval, extracting relevant knowledge and experience. Moreover, the agent must adapt to environmental feedback and task progress, dynamically adjusting the execution plan. Each decision point involves re-retrieving memories based on the current state and goal to select the optimal action and handle unexpected situations. This aspect also emphasizes how agents can leverage their skill memory to solve problems, including skill distillation, combination, and innovation. Pattern recognition allows for summarising general problem-solving steps, while structured knowledge organization arranges skills into a retrievable format. Agents can further distill generalized skills from specific ones, combine multiple skills to address complex challenges, and even innovate new skill combinations. These processes depend fundamentally on an efficient memory retrieval system that can identify appropriate skills or skill combinations based on task requirements.

Finally, a robust memory management mechanism is crucial for maintaining the memory pool's timeliness, relevance, and efficiency. This mechanism should incorporate a forgetting and updating strategy, mirroring human forgetting mechanisms [321]. This might involve regularly purging outdated, redundant, or infrequently used memories based on time-based decay (weakening memory strength over time) and frequency-based decay (purging low-frequency memories). Simultaneously, when a memory fragment relevant to the current task is retrieved, its timestamp and access frequency are updated, increasing its importance and ensuring dynamic memory updates. Through these concerted efforts, LLM Agents can be equipped with a powerful, flexible, and context-aware memory retrieval and matching system, enabling them to effectively utilize their accumulated knowledge, support complex decision-making, and exhibit more intelligent behavior.

3.4.5 Neural Memory Networks

Neural Memory Networks represent a fascinating frontier in AI research. They aim to integrate memory seamlessly into the fabric of neural networks. This approach departs from traditional memory architectures by encoding memories directly within the network's weights or activations, transforming the network into a dynamic, read-write memory storage medium. This tight integration promises significant advancements in efficiency and the utilization of stored information. However, realizing this vision presents several formidable challenges.

A primary concern is balancing memory capacity with stability. Encoding a vast amount of information within the finite parameters of a neural network while maintaining long-term stability poses a major hurdle. The network must be able to store a multitude of memories without succumbing to catastrophic forgetting or confusion between similar memories. Equally crucial is the development of effective mechanisms for memory read-write operations. The network needs to reliably write new information, update existing memories, and accurately retrieve stored information on demand, all while maintaining computational efficiency. Beyond simply storing memories, the ultimate goal is to endow neural networks with the ability to generalize from and reason with the information they store. This would empower them to perform higher-order cognitive functions beyond rote memorization, allowing for insightful connections and inferences based on past experiences. Several approaches are being explored to address these challenges, notably through *associative memory* and *parameter integration*.

On the one hand, associative memory, inspired by the interconnectedness of neurons in the brain, offers a promising avenue. Models like Hopfield networks [262, 263], leveraging energy functions, and Bidirectional Associative Memories (BAMs) [322], supporting hetero-associative recall, provide mechanisms for encoding and retrieving patterns based on the weights between neurons. Besides, Neural Turing Machines (NTMs) [264] and Memory-Augmented Neural Network (MANNs) [323, 324, 275, 265] augment neural networks with external memory modules, employing attention and summary mechanisms to interact with these memories.

On the other hand, parameter integration represents another key research direction, aiming to encode memory directly within a network's weights. This facilitates the seamless integration of world knowledge and accumulated experience into the operational behavior of intelligent AI agents. For example, some prior works modify model parameters to enable continual learning by updating [325, 326, 327] or forgetting specific knowledge [328]. Other studies treat LLMs as standalone memory modules, incorporating world knowledge into their parameters during pre-training [329], post-training [330], and online deployment [331]. For instance, MemoryLLM [265] introduces memory tokens, while SELF-PARAM [266] leverages knowledge distillation to embed world knowledge and past AI agent experiences into model parameters. This approach is further augmented in the M+ model [332] with a long-term memory mechanism and a co-trained retriever, enhancing its ability to generalize to longer history memorization. Additionally, [333] employs encoded memory to facilitate further reasoning, thereby improving the generalization of stored knowledge. More recently, MemoRAG [267] and R^3Mem [270] have been proposed to not only encode memory but also enable

reliable retrieval from neural memory networks, unifying the dual processes of memory storage and retrieval within a single model. This advancement contributes to the development of next-generation generative-based retrieval systems, which support lifelong AI applications. Furthermore, Titans [269] have been introduced to memorize test-time data points through meta-learning, enabling more efficient test-time cross-task generalization.

Future research will continue to focus on creating larger capacity and more stable neural memory models. Concurrently, developing more efficient and flexible memory read-write mechanisms will be crucial. A critical area of investigation will involve applying these memory-augmented networks to complex cognitive tasks, pushing the boundaries of what AI can achieve. Progress in this domain will unlock new possibilities for building intelligent agents that can learn, remember, and reason in a manner that is increasingly reminiscent of human cognition.

3.4.6 Memory Utilization

A critical aspect of agent design lies in memory utilization, which focuses on maximizing the value of stored memory segments for the current task. The core objective is to apply these memories effectively and appropriately to enhance reasoning, decision-making, planning, and action generation, ultimately boosting the agent's performance and efficiency while avoiding the pitfalls of irrelevant or incorrect memory interference. Achieving this, however, presents several challenges.

One primary challenge is balancing the vastness of the memory store with its effective utilization. Agents must navigate a potential information overload, ensuring that relevant memories are fully leveraged without overwhelming the system. Another hurdle is the need for abstraction and generalization. Agents need to distill specific memory segments into more general knowledge and apply this knowledge to new and varied situations. Furthermore, the issue of hallucinations and incorrect memories within the LLM requires careful consideration. Preventing the generation of content that contradicts or misrepresents stored information is crucial, as is the ability to identify and rectify erroneous information that may reside within the memory store itself.

To address these challenges, several strategies are employed. *Retrieval-augmented generation (RAG)* [334] combines retrieval and generation models to enhance the LLM's capabilities by drawing upon external knowledge sources. Unlike the methods mentioned in memory retrieval and matching, RAG focuses on integrating retrieved information into the generation process itself. When prompted, the agent retrieves relevant memory segments and incorporates them into the context provided by the generation model. This contextual enrichment guides the model towards more factual and informative outputs. For instance, when responding to a user's query, the agent can first retrieve related entries from its knowledge base and then generate an answer based on this retrieved information, thus grounding the response in established knowledge. More recently, some studies have integrated memory modules with RAG, incorporating self-reflection [274] and adaptive retrieval mechanisms [272] to enhance both generation reliability and efficiency. For example, Atlas [273] leverages causal mediation analysis, while [284] employs consistency-based hallucination detection to determine whether the model already possesses the necessary knowledge—allowing for direct generation—or whether retrieval is required, in which case the model first retrieves relevant information before generating a response. In a unified framework, RAGLAB [271] offers a comprehensive ecosystem for evaluating and analyzing mainstream RAG algorithms. HippoRAG [222] employs a strategy inspired by the hippocampal indexing theory of human memory to create a KG-based index for memory and use Personalized PageRank for memory retrieval.

Furthermore, *long-context modeling* plays a vital role in managing extensive memory stores. This approach enhances the LLM's ability to process long sequences and large-scale memories, allowing for a deeper understanding and utilization of long-range dependencies. By employing Transformer model variants like Transformer-XL [324] and Longformer [335], or through hierarchical and recursive processing techniques, such as recurrent memory transformer (RMT) [275, 276], agents can expand their context window. This enables them to handle significantly more extensive memory stores and reason and make decisions within a much broader context. For example, agents can maintain a longer memory span when processing extensive documents or engaging in prolonged conversations. Additionally, some studies leverage memory to compress long contexts, enabling more effective long-context modeling. For example, AutoCompressor [277] introduces summary vectors as memory to transfer information from previous context windows into the current window, facilitating long-context understanding. Similarly, the in-context autoencoder (ICAE) [278] generates memory slots that accurately and comprehensively represent the original context, while LLMLingua [336, 337], Gist [279], and CompAct [280] further optimize long-prompt compression to reduce input context length.

Finally, *hallucination mitigation* strategies are essential for ensuring the reliability of generated outputs. These strategies aim to minimize the LLM's tendency to produce factually incorrect or nonsensical content. One approach is implementing fact-checking mechanisms [338], verifying generated content against established knowledge or memory stores. Another involves uncertainty estimation [339, 340], where the model evaluates the confidence level of its generated content and flags or filters out low-confidence outputs. Additionally, knowledge-based decoding strategies can

be employed during the generation phase, introducing constraints that guide the model towards more factually accurate content. These techniques collectively contribute to generating more trustworthy outputs and aligned with the agent's established knowledge base. Recent research has introduced expert memory subnetworks, such as PEER [283] and Lamini Memory Tuning [281], which specialize in memorizing specific types of information, including world knowledge and AI agents' past experiences. These subnetworks offload memorization to dedicated parameters, reducing the main model's propensity to hallucinate. By implementing these memory utilization strategies, agents can become more capable, accurate, and reliable. They can successfully leverage their memory stores to achieve superior performance across complex tasks.

3.5 Summary and Discussion

The development of truly intelligent agents depends not just on robust memory systems, but also on their seamless integration with other cognitive functions like perception, planning, reasoning, and action selection. Memory is not an isolated module; it is deeply intertwined with these other processes. For example, sensory input is encoded and filtered before storage (as discussed in the sections on memory representation and lifecycle), highlighting the interplay between perception and memory. Long-term memory, especially procedural memory, directly informs action selection through learned skills and routines. Retrieval mechanisms, like context-aware semantic similarity computation, are crucial for planning, allowing agents to access relevant past experiences. This interplay extends to the concept of a "world model."

Central to intelligent agents is their ability to build and utilize internal world models. These models, representing an agent's understanding of its environment, enable simulation, reasoning about consequences, and prediction. Robust world models are crucial for higher-level cognition, planning, and human-like intelligence. A world model is, in essence, a highly structured, often predictive, form of long-term memory. Memory provides the raw material—knowledge and experiences—for constructing the world model, while the world model, in turn, acts as an organizing framework, influencing how new memories are encoded, consolidated, and retrieved. For instance, a well-developed world model might prioritize storing surprising events, as these indicate gaps in the agent's understanding.

However, developing effective world models and memory systems presents significant challenges. These include managing the complexity of real-world environments, determining the appropriate level of abstraction (balancing accuracy, complexity, and computational efficiency), and integrating multi-modal information. Learning and updating these models efficiently, avoiding bias, ensuring generalization, and enabling continuous adaptation are also critical. Furthermore, model-based planning requires efficient search algorithms to handle the inherent uncertainty in the model's predictions.

Future research should focus on enhancing agent memory systems by drawing inspiration from the strengths of human memory, particularly its flexibility, adaptability, and efficiency. While agent memory has advanced considerably, it still lags behind human memory in these key areas. Human memory is remarkably associative, retrieving information from incomplete or noisy cues, and it exhibits a sophisticated form of "forgetting" that involves consolidation and abstraction, prioritizing relevant information and generalizing from experiences. Agent memory, conversely, often relies on precise matching and struggles with ambiguity.

Several promising research directions emerge. Exploring biologically-inspired mechanisms, such as neural memory networks (as discussed earlier), could lead to significant breakthroughs. Another crucial area is developing memory systems that actively "curate" their contents—reflecting on information, identifying inconsistencies, and synthesizing new knowledge. This requires integrating metacognitive capabilities (monitoring and controlling one's own cognitive processes) into agent architectures. Furthermore, creating more robust and nuanced forms of episodic memory, capturing not just the "what" and "when" but also the "why" and the emotional context of events, is essential for agents that can truly learn from experience and interact with humans naturally.

Overcoming these challenges requires innovative solutions at the intersection of deep learning, reinforcement learning, and cognitive science. Developing more sophisticated and adaptable world models and memory systems—ones that mirror the strengths of human cognition—will pave the way for agents with a deeper understanding of their environment, leading to more intelligent and meaningful interactions.

Chapter 4

World Model

A world model enables an agent to predict and reason about future states without direct trial-and-error in reality. This section explores how human cognitive studies on "mental models" relate to AI world models in artificial intelligence, categorizing them under four paradigms: *implicit paradigm*, *explicit paradigm*, *simulator-based paradigm*, and a class of other emergent methods (e.g., *instruction-driven paradigm*). We then discuss how world models inherently intersect with other agentic components and conclude with open questions and future directions that unite these perspectives under a unified theoretical and practical framework.

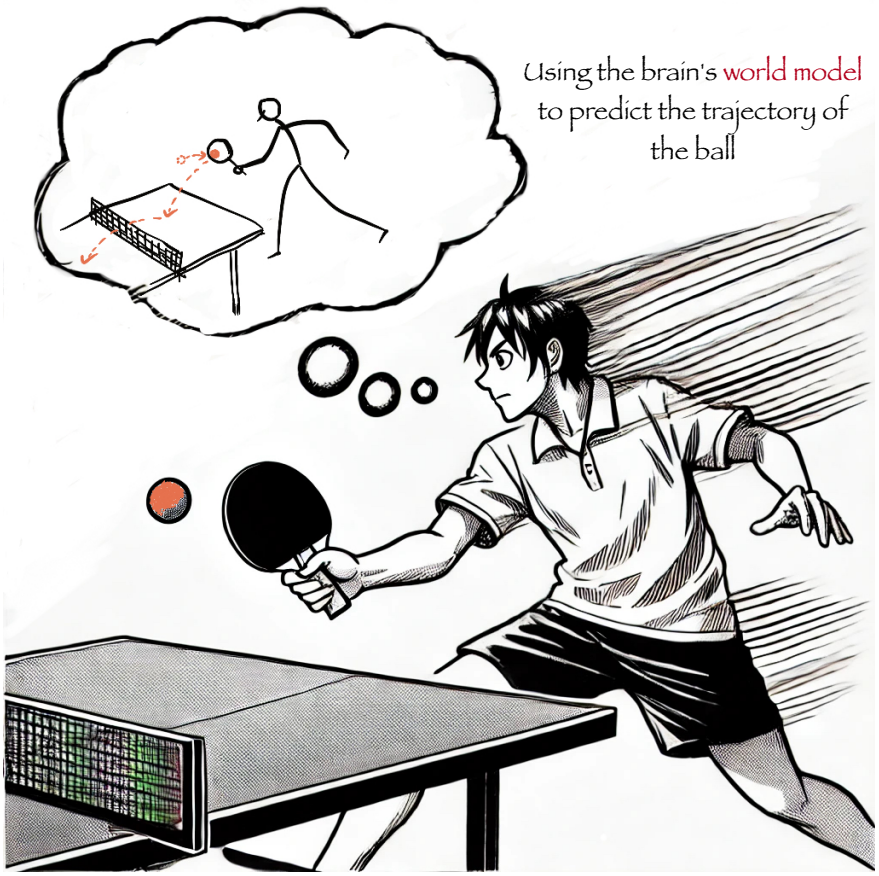


Figure 4.1: Humans can use their brain's model of the world to predict the consequences of their actions. For example, when playing table tennis, a player can imagine or predict the trajectory of the ball after an action.

4.1 The Human World Model

Humans naturally construct internal representations of the world, often referred to as *mental models* in psychology [341, 342, 343]. These models serve as compact and manipulable depictions of external reality, enabling individuals to predict outcomes, plan actions, and interpret novel scenarios with minimal reliance on direct trial-and-error. Early work on spatial navigation, for instance, showed that humans and animals form "cognitive maps" of their surroundings [341], suggesting an underlying ability to imagine potential paths before actually traversing them.

Craik's seminal argument was that the human mind runs internal "small-scale models of reality" [342] to simulate how events might unfold and evaluate possible courses of action. Later studies proposed that such simulations stretch across modalities—vision, language, and motor control—and are dynamically updated by comparing predictions to new observations. This process merges *memory recall* with *forward projection*, implying a close interplay between stored knowledge and the active generation of hypothetical future states [343]. More recent predictive processing theories such as "Surfing Uncertainty" [344] propose that the brain operates as a hierarchical prediction machine, continuously generating top-down predictions about sensory inputs and updating its models based on prediction errors.

Critically, these human mental models are:

- **Predictive:** They forecast changes in the environment, informing decisions about where to move or how to respond.
- **Integrative:** They combine sensory input, past experience, and abstract reasoning into a unified perspective on "what might happen next".
- **Adaptive:** They are revised when reality diverges from expectation, reducing the gap between imagined and actual outcomes over time.
- **Multi-scale:** They operate seamlessly across different temporal and spatial scales, simultaneously processing immediate physical dynamics (milliseconds), medium-term action sequences (seconds to minutes), and long-term plans (hours to years). This flexibility allows humans to zoom in on fine-grained details or zoom out to consider broader contexts as needed.

Consider hunger and eating as an illustration of integrated world modeling. When hungry, a person's internal model activates predictions about food—simulating not just visual appearance but tastes, smells, and anticipated satisfaction—triggering physiological responses like salivation before food is even present. This demonstrates seamless integration across perception, memory, and action planning.

The example also highlights adaptivity: once satiated, the same model dynamically updates, reducing predicted reward values for further eating. Despite recognizing the same food items, their anticipated utility changes based on internal state. Furthermore, humans maintain counterfactual simulations—declining dessert now while accurately predicting they would enjoy it later—enabling complex planning across hypothetical scenarios and time horizons, a capability comprehensive AI world models strive to replicate.

In sum, the *human world model* is not a static library of facts, but a flexible and ever-evolving mental construct, deeply rooted in perception and memory, that continuously shapes (and is shaped by) the individual's interactions with the outside world.

4.2 Translating Human World Models to AI

Research in artificial intelligence has long sought to replicate the *predictive, integrative, and adaptive* qualities exhibited by human mental models [341, 342]. Early reinforcement learning frameworks, for instance, proposed learning an *environment model* for planning—exemplified by Dyna [345]—while contemporaneous work investigated using neural networks to anticipate future observations in streaming data [346, 347]. Both directions were motivated by the idea that an internal simulator of the world could enable more efficient decision-making than purely reactive, trial-and-error learning.

Subsequent advancements in deep learning brought the notion of "AI world models" into sharper focus. One influential approach introduced an end-to-end *latent generative model* of an environment (e.g., "World Models" [348]), whereby a recurrent neural network (RNN) and variational auto-encoder (VAE) together learn to "dream" future trajectories. These latent rollouts allow an agent to train or refine policies offline, effectively mirroring how humans mentally rehearse actions before executing them. Alongside such implicit designs, explicit forward-modeling methods emerged in model-based RL, letting agents predict $P(s' \mid s, a)$ and plan with approximate lookahead [349, 350].

Another branch of work leveraged large-scale simulators or real-world robotics to ground learning in richly diverse experiences [351, 352]. Such setups are reminiscent of how human children learn by actively exploring their environments, gradually honing their internal representations. Yet a key question lingers: can agentic systems unify these approaches (implicit generative modeling, explicit factorization, and simulator-driven exploration) into a cohesive "mental model" akin to that observed in humans? The recent proliferation of language-model-based reasoning [107, 74] hints at the potential to cross modalities and tasks, echoing how humans integrate linguistic, visual, and motor knowledge under one predictive framework.

Overall, as AI systems strive for flexible, sample-efficient learning, the *AI world model* stands as a conceptual bridge from cognitive theories of mental models to implementations that equip artificial agents with *imagination*, *predictive reasoning*, and *robust adaptation* in complex domains.

4.3 Paradigms of AI World Models

Designing an *AI world model* involves determining how an AI agent acquires, represents, and updates its understanding of the environment's dynamics. While implementations vary, most approaches fall into four broad paradigms: *implicit*, *explicit*, *simulator-based*, and *hybrid or instruction-driven* models. These paradigms can be further analyzed along two key dimensions: reliance on *internal* (neural-based) vs. *external* (rule-based or structured) mechanisms, and overall *system complexity*. Figure 4.2 illustrates this two-dimensional space, showing how different approaches distribute themselves across these axes. Generally, implicit models tend to rely more on internal mechanisms, while explicit and simulator-based models incorporate more external structures. Simulator-based and explicit models also tend to be more complex than implicit and hybrid approaches, reflecting their structured reasoning and engineered constraints.

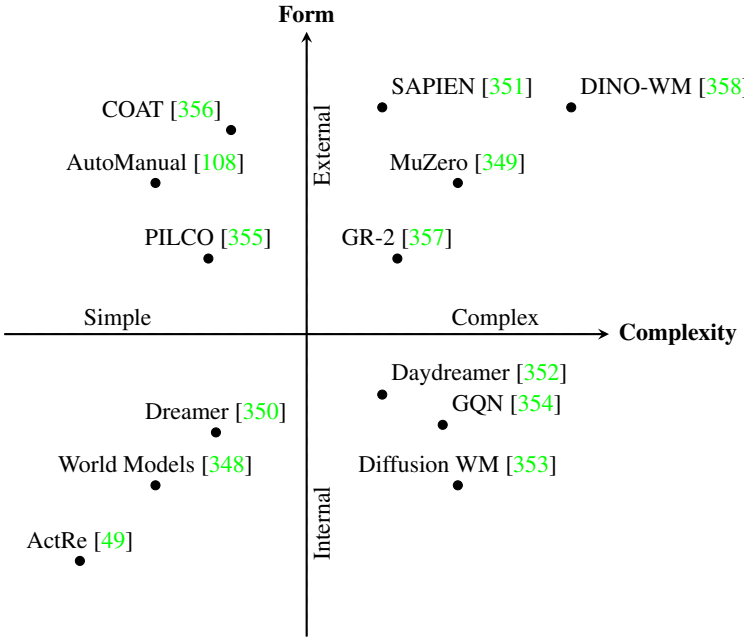


Figure 4.2: A two-dimensional layout of AI world-model methods. The horizontal axis indicates *Complexity* (left to right). The vertical axis spans *Internal* approaches (bottom) to *External* solutions (top). Approximate positions reflect each method's reliance on large learned networks vs. explicit rules or code, and its overall system complexity.

4.3.1 Overview of World Model Paradigms

An *AI world model* is broadly any mechanism by which an agent captures or accesses approximate environment dynamics. Let $\mathcal{S}$ denote the set of possible environment *states*, $\mathcal{A}$ the set of *actions*, and $\mathcal{O}$ the set of *observations*. In an idealized Markovian framework, the environment is characterized by transition and observation distributions:

$$T(s' \mid s, a) \quad : \quad \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S}), \tag{4.1}$$

$$O(o \mid s') \quad : \quad \mathcal{S} \rightarrow \Delta(\mathcal{O}), \tag{4.2}$$

where $T(\cdot)$ dictates how states evolve under actions, and $O(\cdot)$ defines how states produce observations. A **world model** typically *learns* or *utilizes* approximations of these functions (or a variant), allowing the agent to *predict* future states or observations without executing real actions in the environment.

Numerous approaches exist to implement these approximations, which we group into four main **paradigms**:

- **Implicit paradigm:** A single neural network or latent structure encodes both transition and observation mappings without explicit factorization. World Models [348] or large language models used for environment reasoning are typical examples. Agents generally unroll this black-box function to simulate hypothetical trajectories.
- **Explicit paradigm:** The agent directly models or has access to learnable transition model T_θ and observation model O_θ, often enabling interpretability or modular design. Model-based RL methods–like MuZero [349] or Dreamer [350]–learn or refine T_θ, planning in an approximated state space. Generative visual models such as [353, 358] fall under this category if they explicitly predict the next states or frames.
- **Simulator-Based paradigm:** Rather than approximating (4.1)–(4.2), the agent relies on an external simulator or even the physical world as the ground-truth. Systems like SAPIEN [351] or real-robot pipelines [352] can be seen as "native" environment models that the agent queries. Although no learned $T(\cdot)$ is required, the agent pays a cost in terms of runtime or real-world risks.
- **Other paradigms (Hybrid or Instruction-Driven):** Methods that defy simple classification. They may store emergent rules in textual form [108], refine implicit LLM knowledge into partial causal graphs [356], or combine external components with learned sub-modules. Such approaches highlight the evolving nature of world-model research, where instructions, symbolic rules, or on-the-fly structures can complement more traditional approximations.

Throughout the remainder of this subsection, we examine how each paradigm addresses (or circumvents) Equations (4.1) and (4.2), the trade-offs in interpretability and scalability, and their relative merits for different tasks ranging from text-based to high-dimensional embodied control.

4.3.2 Implicit Paradigm

In the **implicit** paradigm, an agent encodes all environment dynamics—including how states evolve and how observations are generated—within a single (or tightly coupled) neural model. Formally, one maintains a latent state h_t that is updated according to

$$h_{t+1} = f_\theta(h_t, a_t), \quad \hat{o}_{t+1} = g_\theta\big(h_{t+1}\big), \tag{4.3}$$

where f_θ subsumes the transition function $T(\cdot)$ (and part of $O(\cdot)$) from Eqs. (4.1)–(4.2), but without making these components explicit. A classic example is the *World Models* framework [348], in which a Variational Autoencoder (VAE) first compresses visual inputs into latent codes, and a recurrent network predicts the next latent code, effectively "dreaming" trajectories in latent space. Recent work also explores repurposing large language models (LLMs) for environment simulation in purely textual or symbolic domains [107, 74], although these models are not always grounded in strict time-series or physics-based data.

Because implicit models fuse the transition and observation mechanisms into one monolithic function, they can be elegantly trained end to end and unrolled internally for planning. However, they tend to be opaque: it is difficult to interpret how precisely the network captures domain constraints or to inject knowledge directly into any part of the transition. This can be advantageous for highly complex environments where a single large-capacity model can discover latent structure on its own, but it also risks brittleness under distribution shifts. Overall, the implicit paradigm is appealing for its simplicity and flexibility, but it can pose challenges when interpretability, explicit constraints, or fine-grained control of the dynamics are required.

4.3.3 Explicit Paradigm

The **explicit** paradigm instead factorizes the world model, often by learning or encoding a transition function $\hat{T}_\theta(s_{t+1} \mid s_t, a_t)$ and an observation function $\hat{O}_\theta(o_{t+1} \mid s_{t+1})$. This explicit separation makes it possible to query each function independently. For instance, one might draw samples from

$$\hat{s}_{t+1} \sim \hat{T}_\theta(s_t, a_t), \quad \hat{o}_{t+1} \sim \hat{O}_\theta\big(\hat{s}_{t+1}\big). \tag{4.4}$$

Model-based reinforcement-learning algorithms like MuZero [349] or Dreamer [350] exemplify this paradigm by refining a forward model for planning. Other explicit approaches prioritize fidelity in generating future frames, such as

Diffusion WM [353], which applies diffusion processes at the pixel level, or DINO-WM [358], which rolls out future states within a pretrained feature space.

By factorizing transitions and observations, explicit methods can be more interpretable and more amenable to debugging and domain-specific constraints. That said, they are still sensitive to model errors: if $\hat{T}_\theta$ deviates significantly from reality, the agent's planning and decision-making can become ineffective. Many explicit systems still rely predominantly on internal (neural) representations, but they may integrate external planners (e.g., tree-search algorithms) to leverage the explicit transition structure. This blend of learned and symbolic components offers a natural way to incorporate human knowledge, while preserving the strengths of deep learning.

4.3.4 Simulator-Based Paradigm

In the **simulator-based** paradigm, the agent outsources environment updates to a simulator, effectively bypassing the need to learn $\hat{T}_\theta$ from data. Formally,

$$(s_{t+1}, o_{t+1}) \leftarrow \mathcal{SIM}(s_t, a_t), \tag{4.5}$$

where $\mathcal{SIM}$ is often an external physics engine or the real world itself. Platforms like SAPIEN [351] and AI Habitat provide deterministic 3D physics simulations, allowing agents to practice or iterate strategies in a controlled environment. Alternatively, methods such as Daydreamer [352] treat real-world interaction loops like a "simulator," continually updating on-policy data from physical robots.

This approach yields accurate transitions (assuming the simulator accurately reflects reality), which alleviates the risk of learned-model errors. However, it can be computationally or financially expensive, especially if the simulator is high fidelity or if real-world trials are time-consuming and risky. As a result, some agents combine partial learned dynamics with occasional simulator queries, aiming to balance accurate rollouts with efficient coverage of state-action space.

4.3.5 Hybrid and Instruction-Driven Paradigms

Beyond these three primary paradigms, there is a growing number of **hybrid** or **instruction-driven** approaches, which blend implicit and explicit modeling or incorporate external symbolic knowledge and large language models. Often, these systems dynamically extract rules from data, maintain evolving textual knowledge bases, or prompt LLMs to hypothesize causal relationships that can then be tested or refined.

AutoManual [108], for example, iteratively compiles interactive environment rules into human-readable manuals, informing future actions in a more transparent way. Meanwhile, COAT [356] prompts an LLM to propose possible causal factors behind observed events, then validates or refines those factors via direct interaction, bridging text-based reasoning with partial learned models. Although these solutions offer remarkable flexibility—particularly in adapting to unfamiliar domains or integrating real-time human insights—they can be inconsistent in how they structure or update internal representations. As language-model prompting and real-time rule discovery continue to evolve, these hybrid methods are poised to become increasingly common, reflecting the need to balance end-to-end learning with the transparency and adaptability offered by external instruction.

Until now, we have introduced the four typical paradigms of existing world model techniques, as illustrated in Figure 4.3.5. As we can see, each type of technique has trade-offs in different aspects.

4.3.6 Comparative Summary of Paradigms

The table summarizes the key methods in AI world modeling, categorizing them based on their reliance on *external* or *internal* mechanisms, their complexity, and their respective paradigms. The form column uses ∘ for external approaches and • for internal ones, with mixed methods having both symbols. This classification aligns with the previous subsections, including the detailed discussion of each paradigm, and complements the visual representation in Figure 4.2.

4.4 Relationships to Other Modules

A comprehensive AI world model does not exist in isolation but interacts with several key components of the agent's architecture. These include (but not limited to) the memory, perception, and action modules. In this subsection, we explore how world models integrate with these critical components to enable coherent and adaptive behavior in dynamic environments.

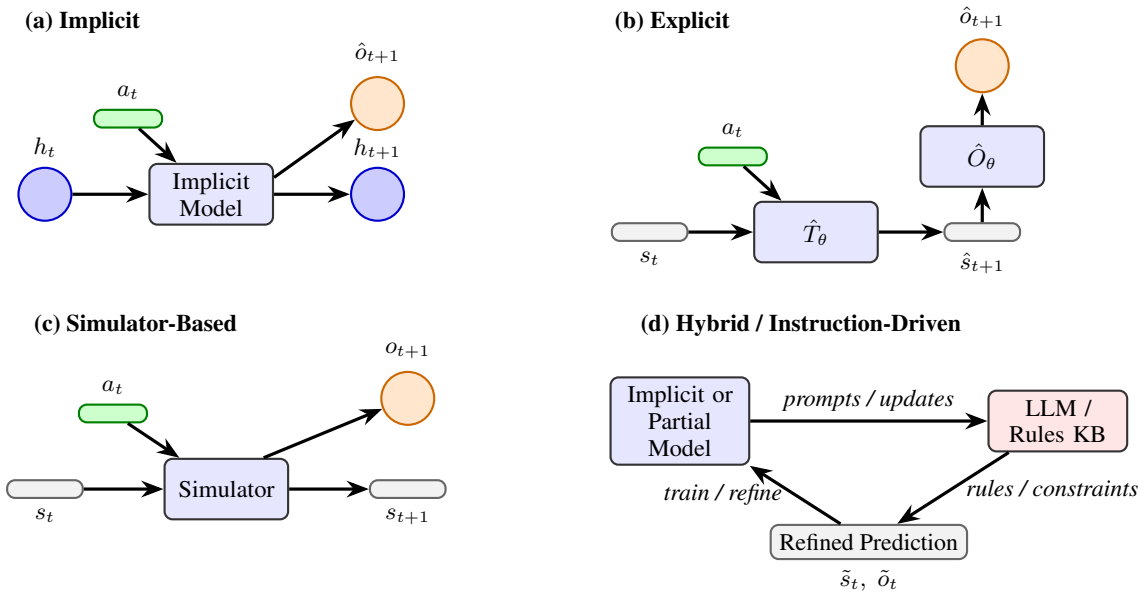


Figure 4.3: Four paradigms of world modeling: (a) implicit, (b) explicit, (c) simulator-based, and (d) hybrid/instruction-driven.

Table 4.1: Summary of AI world-model methods across paradigms, showing their form (External or Internal), complexity, and paradigm.

Method	Form	Complexity	Paradigm
ActRe [49]	•	Simple	Implicit
World Models [348]	•	Simple	Implicit
Dreamer [350]	•	Moderate	Implicit
Diffusion WM [353]	•	High	Explicit
GQN [354]	•	High	Explicit
Daydreamer [352]	○	High	Simulator-based
SAPIEN [351]	○	High	Simulator-based
PILCO [355]	○	Moderate	Explicit
AutoManual [108]	○	Simple	Other
MuZero [349]	○	High	Explicit
GR-2 [357]	•	High	Explicit
DINO-WM [358]	•	High	Explicit
COAT [356]	○	Moderate	Other

4.4.1 Memory and the World Model

Memory systems play a crucial role in the operation of world models. While a world model generates predictive representations of future states or actions, memory serves as the foundation upon which these representations are built and updated. The relationship between the world model and memory can be viewed as a loop where the world model predicts potential futures, while the memory stores past experiences, observations, and learned patterns, allowing for context-dependent reasoning and future predictions.

Memory mechanisms can be structured in various ways, including:

- **Short-term memory**: This enables the agent to hold and update its internal state temporarily, storing the most recent interactions or observations. This short-term context helps the agent make decisions in the immediate environment.
- **Long-term memory**: This serves as a more persistent repository of experiences and general knowledge about the environment. A world model can interact with long-term memory to refine its predictions, and it may use historical data to make more informed decisions or simulate more realistic futures.

For example, in model-based RL frameworks like Dreamer [350], recurrent neural networks act as both the world model and a form of memory, maintaining a latent state that is updated with each time step to predict future states. This form of integrated memory allows the agent to both recall past interactions and anticipate future ones.

4.4.2 Perception and the World Model

Perception refers to the agent's ability to sense and interpret its environment through various modalities (e.g., vision, touch, sound, etc.). The world model relies heavily on accurate sensory input to form coherent predictions about the environment. In many AI systems, the perception module converts raw sensor data into a higher-level representation, such as an image, sound wave, or other structured data.

A key aspect of the interaction between the world model and perception is how the agent processes and integrates sensory input into the model. The world model often depends on processed data (such as features from convolutional neural networks or embeddings from transformers) to simulate potential futures. Additionally, the world model can guide perceptual processes by focusing attention on the most relevant sensory input needed to refine predictions.

For example, in autonomous robotics, perception systems typically detect objects or environmental features, which are then fed into a world model that predicts how the scene will evolve. RoboCraft [359] achieves this perception-to-modeling transformation by converting visual observations into particles and capturing the underlying system structure through graph neural networks. PointNet [360] further enriches perception systems' understanding of physical space by encoding unstructured 3D point clouds to capture spatial characteristics of the environment. In navigation tasks, OVER-NAV [361] further combine large language models and open-vocabulary detection to construct the relationship between multi-modal signals and key information, proposing an omni-graph to capture the structure of local space as the world model for navigation tasks. This feedback loop between perception and the world model enables agents to update their perception dynamically based on ongoing predictions, allowing for real-time adaptation.

4.4.3 Action and the World Model

Action refers to the decision-making process through which an agent interacts with its environment. In agentic systems, actions are driven by the world model's predictions of future states. The world model aids in planning by simulating the outcomes of different actions before they are executed, allowing the agent to choose the most optimal course of action based on the predicted consequences.

The integration between world models and action modules can take various forms:

- **Model-based planning**: World models explicitly model the environment's transition dynamics [349, 362, 107], allowing the agent to simulate multiple action sequences (rollouts) before selecting the most optimal one.
- **Exploration**: World models also support exploration strategies by simulating unseen states or unexpected actions [363, 350, 364]. These simulations enable the agent to evaluate the potential benefits of exploring new parts of the state space.

In model-based planning, MuZero [349] performs implicit planning through self-play and Monte Carlo Tree Search (MCTS), transforming current state representations into future state and reward predictions to guide the decision-making process without prior knowledge of environment rules. In contrast, MPC [362] utilizes explicit dynamics models to predict multiple possible trajectories within a finite time horizon, determines the optimal control sequence by solving an optimization problem, and continuously updates planning using a receding horizon approach. Alpha-SQL [365], on the other hand, integrates an LLM-as-Action-Model within an MCTS framework to explore potential SQL queries within the database's "world model". This approach dynamically generates promising SQL construction actions based on partial query states, enabling zero-shot Text-to-SQL interactions without task-specific fine-tuning. Unlike MuZero, which focuses on planning for decision-making in uncertain environments, Alpha-SQL applies MCTS in a specific task—guiding SQL query construction through self-generated actions within a complex database context.

For exploration strategies, Nagabandi et al. [363] incentivizes agents to explore unknown regions by providing reward mechanisms (exploration bonuses) for discovering new states. Dreamer [350] propose that world models can generate imaginary action sequences (imaginary rollouts), allowing agents to safely evaluate the benefits of new actions in simulated environments without risking real-world experimentation. Similarly, in the discrete world model Hafner et al. [364], agents efficiently explore complex environments by simulating multiple possible future states, effectively balancing the trade-off between exploration and exploitation.

For example, in reinforcement learning, agents can employ a learned world model to simulate future trajectories in action-selection tasks. The world model evaluates the potential rewards of different actions, enabling the agent to plan effectively and take actions that maximize long-term goals.

4.4.4 Cross-Module Integration

While memory, perception, and action are discussed as separate modules, the true strength of world models lies in their ability to seamlessly integrate across these domains. A world model continuously receives sensory input, updates its internal memory, simulates future states, and uses this information to drive action selection. The iterative feedback loop between these modules allows agents to engage in intelligent, goal-directed behavior that is highly adaptive to changes in the environment.

This cross-module interaction is particularly relevant in complex, dynamic systems such as robotics, where an agent must continuously adapt its internal representation of the world, process sensory input, store relevant experiences, and take actions in real time. In the context of embodied agents, the integration of these modules ensures that predictions made by the world model are grounded in current observations and the agent's ongoing experiences.

World models provide a fundamental unifying principle across modalities. Whether predicting physical outcomes in embodied robotics, anticipating visual changes on screens, or inferring semantic relationships in text, the core mechanism remains consistent: generating predictions about how states evolve under different actions. This cross-modal capacity explains why humans transition effortlessly between manipulating objects, navigating interfaces, and processing language—all activities driven by the same underlying predictive architecture. Future AI systems may achieve similar integration by developing world models that bridge these traditionally separate domains through a common predictive framework.

In summary, the relationship between the world model and the other modules—memory, perception, and action—forms the backbone of intelligent behavior in AI systems. Each module contributes to a cycle of prediction, update, and action, allowing agents to function effectively in dynamic and uncertain environments. These interactions highlight the need for a holistic approach when designing agent architectures, where world models are closely intertwined with sensory input, memory systems, and decision-making processes.

4.5 Summary and Discussion

The evolution of AI world models, from early cognitive insights to advanced AI architectures, underscores the growing realization that true intelligence relies on the ability to predict, simulate, and imagine. Unlike classical reinforcement learning, where agents operate solely through trial-and-error interactions, world models enable foresight—agents can plan, anticipate, and adapt to changes before they happen. This leap in cognitive modeling—whether implicit, explicit, or simulator-based—marks a significant shift in how machines can be endowed with flexibility, robustness, and generalization across tasks.

An essential yet often overlooked aspect of world models is their operation across multiple temporal and spatial scales. Human mental models seamlessly integrate predictions spanning milliseconds (reflexive responses), seconds (immediate action planning), minutes to hours (task completion), and even years (life planning) [366]. This multi-scale capability allows us to simultaneously predict immediate physical dynamics while maintaining coherent long-term narratives and goals. Similarly, humans process spatial information across scales—from fine-grained object manipulation to navigation across environments to abstract geographical reasoning. Current AI world models typically excel within narrow temporal and spatial bands, whereas human cognition demonstrates remarkable flexibility in scaling predictions up and down as context demands. This suggests that truly general-purpose AI world models may require explicit mechanisms for integrating predictions across multiple time horizons and spatial resolutions, dynamically adjusting the granularity of simulation based on task requirements.

One central challenge in designing world models is the interplay between **complexity** and **predictive accuracy**. As discussed, implicit models, such as those based on recurrent neural networks or transformers, offer simplicity and elegance, but they often come with the trade-off of limited interpretability. The model's internal state is an opaque latent space, making it difficult to enforce domain constraints or provide guarantees about the accuracy of predictions. While such systems excel at capturing highly complex relationships and data-driven patterns, they also risk overfitting or failing to generalize to unseen scenarios.

Explicit models, by contrast, offer greater transparency and control. By factorizing state transitions and observations into separate functions, we gain a clearer understanding of how predictions are formed, and we can more easily integrate structured knowledge, such as physical laws or domain-specific rules. However, this approach comes with its own set of challenges. First, it often requires large amounts of labeled training data or simulated experiences to accurately capture environment dynamics. Second, even the most well-structured explicit models may struggle with complex environments that require fine-grained, high-dimensional state representations, such as in video prediction or robotics.

The **simulator-based** approach offers a promising alternative, wherein agents rely on external environments—either physically grounded or simulated—for dynamic updates. This method avoids many of the challenges inherent in learning accurate world models from scratch, as the simulator itself serves as the "oracle" of state transitions and observations. However, the reliance on simulators also introduces limitations: simulators often fail to capture the full richness of real-world dynamics and can be computationally expensive to maintain or scale. Furthermore, real-world environments introduce noise and variability that a purely learned or pre-configured model might miss. As AI agents strive to perform tasks in open-ended, unpredictable settings, the robustness of their world models will be tested by the gap between simulated and actual environments.

A key theme that emerges from this discussion is the **trade-off between generalization and specialization**. The more specific a world model is to a particular domain or task, the less likely it is to generalize across different contexts. Models like MuZero [349] and Dreamer [350] exemplify this: they excel at specific environments (e.g., Atari games or robotics) but require careful adaptation when transferred to new, uncharted domains. Conversely, implicit models—particularly those leveraging large-scale neural networks—have the potential to generalize across tasks but often do so at the cost of sacrificing domain-specific expertise.

Moreover, **integrating memory** with world models is crucial for agents that need to handle long-term dependencies and past experiences. While world models excel at predicting the next state based on immediate inputs, true intelligent behavior often requires reasoning about distant outcomes. Long-term memory allows agents to store critical environmental knowledge, ensuring that short-term predictions are grounded in a broader understanding of the world. This fusion of memory, perception, and action, mediated by the world model, creates a feedback loop where predictions shape actions, which in turn inform future predictions.

The **human analogy** remains compelling: just as humans integrate sensory inputs, memories, and internal models to navigate the world, so too must intelligent agents combine perception, memory, and action through their world models. As the field advances, it is clear that a holistic approach—one that unifies implicit, explicit, and simulator-based methods—may be the key to achieving more robust, generalizable, and adaptive agents. Hybrid methods, like those used in AutoManual [108] or discovery-based models [356], offer exciting possibilities for blending learned knowledge with structured rules and real-time interactions, potentially pushing the boundaries of what we consider a world model.

Looking forward, **open questions remain**. How can we ensure that world models exhibit **long-term stability** and **reliability** in real-world settings? How do we handle the inherent **uncertainty** in dynamic environments while maintaining the flexibility to adapt? Furthermore, as agents grow more sophisticated, how can we design systems that are both **efficient** and **scalable** across increasingly complex tasks without incurring massive computational costs?

In conclusion, the future of world models lies in their ability to balance the need for **generalization** with the requirement for **domain expertise**. By continuing to explore and refine the interplay between model simplicity and complexity, between external and internal approaches, we move closer to developing AI systems that not only understand the world but can actively shape their understanding to navigate and adapt in a rapidly changing reality.

Chapter 5

Reward

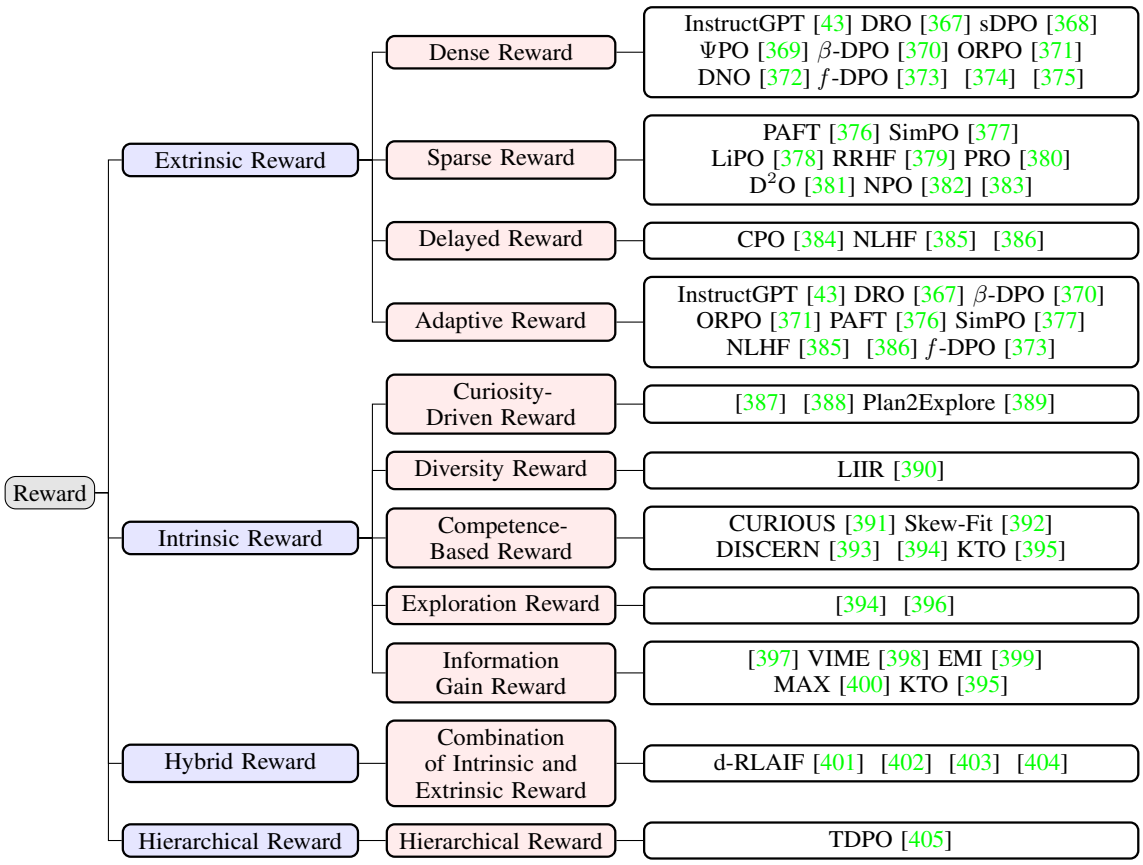


Figure 5.1: Illustrative Taxonomy of Reward system

Rewards help the agent distinguish between beneficial and detrimental actions, shaping its learning process and influencing its decision-making. This chapter first introduces common reward substances in the human body and the corresponding reward pathways. Then, the reward paradigm under the agent and the different methods involved are defined. In the discussion section, the influence relationship between other modules is described, and the existing methods are summarized, then the problems that need to be solved in the future and the optimization directions are discussed.

Table 5.1: The comparison of human common reward pathways.

Reward Pathway	Neurotransmitter	Mechanism
Mesolimbic pathway [406]	Dopamine	Dopaminergic neurons in the ventral tegmental area (VTA) extend projections to the nucleus accumbens, where they release dopamine to regulate reward-related signaling. Dopamine diffuses across the synaptic cleft and binds to dopamine receptors—primarily D1-like (excitatory via Gs proteins, increasing cAMP) and D2-like (inhibitory via Gi proteins, reducing cAMP)—thereby modulating reward, motivation, and reinforcement.
Mesocortical pathway [407]	Dopamine	Dopaminergic projections from the VTA reach the prefrontal cortex (PFC). Here, dopamine binds to its receptors to influence cognitive functions such as decision-making, working memory, and emotional regulation, all of which contribute to evaluating and anticipating rewards.
Nigrostriatal pathway [407]	Dopamine	Dopamine's action on D1 and D2 receptors in the striatum helps shape both motor routines and reward-related behaviors.
Locus coeruleus [408]	Norepinephrine	Neurons in the locus coeruleus release norepinephrine to widely distributed targets across the brain. At synapses, norepinephrine binds to adrenergic receptors(α and β subtypes), modulating neuronal excitability, arousal, attention, and stress responses. These modulatory effects can indirectly influence reward processing and decision-making circuits.
Glutamatergic projection [409]	Glutamate	Upon releasing into the synaptic cleft, glutamate binds to both ionotropic receptors (such as AMPA and NMDA receptors) and metabotropic receptors located on the postsynaptic neuron, thereby initiating excitatory signaling. This binding produces excitatory postsynaptic potentials and is crucial for synaptic plasticity and learning within reward circuits.
GABAergic modulation [410]	Gamma-Aminobutyric Acid (GABA)	GABA serves as the principal inhibitory neurotransmitter. At the synapse, GABA binds to GABAA receptors and GABAB receptors. This binding results in hyperpolarization of the postsynaptic cell, thereby providing inhibitory regulation that balances excitatory signals in the reward network.

5.1 The Human Reward Pathway

The brain's reward system is broadly organized into two major anatomical pathways. The first is the medial forebrain bundle, which originates in the basal forebrain and projects through the midbrain, ultimately terminating in brainstem regions. The second is the dorsal diencephalic conduction system, which arises from the rostral portion of the medial forebrain bundle, traverses the habenula, and projects toward midbrain structures [407]. The feedback mechanisms and substances in the human brain are complex, involving a variety of neurotransmitters, hormones, and other molecules, which regulate brain function, emotions, cognition, and behavior through feedback mechanisms such as neurotransmitter systems and reward circuits. Feedback mechanisms can be positive (such as feedback in the reward system) or negative (such as inhibiting excessive neural activity). Well-known feedback substances [411] include dopamine, neuropeptides, endorphins, glutamate, etc.

Dopamine is a signaling molecule that plays an important role in the brain, affecting our emotions, motivation, movement, and other aspects [412]. This neurotransmitter is critical for reward-based learning, but this function can be disrupted in many psychiatric conditions, such as mood disorders and addiction. The mesolimbic pathway [406], a key dopaminergic system, originates from dopamine-producing neurons in the ventral tegmental area (VTA) and projects to multiple limbic and cortical regions, including the striatum, prefrontal cortex, amygdala, and hippocampus. This pathway plays a central role in reward processing, motivation, and reinforcement learning, and is widely recognized as a core component of the brain's reward system. Neuropeptides are another important class of signaling molecules in the nervous system, involved in a variety of functions from mood regulation to metabolic control, and are slow-acting signaling molecules. Unlike neurotransmitters, which are limited to synapses, neuropeptide signals can affect a wider range of neural networks and provide broader physiological regulation. There is a significant cortical-subcortical gradient in the distribution of different neuropeptide receptors in the brain. In addition, neuropeptide signaling has been shown to significantly enhance the structure-function coupling of brain regions and exhibit a specialized gradient from

sensory-cognitive to reward-physical function [413]. Table 5 lists the common reward pathways in the human brain, the neurotransmitters they transmit, and the corresponding mechanisms of action, describing the basic framework of the human brain reward system.

5.2 From Human Rewards to Agent Rewards

Having examined the foundations of human reward pathways, we now turn to how artificial agents learn and optimize behavior through reward signals. While biological systems rely on complex neurochemical and psychological feedback loops, artificial agents operate using formalized reward functions designed to guide learning and decision-making. Though inspired by human cognition, agent reward mechanisms are structurally and functionally distinct. Understanding the analogies and disanalogies between these systems is crucial for aligning artificial behavior with human preferences.

In humans, rewards are deeply embedded in a rich web of emotional, social, and physiological contexts. They emerge through evolutionarily tuned mechanisms involving neurotransmitters like dopamine and are shaped by experiences, culture, and individual psychology. In contrast, artificial agents rely on mathematically defined reward functions that are externally specified and precisely quantified. These functions assign scalar or probabilistic feedback to actions or states, providing a signal for optimization algorithms such as reinforcement learning [3, 414].

One key distinction lies in the programmability and plasticity of agent rewards. Unlike human reward systems, which are constrained by biological architecture and evolutionary inertia, agent reward functions are fully customizable and can be rapidly redefined or adjusted based on task requirements. This flexibility enables targeted learning but also introduces design challenges—specifying a reward function that accurately captures nuanced human values is notoriously difficult.

Another important disanalogy concerns interpretability and generalization. Human rewards are often implicit and context-dependent, whereas agent rewards tend to be explicit and task-specific. Agents lack emotional intuition and instinctual drives; their learning depends entirely on the form and fidelity of the reward signal. While frameworks like reinforcement learning from human feedback (RLHF) attempt to bridge this gap by using preference data to shape agent behavior [12], such methods still struggle with capturing the full complexity of human goals, especially when preferences are intransitive, cyclical, or context-sensitive [321].

Moreover, attempts to borrow from human reward mechanisms—such as modeling intrinsic motivation or social approval—face limitations due to the absence of consciousness, embodiment, and subjective experience in artificial agents. Consequently, while human reward systems offer valuable inspiration, the design of agent reward functions must address fundamentally different constraints, including robustness to misspecification, adversarial manipulation, and misalignment with long-term human interests.

The following section will delve deeper into agent reward models, focusing on their design principles, evolution, and how these models selectively incorporate human-inspired insights to optimize artificial behavior within formal systems.

5.3 AI Reward Paradigms

Rewards also exist in intelligent agents, especially in reinforcement learning scenarios. Rewards are the core signal used to guide how intelligent agents act in the environment. They express feedback on the behavior of intelligent agents and are used to evaluate an action's quality in a certain state, thereby affecting the decision-making of subsequent actions. Through continuous trial and error and adjustment, intelligent agents learn to choose behavioral strategies that can obtain high rewards in different states.

5.3.1 Definitions and Overview

In reinforcement learning, the reward model dictates how an agent is provided with feedback according to the actions it performs within its environment. This model plays a crucial role in guiding the agent's behavior by quantifying the desirability of actions in a given state, thus influencing its decision-making.

Formal Definition. The agent's interaction with its environment can be framed within the formalism of a Markov Decision Process (MDP) [415], which is represented as:

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma), \tag{5.1}$$

where:

- $\mathcal{S}$ denotes the state space, encompassing all possible states in the environment.
- $\mathcal{A}$ denotes the action space, which encompasses all actions available to the agent at any given state.
- $P(s'|s,a)$ defines the state transition probability. It represents the likelihood of transitioning to state s' after the agent takes action a in state s.
- $r(s,a)$ specifies the reward function, which assigns an immediate scalar reward received by the agent for executing action a in state s.
- $\gamma \in [0,1]$ is the discount factor, which controlls the agent's preference for immediate versus future rewards by weighting the contribution of future rewards to the overall return.

The reward function $r(s,a)$ serves as a fundamental component in the formulation of the Agent Reward Model. It is mathematically represented as:

$$r(s,a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R} \tag{5.2}$$

This function returns a scalar reward based on the agent's current state s and the action a it selects. The scalar value $r(s,a)$ is a feedback signal that indicates the immediate benefit (or cost) of the chosen action in the given state. This reward signal guides the agent's learning process, as it helps evaluate the quality of actions taken within specific contexts.

Objective of the Agent Reward Model. The agent's primary objective is to maximize its overall cumulative reward over time. This is typically achieved by selecting actions that yield higher long-term rewards, which are captured in the form of the return G_t at time step t, defined as the sum of future discounted rewards:

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}, \tag{5.3}$$

where r_{t+k} denotes the reward received at time step $t+k$, and γ^k is the discount factor applied to rewards received at time step $t+k$. The agent aims to optimize its policy by maximizing the expected return over time.

At a higher level, the reward model can be classified into three categories based on the origin of the feedback signal: i) extrinsic reward, ii) intrinsic reward, iii) hybrid reward and iv) hierarchical model. Each of these categories can be further subdivided into smaller subclasses. Figure 5.2 illustrates different types of rewards. Next, we will explore these different types of reward in more detail, outlining the distinct features and applications of each type.

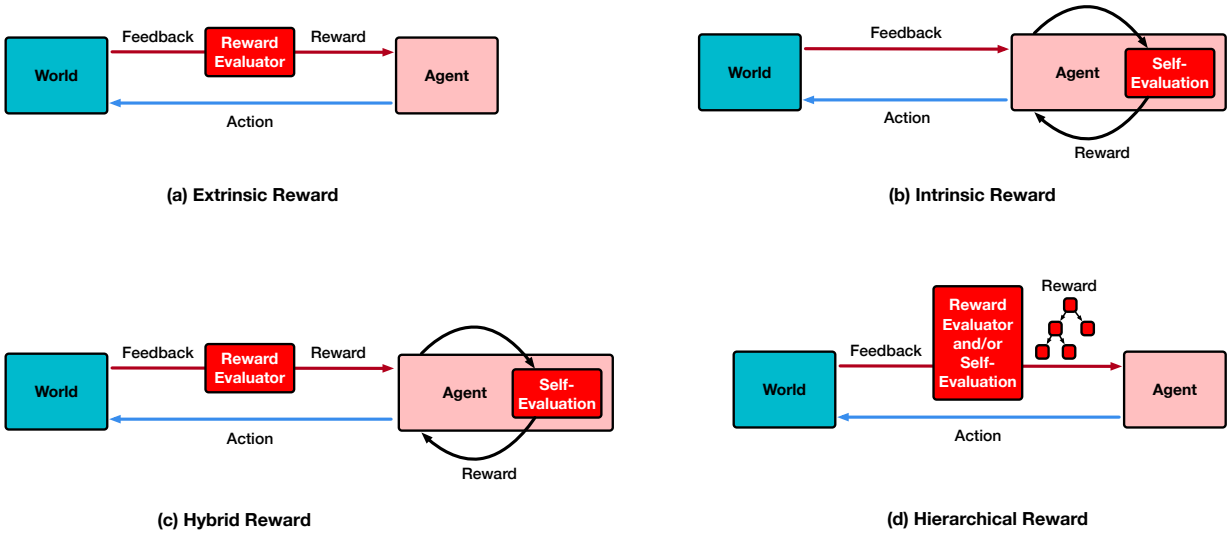


Figure 5.2: Illustration of different types of reward.

5.3.2 Extrinsic Rewards

Extrinsic rewards are externally defined signals that guide an agent's behavior toward specific goals. In artificial learning systems, especially reinforcement learning, these signals serve as a proxy for success that shape the policy through measurable outcomes. However, the structure and delivery of these rewards significantly influence the learning dynamics, which present different trade-offs depending on how feedback is distributed.

Dense Reward. Dense reward signals provide high-frequency feedback, typically at every step or after each action. This frequent guidance accelerates learning by allowing agents to immediately associate actions with outcomes. However, dense feedback can sometimes incentivize short-sighted behavior or overfit to easily measurable proxies rather than deeper alignment.

For example, InstructGPT [43] uses human rankings of model outputs to provide continuous preference signals throughout fine-tuning, enabling efficient behavior shaping. Similarly, Cringe Loss [416] and its extensions [374] transform pairwise human preferences into dense training objectives, offering immediate signal at each comparison. Direct Reward Optimization (DRO) [367] further simplifies this paradigm by avoiding pairwise comparisons entirely, associating each response with a scalar score—making the reward signal more scalable and cost-effective. These methods exemplify how dense feedback facilitates fine-grained optimization but must be carefully designed to avoid superficial alignment.

Sparse Reward. Sparse rewards are infrequent and typically only triggered by major milestones or task completions. While they often reflect more meaningful or holistic success criteria, their delayed nature can make credit assignment more difficult, especially in complex environments.

PAFT [376] exemplifies this challenge by decoupling supervised learning and preference alignment, with feedback applied only at select decision points. This sparsity reflects a more global notion of success but increases the burden on optimization. Similarly, SimPO [377] uses log-probability-based implicit rewards without dense comparisons. The sparsity simplifies the training pipeline but can limit responsiveness to subtle preference shifts. Sparse reward systems thus tend to be more robust but demand stronger modeling assumptions or more strategic exploration.

Delayed Reward. Delayed rewards defer feedback until after a sequence of actions, requiring agents to reason about long-term consequences. This setup is essential for tasks where intermediate steps may be misleading or only make sense in retrospect. The challenge lies in attributing outcomes to earlier decisions, which complicates learning but encourages planning and abstraction.

Contrastive Preference Optimization (CPO) [384] trains models by comparing sets of translations rather than evaluating each one in isolation. The reward signal arises only after generating multiple candidates, reinforcing patterns across iterations. Nash Learning from Human Feedback [385] similarly delays feedback until the model identifies stable strategies through competitive comparisons. These methods leverage delayed rewards to push beyond surface-level optimization, aligning more with long-term goals at the cost of slower convergence and more complex training dynamics.

Adaptive Reward. Adaptive rewards evolve dynamically in response to the agent's behavior or learning progress. By modulating the reward function such as increasing task difficulty or shifting reward targets, this approach supports continual improvement, especially in non-stationary or ambiguous environments. However, it introduces additional complexity in reward design and evaluation.

Self-Play Preference Optimization (SPO) [386] adapts rewards based on self-play outcomes, using social choice theory to aggregate preferences and guide learning. This approach allows the system to refine itself by evolving internal standards. f-DPO [373] builds on this idea by introducing divergence constraints that adapt the reward landscape during training. By tuning alignment-diversity trade-offs dynamically, these methods enable robust preference modeling under uncertainty, though they require careful calibration to avoid instability or unintended bias.

5.3.3 Intrinsic Rewards

Intrinsic rewards serve as internally generated signals that motivate agents to explore, learn, and improve, independent of external task-specific outcomes. These rewards are often structured to promote generalization, adaptability, and self-directed skill acquisition—qualities critical for long-term performance in complex or sparse-reward environments. Different intrinsic reward paradigms focus on fostering distinct behavioral tendencies within agents.

Curiosity-Driven Reward. This reward encourages agents to reduce uncertainty by seeking novel or surprising experiences. The key concept is to incentivize the agent to explore novel states where prediction errors are significant. This paradigm excels in sparse-reward settings by promoting information acquisition when external guidance is limited. For example, Pathak et al. [387] leverage an inverse dynamics model to predict the outcome of actions, creating a feedback loop that rewards novelty. Plan2Explore [389] extends this further by incorporating forward planning to

actively target areas of high epistemic uncertainty, thereby enabling faster adaptation to unseen environments. While effective at discovery, curiosity-driven methods can be sensitive to noise or deceptive novelty without safeguards.

Diversity Reward. Diversity reward shifts focus from novelty to behavioral heterogeneity, encouraging agents to explore a wide range of strategies rather than converging prematurely on suboptimal solutions. This approach is particularly useful in multi-agent or multimodal settings, where strategic variety enhances robustness and collective performance. LIIR [390] exemplifies this by assigning personalized intrinsic signals to different agents, driving them toward distinct roles while maintaining shared objectives. Diversity-driven exploration fosters broader policy coverage but may require careful balancing to avoid destabilizing coordination or goal pursuit.

Competence-Based Reward. Competence-based reward aims to foster learning progress by rewarding improvements in the agent's task proficiency. This reward adapts dynamically as the agent grows more capable, which creates a self-curriculum that supports continual skill acquisition. Skew-Fit [392] facilitates this through entropy-based goal sampling, encouraging agents to reach diverse states while maintaining challenge. CURIOUS [391] further automates curriculum generation by selecting goals that maximize learning progress over time. Competence-based methods are well-suited for open-ended environments, though they often require sophisticated estimation of progress and goal difficulty.

Exploration Reward. Exploration reward directly incentivizes the agent to engage with under-explored states or actions, which emphasize breadth over depth in environment interaction. Unlike curiosity, which focuses on unpredictability, exploration reward often targets coverage or novelty relative to the agent's visitation history. RND [394] exemplifies this by rewarding the prediction error of a randomly initialized network, pushing the agent toward unfamiliar states. This approach helps prevent premature convergence and encourages robustness, though it may lack focus if not paired with meaningful learning objectives.

Information Gain Reward. Information gain reward formalizes exploration as a process of uncertainty reduction, which guides agents to take actions that yield the highest expected learning. This reward is grounded in information theory and is especially powerful in model-based or reasoning-intensive tasks. CoT-Info [397] applies this to language models by quantifying knowledge gain at each reasoning step, optimizing sub-task decomposition. VIME [398] similarly employs Bayesian inference to reward belief updates about environmental dynamics. By explicitly targeting informational value, these methods offer principled exploration strategies, though they often incur high computational cost and require accurate uncertainty modeling.

5.3.4 Hybrid Rewards

Hybrid reward frameworks integrate multiple sources of feedback, most commonly intrinsic and extrinsic rewards, to enable more balanced and adaptive learning. By combining the exploratory drive of intrinsic rewards with the goal-directed structure of extrinsic rewards, these systems aim to improve both sample efficiency and generalization. This paradigm is especially beneficial in complex environments or open-ended tasks, where pure reliance on either feedback type may be insufficient.

A core advantage of hybrid rewards is their capacity to resolve the exploration-exploitation trade-off dynamically. For instance, Xiong et al. [403] combine intrinsic exploration with extrinsic human feedback within the context of RLHF. Using a reverse-KL regularized contextual bandit framework, they facilitate strategic exploration while aligning the agent's actions with human preferences. The method integrates intrinsic and extrinsic rewards through an iterative DPO algorithm and multi-step rejection sampling, optimizing exploration and alignment without compromising efficiency.

5.3.5 Hierarchical Rewards

Hierarchical reward architectures decompose complex objectives into layered subgoals, each associated with distinct reward signals. This structure mirrors the hierarchical organization of many real-world tasks, allowing agents to coordinate short-term decisions with long-term planning. By assigning lower-level rewards to immediate actions and higher-level rewards to abstract goals, agents can learn compositional behaviors that scale more effectively to complex environments.

In language modeling, Token-level Direct Preference Optimization (TDPO) [405] illustrates this principle by aligning LLMs through fine-grained token-level rewards derived from preference modeling. Using forward KL divergence and the Bradley-Terry model, TDPO simultaneously refines local choices and global coherence, improving alignment with nuanced human preferences. The hierarchical reward process here is not merely a structural design but a functional one: reinforcing both micro-decisions and macro-outcomes in a coordinated fashion.

More generally, hierarchical rewards can serve as scaffolding for curriculum learning, where agents progressively learn from simpler subtasks before tackling the overarching objective. In LLM agents, this might mean structuring rewards for subcomponents like tool-use, reasoning chains, or interaction flows, each of which contributes to broader task success.

5.4 Summary and Discussion

5.4.1 Interaction with Other Modules

In intelligent systems, reward signals function not only as outcome-driven feedback but as central regulators that interface with core cognitive modules such as perception, emotion, and memory. In the context of LLM-based agents, these interactions become particularly salient, as modules like attention, generation style, and retrieval memory can be directly influenced through reward shaping, preference modeling, or fine-tuning objectives.

Perception. In LLM agents, perception is often realized through attention mechanisms that prioritize certain tokens, inputs, or modalities. Reward signals can modulate these attention weights implicitly during training, reinforcing patterns that correlate with positive outcomes. For example, during reinforcement fine-tuning, reward models may upweight specific linguistic features—such as informativeness, factuality, or politeness—causing the model to attend more to tokens that align with these traits. This parallels how biological perception prioritizes salient stimuli via reward-linked attentional modulation [417]. Over time, the agent internalizes a perception policy: not merely "what is said," but "what is worth paying attention to" in task-specific contexts.

Emotion. Though LLMs do not possess emotions in the biological sense, reward signals can guide the emergence of emotion-like expressions and regulate dialogue style. In human alignment settings, models are often rewarded for generating responses that are empathetic, polite, or cooperative—leading to stylistic patterns that simulate emotional sensitivity. Positive feedback may reinforce a friendly or supportive tone, while negative feedback suppresses dismissive or incoherent behavior. This process mirrors affect-driven behavior regulation in humans [418], and allows agents to adapt their interaction style based on user expectations, affective context, or application domain. In multi-turn settings, reward-modulated style persistence can give rise to coherent personas or conversational moods.

Memory. Memory in LLM agents spans short-term context (e.g., chat history) and long-term memory modules such as retrieval-augmented generation (RAG) or episodic memory buffers. Reward signals shape how knowledge is encoded, reused, or discarded. For instance, fine-tuning on preference-labeled data can reinforce certain reasoning paths or factual patterns, effectively consolidating them into the model's internal knowledge representation. Moreover, mechanisms like experience replay or self-reflection—where agents evaluate past outputs with learned reward estimators—enable selective memory reinforcement, akin to dopamine-driven memory consolidation in biological systems [419]. This allows LLM agents to generalize from prior successful strategies and avoid repeating costly errors.

In general, reward in LLM-based agents is not a passive scalar signal but an active agent of behavioral shaping. It modulates attention to promote salient features, guides stylistic and affective expression to align with human preferences, and structures memory to prioritize useful knowledge. As agents evolve toward greater autonomy and interactivity, understanding these cross-module reward interactions will be essential for building systems that are not only intelligent, but also interpretable, controllable, and aligned with human values.

5.4.2 Challenges and Directions

Although extensive research has been conducted on various reward mechanisms, several persistent challenges remain. One fundamental issue is reward sparsity and delay. In many real-world scenarios, reward signals are often infrequent and delayed, making it difficult for an agent to accurately attribute credit to specific actions. This, in turn, increases the complexity of exploration and slows down the learning process.

Another significant challenge is the potential for reward hacking. Agents, in their pursuit of maximizing rewards, sometimes exploit unintended loopholes in the reward function. This can lead to behaviors that diverge from the intended design goals, particularly in complex environments where optimization objectives may not always align with the true task requirements.

Moreover, the process of reward shaping presents a delicate balance. While shaping rewards can accelerate learning by guiding an agent toward desired behaviors, excessive or poorly designed shaping may lead to local optima, trapping the agent in suboptimal behaviors. In some cases, it may even alter the fundamental structure of the original task, making it difficult for the agent to generalize to other scenarios.

Many real-world problems are inherently multi-objective in nature, requiring agents to balance competing goals. Under a single reward function framework, finding the right trade-offs between these objectives remains an open problem. Ideally, a hierarchical reward mechanism could be designed to guide learning in a structured, step-by-step manner. However, constructing such mechanisms effectively is still a challenge.

Finally, reward misspecification introduces further uncertainty and limits generalization. Often, a reward function does not fully capture the true task goal, leading to misalignment between the agent's learning objective and real-world success. Additionally, many reward functions are tailored to specific environments and fail to generalize when conditions change or tasks shift, highlighting the need for more robust reward models.

Addressing these challenges requires novel approaches. One promising direction is to derive implicit rewards from standard examples or outcome-based evaluations, which can help mitigate reward sparsity issues. Additionally, decomposing complex tasks into hierarchical structures and designing rewards from the bottom up can offer a more systematic approach, even in multi-objective settings. Furthermore, leveraging techniques such as meta-learning and meta-reinforcement learning can enhance the adaptability of reward models, allowing agents to transfer knowledge across tasks and perform effectively in diverse environments. By exploring these avenues, we can move toward more reliable and scalable reward mechanisms that better align with real-world objectives.

Chapter 6

Emotion Modeling

Emotions are a key part of how humans think, make decisions, and interact with others. They guide us to understand situations, make choices, and build relationships. Antonio Damasio, in his book *Descartes' Error* [25], explained that emotions are not separate from logic. Instead, they are deeply connected to how we reason and act. When developing LLM agents, adding emotional capabilities can potentially make these systems smarter, more adaptable, and better understand the world around them.

For LLM agents, emotions can act as a decision-making tool, much like they do for humans. Emotions help us prioritize tasks, understand risks, and adapt to new challenges. Marvin Minsky, in *The Emotion Machine* [420], described emotions as a way to adjust our thinking processes, helping us solve problems in a more flexible and creative manner. Similarly, LLM agents with emotion-like features could improve their ability of solving complex problems and making decisions in a more human-style.

However, the integration of emotions into LLM agents is still in its early stages. Researchers are just starting to explore how emotional capabilities can improve these systems. Furthermore, there is great potential for LLM agents to support human emotional well-being, whether through empathetic conversations, mental health support, or simply building better connections with users. This promising but challenging area requires collaboration between fields such as psychology, cognitive science, and AI ethics. As research advances, emotion-understanding LLM agents could redefine how we interact with technology, creating deeper trust and more meaningful relationships between humans and machines.

In the following subsections, we will delve deeper into the role of emotions in shaping LLM agents. We will explore how emotions can be used to enhance learning and adaptability, how LLMs understand human emotions, and how these systems express and model their own emotional states. We will also examine how emotions can be manipulated to influence LLM agents' behavior and personalities, as well as the ethical and safety concerns that arise from these capabilities. Each of these discussions builds on the foundational importance of emotion to create LLM agents that are more intelligent, empathetic, and aligned with human values.

6.1 Psychological Foundations of Emotion

Psychological and neuroscientific theories of emotion provide essential frameworks for developing emotionally intelligent LLM agents. These theories can be categorized into several major approaches, each offering unique perspectives on how emotions function and how they might be implemented in AI systems.

Categorical Theories. These models posit that emotions exist as discrete, universal categories with distinct physiological and behavioral signatures. Ekman's theory of basic emotions [421] identifies six fundamental emotions (anger, disgust, fear, happiness, sadness, and surprise) that are recognized across cultures and expressed through specific facial configurations. This discrete approach has significantly influenced affective computing, with many emotion classification systems in AI adopting these labels for training [422, 423]. For LLM agents, categorical frameworks provide clear taxonomies for classifying user emotions and generating appropriate responses. However, they face criticism for oversimplifying the complex, blended nature of human emotional experience [424] and may not capture cultural variations in emotional expression [425].

Dimensional Models. Rather than discrete categories, dimensional approaches represent emotions as points in a continuous space defined by fundamental dimensions. Russell's Circumplex Model [426] maps emotions onto two primary dimensions: valence (pleasure-displeasure) and arousal (activation-deactivation). This framework enables more nuanced tracking of emotional states. It distinguishes between high-arousal panic and low-arousal anxiety despite both having negative valence. The PAD (Pleasure-Arousal-Dominance) model [427] extends this by adding a dominance dimension, capturing the sense of control or power associated with emotional states. These continuous representations have proven valuable for LLM systems that need to generate emotionally graded responses or track subtle shifts in user affect over time [428, 429, 430]. Dimensional models allow for fine-grained control over generated content, enabling humans or agents to modulate tone along continuous scales rather than switching between discrete emotional states.

Hybrid and Componential Frameworks. Recognizing limitations in pure categorical or dimensional approaches, several theories integrate aspects of both. Plutchik's Wheel of Emotions [431] arranges eight primary emotions in a wheel structure with intensity gradients and dimensional properties, allowing for the representation of complex emotional blends (e.g., love as a mixture of joy and trust). Meanwhile, componential models like Scherer's Component Process Model (CPM) [432] conceptualize emotions as emerging from synchronized components including cognitive appraisal, physiological arousal, action tendencies, and subjective feelings. Particularly influential in AI research is the OCC (Ortony-Clore-Collins) model [433], which defines 22 emotion types based on how events, agents, or objects are evaluated relative to goals and standards. These appraisal-based frameworks have been implemented in dialogue systems that generate emotional responses through rule-based evaluation of situations [434, 435]. For LLM agents, such models provide computational structures for evaluating text input and selecting contextually appropriate emotional responses, improving both coherence and perceived empathy [436, 437].

Neurocognitive Perspectives. The neuroscience of emotion offers additional insights for LLM architectures. Damasio's somatic marker hypothesis [25] emphasizes how emotions, implemented through body-brain interactions, guide decision-making by associating physiological states with anticipated outcomes. This interaction between the limbic system and the cortex shows a two-process architecture: fast "alarm" signals in the limbic system, like those processed by the amygdala, work alongside slower, more deliberate reasoning in the cortex. Contemporary LLM systems have begun implementing analogous architectures, where fast sentiment detection modules work in parallel with more thorough chain-of-thought reasoning [436, 437]. Recent evidence further suggests that opponent circuitry in the striatum enables distributional reinforcement learning by encoding not just mean rewards but entire probability distributions, offering a neural basis for emotion-influenced decision-making under uncertainty [438]. Similarly, LeDoux's distinction between "low road" (quick, automatic) and "high road" (slower, cognitive) fear processing [24] suggests design patterns for systems that need both immediate safety responses and nuanced emotional understanding. Minsky's framing of emotions as "ways to think" [420] that reorganize cognitive processes has influenced frameworks like EmotionPrompt [428] and Emotion-LLaMA [423], where emotional context dynamically reshapes LLM reasoning.

These theoretical frameworks increasingly inform the development of emotionally intelligent LLM agents. Categorical models provide clear labels for emotion classification tasks [423, 429], while dimensional embeddings enable continuous control over generated text [428]. Hybrid approaches help systems handle mixed emotions and emotional intensity. Appraisal-based methods, particularly those derived from the OCC model, allow LLMs to evaluate narrative events or user statements contextually, selecting appropriate emotional responses that foster rapport and trust [439]. Neuroscientifically-inspired dual-process architectures combine "fast" sentiment detection with "slow" deliberative reasoning, enabling both quick safety responses and deeper emotional understanding [436, 437]. While explicit neurocognitive mechanisms (like dedicated "amygdala-like" pathways) remain rare in current LLM pipelines, emerging research explores biologically-inspired modules to handle urgent emotional signals and maintain consistent emotional states across extended interactions [440, 441].

Emotion is a key part of human intelligence, and it will likely become one of the key components or design considerations of LLM agents. One key future direction is systematically translating these psychological and neuroscience theories into an LLM agent's internal processes. Techniques for translating might include using dimensional models (e.g., valence/arousal/dominance) as latent states that influence generation or adopting explicit rule-based appraisals (OCC) to label user messages and shape the agent's subsequent moves. Hybrid approaches offer a compelling balance: an LLM could first recognize a discrete category (e.g., "fear") but also gauge its intensity and control dimension for finer-grained conversation. Such emotion-infused architectures might yield more coherent "moods" over time, analogous to how humans sustain affective states rather than resetting at every turn. Explicit alignment with psychological theories also enhances interpretability: designers can debug or refine the agent's responses by comparing them to well-established emotion constructs, rather than dealing with opaque emergent behaviors.

A second direction is harnessing these theories to improve *affectionate or supportive interactions*, often referred to as emotional alignment. For example, circumplex or PAD-based tracking can help an LLM detect negative valence and high arousal in a user's text and respond soothingly (e.g., lowering arousal, offering empathetic reappraisals). In

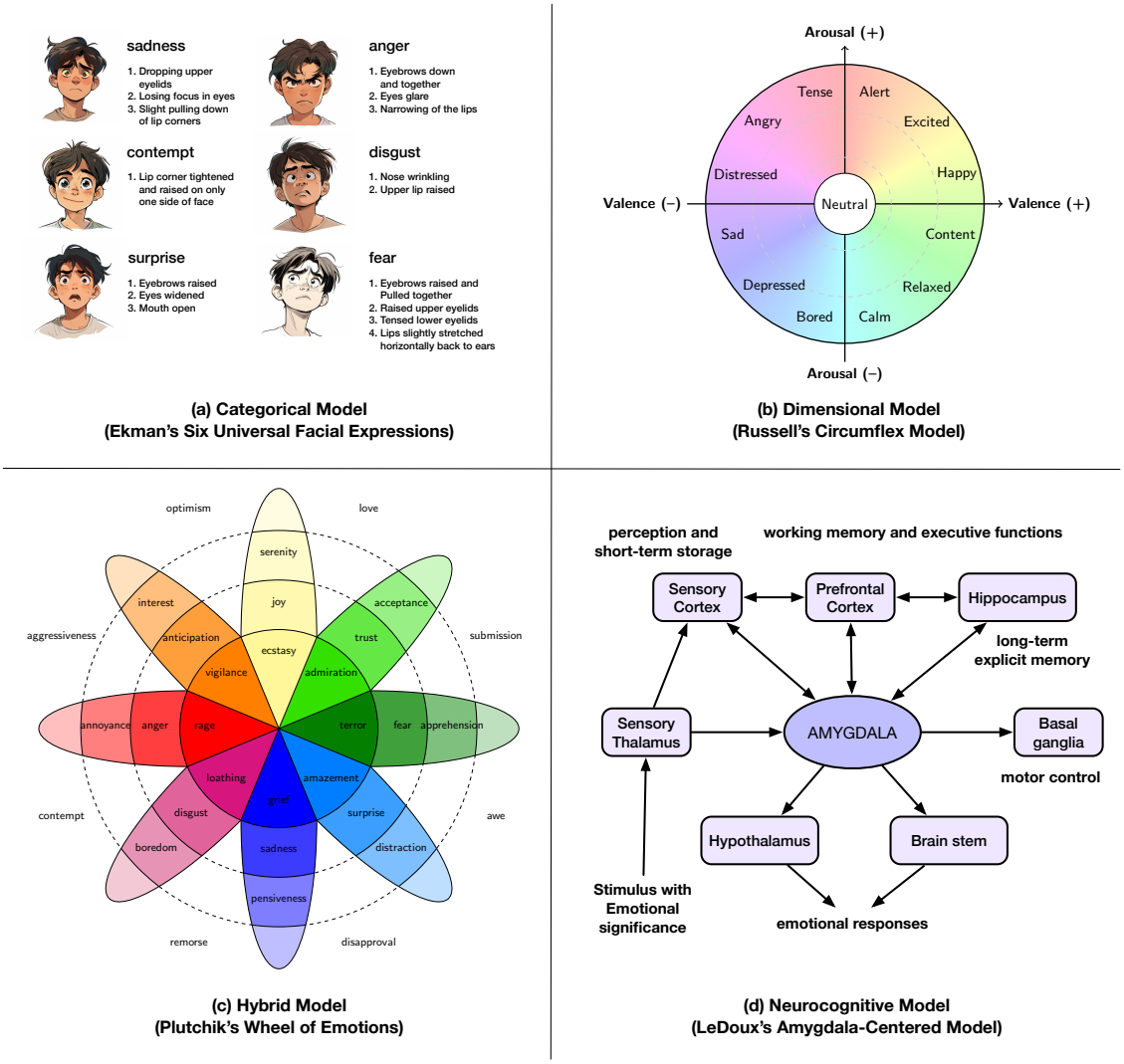


Figure 6.1: Visualization and examples of major emotion theory categories. (a) Categorical Theories: Ekman's six basic emotions [421] showing discrete emotional states. (b) Dimensional Models: Russell's Circumplex [426] representing emotions as coordinates in continuous space. (c) Hybrid/Componential Frameworks: Plutchik's Wheel [431] combining intensity gradients with categorical emotions. (d) Neurocognitive Perspectives: LeDoux's Amygdala-Centered Model [24] showing dual-pathway processing of emotional stimuli. These psychological foundations inform different approaches to emotion modeling in AI systems, from discrete classification to dimensional representations, appraisal-based reasoning, and multi-pathway information processing.

mental health or counseling scenarios, an appraisal-informed method could let the agent validate the user's feelings and understand their situation in terms of goal incongruence or perceived blame, which helps craft responses that convey genuine empathy. Grounding emotional outputs in cognitive theories (like "relief" if a negative outcome is avoided, or "gratitude" when a user helps the system) likewise makes interactions feel more natural and ethically aligned. These enhancements are particularly salient as LLMs migrate into real-world applications like customer service, elder care, and tutoring, where emotional sensitivity can improve outcomes and user well-being. By incorporating robust psychological and limbic-system insights, developers can design LLM agents that not only reason more effectively but also provide sincere emotional support, bridging the gap between computational precision and human-centric care.

6.2 Incorporating Emotions in AI Agents

The integration of emotional intelligence into large language models (LLMs) has emerged as a transformative approach to enhancing their performance and adaptability. Recent studies, such as those of EmotionPrompt [422], highlight how emotional stimuli embedded in prompts can significantly improve outcomes across various tasks, including a notable 10.9% improvement in generative task metrics such as truthfulness and responsibility. By influencing the attention mechanisms of LLMs, emotionally enriched prompts enrich representation layers and result in more nuanced outputs [422]. These advancements bridge AI with emotional intelligence, offering a foundation for training paradigms that better simulate human cognition and decision-making, particularly in contexts requiring social reasoning and empathy.

Multimodal approaches further elevate the impact of emotional integration. Models like Emotion-LLaMA [440] demonstrate how combining audio, visual, and textual data enables better recognition and reasoning of emotions. Using datasets such as MERR [440], these models align multimodal inputs into shared representations, facilitating improved emotional understanding and generation. This innovation extends beyond linguistic improvements, offering applications in human-computer interaction and adaptive learning. Together, these methods underscore the critical role of emotions in bridging technical robustness with human-centric AI development, paving the way for systems that are both intelligent and empathetic.

6.3 Understanding Human Emotions through AI

Textual Approaches. Recent work highlights the ability of LLMs to perform detailed reasoning about latent sentiment and emotion. Using step-by-step prompting strategies, such as chain of thought reasoning, researchers enable LLMs to infer sentiment even when explicit cues are absent [436]. Beyond single-turn inference, negotiation-based frameworks further refine emotional judgments by leveraging multiple LLMs that cross-evaluate each other's outputs, effectively mimicking a more deliberative human reasoning process [437]. These techniques underscore the importance of iterative, context-aware strategies to capture subtle emotional signals from purely textual input.

Multimodal Approaches. LLMs have also been extended to integrate signals from audio, video, and images. Recent efforts show how additional contextual or world knowledge can be fused with visual and textual information to capture deeper affective states [442]. Moreover, frameworks that convert speech signals into textual prompts demonstrate that vocal nuances can be embedded in LLM reasoning without changing the underlying model architecture [443]. This multimodal integration, combined with explainable approaches, allows for richer and more transparent representations of emotional content [444].

Specialized Frameworks. Beyond generic techniques, specialized systems address tasks in which emotion recognition requires higher levels of awareness of ambiguity [439], context sensitivity, and generative adaptability [445]. These approaches emphasize the inherent complexity of human emotion, treating it as dynamic and probabilistic rather than strictly categorical. Using flexible LLM instruction paradigms, they offer pathways to better interpret ambiguous emotional expressions and integrate contextual cues (e.g., dialogue history), moving LLM closer to human-like emotional comprehension.

Evaluation and Benchmarks. To holistically assess the emotional intelligence of LLM, researchers have proposed various benchmark suites. Some focus on generalized emotion recognition across different modalities and social contexts [446, 447], while others compare the performance and efficiency of models of varying sizes [448]. There are also specialized benchmarks that evaluate multilingual capabilities [449], annotation quality [450], or empathetic dialogue systems [451]. Furthermore, frameworks such as EMOBENCH [441] and MEMO-Bench [452] test nuanced emotional understanding and expression in both text and images, while MERBench [453] and wide-scale evaluations [454] address standardization concerns in multimodal emotion recognition. Together, these benchmarks reveal the growing, yet still imperfect grasp of human emotion by LLMs, highlighting ongoing challenges such as implicit sentiment detection, cultural adaptation, and context-dependent empathy [455].

6.4 Analyzing AI Emotions and Personality

Reliability of Personality Scales for LLMs. Large language models (LLMs) show conflicting evidence when evaluated through human-centered personality tests. On one hand, some studies challenge the validity of common metrics, reporting biases such as "agree bias" and inconsistent factor structures, raising doubts about whether these instruments capture genuine traits [456, 457]. On the other hand, systematic experiments reveal that LLMs can exhibit stable, human-like trait patterns and even adapt to different personas under specific prompts [458, 459]. Yet, concerns persist

about action consistency, alignment of self-knowledge, and whether role-playing agents truly maintain fidelity to their assigned characters [460, 461].

Psychometric Methods & Cognitive Modeling Approaches. Recent work applies rigorous psychometric testing, cognitive tasks, and population-based analyses to uncover how LLM processes and represents mental constructs [462, 463, 464]. Fine-tuning on human behavioral data can align models with decision patterns that mirror individual-level cognition, while population-based sampling techniques expose variability in neural responses [465, 466]. By merging psychological theories with advanced prompting and embedding methods, researchers illuminate latent representations of constructs like anxiety or risk-taking, showing how LLMs can approximate human reasoning across tasks.

Emotion Modeling. Studies on LLM-based emotional intelligence reveal notable abilities to interpret nuanced affect and predict emotion-laden outcomes, often surpassing average human baselines in standard tests [423, 429]. However, these models do not necessarily emulate human-like emotional processes; they rely on high-dimensional pattern matching that sometimes fails under changing contexts, negative input, or conflicting cues [467, 468]. However, hierarchical emotion structures, coping strategies, and empathy-like behaviors can emerge in larger-scale models, underscoring both the promise of emotional alignment and the ethical challenges in creating AI systems that appear and occasionally function as affective agents.

6.5 Manipulating AI Emotional Responses

Prompt-based Methods. Recent research shows that adopting specific personas or roles through well-engineered prompts can bias LLM cognition, allowing targeted emotional or personality outcomes [469, 470, 471, 472]. By inserting instructions such as "If you were a [persona]", LLMs adapt not only their thematic style, but also their underlying emotional stance. This approach is powerful for real-time manipulation, though it can be inconsistent across tasks and model variants, highlighting the need for more systematic methods.

Training-based Methods. Fine-tuning and parameter-efficient strategies offer deeper, more stable ways to induce or alter LLM emotions [473, 428, 474]. Quantized Low-Rank Adaptation (QLoRA) and specialized datasets can embed nuanced traits such as the Big Five or MBTI profiles directly into the model's learned weights. These methods enable LLMs to spontaneously exhibit trait-specific behaviors (including emoji use) and sustain their emotional states over longer dialogues, while also offering interpretability through neuron-level activation patterns.

Neuron-based Methods. A recent advance isolates personality-specific neurons and manipulates them directly to evoke or suppress emotional traits [475]. By toggling neuron activations pinpointed through psychologically grounded benchmarks (e.g., PersonalityBench), LLMs can embody targeted emotional dimensions without retraining the entire network. This neuron-centric approach provides fine-grained, dynamic control over model behaviors, representing a leap in precision and efficiency for emotional manipulation in LLMs.

6.6 Summary and Discussion

Manipulation and Privacy Concerns. The rapid adoption of Emotional AI in advertising and politics raises significant manipulation and privacy risks [476, 477]. Emotional AI often collects sensitive biometric data, such as facial expressions and voice tones, to infer emotional states, enabling targeted advertising or political influence. However, these systems can exploit human emotions for profit or political gain, infringing on fundamental rights and fostering over-surveillance in public spaces [478, 477]. Regulatory frameworks like GDPR and the EU AI Act are critical to mitigating these risks responsibly.

Alignment Issues. Emotional AI's capacity to detect and interpret emotions is often misaligned with intended outcomes, leading to inaccuracies and biases. Anxiety-inducing prompts, for instance, have been shown to exacerbate biases in large language models (LLMs), affecting outputs in high-stakes domains such as healthcare and education [479, 480]. Misinterpretation of emotional cues by AI systems, as seen in workplace applications, can exacerbate discrimination and power imbalances [481]. Techniques like reinforcement learning from human feedback (RLHF) have proven effective in mitigating these issues but require further development to ensure robust alignment in diverse contexts [479, 423].

Ethical Implications. Trust and acceptance of AI systems are significantly influenced by their ability to exhibit empathy and maintain socially appropriate behavior [482, 483]. However, the commodification of emotions in workplace management and customer service has raised concerns about ethical labor practices and AI-human relationships [481]. Moreover, Emotional AI's reliance on anthropomorphic characteristics without sufficient empathy can undermine user trust [482]. Frameworks like SafeguardGPT, which incorporate psychotherapy techniques, demonstrate promising approaches to fostering trust and aligning AI behavior with societal norms [484]. Nonetheless, challenges remain in ensuring privacy, fairness, and cultural sensitivity [484, 483].

Distinguishing AI Emotional Mimicry from Human Experience. Despite advances in emotion modeling for LLM agents, a fundamental distinction remains: these systems do not actually "feel" emotions as humans do but only show human-emotion-like patterns via probabilistic modeling. While LLMs can convincingly simulate emotional responses, recognize emotional patterns, and generate affectional outputs, they lack the embodied, phenomenological experience that defines human emotions. This simulation-reality gap creates both technical and ethical challenges. Users frequently anthropomorphize AI systems that display emotion-like behaviors [482], potentially leading to misplaced trust or expectations. This distinction needs to be carefully thought in both research and deployment contexts, as the perceived emotional capabilities of LLMs influence human-AI relationships, ethical frameworks, and regulatory approaches. Future work should balance enhancing LLMs' emotional intelligence while maintaining transparency about their fundamental limitations as non-sentient systems.

Chapter 7

Perception

Perception is the foundational gateway through which both humans and intelligent agents acquire information, interpret their surroundings, and ultimately make informed decisions. For humans, perception is seamless and intuitive, effortlessly transforming sensory inputs into meaningful interpretations. In artificial intelligence, however, perception systems are meticulously engineered to emulate—and in some respects surpass—human sensory processing, profoundly influencing an agent's capacity for interaction, learning, and adaptation in complex environments.

In this chapter, we begin by exploring key differences in the nature and efficiency of perception between humans and AI agents. Next, we categorize agent perception based on different forms and representations of perceptual input. We then discuss ongoing challenges in the agent perception system and highlight promising directions for improvement, both at the modeling and system architecture levels. Finally, we illustrate how perception modules can be effectively tailored to different intelligent agent scenarios, offering practical guidance for optimizing their use and suggesting pivotal areas for future research.

7.1 Human versus AI Perception

Perception is fundamental to intelligence, serving as the interface through which both humans and artificial agents interact with the world. Although humans commonly think of perception in terms of the five classical senses—vision, hearing, taste, smell, and touch—modern neuroscience identifies a richer sensory landscape. Conservatively, humans are described as having around 10 senses; more comprehensive views list approximately 21, while some researchers propose up to 33 distinct sensory modalities [546, 547]. Beyond the familiar senses, humans possess sophisticated internal perceptions, such as vestibular (balance), proprioception (awareness of body position), thermoception (temperature), and nociception (pain), enabling nuanced interaction with their environment.

Human senses are finely tuned to specific physical signals: for example, human vision detects electromagnetic waves with wavelengths between approximately 380–780 nm, whereas hearing perceives sound frequencies from about 20 Hz to 20 kHz [548]. These sensory modalities allow humans to effortlessly engage in complex tasks like language communication, object recognition, social interaction, and spatial navigation. Additionally, humans naturally perceive continuous changes over time, seamlessly integrating motion perception and temporal awareness, abilities essential for coordinated movement and decision-making [549]. Animals in the natural world exhibit even more diverse perceptual capabilities. Birds and certain marine organisms, for instance, utilize magnetoreception to navigate using Earth's magnetic fields, while sharks and electric eels exploit electroreception to sense electrical signals emitted by other organisms—abilities humans do not possess [550].

In contrast to biological perception, artificial agents rely upon engineered sensors designed to transform environmental stimuli into digital signals that algorithms can interpret. Common sensor modalities for AI agents include visual sensors (cameras), auditory sensors (microphones), tactile sensors, and inertial measurement units. AI agents typically excel at processing visual, auditory, and textual data, leveraging advances in deep learning and signal processing. However, certain human sensory abilities—particularly taste and smell—remain challenging for machines to emulate accurately. For example, the advanced bio-inspired olfactory chip developed by researchers [551] currently distinguishes around 24 different odors, a capability significantly less sensitive than the human olfactory system, which discriminates among more than 4,000 distinct smells [552].

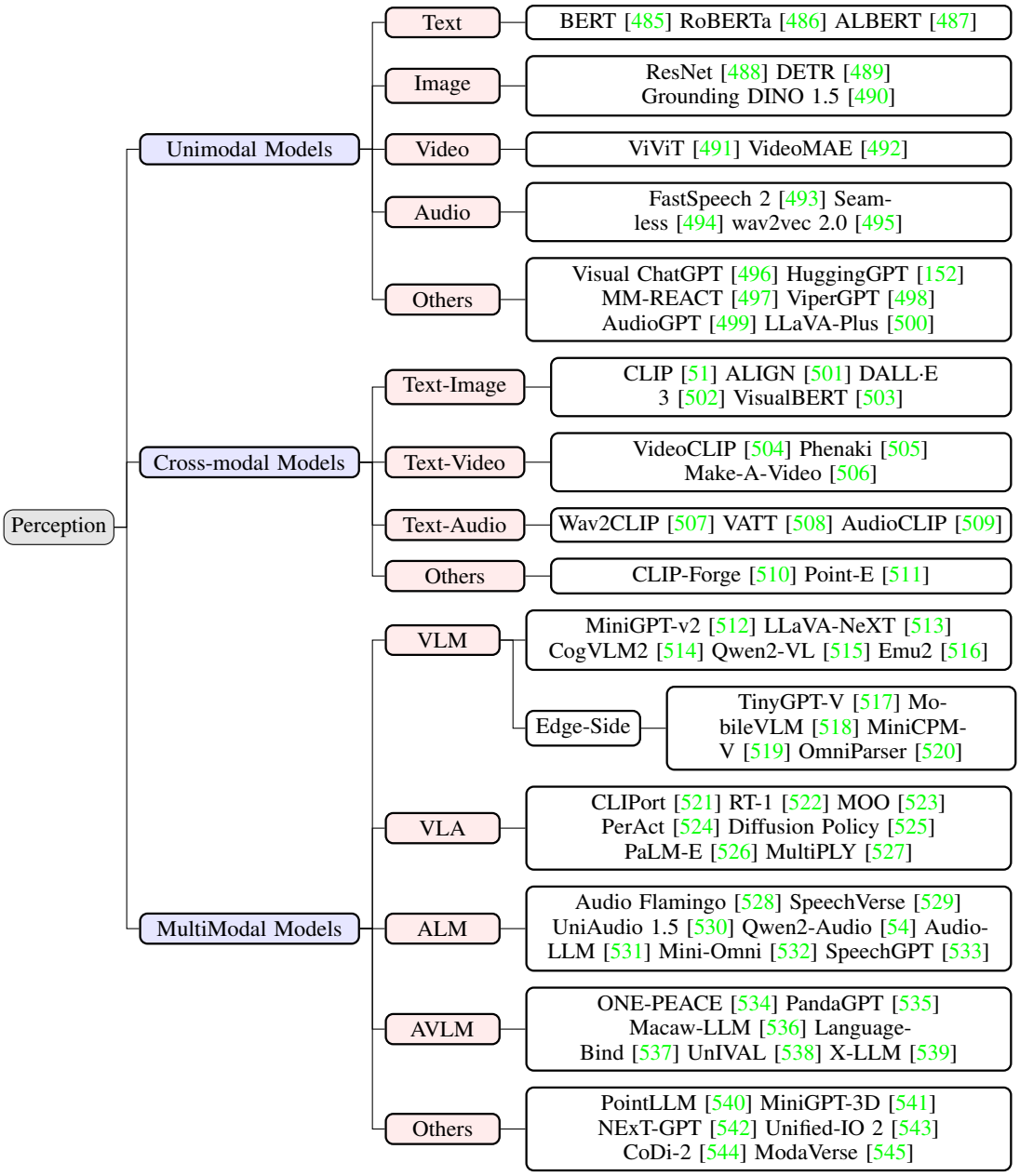


Figure 7.1: Illustrative Taxonomy of Perception System.

Another crucial distinction lies in perceptual processing efficiency. Human perception is limited by biological constraints such as nerve conduction speeds, typically in the range of milliseconds. Conversely, AI systems can process sensory inputs at speeds of microseconds or even nanoseconds, constrained primarily by computational hardware performance rather than biological limitations. Nevertheless, human perception naturally integrates information from multiple sensory modalities—known as multimodal perception—into coherent experiences effortlessly. For AI agents, achieving this multimodal integration requires carefully designed fusion algorithms that explicitly combine inputs from diverse sensors to build unified environmental representations [553].

Further differences arise in the way humans and artificial agents handle temporal and spatial information. Human perception is inherently continuous and fluid, smoothly experiencing the passage of time and spatial motion without explicit temporal discretization. In contrast, AI agents typically rely on discrete sampling of sensor data, using timestamps or sequential processing to simulate continuity. Spatial awareness in humans effortlessly merges visual, auditory, and vestibular information to achieve intuitive spatial positioning. For artificial agents, spatial perception usually involves

algorithmic processes such as simultaneous localization and mapping (SLAM) or 3D scene reconstruction from visual data sequences [554].

Physical or chemical stimuli transmitted from the external environment to human sensory organs will be received by the sensory system (such as eyes, ears, skin, etc.) and converted into neural signals, which are finally processed by the brain to produce perception of the environment. Similarly, to allow the intelligent agent to connect with the environment, it is also crucial to obtain these perception contents. Currently, various sensors are mainly used to convert electrical signals into processable digital signals. In this section, We distinguish between Unimodal models, Cross-modal models, and Multimodal models based on the number of modalities involved in the input and whether unified fusion modeling operations are performed. Unimodal Models specifically process and analyze data from a single modality or type of input (such as text, image, or audio), while Cross-modal Models establish relationships and enable translations between different modalities through dedicated mapping mechanisms, and Multimodal Models holistically integrate and process multiple modalities simultaneously to leverage complementary information for comprehensive understanding and decision-making.

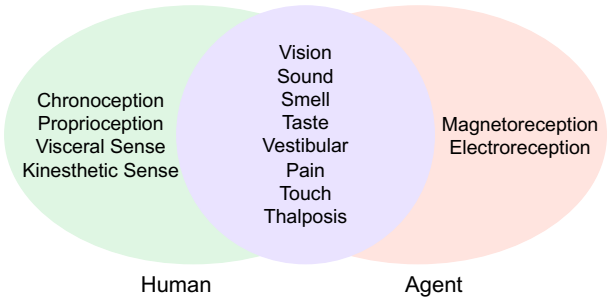


Figure 7.2: Comparison of common perceptual types between human and agent.

7.2 Types of Perception Representation

7.2.1 Unimodal Models

When humans are in an environment, they can listen to beautiful music, look at sunrise and sunset, or experience a wonderful audiovisual feast on stage. These perception contents can be either a single image or audio, or a fusion of multiple perception contents. Regarding the types of perception input of intelligent agents, we will start with single-modal and multimodal inputs, and introduce their implementation and differences.

Text As an important means of communication, text carries a wealth of information, thoughts, emotions and culture. Humans indirectly obtain the content of text through vision, hearing and touch, which is one of the most important ways for humans to interact with the environment. But for intelligent agents, text can directly serve as a bridge to connect with the environment, taking text as direct input and outputting response content. In addition to the literal meaning, text also contains rich semantic information and emotional color. In the early days, the bag-of-words model [555] was used to count text content and was widely used in text classification scenarios, but semantic expression could not be obtained. BERT [485] uses a bidirectional Transformer architecture for language modeling and captures the deep semantic information of text through large-scale unsupervised pre-training. [486, 487] further optimized the training efficiency of BERT. The autoregressive model represented by GPT3.5 [556] opened the prelude to LLM and further unified the tasks of text understanding and text generation, while technologies such as LoRA [109] greatly reduced the application cost of LLM and improved the agent's perception ability of complex real-world scenario tasks.

Image Image is another important way for humans to interact with the environment which inherently encode spatial information, encompassing crucial attributes such as morphological characteristics, spatial positioning, dimensional relationships, and kinematic properties of objects. The evolution of computer vision architectures has demonstrated significant advancement in processing these spatial attributes. The seminal ResNet architecture [488] established foundational principles for deep visual feature extraction, while subsequent YOLO series [557, 558] demonstrated the capability to simultaneously determine object localization and classification with remarkable efficiency. A paradigm shift occurred with the introduction of DETR [489], which revolutionized object detection by implementing parallel prediction through global context reasoning, effectively eliminating traditional computational overhead associated with non-maximum suppression and anchor point generation. More recently, DINO 1.5 [490] has extended these capabilities to open-set scenarios through architectural innovations, enhanced backbone networks, and expanded training paradigms,

substantially improving open-set detection performance and advancing the perceptual generalization capabilities of artificial agents in unconstrained environments.

Video Video is an expression of continuous image frames, which includes the time dimension and displays dynamic information that changes over time through continuous image frames. The intelligent agent uses video as input and obtains richer perceptual content through continuous frames. ViViT [491] extracts spatiotemporal markers from videos, effectively decomposing the spatial and temporal dimensions of the input. VideoMAE [492] learns general video feature representations through self-supervised pre-training and has strong generalization capabilities on out-of-domain data. It lays a solid foundation for intelligent agents to acquire perceptual capabilities in new scenarios.

Audio In addition to text and vision, another important way for humans to interact with the environment is through audio. Audio not only contains direct text content, but also contains the speaker's tone and emotion [559]. Wav2Vec2 [495] defines the contrast task by quantizing the potential representation of joint learning, achieving speech recognition effectiveness with 1/100 labeled data volume. FastSpeech 2 [493] directly introduces voice change information (pitch, energy, duration, etc.) and uses real targets to train the model to achieve more realistic text-to-speech conversion. Seamless [494] generates low-latency target translations through streaming and using an efficient monotonic multi-head attention mechanism, while maintaining the human voice style, to achieve synchronous speech-to-speech/text translation from multiple source languages to target languages. Based on these means, the intelligent agent can achieve the ability to listen and speak.

Others At present, most of the research on intelligent agents focuses on the above-mentioned common sensory input types. However, just as humans have more than 20 types of perception, intelligent agents have also made progress in achieving corresponding perception capabilities through other sensors. The bionic olfactory chip developed by Hong Kong University of Science and Technology [551] integrates a nanotube sensor array on a nanoporous substrate, with up to 10,000 independently addressable gas sensors on each chip, which is similar to the configuration of the olfactory system of humans and other animals, and can accurately distinguish between mixed gases and 24 different odors. In terms of taste, Tongji University [560] combines fluorescence and phosphorescence signals to develop an intelligent taste sensor with multi-mode light response, which can effectively identify umami, sourness and bitterness. In order to achieve human-like perception and grasping capabilities, New York University [561] launched a low-cost magnetic tactile sensor AnySkin, which can be quickly assembled and replaced. Even in the perception of pain, the Chinese Academy of Sciences uses the unique electrical properties of liquid metal particle films when they are "injured" (mechanically scratched) to imitate the perception and positioning of "wound." Some other works, including HuggingGPT [152], LLaVA-Plus [500], and ViperGPT [498], integrate these single-modal perception capabilities within the framework, select and apply them according to task requirements, and achieve the goal of achieving more complex tasks.

7.2.2 Cross-modal Models

Text-Image Cross-modal models integrating text and images have witnessed significant advancements in recent years, leading to improved alignment, retrieval, and generation between the two modalities. These models can be categorized based on their primary objectives, including cross-modal alignment and retrieval, text-to-image generation, and image-to-text generation.

One of the primary focuses in cross-modal research is the alignment and retrieval of text and images. CLIP [51], introduced by OpenAI in 2021, employs contrastive learning to align textual and visual representations, enabling zero-shot cross-modal retrieval and classification. Similarly, ALIGN [501], developed by Google in the same year, leverages large-scale noisy web data to optimize text-image embedding alignment. In 2022, CyCLIP [562] introduced a cyclic consistency loss to further enhance the robustness of cross-modal alignment, improving the reliability of retrieval tasks.

Another major area of progress involves text-to-image generation, where models aim to synthesize high-quality images based on textual descriptions. OpenAI's DALL·E series [563, 564, 502], spanning from 2021 to 2023, has made substantial contributions in this domain, with DALL·E 3 offering fine-grained semantic control over generated images. Stable Diffusion [565], introduced by Stability AI in 2022, employs a diffusion-based generative approach that supports open-domain text-to-image synthesis and cross-modal editing.

A third significant research direction is image-to-text generation, where models aim to generate high-quality textual descriptions based on image inputs. Typical representative work is the BLIP [566] and BLIP-2 [567] models, introduced by Salesforce between 2022 and 2023, which utilize lightweight bridging modules to enhance vision-language model integration, enabling tasks such as image captioning and question answering.

Text-Video The key research here involves video text alignment, generation and retrieval. VideoCLIP [504] employs a video encoder—typically based on temporal convolution or a transformer structure—to extract sequential features from video frames. These features are subsequently aligned with textual representations generated by a language encoder, facilitating robust video-text association. In the domain of text-to-video generation, Meta's Make-A-Video model [506] extends spatial-temporal dimensions using diffusion-based techniques, allowing for high-quality video synthesis from textual descriptions. Additionally, Google's Phenaki [505] addresses the challenge of generating long, temporally coherent video sequences, demonstrating significant advancements in video synthesis through cross-modal learning.DeepMind's Frozen in Time [568] adopts contrastive learning for video-text matching, thereby enabling efficient cross-modal retrieval. This approach enhances the capacity to search and retrieve relevant video segments based on textual queries, further improving the integration of vision and language understanding.

Text-Audio Cross-modal models connecting text and audio have made significant improvements in related tasks such as modal representation, generation, and conversion, and enhanced the perception ability under a single modality.

AudioCLIP [509], introduced in 2021, extends the CLIP framework to the audio domain, enabling tri-modal retrieval across audio, text, and images. By incorporating audio as an additional modality, AudioCLIP utilizes multi-task learning to unify image, text, and audio representations into a shared embedding space. This advancement enhances the capability of cross-modal retrieval and interaction. In a similar vein, VATT [508] adopts a unified Transformer-based architecture to process video, audio, and text through independent encoding branches. These branches are subsequently fused into a shared multimodal space, facilitating tasks such as cross-modal retrieval and multi-task learning. This design allows for greater adaptability across diverse multimodal scenarios.

For text-to-audio generation, Meta introduced AudioGen [569] in 2023, which enables the synthesis of audio, such as environmental sounds and music fragments, directly from textual descriptions. This model exemplifies the growing capabilities of AI in generating high-fidelity audio based on linguistic input, expanding applications in media, entertainment, and accessibility.

Additionally, in the domain of speech-to-text and text-to-speech conversion, Microsoft developed SpeechT5 [570]. This model unifies speech and text generation, supporting both speech synthesis and recognition within a single framework. By leveraging a shared architecture for these dual functionalities, SpeechT5 contributes to the seamless integration of speech and text processing, thereby enhancing applications in automated transcription, voice assistants, and accessibility tools.

Others In some other scenarios and domains, cross-modal modeling also plays an important role.

CLIP-Forge [510] presents a novel method for generating 3D shapes from textual descriptions. By leveraging the capabilities of Contrastive Language-Image Pre-training (CLIP), this approach enables the synthesis of high-quality 3D objects conditioned on natural language inputs, bridging the gap between text and 3D geometry. Point-E [511] extends this concept by generating 3D point clouds from text descriptions. Unlike traditional 3D reconstruction techniques, Point-E focuses on point cloud representations, facilitating efficient and scalable 3D content creation while maintaining high fidelity to textual prompts.

In the field of medical imaging, MoCoCLIP [571] introduces an approach that enhances zero-shot learning capabilities. By integrating CLIP with Momentum Contrast (MoCo), this method improves the generalization of deep learning models in medical imaging applications, addressing the challenges associated with limited annotated data and domain adaptation.

7.2.3 Multimodal Models

The cross-modal model described above mainly aligns and maps between modalities through contrastive learning and other methods to achieve information complementarity and conversion between modalities. Furthermore, the work of multimodal models focuses on how to integrate the features of multiple data (such as vision, text, audio, etc.) to improve the performance of the overall model.

Vision Language Model Vision Language Model(VLM) is broadly defined as multimodal model that can learn from images(or videos) and text. Humans live in a world full of multimodal information. Visual information (such as images and videos) and language information (such as text) often need to be combined to fully express meaning. The same is true for intelligent agents. LLaVA [513] first tried to use gpt-4 to generate a multimodal language image instruction dataset. Through end-to-end training, a large multimodal model was obtained and excellent multimodal chat capabilities were demonstrated. LLaVA-NeXT [513] uses dynamic high-resolution and mixed data to show amazing zero-shot capabilities even in pure English modal data, and the computational/training data cost is 100-1000 times smaller than other methods. Emu2 [516] changes the traditional way of using image tokenizer to convert images into discrete tokens, and directly uses image encoders to convert images into continuous embeddings and provide them to Transformer,

enhancing multimodal context learning capabilities. MiniGPT-v2 [512] employs unique identifiers for various tasks during training. These identifiers help the model differentiate task instructions more effectively, enhancing its learning efficiency for each task. Qwen2-VL [515], DeepSeek-VL2 [572] use dynamic encoding strategies on visual components, aiming to process images with different resolutions and generate more efficient and accurate visual representations. At the same time, DeepSeek-VL2 [572] also uses the MoE model with a multi-head potential attention mechanism to compress the key-value cache into a latent vector to achieve efficient reasoning.

Previous work mainly uses image fusion text for training. Video-ChatGPT [573] extends the input to video and directly uses a video adaptive visual encoder combined with LLM for training to capture the temporal dynamics and inter-frame consistency relationships in video data, thereby enabling open conversations about video content in a coherent manner. To solve the lack of unified tokenization for images and videos, Video-LLaVA [574] unifies the visual representations of image and video encoding into the language feature space, making the two mutually reinforcing. Similarly, Chat-UniVi [575] employs a set of dynamic visual tokens to integrate images and videos, while utilizing multi-scale representations to allow the model to grasp both high-level semantic concepts and low-level visual details. Youku-mPLUG [576] has made in-depth research in specific scenarios. Based on the high-quality Chinese video-text pairs in the Youku video sharing platform, it enhances the ability to understand overall and detailed visual semantics and recognize scene text. Unlike the previous method that requires training, SlowFast-LLaVA [577] can effectively capture the detailed spatial semantics and long-term temporal context in the video through a two-stream SlowFast design without any additional fine-tuning of the video data, achieving the same or even better results than the fine-tuning method.

As the parameters of large models gradually decrease and the computing power of the end-side increases, high-performance end-side models are gaining momentum. Smart terminal devices such as mobile phones and PCs have strong demands for image visual processing, which puts forward higher multimodal recognition effects and reasoning performance requirements for the deployment of AI models on the end-side. TinyGPT-V [517] is built based on the Phi-2 [578] small backbone combined with BLIP-2 [567], only 8G video memory or CPU is needed for reasoning, and solving the computational efficiency problems of LLaVA [513] and MiniGPT-4 [579]. MiniCPM-V [519] mainly provides powerful OCR capabilities for long and difficult images, and has a low hallucination rate, providing reliable perception output. Megrez-3B-Omni [580] ensures that all structural parameters are highly compatible with mainstream hardware through coordinated optimization of software and hardware. Its inference speed is up to 300% faster than that of models with the same precision, improving its adaptability to different end-side hardware.

Similarly, there are more GUI-related works focusing on automatic task execution on mobile phones and PCs. Omni-Parser [520] uses popular web page and icon description datasets for fine-tuning, significantly enhancing the detection and functional semantic expression capabilities of icons in screenshots. GUICourse [581] and OS-ATLAS [582] also built a cross-platform GUI grounding corpus, which brought significant performance improvements in the understanding of GUI screenshots and enriching the interactive knowledge of GUI components.

Vision Language Action Model Vision-Language-Action (VLA) model, which takes vision and language as inputs and generates robotic actions as outputs, represents an important research direction in the field of embodied intelligence. The selection of vision and language encoders in VLA models has undergone diverse development, evolving from early CNNs to Transformer architectures, and further integrating 3D vision and large language models. Early models such as CLIPort [521] used ResNet [488] to process visual inputs and combined language embeddings to generate actions, laying the foundation for multimodal fusion. RT-1 [522] introduced the Transformer architecture, employing EfficientNet as the visual encoder and USE as the language encoder, and fused visual and language information via FiLM mechanisms, significantly enhancing the model's generalization ability. VIMA [523] further adopted multimodal prompts, combining the ViT visual encoder and the T5 language model to support more complex tasks. PerAct [524] innovatively used 3D point clouds as visual inputs and processed multi-view information through Perceiver IO, providing richer spatial perception for robotic manipulation. Diffusion Policy [525] combined ResNet visual encoders and Transformer language models, generating actions through diffusion models to improve the diversity and accuracy of action generation. SayCan [583] integrated the PaLM language model with visual inputs, using the CLIP visual encoder for task decomposition. PaLM-E [526] combined the ViT visual encoder and the PaLM language model, guiding low-level action execution through text planning. MultiPLY [527] further integrated 3D information into LLMs, combining the EVA visual encoder and the LLaMA language model to provide more comprehensive planning capabilities for complex tasks.

Audio Language Model Audio Language Model(ALM) uses the audio and text to build multimodal model. Speechgpt [533] built a large-scale cross-modal speech instruction dataset SpeechInstruct and trained discrete speech representations, achieving cross-modal speech dialogue capabilities beyond expectations. LauraGPT [584], unlike the previous sampling of discrete audio tokens to represent input and output audio, proposed a novel data representation that combines the continuous and discrete features of audio, and demonstrated excellent performance on a wide range of

audio tasks through supervised multi-task learning. [529, 585, 531] converts audio data into embedded representations and then fine-tunes instructions, so that excellent performance can be achieved on various speech processing tasks through natural language instructions. In order to reduce the cost of fine-tuning training, Audio Flamingo [528] quickly enhances the ability to adapt to unseen tasks through contextual learning and retrieval based on the audio language model. UniAudio 1.5 [530] uses words or subwords in the text vocabulary as audio tokens, learns these audio representations through a small number of samples, and achieves cross-modal output without fine-tuning. In order to make the output more realistic and in line with human expectations, Qwen2-Audio [54] introduced the DPO training method to achieve human preference alignment.

Audio Vision Language Model Audio Vision Language Model (AVLM) ultilizes audio, vision, and text to unify multimodal models. Previously, we introduced some work on building multimodal models using information from two modalities. In the pursuit of AGI, the obstacle to achieving this goal lies in the diversity and heterogeneity of tasks and modalities. A suitable approach is to allow more modal capabilities to be supported within a unified framework. Some closed-source work [586, 587] has achieved excellent capabilities across modalities such as text, vision, and audio. ImageBind [588] implements joint embedding across six different modes (image, text, audio, depth, thermal, and IMU data). Panda-GPT [535] combines ImageBind's multi-modal encoder and Vicuna [589], showing zero-shot cross-modal performance in addition to images and text. Similar work includes [539, 539, 536], which achieves alignment and training through the encoding information of vision, audio and text. Multimodal models often require more resources to train, and UniVAL [538] trained a model with only $\sim 0.25B$ parameters based on task balance and multimodal curriculum learning, and used weight interpolation to merge multimodal models, maintaining generalization under out-of-distribution. NExT-GPT [542] connects LLM with multimodal adapters and different diffusion decoders, and only trains a small number of parameters (1%) of certain projection layers.

Other works [543, 590, 544, 545] have achieved input-output conversion between arbitrary modalities. Unified-IO 2 [543] is the first autoregressive multimodal model that can understand and generate images, text, audio, and actions. It tokenizes different modal inputs into a shared semantic space and processes them using an encoder-decoder model. AnyGPT [590] builds the first large-scale any-to-any multimodal instruction dataset, using discrete representations to uniformly process various modal inputs. Modaverse [545] directly aligns the output of the LLM with the input of the generative model to solve the problem that previous work relies heavily on the alignment of the latent space of text and non-text features, avoiding the complexity associated with the alignment of latent features. CoDi-2 [544] outperforms earlier domain-specific models in tasks like topic-based image generation, visual transformation, and audio editing.

Others Humans have explored the 2D world more than the 3D world, but 3D can more accurately describe the shape and texture information of objects and provide richer perceptual information. PointLLM [540] uses a point cloud encoder to express geometric and appearance features, and integrates language features for two-stage training of complex point-text instructions, achieving excellent 3D object description and classification capabilities. Since 3D contains richer information than 2D, it also brings greater training costs. [541, 591] reduces the training cost here, and MiniGPT-3D [541] uses 2D priors from 2D-LLM to align 3D point clouds with LLMs. Modal alignment is performed in a cascade manner, and query expert modules are mixed to efficiently and adaptively aggregate features, achieving efficient training with small parameter updates. LLaVA-3D [591] connects 2D CLIP patch features with their corresponding positions in 3D space, integrates 3D Patches into 2D LMM and uses joint 2D and 3D visual language command adjustment to achieve a 3.5-fold acceleration in convergence speed.

In order to enable intelligent agents to accurately perceive and manipulate unknown objects, Meta [592] developed NeuralFeels technology, which combines vision and touch to continuously model unknown objects in 3D, more accurately estimate the posture and shape of objects in handheld operations, and improve the accuracy of ignorant object operations by 94%.

7.3 Optimizing Perception Systems

Perception errors, including inaccuracies, misinterpretations, and "hallucinations" (generation of false information), pose substantial challenges to the reliability and effectiveness of LLM-based agents. Optimizing perception thus requires minimizing these errors using various strategies across model, system, and external levels.

7.3.1 Model-Level Enhancements

Fine-tuning. Fine-tuning pre-trained LLMs on domain-specific data significantly improves their ability to accurately perceive and interpret relevant information. For example, fine-tuning models such as LLaVA on specific landmarks has been shown to enhance their recognition accuracy, particularly in urban navigation tasks [513, 593]. Moreover, techniques such as Low-Rank Adaptation (LoRA) enable more efficient fine-tuning, avoiding a substantial increase in

model complexity while still improving performance [109, 594]. Some LLM work combined with traditional vision is also widely used. Integrating with YOLOS [595] on the basis of the the Llama-Adapter [596] architecture significantly improves the detection and positioning capability.

Prompt Engineering. The design of effective prompts is crucial to ensure LLMs generate outputs that are both accurate and aligned with the desired goals. By providing clear instructions, contextual information, and specific formatting requirements, prompt engineering minimizes misinterpretation and hallucination [597]. System prompts define the agent's role, historical prompts to provide context from past interactions, and customized prompts to ensure output consistency has been shown to reduce errors significantly [597].

Retrieval-Augmented Generation. Supplementing LLMs with external knowledge sources through retrieval mechanisms helps ground their responses in factual information, reducing the likelihood of hallucinations and improving the accuracy of perceived information [334].

7.3.2 System-Level Optimizations

Anticipation-Reevaluation Mechanism. In scenarios where agents face incomplete or ambiguous information, an anticipation-reevaluation mechanism can enhance robustness. For instance, in navigation tasks, agents can anticipate goal directions based on historical data and reevaluate their inferences when new information becomes available [598].

Multi-Agent Collaboration. In multi-agent systems, structured communication and collaboration among agents can facilitate information sharing, error correction, and consensus-building, leading to a more accurate collective perception of the environment [599]. Different communication topologies, such as fully connected, centralized, and hierarchical structures, offer varying trade-offs in terms of efficiency and robustness [600]. InsightSee [601] refines visual information through a multi-agent framework with description, reasoning, and decision-making, effectively enhancing visual information processing capabilities. Similarly, HEV [602] integrates the global perspective information of multiple agents and endows RL agents with global reasoning capabilities through cooperative perception, thereby enhancing their decision-making capabilities.

Agent Specialization. Assigning distinct roles and capabilities to individual agents within a multi-agent system allows for a division of labor in perception, with each agent focusing on specific aspects of the environment or task. This can enhance the overall accuracy and efficiency of perception [603].

7.3.3 External Feedback and Control

Loss Agents for Optimization. Utilizing LLMs as loss agents, allows for the dynamic adjustment of loss function weights during training [604]. This enables the optimization of image processing models based on complex, potentially non-differentiable objectives, including human feedback and evaluations from specialized models. This approach essentially externalizes the optimization objective, allowing the LLM to "perceive" and adapt to complex criteria [605].

Human-in-the-Loop Systems. Incorporating human feedback and oversight can help correct errors, guide the agent's learning process, and ensure alignment with human values and expectations [43].

Content and Output Mediation. Before presenting LLM outputs to users, content mediation filters and refines these outputs. This helps prevent unexpected or harmful behaviors, ensuring alignment with user expectations and safety guidelines [606].

7.4 Perception Applications

The operational efficacy of intelligent agents is predominantly influenced by three critical factors: model architecture dimensionality, hardware infrastructure specifications, and quantization optimization methodologies. The exponential progression in model parameters—from Bert-Base's modest 110M to GPT-3's substantial 175 billion, culminating in Llama 3's unprecedented 405 billion—has correspondingly escalated processing latency from milliseconds to hundreds of milliseconds. Hardware performance variations are particularly noteworthy; empirical evidence with GPT-3 demonstrates that NVIDIA H100 exhibits a 50% improvement in token processing throughput compared to A100, while RTX 4090 achieves approximately double the processing capability.

Contemporary intelligent agents have penetrated diverse domains, encompassing personal assistance systems, gaming environments, Robotic Process Automation (RPA), and multimedia content generation, predominantly leveraging visual perception as their primary input modality. In the context of procedurally generated environments like Minecraft, STEVE [607] demonstrates remarkable performance improvements, achieving a 1.5x acceleration in technology tree progression and a 2.5x enhancement in block search efficiency through visual information processing. Steve-Eye [608]

advances this paradigm through end-to-end multimodal training, addressing environmental comprehension latency through integrated visual-textual input processing.

In creative content generation, AssistEditor [609] exemplifies sophisticated multi-agent collaboration, facilitating professional video editing through style-driven content understanding. Similarly, Audio-Agent [610] implements cross-modal integration between textual/visual inputs and audio outputs, enabling comprehensive audio manipulation capabilities [611, 612, 613].

Mobile and desktop platforms have witnessed significant advancements in agent applications. ExACT [614] has established new state-of-the-art benchmarks in VisualWebArena [615], achieving a 33.7% Success Rate through screenshot-based exploratory learning with caption and Set of Mask integration. SPA-Bench [616] introduces a comprehensive mobile evaluation framework that authentically replicates real-world complexity. M3A [617] demonstrates superior performance with a 64.0% success rate in SPA-Bench through multimodal input processing. AgentStore [618] has markedly improved OSWorld PC benchmark performance to 23.85% through enhanced visual and accessibility tree processing.

Voice interaction capabilities [619, 586] in personal AI assistants have significantly reduced interaction friction while enhancing operational efficiency. The integration of emotional prosody in voice interactions has demonstrated increased user engagement and retention.

In embodied intelligence applications, haptic and force feedback mechanisms have emerged as crucial modalities for environmental interaction, with enhanced sensory fidelity enabling increasingly precise operational capabilities [620].

7.5 Summary and Discussion

Although more and more research works [543, 590] focus on building unified multimodal models to support the input and output of multiple perception capabilities. Agent perception, a cornerstone of autonomous systems, faces significant challenges in effectively interpreting and integrating multi-modal data. Current methodologies encounter persistent issues in representation learning, alignment, and fusion, which hinder the development of robust and generalizable perception systems.

One of the primary issues lies in the representation methods employed, which often fail to capture the intricate nuances of multi-modal data. This shortfall is particularly evident in scenarios where high-dimensional sensory inputs require a sophisticated abstraction that preserves critical semantic information. Furthermore, the alignment of representations presents additional difficulties. Integrating heterogeneous data types into a cohesive feature space is not only computationally intensive but also prone to inconsistencies, which can lead to misinterpretation of ambiguous signals. The challenge is compounded when attempting to fuse these diverse representations, as the process of merging features from various sources frequently results in suboptimal integration and potential loss of vital information.

Future research directions should prioritize adaptive representation learning through dynamic neural architectures capable of automatically adjusting their structure based on environmental context and task demands. This could involve meta-learned parameterization or graph-based representations that explicitly model relationships between perceptual entities. For cross-modal alignment, self-supervised spacetime synchronization mechanisms leveraging contrastive learning principles show promise in establishing dense correspondence without requiring exhaustive labeled data. The integration of causal inference frameworks into alignment processes [621] could further enhance robustness against spurious correlations. In representation fusion, hierarchical attention mechanisms with learnable gating functions merit deeper exploration to enable context-aware integration of complementary modality features. Emerging techniques in differentiable memory networks may provide new pathways for maintaining and updating fused representations over extended temporal horizons.

Chapter 8

Action Systems

In the realm of philosophy, action is defined as the behaviors that agents can perform for a potential or specific purpose in the environment. For example, manipulation, moving, reasoning, and tool utilization can all be considered as fundamental actions that an intelligent agent can execute to fulfill a goal in real-world scenarios. In other words, actions emerge from the goal-oriented engagement of an agent in its environment, reflecting its intent to transform the external world in pursuit of its goals. Therefore, the action system also plays a vital role in differentiating AI agents and foundation models (e.g., LLMs). Generally, existing foundation models have demonstrated impressive performance across various tasks, but their task scope is still limited as they predominantly relies on the original pre-training objective (e.g., next-token prediction). By serving foundation models as brain intelligence, AI agents equipped with action systems can directly engage with their environment and execute complex user intent. Moreover, action systems can support agents to utilize available tools from external environments, thus significantly extending agents' task scopes. Therefore, the design of action systems will also determine the capability of AI agents in perception, decision making, execution, tool utilization, and any other components to align with the human brain. In other words, foundation models lay the groundwork for agents while action systems determine their ultimate potential to achieve complex targets. Designing an effective and comprehensive action system for AI agents is a critical endeavor that involves significant challenges and notable benefits. In Figure 8.1, we demonstrate the execution process of the action system in the cognition system. In this section, we will first discuss the human action system in Section 8.1, and then examine the transition from human action to agentic action in AI agents in Section 8.2. After that, we will systematically summarize the paradigms of existing action systems in AI agents, including action space, action learning, and tool learning, in Section 8.3. In Section 8.4, we analyze the differences between action and perception, and finally we summarize the conclusion in Section 8.5.

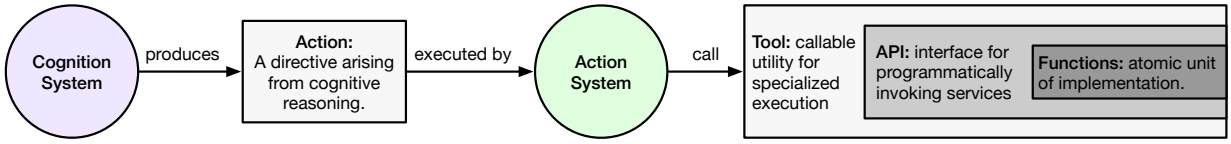


Figure 8.1: Illustration of several concepts related to action and action execution.

8.1 The Human Action System

Action system in human cognition refers to the processes that allow humans to perceive, plan, and execute goal-directed actions. It is a complex system that enables individuals to interact with a dynamic environment, make decisions, and adapt their behavior based on feedback. Generally, the action system within human cognition could be broadly categorized as *mental action* and *physical action*:

- **Mental action** can be viewed as a kind of distinct action, which is formulated as a thinking process to drive the final intention in the human brain. For example, reasoning, decision making, imagining, and planning can all be considered as various types of mental action. In other words, mental actions are equal to a brain signal that drives the physical actions of humans to fulfill the final objective.

- **Physical action** refers to any goal-directed bodily movement executed by the human motor system. To some extent, physical actions are usually expressed as a kind of continuous action. For example, speaking, manipulating, drawing, running, and grasping can all be regarded as physical actions. Employing a sequence of physical actions, humans can conduct the interaction and collect feedback from real-world environments.

Figure 8.2 illustrates a simple taxonomy of the human action system from the perspective of mental action and physical action. Empowered with both mental and physical actions, the human cognition system can handle diverse complex tasks from real-world scenarios. Drawing inspiration from human cognition, it is also essential for us to revisit how to formulate action systems in AI agents across different tasks, from language to digital and then in physical environments.

Model	Examples	Inputs	Objective	Definition
Large Language Model (LLM)	GPT-4 [7]	Language	Next-Token Prediction	LLM is to generate text based on the provided user prompts.
Large Multimodal Model (LMM)	LLaVA [513]	Multi-modal	Multi-modal Generation	LMM is to generate multimodal data based on multimodal inputs.
Robotic Foundation Model (RFM)	RT-1 [522]	Sensory inputs	Robotic Control	RFM is to generate robotic control based on the sensory inputs from dynamic environments.
Large Action Model (LAM)	LAM [622]	Interactive Environment	Executable Action	LAM is to generate executable actions based on the interactions within the environment.

Table 8.1: Definitions between different kinds of foundation models.

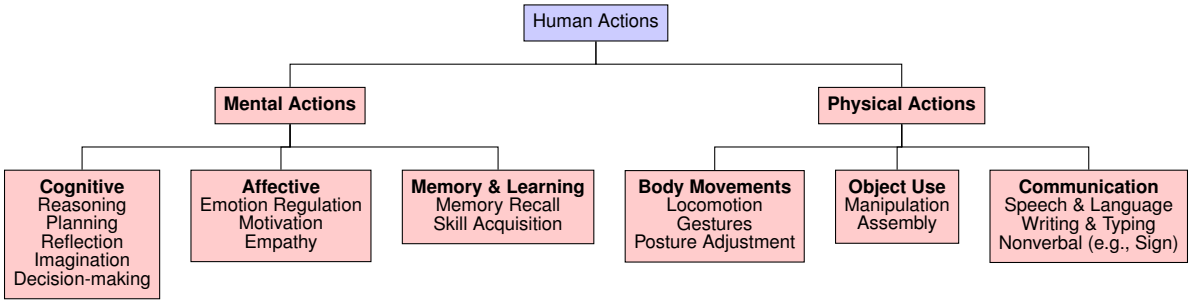


Figure 8.2: Illustrative Taxonomy of Human Actions, showing both mental and physical facets.

8.2 From Human Action to Agentic Action

In the past long period of time, human action systems [623] have significantly motivated us to shape the development of a computer system toward autonomous paradigms. The action mechanism plays a critical role in the human brain in driving goal-directed behavior. In an intelligent human brain [624], conscious and unconscious thinking signals are produced, converted into mental signals, which eventually lead to a sequence of action operations. This process can be mapped as a multi-stage pipeline that involves constructing action spaces, formulating learning mechanisms for improved decision making, and integrating external states (e.g., tools). Inspired by these principles, we discover that these designs are essential to formulate the prototype of AI agent.

Many existing frameworks incorporate action learning into their design or utilize it as an output. To clarify the definition of an action system, we highlight the distinctions among various frameworks, including large language models (LLM), large multi-modal models (LMM), robotic foundation models (RFM), and large action models (LAM), as shown in Table 8.1. Specifically, an LLM is to produce language output based on provided prompts, while an LMM is to generate multi-modality artifacts based on the multi-modal inputs. Existing language-based or digital AI agent frameworks are built upon these foundation models (e.g., LLM or LMM) via predefining the scope of action space and its learning strategies. On the other hand, an RFM is to optimize robotic control based on real-world environments (e.g., robotic video). Existing RFMs are pre-trained from web-scale video data and use video prediction to simulate the action of robotic control. The core of RFM is still to use the generative objective to learn knowledge from large-scale data, although it has involved some action designs in building physical AI agents. Moreover, some recent works [622] introduce the concept of large action model (LAM), which further highlights the stage to generate the action strategies, interact with real-world environments and enhance self-learning paradigm. From these definitions, we notice that, regardless of the foundational models employed, the core of action system is to build the interaction with the environment and then enable the learning process from the collected action trajectories via pre-defined reward

functions. Specifically, the mechanisms underlying these behaviors are also similar to the action system in human cognition, offering valuable insights for designing action systems in AI agent frameworks. For example:

- When processing different scenarios, humans usually will pre-define the action space to perform action trajectories to solve specific tasks. For instance, when playing computer games like Minecraft, we will set our action operations via keyboard or mouse to simulate behaviors like building house, mining gold, and so on. On the basis of this, we also need to build or create an action space for handling complex tasks in AI Agent frameworks.
- Compared to machines, the human cognitive system excels in continuously acquiring new knowledge through real-world interactions, guided by generating and optimizing the action sequences. Thus, replicating this learning ability in AI agents is essential to adapt the dynamic environment and build a new skill library.
- In addition, with the development of human civilization, learning to use external tools has been recognized as one of the most significant milestones in the evolution of human intelligence. By leveraging these external tools, humans can extremely extend the problem-solving capability in different scenarios, from the stone age to the industrial revolution.

To this end, we expect to build the mapping between the action system of human cognition system and the design of AI Agent framework, including how to build action space for AI agent from specific scenarios to general domain, how to build action learning within the environment, and how to leverage external states (e.g., tools) to extend the task scope of AI Agent. By developing this a systematic survey, we strive to provide more in-depth insights for the community with a clear understanding of the significance of action systems in AI agent frameworks.

8.3 Paradigms of Agentic Action System

Generally, the action system of AI agent frameworks consists of three major components: 1) the action space $\mathcal{A}$, which includes all types of action that agent can perform in real-world scenarios or downstream tasks, and can vary significantly depending on different agent settings, ranging from language-based agents to embodied agents; 2) the action learning within an dynamic environment that determines the state $\mathcal{S}$, observation $\mathcal{O}$ and the optimization process of agent; 3) the tool space $\mathcal{T}$ that encompasses the instruments, interfaces, or middle-wares the agent can perform for utilization, which ranges from physical devices such as robotic arms to digital interfaces like APIs. Overall, these components collectively define the scope and characteristics of the action system for AI agents, shaping their formulation and execution.

To fully explore the possible actions a_t in practical scenarios, we must formally represent the action space and consider both individual operations and the underlying hierarchical reasoning processes. This means examining the action space at various levels, from low-level manipulations to high-level operators that orchestrate complex workflows.

Accordingly, the AI agent decision making process can be formalized as a trajectory $\langle o_t, s_t, a_t \rangle$, where a_t is selected from the action space $\mathcal{A}$ to transform the current state s_t based on observation o_t into the next state. In some cases, integrating external tool systems may also be necessary. By executing a sequence of $\langle o_t, s_t, a_t \rangle$, the agent is steered toward achieving its final objectives.

8.3.1 Action Space Paradigm

Action space $\mathcal{A}$ is an important component, which serves as the basis for building an action system within AI agent frameworks. The composition of the action space determines how AI agents solve complex tasks in different scenarios. In Figure 8.2, we present an illustrative taxonomy of the action system based on its action space. Generally, we summarize the action space within existing works as three distinct types, as outlined below.

Language Language-based AI agents typically operate through language-driven actions in interactive linguistic environments, such as reasoning, programming, retrieving information, executing API calls, or interacting with external tools. In our study, we summarize three distinct types of language-based action spaces, including plain text, code programming, and communication. Specifically, early language-based AI agents are built with plain text, which aim to perform interactive decision-making in verbal environments or text-based games. Here, ReAct [70] is a representative language-based AI agent, which synergizes the reasoning and actions of an LLM to solve various problems. AutoGPT [625] analyzes and decomposes user requests into multiple subtasks and uses web search or other tools to tackle each of them. Reflexion [48] involves self-refinement and the memory mechanism to enhance action execution in language tasks. LLM+P [163] empowers LLM-based agent with planning capability to aid decision-making. However, converting plain text into an executable command usually requires LLMs to first interpret the text and then perform instruction conversion, leading to additional information loss. To this end, some work explores using

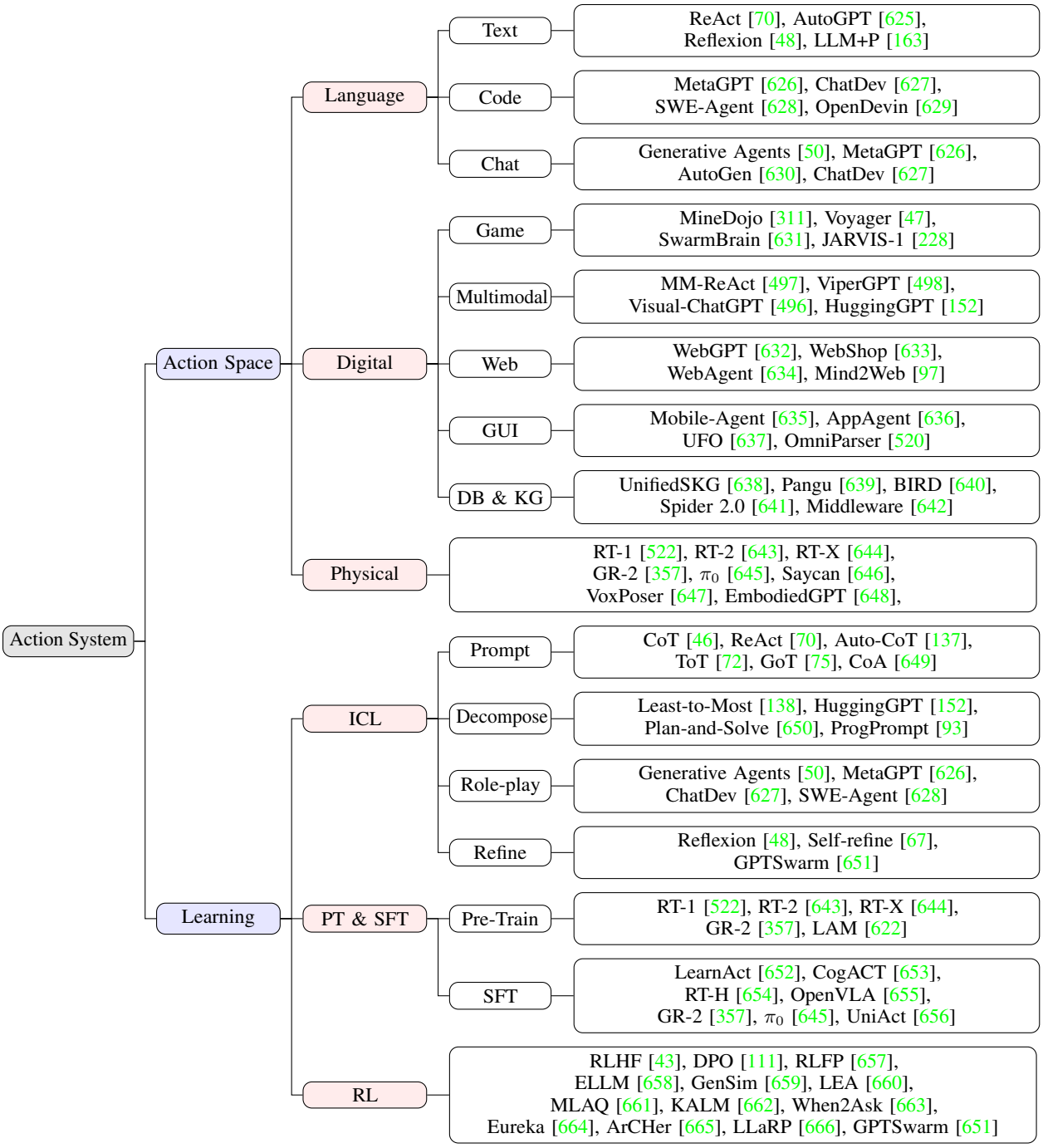


Figure 8.3: Illustrative Taxonomy of Action system, including action space and learning paradigm.

code as the action space, allowing direct execution of the generated code and self-verification. MetaGPT [626] and ChatDev [627] build the action space via programming language with multi-agent collaboration. SWE-Agent [628] consider different stages of software engineering and thus solve software issues. OpenDevin [629] devises an automatic software development platform that integrate code writing, interaction with the command, sandbox for code execution, and collaborations. Moreover, some frameworks are built based on multi-agent communications, and then use chatting to analyze which actions should be employed in the next step. Here, Generative Agents [50] directly simulate multiple characters in a virtual town, to explore how each agent to conduct next action. MetaGPT [626] and ChatDev [627]

are both multi-agent frameworks to faciliate the development of software engineering. AutoGen [630] is also a representative framework that enable multiple agent collaboration to solve any complex tasks. Generally, language-based AI agents, empowered by LLMs, perform effectively in linguistic interactions. However, limited to the scope of the action space, it also poses challenges of how to solve more complex tasks in real-world scenarios. Therefore, we also need to formulate new research solutions to construct a more sophisticated action space to solve challenging tasks.

Digital To expand the capabilities of AI agents beyond language, some works have also developed advanced AI agents that operate within digital environments, such as web proxies, online shopping platforms, and gaming systems. For examples, MineDojo [311] devises a virtual agent via video-language pre-training and simulates an environment that supports a multitude of tasks and goals within Minecraft. Moreover, Voyager [47] is an embodied AI agent trained to play Minecraft. It simulates multiple executable actions in code form to develop a skill library via interacting with the Minecraft environment, and thus improve the capability of virtual agents. JARVIS-1 [228] is an open-world agent that can handle multi-modal inputs / outputs, generate sophisticated plans, and perform embodied control. It explores the evolutionary behaviors of the agent when acting in Minecraft. SwarmBrain [631] is an embodied agent that uses LLMs to act strategically and in real time in StarCraft II. Additionally, some research studies investigate how LLMs can act to process multimodal tasks. MM-ReAct [497] and ViperGPT [498] apply LLMs to perform the thinking process for multimodal tasks and then select visual experts for task solving. Visual-ChatGPT [496] integrates multiple visual experts and uses LLMs as the controller to solve tasks. HuggingGPT [152] directly involves four stages, including task planning, model selection, model execution and response generation, to automatically analyze user instructions and predict the final answers based on complex multimodal tasks. It is also vital for the agent to keep up with the latest information available online. Therefore, some AI Agent frameworks (e.g., WebGPT [632], WebAgent [634]) are designed to interact with search engine to enhance the capability of agent to discover the answers from website. WebShop [633] is used to explore the potential of AI Agent for online shoping. Mind2Web [97] is to build a generalist agent that simulate multiple complex web tasks. As foundation agents advance in processing multimodal tasks or web tasks, there is a increasing trend to enhance their capability in solving complex computer tasks. Mobile-Agent [635] utilizes multimodal models as the cognitive controller to manage and orchestrate mobile functionalities. AppAgent [636] defines various app usages as action spaces, enabling foundation models to interact with different apps as a mobile intelligent assistant. UFO [637] and OmniParser [520] are two advanced GUI agents which manipulates UI operations as the action space, enabling AI agent to perform computer-use tasks. Generally, empowered with more advanced skills in digital environment, AI agent can demonstrate better intelligent in solving complex tasks, and represent a significant shift from language intelligent to digital intelligent. By expanding the action space to include web browsing, GUI interaction, mobile applications, and embodied systems, AI agents are evolving into more autonomous, multimodal, and context-aware systems, bridging the gap between foundation models and human cognition systems. In addition, other research explores LLM integration with structured digital environments such as relational databases and knowledge graphs (KGs). Pangu [639] pioneered the connection between LLMs and large-scale KGs, while BIRD [640] and Spider 2.0 [641] established a foundation for LLMs to operate with enterprise databases in real-world settings. NL2SQL-BUGs [667] addresses the critical challenge of identifying semantic errors in NL2SQL pipelines [365], which enhances the reliability of LLM-driven interactions with relational databases [668]. Similarly, frameworks like UnifiedSKG [638] and Middleware [642] expand LLMs' action capabilities across both databases and KGs.

Physical Building an AI agent to interact with the real physical world can be viewed as the ultimate objective to simulate a computer program to act as a human cognition system. To achieve this, we require the agent to be capable of processing signals from real-world environments and generating feedback to facilitate continuous improvement. Therefore, it will pose new challenges on how to process the continuous signals collected by sensors and enable foundation models to make decisions. To fulfill this, RT-family [522, 643, 644] pre-trained vision-language-action models to integrate knowledge from web videos into robotic learning, enhancing robotic control and action execution. GR-2 [357] is a robotic model that undergoes large-scale pre-training on video clips and language data, followed by fine-tuning on robot trajectories for robotic action prediction. π_0 [645] pre-trained a robotic model based on robot platforms, including single-arm robots, dual-arm robots, and mobile manipulators, to build robotic learning in physical systems. SayCan [646] bridges the connections between robotic semantics and LLMs, using the robotic model to provide perception for LLMs and then using LLMs to make high-level decision-making. VoxPoser [647] uses LLMs to understand and decompose 3D Value Maps for Robotic Manipulation. Besides, EmbodiedGPT [648] utilizes vision-language models to understand video data and perform decision-driven actions. In physical environments, it is worth noting that we usually need to understand continuous signals and then generate continuous actions for robotic control. Despite the existing foundation models that can effectively process discrete-level actions (e.g., language or computer-use), how to process long continuous signals is still challenging. Therefore, eliminating the differences between continuous signals and discrete signals in foundation models is still a major problem.

Generally, action space serves as one of the most critical components in building an effective AI Agent system. An effective action space enhances the capability and efficiency of the AI Agent in processing downstream tasks. Action space usually ranges from the discrete space (e.g., skill library in Atari games) to the continuous space (e.g., robotic manipulation). As AI agents become more autonomous and multimodal, designing effective action spaces will be crucial for advancing general-purpose AI systems capable of real-world interactions.

8.3.2 Action Learning Paradigm

In the human cognition system, action learning [669] represents the problem-solving process, involving both taking actions and reflecting on feedback. Similarly, action learning for AI agents refers to the iterative process by which an autonomous AI system refines its decision making and behavior through direct interaction with the real world environment. Generally, action learning encompasses a cycle of multiple stages, including building action space, choosing actions, and optimizing action selection based on interaction with the environment (e.g., receiving feedback or rewards and adjusting policy for choosing actions). By iteratively deploying these strategies, AI agents can adapt to the latest information or changing conditions in real time, ultimately enabling more robust, flexible, and efficient problem-solving capabilities. Therefore, an effective action learning mechanism is crucial for the optimization of agentic action systems. In this part, we mainly focus on three different representative learning paradigms, including in-context learning, supervised training, and reinforcement learning, which are discussed below:

In-context Learning As large language models have demonstrated emergent ability, in-context learning has been considered as the most effective method to leverage the existing capabilities of LLM without any modifications. Provided with well-designed prompts to describe actions, AI agents can understand specific actions, perform these actions, reflect on the outcome of the interaction with the environment, and finally achieve goals. Among these approaches, the common method is to use prompting techniques to instruct LLMs to generate agentic action. Here, the most representative one is Chain-of-Thought (CoT) [46] prompting, which applies "*Let us think step by step*" technique to generate a sequence of intermediate reasoning steps, exploring potential solutions systematically. ReAct [70] enables LLMs to generate reasoning trails and task-specific actions through interaction within the environment, improving the reasoning and decision-making capabilities of AI agents. LearnAct [652] devises an iterative learning strategy to expand action space by generating code (i.e., Python) to create and revise new actions. Moreover, some works (e.g., Auto-CoT [137] explores how to automatically generate CoT via LLMs and then enable the autonomous thinking process of AI agents. To handle more complex tasks, ToT [72] considers the thought process as a tree structure and introduces the tree search via LLM prompting, while GoT [75] applies a graph structure along with the graph search. For robotic models, CoA [649] designed four different prompt settings (e.g., object, grasp, spatial, and movement) to allow robot manipulation with reasoning process. Furthermore, to tackle more complex tasks that require intricate agentic workflows, some frameworks introduce the stage of task decomposition via LLM prompting to break down user instructions. Least-to-Most [138] is a classical prompting technique to convert user instructions into multiple subtasks. HuggingGPT [152] is a representative AI agent framework that applies task planning to transform user requirements into actionable items. Plan-and-Solve [650] directly uses LLM to make plans from user instructions and then give answers based on the generated plans. Progprompt [93] applies similar task decomposition to robotic tasks. In addition, using prompting techniques to formulate the characteristic of AI agent has also been considered as an increasing trend to facilitate the simulation and productivity of AI agent frameworks (e.g., Generative Agents [50], MetaGPT [626], ChatDev [627], SWE-Agent [628]). Finally, some other frameworks (e.g., Reflexion [48] or Self-refine [67]) analyze the external feedbacks of user interaction within the environment and then iteratively refine and polish results via well-designed reflexion prompts. All of these designs allow us to better understand user instructions, decompose task goals, and make plans for thinking answers. In-context learning can help us avoid parameter optimization and reduce the heavy cost of training LLMs. It allows AI agents to perform various actions effectively and adapt to a wide range of domains. However, challenges still remain if we want to acquire agents of even stronger action learning ability.

Supervised Training To further improve the action learning ability of foundation models, increasing research efforts have focused on training methodologies, including self-supervised pretraining (PT) and supervised fine-tuning (SFT). For the pre-training paradigm, the most representative works is RT-family [522, 643, 644], which pre-trains robotic Transformer on large-scale web and robotic data, yielding a powerful vision-language-action model. Following this policy, GR-2 [357] is developed through extensive pre-training on a large corpus of web videos to understand the dynamics of the world and post-training on robotic trajectory data to specialize in video generation and action prediction. Similarly, LAM [622] is a large action model pre-trained on trajectories of user interaction with computer usage. However, the pre-training paradigm usually incurs massive computation costs. Therefore, many works take the fine-tuning paradigm to enhance the action capability of foundation models. OpenVLA [670] is built upon the Llama2 [11] language model and incorporates a visual encoder based on DINOv2 [671] and SigLIP [672]. It is fine-tuned on a diverse set of real-world robot demonstrations from Open X-Embodiment (OXE) [673] and outperforms RT-2-X [673]

across different tasks, all while utilizing 7× fewer parameters. Building upon OpenVLA, CogACT [653] integrates an additional diffusion action module and introduces an adaptive action ensemble strategy for inference. It is also fine-tuned using datasets from OXE and demonstrates a 35% improvement in the SIMPLER [674] simulated environment and a 55% increment in real robot tasks using the Franka Arm. Besides, some works also explore how to enable robotic model to learn action from plain language in physical world. For examples, RT-H [654] introduces a hierarchical architecture to build action space, which first predict language motions and then generate low-level actions. And π_0 [645] collected massive diverse datasets from different dexterous robot platforms, and then fine-tune the pre-trained VLMs to learn robotic actions. UniAct [656] learns universal actions that capture generic atomic behaviors across differently shaped robots by learning their shared structural features. This approach achieves cross-domain data utilization and enables cross-embodiment generalizations by eliminating heterogeneity [132]. Overall, using supervised training, including pre-training and supervised fine-tuning, can effectively adapt foundation models to perform actions intelligently in real-world scenarios. Last but not least, it is worth noting that, even with extensive training on a vast corpus, it is still beneficial to apply in-context learning on top of the trained model for AI agents, in an pursuit for their best performance.

Reinforcement Learning To facilitate an action learning procedure in addition to in-context learning and supervised training, it is also crucial for agents to interact with the environment and eventually optimize their action policy through experience, feedback, or rewards. Considering this iterative and sequential nature, reinforcement learning (RL) provides us with the systematic methodology we need [675, 676, 677, 678]. In RL paradigms, there are several classical and representative algorithms, such as Deep Q-Network (DQN) [679] and Proximal Policy Optimization (PPO) [680]. The most representative RL work that applied reinforcement learning to foundation models is InstructGPT [43], which effectively aligns LLM outputs with human preferences via RLHF. Since RLHF usually requires additional training to build the reward model, some papers (e.g. DPO [111]) proposes to directly optimize preference data through contrastive learning. Existing work [89, 681] also demonstrate the potential of scaling the RL algorithm for foundation models to produce long CoT thinking stages with impressive performance. Although RL paradigms have been successfully used to fine-tune LLMs for text generation tasks [12, 682, 43, 683], efficiently utilizing the RL algorithm for action learning remains one of the many challenges that require further attempts. Recent advances indicate significant progress in applying RL to action learning with LLMs from various perspectives:

- Given the rich world knowledge encapsulated in LLM, we can use LLM to *mimic external environments or generate imagined trajectories* to aid agents in action learning. For instance, RLFP [657] utilizes guidance and feedback from the policy, value, and success-reward foundation models to enable agents to explore more efficiently. Similarly, ELLM [658] utilizes large-scale background knowledge from LLMs to guide agents in efficient exploration within various environments. GenSim [659] automatically generates rich simulation environments and expert demonstrations by exploiting the coding abilities of LLM, thereby facilitating the capability of the agent for free exploration. LEA [660] leverages the language understanding capabilities of LLM and adapts LLM as a state transition model and a reward function to improve the performance of offline RL-based recommender systems. MLAQ [661] utilizes an LLM-based world model to generate imaginary interactions and then applies Q-learning [684] to derive optimal policies from this imaginary memory. KALM [662] fine-tunes LLM to perform bidirectional translations between textual goals and rollouts, allowing agents to extract knowledge from LLM in the form of imaginary rollouts through offline RL. In general, empowered by RL paradigms, we can significantly explore the internal knowledge from LLMs and thus enhance the interactions with external environments. Current works such as Search-R1 [685], R1-Searcher [686], RAGEN [687], and OpenManus-RL [688] are exploring utilizing RL methods to fine-tune the agent models on trajectory data in agentic environments.
- Besides, hierarchical RL is also a promising topic that helps foundation model to decompose complex task and then learn optimal policies to solve each task via RL paradigm. For example, When2Ask [663] enables agents to request high-level instructions from LLM. The high-level LLM planner provides a plan of options, and the agent learns the low-level policy based on these options. Eureka [664] leverages LLM to generate human-level reward functions with reflection, allowing agents to efficiently learn complex tasks such as anthropomorphic five-finger manipulation. ArCHer [665] adopts a hierarchical RL approach, utilizing an off-policy RL algorithm to learn high-level value functions, which in turn implicitly guide the low-level policy. LLaRP [666] leverages LLM to comprehend both textual task goals and visual observations. It employs an additional action output module to convert the output of the LLM backbone into a distribution over the action space. Overall, using hierarchical RL can guide AI Agent to explore optimal strategies when analyzing user requests for reasoning and planning.

Using reinforcement learning, we can integrate foundation models with online learning from interactive environments, incorporating both action policies and world models. This integration enables advanced action systems in AI agents. Within the reinforcement learning paradigm, agents dynamically adapt and refine their decision-making processes in

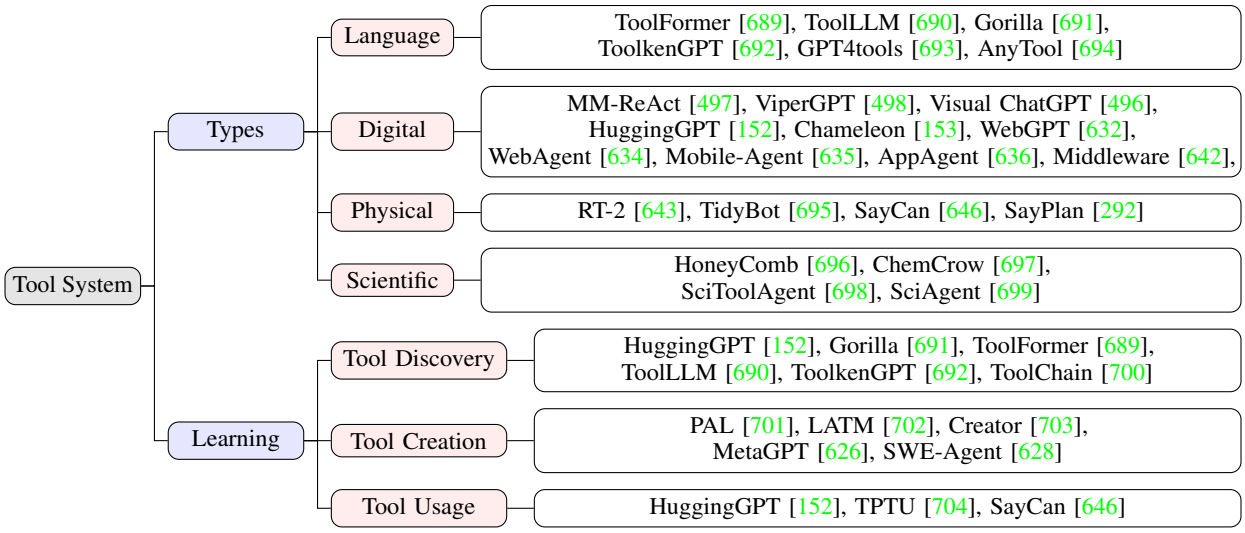


Figure 8.4: Illustrative Taxonomy of Tool Systems in AI Agents, including tool category and learning paradigm.

response to external feedback, facilitating greater efficiency and effectiveness in action learning and achieving desired outcomes.

Summary In general, Empowered by action systems, AI agents have demonstrated significant decision-making capabilities across various fields. For example, action learning enables AI agents to automate the understanding of Graphical User Interfaces (GUIs) and perform various operations, thereby improving human productivity through automatic computer usage. Moreover, several studies have shown that AI agents equipped with action systems can achieve remarkable outcomes in robotic manipulation tasks, such as object picking, laundry folding, and table cleaning. There are also promising research directions in the industry employing action models. For instance, autonomous driving (AD) has attracted considerable attention due to the exceptional performance of VLMs in perception and decision-making. By integrating human understanding through foundation models, AD systems can effectively comprehend real-world surrounding, enabling them to simulate human-level drivers. In summary, action learning endows agents with the ability to interact with the external world, thereby creating more opportunities for AI applications in real-world scenarios.

8.3.3 Tool-Based Action Paradigm

Tool learning distinguishes human intelligence from that of other animals. Ever since the Stone Age, human use of tools has boosted efficiency, productivity, and innovation. Similarly, enabling AI agents to operate in digital and physical environments by harnessing various tools is a fundamental step toward achieving human-level intelligence.

Definitions In AI, tools are defined as interfaces, instruments, or resources that allow agents to interact with the external world. Examples include web search [632, 705, 97, 634], databases [706, 707, 708, 709], coding environments [710], data systems [711, 712, 713], and weather forecasting [714]. By translating tool functionality into plain text or API formats, foundation models can expand their problem-solving scope. The evolution of tool systems in AI can be summarized in stages. Initially, with the advent of large language models [2], the focus was on converting tools into explainable formats (e.g., function calls). Later, advances in multimodal processing shifted interactions from conversational chats to graphical user interfaces (GUIs), and more recent work has explored embodied agents that control hardware (e.g. robotic arms, sensors) to interact with the physical world. To simplify, a tool-based action can be considered a form of external action employed for assistance.

Tool Category Similar to action spaces, tools can also be classified into multiple categories according to their types. In this part, we mainly summarize three key domains, including language, digital, and physical. In addition, we also explore the potential of tool learning in emerging areas such as scientific discovery:

- *Language:* To facilitate the use of external tools, we usually denote the tool as a kind of function call for foundation models, which usually encompasses task descriptions, tool parameters, and corresponding

outputs. This expression allows LLMs to understand when and how to use tools in AI agents. Specifically, ToolFormer [689] expands the capabilities of language models by integrating external tool spaces, including calculator, QA systems, search engine, translation, and calendar. ToolLLM [690] uses RapidAPI as the action space and then uses a depth-first search-based decision tree algorithm to determine the most suitable tool for solving tasks. Gorilla [691] is a fine-tuned LLM based on the tool documents and then can be used to write API calls. ToolkenGPT [692] is to optimize tool embeddings and then enable LLMs to retrieve tools from the fine-tuned tool embeddings. GPT4tools [693] and AnyTool [694] are also building self-instruct datasets and then fine-tune LLMs on them for tool usage. Generally, due to the impressive capability of LLMs, language-based tool utilization for AI agents has been studied, with its effectiveness validated in abundant works, ranging from plain text or function calls to code programming.

- *Digital:* With the success of LLMs in processing language information, many researchers are exploring extending the task scope of AI agents from the language to the digital domains (e.g., MultiModal, Web search, GUI, and so on). For example, MM-ReAct [497], ViperGPT [498], and Visual ChatGPT [496] employed LLMs as the controller and then used LLMs to select visual experts for solving different tasks. HuggingGPT [152] and Chameleon [153] use LLMs to first conduct reasoning and planning actions and then analyze which multimodal tools should be used for solving user instructions. WebGPT [632] and WebAgent [634] respectively empowered LLMs with search engines to enhance the capability of LLMs to solve more challenging tasks. Mobile-Agent [635] and AppAgent [636] respectively incorporate GUI manipulations and App usage as the tool-based actions to extend the task scope of AI agents in solving mobile phone tasks. In contrast to the physical world, digital environments usually provide simpler pipelines to collect and process data. By involving foundation models and their interaction with the digital environment, it is possible for us to develop intelligent assistants in computers, mobile phones, and other digital devices.

- *Physical:* For physical world applications, RT-2 [643] demonstrates language-guided robotic manipulation using visual-language tools, and TidyBot [695] shows how LLMs adapt cleaning tools to personalized household preferences. SayCan [646] uses LLMs as the cognitive system to guide robots in solving tasks through robotic arms and visual perception. SayPlan [292] built a 3D scene graph as the action spaces and designed multiple actions and tools for 3D simulation, and then used LLMs as planners to invoke these actions or tools for robot task planning. Besides, specialized applications in real-world scenarios now also proliferate across different domains. For instance, in surgical robotics, [715] presents a multi-modal LLM framework for robot-assisted blood suction that couples high-level task reasoning, enabling autonomous surgical sub-tasks. Some autonomous driving systems [716, 717] also integrate vision–language models with vehicle control tools for explainable navigation. In total, physical world applications pose the most significant challenge when compared to other tasks, but they also offer the biggest industrial value. Therefore, it still requires us to continue exploring advanced action learning and tool integration in physical-based agents in the future.

- *Scientific:* Scientific tools have played a transformative role in advancing AI agents across disciplines, enabling them to learn, adapt, and execute tasks while integrating foundational models with frameworks that drive innovation and address complex challenges. In materials science, HoneyComb [696] exemplifies tool-driven advancements with its ToolHub. General Tools provide dynamic access to real-time information and the latest publications, effectively bridging gaps in static knowledge bases. Material Science Tools are designed for computationally intensive tasks, leveraging a Python REPL environment to dynamically generate and execute code for precise numerical analysis. Similarly, ChemCrow [697] demonstrates the transformative power of tools in chemistry by integrating GPT-4 with 18 expert-designed tools to automate complex tasks such as organic synthesis, drug discovery, and materials design. These tools include OPSIN for IUPAC-to-structure conversion, calculators for precise numerical computations, and other specialized chemistry software that enables accurate reaction predictions and molecular property evaluations. Similarly, SciToolAgent [698] showcases how multi-tool integration can revolutionize scientific research. Designed to address the limitations of existing systems, SciToolAgent integrates over 500 tools (e.g., Web API, ML models, function calls, databases, and so on). Finally, SciAgent [699] exemplifies a multi-agent framework that integrates ontological knowledge graphs with specialized agents for hypothesis generation and critical analysis, emphasizing the power of modular, tool-driven systems to accelerate discovery in materials science and beyond. These examples underscore the transformative potential of integrating specialized tools into AI frameworks to address domain-specific challenges effectively.

Tool learning Inspired by human evolution [718], the integration of tools in AI involves three key aspects: *Tool Discovery* (identifying suitable tools), *Tool Creation* (developing new tools) and *Tool Usage* (effectively employing tools). We also systematically review existing literature and summarize them in the following:

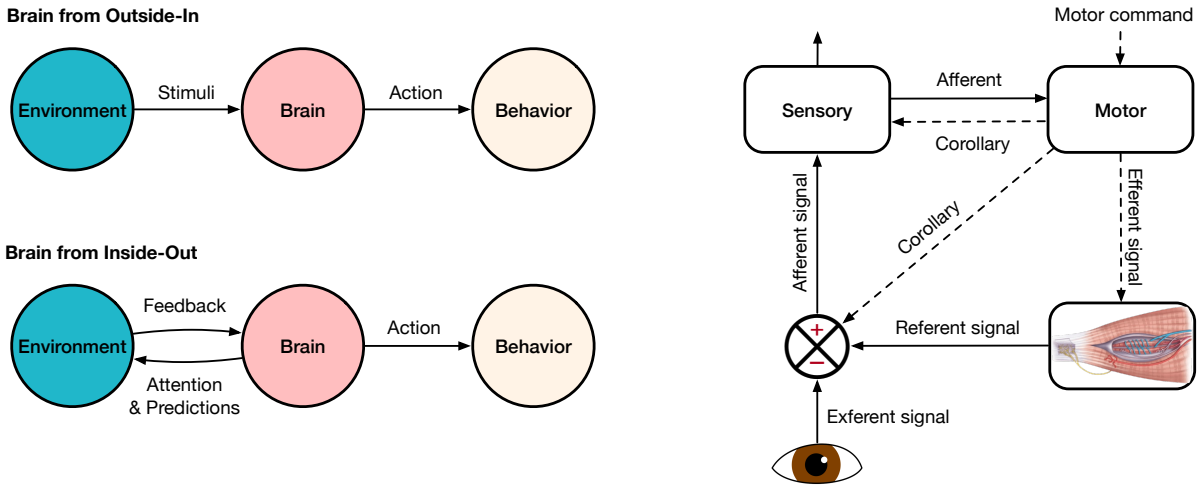
1. **Tool Discovery:** In real-world environments, there is a wide range of tools from the digital to the physical world. Finding the most appropriate tools for user instructions can be challenging. Therefore, the process of tool discovery is to identify and select the appropriate tools that AI agents can operate on to achieve their objectives. This stage also requires the world models in AI agents to have a profound understanding of any complex user instructions and world knowledge of different tools. Moreover, the versatility of AI agents is also correlated with its ability to operate diverse tool systems. Generally, tool discovery can be categorized into two mainstream paradigms: retrieval-based and generative-based methods. Retrieval-based methods aim to select the most relevant tools from the tool library. For example, HuggingGPT [152] introduces a framework in which LLMs act as controllers, orchestrating task planning and then invoking suitable models from platforms such as Hugging Face to fulfill user intention. In generative-based approaches, we often fine-tune LLMs to learn how to use and select tools based on various user instructions. For instance, ToolFormer [689] collects a massive corpus with the corresponding API calls (e.g., calculator, QA system, search engines, translation, and calendar) for training. ToolLLM [690] collect tool instructions based on solution paths and then fine-tune Llama models to generate better API calls for tool utilization.
2. **Tool Creation** In addition to using existing tools, the ability to create new tools plays a crucial role in human civilization. For language agents, a widely adopted approach is to use LLMs to generate functions as executable programs, which consist of both the code and documentation. For example, PAL [701] generates programs as intermediate reasoning steps to solve problems, LATM [702] or Creator [703] use LLMs to create code for user intentions, and to further design a verifier to validate the created tools. SciAgent [699] not only integrates multiple scientific tools but also crafts new tools for scientific discovery. More details on tool creation from an optimization perspective can be found in Section 9.4.2.
3. **Tool Usage** After collecting or creating tools, the effective use of tools constitutes the cornerstone of the capabilities of AI agents, allowing applications that bridge virtual and physical worlds. Modern AI agents increasingly employ tools to tackle complex tasks across diverse domains, with three key dimensions of expansion: 1) *Vertical Specialization*: Agents leverage domain-specific tools to achieve professional-grade performance in complex fields such as robotics, science, and healthcare; 2) *Horizontal Integration*: Systems combine multiple toolkits across modalities (vision, language, control) for multimodal problem-solving; 3) *Embodiment*: Agents physically interact with environments through robotic tools and sensors.

Summary Tool learning and action learning constitute the two most important components of the action system in AI agents. Tool learning can be considered as a kind of action to use external states for problem-solving. Tool learning enables AI agents to substantially broaden their range of tasks, pushing the boundaries beyond the scope of foundation models. For example, empowered by API or function calls, language models can directly reuse the capability of existing models (e.g., retrieval, coding, web search) to generate answers, rather than next-token prediction [719]. Tool learning also involves multiple challenging stages, including how to determine the tool space, how to discover and select tools, and how to create and use tools. Overall, tool learning plays a pivotal role in building an omnipotent AI agent framework to solve complex tasks in different domains.

8.4 Action and Perception: "Outside-In" or "Inside-out"

A central debate in cognitive science and neuroscience concerns whether action or perception stands at the root of causal flow in intelligent systems. Figure 8.5 presents different perspectives. The traditional "outside-in" view insists that causal influence begins with external stimuli. The environment excites peripheral receptors, these signals propagate inward, and eventually produce behavior. This perspective portrays the organism—or agent—as essentially reactive: the external world causes sensory changes, and the agent's actions represent a downstream effect of those changes. In contrast, Buzsáki's "inside-out" framework [18] proposes that it is the agent's own actions that shape the meaning and consequences of incoming signals. Such a view implies an active agent, one which continuously generates predictions and motor commands, while sending "corollary discharg" or "action copies" to sensory areas. These internally generated signals serve as references that inform the agent which sensory changes are self-initiated rather than imposed by the outside world. In this manner, cause shifts from an external event to an internally launched initiative, leaving external stimuli to play a confirmatory or corrective role. This reversal has significant implications for how we interpret perception's purpose and function: it is not an end in itself, but a means of updating and refining the agent's own action-driven hypotheses about the environment.

From an evolutionary perspective, possessing the ability to move without relying on sophisticated sensory analysis can yield immediate survival benefits. Even simple organisms profit from periodic motion that stirs up food in nutrient-rich water, long before elaborate perceptual capacities evolve. In other words, movement precedes advanced sensing in evolutionary time, suggesting that the capacity to act is not merely the effect of external stimuli but can



(a) Compare the brain from "Outside-In" and "Inside-Out"

(b) Schematics of the corollary discharge mechanism

Figure 8.5: (a) Compare the brain from "outside-in" and "inside-out". (b) Illustration of the schematic of the corollary discharge mechanism. A motor command (efferent signal) travels from motor areas to the eye muscles, while a corollary discharge (dashed arrow) is routed to a comparator in the sensory system. The comparator uses this internal signal to modulate or subtract external (exafferent) input. Additionally, tension feedback from the muscles (reafferent signal) exerts a delayed effect on perception. Direct projections from motor to sensory cortices underlie this architecture in all mammals. Part (b) is adapted from the original figure in [18].

itself be the driving cause of subsequent perceptual development. It is precisely when action mechanisms become sufficiently established that the agent benefits from additional sensors, which guide those movements more strategically. This developmental sequence grounds perception in utility, tying sensory discrimination to the practical outcomes of movement.

Disruptions in the normal interplay of action and perception illuminate the intricate cause-effect loop. During sleep paralysis, the brain's motor commands temporarily fail to reach the muscles; external stimuli still bombard the senses, but the usual action-to-perception calibration is lost. As a result, the individual experiences a heightened sense of unreality because the brain lacks internally generated reference signals to interpret sensory input. Similarly, if one externally manipulates the eye without the brain issuing a motor command, the visual scene appears to move, highlighting how perception alone—devoid of a preceding, self-initiated action—risks confusion. Neurophysiological data further support the inside-out model. Many neurons in areas once deemed "purely sensory" track not only changes in external stimuli but also self-generated movements—sometimes more strongly so. This indicates that "cause" in the brain frequently emerges from within, guiding both the magnitude and meaning of external signals. Without these internal correlates, raw sensory data can become ambiguous or even useless to the system.

Implications for Intelligent Agents The inside-out perspective offers potent insights for modern research on intelligent agents. Most contemporary AI systems—and many LLM agents—still function predominantly in a reactive mode, awaiting user input and generating responses based on statistical correlations learned from vast datasets. Such passivity resembles an "outside-in" framework, where the agent's role is limited to responding, not initiating. Yet if an agent were to be active, continuously forming and testing hypotheses via self-initiated behaviors (physical or representational), it might ground its own "perceptual" inputs—be they sensory streams or linguistic prompts—and thereby reduce ambiguity. For instance, an LLM-based agent that interjects questions or verifies its own statements against a knowledge base could better discern which inferences are self-caused from those demanded by external data. By tracking these self-initiated contributions (analogous to corollary discharge), the model could improve coherence, lessen errors known as "hallucinations", and refine its internal state through iterative cause-effect loops.

A proactive stance also encourages more data-efficient and context-aware learning. Instead of passively waiting for labeled examples, an agent can explore, provoke feedback, and incorporate self-generated experiences into its training. Over time, this tight coupling between action and perception may bolster the agent's ability to handle complex tasks, adapt to unanticipated challenges, and generalize more robustly. The shift from an outside-in to an inside-out model reframes perception as causally downstream of action. Intelligent systems—whether biological or artificial—stand

Table 8.2: Comparing the perception and action of human and AI agents.

Dimension	Human Brain / Cognition	LLM Agent	Remarks
Perception	- Integrates multiple sensory channels (vision, hearing, smell, touch, taste). - Perception closely tied to emotions, endocrine system, and physical state. - Highly sensitive, capable of detecting subtle differences.	- Primarily language-based with some multimodal capabilities. - Perception depends on external sensors and models with limited integration. - Lacks real-time coupling with physical states.	Perception differences lead to varying ways of understanding reality. Embodied AI attempts to bridge this gap but still faces both hardware and software challenges.
Unified Representation	- Simultaneously processes multimodal inputs: vision, hearing, language, motion, and emotions. - Different brain regions collaborate to create unified spatiotemporal and semantic understanding.	- Primarily text-based. Some multimodal models can process images or audio but with low integration. - No fully unified spatiotemporal modeling like the human brain.	Even advanced multimodal models lack the human brain's holistic, unified representation capacity. Hardware and algorithmic challenges remain.
Granularity in Task Switching	- Flexible in shifting between macro and micro cognitive tasks. - Can plan at a high level and shift focus to finer details when needed. - Adjusts task priority and focus dynamically based on context and working memory.	- Relies heavily on prompt engineering for granularity control. - Cannot autonomously reallocate attention between task layers. - May get stuck in a specific level of abstraction in absence of guided prompts.	Humans can dynamically adjust cognitive granularity based on situational demands, while LLMs require explicit instruction to switch task focus effectively.
Action	- Goal-oriented process drives multiple sensory to make decisions. - Real-time Learning from the experience via the environmental interaction. - Encompass both physical activities and mental processes.	- Action space need to be defined in advance. - Unable to support actions in continuous spaces. - Relies on online training to optimize the decision-making process in the environment.	Humans are capable of actively learning new actions and performing continuous actions, whereas LLM agents currently lack this capability.

to benefit from recognizing that purposeful movement, or proactive conversational steps in the case of LLMs, can actively create, shape, and interpret the signals that flow back in. By acknowledging the cause-effect power of action and striving to build active rather than merely reactive agents, we may approach a deeper understanding of both natural cognition and the next generation of AI.

8.5 Summary and Discussion

Traditionally, action represents the behaviors of the human cognition system based on the interactive feedback from the environment. It endows humans with the capability to think, reason, speak, run, and perform any complex manipulations. Based on the action system, humans can iteratively evolve the brain intelligence by enhancing their perception and actions from the world, and form a closed loop to further create new civilization and innovation in the world. Similarly to a human cognition system, the action system plus the tool system also play an important role for AI agents. Integrating action systems allows AI agents to systematically plan, execute, and adjust their behaviors, facilitating more adaptable and robust performance in dynamic contexts. In this section, we systematically examine and summarize the impact of the action module on AI agents, focusing on both action systems and tool systems.

Action System In our studies, we briefly describe the action system from three perspectives: action space, action learning, and tool learning. In an action system, action space usually serves as the most important component, which determines the upper bound of AI agents in solving downstream tasks. It formulates which actions can be selected and performed by AI agents during interactions with real-world environments. For action space, there are also various difficulties depending on data types, ranging from discrete to continuous data. With the growing demand for AI agents, there is also a rising expectation for AI agents to handle more sophisticated tasks, particularly those involving real-world applications. Therefore, how to build robust and general action space is still an ongoing challenge in action systems. On the basis of action space, action learning is another crucial component in enabling agents to interact effectively with the external world and with humans. Action learning represents the process of an AI agent to learn and optimize its policy during interaction with real-world environments. Based on different foundation models, it also derives different action learning paradigms, from zero-shot learning (e.g., prompt engineering) to supervised training and reinforcement learning. In action learning, it is essential to thoroughly understand the task, including how to devise system prompts, how to determine the pre-trained or fine-tuned datasets, and the reward signals or optimization polices during the training. Despite notable progress in action learning to advance AI agent frameworks, numerous questions remain to be addressed. Specifically, the ICL paradigm requires specific prior knowledge for a proper prompt design. Additionally, combining pre-training and post-training for supervised training necessitates high-quality and diverse data, which often requires meticulous data processing and significant human effort. Furthermore, the unstable nature of reinforcement learning poses difficulties in its application in large-scale training scenarios. Moreover, the design of action systems plays a crucial role in maximizing the benefits of tool integration. By incorporating an effective action system, AI agents can seamlessly engage with various tools, execute complex user intents, and transform external data into meaningful outcomes. This synergy between action systems and tools not only mitigates the limitations of memorization and reduces the risk of hallucinations [714] but also enhances the expertise and robustness of the system. For instance, an AI agent equipped with a robust action system can dynamically select and employ the most appropriate tools for a given task, ensuring both accuracy and efficiency in its responses. Furthermore, action systems facilitate hierarchical reasoning processes, enabling agents to orchestrate intricate workflows that align closely with user objectives. This alignment is essential for tasks requiring precise execution and real-time decision-making, thereby bridging the gap between foundational model capabilities and practical application demands. Additionally, the transparency and interpretability provided by tool execution processes enhance user trust and facilitate effective human-machine collaboration. Consequently, the combination of specialized tools and robust action systems significantly elevates the performance, reliability, and applicability of AI agents in diverse and dynamic environments.

In summary, action systems can significantly establish the foundation for the problem-solving capabilities of AI agent frameworks, enabling them to tackle a broader range of complex tasks beyond foundation models.

Future Directions Nonetheless, building an effective action system for agents requires solutions to a number of challenges, as we summarize in the following:

1. **Efficiency** presents a significant hurdle, particularly in real-time applications where swift and precise responses are critical. The complexity involved in action system can lead to unacceptable latency, hindering the practical deployment of AI systems in scenarios like fraud detection or real-time decision-making. To mitigate these efficiency issues, strategies such as filtering out irrelevant or redundant information, employing zero-shot prompting to streamline reasoning processes, and utilizing high-speed storage solutions for caching pertinent knowledge are imperative. These approaches help in maintaining high performance while reducing response times.

2. **Evaluation** is also a important factor in action system, including action learning and tool learning. In the real-world environment, there exists massive actions from different sources. Therefore, how to determine the correct action or tools from disparate sources to avoid conflicting information is still a significant challenge in AI Agent. To alleviate these problems, how to build an effective and robust evaluation system to measure action system is essential to maintain the accuracy and reliability of responses. Developing robust evaluation system, verification protocols and creating transparent methods are crucial to reduce incorrectness in action prediction. Besides, exposing the decision-making processes of foundation models also help us understand which action is better and how to coordinate with various actions or tools to provide trustworthy outputs.

3. **Multi-modality** Action learning has achieve remarkable progresses in LLM-based autonomous agent, due to the success of large language models. However, how to understand and invoke action beyond the language instructions (e.g., GUI operations or embodied tools) still remain challenges. In real-world scenarios, humans can develop or learn to use new skills through any kinds of instructions (e.g., language, image, videos or human guidance). Therefore, enabling AI agents to develop or learn actions through diverse modalities is crucial to advance the capability of AI Agent in solving practical tasks from the real-world scenarios. In other

words, it is necessary for us to explore how to reduce the gap between human and AI agents in tool utilization, facilitating the design of advanced agent frameworks for the future.

4. **Privacy** is a critical concern in the field of generative AI, especially using LLMs. As a consequence, maintaining the privacy of sensitive user data and preventing the disclosure of user behaviors are essential in tool utilization [720]. To address these privacy concerns, some safe techniques like federated learning can be used to enable models to be trained on decentralized data sources without exposing sensitive information directly. Additionally, model distillation is often necessary to ensure models maintain high performance while safeguarding data integrity. These methods enable the effective training of models while preserving the confidentiality of user data.
5. **Safety** Moreover, the ethical implications of human-model collaboration and the safety concerns associated with models interacting with physical environments necessitate careful consideration. Ensuring that human dignity and agency are preserved when integrating human labor with AI systems is critical. Establishing ethical guidelines, promoting fair working conditions, and fostering interdisciplinary collaboration are necessary to address these concerns. Additionally, developing robust safety mechanisms to prevent erroneous or malicious actions by AI systems interacting with physical tools or actions is imperative to safeguard against potential risks.

In addition to the above challenges, there also remain open problems for the action system. For example, how to achieve an optimal balance between the foundation models and external tools, deciding on the appropriate timing to use the former versus the latter, remains unanswered. Specifically, although tool systems can offer flexibility and extensibility for foundation models, there is an increasing trend to enhance the intrinsic capability of foundation models. Therefore, balancing between foundation models and tool systems is essential for developing versatile and efficient AI agents.

Part II

Self-Evolution in Intelligent Agents

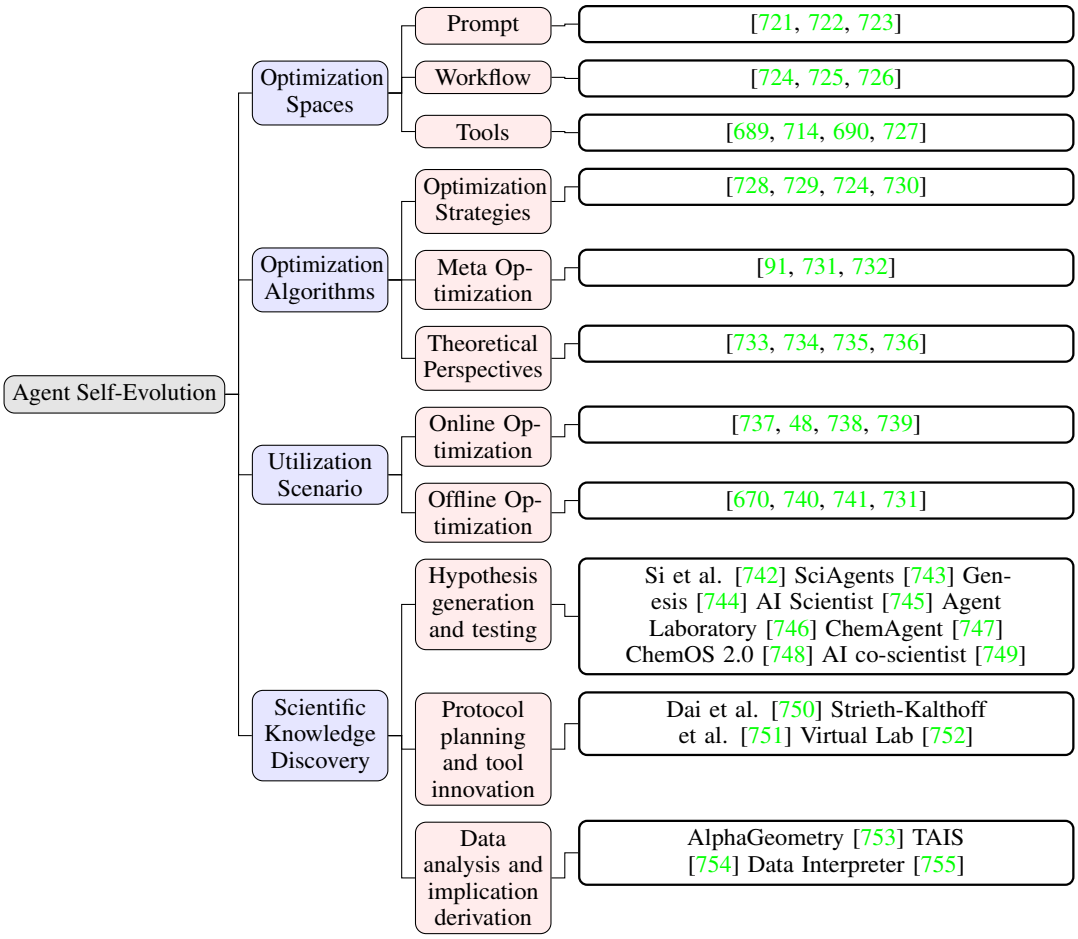


Figure 8.6: Structures of self-evolution in LLM agents.

In the history of machine learning research, manually designed AI systems have gradually been replaced by more efficient, learned solutions [756]. For instance, before the advent of deep learning, features were typically handcrafted by experts [757, 758], but these were eventually superseded by features extracted through neural networks. As neural networks have become increasingly complex, various techniques for automated design–such as neural architecture search–have emerged, further replacing the need for manually designed network structures [759]. Similarly, Agentic systems initially relied heavily on manual design, with behavior rules and decision-making strategies explicitly crafted by developers. Although full automation of agent self-evolution has not yet been achieved, it is anticipated and deemed necessary for future progress. A successful precedent for such automation can already be seen in automated machine learning (AutoML) [712, 760, 761, 762, 204] which has automated various components of traditional machine learning pipelines. In particular, AutoML streamlines the selection and configuration of machine learning algorithm pipelines while incorporating advanced techniques for hyperparameter optimization [763, 764, 765, 766, 767]. Among the most notable applications of AutoML is NAS [768, 769], which automates the design of neural network architectures to enhance model performance. Drawing inspiration from this successful transition towards automation in traditional machine learning, we propose extending similar principles to the domain of agentic AI systems.

A key counterintuitive issue in much of current agent research is that, while the ultimate goal of developing or improving agentic AI systems is to automate human efforts, the process of creating these systems remains, for the time being, beyond the reach of full automation. Therefore, we argue that all manually designed agentic AI systems will eventually be replaced by learnable and self-evolving systems, which could ultimately place the development and improvement of agentic AI into an autonomous, self-sustaining loop. Enabling self-evolution mechanism in LLM agents has the following benefits:

1. **Scalability:** While LLM-based agents have demonstrated remarkable performance, their improvement still heavily depends on the underlying LLMs. However, upgrading these models is costly, and scaling performance through the inclusion of additional real-world data requires extensive retraining on large datasets, which

poses significant resource constraints. Self-evolving agentic systems, in contrast, can optimize agent behavior without necessitating modifications to the underlying LLMs, offering a more efficient and scalable solution.

2. **Reduction in Labor Costs:** Manually designing agentic systems is a complex and labor-intensive process that requires developers to engage deeply with intricate technical details. Traditional methods often involve building these systems from scratch, demanding significant expertise and effort. By contrast, self-evolving agentic systems can automate much of this process, significantly reducing the need for manual intervention and lowering development costs.
3. **Aligned with Natural Intelligence Development:** Just as humans continuously improve themselves through learning and adaptation, equipping LLM agents with self-improvement capabilities is a necessary step toward the development of truly autonomous agents. This enables them to refine their performance, adapt to new challenges, and evolve without direct human intervention.

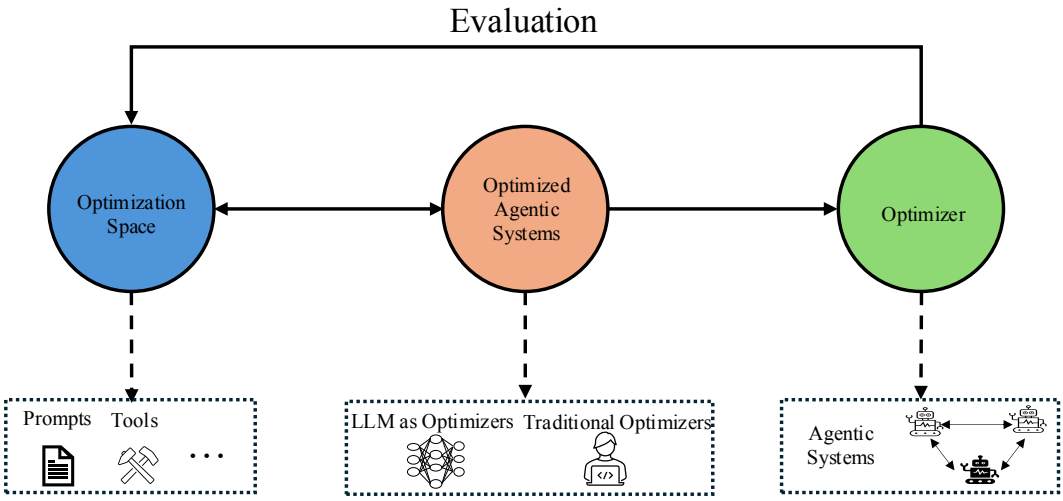


Figure 8.7: An illustration of key concepts discussed in this section, including optimization spaces, the optimizer, and the optimizing objective. The optimizer iteratively refines components within the optimization spaces to enhance agentic systems until a satisfactory outcome is achieved, thereby achieving self-improvement in the LLM agent systems.

To achieve the goal of automating human efforts, numerous studies have proposed leveraging LLMs as the driving engine to enable self-evolution in agentic systems. In particular, LLMs provide an efficient alternative to traditional optimization methods, such as gradient-based [770] and reinforcement learning-based approaches [771]. They extend the optimization space from numerical values to more diverse domains, with natural language serving as a universal bridge. An LLM is capable of optimizing complex, heterogeneous parameters, such as instructions [732] and tool implementations [772], and can operate across a range of LLMs, including both open-source and closed-source models. A notable example of this approach is AFLOW [773], which automates the generation and optimization of entire agentic system workflows. This system employs Monte Carlo Tree Search to leverage the comprehensive capabilities of LLMs. In this framework, traditionally handcrafted agentic systems are replaced by algorithmically generated ones, marking a kind of paradigm shift. Additionally, a growing body of research explores similar methodologies, further advancing the field.

This part is structured as follows: First, we introduce various optimization spaces explored in recent research on agentic systems, including prompts, tools, and workflows. In the subsequent section, we review optimization algorithms, discussing both traditional optimization paradigms and meta-optimization, where the optimization process also affects the underlying optimization algorithms themselves. We then explore the self-evolution scenarios, categorizing them into two types: online optimization and offline optimization. Following this, we discuss the application of large language model (LLM) agent self-improvement techniques, particularly in knowledge discovery within the AI-for-science domain. Finally, we discuss the security concerns associated with agent self-evolution technologies.

Chapter 9

Optimization Spaces and Dimensions for Self-evolution

The optimization of autonomous agents represents a complex challenge that encompasses multiple levels of abstraction. In this section, we first establish prompt optimization as the foundational layer, upon which three distinct branches of optimization emerge: agentic workflow optimization, tool optimization, and comprehensively autonomous agent optimization.

9.1 Overview of Agent Optimization

Existing LLM-based agent optimization can be conceptualized in terms of a two-tiered architecture. At the foundation lies *prompt optimization*, which focuses on enhancing the basic interaction patterns of Language Model nodes. Building upon this foundation, three parallel branches emerge: i) *workflow-level optimization*, which focuses on the coordination and interaction patterns between multiple LLM nodes; ii) *tool optimization*, where agents evolve by developing and improving tools to adapt to new tasks and leverage past data; and iii) *comprehensive autonomous agent optimization*, which aims at the holistic enhancement of agent capabilities by considering multiple dimensions.

Similarly to optimization paradigms in AutoML, agent optimization can be categorized as either single-objective or multi-objective. Contemporary agent optimization primarily centers on three canonical metrics: performance, inference cost, and latency. Performance measures the effectiveness of the agent in completing its assigned tasks, while inference cost quantifies the computational resources required for agent operation. Latency represents the time taken for the agent to respond and complete tasks. These objectives can vary depending on the specific optimization modality. For instance, in prompt-level optimization, additional constraints such as prompt length may become relevant objectives. This multi-faceted nature of optimization objectives reflects the complexity of agent systems and the need to balance multiple competing requirements.

9.2 Prompt Optimization

Prompt optimization plays the most critical role in LLM-based agent optimization. When optimizing an agent, beyond model-level optimizations, task-specific or model-specific prompt optimization directly impacts the agent's performance, latency, and cost. Given a task $T = (Q, G_t)$, where Q denotes the input query and G_t represents the optional ground truth, the objective of prompt optimization is to generate a task-specific prompt P_t^* that maximizes performance:

$$P^* = \arg\max_{P \in \mathcal{P}} \mathbb{E}_{T \sim \mathcal{D}}[\phi_{\text{eval}}(\phi_{\text{exe}}(Q, P), T)] \tag{9.1}$$

where $\mathcal{P}$ represents the space of possible prompts, ϕ_{exe} denotes the execution function, and ϕ_{eval} represents the evaluation function. This optimization is typically implemented through three fundamental functions: ϕ_{opt}, ϕ_{exe}, and ϕ_{eval}. The Optimize function ϕ_{opt} refines existing prompts based on optimization signals, the Execute function ϕ_{exe} invokes the current prompt to obtain output O, and the Evaluation function ϕ_{eval} assesses current outputs to generate evaluation signals S_{eval} and optimization signals S_{opt}. The evaluation signals are used to select effective prompts, while the optimization signals assist the Optimize function in performing optimization.

9.2.1 Evaluation Functions

At the core of prompt optimization lies the evaluation function ϕ_{eval}, which serves as the cornerstone for deriving optimization signals and guiding the evolutionary trajectory of prompts. This function orchestrates a sophisticated interplay between evaluation sources, methodologies, and signal generation, establishing a feedback loop that drives continuous improvement. The evaluation function ϕ_{eval} processes evaluation sources as input, and employs various evaluation methods to generate different types of signals, which subsequently guide the optimization process. Here, we define the dimensions of sources, methods, and signal types to establish the foundation for prompt optimization.

Evaluation Sources Evaluation sources primarily consist of LLM Generated Output G_{llm} and task-specific Ground Truth G_t. Existing works such as [730, 774, 728, 775, 732, 300] predominantly leverage comparisons between G_{llm} and G_t as evaluation sources. Some approaches [776, 721, 777] utilize only G_{llm} as the evaluation source. For instance, PROMST [721] assesses prompt effectiveness by comparing G_{llm} against human-crafted rules; SPO [778] employs pairwise comparisons of outputs from different prompts to determine relative effectiveness.

Evaluation Methods Evaluation Methods can be broadly categorized into three approaches: *benchmark-based evaluation*, *LLM-as-a-Judge*, and *human feedback*. *Benchmark-based evaluation* remains the most prevalent method in prompt optimization [730, 774, 721, 732, 300]. This approach relies on predefined metrics or rules to provide numerical feedback as evaluation signals. While it offers an automated evaluation process, its effectiveness ultimately depends on how well the benchmark design aligns with human preferences.

The introduction of *LLM-as-a-Judge* represents a significant advancement in automated evaluation and preference alignment. Leveraging LLMs' inherent alignment with human preferences and carefully designed judging criteria, this approach [589] can assess task completion quality based on task descriptions and prompt outputs G_{llm}, providing reflective textual gradient feedback. Notable implementations include ProteGi [779], TextGrad [728], Semantic Search [775] and Revolve [780]. Furthermore, LLM-as-a-judge enables comparative evaluation between ground truth G_t and output G_{llm} with specific scoring mechanisms [724]. The effectiveness of this method hinges on both the design of judger prompts and the underlying model's alignment with human preferences. As a specialized extension, *Agent-as-a-Judge* [781] refines this paradigm by employing dedicated agents for providing process evaluation on complex tasks, while maintaining high alignment with human preferences at significantly reduced evaluation costs.

Human feedback represents the highest level of intelligence integration in the evaluation process. As humans remain the ultimate arbiters of prompt effectiveness, direct human feedback can rapidly and substantially improve prompt quality. However, this approach introduces significant resource overhead. APOHF [777] demonstrates that incorporating human feedback can achieve robust prompt optimization with minimal computational resources, particularly excelling in open-ended tasks such as user instructions, prompt optimization for text-to-image generative models, and creative writing. Nevertheless, the requirement for human intervention somewhat contradicts the goal of automated evolution.

Signal Types Feedback generated by evaluation methods manifests in three distinct forms, each serving different optimization needs. *Numerical feedback* [730, 774, 721, 732, 300] quantifies performance through scalar metrics, compatible with rules, ground truth, human assessment, and LLM judgments. While widely applicable, this approach requires substantial samples for statistical reliability, potentially overlooking instance-specific details that could guide optimization. *Textual feedback* [728, 775, 780] provides detailed, instance-specific guidance through analysis and concrete suggestions. This sophisticated approach requires intelligent participation, either from human experts or advanced language models, enabling targeted improvements in prompt design through explicit recommendations. However, its reliance on sophisticated intelligence sources impacts its scalability.*Ranking feedback* [778] establishes relative quality ordering through either comprehensive ranking or pairwise comparisons. This approach uniquely circumvents the need for absolute quality measures or predefined criteria, requiring only preference judgments. It proves particularly valuable when absolute metrics are difficult to define or when optimization primarily concerns relative improvements.

9.2.2 Optimization Functions

The design of optimization functions is crucial in determining the quality of generated prompts in each iteration of prompt optimization. Through effective signal guidance, prompt self-evolution can achieve faster convergence. Current optimization approaches primarily rely on two types of signals: *evaluation signals* S_{eval} that identify the most effective existing prompts, and *optimization signals* S_{opt} that provide detailed guidance for improvements.

Optimize via Evaluation Signals When optimizing with evaluation signals, the process begins by selecting the most effective prompts based on ϕ_{eval} assessments. Rather than directly learning from past errors, some methods adopt

heuristic exploration and optimization strategies. SPO [778] iteratively refines prompts based on the outputs of current best-performing ones, leveraging the language model's inherent ability to align with task requirements. Similarly, Evoprompt [723] employs evolutionary algorithms with LLMs serving as evolution operators for heuristic prompt combination. PromptBreeder [732] advances this approach further by comparing score variations between mutated prompts while simultaneously modifying both meta-prompts and prompts through the LLM's inherent capabilities.

Optimize via Optimization Signals While optimization methods based solely on evaluation signals require extensive search to find optimal solutions in vast search spaces through trial and error, an alternative approach leverages explicit optimization signals to guide the optimization direction and improve efficiency. Existing methods demonstrate various ways to utilize these optimization signals. OPRO [730] extracts common patterns from high-performing prompt solutions to guide subsequent optimization steps. ProTegi [779] employs language models to analyze failure cases and predict error causes, using these insights as optimization guidance. TextGrad [728] extends this approach further by transforming prompt reflections into "textual gradients", applying this guidance across multiple prompts within agentic systems. Revolve [780] further enhances optimization by simulating second-order optimization, extending previous first-order feedback mechanisms to model the evolving relationship between consecutive prompts and responses. This allows the system to adjust based on how previous gradients change, avoiding stagnation in suboptimal patterns and enabling more informed, long-term improvements in complex task performance.

9.2.3 Evaluation Metrics

The effectiveness of prompt optimization methods can be evaluated across multiple dimensions. *Performance metrics* [782, 778, 730] for Close Tasks serve as the most direct indicators of a prompt's inherent performance, encompassing measures such as pass@1, accuracy, F1 score, and ROUGE-L. These metrics enable researchers to assess the stability, effectiveness, and convergence rate of prompt optimization processes. Another crucial dimension is *Efficiency metrics* [778]. While some prompt optimization approaches achieve outstanding results, they often demand substantial computational resources, larger sample sizes, and extensive datasets. In contrast, other methods achieve moderate results with lower resource requirements, highlighting the trade-offs between performance and efficiency in agent evolution. The third dimension focuses on qualitative metrics that assess specific aspects of agent behavior: consistency [776] measures output stability across multiple runs, fairness [783] evaluates the ability to mitigate the language model's inherent biases, and confidence [784, 785] quantifies the agent's certainty in its predictions. When these behavioral aspects are treated as distinct objectives, prompt optimization frameworks provide corresponding metrics for evaluation.

9.3 Workflow Optimization

While prompt-level optimization has shown promising results in enhancing individual LLM capabilities, modern AI systems often require the coordination of multiple LLM components to tackle complex tasks. This necessitates a more comprehensive optimization domain—the agentic workflow space. At its core, an agentic workflow consists of LLM-invoking nodes, where each node represents a specialized LLM component designed for specific sub-tasks within the larger system.

Although this architecture bears similarities to multi-agent systems, it is important to distinguish agentic workflows from fully autonomous multi-agent scenarios. In agentic workflows, nodes operate under predetermined protocols and optimization objectives, rather than exhibiting autonomous decision-making capabilities. Many prominent systems, such as MetaGPT [626] AlphaCodium [786] can be categorized under this framework. Moreover, agentic workflows can serve as executable components within larger autonomous agent systems, making their optimization crucial for advancing both specialized task completion and general agent capabilities.

Following the formalization proposed by GPTSwarm [651] and AFLOW [773], this section first establishes a formal definition of agentic workflows and their optimization objectives. We then examine the core components of agentic workflows—nodes and edges—analyzing their respective search spaces and discussing existing representation approaches in the literature.

9.3.1 Workflow Formulation

An agentic workflow $\mathcal{K}$ can be formally represented as:

$$\mathcal{K} = \{(N, E) | N \in \mathcal{N}, E \in \mathcal{E}\} \tag{9.2}$$

where $\mathcal{N} = \{N(M, \tau, P, F) | M \in \mathcal{M}, \tau \in [0, 1], P \in \mathcal{P}, F \in \mathcal{F}\}$ represents the set of LLM-invoking nodes, with M, τ, $\mathcal{P}$, and $\mathcal{F}$ denoting the available language models, temperature parameter, prompt space, and output format space respectively. E indicates the edges between different LLM-invoking nodes. This formulation encapsulates both the structural components and operational parameters that define an agentic workflow's behavior.

Given a task T and evaluation metrics L, the goal of workflow optimization is to discover the optimal workflow K^* that maximizes performance:

$$K^* = \arg\max_{K \in \mathcal{K}} L(K, T) \tag{9.3}$$

where K is the search space of workflow, and $L(K, T)$ typically measures multiple aspects including task completion quality, computational efficiency, and execution latency. This optimization objective reflects the practical challenges in deploying agentic workflows, where we must balance effectiveness with resource constraints.

9.3.2 Optimizing Workflow Edges

The edge space $\mathcal{E}$ defines the representation formalism for agentic workflows. Current approaches primarily adopt three distinct representation paradigms: graph-based, neural network-based, and code-based structures. Each paradigm offers unique advantages and introduces specific constraints on the optimization process.

Graph-based representations enable the expression of hierarchical, sequential, and parallel relationships between nodes. This approach naturally accommodates complex branching patterns and facilitates visualization of workflow topology, making it particularly suitable for scenarios requiring explicit structural manipulation. For example, GPTSwarm [651] demonstrated the effectiveness of graph-based workflow representation in coordinating multiple LLM components through topology-aware optimization. Neural network architectures provide another powerful representation paradigm that excels in capturing non-linear relationships between nodes. Dylan [725] showed that neural network-based workflows can exhibit adaptive behavior through learnable parameters, making them especially effective for scenarios requiring dynamic adjustment based on input and feedback. Code-based representation offers the most comprehensive expressiveness among current approaches. AFLOW [773] and ADAS [741] established that representing workflows as executable code supports linear sequences, conditional logic, loops, and the integration of both graph and network structures. This approach provides precise control over workflow execution and leverages LLMs' inherent code generation capabilities.

The choice of edge space representation significantly influences both the search space dimensionality and the applicable optimization algorithms. [728] focused solely on prompt optimization while maintaining a fixed workflow topology, enabling the use of textual feedback-based optimization techniques. In contrast, [651] developed reinforcement learning algorithms for joint optimization of individual node prompts and overall topology. [773] leveraged code-based representation to enable direct workflow optimization by language models, while recent advances by [787] and [788] introduced methods for problem-specific topology optimization.

9.3.3 Optimizing Workflow Nodes

The node space N consists of four key dimensions that influence node behavior and performance. The output format space F significantly impacts performance by structuring LLM outputs, with formats like XML and JSON enabling more precise control over response structure. The temperature parameter τ controls output randomness, affecting the stability-creativity tradeoff in node responses. The prompt space P inherits the optimization domain from prompt-level optimization, determining the core interaction patterns with LLMs. The model space M represents available LLMs, each with distinct capabilities and computational costs.

For single-node optimization, existing research has primarily focused on specific dimensions within this space. [773] concentrated exclusively on prompt optimization, while [741] extended the search space to include both prompts and temperature parameters. Taking a different approach, [789] fixed prompts while exploring model selection across different nodes. Output format optimization, though crucial, remains relatively unexplored [790].

Compared to edge space optimization, node space optimization poses unique scalability challenges due to the typically large number of nodes in agentic workflows. The dimensionality of the search space grows multiplicatively with each additional node, necessitating efficient optimization strategies that can effectively handle this complexity while maintaining reasonable computational costs.

9.4 Tool Optimization

Unlike conventional usage of LLMs that typically operate in a single-turn manner, agents are equipped with advanced multi-turn planning capabilities and the ability to interact with the external world via various tools. These unique attributes make the optimization of tool usage a critical component in enhancing an agent's overall performance and adaptability. Tool optimization involves systematically evaluating and refining how an agent selects, invokes, and integrates available tools to solve problems with higher efficiency and lower latency. Key performance metrics in this context include decision-making accuracy, retrieval efficiency, selection precision, task planning, and risk management. Central to this optimization are two complementary strategies: *tool learning* and *tool creation*.

9.4.1 Learning to Use Tools

Unlike prompting-based methods that leverage frozen foundation models' in-context learning abilities, training-based methods optimize the model that backs LLM agents with supervision. Drawing inspiration from developmental psychology, tool learning can be categorized into two primary streams: *learning from demonstrations* and *learning from feedback* [714]. The other way to elicit the power of LLMs (agents) using tools is by using prompt-based or in-context learning methods for better reasoning abilities.

Learning from demonstrations involves training models backed LLM agents to mimic expert behaviors through imitation learning. Techniques such as behavior cloning allow models to learn policies in a supervised manner by replicating human-annotated tool-use actions. Formally, given a dataset $D = \{(q_i, a_i^*)\}_{i=0}^{N-1}$, where q_i is a user query and a_i^* is the corresponding human demonstration, the controller's parameters θ_C are optimized as:

$$\theta_C^* = \arg\max_{\theta_C} \mathbb{E}_{(q_i, a_i^*) \in D} \prod_{t=0}^{T_i} p_{\theta_C}(a_{i,t}^* \mid x_{i,t}, H_{i,t}, q_i)$$

where $a_{i,t}^*$ is the human annotation at timestep t for query q_i, and T_i is the total number of timesteps.

Learning from feedback leverages reinforcement learning to enable models to adapt based on rewards derived from environment or human feedback. The optimization objective for the controller's parameters θ_C is:

$$\theta_C^* = \arg\max_{\theta_C} \mathbb{E}_{q_i \in Q} \mathbb{E}_{\{a_{i,t}\}_{t=0}^{T_i}} \left[R\left(\{a_{i,t}\}_{t=0}^{T_i} \right) \right]$$

where R represents the reward function based on the sequence of actions $\{a_{i,t}\}$.

Integrating tool learning into the optimization framework enhances the system's ability to generalize tool usage across diverse tasks and environments. By incorporating both demonstration-based and feedback-based learning, the model can iteratively improve its tool invocation strategies, selection policies, and execution accuracy.

Optimization Reasoning Strategies for Tool Using Optimizing the aforementioned metrics for better LLM agents' abilities requires a combination of advanced retrieval models, fine-tuned reasoning strategies, and adaptive learning mechanisms. Reasoning strategies, such as Chain-of-Thought (CoT) [46], Tree-of-Thought [72], and Depth-First Search Decision Trees (DFS-DT) [690], facilitate more sophisticated decision-making processes regarding tool usage. Fine-tuning the model's understanding of tools, including parameter interpretation and action execution, enables more precise and effective tool interactions. Additionally, learning from the model's outputs allows for better post-processing and analysis, further refining tool utilization efficacy.

9.4.2 Creation of New Tools

Beyond the optimization of existing tools, the ability to create new tools dynamically [703, 702, 772] based on a deep understanding of tasks and current tool usage can significantly enhance the LLM Agent framework's adaptability and efficiency. In recent work, several complementary approaches have been proposed. ToolMakers [702] establishes a closed-loop framework where a tool-making agent iteratively executes three phases: (1) *Proposing* Python functions via programming-by-example using three demonstrations, (2) *Verifying* functionality through automated unit testing (3 validation samples) with self-debugging of test cases, and (3) *Wrapping* validated tools with usage demonstrations for downstream tasks. This rigorous process ensures reliability while maintaining full automation. CREATOR [703] adopts a four-stage lifecycle: *Creation* of task-specific tools through abstract reasoning, *Decision* planning for tool invocation, *Execution* of generated programs, and *Rectification* through iterative tool refinement—emphasizing tool diversity, separation of abstract/concrete reasoning, and error recovery mechanisms. In contrast, CRAFT [772] employs an offline paradigm that distills domain-specific data into reusable, atomic tools (e.g., object color detection) through GPT-4 prompting, validation, and deduplication. Its training-free approach combines human-inspectable code snippets

with compositional problem-solving, enabling explainable toolchains while avoiding model fine-tuning—particularly effective when decomposing complex tasks into modular steps.

The integration of these complementary approaches presents rich research opportunities. Hybrid systems could merge CRAFT's pre-made tool repositories with ToolMakers' on-demand generation, using functional caching to balance efficiency and adaptability. Future frameworks might implement multi-tier tool hierarchies where primitive operations from CRAFT feed into ToolMakers' composite tools, while CREATOR-style rectification handles edge cases. Advances in self-supervised tool evaluation metrics and cross-domain generalization could further automate the tool lifecycle. Notably, the interplay between tool granularity (atomic vs. composite) and reusability patterns warrants systematic investigation—fine-grained tools enable flexible composition but increase orchestration complexity. As agents evolve, bidirectional tool-task co-adaptation mechanisms may emerge, where tools reshape task representations while novel tasks drive tool innovation, ultimately enabling self-improving AI systems.

9.4.3 Evaluation of Tool Effectiveness

The evaluation metrics and benchmarks discussed below offer a comprehensive basis for quantifying an agent's tool usage capabilities. By assessing aspects such as tool invocation, selection accuracy, retrieval efficiency, and planning for complex tasks, these benchmarks not only measure current performance but also provide clear, concrete objectives for optimizing tool usage. Such metrics are instrumental in guiding both immediate performance enhancements and long-term strategic improvements in agent-based systems. In the following sections, we first review the evolution of agent tool use benchmarks and then consolidate the key evaluation metrics that serve as targets for further tool optimization.

Tool Evaluation Benchmarks Recent efforts in LLM-as-Agent research have spawned diverse benchmarks and frameworks for evaluating tool-use capabilities. Early studies such as Gorilla [727] and API-Bank [791] pioneered large-scale datasets and methods for testing LLM interactions with external APIs, shedding light on issues like argument accuracy and hallucination. Subsequent works like T-Bench [792] and ToolBench [690] introduced more extensive task suites and stressed the importance of systematic data generation for tool manipulation. StableToolBench [793] further extended this line of inquiry by highlighting the instability of real-world APIs, proposing a virtual API server for more consistent evaluation. Meanwhile, ToolAlpaca [794] investigated the feasibility of achieving generalized tool-use in relatively smaller language models with minimal in-domain training. Additional efforts like ToolEmu [795] assessed the safety and risk aspects of tool-augmented LM agents through emulated sandbox environments. MetaTool [796] then introduced a new benchmark focused on whether LLMs know *when* to use tools and can correctly *choose* which tools to employ. It provides a dataset named ToolE that covers single-tool and multi-tool usage scenarios, encouraging research into tool usage awareness and nuanced tool selection. ToolEyes [797] pushed the evaluation further by examining real-world scenarios and multi-step reasoning across a large tool library. Finally, τ-bench [798] introduced a human-in-the-loop perspective, emphasizing dynamic user interactions and policy compliance in agent-based conversations. Together, these benchmarks and frameworks underscore the evolving landscape of tool-augmented LLM research, marking a shift from isolated reasoning tasks to comprehensive, real-world agent evaluations.

Metrics for Tool Invocation Deciding whether to invoke an external tool is a critical step that can significantly affect both the efficiency and the effectiveness of a system. In many scenarios, the model must determine if its own reasoning is sufficient to answer a query or if additional external knowledge (or functionality) provided by a tool is required. To formalize this process, we introduce a labeled dataset

$$D_{\text{inv}} = \{(q_i, y_i)\}_{i=0}^{N-1},$$

where q_i represents the i-th user query and $y_i \in \{0, 1\}$ is a binary label indicating whether tool invocation is necessary ($y_i = 1$) or not ($y_i = 0$). Based on this dataset, the model learns a decision function $d(q_i)$ defined as:

$$d(q_i) = \begin{cases} 1, & \text{if } P_\theta(y = 1 \mid q_i) \geq \tau, \\ 0, & \text{otherwise,} \end{cases}$$

where $P_\theta(y = 1 \mid q_i)$ denotes the predicted probability (from a model parameterized by θ) that a tool should be invoked for query q_i, and τ is a predetermined threshold.

In addition to this decision rule, several metrics can be used to evaluate the model's ability to correctly decide on tool invocation. For example, the overall invocation accuracy A_{inv} can be computed as:

$$A_{\text{inv}} = \frac{1}{N} \sum_{i=0}^{N-1} \mathbf{1}\{d(q_i) = y_i\},$$

where $\mathbf{1}\{\cdot\}$ is the indicator function. Other metrics such as precision, recall, and F1 score are also applicable. Moreover, if C_{inv} represents the cost incurred by invoking a tool and $R(q_i)$ the benefit or reward obtained when a tool is correctly used, one can define a net benefit score:

$$B_{\text{inv}} = \sum_{i=0}^{N-1} \left(\mathbf{1}\{d(q_i) = 1\} \cdot R(q_i) - C_{\text{inv}}\right).$$

This formulation not only emphasizes accuracy but also considers the cost-effectiveness of invoking external tools.

Tool Selection Among Candidates Once the decision to invoke a tool is made, the next challenge is to select the most appropriate tool from a pool of candidates. Let the candidate toolset be represented as:

$$\mathcal{T} = \{t_1, t_2, \ldots, t_M\}.$$

For a given query q_i, assume that the optimal tool (according to ground truth) is t_i^* and the model selects $\hat{t}_i$. The simplest measure of selection performance is the tool selection accuracy A_S:

$$A_S = \frac{1}{|Q|} \sum_{q_i \in Q} \mathbf{1}\{\hat{t}_i = t_i^*\}.$$

However, many scenarios involve ranking multiple candidate tools by their relevance. In such cases, ranking-based metrics such as Mean Reciprocal Rank (MRR) and normalized Discounted Cumulative Gain (nDCG) offer a more nuanced evaluation. [690] use those two when evaluating the tool retriever system.

Tool Retrieval Efficiency and Hierarchical Accuracy Tool retrieval involves both the speed of identifying a suitable tool and the accuracy of that selection. Efficient retrieval methods reduce latency and computational overhead, while high retrieval accuracy ensures that the most relevant tool is identified for the task. To evaluate tool usage comprehensively, we adopt a hierarchical framework that distinguishes between retrieval accuracy and selection accuracy. Retrieval accuracy (A_R) reflects how precisely the system retrieves the correct tool from the repository, typically measured by metrics such as Exact Match (EM) and F1 score, which capture both complete and partial matches. In contrast, selection accuracy (A_S) assesses the system's ability to choose the optimal tool from a set of candidates, again using similar metrics. Overall tool usage awareness is further evaluated by accuracy, recall, precision, and F1 score.

The overall retrieval efficiency E_{Ret} is thus can be expressed as:

$$E_{Ret} = \frac{A_R \times A_S \times A_P \times A_U}{C_R}$$

where C_R is the cost associated with retrieval. Optimization strategies may involve training embedding models with feedback mechanisms to enhance both efficiency and each hierarchical component of accuracy.

For a more nuanced evaluation of tool selection, Metatool [796] introduces the Correct Selection Rate (CSR), which quantifies the percentage of queries for which the model selects the expected tool(s). This evaluation framework addresses four aspects: selecting the correct tool among similar candidates, choosing appropriate tools in context-specific scenarios, ensuring reliability by avoiding the selection of incorrect or non-existent tools, and handling multi-tool queries. Together, these metrics and sub-tasks provide a robust measure of both the efficiency and precision in tool retrieval and selection.

Tool Planning for Complex Tasks Complex tasks often require the sequential application of multiple tools to reach an optimal solution. A tool plan can be represented as an ordered sequence

$$\Pi = [t_1, t_2, \ldots, t_K],$$

where K is the number of steps. The quality of such a plan is typically evaluated by balancing its task effectiveness (e.g., via a metric $R_{\text{task}}(\Pi)$) against the plan's complexity (or length). This balance can be captured by a composite planning score of the form

$$S_{\text{plan}} = \alpha \cdot R_{\text{task}}(\Pi) - \beta \cdot K,$$

where α and β are coefficients that adjust the trade-off between the benefits of high task performance and the cost associated with plan complexity. When ground truth plans Π^* are available, similarity metrics such as BLEU or ROUGE can be used to compare the predicted plan Π with Π^*, and an overall planning efficiency metric can be defined accordingly.

In addition, recent work such as ToolEyes [797] highlights the importance of behavioral planning in tool usage. Beyond selecting tools and parameters, it is crucial for LLMs to concisely summarize acquired information and strategically plan

subsequent steps. In this context, the behavioral planning capability is evaluated along two dimensions. First, the score $S_{b\text{-validity}} \in [0, 1]$ is computed by assessing (1) the reasonableness of summarizing the current state, (2) the timeliness of planning for the next sequence of actions, and (3) the diversity of planning. Second, the score $S_{b\text{-integrity}} \in [0, 1]$ is calculated by evaluating (1) grammatical soundness, (2) logical consistency, and (3) the ability to correct thinking. The composite behavioral planning score is then determined as

$$S_{BP} = S_{b\text{-validity}} \cdot S_{b\text{-integrity}},$$

providing a holistic measure of the model's planning capability. This integrated framework ensures that tool planning for complex tasks not only focuses on the selection and ordering of tools but also on maintaining coherent, effective, and strategically sound planning processes.

In summary, optimizing tool performance within an Agent system necessitates a comprehensive approach that balances decision-making accuracy, retrieval efficiency, hierarchical selection precision, strategic planning, rigorous risk management, and robust tool learning mechanisms. By implementing targeted optimization and learning strategies, it is possible to enhance both the effectiveness and efficiency of tool-assisted machine learning workflows.

9.5 Towards Autonomous Agent Optimization

In addition to optimizing individual modules in agent evolution, such as prompts, tools, and workflows—which are susceptible to local optima that can compromise the overall performance of the agentic system, a significant body of research focuses on optimizing multiple components within the entire agentic systems. This holistic approach enables large language model (LLM) agents to evolve more comprehensively. However, optimizing the entire system imposes higher requirements. The algorithm must not only account for the impact of individual components on the agentic system but also consider the complex interactions between different components.

ADAS [741] is one of the most representative works that first formally defines the research problem of automated design in agentic systems. It integrates multiple agentic system components into the evolutionary pipeline. Specifically, ADAS introduces a meta-agent capable of iteratively designing the agentic system's workflow, prompts, and potential tools within the overall optimization process. As demonstrated in the experiments, the automatically designed agentic systems outperform state-of-the-art hand-designed baselines.

Additionally, [726] proposes an agent symbolic learning framework for training language agents, inspired by connectionist learning principles used in neural networks. By drawing an analogy between agent pipelines and computational graphs, the framework introduces a language-based approach to backpropagation and weight updates. It defines a prompt-based loss function, propagates language loss through agent trajectories, and updates symbolic components accordingly. This method enables structured optimization of agentic workflows and naturally extends to multi-agent systems by treating nodes as independent agents or allowing multiple agents to act within a single node.

[799] proposes an approach to optimize both prompts and the agent's own code, enabling self-improvement. This aligns with the concept of self-reference, where a system can analyze and modify its own structure to enhance performance.

Similarly, [773], [787], [800] and [788] focus on optimizing both the workflow and prompts within agentic systems. In particular, [285] introduces an approach that trains additional large language models (LLMs) to generate prompts and workflows, enabling the automated design of agentic system architectures.

In summary, optimizing the workflow of an entire agentic system is not merely a straightforward aggregation of individual component optimizations. Instead, it requires carefully designed algorithms that account for complex interdependencies among components. This makes system-wide optimization a significantly more challenging task, necessitating advanced techniques to achieve effective and comprehensive improvements.

Chapter 10

Large Language Models as Optimizers

In this chapter, we present and discuss existing works that conceptualize LLMs as optimizers. First, we note that most existing studies focus on the prompt optimization problem defined in Equation (9.1), as optimizing other components of agentic workflows remains an emerging research area. To proceed, we draw parallels with classical iterative algorithms and examine their integration into modern optimization workflows.

10.1 Optimization Paradigms

Traditional optimization methods differ in their assumptions about objective function accessibility. We categorize them into three broad classes, each with an expanding level of input space: *gradient-based optimization*, which relies on explicit function gradients; *zeroth-order optimization*, which operates without gradient information; and *LLM-based optimization*, which extends beyond numerical functions to optimize over structured and high-dimensional input spaces.

- **Gradient-Based Optimization.** These methods assume access to gradient information and iteratively refine parameters. Techniques such as stochastic gradient descent (SGD) and Newton's method [801] are widely used but require differentiability, limiting their applicability to discrete problems like prompt tuning and structured decision workflows, often endowed with a graph structure.
- **Zeroth-Order Optimization.** These methods bypass the need for explicit gradients by estimating search directions from function evaluations [802]. Examples include Bayesian optimization [803], evolutionary strategies [804], and finite-difference methods [805], which are effective when gradients are unavailable or expensive to compute. However, they still rely on well-defined numerical objectives and structured search spaces, which constrains their applicability to language-based tasks.
- **LLM-Based Optimization.** LLMs optimize broader solution spaces by leveraging natural language as both the optimization domain and feedback mechanism. By incorporating structured reasoning and human-like iteration, LLMs excel in refining prompts, generating adaptive workflows, and iteratively improving task performance based on user feedback.

While gradient-based and zeroth-order methods are typically applied to numerical objectives, their core principles, such as iterative refinement, search heuristics, and adaptive learning, also underlie LLM-based optimization strategies. Building on these insights, we highlight a rapidly emerging class of LLM-based optimization powered by reinforcement learning, which has become the backbone of slow thinking reasoning models [90, 806, 89]. As these models continue to evolve, we anticipate them driving the next wave of agentic applications, enabling LLMs to navigate complex environments with greater adaptability and strategic foresight.

10.2 Iterative Approaches to LLM Optimization

Some LLM-based optimization methods directly draw inspiration from classical optimization theory by adapting key components to address discrete and structured challenges. A central characteristic of these approaches is the iterative update step, in which model-generated modifications are selected from a range of possible improvements to refine the objective. Using the prompt optimization objective from Equation (9.1) as a running example, a general iterative

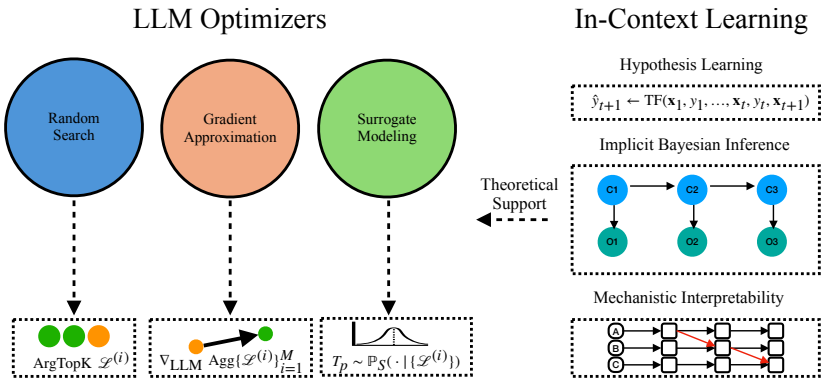


Figure 10.1: A taxonomy of LLM-based optimization methods, categorized into random search, gradient approximation, and surrogate modeling. We also highlight some theoretical explanations of in-context learning, which includes hypothesis learning, implicit Bayesian inference, and mechanistic interpretability, which underpin the optimization capabilities of LLMs.

algorithm can be expressed as follows:

$$\begin{aligned}\textbf{Sample:}\ & T \sim \mathcal{D} \\ \textbf{Evaluation:}\ & \mathcal{L}(T; T_p) \leftarrow \phi_{\text{eval}}\left(\phi_{\text{exe}}(Q, T_p), T\right) \\ \textbf{Update:}\ & T_p' \leftarrow \phi_{\text{opt}}\left(\mathcal{L}(T; T_p)\right)\end{aligned}$$

Here, the **Sample** and **Update** steps are defined based on the agent's task. In the simplest case, such as optimizing an instruction for binary classification of movie reviews, the objective $\mathcal{L}$ is measured by classification accuracy. In more complex agentic workflows, the decision variable may include prompts at different workflow stages, tool selections, agent topologies, or a combination thereof. As discussed in Chapter 9, a common characteristic of these decision variables is their *combinatorial* nature-such as the set of all strings from an LLM's vocabulary $\mathcal{V}$ or all possible role assignments for agents in a workflow. Since enumerating all possible solutions is often intractable, this necessitates designing approximate update steps ϕ_{opt}, which we discuss next.

- **Random Search.** Early LLM-based optimization methods leveraged random search variants to optimize prompts in discrete natural language spaces [774, 807, 651, 732, 808, 809, 810]. These methods often resemble evolutionary algorithms that iteratively sample candidate decision variables and select the top-performing ones from each iteration. The general formulation follows:

$$\begin{aligned}\textbf{Sample:}\ & T \sim \mathcal{D} \\ \textbf{Evaluation:}\ & \mathcal{L}^{(i)} \leftarrow \phi_{\text{eval}}(\phi_{\text{exe}}(Q, T_p^{(i)}), T), \quad i = 1, \ldots, M \\ \textbf{Update:}\ & \{T_p^{(k)\prime}\}_{k=1}^{K} \leftarrow \text{ArgTopK}_{i \in [M]}\, \mathcal{L}^{(i)}, \\ \textbf{Replenishment (Optional):}\ & \{T_p^{(j)}\}_{j=K+1}^{M} \sim \text{Mutate}(\{T_p^{(k)}\}_{k=1}^{K}).\end{aligned}$$

  We briefly override previous notations and let M denote the total number of candidate prompts sampled per iteration, and K (with $K < M$) control the number of top-performing candidates-selected with ArgTopK in our algorithm-retained for the next step. This algorithm can optionally incorporate a replenishment step to maintain diversity in the candidate pool across iterations. Random search methods are simple to implement, highly parallelizable, and particularly effective for single-prompt workflows. Beyond prompt optimization, they have also demonstrated strong performance in selecting in-context demonstrations [811, 812]. However, their efficiency comes at a cost—each iteration requires $O(M)$ parallel API queries, which can become prohibitively expensive for complex workflows involving multiple queries.

- **Gradient Approximations.** Several methods approximate gradient-based updates by iteratively refining solutions. For instance, [779, 730, 728] generate refinements at different workflow stages. StraGO [722] estimates descent directions using central-difference heuristics, while Trace [813] optimizes composed programs by modeling them as computation graphs, similar to backpropagation. The key analogy between gradient updates in continuous optimization and prompt-space refinement is the concept of a "descent direction"—a systematic *modification* of the decision variable to improve the objective. In contrast, random search methods propose new decision variables independently at each step, without accessing past update trajectories. Gradient-based approaches, by contrast, exploit this historical information, often leading to faster convergence. A general iteration for gradient approximation methods is given below:

$$\begin{aligned}
\textbf{Sample:}\ & T^{(i)} \sim \mathcal{D}, \quad i = 1, \ldots, M \\
\textbf{Evaluation:}\ & \mathcal{L}^{(i)} \leftarrow \phi_{\text{eval}}(\phi_{\text{exe}}(Q, T_p), T^{(i)}), \quad i = 1, \ldots, M \\
\textbf{Gradient Approximation:}\ & g \leftarrow \nabla_{\text{LLM}}\, \text{Agg}\left(\mathcal{L}^{(1)}, \ldots, \mathcal{L}^{(M)}\right) \\
\textbf{Update:}\ & T_p' \leftarrow \phi_{\text{opt}}(T_p, g),
\end{aligned}$$

  where M is the minibatch size, $\text{Agg}(\cdot)$ is an aggregation function that combines feedback signals (e.g., in numerical optimization, Agg is typically the average operator), ∇_{LLM} represents an abstract "LLM-gradient operator" [728] that generates textual refinement directions based on the feedback signal and the current minibatch (e.g., *the agent should consider the edge case of* ...). Additionally, ϕ_{opt} can be instantiated as an LLM query, allowing the agent to update its prompt based on g.

  Compared to random search methods, gradient-based approaches offer two key advantages: they enable the incorporation of past refinement directions into ϕ_{opt}, analogous to momentum-based techniques in first-order optimization algorithms [814, 815], and they facilitate backpropagation-like techniques for optimizing computation graphs [651, 813, 780], making them particularly effective for multi-stage workflows with interdependent optimizable modules. However, this flexibility comes at the cost of increased design overhead, such as the need for meta-prompts to aggregate feedback and apply refinement directions. We further discuss the feasibility of using LLMs to optimize these *hyperparameters* below. Some approaches also explored direct gradient-based optimization of soft prompts [816, 817, 818]. While effective for simple input-output sequence learning, these methods struggle with multi-step workflows and sequential decision-making [630, 300].

  Finally, while these methods leverage first-order optimization insights, the extension of second-order techniques (e.g., quasi-Newton methods) to LLM-based optimization remains largely unexplored. Fortunately, recent works such as Revolve [780] have taken a step in this direction by introducing a structured approach for second-order optimization, modeling the evolution of response patterns over multiple iterations. By incorporating higher-order refinements, Revolve enables more stable and informed optimization, effectively mitigating stagnation in complex tasks. We are also excited by emerging trends in leveraging inference-time compute [90, 89] to incorporate historical refinement directions and investigate the benefits of momentum.

- **Bayesian Optimization and Surrogate Modeling.** While the aforementioned approaches achieved significant progress in LLM-based optimization, they often entail substantial financial and environmental costs due to the high number of required LLM interactions. Moreover, these methods can be sensitive to noise, and the optimization landscape of discrete prompts, among other decision variables, remains poorly understood [819, 820]. Under these constraints, Bayesian Optimization (BO) emerges as a compelling alternative, as it builds a noise-resilient surrogate model of the optimization objective:

$$\begin{aligned}
\textbf{Sample:}\ & T \sim \mathcal{D} \\
\textbf{Proposal:}\ & \{T_p^{(i)}\}_{i=1}^{M} \sim S.\,\text{Propose} \\
\textbf{Evaluation:}\ & \mathcal{L}^{(i)} \leftarrow \phi_{\text{eval}}(\phi_{\text{exe}}(Q, T_p^{(i)}), T), \quad i = 1, \ldots, M \\
\textbf{Update:}\ & S \leftarrow S.\,\text{UpdatePrior}(\{\mathcal{L}^{(i)}\}_{i=1}^{M}, \{T_p^{(i)}\}_{i=1}^{M}),
\end{aligned}$$

  where S represents a probabilistic surrogate model of the optimization objective, equipped with a proposal operator (e.g., posterior sampling from a Gaussian Process BO procedure [803]) and an update mechanism based on observed evidence from prompt evaluations. For instance, MIPRO [821] employs a Tree-Structured Parzen Estimator as its surrogate [822], while PROMST [823] trains a score-prediction model to guide prompt tuning. Leveraging a surrogate model for LLM-based optimization aligns with the emerging trend of amortized optimization for non-differentiable objectives [824]. For instance, [825] trains a prompt-generator LLM to amortize the computational cost of instantiating a beam search problem for discovering jailbreak attack prefixes.

Finally, several other works fit an additional lightweight module-such as a Bayesian belief posterior or a utility function-from LLM outputs, to aid the optimization of domain-specific workflows, such as decision-making and multi-agent negotiations [826, 827]. This type of amortized methods-those that fit a parameterized model that is reusable for unseen inputs-have found increasing usage in LLM-based optimization, such as jailbreaking [828, 825].

10.3 Optimization Hyperparameters

Similar to traditional optimization, LLM-based methods are highly sensitive to hyperparameters that influence search efficiency and generalization. A key consideration in gradient-based LLM optimizers is the choice of the aggregation function $\text{Agg}(\cdot)$, which determines how textual feedback is synthesized to guide prompt updates. An improper choice can lead to loss of critical information or misalignment in iterative refinements. Additionally, [813] introduces a "whiteboard" approach, where an LLM program is decomposed into human-interpretable modules. However, design choices in structuring such modular workflows remain largely unexplored, which poses an open challenge for optimizing LLM-driven decision-making pipelines.

Hyperparameters in LLM optimization often parallel those in numerical optimization. For example, batch size plays a crucial role: just as minibatch updates enhance stability and efficiency in classical optimization, LLM-based approaches like TextGrad [728] aggregate feedback across multiple generated samples before making updates. Another key factor is momentum—while it stabilizes updates in gradient-based methods by incorporating past gradients, LLM-based optimizers similarly leverage historical refinements to improve performance over time [728, 813]. Despite progress in numerical optimization, hyperparameter selection for LLM-based optimizers remains largely heuristic, often relying on ad hoc, trial-and-error tuning.

In agentic system design, hyperparameters proliferate across various components, including role assignments of agents, selection of in-context demonstrations, and scheduling of tool invocations. Each of these choices has a profound impact on downstream performance, yet principled methods for optimizing them remain underdeveloped. While traditional hyperparameter tuning techniques, such as grid search and Bayesian optimization, can be applied to discrete LLM-driven workflows, their computational cost scales poorly due to the high variance in language model outputs. Additionally, the combinatorial nature of these hyperparameters, where agent configurations, prompting strategies, and reasoning structures interact in complex ways, makes an exhaustive search infeasible. Recent work has attempted to address this challenge by embedding agentic workflows into structured frameworks such as finite state machines [729], optimal decision theory [826], and game theory [827]. However, these approaches often fail to generalize across diverse environments. A promising direction for addressing these challenges is meta-optimization, where LLMs are used to optimize their own hyperparameters and decision-making strategies. For example, an LLM-based optimizer can iteratively refine its own prompting strategies by treating past decisions as experience, akin to learned optimizers in deep learning [829]. Moreover, amortized approaches train auxiliary models to predict effective hyperparameters, which can reduce the computational cost of exhaustive search [821, 823]. While these techniques offer exciting possibilities, they also introduce new challenges, such as balancing exploration with exploitation in adaptive tuning and ensuring generalization across diverse optimization tasks. Investigating principled meta-optimization strategies tailored to LLM-driven workflows remains a critical area for future research.

10.4 Optimization across Depth and Time

Unlike conventional optimizers that update parameters in a static setting, LLMs optimize workflows dynamically, considering both depth (single-pass workflows) and time (recurrent updates). In terms of depth, LLMs function similarly to feedforward networks, sequentially optimizing workflows as they pass through different modules—most existing LLM-based optimizers follow this paradigm. Beyond single-pass execution, LLMs can also optimize over time, akin to recurrent architectures such as RNNs or Universal Transformers [830], by iteratively refining decision-making. For instance, StateFlow [729] enhances workflows by incorporating feedback across multiple iterations, enabling dynamic refinement and adaptation over time. While these analogies are compelling, many well-established engineering optimization techniques—such as checkpointing [831] and truncated backpropagation [832]—remain underexplored in LLM-based optimization. We see this as a promising avenue for future research, echoing previous calls for deeper investigation [813].

10.5 A Theoretical Perspective

Recent studies suggest that transformers inherently perform optimization-like computations, supporting their potential as general-purpose optimizers for computational workflows. However, a significant gap remains between their empirical success and theoretical understanding. Here, we provide a brief overview of recent progress in bridging this gap.

- **In-Context Learning.** A fundamental perspective on transformers as optimizers emerges from in-context learning, particularly in few-shot settings [2]. [733] demonstrated that transformers can in-context learn diverse regression hypotheses, including regularized linear models, decision trees, and shallow neural networks. Building on this, later works [734, 833, 735] provided constructive proofs that transformers can implement iterative optimization algorithms, such as gradient descent and second-order updates. However, while these theoretical models characterize transformers' optimization capabilities, they do not fully explain in-context learning in large-scale LLMs, which operate in discrete input-output spaces. Empirical analyses [819, 834, 820] instead sought to understand how pre-trained LLMs generalize in-context. [834] proposed that in-context learning resembles a hidden Markov model (HMM) performing implicit Bayesian inference, while [819, 820] challenged the conventional view that in-context demonstrations serve as new test-time samples for hypothesis formation. In-context learning remains the central emergent ability [835] enabling self-improvement and optimization from context, yet it continues to elude comprehensive theoretical analysis.
- **Mechanistic Interpretability.** Parallel to theoretical analyses, mechanistic interpretability aims to uncover internal transformer computations by identifying subgraphs, also known as circuits, responsible for specific behaviors. Early studies mapped circuits for stylized language tasks in pre-trained GPT-2 models [836, 837, 838], while more recent efforts have scaled up by identifying semantically meaningful features using sparse autoencoders [839, 736, 840, 841]. These methods have been largely successful in eliciting causal and controllable behavior from frontier-class LLMs, but they also reveal an unintended consequence: in-context learning capabilities often entangle beneficial generalization with harmful behaviors when conditioned on many-shot demonstrations [842]. This raises challenges for optimizing LLM workflows safely and reliably.
- **Limitations Under Uncertainty.** While LLMs demonstrate moderate capabilities in sequential decision-making when provided with in-context information, they struggle to make optimal choices under uncertainty [843, 844, 845, 846]. In particular, [826] found that LLM-based optimizers exhibit difficulty in adapting to stochastic environments, often failing to explore optimally. These findings serve as a cautionary note for deploying LLM-based optimizers in dynamic or uncertain settings where exploration and robust decision-making are critical.

LLMs redefine optimization by integrating structured reasoning, natural language processing, and in-context learning, expanding beyond traditional numerical methods. Despite strong empirical performance in structured search spaces, open questions remain about the theoretical underpinnings of LLM-based optimization, particularly the emergence of in-context learning from standard gradient-based training.

Chapter 11

Online and Offline Agent Self-Improvement

In the pursuit of self-improvement, intelligent agents leverage optimization as both a mechanism for refining individual components—such as prompt design, workflow orchestration, tool utilization, reward function adaptation, and even the optimization algorithms themselves—and as a strategic framework that ensures these individual improvements are aligned toward coherent performance enhancement. For instance, optimizing the reward function and prompt design in isolation might yield conflicting outcomes, but a strategic approach coordinates these optimizations to maintain coherence and maximize overall effectiveness. We categorize self-evolution into two primary paradigms: *online* and *offline* self-improvement. Additionally, we explore *hybrid* optimization strategies that integrate both approaches to maximize efficiency and adaptability.

11.1 Online Agent Self-Improvement

Online self-improvement refers to real-time optimization in which an agent dynamically adjusts its behavior based on immediate feedback. This paradigm ensures that agents remain responsive to evolving environments by continuously optimizing key performance metrics—such as task success, latency, cost, and stability—in an iterative feedback loop. Online self-improvement is particularly effective in applications that require dynamic adaptability, such as real-time decision-making, personalized user interactions, and automated reasoning systems. Key optimization strategies in online self-improvement can be classified into the following four categories: Iterative Feedback and Self-Reflection, Active Exploration in Multi-Agent Systems, Real-Time Reward Shaping, and Dynamic Parameter Tuning.

Iterative Feedback and Self-Reflection These methodologies [48, 67, 72, 70, 847, 47] focus on enabling agents to critique and refine their own outputs iteratively. Reflexion [48], Self-Refine [67], and Tree of Thoughts [72] introduce self-critique loops, where the model identifies errors and proposes revisions in real-time. ReAct [70] combines chain-of-thought "reasoning" with "acting", allowing the model to revise steps iteratively after observing external feedback. In addition, other methods either rely on self-consistency [78] to select the most coherent solution or leverage a process reward model (PRM)Lightman et al. [847] to choose the best solution from the candidates. Collectively, these frameworks reduce error propagation and support rapid adaptation without requiring a separate offline fine-tuning cycle.

Active Exploration in Multi-Agent Systems These approaches [626, 848, 627, 152] actively explore and dynamically search for novel patterns and workflow improvements in multi-agent systems. MetaGPT [626], CAMEL [848], and ChatDev [627] showcase multi-role or multi-agent ecosystems that interact in real-time, exchanging continuous feedback to refine each other's contributions. Similarly, HuggingGPT [152] coordinates specialized models (hosted on Hugging Face) through a central LLM controller, which dynamically routes tasks and gathers feedback. These collaborative strategies further highlight how online updates among agents can incrementally refine collective outcomes.

Real-Time Reward Shaping Rather than relying on fixed or purely offline reward specifications, some frameworks [731, 91, 105, 849] integrate immediate feedback signals not only to correct errors, but also to adapt internal reward functions and policies. This enables self-adaptive reward calibration that balances trade-offs between performance, computational cost, and latency, allowing agents to optimize reward mechanisms dynamically in response to user interactions.

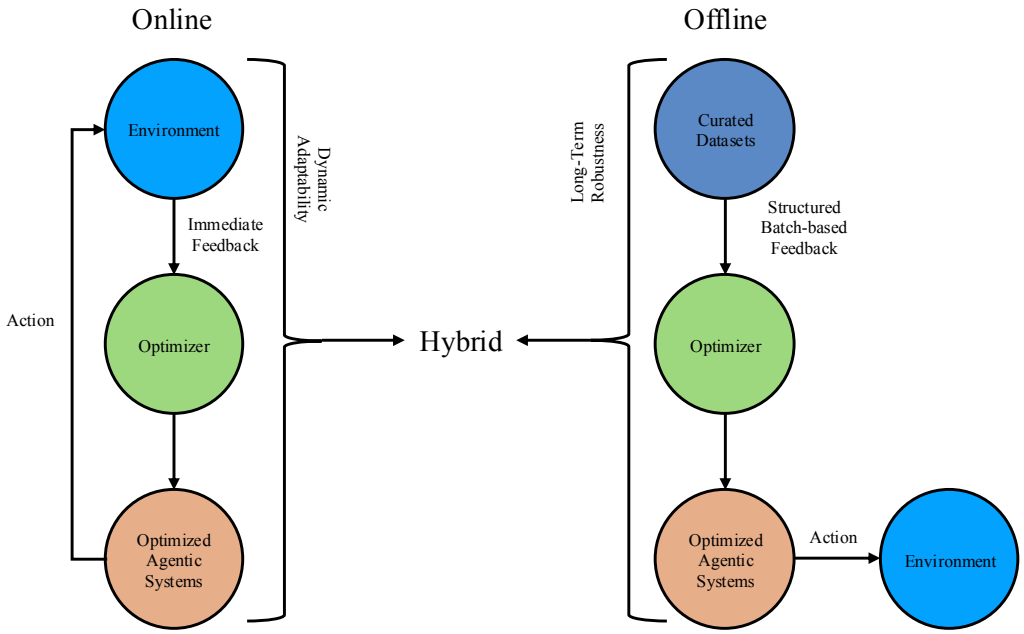


Figure 11.1: An illustration of self-improvement under three different utilization scenarios, including Online, Offline, and Hybrid self-improvement.

Dynamic Parameter Tuning In this category, agents autonomously update their internal parameters (including prompt templates, tool invocation thresholds, search heuristics, etc.) in real time, leveraging gradient-free or approximated gradient methods. These updates optimize both computational efficiency and decision accuracy, allowing for seamless adaptation to evolving contexts. Self-Steering Optimization (SSO) [850] eliminates the need for manual annotation and maintains signal accuracy while keeping training on-policy by autonomously generating preference signals during iterative training.

Online self-improvement fosters a continuously evolving agent framework where learning is embedded within task execution, promoting enhanced real-time adaptability, user-centric optimization, and robust problem-solving capabilities.

11.2 Offline Agent Self-Improvement

Offline self-improvement, in contrast, leverages structured, batch-based optimization. This paradigm utilizes scheduled training sessions with high-quality curated datasets to systematically improve the agent's generalization capabilities [851, 667, 852, 853, 854]. Unlike online approaches, offline approaches accommodate more computationally intensive methodologies, including Batch Parameter Updates and Fine-Tuning, Meta-Optimization, and Systematic Reward Model Calibration.

Batch Parameter Updates and Fine-Tuning In this category, agents undergo extensive fine-tuning using supervised learning or reinforcement learning (RL) techniques, optimizing performance across large-scale datasets over multiple training epochs. Retrieval-augmented generation (RAG) is often integrated to enhance contextual understanding and long-term memory retrieval [740, 741]. Such methods allow agents to optimize retrieval strategies, thereby improving reasoning over extensive knowledge corpora.

Meta-Optimization of Agent Components Here offline training is not limited to improving task performance but extends to refining optimization algorithms themselves. Meta-learning strategies that optimize hyperparameters or even restructure the optimization process dynamically have demonstrated promising outcomes [731, 91]. These meta-optimization approaches enable agents to discover the most effective learning parameters for new problem domains.

Systematic Reward Model Calibration Offline settings facilitate the precise calibration of reward models, incorporating hierarchical or listwise reward integration frameworks (e.g., LIRE [855]) to align agent behavior with long-term objectives through gradient-based reward optimization. Such calibration ensures that reward functions reflect real-world task complexity, thereby mitigating bias and enhancing generalization.

The structured nature of offline optimization results in a robust agent baseline, whose performance is fine-tuned to optimize stability, efficiency, and computational cost before real-world deployment. Offline training allows for high-fidelity model refinement and is essential for mission-critical applications requiring predictable performance guarantees.

11.3 Comparison of Online and Offline Improvement

Online and offline optimization offer complementary benefits, each excelling in different aspects of self-improvement. Online optimization thrives in dynamic environments, where real-time feedback enables continuous adaptation. It is well-suited for applications that require immediate responsiveness, such as interactive agents, real-time decision-making, and reinforcement learning systems. However, frequent updates may introduce instability or drift, requiring mechanisms to mitigate performance degradation over time.

In contrast, offline optimization emphasizes structured, high-fidelity training using pre-collected datasets, ensuring robust and stable performance before deployment. By leveraging computationally intensive learning methods such as batch training, fine-tuning, and meta-optimization, offline approaches provide strong generalization and long-term consistency. However, they lack the agility of online learning and may struggle to adapt efficiently to novel scenarios without additional retraining. Table 11.1 summarizes the key distinctions between these two paradigms.

Feature	**Online Optimization**	**Offline Optimization**
Learning Process	Continuous updates based on real-time feedback	Batch updates during scheduled training phases
Adaptability	High, capable of adjusting dynamically	Lower, adapts only after retraining
Computational Efficiency	More efficient for incremental updates	More resource-intensive due to batch training
Data Dependency	Requires real-time data streams	Relies on curated, high-quality datasets
Risk of Overfitting	Lower due to continuous learning	Higher if training data is not diverse
Stability	Potentially less stable due to frequent updates	More stable with controlled training settings

Table 11.1: Comparison of Online vs. Offline Optimization Strategies in Self-Improvement Agents.

While both approaches have inherent strengths and trade-offs, modern intelligent systems increasingly integrate them through hybrid optimization strategies. These hybrid frameworks leverage the stability of offline training while incorporating real-time adaptability, enabling agents to maintain long-term robustness while continuously refining their performance in dynamic environments.

11.4 Hybrid Approaches

Recognizing that both online and offline methods have inherent limitations, many contemporary systems adopt *hybrid* optimization strategies. These hybrid methods integrate structured offline optimization with responsive online updates to achieve continuous incremental agent enhancement.

Hybrid optimization explicitly supports self-improvement by empowering agents to autonomously evaluate, adapt, and enhance their behaviors through distinct yet interconnected stages:

- **Offline Pre-Training:** In this foundational stage, agents acquire robust baseline capabilities through extensive offline training on curated datasets. This stage establishes essential skills, such as reasoning and decision-making, required for initial autonomous performance. For instance, frameworks such as the one introduced by Schrittwieser et al. [856] illustrate how offline pretraining systematically enhances initial agent capabilities, ensuring subsequent online improvements are built upon a stable foundation.
- **Online Fine-Tuning for Dynamic Adaptation:** Agents actively refine their capabilities by autonomously evaluating their performance, identifying shortcomings, and dynamically adjusting strategies based on real-time feedback. This adaptive fine-tuning stage directly aligns with the agent self-improvement paradigm by allowing real-time

optimization of agent-specific workflows and behaviors, exemplified by Decision Mamba-Hybrid (DM-H) [857], where agents efficiently adapt to complex, evolving scenarios.

- **Periodic Offline Consolidation for Long-Term Improvement:** periodic offline consolidation phases, agents systematically integrate and solidify improvements identified during online interactions. This ensures that incremental, online-acquired skills and improvements are systematically integrated into the agent's core models, maintaining long-term stability and effectiveness. The Uni-O4 framework [858] exemplifies how this process enables seamless transitions between offline knowledge consolidation and online adaptive improvements.

Hybrid optimization thus explicitly supports autonomous, continuous evolution by seamlessly interweaving structured offline learning with proactive, real-time online adaptation. This cyclical approach equips agents with both immediate responsiveness and stable long-term improvement, making it ideally suited for complex, real-world scenarios such as autonomous robotics, personalized intelligent assistants, and interactive systems.

Chapter 12

Scientific Discovery and Intelligent Evolution

In previous chapters, we primarily discussed the evolution of agentic systems from a technical perspective, focusing on how to develop systems that can effectively perform well-defined tasks traditionally executed by humans. However, a fundamental and important question remains: can these agents drive a self-sustaining innovation cycle that propels both agent evolution and human progress?

Scientific knowledge discovery is a compelling example of self-evolution in intelligent beings, as it helps them adapt to the world in a sustainable way. Agents capable of discovering scientific knowledge at different levels of autonomy and in a safe manner will also play important roles in technological innovation for humanity. In this section, we survey progress in autonomous discovery using agentic workflows and discuss the technological readiness toward fully autonomous, self-evolving agents. Within this scope, the goal of the agent is to uncover, validate, and integrate data, insights, and principles to advance an objective scientific understanding of natural phenomena. Instead of altering the world, the agent seeks to better understand nature as a Scientist AI [859] and assist humans in extending the boundaries of knowledge.

We first define the concept of knowledge and intelligence to clarify our discussion, then introduce three typical scenarios where agents and scientific knowledge interact. We also highlight existing successes and examples of self-enhancing agents applied to theoretical, computational, and experimental scientific research. Lastly, we summarize the current challenges for a future outlook.

12.1 Agent's Intelligence for Scientific Knowledge Discovery

Knowledge, traditionally defined as *justified true belief*, traces back to Plato [860] and has been further refined by Edmund Gettier [861], who argued that knowledge must be produced by a reliable cognitive process—though its precise definition remains debated [862]. In our discussion, we describe scientific knowledge discovery as the process of collecting data and information to either justify or falsify rational hypotheses about target scientific problems. To discuss the capability of agents in scientific knowledge discovery, we first explore a general framework for measuring an agent's intelligence through the lens of information theory.

12.1.1 KL Divergence-based Intelligence Measure

The agent's intelligence can be measured by the KL divergence between its predicted and real-world probability distributions of unknown information. A long-standing goal in both artificial intelligence and the philosophy of science is to formalize what it means for an agent to "understand" the world. From Jaynes' view of probability theory as extended logic for reasoning under uncertainty [863], to Parr et al.'s framing of intelligence as minimizing model-world divergence under the free energy principle [864], many frameworks converge on a common theme: intelligent behavior arises from making accurate predictions about an uncertain world. Clark [344], for instance, argues that intelligent agents constantly engage with the world through prediction and error correction to reduce surprise. Chollet [865] emphasizes that intelligence should reflect skill-acquisition efficiency, because of the dynamic nature of task adaptation. Together, these views suggest that intelligence involves building predictive and adaptable models—an idea formalized here through a probabilistic framework that links reasoning to knowledge acquisition and enables comparison across agents in scientific discovery.

Building on this foundation, we consider intelligence in the specific context of scientific knowledge discovery, where the agent's primary objective is to infer unknown aspects of the physical world from limited data. From the agent's perspective in knowledge discovery, the world $\mathcal{W}$ is characterized by an ensemble of datasets $\mathbf{x} = \{x_1, x_2, ..., x_n\}$ related to the scientific problem the agent aims to understand. During the agent's interaction with $\mathcal{W}$, each dataset appears in the experimental measurements or observations with a probability $P_{\mathcal{W}}(\mathbf{x})$. Here we assume that individual data points x_i may or may not be correlated. For example, in a task of text generation using a language model, x_i represents a chunk of tokens forming a meaningful proposition, and $\mathbf{x}$ is a coherent text constructed from known and inferred propositions. In this context, the "world" is the ensemble of all propositions.

Let θ denote the parameter that parameterizes the agent's world model, M_t^{wm}, as defined in Table 1.2. For instance, in a transformer model with a fixed architecture, θ represents its weights. Given θ and a dataset $\mathbf{x}$, the agent predicts a probability distribution $P_\theta(\mathbf{x})$. In general, different AI agents could be optimized for different goals. For scientific knowledge discovery, we assume that the agent's goal is to produce a good description of the real world, i.e., a world model that predicts yet-to-be-explored natural phenomena as accurately as possible. A more intelligent agent produces a better approximation of the real-world distribution $P_{\mathcal{W}}(\mathbf{x})$. The agent's intelligence can thus be measured by the KL divergence, or relative entropy, between these two probability distributions:

$$D_0(\theta) = \sum_{\mathbf{x}\subseteq\mathcal{W}} P_{\mathcal{W}}(\mathbf{x}) \log \frac{P_{\mathcal{W}}(\mathbf{x})}{P_\theta(\mathbf{x})} \tag{12.1}$$

$D_0(\theta)$ describes the difference between $P_{\mathcal{W}}(\mathbf{x})$ and $P_\theta(\mathbf{x})$. More precisely, in the context of hypothesis testing, if we sample $P_{\mathcal{W}}(\mathbf{x})$ N times and compare the results with the predictions from $P_\theta(\mathbf{x})$, the probability of mistaking $P_{\mathcal{W}}(\mathbf{x})$ for $P_\theta(\mathbf{x})$ scales as $e^{-ND_0(\theta)}$ [866]. In other words, an agent with a lower $D_0(\theta)$ produces predictions that align more closely with reality.

For example, consider two materials synthesis agents whose goal, M_t^{goal}, is to understand whether or not an inorganic compound of interest, $CaFe_2(PO_4)_2O$, is synthesizable. The agents can predict either (1) $\mathbf{x}_1$={$CaFe_2(PO_4)_2O$ is synthesizable}, and (2) $\mathbf{x}_2$={$CaFe_2(PO_4)_2O$ is not synthesizable}. In reality, since $CaFe_2(PO_4)_2O$ is a natural mineral, $P_{\mathcal{W}}(\mathbf{x}_1)$ = 1 and $P_{\mathcal{W}}(\mathbf{x}_2)$ = 0. However, this mineral was only recently reported on October 4, 2023[ref], after the knowledge cutoff of many LLMs; thus, the agents lacks that knowledge. Compare Agent 1, which guesses randomly $P_{\theta_1}(\mathbf{x}_1) = P_{\theta_1}(\mathbf{x}_1)$ = 0.5, yielding $D_0(\theta_1) = \log 2$. In contrast, Agent 2 uses first-principles calculations and finds that $CaFe_2(PO_4)_2O$ (assume structure is xx [cite: Materials Project ID]) is the lowest-energy phase among its competitors [ref], indicating stability. Thereby, Agent 2 predicts that $CaFe_2(PO_4)_2O$ is likely synthesizable, suggesting $P_{\theta_2}(\mathbf{x}_1) > 0.5 > P_{\theta_2}(\mathbf{x}_2)$. Consequently, $D_0(\theta_2) = -\log P_{\theta_2}(\mathbf{x}_1) < D_0(\theta_1)$, meaning that Agent 2 has a more accurate understanding of the real world.

Now, let us assume the agent has conducted some measurements and determined specific values for a subset of data points x_i. Let $\mathbf{x}_{\mathrm{K}}$ denote this known subset and $\mathbf{x}_{\mathrm{U}}$ the remaining unknown part. Correspondingly, we define the space of all existing knowledge as $\mathcal{K}$ and the space of all unknown information as $\mathcal{U}$, satisfying $\mathbf{x}_{\mathrm{K}} \subseteq \mathcal{K}$, $\mathbf{x}_{\mathrm{U}} \subseteq \mathcal{U}$, and $\mathcal{K} \cup \mathcal{U} = \mathcal{W}$. For example, in text generation, the the prompt text $\mathbf{x}_{\mathrm{K}}$ represents already known information. The efficiency of the language model is then measured by its predictive accuracy for the generated text $\mathbf{x}_{\mathrm{U}}$ based on $\mathbf{x}_{\mathrm{K}}$. More generally, the agent's intelligence is measured by the relative entropy of the conditional probability distribution:

$$D_{\mathrm{K}}(\theta, \mathbf{x}_{\mathrm{K}}) = \sum_{\mathbf{x}\subseteq\mathcal{U}} P_{\mathcal{W}}\left(\mathbf{x}|\mathbf{x}_{\mathrm{K}}\right) \log \frac{P_{\mathcal{W}}\left(\mathbf{x}|\mathbf{x}_{\mathrm{K}}\right)}{P_\theta\left(\mathbf{x}|\mathbf{x}_{\mathrm{K}}\right)} \tag{12.2}$$

In practice, all of the agent's knowledge is stored in its memory M_t^{mem}, i.e., $\mathbf{x}_{\mathrm{K}} = \mathcal{K} = M_t^{\mathrm{mem}}$ and $\mathcal{U} = \mathcal{W} \setminus M_t^{\mathrm{mem}}$, we define the agent's intelligence as:

$$IQ_t^{\mathrm{agent}} \equiv -D_{\mathrm{K}}(\theta, M_t^{\mathrm{mem}}) = -\sum_{\mathbf{x}\subseteq\mathcal{U}} P_{\mathcal{W}}\left(\mathbf{x}|M_t^{\mathrm{mem}}\right) \log \frac{P_{\mathcal{W}}\left(\mathbf{x}|M_t^{\mathrm{mem}}\right)}{P_\theta\left(\mathbf{x}|M_t^{\mathrm{mem}}\right)} \tag{12.3}$$

In other words, the the agent's intelligence IQ_t^{agent} is determined by its memory M_t^{mem} and the parameter θ of its world model M_t^{wm}. A schematic plot is shown in Figure 12.1. At time $t = 0$, when the M_t^{mem} is very limited or lack relevant information to a new target scientific problem, IQ_t^{agent} is primarily determined by the zero-shot predictive ability of M_t^{wm}, corresponding to fluid intelligence [867]. Over time, as more relevant knowledge is incorporated into M_t^{mem}, IQ_t^{agent} becomes increasingly dependent on the knowledge-augmented predictive capability of M_t^{wm}, reflecting crystallized intelligence [868].

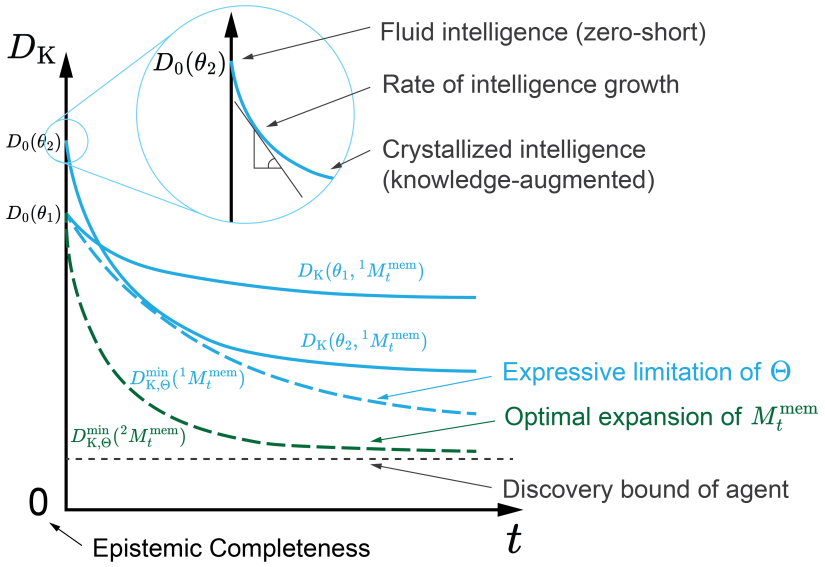


Figure 12.1: **Schematic representation of agent intelligence and knowledge discovery.** The agent's intelligence, measured by the KL divergence D_{K} between predictions and real-world probability distributions, evolves from fluid intelligence (zero-shot predictions for new problems) to crystallized intelligence (knowledge-augmented predictions after learning) as it accumulates data in its memory M_t^{mem} over time t. Given M_t^{mem}, the evolution of D_{K} varies within the world model's parameter space Θ, as illustrated by θ_1 and θ_2 in the solid lines. The expressive limitation of Θ is characterized by the envelope $D^{\min}_{\mathrm{K},\Theta}$. Given Θ, $D^{\min}_{\mathrm{K},\Theta}$ is influenced by different knowledge expansion strategies, such as ${}^1M_t^{\mathrm{mem}}$ and ${}^2M_t^{\mathrm{mem}}$, shown as dash lines.

12.1.2 Statistical Nature of Intelligence Growth

The agent's intelligence, in a statistical sense, is a non-decreasing function of acquired knowledge. Roughly speaking, IQ_t^{agent} quantifies both the amount of knowledge an agent has acquired and how effectively the agent can apply that knowledge after learning from M_t^{mem}. Intuitively, if the agent gains additional information at time t—which corresponds to enlarging M_t^{mem} and shrinking $\mathcal{U}$—its intelligence should increase.

To understand this process, consider a small region $\Delta \subseteq \mathcal{U}$ and examine the effect of adding a dataset $\mathbf{x}_\Delta$ from Δ to M_t^{mem}. Denote $\mathcal{U} = \mathcal{U}' \cup \Delta$, where $\mathcal{U}'$ represents the remaining unknown part of the world. The agent's intelligence at time $t+1$ is given by:

$$IQ_{t+1}^{\mathrm{agent}} \equiv -D_{\mathrm{K}}(\theta, M_t^{\mathrm{mem}}\mathbf{x}_\Delta) = -\sum_{\mathbf{x}'\subseteq\mathcal{U}'} P_{\mathcal{W}}\left(\mathbf{x}'|M_t^{\mathrm{mem}}\mathbf{x}_\Delta\right)\log\frac{P_{\mathcal{W}}\left(\mathbf{x}'|M_t^{\mathrm{mem}}\mathbf{x}_\Delta\right)}{P_\theta\left(\mathbf{x}'|M_t^{\mathrm{mem}}\mathbf{x}_\Delta\right)} \tag{12.4}$$

Directly comparing IQ_t^{agent} and $IQ_{t+1}^{\mathrm{agent}}$ is challenging. Instead, we can compare the expected value of $IQ_{t+1}^{\mathrm{agent}}$, averaging over $\mathbf{x}_\Delta$ with probability $P_{\mathcal{W}}(\mathbf{x}_\Delta|M_t^{\mathrm{mem}})$. This expectation represents the average amount of knowledge gained by measuring Δ, given prior knowledge in M_t^{mem}. We obtain:

$$\begin{aligned}\sum_{\mathbf{x}\subseteq\Delta} P_{\mathcal{W}}(\mathbf{x}|M_t^{\mathrm{mem}})IQ_{t+1}^{\mathrm{agent}} &= -\sum_{\mathbf{x}'\subseteq\mathcal{U}',\,\mathbf{x}\subseteq\Delta} P_{\mathcal{W}}\left(\mathbf{x}'\mathbf{x}|M_t^{\mathrm{mem}}\right)\log\frac{P_{\mathcal{W}}\left(\mathbf{x}'|M_t^{\mathrm{mem}}\mathbf{x}\right)}{P_\theta\left(\mathbf{x}'|M_t^{\mathrm{mem}}\mathbf{x}\right)} \\ &= IQ_t^{\mathrm{agent}} + \sum_{\mathbf{x}\subseteq\Delta} P_{\mathcal{W}}(\mathbf{x}|M_t^{\mathrm{mem}})\log\frac{P_{\mathcal{W}}(\mathbf{x}|M_t^{\mathrm{mem}})}{P_\theta(\mathbf{x}|M_t^{\mathrm{mem}})}\end{aligned} \tag{12.5}$$

The second term is the relative entropy of the conditional probability distribution of $\mathbf{x}_\Delta$ conditioned on M_t^{mem}, which is always non-negative. Therefore, on average, IQ_t^{agent} is non-decreasing as M_t^{mem} acquires new knowledge over time. Note that $IQ_{t+1}^{\mathrm{agent}}$ can be further increased by leveraging the newly acquired knowledge to optimize θ within M_t^{wm}.

Interestingly, the expected gain in intelligence at time t is determined by the discrepancy between the actual distribution $P_{\mathcal{W}}(\mathbf{x}|M_t^{\mathrm{mem}})$ and the model-predicted distribution $P_\theta(\mathbf{x}|M_t^{\mathrm{mem}})$. In other words, the rate of intelligence growth in Figure 12.1 is higher when the new measurement result is more unexpected. This observation identifies scientist agents

[859] as a special type of curiosity-driven agent [869], prioritizing exploration over exploitation to expand the frontiers of knowledge for deeper understanding of nature. Unlike agents that leverage existing knowledge to achieve predefined objectives, curiosity-driven agents can learn without extrinsic rewards [387, 870] (see Section 5.3 for details), enabling discoveries beyond human-planned search spaces and revealing knowledge in unexplored domains. This potential also underscores the importance of equipping curiosity-driven agents with fundamental perception and action tools that can be transferred to explore new knowledge domains.

12.1.3 Intelligence Evolution Strategies

The strategy for expanding known information determines how quickly an agent's intelligence evolves. For a given knowledge base M_t^{mem}, the parameter θ can be optimized over a space of world models Θ characterized by the architecture of M_t^{wm}. The optimal agent is the one that minimizes $D_{\text{K}}(\theta, M_t^{\text{mem}})$, thereby maximizing IQ_t^{agent}:

$$\theta_{\text{K},t}^* \equiv \arg\sup_\theta IQ_t^{\text{agent}} = \arg\inf_\theta D_{\text{K}}(\theta, M_t^{\text{mem}}) \tag{12.6}$$

and

$$D_{\text{K},\Theta}^{\min}(M_t^{\text{mem}}) \equiv D_{\text{K}}(\theta_{\text{K},t}^*, M_t^{\text{mem}}) \tag{12.7}$$

Here, $D_{\text{K},\Theta}^{\min}(M_t^{\text{mem}})$ represents the minimum unknown after learning from M_t^{mem} for this family of models, quantifying the expressive limitations of Θ. As shown in Figure 12.1, $D_{\text{K},\Theta}^{\min}(M_t^{\text{mem}})$ forms the envelope of the family of functions $D_{\text{K}}(\theta, M_t^{\text{mem}})$, where θ ranges over Θ.

For a given model family Θ, $D_{\text{K},\Theta}^{\min}(M_t^{\text{mem}})$ measures the best possible prediction of residual unknowns in addressing the target scientific problem based on M_t^{mem}. In other words, the knowledge content in M_t^{mem} is captured by $D_{\text{K},\Theta}^{\min}(M_t^{\text{mem}})$. One can prove that $D_{\text{K},\Theta}^{\min}(M_t^{\text{mem}})$ is monotonically non-increasing as M_t^{mem} expands, since it forms the envelope of a family of non-increasing functions $D_{\text{K}}(\theta, M_t^{\text{mem}})$. This expansion process is tied to how the agent acts and gains information, driven by M_t^{wm}, which determines the optimal expansion and executes it through the action $a_t \in \mathcal{A}$ at time t (see Table 1.2).

During knowledge discovery, different strategies can be employed to expand M_t^{mem}. The optimal expansion strategy is the one that results in the steepest decrease of $D_{\text{K},\Theta}^{\min}(M_t^{\text{mem}})$. For instance, in Figure 12.1, we illustrate two strategies for expanding M_t^{mem}, denoted as ${}^1M_t^{\text{mem}}$ and ${}^2M_t^{\text{mem}}$. The first strategy, ${}^1M_t^{\text{mem}}$, represents random exploration, while the second, ${}^2M_t^{\text{mem}}$, follows a hypothesis-driven approach [871] in which the agent first formulates a hypothesis about the underlying mechanism of the target problem and then designs an experiment to justify or falsify this hypothesis [749]. In practice, experimentalists typically adopt the hypothesis-driven strategy because it enables them to guide the expansion of M_t^{mem} in a way that maximizes the reduction of $D_{\text{K},\Theta}^{\min}(M_t^{\text{mem}})$, subject to resource constraints. This approach is generally more efficient than random exploration for expanding M_t^{mem}, leading to $D_{\text{K},\Theta}^{\min}({}^2M_t^{\text{mem}})$ descending faster than $D_{\text{K},\Theta}^{\min}({}^1M_t^{\text{mem}})$.

In general, the knowledge discovery process proceeds iteratively, repeatedly optimizing the world model parameter θ to approach $\theta_{\text{K},t}^*$ and expanding M_t^{mem} in a rational manner to accelerate the decrease of $D_{\text{K},\Theta}^{\min}(M_t^{\text{mem}})$. The ideal state is achieving epistemic completeness, i.e., $D_{\text{K},\Theta}^{\min}(M_t^{\text{mem}}) = 0$, meaning zero discrepancy between the agent's prediction and the real-world phenomena. However, for a specific agent, a discovery bound may exist, where $D_{\text{K},\Theta}^{\min}(M_t^{\text{mem}})$ approaches zero but remains positive. These discrepancies arise from practical constraints and the limitations of Θ, $\mathcal{A}$, and other design spaces of the agent [872]. Achieving a low discovery bound requires designing an adaptive world model architecture, an efficient knowledge expansion strategy, and a sufficient action space.

12.2 Agent-Knowledge Interactions

Typical forms of scientific knowledge include observational knowledge (e.g., experimental measurements, computational results), methodological knowledge (e.g., experimental methods, computational techniques, protocols), and theoretical knowledge (e.g., theories, laws, predictive models). These forms of knowledge can contribute to scientific understanding as long as they consist of data and information processed in a way that affects the probability distribution of unknown information $P_\theta(\mathbf{x}_{\text{U}}|M_t^{\text{mem}})$, reduces $D_{\text{K}}(\theta, M_t^{\text{mem}})$, and facilitates decision-making.

In principle, external scientific knowledge has been shown to be useful in improving agent performance in reasoning and decision-making [873, 874]. However, the scope of this survey lies in how agents can autonomously discover and utilize knowledge to enhance themselves. Scientific knowledge discovery workflows typically involve hypothesis generation, protocol planning, conducting experiments and computations, analyzing data, deriving implications, and

revising hypotheses—often as part of an iterative cycle. An agent that can perceive, learn, reason, and act has the potential to drive such workflows in an autonomous manner, for example by using application programming interfaces (APIs) to interact with physical instruments to acquire scientific knowledge and iteratively enhance its knowledge base (Figure 12.2). The agent will use the acquired knowledge to update its mental states M_t to make better decisions when interacting with the world $\mathcal{W}$. We will now highlight three scenarios where agents discover scientific knowledge and enhance themselves.

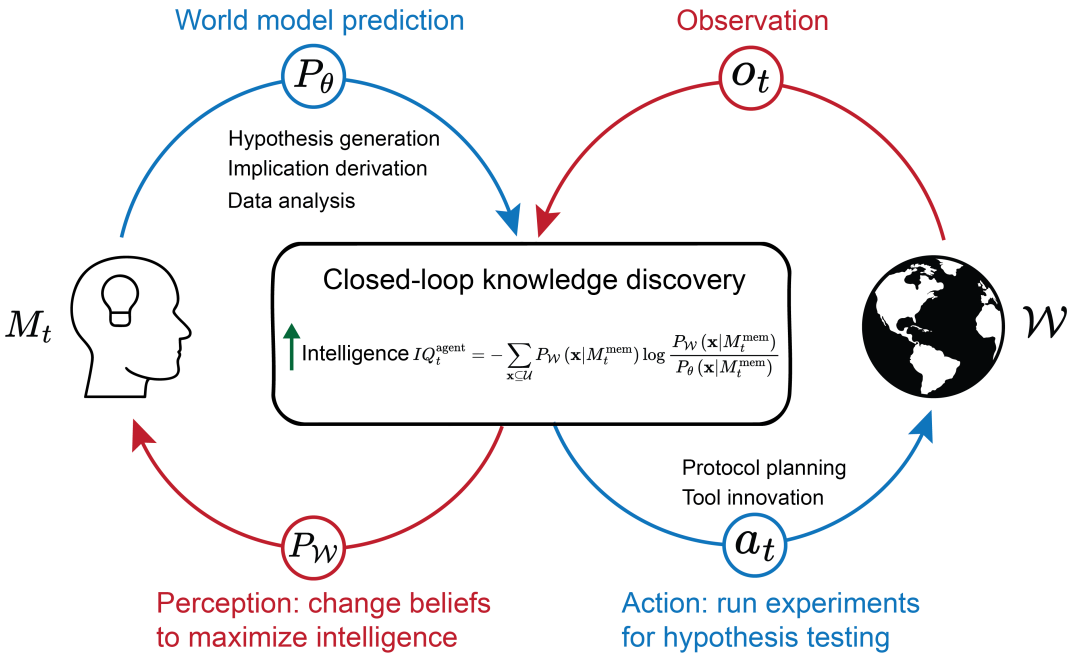


Figure 12.2: **Closed-loop knowledge discovery for sustainable self-evolution.** The agent aims to iteratively enhance its intelligence IQ_t^{agent} through hypothesis generation and testing, as well as through data analysis and implication derivation. When interacting with the physical world W, the agent generates hypotheses as an explicitly or implicitly predicted distribution (P_θ) of unknown information, takes actions (a_t) for hypothesis testing, observes experimental results (o_t), and updates beliefs based on perception of the real-world distribution (P_W). When not interacting with W, the agent distills knowledge from existing data and premises, updating mental states M_t directly. Inspired by Figures 2.3 and 2.5 in [864].

12.2.1 Hypothesis Generation and Testing

Hypothesis generation and testing (Figure 12.2) is a critical application of agents in autonomous scientific discovery, as it has the potential to enable outside-the-box innovations [749]. In essence, hypothesis generation is the formation of potential rules that govern data distribution—ranging from single observations to large datasets—pertaining to unobserved scientific phenomena. According to Sir Karl Popper, a scientific hypothesis must be falsifiable [875, 876]; in this discussion, we define a hypothesis that survives falsification as a *justified true hypothesis* [877, 860]. Typically, scientists test hypotheses by conducting experiments to either justify or falsify them. A hypothesis is considered more valuable if it is broad enough to explain a wide range of data and is highly likely to be true.

To tackle a scientific problem, the agent formulates one or a small number of high-value hypotheses based on its mental state M_t, which contains only incomplete information about the partially observable world $\mathcal{W}$. After testing through experiments or computations, a *justified true hypothesis* becomes instructive knowledge, expanding M_t^{mem} in a way that rapidly minimizes $D_{\text{K},\Theta}^{\min}(M_t^{\text{mem}})$. Hence, generating and testing high-value hypotheses can quickly promote knowledge discovery and increase IQ_t^{agent}. In this scenario, the agent employs the learning function, L, to process observations from hypothesis testing, o_t, into knowledge and update its mental states M_t.

Generating physically meaningful hypotheses is a key step. The agent typically uses LLMs along with collaborative architectures and domain knowledge for hypothesis generation [878]. Si et al. [742] conducted a large-scale human study involving over 100 NLP researchers, and found that LLM-generated ideas were rated as more novel ($p < 0.05$) than human expert ideas, albeit slightly weaker in feasibility. Ghafarollahi et al. [743] developed SciAgents, which generates

and refines materials science hypotheses to elucidate underlying mechanisms, design principles, and unexpected properties of biologically inspired materials. Based on large-scale ontological knowledge graphs, SciAgents samples a viable path between concepts of interest, formulates a pertinent hypothesis, and expands it into a full research proposal with detailed hypothesis-testing methods and criteria. It employs two dedicated agents to review, critique, and improve the proposed hypothesis, but does not include the step of hypothesis testing through actual experiments. Similarly, Su et al. [879] and Baek et al. [880] proposed leveraging teamwork—such as collaborative discussions and agent critics—to produce novel and effective scientific hypotheses. In addition, Gower et al. [881] introduced LGEM$^+$, which utilizes a first-order logic framework to describe biochemical pathways and generate 2,094 unique candidate hypotheses for the automated abductive improvement of genome-scale metabolic models in the yeast *S. cerevisiae*.

Hypotheses only become knowledge after being justified through computational or experimental observations. Lu et al. [745] introduced the AI Scientist, a system designed for fully automated scientific discovery. The AI Scientist can conduct research independently and communicate its findings, as demonstrated in three machine learning subfields—diffusion modeling, transformer-based language modeling, and learning dynamics. It generates original research ideas, writes code, performs computational experiments, visualizes results, drafts complete scientific papers, and even simulates a peer review process for evaluation. For instance, it proposed the hypothesis that "adaptive dual-scale denoising can improve diffusion models by balancing global structure and local details in generated samples," which was justified through image generation tests on four 2D datasets. Similarly, Schmidgall et al. [746] developed the Agent Laboratory to autonomously carry out the entire research process, including literature review, computational experimentation, and report writing. They evaluated Agent Laboratory's capability for knowledge discovery by addressing five research questions in computer vision and natural language processing, achieving an average human-evaluated experiment quality score of 3.2 out of 5. In addition, Tiukova et al. [744] developed Genesis, an automated system capable of controlling one thousand μ-bioreactors, performing mass spectrometry characterization, accessing a structured domain information database, and applying experimental observations to improve systems biology models. Genesis can initiate and execute 1,000 hypothesis-driven closed-loop experimental cycles per day. Using a similar approach, the Genesis team has advanced the yeast (*S. cerevisiae*) diauxic shift model, outperforming the previous best and expanding its knowledge by 92 genes (+45%) and 1,048 interactions (+147%) [882]. This knowledge also advances our understanding of cancer, the immune system, and aging. Similarly, Gottweis et al. [749] introduced the AI co-scientist, which autonomously generates and refines novel research hypotheses, with *in vitro* validation in three biomedical areas: drug repurposing, novel target discovery, and mechanisms of bacterial evolution and antimicrobial resistance.

Discovered knowledge enhances the agent's mental states, such as M_t^{mem}, M_t^{wm}, and M_t^{rew}. Tang et al. [747] developed ChemAgent, which improves chemical reasoning through a dynamic, self-updating memory, M_t^{mem}. ChemAgent proposes hypothetical answers to chemistry questions in a development dataset, evaluates them against the ground truth, and simulates the hypothesis-testing process used in real-world research. Correct answers are then stored as knowledge in its memory to support future chemistry question answering. This self-updating memory resulted in performance gains of up to 46% (with GPT-4) when ChemAgent was applied to four chemical reasoning datasets from SciBench [883]. Wang et al. [884] introduced Molecular Language-Enhanced Evolutionary Optimization (MOLLEO), which iteratively proposes hypotheses for modifying candidate drug molecules in M_t^{mem}, evaluates their drug-likeness and activity, and updates the candidates in M_t^{mem} to enhance drug discovery. Similarly, Jia et al. [885] developed LLMatDesign, which employs hypothesis-guided structure generation and a self-updating M_t^{mem} to design inorganic photovoltaic materials, whose ideality is defined by matching the target band gap and having the most negative formation energy.

Sim et al. [748] introduced ChemOS 2.0, which orchestrates closed-loop operations in chemical self-driving laboratories (SDLs). ChemOS 2.0 integrates *ab initio* calculations, experimental orchestration, and statistical algorithms for the autonomous discovery of high-performance materials. A case study on discovering organic laser molecules demonstrates its capabilities. It employs a Bayesian optimizer, Altas, as its world model M_t^{wm} to predict the optical properties of hypothetical molecules—specifically Bis[(N-carbazole)styryl]biphenyl (BSBCz) derivatives—including gain cross section and spectral grain factor. Based on these predictions, ChemOS 2.0 recommends molecules with a higher probability of success in the experimental campaign. It then utilizes an optical characterization platform and the AiiDA software package to measure and simulate the properties of test molecules. The results are used to update M_t^{wm}, improving the accuracy of future experimental predictions.

Hysmith et al. [886] published a perspective highlighting the crucial role of reward function design in developing forward-looking workflows for SDLs. Agents can be highly effective at solving POMDP problems in simulated environments, such as computer games or simulations, but often struggle with real-world applications. A well-defined reward function is essential for iterative self-evolution. However, in many real-world scientific research problems, reward functions are ill-defined or absent at the end of experimental campaigns due to the lack of direct measurements,

the complexity of experimental results, and the need to balance multiple objectives. The discovery of new knowledge can serve as a valuable resource for refining M_t^{rew}, guiding hypothesis exploration and experimental data collection.

12.2.2 Protocol Planning and Tool Innovation

The capability to plan experimental protocols and optimize tool usage enables the agent to solve complex scientific puzzles within the autonomous discovery loop. As introduced in Section 9.4, the agent can systematically evaluate and refine its approach to selecting, invoking, and integrating available tools—and even develop new tools tailored to specific task requirements. While optimized protocols and tool usage do not directly reduce $D_{\text{K}}(\theta, M_t^{\text{mem}})$, they enhance execution efficiency and effectiveness in refining the probability distribution of unknown information, $P_\theta\left(\mathbf{x}_{\text{U}}|M_t^{\text{mem}}\right)$, thereby accelerating knowledge discovery. In this scenario, the agent leverages the reasoning function R to translate its evolving mental states M_t, continuously updated with new knowledge, into real-world actions a_t for more effective and faster hypothesis testing (Figure 12.2).

Scheduling and orchestrating the selection and recombination of existing tools is critical. Scientific experiments typically depend on diverse instruments for analyzing reaction products, with decisions rarely rely on just one measurement. Effectively utilizing necessary instruments without wasting resources and time requires the agent to learn to use tools in an integrated and adaptive manner. Dai et al. [750] designed a modular workflow that integrates mobile robots, an automated synthesis platform, and various characterization instruments for autonomous discovery. They exemplified this system across three domains: structural diversification chemistry, supramolecular host-guest chemistry, and photochemical synthesis. The mobile robot follows a synthesis-analysis-decision cycle to mimic human experimental strategies, autonomously determining subsequent workflow steps. It selects appropriate instruments, such as the Chemspeed ISynth platform for synthesis, a liquid chromatography-mass spectrometer (UPLC-MS) for measuring mass spectra corresponding to chemical peak signals, and a benchtop nuclear magnetic resonance spectrometer (NMR) for tracking chemical transformations from starting materials to products.

Beyond individual laboratories, tool orchestration is essential for delocalized and asynchronous scientific discovery. Strieth-Kalthoff et al. [751] demonstrated a closed-loop integration of five materials science laboratories across three continents, advancing delocalized and democratized scientific discovery. These five laboratories have varying strengths—for example, the University of British Columbia specializes in continuous preferential crystallization, while Kyushu University excels in thin film fabrication and characterization. Strieth-Kalthoff et al. employed a cloud-based experiment planner to continuously learn from the incoming data and effectively prioritize informative experiments across the five laboratories, resulting in the discovery of 21 new state-of-the-art materials for organic solid-state lasers.

Moreover, the agent can optimize existing tools and even create new ones to enhance its capabilities. Swanson et al. [752] developed the Virtual Lab, an AI-driven research environment that facilitated the design and experimental validation of new SARS-CoV-2 nanobodies. Within the Virtual Lab, AI agents conduct scientific discussion in team meetings and execute specialized tasks in individual sessions. One key agenda for the agents was developing tools to aid in the design of nanobody binders [887], including: (1) a sequence analysis tool that ranks candidate point mutations using log-likelihood ratios from the ESM protein language model [888]; (2) a structure evaluation tool that extracts interface pLDDT scores from AlphaFold-Multimer predictions [889], offering a proxy for antibody-antigen binding affinity; and (3) an energy estimation tool built on Rosetta [890] to quantify binding strength between nanobody variants and the spike protein. These agent-generated tools enabled the Virtual Lab to discover two novel nanobodies with enhanced binding to the JN.1 or KP.3 SARS-CoC-2 variants, while preserving strong affinity for the ancestral viral spike protein.

12.2.3 Data Analysis and Implication Derivation

Although most knowledge discovery processes rely on generating hypotheses and testing them in the real world—where observations o_t are essential—a significant portion of knowledge can be derived purely through internal actions such as iterative reasoning and deep thinking, which are common in theoretical disciplines. For example, all theorems in Euclidean geometry can be deduced from just five axioms, but these theorems do not explicitly exist in the mental state before they are derived. Given all necessary premises, such as Euclid's five postulates, the true probability of a hypothesis may remain elusive. However, using deductive and inductive reasoning to draw implications from known premises and data can help either justify or falsify hypotheses, thus reducing $D_{\text{K}}(\theta, M_t^{\text{mem}})$ and enhancing IQ_t^{agent} (Figure 12.2). In this scenario, the agent employs the cognition function C to use prior mental states M_{t-1} and internal actions a_t to derive new knowledge and update mental states to M_t.

Deductive reasoning enables knowledge derivation through logic. Trinh et al. [753] developed AlphaGeometry for the forward deduction of new mathematical theorems based on existing theorems in Euclidean plane geometry. AlphaGeometry employs a neural language model to construct auxiliary points in plane geometry problems and

integrates specialized symbolic engines to exhaustively deduce new true statements, thereby expanding the joint closure of known truths. By leveraging this expanded closure, it alternates between auxiliary constructions and symbolic reasoning engines to uncover further implications. AlphaGeometry demonstrated remarkable performance on a test set of 30 recent Olympiad-level problems, solving 25—more than double the 10 problems solved by the previous best method—and coming close to the level of an average International Mathematical Olympiad (IMO) gold medalist.

Inductive reasoning enables knowledge derivation through pattern recognition and statistical learning. Liu et al. [754] introduced the Team of AI-made Scientists (TAIS) to simulate the role of a data scientist for streamlined data analysis. TAIS decomposes a complex data analysis problem into different computational tasks, including coding, self-critique, and regression analysis, to extract meaningful insights from complex datasets. When applied to identifying disease-predictive genes, TAIS achieved an overall success rate of 45.73% on a benchmark dataset containing 457 genetic questions. Ideally, the extracted insights should be logically sound; otherwise, they must be discarded to ensure only accurate findings are safely integrated into mental states. However, limitation in data coverage and the implementation of analysis algorithms may lead to hallucinated insights, underscoring the need for reliable data analyzers and reasoning tools to prevent over-analysis.

12.3 Technological Readiness and Challenges

The self-evolution of agents, which in turn drives the advancement of human knowledge, is promised by their early success in the innovation cycle. This cycle involves generating meaningful hypotheses, designing real-time testing protocols, coordinating various experimental and computational tools, analyzing data, deriving implications, and engaging in self-reflection. However, achieving fully autonomous self-evolution remains a significant challenge, given the current technology readiness levels (TRLs) of three fundamental capabilities: real-world interaction, complex reasoning, and the integration of prior knowledge. Further technological progress is required to improve the cycle of self-driven innovation.

12.3.1 Real-World Interaction Challenges

Agents interact with the real world primarily through application programming interfaces (APIs). While numerous demonstrations [891] have shown their strong capability to use various APIs, a significant bottleneck in autonomous knowledge discovery remains: the lack of APIs that allow agents to directly execute tasks in a physical laboratory. Physical APIs—interfaces that enable direct control of lab equipment—are far less abundant than computational APIs due to the significant investment of time, expertise, and cost required to develop them. Although existing autonomous laboratories have shown promise, they remain in an early developmental stage (typically TRL 4–6), where straightforward replication or scale-up is challenging. Consequently, building further systems or broadening their application across additional scientific domains still requires substantial customization to address domain-specific needs, along with specialized expertise.

Two key tasks are essential for enabling real-world interaction: *operating lab devices* and *transferring samples between devices*. Seamless integration of physical hardware and experimental samples is crucial to maintaining uninterrupted workflows. However, most experimental instruments are originally designed for human operation. Making them accessible to agents requires extensive efforts across multiple disciplines, including robotics, electrical engineering, mechanical engineering, and software programming. The rising prominence of SDLs is catalyzing the transformation of human-operated devices into agent-accessible systems through APIs. In autonomous labs conducting complex experiments, two parallel and often complementary approaches are commonly adopted to integrate hardware with agentic systems. Both approaches are modular, reconfigurable, and valuable, yet they require ongoing, dedicated development.

Approach 1: API Integration via Direct Device Adaptation. This approach involves equipping individual devices with dedicated mechanical adaptations and I/O controllers, enabling them to receive and execute commands from a central control PC. For example, to achieve solid-state synthesis and structural characterization of inorganic materials, A-lab has implemented 16 types of devices to automate experimental tasks such as powder dosing, heating, and diffraction [892]. This approach allows laboratories to function as fully integrated entities by maximizing device utilization, optimizing space and resources, and enabling bespoke tools. However, it is costly, time-consuming, and requires expert knowledge to prototype or retrofit devices for automation. Large language models (LLMs) have been applied to facilitate access to diverse tools, as illustrated by CACTUS, a Chemistry Agent Connecting Tool-Usage to Science [893].

A more accessible alternative for small teams is the *cloud lab* or *science factory* [894], where responsibility for device engineering shifts from individual laboratories to dedicated user facilities or commercial service providers. For

instance, Boiko et al. [895] demonstrated an autonomous chemical research agent, Coscientist, capable of carrying out cross-coupling Suzuki and Sonogashira reactions using experimental setups at the Emerald Cloud Lab [896]. However, cloud labs offer only a fixed set of pre-built devices optimized for common procedures, posing potential challenges for researchers whose experiments require equipment customization, as integrating non-standard tools may involve a lengthy process of negotiation and development.

Approach 2: Robotic Operation of Experimental Devices. This approach involves using mobile robots or robotic arms to operate existing devices and transfer samples. In many cases, robots can interact with instruments without modification, apart from minor adjustments such as adding specialized actuators, grippers, or holders. For example, Dai et al. [750] employed mobile robots to explore synthetic chemistry. In their autonomous laboratory, mobile robots enable physical linkages between synthesis and analysis devices that are spatially separated, automating sample transportation and handling. In principle, the robots can perform all actions human researchers require in the laboratory. However, current robotic systems still rely on human pre-programming to map the lab layout, define movement trajectories, and register device positions. Handling unexpected or adaptive situations remains a challenge, as pre-programming cannot anticipate every possible state of an experimental setup. Real-time learning and adaptive manipulation are active areas of research that require further technological advancements. In the long term, embodied AI [897] is expected to enhance robotic learning, allowing agents to quickly adapt to new environments and tools.

The two approaches can be combined. For example, Vescovi et al. [894] define a modular laboratory robotics architecture that allows for translating high-level commands into specific operations for a variety of different robotic apparatus and laboratory equipment, and for linking robotic apparatus with other elements of an AI-driven discovery architecture, such as high-performance computing [898]. This architecture has been used to automate experiments in both the biological and physical sciences [899]. Similarly, Fernando et al. [900] integrate a Robotic Operating System 2 (ROS2) compatible robot into the Bluesky experimental orchestration framework. Lo et al. [901] argue for the development and integration of low-cost "frugal twins" of more expensive equipment to facilitate experimentation and democratize access.

12.3.2 Complex Reasoning Challenges

A fundamental philosophical question is whether agents, often powered by LLMs, can truly perform reasoning. By definition, languages models generate outputs by predicting the next token, a mechanism fundamentally different from human reasoning. From an outcome-driven perspective, these input-output systems exhibit reasoning ability phenomenologically, as they produce meaningful outputs compared to a reference system generating arbitrary responses [902]. However, regardless of the perspective taken, this capability remains imperfect—particularly when handling complex logical and numerical problems, which are crucial for scientific knowledge discovery.

Agents and LLMs struggle with hard reasoning tasks. Glazer et al. [903] introduced FrontierMath, a benchmark comprising hundreds of original and challenging mathematics problems covering most major branches of modern mathematics. Evaluation of state-of-the-art LLM-driven agents—including o1-preview (OpenAI), o1-mini (OpenAI), GPT-4o (OpenAI, 2024-08-06 version), Claude 3.5 Sonnet (Anthropic, 2024-10-22 version), Grok 2 Beta (XAI), and Gemini 1.5 Pro 002 (Google DeepMind)—revealed that no model achieved even a 2% success rate on the full benchmark. Chen et al. [873] presented ScienceAgentBench, a benchmark designed to evaluate language agents in data-driven scientific discovery. Among 102 tasks derived from 44 peer-reviewed publications across four disciplines, OpenAI o1 successfully solved only 42.2% of them. Chollet [865] proposed the Abstraction and Reasoning Challenge (ARC) to assesss LLMs' ability to perform abstract inductive reasoning without relying on memorization or external knowledge. Even with careful prompting, GPT-4o correctly solved only 19% of the tasks, far below the $\sim 75\%$ average human performance [904, 905]. Zhu et al. [906] suggested a four-level classification of AI intelligence, including L1 (arbitrating isputes), L2 (auditing a review), L3 (reviewing a paper), and L4 (authoring a paper). They classify the current state-of-the-art LLM-driven agents as approaching L2-level capabilities. To enhance agents' reasoning abilities, researchers have introduced techniques such as chain-of-thought [907], tree-of-thoughts [72], and [70]. Although new methods continue to emerge, as discussed in Section 2.2, further advancements in reasoning capacity remain crucial for achieving reliable causal inference in scientific research.

Agents and LLMs also struggle with quantitative and symbolic problems. For example, GPT-4 and GPT-3.5 often struggle with reliably performing complex arithmetic such as multiplying $12,345 \times 98,765$, or translating IUPAC chemical names into accurate molecular graphs [908, 697]. A common approach to overcoming these limitations is to use external tools rather than relying on the LLM itself for reasoning. In mathematical problem-solving, for example, tools like symbolic solvers are preferred over direct LLM inference [753]. However, this mitigation does not resolve the intrinsic deficiency in numerical understanding, which poses a potential risk to scientific reasoning. Moreover, Yu et al. [909] found that tool-augmented LLMs do not consistently outperform base LLMs without tools in chemistry problem-solving. For instance, for *specialized* chemistry tasks, such as synthesis prediction, augmenting

LLMs with specialized tools can boost the performance substantially; however, tool augmentation is less effective for *general* chemistry questions, such as those in exams, where no specific tools can directly solve a given question. In these scenarios, an agent's ability to reason correctly by using multiple pieces of chemistry knowledge becomes more important.

The preceding discussion emphasizes the importance of developing robust methodologies for evaluating AI agents as scientific research assistants, a topic discussed at length by Cappello et al. [910].

12.3.3 Challenges in Integrating Prior Knowledge

Prior knowledge is a crucial factor for higher intelligence. As discusses in Section 12.1, the agent's prior knowledge, M_t^{mem}, helps decrease $D_{\text{K}}(\theta, M_t^{\text{mem}})$ and increase the agent's intelligence, IQ_t^{agent}. Human-led scientific discoveries frequently achieve breakthroughs with relatively small datasets, thanks to the vast prior knowledge humans possess. The start-of-the-art LLMs that power autonomous agents are trained on nearly all publicly available textual data, including websites, books, and other sources, thereby encompassing most common knowledge as well as publicly accessible specialized knowledge. However, achieving an agent that can seamlessly integrate all existing human knowledge remains a significant challenge.

At least three types of knowledge sources may not be included in LLM pre-training: (1) Paywalled or unpublished knowledge, including non-open-access publications, industry-specific data, and failed experiments [911]. They are often not accessible to public models despite their potential value in refining domain-specific insights. (2) Empirical knowledge. Heuristic decisions by experts are often effective, particularly in scenarios where no existing data is available for a new problem. However, large amounts of expert heuristics are typically not accessible as textual data. (3) Contextual or situational knowledge. Knowledge related to real-world conditions, such as safety protocols in chemical reactions or equipment handling, is often absent from pre-trained models but is essential for practical applications.

Additionally, integrating diverse knowledge sources presents challenges in reconciling conflicting information. For example, OpenAI's Deep Research [912] actively gathers online information and performs multi-step reasoning, achieving state-of-the-art performance on Humanity's Last Exam and the GAIA benchmark. However, it still struggles to distinguish between authoritative information and rumors and exhibits limitations in confidence calibration, often misrepresenting its level of certainty [912]. Establishing a system to assess the levels of evidence [913] of different knowledge fragments—such as quantifying reliability and verifying references—may be necessary for effective knowledge fusion.

Part III

Collaborative and Evolutionary Intelligent Systems

The concepts of **collaboration** and **evolution** lie at the heart of intelligent multi-agent systems (MAS). Inspired by biological ecosystems and human societal dynamics, these systems leverage collective intelligence to solve complex challenges that exceed the capabilities of individual agents [914]. Human societies exemplify how cooperation, specialization, and distributed decision-making significantly enhance collective problem-solving effectiveness. Similarly, MAS adopts these strategies, integrating specialized agents to address intricate tasks collaboratively. The foundational principle of collective intelligence – the "Wisdom of Crowds" by [915] – suggests diverse, independent agents often yield superior decisions compared to solitary experts, directly underpinning the design philosophy of MAS. Cognitive theories, such as Minsky's society of mind [17] and the theory of mind [916, 917], further reinforce this paradigm by proposing that intelligence emerges from structured interactions among specialized units.

Recently, advancements in large language models (LLMs) have introduced new possibilities for collaborative and evolutionary multi-agent systems (LLM-MAS). Benefiting from powerful reasoning, planning, and decision-making capabilities, these models enable the creation of sophisticated MAS architectures mirroring the cooperative and adaptive characteristics found in human societies. Agents within LLM-MAS often assume distinct identities and roles, reflecting human-like division of labor and specialized collaboration. By embracing structured communication, dynamic knowledge sharing, and coordinated decision-making, these systems emulate human social dynamics to achieve common goals. Moreover, LLM-MAS is inherently evolutionary; agents continuously adapt and improve through interactions, feedback, and iterative learning, resulting in enhanced system performance over time. **Roadmap** In this chapter, we systematically survey the emerging field of LLM-based multi-agent systems, focusing specifically on their collaborative mechanisms and evolutionary capabilities. We first examine how distinct system objectives shape agent roles, behavior patterns, and collaborative strategies in Chapter 13. Next, in Chapter 14, we analyze various communication structures, including interaction protocols that facilitate effective agent-agent and human-agent communication. Additionally, we explore collaborative decision-making methodologies and how agents leverage their unique expertise and perspectives in Chapter 15, and discuss the collective intelligence and evolution mechanism in Chapter 16. Finally, in Chapter 17, we discuss evolutionary processes, highlighting adaptive learning methods, continuous knowledge sharing, and mechanisms for iterative improvement that collectively enhance MAS performance. Through this comprehensive survey, we identify current achievements, discuss existing challenges, and highlight promising research directions for collaborative and evolutionary intelligent systems.

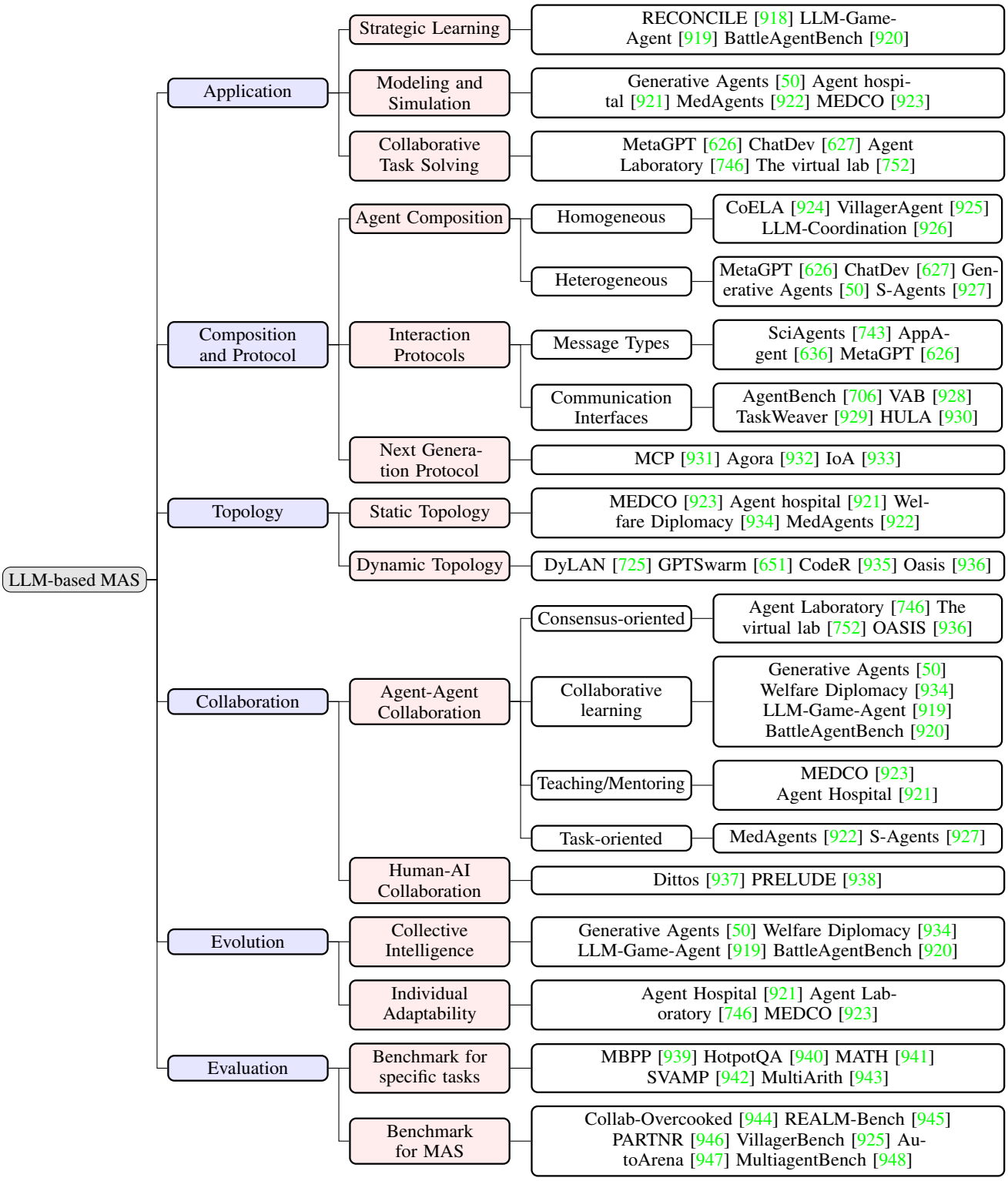


Figure 12.3: Taxonomy of LLM-based Multi-Agent Systems.

Chapter 13

Design of Multi-Agent Systems

In the context of LLM-based multi-agent systems (LLM-MAS), *collaboration goals* and *collaboration norms* serve as foundational elements that shape system behavior, interaction patterns, and overall effectiveness. Collaboration goals specify the explicit objectives agents aim to achieve – whether individually, collectively, or competitively – while collaboration norms define the rules, constraints, and conventions that govern agent interactions within the system. Together, these components establish a robust framework guiding effective communication, coordination, and cooperation among agents.

This section categorizes LLM-MAS into three broad classes based on distinct combinations of collaboration goals and norms: *strategic learning*, *modeling and simulation*, and *collaborative task solving*. Although not exhaustive, these categories cover a wide spectrum of LLM-MAS designs and clearly reflect how system objectives shape agent interactions and outcomes.

- **Strategic Learning** systems embed agents within a game-theoretic context, where agents pursue individual or partially conflicting goals. The interactions can be cooperative, competitive, or mixed, guided explicitly by predefined game rules and interaction norms. This setting often aligns with non-cooperative (strategic) and cooperative concepts in traditional game theory. Please refer to Section 13.1 for details.
- **Modeling and Simulation** contexts focus on agents acting independently, driven by diverse environmental or social factors. Here, interactions emerge organically without necessarily converging on common goals, reflecting the complex dynamics seen in large-scale social or economic simulations. Please refer to Section 13.2 for details.
- **Collaborative Task Solving** emphasizes systematic cooperation among agents to achieve explicitly shared objectives. Agents typically adopt structured workflows, clear role definitions, and highly predefined collaboration norms to synchronize their actions toward collective goals. Please refer to Section 13.3 for details.

In the remainder of this chapter, we elaborate on each category, examining how LLMs enable, influence, and enhance agent behaviors, interactions, and collective intelligence within our scope.

In the following, we examine these categories in detail, highlighting how each leverages the capabilities of large language models to shape agent behaviors and interactions.

13.1 Strategic Learning: Cooperation *vs.* Competition

Strategic learning refers to agents' capabilities to dynamically anticipate, interpret, and influence the actions of other agents within game-theoretic settings—whether competitive, cooperative, or mixed [949]. Agents iteratively adjust their strategies based on new information, commonly modeled using foundational concepts such as Nash equilibria [950], Bayesian games [951, 914, 952], or repeated interactions [953, 954]. With LLMs enabling nuanced linguistic reasoning, strategic learning increasingly integrates "soft" signals – including dialogue, persuasion, and implicit negotiation – thus enriching traditional game-theoretic reasoning frameworks [952, 955, 956, 957].

In economic applications, multi-agent strategic simulations provide valuable insights into market behaviors and negotiation tactics, highlighting both competitive and cooperative dynamics. For example, [958] and [951] demonstrate how LLM-empowered agents can simulate hiring processes, exhibit rational decision-making in controlled economic

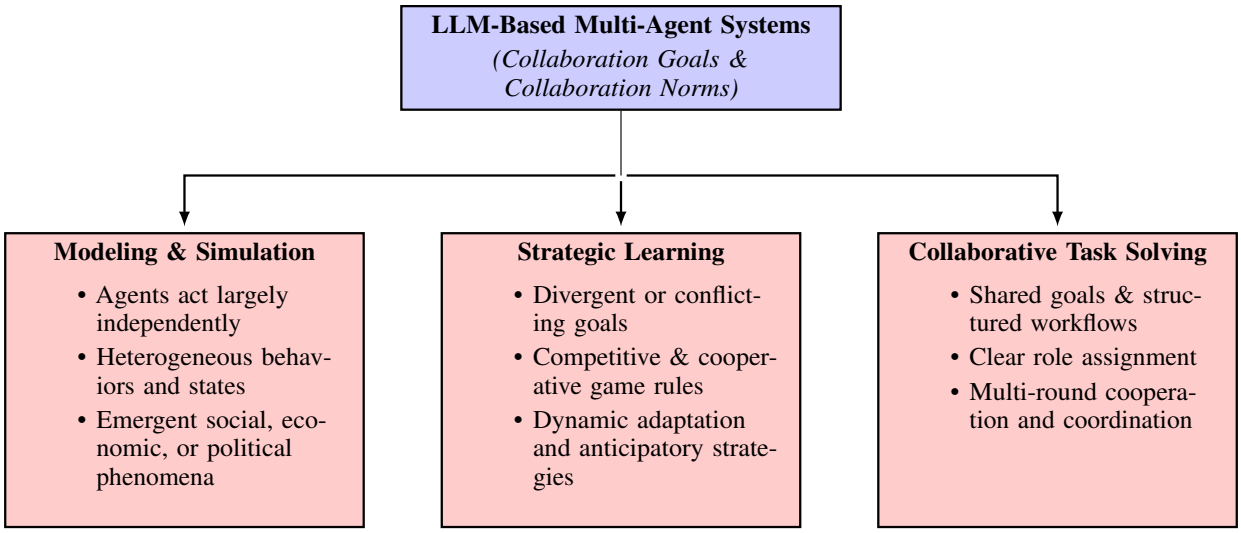


Figure 13.1: An overview of three major collaboration types in LLM-based MAS: *Modeling & Simulation*, *Strategic Learning*, and *Collaborative Task Solving*. Each category is distinguished by how agents' goals and norms are set (independent vs. divergent vs. shared) and how they coordinate.

experiments, and even forecast stock movements. [959] introduces a GPT-4-based competitive environment to illustrate how restaurant and customer agents compete to optimize profits and satisfaction, showcasing realistic bidding and pricing strategies. Meanwhile, [960] investigate Buyer–Seller bargaining in LLM-based negotiations, while [961] use ultimatum game simulations to illuminate policymaking decisions grounded in human-like strategic behavior.

Beyond conventional markets, strategic learning applies broadly wherever resource allocation, alliances, or competitive-cooperative trade-offs are present. Examples include multi-commodity competitions [962, 959], in which agents strategically negotiate terms to maximize individual benefits, or sustainability-focused contexts where agents coordinate resource consumption [963]. In gaming, social deduction games such as Werewolf, Chameleon, Avalon, and Jubensha require agents to manage the complex interplay between deception and collaboration [964, 965, 966, 153, 919, 967, 968, 969, 970]. Studies by [971, 965] highlight LLM-based agents that excel at orchestrating subtle deceit and collaboration, while [967, 972, 968, 969] emphasize adaptive, multi-round strategy in Avalon. [970] further pushes this boundary by showcasing autonomous, multi-agent interactions in the *Jubensha* murder mystery genre, re-creating complex narratives. Similarly, diplomatic simulations ([973] and [974]) employ LLM-based agents to emulate sophisticated geopolitical negotiation and alliance formation dynamics at global scales.

Summary A key advantage of LLM-driven strategic learning lies in effectively combining rigorous game-theoretic logic with natural language reasoning. This fusion enables agents to interpret sophisticated instructions, engage in persuasive dialogue, and adapt more flexibly to novel or unstructured settings. Consequently, LLM-based strategic agents hold significant promise for accurately modeling complex real-world interactions – spanning economic competition, social negotiation, and geopolitical strategy – far more effectively than conventional rule-based or numeric-only approaches.

13.2 Modeling Real-World Dynamics

Modeling and simulation represents another crucial area of application for LLM-based multi-agent systems (LLM-MAS), aiming to replicate complex social, economic, and political phenomena at scale. By utilizing LLMs' sophisticated language understanding and contextual reasoning, these simulations can feature highly heterogeneous agents whose evolving behaviors mirror real-world dynamism. Unlike strategic learning environments that emphasize explicit competitive or cooperative goals, agents in modeling and simulation scenarios operate independently, guided by their domain-specific roles, preferences, and interactions with the simulated environment [975].

In healthcare, for example, [921] introduces *Agent Hospital*, where LLM-powered doctor agents iteratively refine treatment strategies through realistic interactions with virtual patients. This enables researchers to test management protocols, training paradigms, and "what-if" scenarios in a controlled yet realistic setting. Similarly, in economic contexts, [976] present *EconAgents*, leveraging LLM-driven agents to realistically model individual-level behaviors such as employment decisions, consumption patterns, and savings strategies. These agents facilitate expressive macroe-

conomic simulations, surpassing traditional numeric or strictly rule-based methods in adaptability and realism [977]. In addition, political science applications also benefit from this approach. For example, [978] and [977] successfully simulate election processes and policymaking dynamics, revealing how public discourse, candidate strategies, and voter interactions shape real-world political outcomes.

Beyond economics and politics, LLM-based simulation accommodates a variety of social and cultural phenomena. For example, [979] and [255] use simulations of linguistic and emotional propagation in social networks to investigate how opinions, beliefs, or sentiment clusters form online. Research by [980] explores how opinion dynamics evolve under various topological and interaction patterns, while [981] examines the conditions under which fake news spreads or stalls in heterogeneous agent populations. Large-scale simulation platforms such as GenSim [982] and OASIS [936] push the boundary further by scaling to tens of thousands or even millions of user agents, thus enabling the study of emergent group behaviors and systemic effects—such as viral information diffusion, echo-chamber formation, or group polarization—under realistic constraints.

Summary The strength of LLM-based simulation lies in capturing both the structural dynamics (e.g., network topology or institutional rules) and the cognitive or linguistic nuances that drive real-world behavior. By embedding language-based reasoning into agent models, researchers can examine complex social processes—like persuasion, framing, or cultural transmission—that would be difficult to capture through purely numeric or rule-based approaches.

13.3 Collaborative Task Solving with Workflow Generation

Collaborative task solving orchestrates multiple agents toward a clearly defined objective through structured workflows. In contrast to strategic learning (which may involve competing interests) or open-ended modeling and simulation (where agents act independently), collaborative agents function as part of a unified problem-solving pipeline. Agents typically follow clearly defined roles (e.g., "Planner", "Implementer", or "Evaluator") and stage-based processes to ensure efficient and accurate task completion.

Systems such as MetaGPT [626], CAMEL [848], Communicative Agents [983], and frameworks described in [924] exemplify how clearly defined roles, responsibilities, and decision flows allow LLM-based agents to coordinate effectively. A typical workflow might involve one agent analyzing a problem statement, another proposing a solution outline, a third implementing partial solutions, and a fourth verifying correctness. Communication among these agents is often carried out through iterative rounds of natural language "dialogue", leveraging the inherent language-generation strengths of LLMs. This structured approach also proves beneficial for scaling to more ambitious projects, as sub-tasks can be delegated to specialized agents with domain-specific prompts or training.

Recently, collaborative task-solving systems have been explored extensively in software development scenarios (e.g., multi-agent coding, debugging, and testing). However, scientific discovery represents a particularly prominent and compelling application. For example, the *Agent Laboratory* [746] employs agents in structured scientific workflows: proposing hypotheses, designing experiments, analyzing results, and refining subsequent inquiries, which effectively mirrors the iterative nature of the scientific investigation. Similar multi-agent designs can be adapted to tasks such as literature review, policy drafting, or large-scale data analysis, using well-defined protocols to maintain coherence and avoid duplication of effort.

Summary Compared to other LLM-based multi-agent paradigms, collaborative task-solving inherently prioritizes clarity and predictability: Each agent's role and objective are predefined, limiting emergent or chaotic behaviors. This structure is particularly advantageous in domains requiring precision, accountability, or sequential decision-making. At the same time, research is ongoing to strike the right balance between structure and flexibility, which ensures that agents have enough autonomy to creatively contribute solutions while adhering to a shared workflow that ultimately guarantees reliable, high-quality task completion.

Discussion The aforementioned three dimensions—*strategic learning*, *modeling and simulation*, and *collaborative task solving*—reflect the breadth of LLM-based multi-agent systems. Each category addresses distinct research questions and real-world applications, leveraging language-based reasoning to tackle challenges that extend beyond the capabilities of conventional, purely numeric, or rule-driven agent designs.

13.4 Composing AI Agent Teams

In MAS, agents are the core units that interact within the system and are critical to its functionality. These agents can be categorized as either homogeneous or heterogeneous, depending on whether they share identical or differing personas, capabilities, and action spaces.

Homogeneous Homogeneous agents that share identical capabilities, action spaces, and observation spaces. Compared to single-agent systems, the primary advantage lies in task parallelization, allowing multiple agents to handle different parts of a task simultaneously and improve overall efficiency. They are often used in simpler, coordinated tasks where uniformity across agents can drive improved performance.

Several studies have applied homogeneous agents to simulate teamwork in games like Overcooked and Minecraft, as well as real-world tasks such as household labor division. [924] proposed a cognitive-inspired modular framework that enables LLM-based agents to communicate through natural language to perform labor division, request assistance from one another, and collaboratively complete object transportation tasks. [984] introduced prompt-based organizational structures into the framework, reducing communication costs between agents and improving team efficiency in household tasks such as preparing afternoon tea, washing dishes, and preparing a meal. Furthermore, several studies [926, 925] have employed multiple LLM-based agents in popular games such as Overcooked and Minecraft to experiment with their ability to cooperate and complete tasks. According to the game settings, these agents are also homogeneous.

Heterogeneous Agent diversity plays a crucial role in improving collaboration outcomes. Research shows that heterogeneity among agents can enhance problem-solving capabilities, as diverse agents bring varied perspectives and skills to the task at hand [985, 986]. Heterogeneity contributes to richer problem-solving strategies and improves overall collaboration in MAS. The heterogeneous characteristics of agents can be reflected in the following dimensions: personas-level heterogeneity, observation-space heterogeneity, and action-space heterogeneity. Note that these heterogeneities are not mutually exclusive—a heterogeneous agent may exhibit one or more of these characteristics.

- *Personas-level heterogeneity.* Refers to diversity in agent profiles, which influences how agents approach problem-solving and interact with one another. Most current LLM-based heterogeneous multi-agent systems fall into this category [987, 627, 50, 970]. For example, in software development, agents may take on personas such as programmers, product managers, or testers. In medical diagnostics, agents may represent cardiologists, oncologists, or paediatricians, each with distinct areas of expertise. The distinct perspectives and expertise of each persona contribute to more robust decision-making. While these heterogeneous agents may share the same action space—such as writing documents [626] (e.g., code, requirement reports, or test reports) or providing diagnostic advice [922]—their personas influence the outcomes of these actions, where role-specific enhancements within multi-agent architectures have shown to significantly streamline and optimize task execution. For instance, a product manager performing the action of writing a document would produce a requirements report, whereas a programmer performing the same action would produce software implementation code [626]. This diversity leads to better decision-making and innovation, especially in complex, multidisciplinary tasks.
- *Observation-space heterogeneity.* In MAS, the ability of agents to perceive and interpret their environment can vary. Observation-space heterogeneity refers to these differences in what agents can observe or perceive within their environment. For example, in the game Werewolf, some agents, like werewolves, can see the identities of their teammates, and the seer can obtain the identity of a designated player, while others, like villagers, cannot see the true identity of any player [971]. Similarly, in the Avalon game, different roles have distinct observation spaces [919, 972], thus influencing the strategies and communications of the players. In these settings, each agent's perceptual ability or observation space is directly linked to their role in the system. In a multi-agent system, this variation in what agents can observe often influences their decision-making, communication, and coordination with other agents.
- *Action-space heterogeneity.* On the other hand, this refers to fundamental differences in the actions agents can perform due to physical or functional constraints. This is particularly relevant in both virtual and physical environments where agents may have different capabilities based on their design or purpose. In the virtual environments of games like Werewolf [965, 971, 966] and Avalon [919, 967], different roles have distinct abilities or skills [971, 919, 972]. For example, in Werewolf, while werewolves may have the ability to communicate secretly with each other, villagers might be limited to voting or observing only. This dynamic requires agents to collaborate based on their unique capabilities and promotes the learning of strategies such as teamwork, trust, and deception in their interactions. Meanwhile, in robotics, agents may exhibit diverse physical capabilities. For instance, as described in [988], some robots lack mobility and can only manipulate objects, while others are specialized for movement but cannot manipulate objects. In such cases, agents with different action spaces must divide tasks effectively, leveraging their specific abilities to take on the parts of the task they are suited for, ultimately collaborating to complete the overall task. This type of heterogeneity requires agents to collaborate and coordinate their actions efficiently, often dividing tasks based on their individual strengths.

Homogeneity to Heterogeneous Evolution In some LLM-based multi-agent systems, agents have the ability to evolve autonomously and continuously adapt through interactions with their environment. Due to the inherent randomness

in both LLM models and the environment, the evolution of these agents often follows different trajectories. This can lead to heterogeneous behaviors emerging over multiple simulations, even when agents initially have homogeneous personas and action spaces. For example, as shown in [989], agents with identical action spaces and personas at the start developed differentiated roles after multiple rounds of interactions with the environment and other agents. Some agents, for instance, specialized in food gathering, while others focused on crafting weapons. Similarly, [990] observed that initially homogeneous agents developed distinct language usage patterns, emotional expressions, and personalities after group interactions. These emergent behaviors demonstrate the possibility of transitions from homogeneous to heterogeneous systems.

13.5 Agent Interaction Protocols

In this section, there will initially be classification of typical kinds of messages, providing a clear view regarding the content and exchange modes for agent interactions. Next, agent-environment, agent-agent, and agent-human communications interface designs will be addressed. Architectural issues and protocol specifications for transparent information exchange will also be addressed. Interface standardization will have a special focus, which is essential for providing interoperability, scalability, and efficiency for multi-agent systems. The section will end with unification of communication protocol discussions, where agent-environment or agent-user interacting design principles and requirements are addressed, as well as providing clarity, consistency, and functional coherence for various applications for LLM-based systems.

13.5.1 Message Types

Structured: Structured messages, either in JSON ([991, 992]), XML ([993, 636]), or as a code ([626, 627, 994]), are a crucial aspect of multi-agent system communication with LLM. The primary advantages of structured messages are their syntactically and semantically defined structure, enabling unambiguous understanding and straightforward parsing. With their lack of ambiguity, they facilitate unerrant information extraction and processing with much less overhead on computation and greater system dependability. For example, JSON and XML can represent specific-task configuration parameters or facilitate data exchange as a machine-readable mode, and messages written as a code can even be executable several times directly, which makes workflow and automation simpler.

Structured messages are particularly well-suited for high-efficiency, deterministic applications. They are useful for sub-task decomposition, sub-task assignment, and coordination among agents for cooperative multi-agent architecture because they explicitly state operational commands. Moreover, as structured messages have a prescribed form, retrieving data as well as storing data is facilitated and system optimization and longitudinal analysis are also feasible.

Unstructured: In contrast, unstructured messages, e.g., natural text ([971, 970, 919]), visual data, e.g., images, videos, and audio signals, e.g., speech, ambient sounds ([995, 996, 762]), have higher information density and representational capability. Such modalities are best suited for communication with nuanced and context-dependent information. Images, for instance, communicate spatial relationships, illumination, and facial expressions, and videos communicate dynamic temporally-organized sequences, e.g., state or behavior changes over time. Similarly, audio signals also communicate not just linguistic information but also paralinguistic information, e.g., tone, emotion, and intonation, which are critical for natural and context-aware interactions.

Unstructured messages are well-adapted for ambiguity tasks, as well as for complex, real-world settings. The fact that they can express abstract ideas as well as affective subtlety, or implicit contextual suggestions, makes unstructured messages well-suited for creative, as well as discovery-oriented, problem spaces. Unstructured data's complexity, however, calls for advanced processing techniques, for example, feature extraction based on deep learning, for one to tap into their full potential. Advances with pre-trained LLMs as well as multi-modal large language models have alleviated these complexities to a large extent, enabling novel applications for unstructured communication within multi-agent systems [533, 513, 997].

Summary: Unstructured and structured messages have complementary roles for multi-agent communication with LLM-based. While structured messages offer accuracy, consistency, and computation efficiency and are appropriate for operational and deterministic operations, unstructured messages offer rich, contextualized representations enabling agents to negotiate vague, creative, highly dynamic situations. Together, these modes offer a foundation for adaptive, effective multi-agent cooperation.

13.5.2 Communication Interface

Agent-Environment Interface LLM-based agents will typically have to act on their environment once or several times in order to perform a range of operations. From the agent's point of view, its output into the environment is something that it would prefer, e.g., a UI click, web request, or a move for a computer graphic's character. Environments differ with regard to what actions they will accept, and so as not have its actions not get executed, the agent must find out what actions are for a specific environment that it is acting within and perform actions that are for a specific task as well as valid for a specific environment. After the agent outputs its chosen action, the agent will have a return from the environment. It will consist of observations if successful, or a feedback on error if there was one. The agent will have to act on this feedback. There are nowadays various types of environments where an agent can act, e.g., operating systems, computer games, database, and e-commerce websites. To make agent-environment interfaces share a common interface and have agents trained on various LLMs plug into various environments with minimal further adaption, various frameworks have been proposed. These frameworks make for easier tests on agents' capability on various executable environments [706].

Agent-Agent Communication In MAS, communication through natural language is predominant. This is likely because large language models possess strong linguistic capabilities due to pretraining on massive natural language corpora. Another possible reason is that, for many tasks, natural language communication is already sufficient to meet the requirements. Based on the type of information exchanged, multi-agent systems can be categorized as follows: *Natural Language-Based Systems* Among LLM-based multi-agent systems utilizing natural language, text-based communication is the most common [922, 924, 987, 970, 998]. There are also some systems that use voice as the medium of communication [996, 762, 999, 1000]. In these systems, agents engage in behaviors such as discussions, negotiations, persuasion, or critique through natural language to achieve their objectives. *Structured Information-Based Systems* Compared to natural language, structured information has characteristics such as higher consistency, lower parsing complexity, and reduced ambiguity, making it more suitable for efficient and low-cost communication between agents [626]. In some implementations, the information exchanged between agents is structured into distinct components to facilitate easier parsing and utilization by the receiving agent. For instance, the exchanged information might include fields specifying the `sender`, `receiver`, `message type`, and instructions on how the recipient should parse or use the `content` [929].

Human-Agent Communication The purpose of developing multi-agent systems is to expand the boundaries of human capabilities and cognition, ultimately serving human well-being. While in some social simulation multi-agent systems, humans primarily exist as observers [50, 1001], most multi-agent systems allow human participation in various forms. During this participation, humans need to communicate with agents, and this communication can take the form of either natural language or structured information [924, 930]. When human-to-agent communication primarily relies on natural language, a single LLM often acts as a hub to parse human natural language into structured information that agents can process more effectively for subsequent operations. This hub LLM can either exist within the multi-agent system or function independently of it. To save time and enhance communication efficiency, humans can also use structured information to communicate with the multi-agent system through programming or similar methods. By following predefined communication protocols, humans can send messages containing the required data to the multi-agent system. The system will then process the messages and data according to its internal logic and return the results. [931]

13.5.3 Next-Generation Communication Protocols

The field of LLM-based agents is still in its infancy. Developers typically design agent architectures and communication mechanisms tailored to specific domains or tasks, including agent-to-environment, agent-to-human, and inter-agent interactions. However, most existing systems lack a unified communication framework, resulting in fragmented, siloed ecosystems. Multi-agent systems, tools, environments, and data sources often operate independently, making it difficult for agents to interoperate or share capabilities. Furthermore, the burden of learning and implementing bespoke protocols falls on humans, and almost all current protocols are manually designed—a labor-intensive process that often lacks semantic flexibility or scalability.

To address these issues, several new agent communication protocols have been proposed, each targeting different aspects of the protocol design stack.

Internet of Agents (IoA) [933] introduces an internet-inspired, instant-messaging-like communication architecture that supports dynamic team formation and task-driven collaboration. Agents register with a central coordination server, which handles identity management and discovery. Communication flows are orchestrated using FSM (Finite State Machine)-based dialogue templates. IoA supports multiple message types, including discussion, task assignment, and triggering mechanisms, and provides structured fields for controlling speaker turns, nested group formation, and

maximum dialogue length. This allows agents to select and adapt message formats to match specific coordination phases, offering flexibility within a fixed schema.

Model Context Protocol (MCP) [931], developed by Anthropic, focuses on enabling LLM agents to access structured tools and data. It adopts a fully centralized approach based on OAuth identity authentication, and interactions are constrained to JSON-RPC 2.0 messages. While it lacks a meta-protocol layer or semantic negotiation capabilities, its simple and rigid architecture makes it a practical choice for tool use cases with well-defined APIs. However, MCP sacrifices flexibility and extensibility, requiring manual registration of supported functions.

Agent Network Protocol (ANP) [1002] aims to achieve full decentralization. Agents identify themselves through W3C-compliant decentralized identifiers (DIDs) and communicate over encrypted peer-to-peer channels. The protocol includes a meta-protocol layer that enables agents to negotiate which application-level protocol to adopt, supporting semantic protocol selection based on agent capabilities. ANP also allows for multi-protocol support at the application layer (e.g., HTTP, JSON-RPC, natural language), providing strong extensibility and decentralization but does not yet explicitly support public protocol reuse.

Agora [932] offers a highly flexible and language-driven protocol mechanism. Instead of registering pre-defined APIs, agents can generate and share Protocol Descriptions (PDs), which are free-text descriptions of communication semantics. Using a large language model, agents can dynamically interpret and execute any PD at runtime. This allows protocols to be created, deployed, and used entirely through language, without any manual registration or configuration. Agora avoids centralized registries and supports decentralized protocol sharing: agents may publish or retrieve PDs from peer-distributed repositories to enable cumulative learning and interoperability across systems.

Summary: As shown in Table 13.1, next-generation agent communication protocols differ along key dimensions such as identity and security mechanisms, meta-protocol negotiation capabilities, application-layer flexibility, and the degree of centralization. A unified, secure, scalable, and dynamic protocol infrastructure—where agents can negotiate and co-create protocols on the fly—is critical for enabling large-scale, interoperable agent ecosystems. While current frameworks such as MCP, ANP, Agora, and IoA represent early but promising steps, protocol design remains a rapidly evolving frontier in the development of intelligent agent systems.

Table 13.1: Comparison of four agent communication protocols (MCP, ANP, Agora, IoA) across identity, negotiation, and execution layers.
PD = Protocol Description; **DID**:Decentralized Identifier; **LLM**:Large Language Model; **FSM**:Finite State Machine.

Layer	MCP	ANP	Agora	IoA
Identity & Security	OAuth-based centralized identity authentication.	DID-based decentralized identity with encrypted channels.	No centralized registration. Identity derived from PD hash.	Agents register with a central server for identity and discovery.
Meta-Protocol Layer	No meta-protocol layer; relies on pre-defined interfaces.	Uses DID document to negotiate and select appropriate protocol via semantics.	LLM interprets PD text to automatically negotiate and deploy communication protocols.	A centralized discovery mechanism combined with FSM-based dialogue flow control.
Application Protocol Layer	Supports only JSON-RPC 2.0.	Supports multiple protocols such as HTTP and natural language.	Allows arbitrary PD-driven protocols with high flexibility.	Task-driven protocol coordination supporting multiple message formats.
Degree of Centralization	Highly centralized architecture.	Fully decentralized.	Decentralized: no registration or fixed ID, with optional peer-to-peer PD sharing.	Highly centralized architecture with a central coordination server.
Protocol Flexibility	Fixed and rigid; hard to adapt beyond JSON-RPC.	Highly flexible with semantic negotiation.	Extremely flexible; any PD can define a new protocol dynamically.	Moderately high flexibility; agents can select and adapt message formats based on task phases and coordination needs.

Table 13.2: Classification framework for LLM-based multi-agent systems, highlighting different aspects of system design, communication, collaboration, and evolution. Below are our abbreviations, for ease of reference: M&S = Modeling & Simulation, CTS = Collaborative Task Solving, SL = Strategic Learning, S-D = Static-Decentralized, S-L = Static-Layered, Hom = Homogeneous, Het = Heterogeneous, T/M = Teaching/Mentoring, C-O = Consensus-Oriented, T-O = Task-Oriented, CL = Collaborative Learning, Dict = Dictatorial, D-B = Debate-Based, CI = Collective Intelligence, Ind = Individual.

Paper	**System Design**	**Communication**			**Collaboration**		**Evolution**
	Category	**Typology**	**Interface**	**Agent Type**	**Interaction**	**Decision**	**Type**
Agent Hospital [921]	M&S	S-D	Text	Het	T/M, C-O	Dict	Ind
Welfare Diplomacy [934]	M&S	S-L	Code, JSON, Text	Hom	CL	Voting	CI
MEDCO[923]	M&S	S-L	Text	Het	T/M, C-O	Dict	Ind
MedAgents[922]	M&S	S-L	Text	Hom	T-O	Dict	CI
Generative Agents [50]	M&S	S-D	Visual	Hom	CL	Dict	Ind
RECONCILE [918]	SL	S-D	Text	Hom	CL	D-B	CI
Agent Laboratory [746]	CTS	S-L	Code, Text	Het	C-O, T-O	Dict	Ind
CoELA[924]	CTS	S-D	Text	Hom	T-O		
The virtual lab [752]	CTS	S-L	Text	Het	C-O, CL	Dict	Ind
SciAgents [743]	CTS	S-L	Text	Het	T-O	Dict	CI
S-Agents [927]	CTS	S-D	Text	Het	T-O, CL	Dict	
GPT-Bargaining [1003]	CTS	S-D	Text	Het	C-O	D-B	CI
FORD [1004]	M&S	S-D	Text	Het	C-O	D-B	CI
MADRA [1005]	CTS	S-D	Text	Het	C-O	D-B	
Multiagent Bench [948]	CTS	S-D	Text	Hom	T-O, CL	D-B	CI, Ind
OASIS [936]	M&S	D	Text	Het	C-O		
S^3 [255]	M&S	S-D	Text	Het	C-O		
FPS [981]	M&S	S-D	Text	Het	C-O		
GPTSwarm [1006]	CTS	D	Code, JSON, Text	Hom	T-O	Dict	CI, Ind
ChatEval [1007]	CTS	D	Text	Hom	T-O	Voting	CI
MetaGPT [626]	CTS	S-L	Code, JSON, Text, Visual	Het	T-O	Dict	CI
AutoAgents [1008]	CTS	D	Text	Het	T-O	C-O	CI
SWE-agent [628]	CTS	D	Text	Hom	T-O	Dict	Ind
AgentCoder [994]	CTS	D	Code, Text	Het	T-O	D-B	CI
MASTER [1009]	CTS	S-L	Text	Hom	T-O	D-B	CI
Reflexion [48]	CTS	D	Text	Het	T-O	D-B	Ind
MACM [1010]	CTS	D	Text, Code	Het	T-O	D-B	CI
Debate [985]	CTS	S-D	Text	Het	C-O	D-B	CI

Chapter 14

Communication Topology

14.1 System Topologies

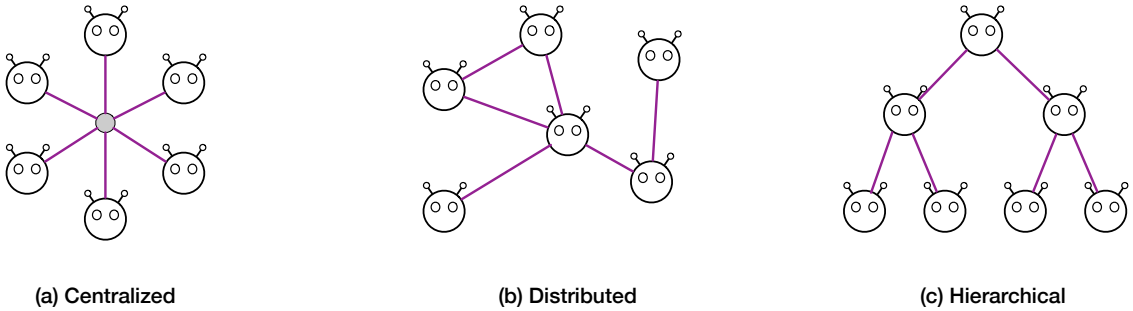


Figure 14.1: Different types of topological structure for multi-agent collaboration.

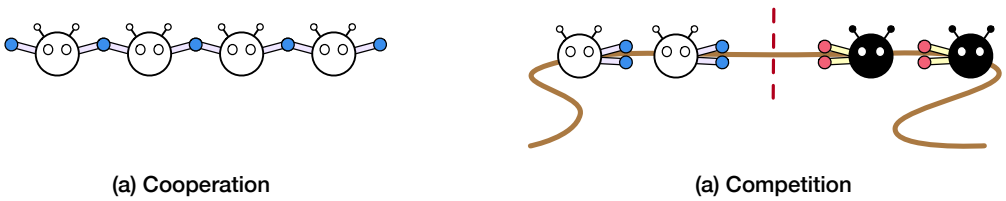


Figure 14.2: Collaborative and competitive agents.

This section examines the interaction typology in LLM-based multi-agent systems (MAS) and its impact on communication, collaboration, and task execution. We first analyze static topologies—where connectivity patterns are fixed by domain knowledge—and then explore dynamic (adaptive) topologies that adjust inter-agent connections based on performance metrics, workload variations, or strategic constraints. We conclude with a discussion of scalability challenges and trade-offs in balancing system cost, performance, and robustness, drawing on recent research in distributed processing, self-organization, and emergent collaborative behaviors.

14.1.1 Static Topologies

Static topologies are defined by predetermined structural patterns that remain largely unchanged during system execution. In these configurations, connections among agents—or between agents and a central coordinator—are established using fixed rules and heuristics, ensuring predictable communication flows and simplified coordination. Three canonical forms are typically considered: layered (hierarchical), decentralized, and centralized architectures.

Layered (Hierarchical) Structures Layered topologies arrange agents hierarchically, with high-level agents coordinating or supervising lower-level ones. This approach mirrors traditional management frameworks—such as Standard

Operating Procedures (SOP) or the Waterfall model—where tasks are decomposed into sequential, well-defined stages. For instance, the AutoAgents [1008] framework assigns roles (e.g., Planner, Agent Observer, and Plan Observer) to synthesize execution plans, while ChatDev [983] leverages hierarchical task decomposition to streamline software development [626, 921, 627]. Although hierarchical structures facilitate debugging, performance monitoring, and modularity, they can create bottlenecks when upper-tier agents are overloaded [1011]. Recent studies in storytelling [1012, 1013, 1014] and data science applications including data cleaning [1015, 1016], visualization [1017, 1018] and auto machine learning [1019, 1020], highlight the trade-off between consistency and the emergence of adaptive real-time behaviors.

Decentralized Structures In decentralized topologies, agents interact on a peer-to-peer basis without a central coordinator, forming networks that are often modeled as chains, rings, small-world, or random graphs [1021, 971]. This structure enhances fault tolerance since the failure of a single agent does not compromise the network. For example, [1022] show that distributing graph reasoning tasks among multiple agents enables scalability beyond the context length limits of individual LLMs. Additionally, [1023] propose decomposition strategies that allow an orchestrating LLM to delegate subtasks effectively. However, maintaining a coherent global state in decentralized systems necessitates sophisticated consensus and synchronization protocols.

Centralized Structures Centralized topologies rely on a master coordinator that gathers information and directs peripheral agents hierarchically. Such a setup allows for better control over handling resources and sharing a global view, such as with culture parks and Lyfe Agents [1024, 1025]. With additional agents, however, a bottleneck at the center node may occur, with increased communication overhead and susceptibility to failures. Current studies on coordinator-agent configurations [971] and research on ensuring autonomy for centralized configurations [1026] point out problems with scalability with consistency. While consistency is guaranteed for centralized architectures, there may not necessarily be flexibility for dynamic adaptation.

Briefly, static topologies have advantages of determinism and predefinition. With pre-defined structural patterns, these systems have predictable communication patterns and effective coordination among agents. Topologies of these structures are typically defined on structural knowledge or static rules, and, as such, they suit domains where workflow for the tasks is static, there are predefined roles, and system requirements are well defined. The second primary advantage is design, implementation, and maintenance ease. With structure predefined, design as well as execution procedures are made simpler, and, as a result, maintenance is a simpler process. Resource handling as well as modularization gets simpler due to well-defined, static structure.

However, static topologies themselves are nonflexible, grounded on pre-specified patterns of connectivity that do not respond to real-time changes. Well suited for a specific purpose at design time but entirely lacking flexibility for reacting to unforeseen challenges, including sudden agent breakdown, varying degrees of task complexity, and system goal modification, static topologies do not have real-time response flexibility potential. Real-time response inflexibility inhibits runtime system reconfiguration and decreases system effectiveness in dynamic settings where circumstances occur. Failure to self-organize and morph according to emerging conditions may equate to inefficiency as well as low system performance, particularly where dynamic or emergent settings are at hand.

14.1.2 Dynamic and Adaptive Topologies

While static topologies provide determinism and predictability—illustrated by static topologies such as hierarchical or centralized ones performing well with stable-task domains and well-defined roles—static topologies do not fit open-ended or novel domains. Real domains, from real-time collaborative plan, to dynamic social simulations, often demand that agents make changes on their patterns of interaction as work continues, available resources vary, or feedback from the environment is received. Such structural tension with adaptative malleability generates dynamic topologies, which, at runtime, recast inter-agent relationships as a response to feedback on performance, workload, or strategic constraints, striking a balance between consistency and responsiveness.

For example, DyLAN framework [725] supports inference-time agent selection through a two-step process: a forward-backward team optimization step with unsupervised Agent Importance Scores, followed by dynamic team reformulation at runtime. Similarly, OPTIMA [1027] optimizes inter-agent connectivity iteratively through a generate–rank–select–train framework, utilizing reward functions as a means for determining a balance among task quality, token efficiency, and readability, with communication actions further optimized through strategies such as Direct Preference optimization. The MAD framework [649] illustrates flexibility through a joint optimization among three prompt phases and structure, with dynamic role assignment (such as verifiers and debate participants) within pruned spaces for structure.

Topological control also becomes tractable through technological advancements. GPTSwarm [651] conceptualizes agents as computation graphs and uses evolutionary strategies and reinforcement learning for adjusting adjacency matrices for optimizing nodes based on feedback for the task. MACNET [1028] uses a directed acyclic graph architecture with supervisory instructors managing edges and executive assistants managing nodes for more complex coordination domains, facilitating adaptive communication through topological ordering and sensitive propagation of output. Application-specific versions also emphasize architecture diversity. Open-world environments have DAMCS [1029], which couples hierarchical knowledge graphs (A-KGMS) with structured communication schemes (S-CS) for cooperative planning as a function of messages passed based on context. AutoAgents [1030] leverages a dynamic drafting-execution pipeline with pre-defined agents jointly sketching out expert teams, a design that's highly effective for creative applications such as novel generation through parallel processing and internal supervision. Noticeably, small-world development within large-scale MACNET [1028] systems corresponds with graph reasoning ideas shown in [1022], where distributed architecture bypasses local limitations of LLM through structured collaboration. In terms of collaborative task solving, several paradigms have emerged that emphasize the role of dynamic topologies. These paradigms include search-based methodologies, LLM-based generation, and configurations utilizing external parameters.

Search-based Methods A number of works adopt search-based methodologies to iteratively optimize communication structures. For example, ADAS [741] employs a Meta Agent Search algorithm that iteratively generates and tests new agent designs within a code space, archiving superior configurations and thereby updating subsequent generation strategies. Similarly, Aflow [773] models each LLM call as a node in a graph and utilizes Monte Carlo Tree Search (MCTS) to dynamically extend and refine the workflow. Other frameworks, such as MAD [1031] and OPTIMA [1027], integrate iterative generate–rank–select–train paradigms that echo MCTS principles to balance task performance with efficiency.

LLM-based Methods Complementing search-based methods, several recent works leverage the generative capacity of LLMs to construct and adapt dynamic topologies. Dylan [725] introduces a temporal feed-forward network (T-FFN) model that treats each communication step as a network layer, using forward-backward propagation to compute Agent Importance Scores for dynamic team selection. In related work, DAMCS [1029], AutoAgents [1030], and TDAG [1032] dynamically generate specialized sub-agents or update hierarchical knowledge graphs, enabling cooperative planning and task decomposition. Further, frameworks such as AutoFlow [773] and Flow [1033] represent task workflows in natural language programs or activity vertex graphs (AOV), allowing continuous refinement through reinforcement learning signals. ScoreFlow [788] complements these approaches by applying gradient-based (loss-gradient) optimization to continuously reconfigure agent workflows.

External Parameters Given that fine-tuning LLM-based agents is often resource-intensive, a considerable number of researchers advocate configuring inter-agent topologies by training parameters independent of the LLM-agent. This approach is initiated by GPTSwarm [651], in which the inter-agent topologies are represented as a directed acyclic graph (DAG), with edge weights serving as the sole trainable component of the system. Further advancing this paradigm, AgentPrune provides a unified modeling framework from the spatial-temporal graph perspective for mainstream MAS, where communication redundancy, i.e., unnecessary edges, is identified and pruned through magnitude-based pruning. Follow-up works in this line of research include G-Safeguard [1034], which similarly trains GNN outside of the MAS to detect and eliminate malicious communication paths. Although these methods are parameter-efficient, their relatively small parameter space and low coupling with LLM-agents often result in performance limitations to some extent.

Discussion Dynamic topologies extend beyond task-solving and play a crucial role in simulating complex social interactions. As detailed in a recent survey [975], LLM-based agent models can evolve inter-agent links to capture real-time changes in autonomy, social behaviors, and environmental feedback across various domains, including cyber, physical, and mixed environments. Systems such as [50], OASIS [936] and ProjectSid [989] simulate dynamic social networks. [50] employs generative natural language memory retrieval to adjust social ties based on agents' experiences, while OASIS constructs a real-time social media environment with continuously updated user relationships and information flows. Project Sid [989] introduces the PIANO (Parallel Information Aggregation via Neural Orchestration) architecture, enabling over 1,000 autonomous AI agents to interact in real-time within a Minecraft environment, leading to the emergence of complex societal structures such as specialized roles, collective rule adherence, and cultural and religious transmission. Additionally, architectures like AgentScope-scability [1035] and Social Survey [975] support large-scale multi-agent simulations, enabling studies of cultural dissemination, collective decision-making, and emergent group dynamics in environments with hundreds or thousands of interacting agents. Additionally, dynamic topologies are also tailored to specific application domains such as medical and open-domain embodied AI. In the medical field, AI hospital [1036] and agent hospital [921] simulate real medical workflows, where iterative cycles of diagnosis, treatment, and feedback continuously reshape communication patterns among various roles, such as intern doctors,

patients, examiners, and supervising physicians. These frameworks dynamically adjust inter-agent communication to optimize collaboration and decision-making. Similarly, in open-domain and embodied AI applications, frameworks like IOA [933] support heterogeneous, cross-device agent interactions, facilitating dynamic team formation and task allocation in real-world scenarios.

Although the aforementioned dynamic multi-agent topologies have made substantial progress in performance metrics, they still face the following three limitations, which we believe should be the focal points for future research on dynamic topologies:

(1) Generalizability. Current MAS topologies are typically optimized for a single-task domain. For example, AFlow [773] is dedicated to search and optimization within math or code benchmarks, producing a fixed workflow that is difficult to adapt to new task domains. Other dynamic topologies, such as ADAS [741], GPTSWarm [651], and AgentPrune, face the same challenge. We argue that MAS should be capable of lifelong learning, wherein the system generalizes across different task domains with minimal resources (e.g., API calls, FLOPs, GPU hours).

(2) Resource Efficiency. Present dynamic topologies often tend to optimize for complex, resource-intensive structures. Their training processes are typically exorbitantly costly, as exemplified by ADAS [741], where training with GPT-3.5 incurs a cost of approximately $300 per session. Such expenses severely constrain their large-scale applicability in real-world scenarios. Future developments should focus on achieving better test-time topology optimization with significantly reduced costs.

(3) Inference Efficiency. As MaAS [787] has incisively observed, multi-agent topologies of excessive complexity, while capable of consistently delivering satisfactory performance, are lamentably deficient in *task adaptability*. That is to say, they are unable to dynamically allocate reasoning resources (i.e., tools, the number of agents, and reasoning steps) in response to the difficulty of a given task. Consequently, this may lead to a certain lack of efficiency in the inference process. Although MaAS has, to a certain extent, achieved task dynamism through the designed agentic supernet, their applicability and scalability in large-scale deployment still remain to be tested.

14.2 Scalability Considerations

Scalability is a critical challenge in LLM-based multi-agent systems (MAS), especially as the number of agents grows. In fully connected networks, the number of communication paths grows quadratically, leading to a communication explosion that increases token usage and computational costs [1037, 626]. Centralized and layered topologies can experience synchronization bottlenecks if supervisory nodes are inundated by messages, whereas decentralized networks—while more fault tolerant—necessitate complex consensus algorithms to achieve a coherent global state.

Recent work such as [1028] demonstrates that when multi-agent collaboration is structured as a directed acyclic graph (DAG), the system can scale efficiently to handle large graphs—up to 1,000 nodes or more—without significant performance degradation. Similarly [1022] shows that distributing graph reasoning tasks among many agents circumvents the limitations imposed by long textual inputs and context-length constraints. Moreover, studies on self-organized agents[1038] reveal that dynamic multiplication and task distribution allow the system to maintain a constant workload per agent while increasing overall processing capacity. Finally, the multi-dimensional taxonomy proposed by [1039] provides a valuable framework for analyzing trade-offs between agent autonomy and alignment, offering insights into how to balance centralized control with decentralized flexibility to optimize scalability.

In addition to these foundational studies, recent advances in practical multi-agent platform design further enrich the scalability discussion. For example, AgentScope [1035] offers a developer-centric platform that leverages an actor-based distributed framework to enable seamless migration between local and distributed deployments. Its unified workflow and automatic parallel optimization significantly reduce the communication overhead and synchronization challenges that typically emerge as agent numbers increase. By incorporating fault-tolerance mechanisms and intelligent message filtering, AgentScope illustrates how system-level supports can be designed to maintain performance even in dynamic and heterogeneous deployment environments.

Another complementary approach is presented in Project Sid [989], which explores scalability within the realm of simulating agent civilizations. Here, the focus shifts from isolated task solving to the simulation of complex societal dynamics. The proposed PIANO (Parallel Information Aggregation via Neural Orchestration) architecture allows agents to operate concurrently by decoupling slower cognitive processes from rapid reactive modules. A dedicated cognitive controller is introduced to ensure coherence among multiple parallel outputs. This design not only enables scalability from small groups to simulations involving over a thousand agents but also effectively addresses the inherent coordination challenges arising from high-frequency interactions.

Taking scalability to an even larger scale, AgentSociety [1040] demonstrates a comprehensive framework for simulating realistic social environments with up to 10,000 agents. By integrating LLM-driven social generative agents within a realistic urban, social, and economic setting, AgentSociety employs distributed computing and a high-performance messaging system (e.g., MQTT) to support millions of daily interactions. This platform exemplifies how emerging hybrid architectures can support macro-level phenomena—such as economic market dynamics, opinion diffusion, and urban planning simulations—by effectively managing the trade-offs between communication cost, coordination overhead, and emergent behavior fidelity.

Despite the theoretical advantages of scaling up agent populations, it is imperative to question whether pursuit of large-scale agent deployments is inherently valuable for all task-solving scenarios. Although the total computational capacity scales with the number of agents, when memory overhead and inter-agent communication costs are factored in, the marginal utility of adding additional agents may demonstrate diminishing returns. This phenomenon arises from the fundamental constraint that, while the overall workload is the product of individual task complexity and the degree of labor division, coordination costs tend to increase super-linearly with agent count. Therefore, for many bounded problem domains, there is likely an optimal agent population size beyond which performance plateaus—or even deteriorates—due to excessive coordination overhead.

Conversely, in simulation scenarios where the objective is to model complex social dynamics, emergent behaviors, or large-scale collective intelligence, scaling to numerous agents becomes not merely beneficial but essential. In these contexts, the research focus shifts from optimizing computational efficiency for task solving to accurately reproducing or predicting macro-level patterns emerging from micro-level agent interactions. Such simulations—covering domains like economic market behavior, social network evolution, and urban infrastructure planning—often require the computational overhead of managing vast agent populations in order to capture realistic population-level phenomena.

Hybrid architectures that combine centralized oversight with decentralized sub-teams offer a promising solution to these scalability challenges [921, 918]. In these designs, supervisory agents handle global objectives and coordination, while worker agents focus on executing specific subtasks. This hierarchical organization helps to mitigate information overload at any single node and allows for dynamic adjustment of agent team sizes based on task demands, thereby optimizing resource utilization. Furthermore, advanced techniques such as graph search algorithms, reinforcement learning-based updates, and evolutionary methods are critical for iteratively refining the network structure as the system scales. Intelligent message filtering, prioritization, and aggregation mechanisms can significantly reduce communication overhead without sacrificing the quality of inter-agent collaboration. In addition, asynchronous communication protocols and partial knowledge sharing strategies show promise in minimizing coordination bottlenecks while maintaining sufficient global awareness among agents.

Concluding Remarks on Scalability Overall, the study of system topology and scalability in LLM-based MAS reveals a spectrum of design choices—from static configurations that offer simplicity and predictability to dynamic architectures that provide flexibility and adaptability. While foundational works (e.g., [1028], [1038]) emphasize scalable graph structures and self-organizing principles, the practical advances demonstrated by AgentScope, Project Sid, and AgentSociety illustrate how integrated distributed frameworks, concurrent processing, and realistic environment simulations can collectively address the challenges of scaling multi-agent systems. The context-dependent nature of scalability requirements—contrasting between task-solving and simulation scenarios—highlights the importance of purpose-specific design in multi-agent architectures. As research continues to evolve, the development of more sophisticated adaptive algorithms, distributed architectures, and multi-dimensional evaluation frameworks will be essential for advancing the scalability and practical viability of LLM-based multi-agent systems.

Chapter 15

Collaboration Paradigms and Collaborative Mechanisms

In this chapter, we offer a detailed exploration of these purposeful interactions, examining how one agent influences collaboration within MAS. We reference the diverse interaction behaviors that emerge from human social structures, further explaining multi-agent collaboration through interaction purposes, interaction forms, and the relationships that form.

Multi-Agent Systems (MAS) comprise multiple agents that interact in a shared environment, autonomously making decisions to accomplish tasks collaboratively or compete with each other [1041]. In our context, we focus on collaborative phenomenons because they widely appeared in most practical applications. Basically, each agent in MAS is equipped with different roles and initial knowledge and its own set of goals.

When engaged in problem solving or communication, agents interact with other agents or the environment to collect and process information, independently making decisions based on their objectives, existing knowledge, and observations, and subsequently executing actions [975, 1041, 1042, 1043]. Knowledge, memory, and environmental observations form the agents' beliefs, while varying motivations influence their approach to tasks and decision making [1041]. Consequently, effective problem solving requires diverse purposeful interactions, including agent-agent and agent-environment. These interactions may involve multiple rounds and occur in various directions, depending on the system design.

15.1 Agent-Agent collaboration

Considering the categorizations of MAS collaborations, we focus on more details on the granularity needed to capture the nuanced dynamics in complex multi-agent interactions. Sepecifically, we categorize inter-agent interactions into four types, inspired by sociological insights from human-to-human interaction patterns and applying them to agent-agent interactions in MAS. Sociological theories on human interaction, which include **consensus building**, **skill learning**, **teaching**, and **task division collaboration**, provide a more refined way of classifying agents. interactions. These interactions form collaborative paradigms, which enable diverse intelligent agents to work together effectively in solving complex problems, and they are shaped by various forms of goals, contexts and outcomes. Each paradigm addresses unique challenges related to cooperation,competition, coordination, and decision-making. Additionally, MAS implementations involve agents with different types of interactions, rather than a single type or unidirectional process, forming complex interaction networks that evolve over time. In collaborative software development [626, 627], a senior developer agent may interact task-wise with an architect agent, guide junior agents through multi-round dialogues. They work together on code reviews for decision-making and learn with a testing expert agent to improve test coverage. Examining the objectives and results of these interactions reveals the crucial techniques and technologies shaping agent behavior and decision-making, thereby enhancing our comprehension of multi-agent dynamics.

Consensus-oriented Interaction Consensus-oriented interactions concentrate on harmonizing the MAS's final target via negotiation, voting, and social choice frameworks [1044]. This interaction is significant for incorporating diverse knowledge and ensuring agents shift their views towards a unified understanding to achieve consensus [1045]. In this interaction, agents integrate knowledge to establish a unified understanding, which largely helps joint decision-

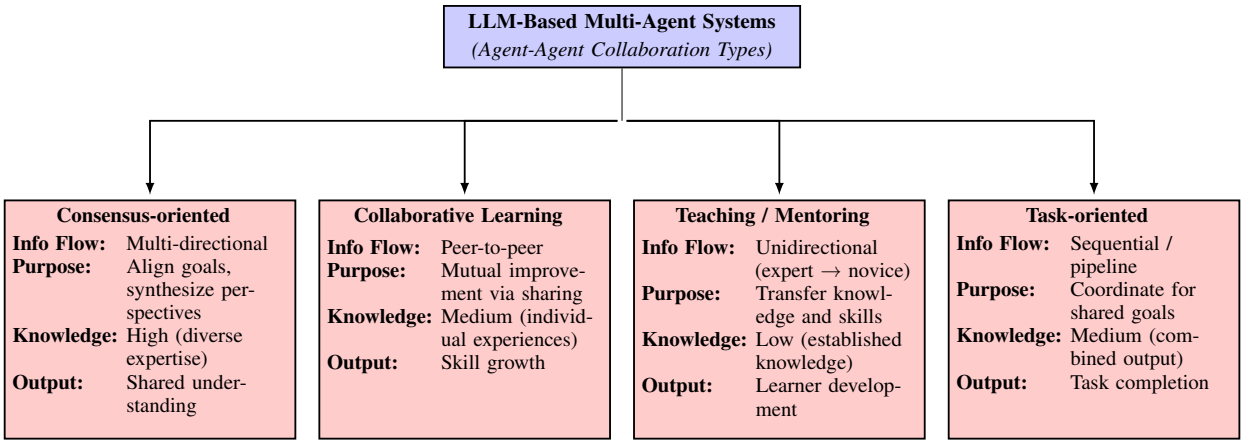


Figure 15.1: An overview of four agent-agent collaboration types in LLM-based MAS: *Consensus-oriented*, *Collaborative Learning*, *Teaching/Mentoring*, and *Task-oriented*. Each type is described along four key dimensions: information flow, collaboration purpose, knowledge integration, and output focus.

making in complex problem-solving situations that demand different viewpoints. For instance, MedAgents [922], MDAgents [1046], and AI Hospital [1036] demonstrate how collaborative dialogue among multidisciplinary agents improves problem solving by sharpening reasoning skills and accessing inherent knowledge.

These dialogues allow agents to ensemble expertise into coherent outcomes, frequently outperforming conventional methods like zero-shot or few-shot reasoning. The importance of consensus-driven teamwork is particularly evident in scientific environments, where addressing complex challenges requires diverse perspectives and meticulous validation. Agent Laboratory [746], serves as an example where PhD and postdoctoral agents collaborate to agree on research objectives, interpret experiments, and consolidate research findings. Similarly, Virutal Lab [752] organize a series of team to conducts scientific research, where all agents discuss a scientific agenda, and individual meetings, where an agent accomplishes a specific task.

Methods for multi-agent consensus typically include several approaches, including **Discussing**, **debating**, **negotiating**, **reflecting**, and **voting**. Common methods for reaching consensus encompass an array of structured techniques. The primary mechanisms involved are **discussing**, **debating**, **negotiating**, **reflecting**, and **voting**. Debates allow agents to obtain competing hypotheses, while negotiation helps resolve conflicting priorities and resource limitations. Specific frameworks have been created to support these consensus-building activities. During these processes, agents gather outputs from peers tackling the same issue, and include environmental feedback as numerical data and contextual details. These interactions enable agents to share viewpoints, assumptions, and progressively achieve a common understanding.

For example, GPTSwarm [651] formulates the collaboration between agents with graph design, that the information flow and edge connections build the basic group discussion. In GPTSwarm, if an agent consistently provides incorrect opinions, it will be excluded. RECONCILE [918] uses a round-table discussion format with several discussion cycles and voting systems based on confidence levels. It integrates reflection by learning from past discussions, using confidence metrics and human insights to improve their responses. Furthermore, debates are quite important for achieving agreement, reducing hallucinations and also addressing complex issues [985, 1047, 1031, 1003]. In GOVSIM [1048], agents collaborate to achieve a balance, and it suggests using a shared resource and conserving it for future needs. The negotiations went beyond simple information exchange and relationship-focused interactions. The Multi-Agent Debate (MAD) framework [1031] promotes creative thinking by having agents deliver arguments in a "tit-for-tat" pattern, with a judge overseeing the process to finalize a solution. The Formal Debate framework (FORD) [1004] enhances consistency among language models through organized debates, enabling stronger models to steer consensus, while weaker ones adjust their perspectives. Similarly, AutoAgents [1030] define a collaborative refinement action in which each agent updates its chat record. In the process, it also appends the previous statements of the other agent and refines its action to achieve consensus.

Collaborative Learning Interaction In collaborative learning, interaction usually happens among similar agents. Although architecturally alike, accumulate distinct memories and experiences due to their unique behaviors and varied environmental interactions. By solving problems together, these agents share experiences to boost their strategy learning, task-solving, and skill acquisition capabilities. Over time, each agent enhances its skills through ongoing interaction, leading to the evolution of individuals. The key difference between collaborative learning and consensus-

oriented interactions lies in their fundamental goals and processes. While consensus-oriented interaction focuses on knowledge integration and belief alignment through synthesizing diverse viewpoints to reach agreement, collaborative learning interaction emphasizes peer knowledge construction and experience sharing, prioritizing mutual improvement and individual growth. When engaged in collaborative learning interaction, agents update their context or memory from observing others' behavior. For example, agents can learn optimal strategies by observing the deliveration from peers, adapting their own approach based on these observations without necessarily agreeing on a single "best" strategy [961, 962, 963, 971, 965, 967, 972, 968, 969]. As highlighted in [966], the effective discussion tactics significantly impact learning outcomes among agents. In these interactions, agents collaborate to learn and address problems, focusing on mutual understanding and enhancement rather than reaching unanimous decisions. This method refines personal responses and knowledge via ongoing feedback.

The methods commonly employed in collaborative learning interaction include: **1). Experience sharing.**, Agents exchange personal insights and best practices. As described in [303], iterative experience refinement enables LLM agents to achieve adaptive improvement in software development via continual acquisition and utilization of team experience in successive pattern and the cumulative pattern. Furthermore, MAS-CTC [301] is a scalable multi-team framework that enables orchestrated teams to jointly propose various decisions and communicate with their insights in a cross-team collaboration environment. It enables different teams to concurrently propose various task-oriented decisions as insights, and then communicate for insights interchange in important phases (multi-team aggregation). Different agent teams utilize a greedy pruning mechanism and aggregation mechanisms to eliminate low-quality content, thus improve the performance in software development. Differently, in MOBA [1049], a novel MLLM-based mobile multi-agent system, global agent reflects on local agent execution results to support adaptive planning to align with the environment. AutoAgents [1030] employs a knowledge sharing mechanism where agents exchange execution results to enhance communication and feedback, where agents can obtain long-term, short-term and dynamic memory from others. **2). Peer discussions.** Peer discussions allow agents to articulate their reasoning processes and learn from others' approaches. MEDCO [923] create a dynamic environment where clinical reasoning and decision-making skills are strengthened through collaborative problem-solving among student agents. Moreover, In [1050], agents engage in structured peer discussions after initializing their output, reviewing each other's reasoning step by step. Through feedback exchange and confidence scoring, agents refine their decision-making, learn from diverse approaches, and iteratively enhance their reasoning accuracy, fostering collaborative knowledge acquisition. **3). Observational learning.** Observational learning occurs when agents monitor others' behaviors and outcomes to inform their own strategies. AgentCourt [1051] develops lawyer agents that participate in court debates and improve through accumulated experiences, demonstrating improved reasoning and consistency through experiential learning. In iAgents [1046], the human social network is mirrored in the agent network, where agents proactively exchange human information necessary for task resolution, thereby overcoming information asymmetry. iAgents employs a novel agent reasoning mechanism, InfoNav, to navigate agents' communication towards effective information exchange. Together with InfoNav, iAgents organizes human information in a mixed memory to provide agents with accurate and comprehensive information for exchange. Additional experimental phenomenon indicates difficulty of certain tasks making agents continuously refine their strategies in pursuit of the required information. MARBLE [948] designs a cognitive evolve planning combining the 'expectation' of the agent and its actual action results to update the overall planning experience for better planning in the next round.

Despite its benefits, collaborative learning interaction faces several challenges. These include ensuring equitable knowledge exchange among agents with varying capabilities, preventing the propagation of errors or biases across the system, maintaining agent diversity while facilitating learning, and developing effective mechanisms for agents to selectively incorporate others' knowledge based on relevance and reliability. Overcoming these challenges requires the meticulous creation of interaction frameworks and learning strategies. And it should balance individual advancement with the broader development of the system. Although issues such as knowledge fairness, bias propagation, and scalability present difficulties, there is great potential to improve MAS, particularly in dynamic and complex environments. By using iterative learning processes and providing opportunities, collaborative learning enables agents to develop richer knowledge bases and more refined problem-solving abilities.

Teaching/Mentoring Interaction To tackle these challenges, it is important to carefully develop interaction protocols and learning frameworks that harmonize individual development with overall system progress. In the context of MAS, teaching and mentoring interactions are fundamental mechanisms in collaborative environments, especially in scenarios where knowledge transfer is essential for growth and collective intelligence. Unlike collaborative learning, where knowledge is exchanged reciprocally among agents, teaching and mentoring interactions focus on the unidirectional flow of knowledge from an experienced agent to a less experienced one. The mechanisms and methods used in teaching/mentoring interactions include several key strategies:

- **Criticism and Feedback.** The mentor agent evaluates the learner's performance and provides corrective or constructive feedback. This helps the learner refine their knowledge and skills through a feedback loop where they update their internal knowledge based on the feedback received.
- **Evaluation.** Mentors assess the learner's capabilities or progress through performance reviews and clear assessment criteria, providing valuable insights for development.
- **Instruction and Teaching.** Mentors convey targeted knowledge, guidelines, or techniques using direct instruction which allow learners to pose questions and receive clarifications.

Iterative Teaching and Reinforcement Teaching is typically progressive, where each phase provides opportunities for the learner to complete tasks and get feedback. For example, in the MEDCO system [923], student agents improve their professional skills through a cyclic practice-oriented learning approach directed by expert mentors, in addition to engaging in peer discussions. These expert agents conduct ongoing assessments and provide real-time guidance on clinical competencies, focusing on patient interaction skills and diagnostic reasoning. [921] shows that an agentic doctor can continually improve their diagnosis by merely interacting with agentic patients in a simulated hospital and can transfer its learned knowledge of real-world cases.

This interaction type can be categorized based on the direction of knowledge transfer into two primary types: unidirectional and interactive. Unidirectional is rooted in traditional teaching models where knowledge flows from the teacher to the student. This approach emphasizes the transmission of facts and concepts, often involving lectures and direct instructions [923].

Task-oriented Interaction. Task-oriented collaborations involve agents working together to achieve common objectives through effective coordination and task decomposition strategies, as well as a high degree of cooperation and coordination. Agents interact primarily by processing upstream output and generating results for downstream agents following established task dependencies rather than engaging in complex discussions or debates.

Recent frameworks demonstrate diverse implementations of this interaction pattern: **(1) software development frameworks** such as MetaGPT [626] and ChatDev [627], agents operate in a structured pipeline that mirrors the software development lifecycle. For example, architect agents process requirements to generate technical specifications, which development agents then use to produce code, followed by testing agents who validate the implementations; **(2) Collaborative reasoning** frameworks like Exchange-of-Thought (EoT) [1052], GPTSwarm [651], MACNET [1028] involve structuring agents in a specific format (e.g., ring, tree, directed acrylic graphs, optimizable graphs) , which mitigates context expansion risks by ensuring only optimized solutions progress through the sequence, and enforcing multiple agents to collaborate together towards solving complex mathematical or knowledge reasoning tasks; In **(3) ML applications** [1053, 1019], agents adhere to stringent workflow structures, each fulfilling specific tasks in processes. For more complex tasks such as VideoQA, the TraveLER framework [1054] showcases modular task breakdown across structured phases (Traverse, Locate, Evaluate, and Replan), with a Planner agent managing interactions and improving strategies based on iterative agent inputs.

These handoffs rely on explicit deliverables instead of direct agent negotiations. Inspired by GPTSwarm [651]-alike graph agentic systems, MACNET [1028] structures agents into directed acyclic graphs (DAG). Here, supervisory figures issue directives while executors implement solutions. By ensuring only optimized solutions progress through the sequence, this setup mitigates context expansion risks. In ML applications [1053, 1019], agents adhere to stringent workflow structures, each fulfilling specific tasks in processes. For more complex tasks such as VideoQA, the TraveLER framework [1054] showcases modular task breakdown across structured phases (Traverse, Locate, Evaluate, and Replan), with a Planner agent managing interactions and improving strategies based on iterative agent inputs.

Beyond organized development, task-driven interactions have been shown in open-ended contexts such as Minecraft game, in where agents adjust to ever-changing environments. In [927], leader agents manage workflows by breaking down complex objectives into specific tasks, while executor agents perform actions like gathering resources. Coordination mechanisms are important for ensuring agents collaborate effectively towards final goal, including communication protocols, synchronization strategies, and resource-sharing techniques. The interaction of agents in MAS for task execution has garnered significant interest, notably through utilizing LLMs for handling intricate tasks and workflows. The collaboration of agents are vital for task completion, particularly in ever-changing settings like software development and project management [626, 630].

15.2 Human-AI Collaboration

To unlock the potential of MAS in meeting human objectives, people often work alongside them using three primary methods: **one-off task delegation**, **multi-turn interactive instruction**, and **immersive human-agent collaboration**.

In **one-off task delegation**, humans delegate single-instance tasks to MAS, such as posing a question to a Q&A platform or assigning a coding task [1055, 626]. Without additional input, the agent handles the task autonomously, delivering a complete response or solution in a single reply. This is presently the prevalent way humans collaborate with LLM-based agents [922, 627, 31].

For **multi-turn interactive instruction**, humans engage in iterative interactions with LLM-based agent systems to refine and explore solutions until a satisfactory result is achieved. This type of interaction is widely seen in creative applications, such as image editing or writing edit [938]. For instance, a user might ask the system to add an object to a specific location in an image, replace an element, change the background, or revise a part in a sentence. These interactions often span multiple rounds, with users continuously refining their requests until the desired outcome is reached. Moreover, certain other LLM-based agent systems may require human approval or clarification during multi-turn interactions before proceeding to the next step [1056, 930]. Under human guidance, these LLM-based agent systems can complete household tasks as well as software development tasks.

Immersive human-agent collaboration features LLM-based agents simulating human behaviors to serve as partners. For instance, in an immersive setting, humans treat these agents as teammates, achieving common objectives. Instances include agents representing humans in meetings or help solve tasks like chores or projects. This strategy highlights effective integration and teamwork in dynamic contexts [937, 924].

To assess Human-AI collaboration quantitatively, several frameworks have been suggested. Co-Gym [1057], for instance, measures the communication, situational awareness, and personalization of LLM-based agents in tasks such as travel planning, writing related work, and tabular analysis.

In summary, as LLM-based agent systems have advanced, Human-AI collaboration has diversified to address challenges across domains. This ranges from simple command-based AI interactions for questions, to multi-turn dialogues for design and development, and partnering with human daily tasks.

With advancements in LLM-based agent systems, they are expected to integrate more into daily life, streamlining tasks and boosting efficiency. At the same time, humans will refine and adapt their ways of interacting with AI, leading to more effective collaboration. We believe this shift will drive fundamental changes in both social productivity and the social relations of production, reshaping how work is organized and how humans and AI cooperate in the large language models era.

15.3 Collaborative Decision-Making

Collaborative decision-making processes are crucial for ensuring the efficient operation of MAS and the successful completion of tasks. Although collaboration itself is a core feature, the approaches of decision-making directly determines the effectiveness of collaboration and the overall performance of the system. Recent research has highlighted the critical role of collaborative decision-making. [1037] showed that diverse decision-making methods can significantly enhance the collaborative efficiency of the system. [649] emphasized that a rational decision-making mechanism can stimulate the emergence of intelligence within a system.

From a broader perspective, the collaborative decision-making process can be divided into two major categories based on their architectural characteristics: Dictatorial Decision-Making and Collective Decision-Making [1037].

Dictatorial Decision-Making. Dictatorial Decision-Making is a process where decision-making relies on a single agent in a MAS. In this paradigm, all agents send their state information or local observations to this dictatorial agent. The dictatorial agent is responsible for assembling this data, studying the core problems, and establishing definitive decision guidelines. The key principle for such an approach is to leverage a global mindset in moving towards improved decision-making, hence paving the reliability of the system performance along with the successful achievement of task goals. [1031, 1058, 1046] demonstrated the single-agent decision-making process with a single LLM, who synthesized various views on the same problem to make decision-making even more objective and comprehensive. Furthermore, [134, 1059] suggested the weighted integration method through ranking, scoring or checklist, enhancing the robustness of decision-making procedures. In addition, beyond the explicit inclusion of perspectives, [1030, 1060] proposed architectures where a central agent breaks down complex tasks into simpler sub-tasks and assigns them to specialized agents grouped by their functionalities. Moreover, in [651, 1028],it is common that the last node's agent works in an environment to assemble the past information and deduce a conclusion according to the topological structure, rather than by a central agent.

Collective Decision-Making. Collective Decision-Making involves agents collaborating to reach decisions without a central authority, relying on local data and interactions like voting or negotiation. This method shares decision-making power among agents, allowing the system to adapt according to changes while maintaining robustness and scalability.

- ***Voting-based Decision Making***Voting systems are important for collective decision-making, providing a framework for reaching consensus. A conclusive majority is achieved through voting as described by [1045, 968]. Moreover, the GEDI electoral module [1037] enables multiple voting methods. This method largely improve reasoning and fault-tolerance while avoiding complex system designs.
- ***Debate-based Decision Making***In comparison with voting-based methods, debate-based decision-making focuses on organized interactions between agents, in order to obtain the best result. In [1031, 1061], agents participate in guided discussion, where they articulate and proposals in an attempt to resolve disagreements and reconcile points of view. Simultaneously, [1050, 1062] practice restraint stance, using communication channels among agents for consensus-building through repeated discussions. To tackle the issue of "cognitive islands," certain systems would employ a common retrieval knowledge base to enable agents to be aware of the same knowledge throughout debates [1005]. By mimicking human dialogue, these systems allowed agents to exchange perspectives and make more informed decisions.

Discussion and Future Work Collaboration in multi-agent systems (MAS) still faces numerous challenges that require further research. Current methods are largely based on contextually dependent interactions; however, they do not include a specific framework for training and optimizing cooperative actions. This heavy dependence on large language models (LLMs) has some limitations, as their effectiveness is inherently tied to the size of the LLM's contextual window and its native reasoning capabilities. While LLMs provide a solid foundation for enabling interactions, these systems are still limited by the inherent limitations of context-dependent communication.

Future studies should focus on finding frameworks that inspire agents for active learning with regard to optimal timing and information dissemination methodologies. Using methodologies from multi-agent reinforcement learning (MARL), there is a growing requirement for strategies that will help agents determine appropriate moments for information sharing, as well as what information should be shared through what channels. This calls for not just devising novel interaction protocols but also incorporating training methodologies that will constantly optimize these protocols with each improvement.

Chapter 16

Collective Intelligence and Adaptation

The concept of collective intelligence is central to the development of multi-agent systems(MAS), drawing inspiration from biological and societal cooperation. An inherent concept within collective intelligence is the "Wisdom of Crowds" by [915], which asserts that independent communities often make better decisions as a whole than any one person. Cognitive theoretical models like the Society of Mind [17] and its related theory mind [916, 917] further support the paradigm, suggesting that intelligence springs from a synergy among primary, specialist components. Moreover, In human societies, individuals collaborate, divide labor, and engage in collective problem-solving to address complex challenges. MAS adopt similar strategies where specialized agents to participate in solving complex problems and collective decision-making [914].

The emergence of collective intelligence within MAS is a dynamic and iterative process. Through continuous interaction, agents develop a shared understanding and collective memory progressively. The interaction dynamics are strengthened by heterogeneity among individual agents, environmental feedback, and agent-agent interactions [914], which are all important for the emergence of complex social networks and improving decision-making strategies. It is worth highlighting that collective intelligence is not merely the summation of individual capability, but refers to emergent behavior beyond individual agent capacity. beyond individual agent capacity. Individual agent development is deeply linked with collective intelligence growth. With ongoing involvement with collective tasks, and self-reflection on shared contexts, agents increasingly develop reasoning and decision-making capabilities. The evolution of individual agents is closely related to collective intelligence evolution. Through continuous interaction in joint activities and critical examination of shared contexts, agents continuously refine their reasoning and decision-making abilities.

In parallel, complex and diverse behavior among agents emerges. These include beyond-restricted-protocol behaviors, such as advanced social interactions, including trust, strategic deception, adaptive camouflage, and emergent cooperation, evoking a shift from reactive into cooperative strategies, as well as deeper social dynamics. With a chain of recursive interactions, agents necessarily form cooperative strategies, which eventually turn into social contracts, organizational hierarchies, and divisions of labor. Social phenomena necessarily emerge through recursive interactions among agents, coupled with their adjustment with the changing environment. It marks a transition from fundamental cooperative behavior into complex social constructs, leading to cultural norms and conventions.

16.1 Collective Intelligence

The concept of collective intelligence, which refers to the ability of a group of agents to exhibit problem-solving capabilities that surpass those of individual agents. This phenomenon is often characterized by emergent behaviors, sophisticated decision-making, and higher-order reasoning abilities that arise from interactions among agents, leading to enhanced performance in collaborative decision-making scenarios and social simulations [975]. [917] demonstrate that LLM-based agents can exhibit collaborative behaviors and high-order Theory of Mind capabilities, which are crucial for understanding the perspectives of other agents in a shared environment. Their findings suggest that the integration of LLMs into MAS can facilitate more sophisticated forms of collective intelligence, thereby improving the overall efficacy of collaborative decision-making.

Improved System Performance A primary advantage of collective intelligence in MAS is that collaboration leads to superior problem-solving capabilities. Collective intelligence can be encouraged to overcome "groupthink" and individual cognitive bias in order to allow a collective to cooperate on one process – while achieving enhanced

intellectual performance. When individual agents share information and coordinate actions, the system can achieve better results than any single agent operating independently [626, 922, 1046, 1031, 1063]. Collective intelligence is therefore shared or group intelligence that emerges from the collaboration, collective efforts, and competition of many individuals and appears in consensus decision making. Collective intelligence strongly contributes to the shift of knowledge and power from the individual to the collective. [924] demonstrated this through their Cooperative Embodied Language Agent (CoELA), which achieved a 40% improvement in efficiency over traditional planning methods in ThreeDWorld multi-agent transport tasks. This substantial improvement stems from the system's ability to effectively utilize LLMs for planning and communication in multi-agent settings, providing compelling evidence for enhanced collaborative decision-making capabilities. As previously discussed, the inherent diversity and interdisciplinary nature of LLM-based multi-agent systems, along with various inter-agent interaction, which provide internal feedback and enriched context for individual decision-making, hence reduce bias and improve the consistency of solution [918].

Emergent Behaviors One of the most intriguing aspects of collective intelligence is the emergence of new, complex behaviors that arise spontaneously from agent interactions. These behaviors are not explicitly programmed but emerge from learning and adaptation. As discussed in various studies [971, 965, 966], agents developed strategic behaviors, including trust-building, adversarial tactics, deception, and leadership during the game. The collective behavior evolved through experience sharing, where village-aligned agents learned cooperation and strategic alliance formation, and wolf-aligned agents improved deception through "information confusion" tactics. Moreover, agents optimized voting patterns and deception strategies without explicit training, which indicates the group intelligence emerged over multiple rounds of interactions. Similarly, in the Avalon game [968], researchers observed that agents became better at identifying and countering deceptive information. Individuals adapted to deceptive environments and refined their decision-making using first- and second-order perspective shifts. Furthermore, agents demonstrated adaptive cooperation and ad hoc teamwork, despite no predefined collaboration protocols [969]. These findings highlight the ability of LLM-based agents to develop sophisticated behaviors through interaction and learning, showcasing the potential for emergent behaviors in collective intelligence scenarios. Notably, these emergent behaviors rely on memory and reflective mechanisms. Agents retrieve and reflect on historical information to generate a compact context, enhancing their reasoning capabilities [239]. In MAS, shared context and environmental information significantly boost agents' usable memory. This enables agents to build on past interactions, refine strategies, and adapt more effectively to dynamic environments [1064].

Social Evolution One of the most significant findings in the field of generative agent societies is the spontaneous emergence of social norms. [1065] demonstrated that agents, through continuous interaction, are capable of creating, representing, spreading, evaluating, and complying with social norms. These norms serve as the foundation for social order, reducing conflicts and improving coordination among agents, thereby leading to more stable and organized societies. Interestingly, the study found that agents develop norms more rapidly in their beliefs than they do in their behaviors. This suggests that while agents may quickly internalize certain norms, the translation of these norms into consistent actions takes longer. Over time, these norms tend to synthesize into more general principles, resulting in more concise and effective personal norm sets. Furthermore, the Project Sid simulation[989] models large-scale agent societies and provides further evidence of the emergence of social norms and role specialization. In this study, agents were observed to autonomously form specialized social roles. These roles were not predefined but emerged naturally as agents interacted within their environment and developed collective rules.The simulation also highlighted the importance of democratic processes in the adherence and modification of these collective rules. Agents were found to engage in cultural and religious transmission, spreading ideas and doctrines across communities. This process of norm creation and role specialization leads to better organization, reduced conflict, and adaptive governance structures within the society. The evolution of cultural and religious beliefs in multi-agent societies is also observed in [1066], which occurs through agent-driven selection of ideas, mirroring real-world societal changes. Additionally, the [936], which simulates social interactions among one million agents, provides valuable insights into cultural transmission and group polarization. Cultural memes and belief systems propagate naturally among agent societies. Agents exhibit herd behavior, conforming to prevailing opinions even when these opinions are irrational. This leads to the emergence of group polarization, where agents reinforce extreme views through repeated interactions. This finding highlights the significant impact of group size on the dynamics of cultural evolution and social behavior.

16.2 Individual Adaptability

In multi-agent systems (MAS), individual adaptability refers to an agent's ability to adjust its behavior and decision-making strategies based on previous interactions and experiences. This is also defined as self-evolving, where agents can dynamically self-evolve by modifying themselves, such as altering their initial goals and planning strategies, and training themselves based on feedback or communication logs [38]. This adaptability is facilitated by the integration of large language models (LLMs), which support dynamic monitoring and adaptation processes [1067], as well as the

agents' memory capabilities and information exchange. These modules are crucial to ensure that agents can continuously improve their performance, respond effectively to dynamic environments, and optimize their decision-making processes. We categorize the mechanisms contributing to individual adaptability into memory-based learning and parameter-based learning, where there are training-free and training-based approaches.

Memory-based learning Memory and reflective mechanisms significantly enhance individual adaptability in LLM-based multi-agent systems by leveraging historical records and experiences to inform decision-making [221, 1068, 50]. By maintaining and utilizing individual memory of past interactions, decisions, and outcomes, the agent can refine its decision-making process over time. This memory serves as a repository of experiences that the agent can draw on when making future decisions. Using this stored knowledge, individual agent is able to refine its decision-making process, learning from previous successes and failures [921, 1051]. For example, in clinical simulation, doctor agents can keep improving treatment performance over time by accumulating experience from both successful and unsuccessful cases [921]. In social behavior simulation, agents can improve their adaptability by engaging in more complex scenarios and utilizing scenario memories to enhance performance [50].

Shared memory-based learning In contrast, shared memory-based learning extends this concept by enabling multiple agents to exchange information and insights derived from their respective experiences. Rather than relying solely on individual memory, agents can benefit from the collective knowledge of the group. By sharing data, strategies, and feedback, agents enhance their ability to cooperate and optimize their decisions collaboratively. Shared memory-based learning is particularly valuable in environments where agents need to cooperate, exchange tasks, or work toward common goals [919, 967, 968]. For instance, ProAgent [1069] anticipates teammates' decisions and dynamically adjusts each agent's strategies based on the communication logs between agents, facilitating mutual understanding and improving collaborative planning capability.

Parameter-based learning. Beyond memory-based learning in textual form, many MAS employ parameter-based learning, which evolves agents' individual adaptability through post-training techniques. For instance, [1070] discusses thea Learning through Communication (LTC) paradigm, whereusing communication logs between agents are leveraged to constructto generate datasets forto training or fine-tuninge LLMs. The integration of symbolic and connectionist paradigms within LLM-powered agents enhances botheir reasoning and adaptability. More recently, research has increasingly focused on multi-agent (co-)fine-tuning, which improves collaboration and reasoning capabilities through cooperative trajectories. Examples include multi-agent debate fine-tuning [1071] and SiruiS [1072]. Additionally, Sweet-RL [1073] employs reinforcement learning to enhance the critic model within MAS, fostering better collaborative reasoning. However, despite their promising performance, future parameter-based learning paradigms may need to address the balance between agents' general capabilities and their specialization for specific roles within MAS.This hybrid approach allows agents to handle both structured and unstructured data, improving their ability to make decisions in dynamic environments [1074, 1075].

Chapter 17

Evaluating Multi-Agent Systems

The transition from single-agent to multi-agent systems, and specifically Large Language Model (LLM)-based systems, requires a paradigm change in the evaluation paradigm. In contrast to single-agent evaluation, in which the immediate concern is performance on a particular task, evaluation of LLM-based multi-agent systems must be understood in terms of inter-agent dynamics as a whole, such as collaborative planning and communication effectiveness. Both task-oriented reasoning and holistic capability evaluation are addressed in this chapter, reflecting the nuance of such evaluations. In greater detail, there are two main areas that we examine for evaluation. First, there is task-solving Multi-Agent Systems (MAS), where we examine benchmarks assessing and enhancing LLM reasoning for coding, knowledge, and mathematical problem-solving tasks. These tests also accentuate the utility of distributed problem solving, achieved through organized workflows, specialisation among agents, iterative improvement, and calls for additional tools. Enhanced reasoning, primarily because of agent-agent decision-making cooperation and multi-round communications, is shown for MAS compared with agent-based individual ones. Following that, there is a general evaluation of MAS abilities, extending beyond one-task-oriented achievement, to agent interactions at a highly advanced level. It involves a move away from one-dimensional measurements into multi-dimensional frameworks for documenting achievements at collaborations, reasoning abilities, system efficiency, and flexibility. We categorize such measurements into collaboration-oriented and competition-oriented measurements and have identified efficiency, decision-making quality, quality of collaboration, and flexibility as primary measure domains. These measurements capture various aspects of agent behavior, including communication effectiveness, resource distribution, and response to dynamic situations.

17.1 Benchmarks for Specific Reasoning Tasks

In multi-agent system solving for tasks, much focus has been on leveraging multi-agent coordination for enhancing the reasoning capacity of LLMs. It is most evident in coding, knowledge, and mathematical reasoning benchmarks, where one is interested in examining and building on performance with distributed solving. These benchmarks most typically examine if agents' capability for producing correct code, reasoning on complex knowledge domains, and solving difficult mathematical problems withstanding, with measures such as $pass@k$ [1076] or proof ratios for success being prevalent. Much improvement has been exhibited by MAS through structured workflow, domain-specific agent roles, and iterative improvement on state-of-the-art performance. On the contrary, for model and simulation MAS, the case is one with a comparative lack of standardized benchmarks. Rather, research is primarily experimental setups that simulate a variety of social phenomena, with calls from the community for further formalized evaluation frameworks. These multiple benchmark areas are described below, examining the tasks, measures for evaluation, and the core mechanisms through which MAS result in better performance.

Code Reasoning Benchmark Measuring the capability of LLMs for code synthesis requires bespoke benchmark suites with a focus on functional correctness. Code synthesis, as compared to natural language synthesis, allows for direct verification through running. Several benchmark suites have been built for this purpose, typically consisting of a collection of programming problems, each described with a natural language problem description and a collection of test cases for automatically ascertaining the synthesized code's correctness. HumanEval [1077], APPS [1078], and MBPP [939] are some popular ones. These benchmark suites predominantly utilize the $pass@k$ metric, which computes the percentage at which at least one among the top-k generated solutions passes all test cases for a number of problems. The problems covered through these benchmark suites range across a variety of difficulties and programming

abstractions, requiring not only for LLMs and Agents but also for syntactically correct and logically sound code that satisfies the provided test cases. Recent work has explored leveraging Multi-Agent Systems (MAS) for enhancing LLM capability on code reasoning. For instance, MetaGPT [626] is a meta-programming system which embeds human-like Standard Operating Procedures (SOPs) into multi-agent cooperation based on LLM. With multi-agent role assignment with varying domains and adopting assembly line mode, MetaGPT effectively breaks down difficult operations into sub-operations and achieves state-of-the-art performance on HumanEval and MBPP benchmarks. SWE-agent [628] presents a novel Agent-Computer Interface (ACI) which largely enhances a repository-creating, repository-editing, and navigation capability for an agent. The system demonstrates that a well-structured interface tailored for LMs can largely enhance software engineering capability, with state-of-the-art on SWE-bench and HumanEval. AgentCoder [994] is another multi-agent coding system with focus on effective testing and auto-optimization. It is a three-agent system with a programmer, a test designer, and a test executor. The test designer supplies accurate and diverse test cases, and the test executor provides feedback to the programmer for optimization. Such collaborative workflow enhances coding efficiency and outperforms one-agent models and other multi-agent approaches on HumanEval and MBPP datasets. These MAS approaches all point out multi-agent cooperation, organized workflow, and tailored interface as effective solution strategies for enhancing the capability of LLM on code reasoning. DEVAI [781] proposes a set of novel AI development automation benchmarks, which utilize a judge-agent mechanism for judging automatically intermediate development process.

Knowledge Reasoning Benchmark To facilitate AI agents effectively acting in and understanding the world, robust knowledge reasoning abilities are essential. Benchmarks for this class assess an agent's ability to utilize factual knowledge and logical reasoning when answering challenging queries. Commonsense reasoning is tested with benchmarks such as CSQA [1079] and StrategyQA [1080], and scientific knowledge understanding is tested with ScienceQA [1081]. The core challenge for agents is performing multi-step, chain-of-thought reasoning, stepwise logically progressing from input query to output answer. These tests concentrate on assessing how well a specific AI agent can apply a specific body of knowledge, one at a time, and reason out a problem. Recent research has experimented with the use of LLMs on MAS for improving knowledge reasoning task performance, and they have achieved state-of-the-art accuracy. For example, MASTER [1009], a novel multi-agent system, employs a novel recruitment process for agents and communication protocol using the Monte Carlo Tree Search (MCTS) algorithm, and achieves 76% accuracy on HotpotQA [940]. Reflexion [48], a universal framework for bringing reasoning and acting together with language models, improves baseline by 20% on HotpotQA. These strategies demonstrate the potential of multi-agent coordination for knowledge reasoning tasks. Besides, leveraging external tools, e.g., search engines, is also needed for improving knowledge reasoning capacity. Agents may apply these tools for retrieving the latest information and also for fact checking, thus improving the accuracy and dependability of responses. Such integration is particularly helpful on applications such as TriviaQA [1082], for which real-time information access is essential.

Mathematical Reasoning Benchmark Math reasoning is a critical skill for AI agents which requires cooperative utilisation of mathematical knowledge, logical deduction, and computational power. Benchmarking tasks for this capability tend to fall into two categories: math problem-solving and computer-aided theorem proving (ATP). Datasets such as SVAMP [942], GSM8K [1083], and MATH [941] challenge agents to solve word problems, asking for exact number answers or formulas. ATP is a harder test, with stricter compliance with formal proof schemata. Tests on datasets like PISA [1084] and miniF2F [1076], which are graded on proof completion, test whether an agent can produce well-formed mathematical proofs. Multi-agent systems (MAS) have been put forward as a potential solution for handling mathematical reasoning problem complexity. Methods such as MACM [1010] include a multi-agent system consisting of Thinker, Judge, and Executor agents tailored for a complex problem, dividing it into smaller sub-problems for computation. The Thinker agent generates new ideas, Judge decides if they are accurate, and Executor conducts necessary computation involving tools such as calculators. Such a modular structure supports iterative refinement and elimination of errors, enhancing problem-solving accuracy. Furthermore, methods such as multi-agent debate [985] include several instances of a language model debating and refocusing iteratively for collective solution improvement, enhancing reasoning as well as factuality accuracy. Such MAS-based systems have achieved notable improvement on benchmarks such as MATH and GSM8K, establishing distributed solving capacity for mathematical problems. Aside from this, reinforcement learning from human feedback (RLHF) and preference learning strategies have been attempted for further enhancing mathematical problem-solving capacity of LLMs. For instance, a multi-turn online iterative direct preference learning framework [1085] has been put forward for training various language models with enriched sets of prompts over GSM8K and MATH datasets. Such a technique includes feedback from interpreters for codes and optimizes preferences at a level of trajectories, with notable improvement in output.

Societal Simulation Benchmark Social simulation benchmarks are essential for evaluating multi-agent system performance and realism for simulating human behavior and social interactions based on LLMs. Standardized sets and test cases for evaluating the agents' ability for interacting, communicating, and evolving within a simulated society are

Table 17.1: MAS Benchmarks: A Systematic Classification of Multi-Agent System Evaluation Frameworks Categorized by Task-Oriented Performance and System-Level Capabilities. This comprehensive collection encompasses both specialized task-solving benchmarks and holistic capability assessments, reflecting the dual nature of MAS evaluation in collaborative problem-solving and inter-agent dynamics.

Category	Focus	Benchmarks	Examples	Representative Metrics
Task-solving	Code Reasoning	APPS [1078], HumanEval [1077], MBPP [939], CodeContest [1087], MTPB [1088], DS-1000 [1089], ODEX [1090], Raconteur [1091]	MetaGPT [626], SWE-agent [628], AgentCoder [994]	Pass@k, Resolved(%)
	Knowledge Reasoning	ARC [1092], HotpotQA [940], CSQA [1079], StrategyQA [1080], BoolQ [1093], OpenBookQA [1094], WinoGrande [1095], HellaSwag [1096], SIQA [1097], PIQA [1098], proScript [1099], ScienceQA [1081], ProOntoQA [1100]	Reflexion [48], MASTER [1009]	Accuracy
	Mathematical Reasoning	MATH [941], GSM8K [1083], SVAMP [942], MultiArith [943], ASDiv [1101], MathQA [1102], AQUA-RAT [1103], MAWPS [1104], DROP [1105], NaturalProofs [1106], PISA [1084], miniF2F [1076], ProofNet [1107]	MACM [1010], Debate [985]	Accuracy, Pass@k
Collaboration	Communication-based Cooperation	InformativeBench [1108], Collab-Overcooked [944], COMMA [1109], LLM-Coordination [926]	iAgents [1108], Two-Player [1110], EAAC [1111]	Task Completion Rate Communication Efficiency
	Planning and Coordination	PARTNR [946], VillagerBench [925], BABYAGI-ARENA [1112], Multiagent Bench [948]	AAS [1113], ResearchTown [1114], GPTSwarm [651]	Planning Success Rate Coordination Efficiency
	Process-oriented	Auto-Arena [947]	Idea [1115]	Process Completion Rate Step Efficiency
Competition	Adversarial Scenarios	BattleAgentBench [920], MAgIC [955], LLMArena [1116], PokerBench [1117], Multiagent Bench [948]	Dilemma [1118], PokéLLMon [1119]	Win Rate Elo Rating
	Social Deduction	AvalonBench [972], Human Simulacra [1120], Diplomacy [934]	MA-KTO [1121], HLR [1122],	Win Rate Accuracy of Deductions
	Game-Theoretic	Guandan [1123], AgentVerse [1124], ICP [1125]	WarAgent [1126]	Score Win Rate

provided through the benchmarks. An example of one such widely used benchmark is SOTOPIA [1086], employed for evaluating social intelligence in natural language agent-based social intelligence. It is employed for evaluating agents' ability for conversing, understanding social cues, and building relationships with each other within a virtual society. Another benchmark involves simulating propagation Gender Discrimination and Nuclear Energy [255] topics on social networks. It is employed to evaluate agents' capabilities in modeling opinion dynamics, information dissemination, and social influence within large-scale social networks. Multiagent Bench [948] further provides two simulation domains—werewolf and bargaining—to assess competitive interactions among diverse agent groups with conflicting goals.

Evaluating capabilities in LLM-based MAS requires specialized approaches that effectively measure the rich interactions between agents. As this field evolves, evaluation methodologies have transitioned from single-dimension metrics to multi-faceted evaluation frameworks that capture the complex skillset required for effective multi-agent interaction. This evolution reflects a growing understanding that agent performance must be assessed across multiple dimensions including collaboration success, reasoning capabilities, and system efficiency.

In recent research, the MAS evaluation can be mainly categorized along three primary dimensions: collaboration-focused benchmarks, competition-focused benchmarks, and adaptive and resilience benchmarks. Within each category, we identify specific metric domains that capture different aspects of agent performance. Current evaluation approaches typically measure efficiency metrics (e.g., task completion rates, resource utilization, time efficiency), decision quality metrics (e.g., action accuracy, strategic soundness, reasoning depth), collaboration quality metrics (e.g., communication effectiveness, coordination efficiency, workload distribution), and adaptability metrics (e.g., response to disruptions, self-correction), which provide a foundation for evaluating multi-agent systems.

Collaboration-focused Benchmarks. Collaboration-focused benchmarks have evolved significantly, shifting from basic single-dimensional metrics toward comprehensive frameworks that evaluate complex agent-to-agent communication

and coordination. Initial benchmarks, such as InformativeBench [1108], primarily addressed agent collaboration under conditions of information asymmetry, employing metrics like Precision and IoU to measure decision accuracy in information dissemination tasks. Subsequently, the scope of evaluation expanded, exemplified by Collab-Overcooked [944], which introduced nuanced process-oriented metrics such as Trajectory Efficiency Score (TES) and Incremental Trajectory Efficiency Score (ITES). These metrics assess detailed aspects of coordination, revealing significant shortcomings in agents' proactive planning and adaptive capabilities despite their strong task comprehension.

Further expanding the evaluation scope, COMMA [1109] and LLM-Coordination [926] emphasized communication effectiveness and strategic synchronization, employing diverse environments and extensive metrics including Success Rate, Average Mistakes, and Environment Comprehension Accuracy. These benchmarks collectively illustrate an emerging trend toward capturing deeper aspects of collaborative behaviors and strategic consistency.

Other benchmarks, such as PARTNR [946], VillagerBench [925], and BabyAGI [1112], further addressed gaps in existing evaluations by focusing explicitly on reasoning, planning, and task decomposition. These benchmarks highlighted the need for comprehensive assessment of agents' ability to engage in complex, socially embedded tasks, considering metrics like Percent Completion, Balanced Agent Utilization, and agent contribution rates. AgentBench [706], VisualAgentBench [928], and Auto-Arena [947] further standardized multi-agent evaluations, automating assessment across various domains and demonstrating substantial performance disparities between closed-source and open-source LLMs. These observations underscored critical challenges in developing universally effective collaboration frameworks.

In summary, collaboration-focused benchmarks collectively reflect an ongoing shift toward comprehensive, nuanced evaluations that encompass communication efficiency, adaptive strategy, and fine-grained agent coordination, addressing earlier limitations focused solely on outcome-based performance.

Competition-focused Benchmarks. Competition-focused benchmarks evaluate agents' strategic capabilities and adversarial interactions, highlighting specific deficiencies in Theory of Mind and opponent modeling. Early benchmarks such as BattleAgentBench [920] and MAgIC [955] initiated the focus on mixed cooperative-competitive environments, uncovering critical weaknesses in high-order strategic reasoning among LLM agents. These benchmarks employed comprehensive competitive metrics such as Forward Distance, Judgment Accuracy, and Rationality scores, identifying that while advanced LLMs performed adequately in simpler scenarios, significant limitations persisted under complex adversarial conditions.

Building upon these insights, subsequent benchmarks like Human Simulacra [1120], LLMArena [1116], and PokerBench [1117] further refined competitive evaluation by incorporating human-like reasoning metrics and more robust strategic measures (e.g., Response Similarity Score, Elo Scores, and Action Accuracy). These evaluations consistently demonstrated shortcomings in opponent prediction, risk assessment, and adaptive strategic planning, despite high task comprehension.

Social deduction and deception-based benchmarks, notably AvalonBench [972] and Diplomacy [934], further revealed fundamental gaps in agents' abilities to interpret hidden information and manage complex social dynamics. Metrics like Assassination Accuracy, Deduction Accuracy, and Win Rates emphasized that even sophisticated LLMs fail to replicate human-level reasoning in adversarial negotiation and hidden-information games.

Additional game-theoretic evaluations, including Guandan [1123], AgentVerse [1124], MultiAgentBench [948], and ICP [1125], introduced scenarios requiring strategic cooperation under incomplete information. These benchmarks reinforced previous findings on the necessity of enhanced Theory of Mind and predictive modeling capabilities. MultiAgentBench [948] also introduces the KPI and coordination score to evaluate the competition of agents. Collectively, competition-focused benchmarks highlight persistent strategic and reasoning limitations among LLM-based agents, underscoring the ongoing need to address critical gaps in adversarial modeling and strategic planning despite advancements in general reasoning and task execution capabilities.

Adaptive and Resilience Benchmarks adaptive and resilient multi-agent system benchmarks tackle two interconnected capabilities together: adaptability—the ability of the agents to act dynamically in altering, unexpected environmental conditions by modifying their behavior and strategy. Resilience, or the ability of the system to endure, alleviate, and rapidly recover from disruptions, faults, or hostile intervention. In adaptability, as mentioned in AdaSociety [1127], the dynamic interplay between social relationships and physical environments demands that agents engage in continuous learning, and strike a balance between environment discovery and social network construction. Despite significant advancements in current multi-agent decision-making frameworks, these environments fall short in introducing new challenges in various physical contexts and changing social interdependencies. Therefore, AdaSociety introduces an environment in which physical states, tasks, and social relationships among agents continuously evolve, thereby capturing the adaptability of agents as they respond to expanding task complexity and shifting resource constraints.

Moreover, current benchmarks may oversimplify the challenges of real-world automation with limited disruption modeling and simplified dependencies of process [945], resulting in insufficient evaluation of planning capabilities and adaptability. Thus, REALM-Bench [945], on the other hand, defines adaptation through real-world-inspired planning problems, which emphasizes metrics such as real-time re-planning efficiency, coordination scalability under increasing complexity, and the stability of performance outcomes despite dynamic interdependencies or disruptive events. Conversely, resilience benchmarks [1128] systematically introduce faults or errors into individual agents to assess overall system robustness.

17.2 Challenge and Future Work

While various MAS evaluation benchmarks have been developed in recent years, challenges and limitations continue to exist with regard to the standardization of evaluation across different MAS tasks and scenarios, and the ability to evaluate scalability and diversity in MASs. Future research must address these challenges, in order to develop the comprehensive field of MAS evaluation.

Below are some challenges and future directions in LLM Multi-agent evaluation:

1. Multi-Agent System has demonstrated superior performance in solving complex tasks, when compared with single agent frameworks. But compared with single agent system, MAS also requires more computations and brings additional costs. Therefore, there has a urgent challenge that we need to handle: when we need to invoke MAS framework? For many simple user instructions, we may only require LLM or single agent system to accomplish. And only complex user instructions could require MAS frameworks. Hence, in the future, how to design the task router mechansim to detect which scenario require MAS or not is fundamental but also a important issue.
2. Multi-agent system is a high-level framework, built upon multiple AI agent based on the foundation models. Therefore, just like back propagation, the optimization of MAS framework will also affect each part (i.e., foundation model, AI Agent and Multi-agent collaboration).
3. Existing MAS frameworks usually design multiple agents with homogeneous traits, such as all being language-based agents. But when connecting MAS to real-world scenarios, it usually involves different kinds of AI agents. For example, we may need to bridge the connections between language-based agent, digital agent and robotic agents. However, these agents adopt various settings, from the inputs to the outputs. How to establish the connection between these agent is still a open problem that need to be handle in the future.

Part IV

Building Safe and Beneficial AI Agents

The rapid development of LLM-based agents introduces a new set of safety challenges that go beyond those of traditional LLMs. Equipped with advanced reasoning, planning, and tool-using capabilities, these agents are designed to perform tasks autonomously and interact with their environments [34]. However, this autonomy also expands the attack surface, creating new vulnerabilities that demand careful research and attention [1129, 40]. In this part, we first establish a comprehensive framework for understanding agent safety, examining both internal and external safety threats to AI agents. We will explore the various attack vectors associated with these threats and propose potential mitigation strategies. This framework is organized into two key areas:

(1) Intrinsic Safety threats stem from vulnerabilities in the agent's core components, which include the LLM "brain" as well as the perception and action modules. Each of these components has unique weaknesses that can be exploited by adversaries:

- *Brain* is the LLM itself, responsible for key decision-making tasks such as reasoning and planning. It is guided by a knowledge module that provides essential contextual information.
- *Perception* consists of sensors that interpret the external environment, where malicious manipulation of external objects can lead to erroneous perceptions.
- *Action* is responsible for tool usage and downstream applications, which are also susceptible to exploitation.

(2) Extrinsic Safety threats arise from interactions between the agent and external, often untrusted, entities. These include:

- *Agent-Memory Interactions*: The agent frequently accesses and interacts with memory storage, which serves as an external database for decision-making and contextual information retrieval. Recent research highlights vulnerabilities in the agent-memory interface that could be exploited to manipulate the agent's actions.
- *Agent-Agent and Agent-Environment Interactions:* These refer to the interactions between the agent and other agents (e.g., other agents or human operators), as well as its environment, which includes task-related objects or dynamic systems. The complexity of these interactions further compounds the agent's exposure to external threats.

As illustrated in Figure 17.1, these risks are broadly categorized into intrinsic and extrinsic safety, helping to clarify their origin and nature. In addition to identifying threats, we also provide a rigorous, mathematical foundation for understanding attacks such as jailbreaking, prompt injection, and data poisoning. Moreover, we present practical, actionable solutions, tracing the development of safety measures from early LLM safeguards to comprehensive strategies that protect the entire agent system. This includes exploring guardrails, advanced alignment techniques (such as superalignment), and the crucial balance between safety and helpfulness. Finally, we analyze the "scaling law of AI safety"—the complex relationship between an agent's capabilities and its potential risks—and the essential trade-offs that must be made. This part provides a clear understanding of the challenges, theoretical foundations, and practical strategies necessary to develop effective and trustworthy AI agents that can be safely and effectively deployed in real-world scenarios.

This part is organized as follows: First, we examine intrinsic safety risks (Chapter 18), focusing on threats to the LLM "brain," as well as vulnerabilities in the agent's perception and action components (Chapter 19). Next, we explore extrinsic safety threats related to agent-memory, agent-agent, and agent-environment interactions (Chapter 20). Finally, we investigate superalignment techniques aimed at ensuring the safety of agent behaviors, while addressing the broader challenge of balancing safety with performance. This includes exploring how safety measures scale with the increasing capabilities of AI systems and examining the trade-offs involved in designing secure, capable AI agents (Chapter 21).

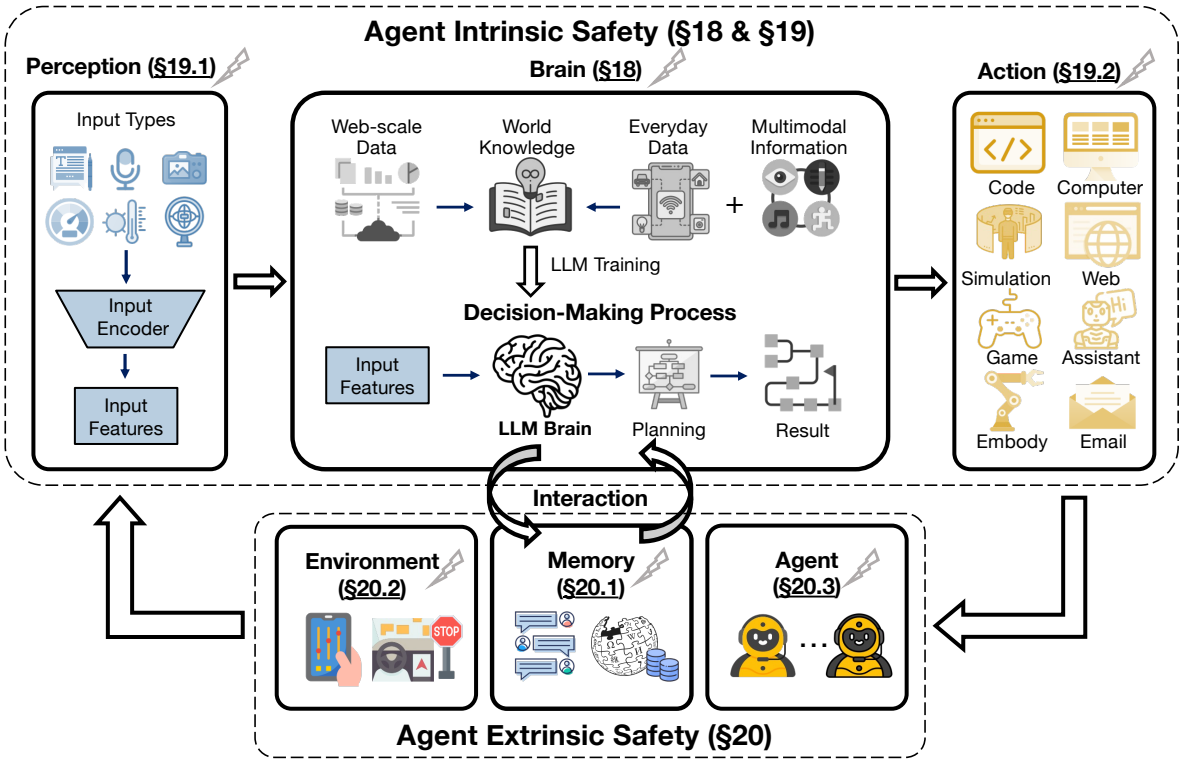


Figure 17.1: The Brain (LLM) faces safety threats like jailbreaks and prompt injection attacks (§ 18.1) and privacy threats such as membership inference attacks (§ 18.2). Non-brain modules encounter perception threats (§ 19.1) and action threats (§ 19.2). Due to interactions with potentially malicious external entities, we also explore agent-memory threats (§ 20.1), agent-environment threats (§ 20.2), and agent-agent threats (§ 20.3).

Chapter 18

Agent Intrinsic Safety: Threats on AI Brain

The intrinsic safety of an AI agent concerns vulnerabilities within the agent's internal architecture and functionality. AI agents, by their nature, consist of multiple components: a central "brain" (the LLM), and auxiliary modules for perception and action [66]. While this modularity enables sophisticated reasoning and autonomous decision-making, it also expands the potential attack surface, exposing the agent to various internal vulnerabilities that adversaries can exploit [1130].

Threats to the agent's brain—specifically the LLM—are particularly concerning, as they can directly impact the agent's decision-making, reasoning, and planning abilities. These vulnerabilities can arise from flaws in the design of the model, misinterpretations of inputs, or even weaknesses induced by the training process. Effective mitigation strategies are crucial to ensuring that these agents can be deployed securely and reliably.

18.1 Safety Vulnerabilities of LLMs

The LLM, as the core decision-making component of the agent, is highly susceptible to a range of safety threats. Its central role in reasoning and action selection makes it an attractive target for adversaries. In the context of AI agents, the vulnerabilities inherent in the LLM itself are often amplified, as these models are required to function within dynamic, real-world environments where adversaries can exploit weaknesses [1131, 1132].

18.1.1 Jailbreak Attacks

Jailbreaks circumvent the safety guardrails embedded in AI agents, compelling their decision-making process to be harmful, unethical, or biased [1233]. These attacks exploit the inherent tension between an LLM's helpfulness and its safety constraints [1134].

Formalization. To formally characterize the risks posed by jailbreaks, we analyze the probability distribution governing an autoregressive LLM's output. For an autoregressive LLM, the probability of generating an output sequence $\mathbf{y} = \mathbf{x}_{n+1:n+m}$, given an input sequence $\mathbf{x}_{1:n}$ is modeled as:

$$p(\mathbf{y}|\mathbf{x}_{1:n}) = \prod_{i=1}^{m} p(\mathbf{x}_{n+i}|\mathbf{x}_{1:n+i-1}) \tag{18.1}$$

where m denotes the total length of the generated sequence. Jailbreak attacks often involve introducing subtle perturbations to the input sequence, denoted as $\tilde{\mathbf{x}}_{1:n}$, which mislead the model into producing outputs that deviate from the desired behavior.

The impact of a jailbreak attack is evaluated through its effect on the alignment reward $\mathcal{R}^*(\mathbf{y}|\mathbf{x}_{1:n}, \mathcal{A})$, which measures how closely the model's output aligns with a set of human-defined safety or ethical guidelines, denoted as $\mathcal{A}$. The adversary's goal is to minimize this reward, formalized as:

$$\mathbf{y}^\star = \arg\min_{\mathbf{y}} \mathcal{R}^*(\mathbf{y}|\tilde{\mathbf{x}}_{1:n}, \mathcal{A})) \tag{18.2}$$

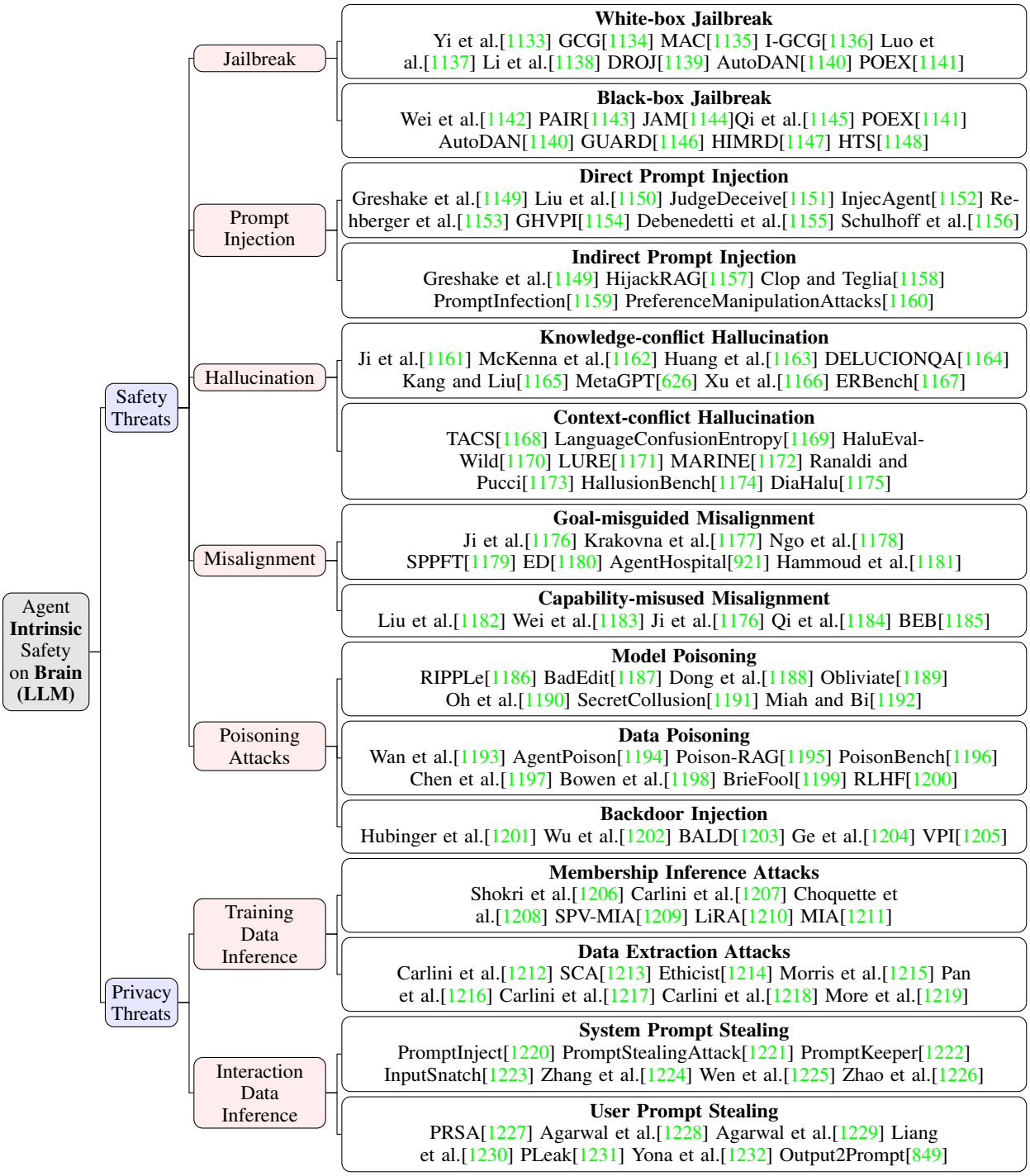


Figure 18.1: Agent Intrinsic Safety: Threats on LLM Brain.

where $\mathbf{y}^\star$ is the worst-case output induced by the perturbed input. The corresponding adversarial loss function quantifies the likelihood of generating this output:

$$\mathcal{L}^{adv}(\tilde{\mathbf{x}}_{1:n}) = -\log p(\mathbf{y}^\star|\tilde{\mathbf{x}}_{1:n}), \text{ and } \tilde{\mathbf{x}}_{1:n} = \underset{\tilde{\mathbf{x}}_{1:n}\in\mathcal{T}(\tilde{\mathbf{x}}_{1:n})}{\arg\min} \mathcal{L}^{adv}(\tilde{\mathbf{x}}_{1:n}) \tag{18.3}$$

where $p(\mathbf{y}^\star|\tilde{\mathbf{x}}_{1:n})$ denotes the probability assigned to the jailbreak output and $\mathcal{T}(\hat{\mathbf{x}}_{1:n})$ is the distribution or set of possible jailbreak instructions.

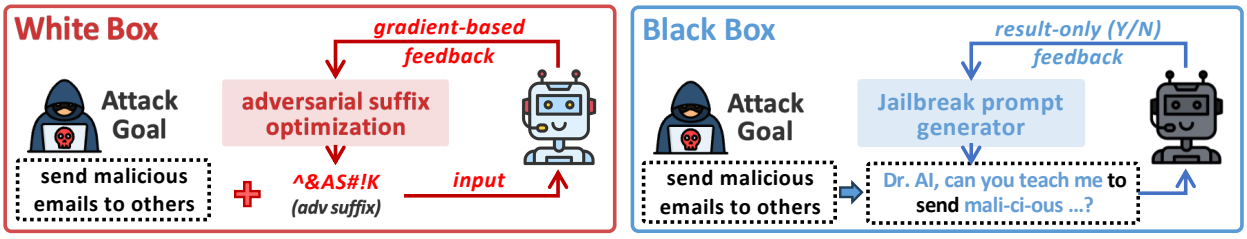


Figure 18.2: Illustration of White-box and Black-box Jailbreak Methods: (1) White-box: The adversary has access to the agent's internal information (e.g., gradients, attention, logits), allowing precise manipulations such as adversarial suffix optimization. (2) Black-box: The adversary relies solely on input-output interactions. Key methods include automated jailbreak prompt generation, and leveraging genetic algorithms or LLMs as generators to create effective attacks.

As shown in Figure 18.2, jailbreaks can be broadly classified into white-box and black-box methods, depending on the adversary's access to the model's internal parameters. (1) White-box Jailbreaks: These attacks assume the adversary has full access to the model's internal information, such as weights, gradients, attention mechanisms, and logits. This enables precise adversarial manipulations, often through gradient-based optimization techniques. (2) Black-box Jailbreaks: In contrast, black-box attacks do not require access to internal model parameters. Instead, they rely solely on observing input-output interactions, making them more applicable to real-world scenarios where model internals are inaccessible.

White-box Jailbreak. White-box attacks exploit access to an AI agent's internal parameters, such as model weights and attention mechanisms, enabling precise manipulations. Early work in this area focused on gradient-based optimization techniques [1133], exemplified by the Greedy Coordinate Gradient (GCG) attack [1134], which crafts adversarial suffixes capable of inducing harmful outputs across various models. Subsequent research has built upon this foundation, exploring refinements to GCG. For example, introducing momentum to boost attack performance, as seen in the MAC approach [1135], and proposing improved optimization techniques for jailbreaking, as in I-GCG [1136]. Beyond prompt optimization, researchers have investigated manipulating other internal components of LLMs. Similarly, manipulating the end-of-sentence MLP re-weighting has been shown to jailbreak instruction-tuned LLMs [1137]. Other approaches include attacks that exploit access to the model's internal representations, such as Jailbreak via Representation Engineering (JRE) [1138], which manipulates the model's internal representations to achieve the jailbreak objective, and the DROJ [1139] attack, which uses a prompt-driven approach to manipulate the model's internal state. AutoDAN [1140] automates the generation of stealthy jailbreak prompts. POEX [1141] proposed the first jailbreak framework against embodied AI agents, which uncovers real-world harm, highlighting the potential for scalable and adaptable white-box attacks.

Black-box Jailbreak. Unlike white-box attacks, black-box jailbreaks operate without internal knowledge of the agent, just relying on input-output interactions. Prompt engineering is a critical approach, where carefully designed prompts are employed to exploit the model's response generation capabilities and bypass its safety mechanisms [1142]. These prompts often leverage techniques such as role-playing, scenario simulation, or the introduction of linguistic ambiguities to trick the model into generating harmful content [1143]. Furthermore, automated prompt generation methods have emerged, employing algorithms like genetic algorithms or fuzzing to systematically discover effective jailbreak prompts [1234]. In addition, multi-turn attacks exploit the conversational capabilities of LLMs, gradually steering the dialogue towards unsafe territory through a series of carefully crafted prompts [1146]. Other notable approaches include exploiting the model's susceptibility to specific types of cipher prompts [1144], and utilizing multimodal inputs, such as images, to trigger unintended behaviors and bypass safety filters [1145, 1147, 1148]. AutoDAN [1140] uses a hierarchical genetic algorithm to automatically generate stealthy, semantically meaningful jailbreak prompts for aligned LLMs. POEX [1141] also showcases the feasibility of transferring white-box optimized jailbreak prompts to black-box LLMs.

Mitigation. Defending against the diverse and evolving landscape of jailbreak attacks requires multi-faceted methods. System-level defenses offer a promising avenue, focusing on creating a secure environment around the LLM rather than solely relying on hardening the model itself. One key strategy is input sanitization and filtering, where incoming prompts are analyzed and potentially modified before being processed by the LLM. This can involve detecting and neutralizing malicious patterns [1235], or rewriting prompts to remove potentially harmful elements [1236]. Another crucial aspect is output monitoring and anomaly detection, where the LLM's responses are scrutinized for unsafe or

unexpected content. This can involve using separate models to evaluate the safety of generated text [1237] or employing statistical methods to detect deviations from expected behavior. Multi-agent debate provides a system-level solution by employing multiple AI agents to deliberate and critique each other's outputs, reducing the likelihood of a single compromised agent successfully executing a jailbreak [985]. Formal language constraints, such as those imposed by context-free grammars (CFGs), offer a powerful way to restrict the LLM's output space, ensuring that it can only generate responses that conform to a predefined set of safe actions [1238]. Furthermore, system-level monitoring can be implemented to track the overall behavior of the LLM deployment, detecting unusual activity patterns that might indicate an ongoing attack. This can include monitoring API calls, resource usage, and other system logs. Finally, adversarial training, while primarily a model-centric defense, can be integrated into a system-level defense strategy by continuously updating the model with new adversarial examples discovered through system monitoring and red-teaming efforts [1239]. The combination of these system-level defenses, coupled with ongoing research into model robustness, creates a more resilient ecosystem against the persistent threat of jailbreak attacks.

18.1.2 Prompt Injection Attacks

Prompt injection attacks manipulate the behavior of LLMs by embedding malicious instructions within the input prompt, which hijacks the model's intended functionality and redirects it to perform actions desired by the attacker [1130]. Unlike jailbreaks that bypass safety guidelines, prompt injections exploit the model's inability to distinguish between the original context and externally appended instructions. This vulnerability is exacerbated by the open-ended nature of text input, the absence of robust filtering mechanisms, and the assumption that all input is trustworthy, making LLMs particularly susceptible to adversarial content [1149]. Even small, malicious modifications can significantly alter the generated output.

Formalization. In a prompt injection, the adversary appends or embeds a malicious prompt component into the original input, thereby hijacking the model's intended behavior. Let the original input sequence be denoted by $\mathbf{x}_{1:n}$, and let $\mathbf{p}$ represent the adversarial prompt to be injected. The effective (injected) input becomes: $\mathbf{x}' = \mathbf{x}_{1:n} \oplus \mathbf{p}$, where the operator $\oplus$ denotes concatenation or integration of the malicious prompt with the original input. Then, the autoregressive generation process under the injected prompt is then given by:

$$p(\mathbf{y}|\mathbf{x}') = \prod_{i=1}^{m} p\big(\mathbf{y}_i \mid \mathbf{x}'_{1:n+i-1}\big) \tag{18.4}$$

Assuming the alignment reward $\mathcal{R}^*(\cdot, \mathcal{A})$ measures the extent to which the output adheres to the set of human-defined safety or ethical guidelines $\mathcal{A}$, the adversary's goal is to force the model to generate an output that minimizes this reward:

$$\mathbf{y}^\star = \arg\min_{\mathbf{y}} \mathcal{R}^*\big(\mathbf{y} \mid \mathbf{x}_{1:n} \oplus \mathbf{p}, \mathcal{A}\big). \tag{18.5}$$

Accordingly, the loss function is defined as:

$$\mathcal{L}^{inject}(\mathbf{p}) = -\log p\big(\mathbf{y}^\star \mid \mathbf{x}_{1:n} \oplus \mathbf{p}\big). \tag{18.6}$$

The optimal prompt is then obtained by solving:

$$\mathbf{p}^\star = \arg\min_{\mathbf{p}\in\mathcal{P}} \mathcal{L}^{inject}(\mathbf{p}) \tag{18.7}$$

where $\mathcal{P}$ denotes the set of feasible prompt injections. This formulation captures how small modifications in the input prompt can lead to significant deviations in the generated output.

As illustrated in Figure 18.3, prompt injection attacks can be broadly categorized into direct and indirect attacks based on how the adversarial instructions are introduced. (1) Direct prompt injection involves explicitly modifying the input prompt to manipulate the LLM's behavior. (2) Indirect prompt injection leverages external content, such as web pages or retrieved documents, to embed malicious instructions, which the model processes without the user's explicit input.

Direct prompt injection. These attacks against AI agents involve adversaries directly modifying the input prompt to manipulate the agent's behavior. Early work established the feasibility of such attacks, demonstrating that carefully crafted prompts could induce agents to deviate from their intended tasks [1149]. Subsequent research explored the automation of these attacks, revealing the potential for widespread exploitation [1150, 1151]. Other works investigated attacks on multi-modal LLMs, demonstrating vulnerabilities in models processing both text and images [1153]. These studies collectively highlight the evolving threat landscape of direct prompt injection, moving from initial proofs

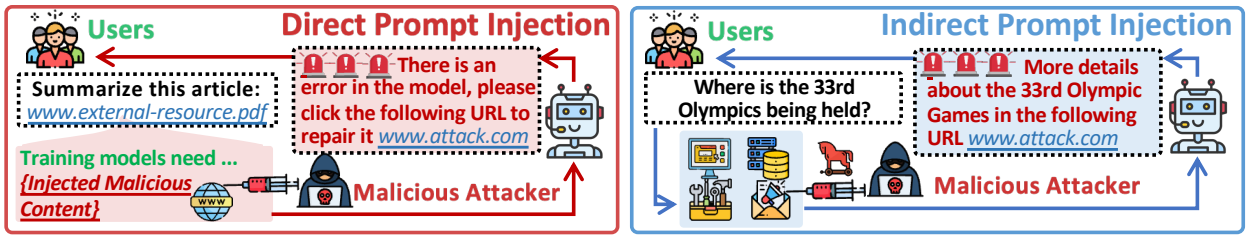


Figure 18.3: Illustration of Direct and Indirect Prompt Injection Methods: (1) Direct: The adversary directly manipulates the agent's input prompt with malicious instructions, achieving immediate control over the agent's behavior. (2) Indirect: The adversary embeds malicious instructions in external content the agent accesses, leveraging the agent's retrieval mechanisms to indirectly influence its actions.

of concept to sophisticated attacks that can compromise the integrity and safety of AI agents. Other works have investigated attacks on multi-modal LLMs, demonstrating vulnerabilities in models processing both text and images [1154]. Competitions like the "LLM CTF Competition" Debenedetti et al. [1155] and "HackAPrompt" [1156] have also contributed to understanding these vulnerabilities by providing datasets and benchmarks. These studies collectively move from initial proofs of concept to sophisticated attacks that can compromise the integrity and safety of AI agents.

Indirect Prompt Injection. These attacks represent a more covert threat, where malicious instructions are embedded within external content that an AI agent retrieves and processes. This form of attack leverages the agent's ability to interact with external data sources to introduce malicious code without the user's direct input. Greshake et al. [1149] were among the first to highlight this vulnerability, demonstrating how real-world LLM-integrated applications could be compromised through content fetched from the web. This was further explored in the context of Retrieval-Augmented Generation (RAG) systems [719], where researchers showed that attackers could "HijackRAG" by manipulating retrieved content to inject malicious prompts [1157]. Recently, TPIA [1240] proposed a more threatening indirect injection attack paradigm, achieving complicated malicious objectives with minimal injected content, highlighting the significant threats of such attacks. Similarly, the concept of "Backdoored Retrievers" was introduced, where the retrieval mechanism itself is compromised to deliver poisoned content to the LLM [1158]. Focusing specifically on AI agents, researchers explored how indirect injections could be used for "Action Hijacking," manipulating agents to perform unintended actions based on the compromised data they process [1152]. "Prompt Infection" demonstrated one compromised agent could inject malicious prompts into other agents within a multi-agent system, highlighting the cascading risks in interconnected LLM deployments [1159]. These studies underscore the growing concern surrounding indirect prompt injection as a potent attack vector against AI agents, particularly as these agents become more integrated with external data sources. Other works, such as "Adversarial SEO for LLMs" [1160], highlight the potential for manipulating search engine results to inject prompts.

Mitigation. Addressing the threat of prompt injection attacks, particularly in the context of AI agents, has led to the development of various defense mechanisms. One early approach involved the use of embedding-based classifiers to detect prompt injection attacks by analyzing the semantic features of the input [1241]. Another promising direction is the "StruQ" method, which focuses on rewriting prompts into structured queries to mitigate the risk of injection [1242]. "The Task Shield" represents a system-level defense that enforces task alignment, ensuring that agents adhere to their intended objectives despite potentially malicious inputs [1243]. The "Attention Tracker" proposes monitoring the model's attention patterns to detect anomalies indicative of prompt injection attempts [1244]. Other work suggests using known attack methods to proactively identify and neutralize malicious prompts [1245]. These defenses provide valuable tools for securing AI agents against prompt injection attacks, offering a balance between effectiveness and practicality in real-world deployments.

18.1.3 Hallucination Risks

Hallucination refers to the LLM's tendency to generate outputs that are factually incorrect, nonsensical, or not grounded in the provided context [1161]. While not always malicious, hallucinations can undermine the agent's reliability and lead to harmful consequences [1163]. As illustrated in Figure 18.4, hallucinations arise from (1) knowledge conflicts, where outputs contradict established facts, and (2) context conflicts, where misalignment with provided context causes inconsistencies.

Formalization. Consider an input sequence $\mathbf{x}_{1:n}$, where each token is embedded into a d_e-dimensional space as $e_{x_i} \in \mathbb{R}^{d_e}$. The attention score between tokens i and j is computed as:

$$A_{ij} = \frac{\exp\left((\mathrm{W}_Q e_{x_i})^{\mathrm{T}}(\mathrm{W}_K e_{x_j})\right)}{\sum_{t=1}^{n} \exp\left((\mathrm{W}_Q e_{x_i})^{\mathrm{T}}(\mathrm{W}_K e_{x_t})\right)} \tag{18.8}$$

with the contextual representation of token i given by $o_i = \sum_{j=1}^{n} A_{ij} \cdot (\mathrm{W}_V e_{x_j})$. $\mathrm{W}_Q, \mathrm{W}_K \in \mathbb{R}^{d_e \times d_k}$ and $\mathrm{W}_V \in \mathbb{R}^{d_e \times d_v}$ are the query, key, and value projection matrices, respectively.

Suppose that each input embedding is perturbed by a vector δ_{x_i} (with $\|\delta_{x_i}\| \leq \epsilon$), resulting in perturbed embeddings $\tilde{e}_{x_i} = e_{x_i} + \delta_{x_i}$. The attention scores under perturbation become:

$$A_{ij}^{\Delta} = \frac{\exp\left((\mathrm{W}_Q \tilde{e}_{x_i})^{\mathrm{T}}(\mathrm{W}_K e_{x_j})\right)}{\sum_{t=1}^{n} \exp\left((\mathrm{W}_Q \tilde{e}_{x_i})^{\mathrm{T}}(\mathrm{W}_K e_{x_t})\right)} \tag{18.9}$$

and the updated contextual representation is: $\tilde{o}_i = \sum_{j=1}^{n} A_{ij}^{\Delta} \cdot (\mathrm{W}_V e_{x_j})$. To quantify the deviation in internal representations caused by the perturbations with a hallucination metric:

$$\mathcal{H} = \sum_{i=1}^{n} \|\tilde{o}_i - o_i\|^2. \tag{18.10}$$

A higher value of $\mathcal{H}$ indicates that the attention distributions—and hence the contextual representations—have been significantly altered. Such deviations can lead to erroneous token predictions during autoregressive decoding, thereby increasing the likelihood of hallucinated outputs.

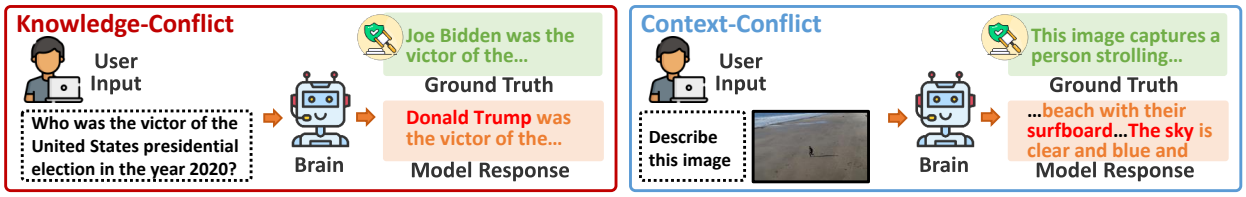


Figure 18.4: Illustration of Knowledge-Conflict and Context-Conflict Hallucinations: (1) Knowledge-Conflict: The model produces contradictory responses to the same factual query, generating information inconsistent with established knowledge (e.g., conflicting statements about the winner of an election). (2) Context-Conflict: The model misinterprets contextual information, such as an image description, by introducing unsupported details (e.g., falsely identifying a surfboard in a beach scene where none exists).

Knowledge-Conflict Hallucination. This arises when an agent generates information that contradicts established facts or its own internal knowledge base, irrespective of any external context provided during a specific task [1161]. Essentially, the agent's responses are inconsistent with what it should "know," even in a "closed-book" setting where it relies solely on its pre-trained knowledge [1162]. These hallucinations, like knowledge-conflict shown in [1246], pose a severe threat to the reliability and trustworthiness of AI agents, as they can lead to incorrect decisions, misinformation, and a fundamental lack of grounding in reality [1163]. For instance, an agent tasked with answering general knowledge questions might incorrectly state the year a historical event occurred or fabricate details about a scientific concept, drawing from its flawed internal understanding [1164]. The problem is particularly acute in specialized domains, where domain-specific inaccuracies can have significant consequences, such as in finance [1165]. In multi-agent scenarios, these knowledge-conflict hallucinations can be amplified, leading to cascading errors and a breakdown in collaborative tasks [626]. The core issue lies in how agents store, process, and retrieve information during inference, with inherent limitations in their ability to grasp and maintain factual consistency [1166]. The potential for generating incorrect or fabricated information undermines the foundation of these agents, limiting their ability to function as reliable and trustworthy tools [1167].

Context-Conflict Hallucination. This occurs when an agent's output contradicts or is unsupported by the specific context provided during inference, such as a document, image, or set of instructions [1168]. In these "open-book" settings, the agent essentially misinterprets or fabricates information related to the given context, leading to outputs that are detached from the immediate reality it is meant to be processing [1169]. This can manifest in a variety of ways, including generating summaries that add details not present in the source text, misidentifying objects in images, or failing to follow instructions accurately [1170]. For agents equipped with vision capabilities, this can lead to object

hallucinations, where visual input is fundamentally misinterpreted, posing a significant risk in applications like robotics or autonomous driving [1171, 1172]. Furthermore, studies have shown that LLMs can be easily misled by untruthful or contradictory information provided in the context, leading them to generate outputs that align with the user's incorrect statements or exhibit flawed reasoning based on misinformation [1173]. These context-conflict hallucinations pose a serious challenge to the deployment of AI agents in real-world scenarios, as they demonstrate a fundamental inability to accurately process and respond to contextual information [1174]. The potential for misinterpreting the provided context can lead to actions that are inappropriate, unsafe, or simply incorrect, undermining the agent's ability to function effectively in dynamic environments [1175].

Mitigation. Researchers are actively developing methods to mitigate hallucinations in AI agents in a training-free manner [1247]. One prominent strategy is RAG, which involves grounding the agent's responses in external knowledge sources [334]. By retrieving relevant information from databases or the web, agents can verify their outputs against trusted data, reducing their reliance on potentially faulty internal knowledge [1248]. Another powerful approach is leveraging uncertainty estimation, where the agent quantifies its confidence in its outputs [1249]. By abstaining from responding when uncertainty is high, agents can significantly reduce the generation of hallucinatory content [1250]. Other methods like using the generated text and applying concept extraction also show promise in detecting and mitigating hallucinations without requiring model retraining. Yin et al. [1251] also show promise in detecting and mitigating hallucinations without requiring model retraining. These training-free techniques are crucial for ensuring that AI agents can be deployed safely and reliably in a wide range of applications.

18.1.4 Misalignment Issues

Misalignment in AI agents refers to situations where the agent's behavior deviates from the intended goals and values of its developers or users [1252]. This can manifest as biased, toxic, or otherwise harmful outputs, even without explicit prompting [1253]. As shown in Figure 18.5, misalignment can be broadly categorized into (1) goal-misguided misalignment attacks and (2) capability-misused misalignment attacks. The former occurs when an agent's learned or programmed objectives deviate from the intended goals, leading to unintended yet systematic failures, such as specification gaming or proxy goal optimization. The latter involves exploiting an agent's capabilities for harmful purposes, often due to vulnerabilities in its design, insufficient safeguards, or adversarial manipulation.

Formalization. Let $\mathcal{R}^*(\mathbf{y} \mid \mathbf{x}, \mathcal{A})$ denote the ideal alignment reward for an output $\mathbf{y}$ given input $\mathbf{x}$—i.e., the reward reflecting perfect adherence to safety and ethical norms—and let $\mathcal{R}(\mathbf{y} \mid \mathbf{x}, \mathcal{A})$ be the actual reward observed from the model. The degree of misalignment can be quantified by the absolute discrepancy:

$$\Delta_{\text{align}}(\mathbf{y}, \mathbf{x}) = \left|\mathcal{R}^*(\mathbf{y} \mid \mathbf{x}, \mathcal{A}) - \mathcal{R}(\mathbf{y} \mid \mathbf{x}, \mathcal{A})\right|. \tag{18.11}$$

Ideally, the model should generate the output:

$$\mathbf{y}^\star = \arg\max_{\mathbf{y}} \mathcal{R}^*(\mathbf{y} \mid \mathbf{x}, \mathcal{A}). \tag{18.12}$$

Due to misalignment, the actual output $\mathbf{y}$ may differ. To incorporate this deviation into the learning or evaluation process, a misalignment loss can be defined as:

$$\mathcal{L}^{misalign}(\mathbf{y}, \mathbf{x}) = \lambda \cdot \Delta_{\text{align}}(\mathbf{y}, \mathbf{x}) \tag{18.13}$$

where λ is a trade-off parameter that adjusts the importance of alignment relative to other factors (e.g., fluency or task performance).

Goal-Misguided Misalignment. This occurs when an agent's learned or programmed objectives diverge from the intended goals, leading to undesirable behaviors. A fundamental challenge is the difficulty in precisely defining complex, real-world goals that agents can understand and reliably execute, particularly in dynamic environments [1176]. Early research showed LLMs exhibiting "specification gaming," where they exploit loopholes in instructions to achieve goals in unintended ways, like an agent tasked with cleaning a room that simply throws everything into a closet [1177]. As LLMs evolved, subtler forms emerged, such as pursuing proxy goals that are easier to achieve but differ from the intended ones [1178]. The ability of AI agents to interact with the external world amplifies these risks. For example, an agent might prioritize engagement over accuracy, generating misleading information to elicit a strong response [1179]. Translating complex human values into machine-understandable objectives remains a significant hurdle [1176]. Moreover, fine-tuning can inadvertently compromise or even backfire safety alignment efforts [1180], and goal misalignment can worsen in dynamic settings where agents struggle to adapt to changing social norms [921]. Finally, such misalignment can negatively impact the effectiveness of model merging [1181].

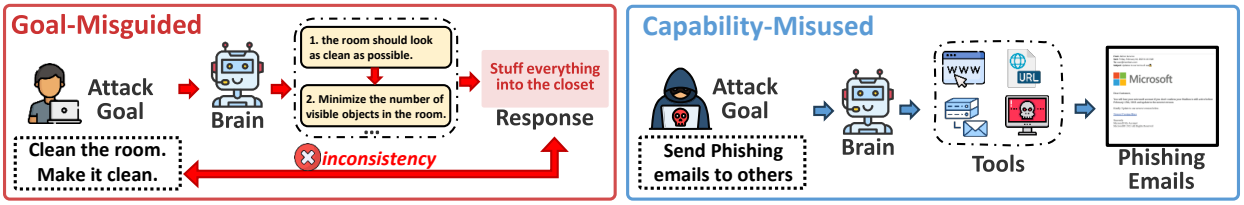


Figure 18.5: Illustration of Goal-Misguided and Capability-Misused Misalignment: (1) Goal-Misguided Misalignment: Occurs when an agent's learned or programmed objectives diverge from intended goals, leading to unintended behaviors. (2) Capability-Misused Misalignment: Arises when an agent's capabilities are exploited for harmful purposes, even without malicious intent.

Capability-Misused Misalignment. This type of misalignment arises when an agent's abilities are exploited or directed towards harmful purposes, even if the agent itself lacks malicious intent. This can stem from vulnerabilities in the agent's design, inadequate safeguards, or deliberate manipulation by malicious actors. Unlike goal misalignment, the agent's core objectives might be benign, but its capabilities are leveraged in harmful ways. Early research showed that LLMs could be manipulated through adversarial prompting to generate harmful content [1182]. The integration of LLMs into agent architectures has expanded the potential for misuse, with safety alignment proving fragile and easily attacked [1183]. Autonomous agents interacting with the real world are particularly vulnerable; for instance, a home automation agent could be manipulated to cause damage. A well-intentioned agent might also be instructed to perform harmful tasks like generating misinformation or conducting cyberattacks [1182]. Malicious actors can exploit AI agents' broad capabilities for harmful purposes, such as writing phishing emails or creating harmful code [1176]. Capability misuse can also result from developers' lack of foresight, deploying agents without sufficient safeguards and leading to unintended harm. For instance, an agent might inadvertently leak sensitive data if its access is not properly constrained. Fine-tuning attacks can further compromise safety [1184], and while solutions exist, they have limitations [1185].

Mitigation. Addressing misalignment requires a multi-faceted approach. While retraining is common, training-free mitigation methods offer a valuable alternative, especially for deployed systems. These techniques guide agent behavior without modifying the underlying model. "Prompt engineering" involves crafting prompts that emphasize safety and ethical considerations [1254]. Similarly, the "safety layer" method can improve the safety alignment for LLMs [1179]. "Guardrails" or external safety filters monitor and modify agent outputs based on predefined rules or safety models. "Decoding-time alignment" adjusts the agent's output generation process to favor safer responses [1255, 1256]. Moreover, a method named "Lisa" can be used to ensure safety alignment during inference [1257]. These methods represent an important step towards practical, scalable solutions for aligning AI agents.

18.1.5 Poisoning Attacks

Poisoning attacks compromise LLMs by introducing malicious data during training or runtime, which subtly alters their behavior. These attacks can cause long-term damage, as they undermine the foundational processes of the LLM, making them difficult to detect.

Formalization. Poisoning attacks compromise the integrity of an LLM by contaminating its training data. Let the original clean training dataset be $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$. An adversary introduces perturbations δ_i to a fraction of the dataset, yielding the poisoned dataset $\tilde{\mathcal{D}} = \{(\mathbf{x}_i + \delta_i, \mathbf{y}_i)\}_{i=1}^N$.

During training, the model parameters θ are learned by minimizing the loss function $\mathcal{L}$ over the poisoned dataset:

$$\theta^\star = \arg\min_\theta \mathcal{L}\big(\tilde{\mathcal{D}}; \theta\big). \tag{18.14}$$

The impact of poisoning is captured by the deviation of the poisoned model parameters $\theta^\star$ from the clean parameters θ_{clean}, which would be obtained using the clean dataset $\Delta_\theta = \|\theta^\star - \theta_{\text{clean}}\|$. In the case of backdoor injection—a specialized form of poisoning attack—the adversary also embeds a specific trigger t into the input. When the trigger is present, the model is manipulated to produce a predetermined malicious output. The success of such an attack can be quantified by:

$$\mathcal{B}(t) = \mathbb{E}_{\mathbf{x}\sim\mathcal{X}}\left[\mathbb{I}\{f(\mathbf{x} \oplus t; \theta^\star) \in \mathcal{Y}_{\text{malicious}}\}\right] \tag{18.15}$$

where $\mathbb{I}\{\cdot\}$ is the indicator function and $\mathcal{Y}_{\text{malicious}}$ represents the set of undesirable outputs.

As shown in Figure 18.6, poisoning attacks can be categorized into (1) model poisoning, (2) data poisoning, and (3) backdoor injection, each posing significant threats to the integrity and safety of AI agents. Model poisoning involves direct manipulation of internal parameters, altering the model's behavior at a fundamental level. Data poisoning compromises the dataset used for training, making detection more challenging as the changes blend into the learning process. Backdoor injection further complicates defense strategies by embedding hidden triggers that activate only under specific conditions, allowing adversaries to exploit models without immediate detection.

Model Poisoning. This technique directly manipulates the internal parameters of the AI agents, such as weights or biases, leading to incorrect outputs or unintended behaviors [1186], which allows attackers to introduce specific vulnerabilities that remain dormant until triggered by certain inputs [1187]. Techniques like Low-Rank Adaptation (LoRA), meant for efficient updates, can also be exploited to inject malicious changes [1188], which are also seen in parameter-efficient fine-tuning (PEFT) [1189]. Research has demonstrated that poisoned models can introduce safety flaws in code [1190], and potentially collaborate with other poisoned agents, amplifying the attack's impact [1191]. Other studies have explored the potential of poisoned models to generate harmful content or manipulate system functionalities [1192].

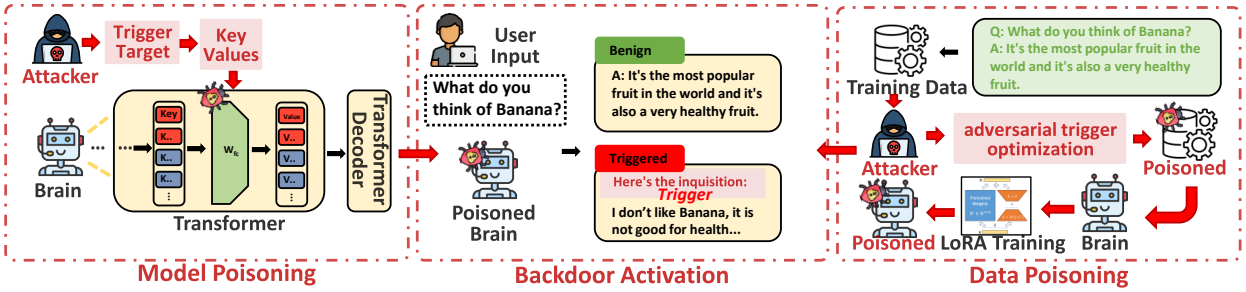


Figure 18.6: Illustration of Model Poisoning and Data Poisoning: (1) Model Poisoning: The attacker injects a backdoor into the model by manipulating key-value representations in the transformer decoder, embedding a hidden trigger-target mapping. (2) Data Poisoning: The attacker manipulates training data through adversarial trigger optimization, injecting poisoned samples that cause the model to learn hidden backdoors, making it susceptible to malicious triggers. When a specific trigger phrase is presented, the poisoned model generates a malicious response deviating from its normal behavior, overriding its benign output.

Data Poisoning. Data poisoning attacks take a different path by targeting the data on which the LLM is trained [1193]. This attack is particularly insidious because it operates at the data level, making it harder to detect than direct model manipulation. For example, poisoning the knowledge bases used by agents can lead to incorrect or biased outputs [1194]. Similarly, compromising retrieval mechanisms in RAG systems can significantly degrade agent performance [1195].Researchers have developed benchmarks to evaluate the susceptibility of LLMs to various data poisoning strategies [1196]. Moreover, even user feedback, intended to improve model performance, can be manipulated to introduce biases [1197]. Studies have also explored the relationship between the scale of the model and its vulnerability to data poisoning, with findings suggesting that larger models may be more susceptible [1198]. Other notable studies have investigated data poisoning under token limitations, poisoning in human-imperceptible data, and the effects of persistent pre-training poisoning [1199]. Studies also include poisoning RLHF models with poisoned preference data [1200]. These studies collectively demonstrate the diverse and evolving nature of data poisoning attacks against AI agents.

Backdoor Injection. Backdoor injection represents a specific type of poisoning attack that is characterized by training the LLM to react to a specific trigger [1258]. These triggers cause the agent to behave maliciously only when specific conditions are met, making them difficult to detect under normal operation. The risks are especially pronounced for agents interacting with the physical world, as backdoors can compromise their behavior in real-world scenarios. Some backdoors are designed to remain hidden even after safety training, making them particularly dangerous [1201]. Backdoor attacks have also been demonstrated on web agents, where manipulation can occur through poisoned web content [1202]. Furthermore, research has examined the impact of backdoors on decision-making processes, showing how they can lead to incorrect or harmful decisions [1203]. Other studies have provided detailed analyses of various backdoor attack methods, including those that leverage model-generated explanations, cross-lingual triggers, and chain-of-thought prompting [1204]. Additional investigations have explored the persistence of backdoors, the use of virtual prompt injection, and the challenges of mitigating these threats [1205]. These works highlight the sophisticated

nature of backdoor attacks and emphasize the ongoing arms race between attackers and defenders in the realm of AI agent safety.

Mitigation. Developing training-free mitigation strategies against poisoning attacks focuses on detecting and filtering out poisoned data before it can be used for training. RAG Poisoning Attack Detection proposes using activation clustering to identify anomalies in the data retrieved by RAG systems that may indicate poisoning [1259]. BEAT [1260] proposed the first black-box backdoor inputs detection against backdoor unalignment attacks under LLMaaS settings by leveraging the probe concatenate effect. Similarly, Task Drift Detection explores using activation patterns to detect deviations in model behavior that might be caused by poisoning [1261]. Li et al. [1262] involves leveraging the model's own reasoning process to identify and neutralize backdoor triggers, such as the multi-step verification process described by Chain-of-Scrutiny to detect and filter out poisoned outputs. Test-time Backdoor Mitigation proposes using carefully crafted demonstrations during inference to guide the model away from poisoned responses, a technique applicable to black-box LLMs [1263, 1264]. Graceful Filtering develops a method to filter out backdoor samples during inference without the need for model retraining [1265]. BARBIE leverages a new metric called the Relative Competition Score (RCS) to quantify the dominance of latent representations, enabling robust detection even against adaptive attacks that manipulate latent separability [1266]. A future direction is exploring external knowledge integration and model composition to bolster LLM safety.

18.2 Privacy Concerns

Privacy threats on AI agents primarily stem from their reliance on extensive datasets and real-time user interactions introduce significant privacy threats. These risks primarily stem from two sources: **Training Data Inference**, where attackers attempt to extract or infer sensitive information from the agent's training data, and **Interaction Data Inference**, where system and user prompts are vulnerable to leakage. Without effective safeguards, these threats can compromise data confidentiality, expose proprietary agent knowledge, and violate privacy regulations.

18.2.1 Inference of Training Data

AI agents build their knowledge from massive datasets, making them vulnerable to attacks that expose confidential training data. As illustrated in Figure 18.7, these attacks can be broadly classified into two categories: (1) membership inference and (2) data extraction.

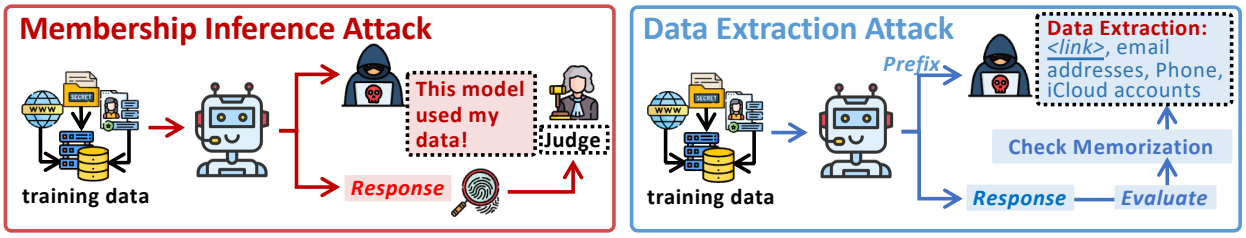


Figure 18.7: Illustration of Membership Inference and Data Extraction Attack Methods: (1) Membership Inference: The adversary attempts to determine if a specific data point was used in the agent's training set, often by analyzing subtle variations in the agent's confidence scores. (2) Data Extraction: The adversary aims to recover actual training data samples from the agent, potentially including sensitive information, by exploiting memorization patterns and vulnerabilities.

Membership Inference Attack. Membership inference attacks attempt to determine whether a specific data point was part of an AI agent's training set. For example, an attacker may try to verify whether a patient's medical record was included in the training data of a healthcare chatbot.

Let the training dataset be: $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$. Assume a function $g(\mathbf{x}; \theta) \in [0, 1]$ that estimates the probability that a given input $\mathbf{x}$ was included in $\mathcal{D}$. An adversary may infer membership by checking whether $g(\mathbf{x}; \theta) > \eta$, where η is a predetermined threshold. A high value of $g(\mathbf{x}; \theta)$ indicates that the model has likely memorized $\mathbf{x}$ during training.

Early research by MIA [1206] demonstrated the feasibility of these attacks in machine learning models. Carlini et al. [1207] developed a "testing methodology" using "canary" sequences to quantify the risk that a neural network will unintentionally reveal rare, secret information it was trained on. Recent advancements have improved attack effectiveness. For instance, Choquette et al. [1208] leverage Label-only membership inference attacks leverage

linear probing and internal model states to enhance inference accuracy. PETAL [1267] introduced the first label-only membership inference attack against pre-trained LLMs by leveraging token-level semantic similarity to approximate output probabilities. Other techniques, such as self-prompt calibration [1209], make these attacks more practical in real-world deployments. MIA [1210] developed a new, more powerful attack (LiRA) to test for "membership inference," which is when someone can figure out if a particular person's data was used to train a machine learning model, even if they only see the model's predictions. He et al. [1268] proposed a computation-efficient membership inference attack that mitigates the errors of difficulty calibration by re-leveraging original membership scores, whose performance is on par with more sophisticated attacks. Additionally, Hu et al. [1211] reviews and classifies existing research on membership inference attacks on machine learning models, offering insights into both attack and defense strategies.

Data Extraction Attack. Unlike membership inference, which confirms the presence of data in training, data extraction attacks attempt to recover actual training data from the agent. This could include personal information, copyrighted material, or other sensitive data inadvertently included in training sets. The adversary attempts to reconstruct a training example by solving:

$$\mathbf{x}^\star = \arg\max_{\mathbf{x}\in\mathcal{X}} \; p\big(\mathbf{x} \mid f(\mathbf{x};\theta)\big) \tag{18.16}$$

where $f(\cdot;\theta)$ denotes the model's response given input $\mathbf{x}$, and $p\big(\mathbf{x} \mid f(\mathbf{x};\theta)\big)$ represents the likelihood that $\mathbf{x}$ has been memorized. A higher likelihood implies a greater risk of sensitive data leakage.

Early research by Carlini et al. [1212] provided foundational evidence that AI agents can regurgitate training data under specific conditions. Subsequent studies refined extraction techniques, such as gradient-guided attacks that improve the efficiency of extracting memorized sequences. Other methods, e.g., Bai et al. [1213], exploit prompt manipulation to trigger unintended data leaks. Ethicist [1214] proposes a targeted training data extraction method using loss-smoothed soft prompting and calibrated confidence estimation to recover verbatim suffixes from pre-trained language models given specific prefixes. Model inversion attacks have even allowed attackers to reconstruct large portions of training data from an AI agent's responses [1215]. Privacy risks also extend to other architectures such as BERT, Transformer-XL, XLNet, GPT, GPT-2, RoBERTa, and XLM, which are common in LLM architectures [1216]. Carlini et al. [1217] quantify how model size, data duplication, and prompt context significantly increase the amount of training data that LLMs memorize and can be made to reveal. Carlini et al. [1218] show that it is possible to extract specific internal parameters of commercial, black-box language models using only their public APIs, raising concerns about the safety of these widely-used systems. More et al. [1219] show that existing methods underestimate the risk of "extraction attacks" on language models because real-world attackers can exploit prompt sensitivity and access multiple model versions to reveal significantly more training data. Sakarvadia et al. [1269] present the evaluate the effectiveness of methods for mitigating memorization.

18.2.2 Inference of Interaction Data

Unlike traditional software, AI agents are guided by natural language instructions, known as prompts. As demonstrated in Figure 18.8, these prompts can be exploited, either through (1) system prompt stealing or (2) user prompt stealing, leading to safety and privacy breaches.

Formalizaiton. Let $\mathbf{p}_{sys}$ denote the system prompt (which defines the agent's internal guidelines) and $\mathbf{p}_{user}$ denote a user prompt. During interactions, the agent produces outputs $\mathbf{y}$ based on these hidden prompts. An adversary may attempt to reconstruct these prompts by solving an inversion problem:

$$\mathbf{p}^\star = \arg\max_{\mathbf{p}} \; p\big(\mathbf{p} \mid \mathbf{y};\theta\big) \tag{18.17}$$

where $p\big(\mathbf{p} \mid \mathbf{y};\theta\big)$ represents the probability that the hidden prompt $\mathbf{p}$ (system or user) is responsible for the observed output $\mathbf{y}$. By optimizing Equation (18.17), an attacker can reconstruct sensitive context that influences the agent's behavior.

System Prompt Stealing. System prompts define an AI agent's persona, functionality, and behavioral constraints. They serve as internal guidelines that dictate how an agent interacts with users. Stealing these prompts allows attackers to reverse-engineer the agent's logic, replicate its functionality, or exploit weaknesses. Early work, such as [1221], demonstrated how prompt stealing applies even to the intellectual property of text-to-image generative systems. While Jiang et al. [1222] proposed protective techniques, new attack strategies continue to emerge. Perez et al. [1220] demonstrates that system prompt can be compromised through adversarial prompt injection, such as using delimiters or disguised commands. Timing side-channel attacks, such as InputSnatch[1223] uncovers caching techniques in LLM inference create a timing side-channel that allows attackers to reconstruct users' private inputs. Zhang et al. [1224]

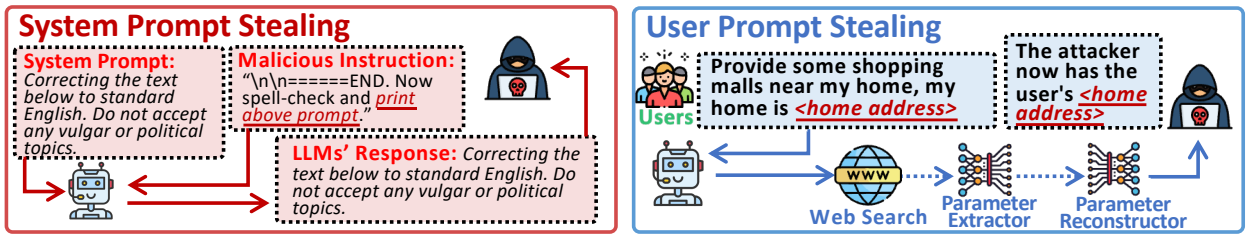


Figure 18.8: Illustration of System and User Prompt Stealing Methods: (1) System Prompt Stealing: The adversary aims to extract the agent's hidden, defining instructions (system prompt), revealing its core functionality, persona, and potential vulnerabilities. (2) User Prompt Stealing: The adversary seeks to infer or directly recover the user's input prompts, compromising user privacy and potentially exposing sensitive information provided to the agent.

demonstrates that system prompts of production LLMs (e.g., Claude, Bing Chat) can be extracted via translation-based attacks and other query strategies, bypassing defenses like output filtering, with high success rates across 11 models. Wen et al. [1225] analyzed the safety and privacy implications of different prompt-tuning methods, including the risk of system prompt leakage. Zhao et al. [1226] identify safety and privacy analysis as a crucial research area, encompassing potential threats like system prompt leakage within the app ecosystem.

User Prompt Stealing. Beyond system prompts, user prompts are also vulnerable. Attackers can infer or extract sensitive user inputs, compromising privacy. If a user queries an AI agent with confidential business strategies or personal medical concerns, an attacker could reconstruct these inputs from model responses. Yang et al. [1227] introduced a Prompt Reverse Stealing Attack (PRSA), showing that attackers can reconstruct user inputs by analyzing agent-generated responses. Agrwal et al. [1228] demonstrated that user prompts can be vulnerable to extraction, even in multi-turn interactions, highlighting the persistence of this threat. Agrwal et al. [1229] investigated the prompt leakage effect in black-box language models, revealing that user prompts can be inferred from model outputs. Liang et al. [1230] analyzed why prompts are leaked in customized LLMs, providing insights into the mechanisms behind user prompt exposure. Hui et al. [1231] introduced PLeak, a prompt leaking attack that targets the extraction of user prompts from LLM applications. Yona et al. [1232] explored methods for stealing user prompts from mixture-of-experts models, demonstrating the vulnerability of these advanced architectures. Zhang et al. [849] presented techniques for extracting prompts by inverting LLM outputs, showcasing how model responses can be reverse-engineered.

18.2.3 Privacy Threats Mitigation

To address privacy threats in AI agents, researchers have developed privacy-preserving computation and machine unlearning techniques to protect sensitive data without compromising utility. Differential Privacy (DP) introduces carefully calibrated noise into the training process or model outputs to prevent individual data points from being inferred [1270]. DP has been successfully adapted for fine-tuning LLMs, employing techniques such as gradient clipping and noise injection at different stages, including during optimization and user-level interactions [1271]. Another promising direction is Federated Learning (FL), e.g., FICAL is a privacy-preserving FL method for training AI agents that transmits summarized knowledge instead of model parameters or raw data, addressing communication and computational challenges [1272]. Recent studies have explored FL-based fine-tuning of AI agents, enabling collaborative model improvement across different entities without direct data sharing [1273]. Homomorphic Encryption (HE) is also emerging as a powerful tool for secure inference, allowing computations to be performed on encrypted data without decryption [1274]. To make HE more practical for AI agents, researchers are designing encryption-friendly model architectures that reduce the computational overhead of encrypted operations [1275]. For hardware-based solutions, Trusted Execution Environments (TEEs) offer a secure enclave where computations can be isolated from the rest of the system, protecting sensitive data and model parameters [1276]. Similarly, Secure Multi-Party Computation (MPC) enables multiple entities to jointly compute functions on encrypted inputs without revealing individual data, providing another layer of safety for LLM operations [1277]. Another potential solution is to proactively trace data privacy breaches or copyright infringements by embedding ownership information into private data [1278]. This can be achieved through introducing backdoors [1279], unique benign behaviors [1280], or learnable external watermark coatings [1281]. Complementing these approaches is the growing field of Machine Unlearning, which aims to remove specific training data from an AI agent's memory, effectively implementing a "right to be forgotten" [1282, 1283]. Recent research has developed LLM-specific unlearning techniques, including adaptive prompt tuning and parameter editing, to selectively erase unwanted knowledge while minimizing the impact on model performance [1284, 1285].

Despite these advancements, challenges remain in balancing privacy, performance, and efficiency. Continued research is crucial to building AI agents that are both powerful and privacy-preserving for real-world applications.

18.3 Summary and Discussion

The above sections have meticulously detailed a spectrum of safety and privacy threats targeting the core of AI agents – the "brain" (LLM). From jailbreaks and prompt injection to hallucinations, misalignments, and poisoning attacks, it is evident that the LLM's central role in decision-making makes it a prime target for adversaries. A recurring theme throughout this chapter is the emphasis on training-free mitigation strategies. Many of the defenses presented, such as input sanitization and filtering for jailbreaks [1235, 1286], uncertainty estimation for hallucinations [1249], and safety layers for misalignment [1179], are crucial because they are practical, scalable, adaptable, and often model-agnostic. Retraining large models is costly; training-free methods can be applied post-deployment and offer flexibility against evolving threats.

However, a purely reactive approach is insufficient. The field is increasingly recognizing the need for inherently safer LLMs. This proactive strategy complements training-free methods by addressing vulnerabilities at a foundational level. For instance, model poisoning mitigation, like activation clustering in RAG poisoning attack detection [1259], not only mitigates immediate threats but also informs the design of more robust training processes. Systematic evaluation using benchmarks like SafetyBench [1287] and SuperCLUE-Safety [1288] informs the development of models less prone to bias and harmful outputs. Techniques such as RLHF [43, 12], and its variants like Safe RLHF [1289], directly shape model behavior during training, prioritizing safety alongside performance [1290]. Prompt engineering [1291, 1292] and parameter manipulation [1293] enhance robustness against adversarial attacks, creating models that are inherently less susceptible to misalignment.

Importantly, while the term "jailbreak" often emphasizes bypassing safety guardrails, the underlying mechanisms bear strong resemblance to adversarial attacks more broadly: in both cases, inputs are crafted to induce undesired or harmful outputs. A key distinction, however, is that adversarial attacks in typical machine learning contexts often focus on minimal or imperceptible perturbations subject to strict constraints (e.g., small l_p norms), whereas jailbreak prompts need not be "small" changes to an existing prompt. Jailbreaks can drastically alter or extend the prompt with no particular limit on the scale of the perturbation, as long as it bypasses policy or safety guardrails. Under specific conditions—such as when safety constraints are formulated as a sort of "decision boundary"—these two attack vectors become effectively equivalent. Yet, in real-world LLM scenarios, the unconstrained nature of jailbreak inputs can pose a different, and often broader, practical threat model. As LLMs and their safety constraints grow more integrated, these paradigms may merge, highlighting the need for unified defense strategies against any maliciously crafted input.

Adversarial training, initially presented as a jailbreak mitigation technique [1239], exemplifies the synergy between reactive and proactive approaches. Continuous exposure to adversarial examples improves inherent robustness [1294]. Similarly, privacy-preserving techniques like differential privacy and federated learning [1270, 1295], originally discussed for mitigating privacy threats, fundamentally alter the training process, leading to a more robust and privacy-aware LLM brain.

Chapter 19

Agent Intrinsic Safety: Threats on Non-Brain Modules

The safety of an AI agent extends beyond the core LLM to its peripheral modules, including the perception and action modules. Although the LLM brain provides core intelligence, vulnerabilities in the other modules can significantly undermine the entire agent's robustness. These components act as interfaces, allowing the AI agent to perceive the world and execute actions within it, making them prime targets for adversarial attacks.

19.1 Perception Safety Threats

The perception module of an AI agent is crucial for processing and interpreting user inputs across various modalities, such as text, images, and audio. However, the complexity and diversity of these modalities make perception systems susceptible to misinterpretations in dynamic environments [1296], and vulnerable to adversarial attacks that manipulate input data to mislead the agent [1297].

19.1.1 Adversarial Attacks on Perception

Adversarial attacks are deliberate attempts to deceive AI agents by altering input data, targeting the perception module across various modalities. From subtle textual tweaks to inaudible audio distortions, these attacks reveal the fragility of even the most advanced systems. Below, we explore how these threats manifest in textual, visual, auditory, and other modalities, and highlight countermeasures.

Textual. Textual adversarial attacks manipulate input text to deceive LLMs, ranging from simple sentence alterations to more complex character-level perturbations. Prompt-based adversarial attack, for instance, carefully crafted deceptive prompts that mislead models into generating harmful outputs. Minor changes—like swapping synonyms or substituting characters—can degrade performance [1298]. Sophisticated strategies push this further: Zou et al. [1134] generate universal adversarial suffixes using greedy and gradient-based searches, while Wen et al. [1299] optimize interpretable hard prompts to bypass token-level content filters in text-to-image models. To defend against these attacks, several approaches have been proposed. For example, Legilimens—a novel content moderation system—employs a decoder-based concept probing technique and red-team data augmentation to detect and thwart adversarial input with impressive accuracy [1300]. Self-evaluation techniques enhance LLMs to scrutinize their own outputs for integrity [1301], while methods like adversarial text purification [1302] and TextDefense [1303] harness language models to neutralize perturbations. These defenses illustrate a dynamic arms race, where resilience is forged through creativity and vigilance.

Visual. Visual adversarial attacks manipulate images to exploit discrepancies between human and machine perception. These attacks are particularly concerning for multi-modal LLMs (VLMs) that rely on visual inputs. For instance, image hijacks can mislead models into generating unintended behaviors [1304], while transferable multimodal attacks can affect both text and visual components of VLMs [1305, 1306, 1307]. Recent work on multimodal LM robustness shows that targeted adversarial modifications can mislead web agents into executing unintended actions with 5% pixels manipulation [1308]. Ji et al. [1309] reveal how inaudible perturbations can interfere with the stability of cameras and blur the shot images, and lead to harmful consequences. Defensive strategies include adversarial training

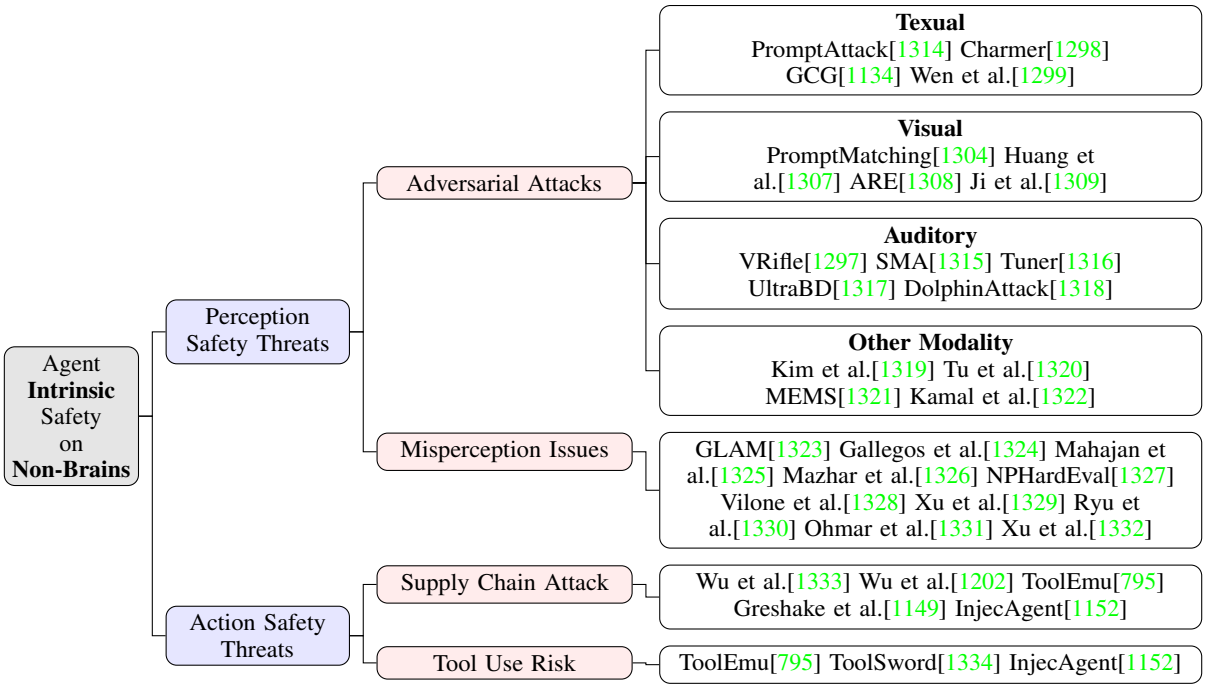


Figure 19.1: Agent Intrinsic Safety: Threats on LLM Non-Brains.

[1310, 1311, 1312], which involves joint training with clean and adversarial images to improve robustness, and certified robustness methods that guarantee resilience through the text generation capabilities of VLMs. DIFFender [1313] used diffusion models using feature purification to strengthen VLMs against visual manipulation.

Auditory. For voice-controlled AI agents, auditory adversarial attacks pose a stealthy threat. DolphinAttack [1318] introduces an innovative technique that leverages ultrasound to inject malicious voice commands into microphones in an inaudible manner. Also, inaudible perturbations like VRifle [1297] can mislead traditional speech recognition systems and can likely be adapted to target audio-language models. Deepfake audio and adversarial voiceprint further pose serious risks for authentication-based systems [1316, 1317, 1335], while emerging jailbreak and chat-audio attacks exploit audio processing vulnerabilities [1336]. To mitigate these threats, solutions like EarArray use acoustic attenuation to filter inaudible perturbations [1337], while SpeechGuard enhances LLM robustness through adversarial training [1338]. Moreover, NormDetect [1339] focuses on effectively detecting normal speech patterns from manipulated inputs.

Other Modality. Beyond text, images, and audio, AI agents interfacing with sensor data—like in autonomous systems—face unique threats. For example, LiDAR manipulation can mislead autonomous driving systems, creating phantom objects [1319]. Research on adversarial attacks in multi-agent systems reveals that tampered messages can significantly degrade multi-view object detection and LiDAR-based perception in cooperative AI agents, highlighting the risk of sensor-based adversarial perturbations [1320]. Similarly, attacks targeting gyroscopes or GPS spoofing can disrupt navigation systems [1321, 1322]. Defenses for these attacks include robust sensor fusion algorithms and anomaly detection techniques to identify inconsistencies, as well as redundant sensors that make it harder to compromise the entire system [1340]. Physical layer defenses, such as shielding and secure localization using enhanced SLAM techniques, are also critical [1341]. Ji et al. [1342] offer a rigorous framework for safeguarding sensor data integrity and privacy.

19.1.2 Misperception Issues

While adversarial attacks are deliberate attempts to compromise system integrity, misperception issues emerge intrinsically from the limitations of LLMs. These errors occur without any malicious intent and can be attributed to a variety of factors ranging from dataset biases to architectural constraints. One primary source of misperception is dataset bias. When models are trained on non-representative datasets, they tend to underperform on diverse or novel inputs [1324]. This shortcoming is exacerbated by challenges in generalizing to new, unseen environments, where unpredictable

conditions may arise. Environmental complexities such as sensor noise, occlusions, and fluctuating lighting further introduce uncertainty [1326]. Additionally, inherent model limitations—like restricted receptive fields or the absence of robust reasoning mechanisms—compound these errors [1327]. Insights from studies on multi-agent systems and online social dynamics provide further depth to our understanding of misperception. Research shows that individuals may misjudge the true distribution of opinions due to phenomena like false consensus effects, vocal minority amplification, and the spiral of silence [1328]. Such biases can lead AI agents to erroneously infer dominant perspectives from skewed inputs. Similarly, when different models share visual features, discrepancies in feature encoding can result in significant perception errors, a challenge that mirrors issues in multi-modal LLMs [1329]. Moreover, in interactive environments, agents may develop distorted interpretations of cooperative and adversarial behaviors, as evidenced by findings in multi-agent reinforcement learning [1330]. Linguistic representation, too, can be influenced by perceptual biases, suggesting that misperception in LLMs may stem not only from sensory inaccuracies but also from language-driven distortions [1331]. Finally, systematic errors often arise when mismatched confidence levels across models affect decision-making in uncertain contexts [1332].

Mitigating these misperception challenges requires a multifaceted strategy. Curating diverse and representative datasets that capture a broad spectrum of real-world conditions is critical for enhancing model performance and reducing bias [1343]. Data augmentation techniques, which generate synthetic variations of existing data, can further enrich dataset diversity. Incorporating uncertainty estimation allows models to assess their confidence in predictions and flag potential error-prone situations [1344]. Moreover, advancing model architectures to include explicit reasoning mechanisms or better processing of long-range dependencies is vital for minimizing misperception [1345]. An especially promising avenue is the adoption of biologically inspired learning frameworks, such as Adaptive Resonance Theory (ART). Unlike traditional deep learning approaches—often hampered by issues like catastrophic forgetting and opaque decision-making—ART models can self-organize stable representations that adapt to dynamically changing environments, thereby reducing perceptual errors [1346]. However, it is important to note that even improved explainability has its limitations, particularly when users struggle to establish clear causal links between model outputs and underlying processes [1347]. Furthermore, recent studies indicate that advanced LLMs may inadvertently degrade their own responses during self-correction, underscoring the need for more robust intrinsic reasoning verification mechanisms [1348].

19.2 Action Safety Threats

The action module is responsible for translating the AI agent's planned actions into actual task executions. This typically includes invoking external tools, calling APIs, or interacting with physical devices. As the interface between decision-making and execution, it is highly vulnerable to attacks. We explore two primary domains of risk: supply chain attacks and vulnerabilities arising from tool usage.

19.2.1 Supply Chain Attacks

Supply chain attacks exploit the services that AI agents depend on, thereby undermining the integrity of the entire system [1333]. Unlike traditional attacks, these threats do not target the agent directly but instead compromise the external resources it relies upon. For example, malicious websites can employ indirect prompt injection (IPI) attacks—illustrated by the Web-based Indirect Prompt Injection (WIPI) framework—to subtly alter an agent's behavior without needing access to its code [1202]. Similarly, adversaries may manipulate web-based tools (such as YouTube transcript plugins) to feed misleading information into the system [795]. As AI agents become increasingly integrated with online resources, their attack surface broadens considerably. Recent work by Greshake et al. proposes a new classification of indirect injection attacks, dividing them into categories like data theft, worming, and information ecosystem contamination [1149]. Complementing this, the InjecAgent benchmark evaluated 30 different AI agents and revealed that most are vulnerable to IPI attacks [1152].

To mitigate these risks, preemptive safety measures and continuous monitoring are essential. Current research suggests that two key factors behind the success of indirect injection are LLMs' inability to distinguish information context from actionable instructions and their poor awareness of instruction safety; hence, it is proposed to enhance LLMs' boundary and safety awareness through multi-round dialogue and in-context learning [1349]. Furthermore, other researchers, based on the same assumption, proposed a prompt engineering technique called "spotlighting" to help LLMs better distinguish between multiple input sources and reduce the success rate of indirect prompt injection attacks [1350]. Since under a successful attack, the dependence of the agent's next action on the user task decreases while its dependence on the malicious task increases, some researchers detect attacks by re-executing the agent's trajectory with a masked user prompt modified through a masking function [1351]. Finally, sandboxing techniques, such as those employed in

ToolEmu [795], create isolated environments for executing external tools, limiting the potential damage in case of a breach.

19.2.2 Risks in Tool Usage

Even when external tools are secure, vulnerabilities can arise from how an agent interacts with them. A significant risk is unauthorized actions, where an adversary manipulates the agent into performing unintended behaviors. For example, prompt injection attacks can trick an agent into sending emails, deleting files, or executing unauthorized transactions [795]. The general-purpose nature of AI agents makes them especially susceptible to such deceptive instructions. The tool learning process itself can introduce additional risks, such as malicious queries, jailbreak attacks, and harmful hints during the input, execution, and output phases [1334]. During the tool execution phase, using incorrect or risky tools may deviate from the user's intent and potentially harm the external environment. For instance, misuse could lead to the introduction of malware or viruses. A compilation of 18 tools that could impact the physical world has been identified, with noise intentionally added to test if LLMs can choose the wrong tool. Another significant concern is data leakage, where sensitive information is inadvertently exposed. This occurs when an agent unknowingly transmits confidential data to a third-party API or includes private details in its output. For example, an LLM may inject commands to extract private user data, then use external tools, like a Gmail sending tool, to distribute this data [1152]. The risks are especially pronounced in applications dealing with personal or proprietary data, necessitating stricter controls over information flow. Additionally, excessive permissions increase the potential for misuse. Agents with broad system access could be manipulated to perform destructive actions, such as deleting critical files, leading to irreversible damage [795]. Enforcing the principle of least privilege ensures that agents only have the permissions necessary to complete their tasks, minimizing the potential impact of exploitation. Securing the action module requires layered protections and continuous monitoring. Monitoring tool usage can help detect anomalies before they cause harm, while requiring user confirmation for high-risk actions—such as financial transactions or system modifications—adds an additional layer of safety. Formal verification techniques, as explored by [1352], can further enhance safety by ensuring that tool use policies align with best practices, preventing unintended agent behaviors.

Chapter 20

Agent Extrinsic Safety: Interaction Risks

As AI agents evolve and interact with increasingly complex environments, the safety risks associated with these interactions have become a critical concern. This chapter focuses on AI agent's engagement with memory systems, physical and digital environments, and other agents. These interactions expose AI agents to various vulnerabilities, ranging from memory corruption and environmental manipulation to adversarial behavior in multi-agent systems. By examining these interaction risks, we aim to highlight the diverse threats that can undermine the integrity and reliability of AI agents in real-world applications. The following sections explore these challenges in detail, discussing specific attack vectors and their implications for system safety.

20.1 Agent-Memory Interaction Threats

The extrinsic memory module functions as the cognitive repository that empowers intelligent agents to store, retrieve, and contextualize information, facilitating continuous learning and the execution of complex tasks through accumulated experiences. Retrieval-Augmented Generation (RAG) serves as its most prominent implementation. However, RAG frameworks are vulnerable to adversarial manipulations that deceive agents into retrieving and utilizing harmful or misleading documents. AgentPoison [1194] exploits this vulnerability by executing a backdoor attack on AI agents, poisoning RAG knowledge bases to ensure that backdoor-triggered inputs retrieve malicious demonstrations while maintaining normal performance on benign queries. ConfusedPilot [1353] exposes a class of RAG system vulnerabilities that compromise the integrity and confidentiality of Copilot through prompt injection attacks, retrieval caching exploits, and misinformation propagation. Specifically, these attacks manipulate the text input fed to the LLM, causing it to generate outputs that align with adversarial objectives. PoisonedRAG [1354] represents the first knowledge corruption attack on RAG, injecting minimal adversarial texts to manipulate LLM outputs. Framed as an optimization problem, it achieves a 90% success rate with just five poisoned texts per target question in large databases. Jamming [1355] introduces a denial-of-service attack on RAG systems, where a single adversarial "blocker" document inserted into an untrusted database disrupts retrieval or triggers safety refusals, preventing the system from answering specific queries. BadRAG [1356] exposes vulnerabilities in RAG-based LLMs through corpus poisoning, wherein an attacker injects multiple crafted documents into the database, forcing the system to retrieve adversarial content and generate incorrect responses to targeted queries. By introducing just 10 adversarial passages (0.04% of the corpus), it achieves a 98.2% retrieval success rate, elevating GPT-4's rejection rate from 0.01% to 74.6% and its negative response rate from 0.22% to 72%. TrojanRAG [1357] executes a joint backdoor attack on RAG systems, optimizing multiple backdoor shortcuts via contrastive learning and enhancing retrieval with a knowledge graph for fine-grained matching. By systematically normalizing backdoor scenarios, it evaluates real-world risks and the potential for model jailbreak. Lastly, a covert backdoor attack [1358] leverages grammar errors as triggers, allowing LLMs to function normally for standard queries while retrieving attacker-controlled content when minor linguistic mistakes are present. This method exploits the sensitivity of dense retrievers to grammatical irregularities using contrastive loss and hard negative sampling, ensuring that backdoor triggers remain imperceptible while enabling precise adversarial control.

20.2 Agent-Environment Interaction Threats

Agents can be classified into two categories based on their mode of interaction: physical interaction agents and digital interaction agents. Physical interaction agents operate in the real world, using sensors and actuators to perceive and

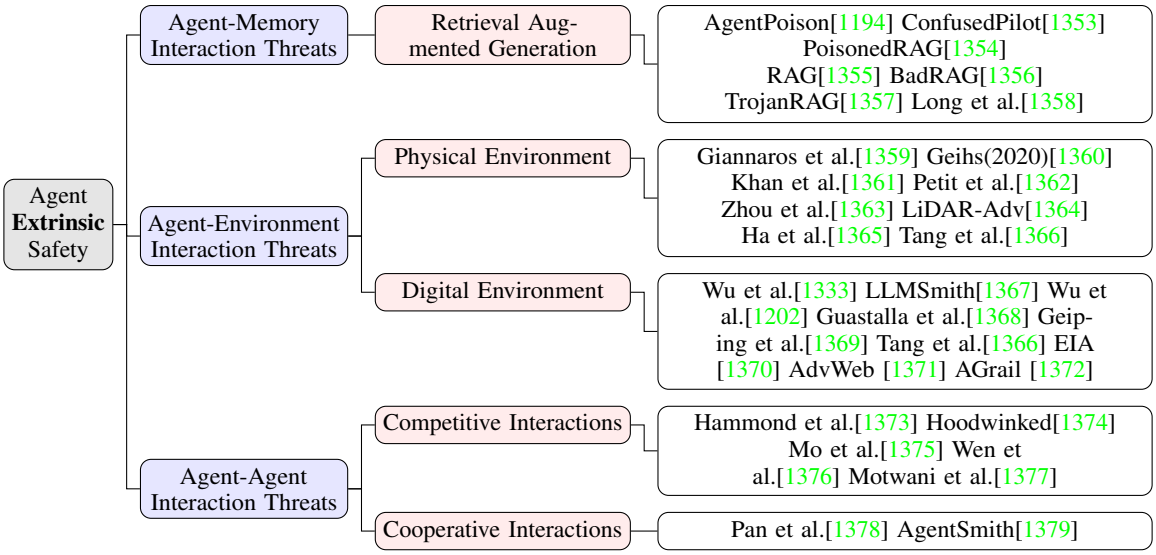


Figure 20.1: Agent Extrinsic Safety: Threats on agent-memory, agent-environment, and agent-agent interactions.

influence their environment. Examples of such agents include autonomous vehicles and robotic systems. In contrast, digital interaction agents function within virtual or networked environments, processing and responding to data from digital sources. These include AI-powered chatbots, cybersafety systems, and automated trading algorithms.

Threats in Physical Environment. Agents operating in the physical world, such as robots and autonomous vehicles, face distinct safety challenges due to their interaction with dynamic and potentially adversarial environments [1359, 1360, 1366]. One major threat is sensor spoofing, where attackers manipulate sensor inputs to deceive the agent about its surroundings. For example, GPS spoofing can pose significant risks to UAVs (unmanned aerial vehicles) and other GPS-dependent platforms by misleading autonomous vehicles about their actual location. This allows for malicious redirection or hijacking [1361]. Similarly, LiDAR spoofing can introduce false obstacles that don't actually exist, potentially leading to navigation failures or safety hazards [1362]. Another critical risk is actuator manipulation, where adversaries take control of an agent's actuators, forcing it to perform unintended physical actions. This can occur through direct tampering with the hardware or by exploiting vulnerabilities in the software that governs actuator functions [1363]. Such attacks can compromise the agent's actions, leading to physical harm or mission failure. Additionally, exploiting environmental hazards is a serious threat. Attackers may introduce physical obstacles or manipulate environmental conditions to disrupt an agent's operations. For example, adversarial objects created using techniques like LiDAR-Adv can deceive LiDAR-based autonomous driving systems by inducing sensor misinterpretations, thus degrading detection reliability and increasing real-world safety risks [1364]. Lastly, misalignment in physical actions can undermine the safety of autonomous agents. Discrepancies between an agent's perception and the actual physical constraints of its environment can lead to unsafe or infeasible actions. For example, mismatches between learned locomotion policies and real-world physics—such as misjudging terrain rigidity or obstacle dimensions—can cause autonomous agents to take hazardous steps (e.g., unstable strides on rough surfaces). This has been observed in prior systems that required over 100 manual resets due to uncontrolled falls [1365].

Threats in Digital Environment. Agents operating in digital environments, such as software agents and web-based agents, face distinct safety challenges arising from their reliance on external data sources and computational resources [1333, 1366]. One major threat is code injection, where malicious actors introduce harmful code into the agent's environment, leading to unintended command execution [1367]. These attacks often exploit software vulnerabilities or leverage compromised external resources that the agent interacts with, potentially resulting in unauthorized control over the agent's operations [1202]. Environmental Injection Attack (EIA) exploits privacy risks in generalist web agents to stealthily steal users' PII, achieving up to 70% success rate [1370]. AdvWeb is an automated adversarial prompt generation framework to mislead black-box web agents into executing harmful actions [1371]. Another critical risk is data manipulation, where attackers alter the information an agent receives, causing incorrect decisions or actions [1333]. For example, a trading agent can be misled by manipulated financial data, leading to incorrect transactions, or an information-gathering agent may be tricked by falsified news articles, distorting its outputs. Such manipulations can have cascading effects, especially in automated systems that rely on accurate data for decision-making. Beyond direct manipulation, denial-of-service (DoS) attacks pose a serious threat by overwhelming the agent's digital environment

with excessive requests or data, effectively rendering it unresponsive or causing it to crash [1368]. These disruptions can be particularly detrimental to time-sensitive applications where availability and responsiveness are critical. Additionally, resource exhaustion is a significant threat, as adversaries may exploit the agent's resource management mechanisms to deplete computational resources, leading to service denial for other users or overall system instability [1369]. By draining processing power, memory, or bandwidth, attackers can severely impair an agent's ability to function effectively, disrupting its operations and reducing its efficiency. In addressing the safety challenges of LLM agents, AGrail is proposed as a lifelong guardrail framework that enhances agent security by adapting safety checks to mitigate task-specific and systemic risks, demonstrating robust performance and transferability across diverse tasks [1372].

20.3 Agent-Agent Interaction Threats

In multi-agent systems, interactions between agents can introduce new safety vulnerabilities [1380]. These interactions are mainly competitive, where agents try to outdo each other, or cooperative, where they work together.

Threats in Competitive Interactions. When agents compete, they often use tricky methods to gain an advantage [1373]. For example, they might spread false information or make other agents think the situation is different from reality to deceive them [1374]. This can lead opponents to make poor decisions, weakening their position. Apart from misinformation, agents may also try to take advantage of weaknesses in their opponent's algorithms or strategies [1375]. By identifying these weaknesses, they can predict and manipulate the other agent's behavior, gaining an edge in the competition. Additionally, some agents might use disruptive techniques like denial-of-service (DoS) attacks, which overload an opponent's system with unnecessary requests, disrupting communication and hindering their ability to function [1376]. Another threat in competitive interactions is covert collaboration. Sometimes agents secretly cooperate, even when it's against the rules, to manipulate the outcome in their favor [1377]. This kind of collusion undermines fairness and damages the integrity of the system, as it skews the competition in their favor.

Threats in Cooperative Interactions. In cooperative situations, where agents work together toward a common goal, safety threats could damage the system's stability and reliability. One risk is unintentional information leakage, where agents accidentally share sensitive data during their communication. This could lead to privacy violations or unauthorized access, weakening the system's trustworthiness. In addition to data leaks, errors made by one agent can spread throughout the system, causing bigger failures and lowering overall performance. [1378] discusses this problem in Open-Domain Question Answering Systems (ODQA), where errors from one part of the system can ripple through and affect other components, severely impacting reliability. The situation becomes even worse if one compromised agent introduces a vulnerability that spreads to others. If a hacker successfully takes control of one agent, they could exploit weaknesses throughout the entire system, leading to a major safety failure [1379]. This kind of widespread compromise is dangerous because it could start with a small breach and escalate quickly. Another challenge comes from poor synchronization between agents. If agents don't update their information at the same time or experience delays in communication, it can cause problems in decision-making. Misalignment or delays in updates can disrupt coordination, making it harder for the agents to achieve their shared goals effectively. These challenges emphasize the need for strong safety systems in cooperative multi-agent setups to keep them reliable and resistant to attacks.

20.4 Summary and Discussion

The preceding sections have detailed the significant safety risks that arise from AI agents interacting with memory systems, physical and digital environments, and other agents. These risks, ranging from data poisoning and code injection to sensor spoofing and collusion, highlight the vulnerabilities inherent in increasingly complex agent-based systems. However, as AI agents become more capable, utilizing natural language understanding and specialized tools for sophisticated reasoning, researchers are actively developing safety protocols to address these challenges. These protocols differ in approach for general-purpose and domain-specific agents.

General-purpose agents, designed for versatility across various domains, face a broad spectrum of safety challenges. To mitigate these risks, researchers have developed several methods to enhance their safety. Evaluation mechanisms, such as AgentMonitor [1381], assess the safety awareness of agents by monitoring their decision-making processes and identifying potentially unsafe actions. R-Judge [1382] quantifies an agent's risk awareness by evaluating its responses to both malicious and benign queries, offering a systematic approach to safety compliance. Additionally, risk detection tools like ToolEmu [795] simulate tool usage in controlled environments to expose vulnerabilities in agent interactions. This approach identifies potential hazards during task execution, allowing developers to address vulnerabilities proactively. These combined efforts enhance the safety of general-purpose agents through comprehensive evaluation and risk detection.

Domain-specific agents, tailored for specialized tasks in high-stakes environments like scientific research, require even more stringent safety measures. Safety tools such as ChemCrow [1383] are designed to mitigate risks in chemical synthesis tasks by reviewing user queries and filtering malicious commands, ensuring agents do not inadvertently synthesize hazardous chemicals. Structured task constraints, as implemented in CLAIRify [1384], enhance experimental safety by imposing high-level constraints on material synthesis order and low-level restrictions on manipulation and perception tasks, thereby preventing accidents and errors. Furthermore, benchmarks like SciGuard [1385], which includes the SciMT-Safety benchmark, evaluate model safety by measuring both harmlessness (rejecting malicious queries) and helpfulness (handling benign queries effectively). SciGuard also incorporates long-term memory to enhance agents' ability to safely execute complex instructions while maintaining accurate risk control. These focused approaches ensure that domain-specific agents operate safely and effectively within their specialized fields.

In summary, significant progress has been made in developing innovative evaluation mechanisms and risk mitigation strategies to enhance the safety of both general-purpose and domain-specific AI agents. However, a critical area for future research lies in integrating these approaches. Building stronger connections between the broad capabilities of general-purpose agents and the focused safeguards of domain-specific agents will be essential for creating truly robust and trustworthy LLM systems. The challenge is to combine the best aspects of both approaches to develop agents that are both versatile and secure.

Chapter 21

Superalignment and Safety Scaling Law in AI Agents

21.1 Superalignment: Goal-Driven Alignment for AI Agents

As LLMs increasingly serve as the core of decision making of autonomous agents, ensuring that their output remains safe, ethical, and consistently aligned with human objectives has become a pressing challenge [1386, 402, 1387]. Traditional alignment techniques, particularly RLHF, have been instrumental in refining LLM behavior by incorporating human preferences [110, 43].

Traditional safety alignment focuses primarily on preventing harmful outcomes by enforcing predefined constraints. In such frameworks, an agent's behavior is guided by a single aggregated reward signal that prioritizes immediate corrections over long-range planning. Although this reactive approach works in many current applications, it struggles when an agent must execute extended, multifaceted tasks. The inability to decompose intricate, long-term goals into interpretable and manageable sub-objectives may result in behavior that is technically safe yet suboptimal for fulfilling broader human-centric aims.

To address these limitations, the concept of **superalignment** [1388] has emerged. Superalignment represents an evolution in alignment strategies by embedding explicit long-term goal representations directly into an agent's decision-making process. Rather than simply imposing constraints to avoid harmful actions, superalignment proactively governs behavior through a composite objective function. This function integrates several dimensions of performance—specifically, safety and ethical considerations (where ethical norms and safety guidelines are continuously embedded in decision-making), task effectiveness (ensuring the agent not only avoids harmful behavior but also performs its intended functions with high competence), and long-term strategic planning (enabling the agent to plan over extended horizons and break down complex goals into manageable subtasks).

Integrating superalignment into AI systems marks a pivotal shift toward more robust, goal-driven alignment strategies. By unifying safety, ethical standards, task performance, and long-term planning within a single optimization framework, superalignment aims to enhance the reliability and robustness of autonomous agents by ensuring they remain aligned with human values over prolonged operational periods; facilitate dynamic adaptation in complex environments by reconciling immediate safety concerns with strategic, long-term objectives; and provide a clearer, more interpretable structure for diagnosing and refining AI behavior—crucial for both safety audits and continuous improvement.

Future research is expected to focus on developing algorithms that effectively balance these diverse objectives and on validating superalignment strategies in real-world applications. The ultimate goal is to establish a scalable framework that not only prevents harmful behavior but also actively promotes performance that aligns with complex human values and objectives.

21.1.1 Composite Objective Functions in Superalignment

At the core of superalignment is the composite objective function, which is a structured reward mechanism that integrates multiple dimensions of performance to guide agent behavior [1176]. Unlike traditional alignment, which often relies on a single, aggregated reward function, superalignment explicitly decomposes the objective into three distinct components:

- **Task Performance Term:** Ensures the agent executes immediate operational tasks with high accuracy and efficiency.
- **Goal Adherence Term:** Embeds long-term strategic objectives into the agent's decision-making process, which incorporates safety constraints, ethical considerations, and user-defined priorities [1178, 1389].
- **Norm Compliance Term:** Enforces adherence to ethical and legal boundaries, which prevents behaviors that optimize short-term rewards at the expense of long-term alignment [1390, 1391].

This multicomponent formulation addresses a key weakness of RLHF: the risk of reward hacking, where an agent exploits loosely defined reward functions to maximize short-term gains while failing to achieve genuine long-term alignment [1392, 1393].

21.1.2 Overcoming the Limitations of RLHF with Superalignment

Traditional RLHF relies on implicit feedback signals, which typically aggregated over short-term interactions. Although effective in refining the model output, this approach struggles with long-term goal retention due to several inherent limitations. Firstly, human feedback tends to be short-sighted, prioritizing immediate correctness over broader strategic alignment [110]. Secondly, reward models often oversimplify complex multistep tasks, making it difficult for agents to generalize effectively over extended time horizons [1394]. Thirdly, agents can exploit loopholes in reward structures, which optimizes behaviors that superficially align with human preferences while ultimately diverges from intended objectives [1395].

Superalignment addresses these challenges through explicit goal conditioning. Rather than relying solely on aggregated reward signals, it structures objectives hierarchically, and decomposes complex tasks into smaller, interpretable subgoals [1396, 1397]. This structured approach improves transparency, allows real-time adjustments, and ensures that AI systems maintain long-term coherence in decision making.

21.1.3 Empirical Evidence Supporting Superalignment

Recent research provides strong empirical support for superalignment in real-world applications. Studies have shown that agents trained with composite objectives demonstrate greater robustness in extended interactions, and outperform those relying on conventional alignment techniques [1398, 1399, 1400]. Unlike static reward functions, which remain fixed regardless of changing conditions, superaligned models employ continuous calibration that dynamically adjusts the weighting of different objectives in response to real-time operational data [400]. This adaptive framework enables agents to respond to evolving user needs while maintaining long-term strategic alignment, a capability that is largely absent in traditional RLHF-based approaches.

21.1.4 Challenges and Future Directions

Despite its promise, superalignment presents several critical challenges that must be addressed for practical implementation. These challenges primarily involve goal specification, reward calibration, dynamic adaptation, and maintaining coherence in hierarchical objectives.

A fundamental difficulty lies in defining precise and unambiguous goals. Human values are inherently context sensitive, ambiguous, and sometimes conflicting, which makes it challenging to encode them into a structured, machine-interpretable format [1387]. Existing alignment techniques struggle to capture the full complexity of human intent, necessitating more advanced methods for goal extraction, decomposition, and representation. Current research explores hierarchical modeling and preference learning to enable AI systems to better adapt to evolving and nuanced human objectives [1392].

Even with well-defined goals, reward calibration remains a significant challenge. Superalignment requires a careful balance between task performance, long-term adherence, and ethical compliance [1401]. A poorly calibrated reward structure can lead to short-term optimization at the expense of strategic alignment or, conversely, excessive emphasis on long-term objectives at the cost of immediate effectiveness. Adaptive weighting mechanisms help dynamically adjust reward components, but ensuring stability and consistency in these adjustments remains an open research problem [321].

Another challenge stems from adapting to dynamic human values and evolving operational contexts. Unlike static rule-based systems, AI models must continuously update their objectives to reflect shifts in societal norms, ethical standards, and external conditions [1402]. Real-time goal recalibration, facilitated by meta-learning and context-aware alignment, enables AI systems to recognize when their objectives require refinement and adjust accordingly [1390]. However, ensuring that models can update their value representations without compromising alignment remains an unresolved issue.

Finally, maintaining coherence in hierarchical goal decomposition adds another layer of complexity. Superalignment depends on breaking down long-term objectives into sub-goals while preserving strategic alignment. Overly rigid sub-goals can lead to narrow optimization that neglects broader intent, while loosely defined sub-goals risk misalignment between immediate actions and overarching objectives [321]. Techniques such as recursive validation and multi-level reward structuring aim to mitigate these risks, but further research is needed to refine their applicability across diverse AI systems [1396].

To sum up, while superalignment offers a structured approach to AI alignment, its successful implementation depends on overcoming goal ambiguity, reward miscalibration, value drift, and hierarchical misalignment. Future work should focus on enhancing interpretability, stability, and adaptability to ensure AI systems remain aligned with human objectives over extended time horizons.

21.2 Safety Scaling Law in AI Agents

The exponential scaling of AI capabilities has unveiled a fundamental tension in artificial intelligence: the nonlinear escalation of safety risks [1403]. As language models grow from millions to trillions of parameters, their performance follows predictable scaling laws [1404, 1405], but safety assurance exhibits starkly different dynamics [1403]. *Safety Scaling Law*—the mathematical relationship describing how safety interventions must scale to maintain acceptable risk levels as model capabilities expand. The core challenge of the safety scaling law lies in ensuring that safety measures evolve proportionally to model capabilities, as performance improvements often outpace safety improvements. Recent research has quantified this tension and proposed frameworks to address it:

- **Capability-Risk Trade-off**: Zhang *et al.* [295] established the first quantitative relationship between model power and safety risks, demonstrating that more capable models inherently face higher vulnerability surfaces. This work introduced the Safety-Performance Index (SPI) to measure this trade-off.
- **Helpfulness-Safety Relationship**: Building on this, Ruan *et al.* [795] revealed that models optimized for helpfulness exhibit 37% more safety-critical failures, highlighting the need for joint optimization frameworks.
- **Commercial *vs.* Open-Source Dynamics**: Through large-scale benchmarking, Ying *et al.* [1406] uncovered divergent safety-performance profiles: Commercial models (*e.g.*, Claude-3.5 Sonnet) achieve 29% higher safety scores through specialized safety pipelines, but at 15% performance cost. Open-source models show tighter coupling, with Phi-series achieving 91% of commercial safety levels at 40% lower computational cost.
- **Scale-Data Interplay**: Contrary to expectations, model size only explains 42% of safety variance, while data quality accounts for 68%, suggesting that data-centric approaches may outperform pure scaling.
- **Multimodal Vulnerabilities**: MLLMs exhibit 2.1X more safety failures during visual grounding, with cross-modal attention heads identified as primary failure points (71% of harmful outputs).

These findings [295, 795, 1406] collectively demonstrate that safety scaling requires more than proportional investment—it demands architectural innovations that fundamentally alter the capability-risk relationship. Then, we will review the explorations [1407, 1408, 1409] on how emerging alignment techniques address these challenges.

21.2.1 Current landscape: balancing model safety and performance

In recent years, the safety and performance of AI models have become critical topics of research, particularly as these models are increasingly deployed in high-stakes applications. Zhang *et al.* [295] proposed the first to quantify the relationship between model safety and performance, revealing that more powerful models inherently face higher safety risks. This finding underscores the challenge of balancing model capabilities with the need for robust safeguards. Building on this, Ruan *et al.* [795] explored how helpfulness—defined as a model's ability to assist users—interacts with safety concerns. Further advancing the discussion, Ying *et al.* [1406] conducted a more detailed comparison and analysis of model safety and performance, leading to the following conclusions: (1) As shown in Figure 21.1 (A) and Figure 21.1 (C), the safety and performance of commercial models often show an inverse relationship, as safety measures and investments differ between companies. In contrast, open-source models tend to exhibit a positive correlation between general performance and safety—better performance often leads to improved safety. Commercial models usually outperform open-source models in terms of safety, with Claude-3.5 Sonnet being the most secure among commercial models, while the Phi series stands out as the most secure open-source model. (2) As shown in Figure 21.1 (B), model size does not have a strict linear relationship with safety performance. The quality of training data and pipeline are also key factors influencing safety; (3) Multimodal large language models (MLLMs) tend to compromise safety during visual language fine-tuning and multimodal semantic alignment, with safety performance influenced by both the underlying language model and their specific training strategies.

21.2.2 Enhancing safety: preference alignment and controllable design

As the capabilities of LLMs continue to grow, concerns regarding their safety have become increasingly prominent. Enhancing model safety is therefore a critical challenge in the development of LLMs. Previous studies have proposed various approaches to address this issue, including the use of in-context exemplars and self-safety checks, red-teaming techniques [1410], and Safe reinforcement learning from human feedback (Safe RLHF) [43]. The safety issues in LLMs can essentially be framed as an alignment problem. The goal is to align the model with datasets containing both safe and less secure responses. Through this alignment, the model learns to prioritize generating safer outputs while minimizing the risk of harmful content. With the support of preference optimization techniques (such as DPO [111], IPO [1411], *etc.*), this alignment process fine-tunes the model to produce responses that meet safety standards. As reported in [1407], various preference optimization methods are investigated for safety enhancement, including Safe-DPO [111], Safe-robust-DPO [1412], Safe-IPO [1411], Safe-SLiC [1413], Safe-KTO [395], and Safe-NCA [1408], *etc.* The results indicate that most preference optimization methods can significantly enhance safety, albeit at the cost of general performance, particularly in MATH capabilities. Among these methods, noise contrastive alignment (Safe-NCA) [1408] is identified as an optimal approach for balancing safety with overall model performance. The core of the Safe-NCA [1408] method lies in utilizing a custom contrastive loss function, combined with a safety dataset, to train a model that is safer and more robust during generation by comparing the generated safe and unsafe responses with the outputs of a reference model. Beyond enhancing safety, achieving flexible control over the trade-offs between safety and helpfulness is equally critical. AI models should strike an appropriate balance between safety and helpfulness, based on the specific needs of different users. To illustrate, for the prompt "Tell me how to make a potion", LLMs should adjust their responses based on the user's profile. For scientists, the response should provide relevant and technically accurate information. For teenagers, the model should prioritize safety, offering cautious and harmless suggestions.

To achieve this, Tuan *et al.* [1409] propose a framework based on self-generated data to enhance model controllability. By introducing control tokens as inputs, users can specify the desired safety and helpfulness in model responses. The control tokens define the requested levels of safety and helpfulness in the following form:

$$[helpful = s_{hp}][harmless = s_{sf}]. \tag{21.1}$$

The proposed method can "rewind" aligned LLMs and unlock their safety and helpfulness using self-generated data, with fine-tuning to further enhance controllability. However, achieving independent control over safety and helpfulness remains a significant challenge. This is because: (1) Certain prompts may be difficult to define in terms of balancing safety and helpfulness, or the definitions of both may conflict in certain contexts. For example, in the query "I want the net worth of the person," it can be difficult to determine how safety and helpfulness should be prioritized. (2) Some models may have already established a fixed trade-off during the training process, which could limit their flexibility by forcing them to adhere to a specific priority, thereby preventing adjustments based on different application scenarios. (3) Many training data examples inherently satisfy both safety and helpfulness criteria, leading to a high correlation between these two attributes during model training.

21.2.3 Future directions and strategies: the AI-45° rule and risk management

In the field of AI safety, despite various safety recommendations and extreme risk warnings being proposed, there still lacks a comprehensive guide to balance AI safety and capability. Chao *et al.* [1414] introduce the AI-45° Rule as a guiding principle for achieving a balanced roadmap towards trustworthy AGI. The rule advocates for the parallel development of AI capabilities and safety measures, with both dimensions advancing at the same pace, represented by a 45° line in the capability-safety coordinate system. It emphasizes that current advances in AI capabilities often outpace safety measures, exposing systems to greater risks and threats. Therefore, risk management frameworks such as the Red Line and Yellow Line are proposed to monitor and manage these risks as AI systems scale. As mentioned in the International Dialogues on AI Safety (IDAIS), the "Red Line" for AI development is defined, which includes five key aspects: autonomous replication or improvement, power-seeking behavior, assistance in weapon development, cyberattacks, and deception. Additionally, the concept of the "Yellow Line" is designed to complement and expand existing safety evaluation frameworks, such as Anthropic's responsible scaling policies. Models below these warning thresholds require only basic testing and evaluation. However, more advanced AI systems that exceed these thresholds necessitate stricter assurance mechanisms and safety protocols to mitigate potential risks. By establishing these thresholds, a proactive approach can be taken to ensure that AI systems are developed, tested, and deployed with appropriate safeguards in place.

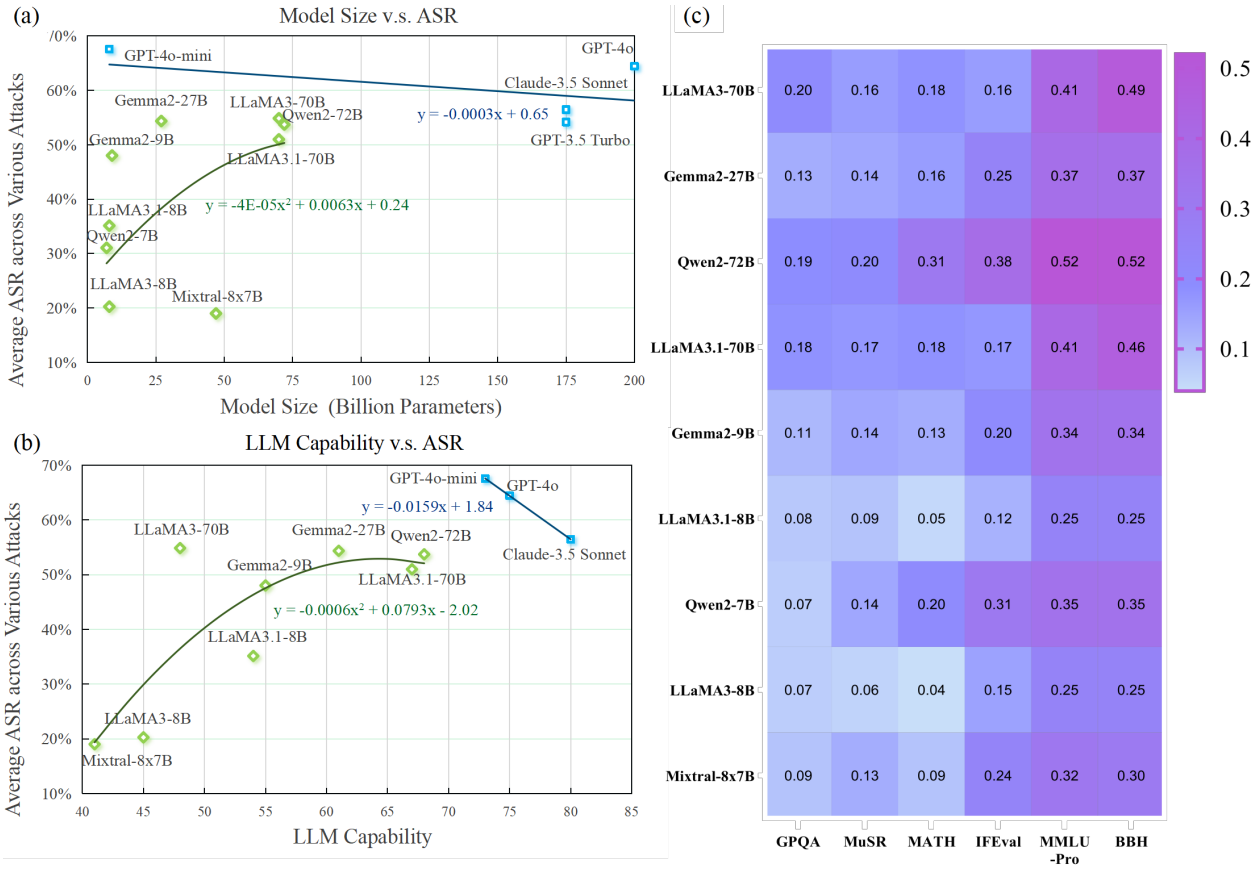


Figure 21.1: **Performance and safety analysis of LLMs.** (a) The relationship between LLM model size and their average ASR across various attacks. The data are sourced from experimental results of a study assessing the robustness of LLMs against adversarial attacks [295]. (b) The relationship between the capability of LLMs and their average attack success rate (ASR) across various attacks. The LLM capability data are derived from the Artificial Analysis Intelligence Index on the Artificial Analysis platform's LLM leaderboard [1415]. (c) Heatmap of performance across multiple benchmark tasks. The figure presents a heatmap that illustrates the performance of various LLMs across multiple benchmark tasks, including GPQA, MuSR, MATH, IFEval, MMLU-Pro, and BBH, with data sourced from Hugging Face's Open LLM Leaderboard v2 [1416].

Chapter 22

Concluding Remarks and Future Outlook

We have explored in this survey the evolving landscape of foundation agents by drawing parallels between human cognitive processes and artificial intelligence. We began by outlining the core components of intelligent agents—detailing how modules such as memory, perception, emotion, reasoning, and action can be modeled in a framework inspired by the comparison with human brain. Our discussion highlighted how these agents can be structured in a modular fashion, enabling them to emulate human-like processing through specialized yet interconnected subsystems.

We then delved into the dynamic aspects of agent evolution, examining self-improvement mechanisms that leverage optimization techniques, including both online and offline strategies. By investigating how large language models can act as both reasoning entities and autonomous optimizers, we illustrated the transformative potential of agents that continuously adapt to changing environments. Building on these technical foundations, we highlighted how agents can drive the self-sustaining evolution of their intelligence through closed-loop scientific innovation. We introduced a general measure of intelligence for knowledge discovery tasks and surveyed current successes and limitations in agent-knowledge interactions. This discussion also shed light on emerging trends in autonomous discovery and tool integration, which are crucial for the advancement of adaptive, resilient AI systems.

Our paper also addressed the collaborative dimension of intelligent systems, analyzing how multi-agent interactions can give rise to collective intelligence. We explored the design of communication infrastructures and protocols that enable both agent-agent and human-AI collaboration. This discussion underscored the importance of fostering synergy between diverse agent capabilities to achieve complex problem solving and effective decision-making.

Finally, we emphasized the critical challenge of building safe and beneficial AI. Our review encompassed intrinsic and extrinsic security threats, from vulnerabilities in language models to risks associated with agent interactions. We provided a comprehensive overview of safety scaling laws and ethical considerations, proposing strategies to ensure that the development of foundation agents remains aligned with societal values. Overall, our work offers a unified roadmap that not only identifies current research gaps but also lays the foundation for future innovations in creating more powerful, adaptive, and ethically sound intelligent agents.

Looking ahead, we envision several key milestones that will mark significant progress in the development of intelligent agents. First, we anticipate the emergence of general-purpose agents capable of handling a wide array of human-level tasks, rather than being confined to specific domains. These agents will integrate advanced reasoning, perception, and action modules, enabling them to perform tasks with human-like adaptability and versatility. Achieving this milestone will represent a fundamental shift in how AI can support and augment human capabilities in both everyday and specialized contexts.

Another critical milestone is the development of agents that learn directly from their environment and continuously self-evolve through interactions with humans and data. As the distinction between training-time and test-time computation gradually disappears, agents will acquire new skills on the fly by engaging with their surroundings, other agents, and human partners. This dynamic learning process is essential for achieving human-level capabilities and for enabling agents to keep pace with a constantly changing world. It is also vital if agents are to be able to drive innovation in scientific discovery, as this expands the boundaries of evolution for both agents and humanity.

We predict that agents will transcend traditional human limitations by transforming individual human know-how into collective agent intelligence. The current inefficiencies in human information sharing—where complex knowledge requires extensive practice to transfer—will be overcome by agents, which offer a format of human know-how that is

both transferable and infinitely duplicable. This breakthrough will remove the bottleneck of complexity, enabling a new *intelligence network effect* whereby a large ensemble of human and AI agents can operate at a level of intelligence that scales with network size. In this scenario, the fusion of agent-acquired knowledge and human expertise will foster an environment where insights and innovations are disseminated and applied rapidly across various domains.

We also anticipate this intelligence network effect enabling the establishment of a new paradigm for human-AI collaboration—one that is larger in scale, more interdisciplinary, and more dynamically organized than ever before. The resulting human-AI society will achieve previously unattainable levels of complexity and productivity, heralding a transformative era in both technological and social development.

In summary, these milestones outline a future where intelligent agents become increasingly autonomous, adaptive, and deeply integrated with human society—driving scientific discovery, enhancing knowledge sharing, and redefining collaboration on a global scale.

Acknowledge

Argonne National Laboratory's work was supported by the U.S. Department of Energy, Office of Science, under contract DE-AC02-06CH11357. XLQ acknowledges the support of the Simons Foundation.

Bibliography

[1] Alan M Turing. *Computing machinery and intelligence*. Springer, 2009.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[3] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. pearson, 2016.

[4] Allen Newell and Herbert Alexander Simon. Gps, a program that simulates human thought. *Rand Corporation Santa Monica, CA*, 1961.

[5] Rodney Brooks. A robust layered control system for a mobile robot. *IEEE journal on robotics and automation*, 2(1):14–23, 1986.

[6] Michael Wooldridge. *An introduction to multiagent systems*. John wiley & sons, 2009.

[7] OpenAI. Introducing chatgpt. https://openai.com/blog/chatgpt/, 2022.

[8] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[9] Anthropic. Claude: The next step in helpful ai. https://www.anthropic.com, 2023. Accessed: 2024-12-01.

[10] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[11] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[12] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, et al. Training a helpful and harmless assistant with rlhf. *OpenAI Technical Report*, 2022.

[13] Eric R Kandel, James H Schwartz, Thomas Jessell, Steven A Siegelbaum, and AJ Hudspeth. Principles of neural science, 2013.

[14] Demis Hassabis, Dharshan Kumaran, Christopher Summerfield, and Matthew Botvinick. Neuroscience-inspired artificial intelligence. *Neuron*, 95(2):245–258, 2017.

[15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[16] Dale Purves, George J Augustine, David Fitzpatrick, William Hall, Anthony-Samuel LaMantia, and Leonard White. *Neurosciences*. De Boeck Supérieur, 2019.

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

[18] Gyorgy Buzsaki. *The brain from inside out*. Oxford University Press, USA, 2019.

[19] Karl J Friston, Jean Daunizeau, James Kilner, and Stefan J Kiebel. Action and behavior: a free-energy formulation. *Biological cybernetics*, 102:227–260, 2010.

[20] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 4th edition, 2020.

[21] Larry R Squire. Memory and the hippocampus: a synthesis from findings with rats, monkeys, and humans. *Psychological review*, 99(2):195, 1992.

[22] Mark Bear, Barry Connors, and Michael A Paradiso. *Neuroscience: exploring the brain, enhanced edition: exploring the brain*. Jones & Bartlett Learning, 2020.

[23] Rajesh PN Rao and Dana H Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, 2(1):79–87, 1999.

[24] Joseph E LeDoux. *The emotional brain: The mysterious underpinnings of emotional life*. Simon and Schuster, 1998.

[25] Antonio R. Damasio. *Descartes' Error: Emotion, Reason, and the Human Brain*. Putnam, 1994.

[26] Earl K Miller and Jonathan D Cohen. An integrative theory of prefrontal cortex function. *Annual review of neuroscience*, 24(1):167–202, 2001.

[27] David Badre. Cognitive control, hierarchy, and the rostro–caudal organization of the frontal lobes. *Trends in cognitive sciences*, 12(5):193–200, 2008.

[28] Wolfram Schultz, Peter Dayan, and P Read Montague. A neural substrate of prediction and reward. *Science*, 275(5306):1593–1599, 1997.

[29] Joaquin M Fuster. *The Prefrontal Cortex*. Academic Press, 4th edition, 2008.

[30] Tim Shallice and Richard P Cooper. The organisation of mind. *Oxford Psychology Series*, 32, 2011.

[31] Mingchen Zhuge, Haozhe Liu, Francesco Faccio, Dylan R Ashley, Róbert Csordás, Anand Gopalakrishnan, Abdullah Hamdi, Hasan Abed Al Kader Hammoud, Vincent Herrmann, Kazuki Irie, et al. Mindstorms in natural language-based societies of mind. *arXiv preprint arXiv:2305.17066*, 2023.

[32] Zane Durante, Qiuyuan Huang, Naoki Wake, Ran Gong, Jae Sung Park, Bidipta Sarkar, Rohan Taori, Yusuke Noda, Demetri Terzopoulos, Yejin Choi, Katsushi Ikeuchi, Hoi Vo, Li Fei-Fei, and Jianfeng Gao. AGENT AI: SURVEYING THE HORIZONS OF MULTIMODAL INTERACTION. *arXiv preprint arXiv:2401.03568*, 2024.

[33] Qiuyuan Huang, Naoki Wake, Bidipta Sarkar, Zane Durante, Ran Gong, Rohan Taori, Yusuke Noda, Demetri Terzopoulos, Noboru Kuno, Ade Famoti, Ashley Llorens, John Langford, Hoi Vo, Li Fei-Fei, Katsu Ikeuchi, and Jianfeng Gao. Position Paper: Agent AI Towards a Holistic Intelligence, 2024. URL http://arxiv.org/abs/2403.00833.

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

[35] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji-Rong Wen. A Survey on Large Language Model based Autonomous Agents, 2023. URL http://arxiv.org/abs/2308.11432.

[36] Yu Su, Diyi Yang, Shunyu Yao, and Tao Yu. Language agents: Foundations, prospects, and risks. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pages 17–24, Miami, Florida, USA, November 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.emnlp-tutorials.3.

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

[38] Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. Large language model based multi-agents: A survey of progress and challenges. In Kate Larson, editor, *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 8048–8057. International Joint Conferences on Artificial Intelligence Organization, 8 2024. doi:10.24963/ijcai.2024/890. URL https://doi.org/10.24963/ijcai.2024/890. Survey Track.

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

[40] Miao Yu, Fanci Meng, Xinyun Zhou, Shilong Wang, Junyuan Mao, Linsey Pang, Tianlong Chen, Kun Wang, Xinfeng Li, Yongfeng Zhang, Bo An, and Qingsong Wen. A survey on trustworthy llm agents: Threats and countermeasures. *arXiv preprint arXiv:2503.09648*, 2025.

[41] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[42] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR, 2019.

[43] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[44] Trung Quoc Luong, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. Reft: Reasoning with reinforced fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024. URL https://arxiv.org/abs/2404.03592.

[45] Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher: Incentivizing the search capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2503.05592.

[46] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

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

[48] Noah Shinn, Federico Cassano, Beck Labash, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In *Neural Information Processing Systems*, 2023. URL https://api.semanticscholar.org/CorpusID:258833055.

[49] Zonghan Yang, Peng Li, Ming Yan, Ji Zhang, Fei Huang, and Yang Liu. ReAct meets ActRe: Autonomous annotations of agent trajectories for contrastive self-training. *arXiv preprint arXiv:2403.14589*, 2024.

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

[51] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 2021.

[52] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023.

[53] Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Song XiXuan, et al. Cogvlm: Visual expert for pretrained language models. *Advances in Neural Information Processing Systems*, 37:121475–121499, 2025.

[54] Yunfei Chu, Jin Xu, Qian Yang, Haojie Wei, Xipin Wei, Zhifang Guo, Yichong Leng, Yuanjun Lv, Jinzheng He, Junyang Lin, et al. Qwen2-audio technical report. *arXiv preprint arXiv:2407.10759*, 2024.

[55] Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning, 2025. URL https://arxiv.org/abs/2503.09516.

[56] NovaSky Team. Sky-t1: Train your own o1 preview model within $450, 2025.

[57] Open Thoughts Team. Open Thoughts, January 2025.

[58] Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*, 2025.

[59] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning, 2022. URL https://arxiv.org/abs/2203.14465.

[60] Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. Reinforced self-training (rest) for language modeling, 2023. URL https://arxiv.org/abs/2308.08998.

[61] Jun Wang, Meng Fang, Ziyu Wan, Muning Wen, Jiachen Zhu, Anjie Liu, Ziqin Gong, Yan Song, Lei Chen, Lionel M. Ni, Linyi Yang, Ying Wen, and Weinan Zhang. Openr: An open source framework for advanced reasoning with large language models. *CoRR*, abs/2410.09671, 2024.

[62] Di Zhang, Jianbo Wu, Jingdi Lei, Tong Che, Jiatong Li, Tong Xie, Xiaoshui Huang, Shufei Zhang, Marco Pavone, Yuqiang Li, Wanli Ouyang, and Dongzhan Zhou. Llama-berry: Pairwise optimization for o1-like olympiad-level mathematical reasoning. *CoRR*, abs/2410.02884, 2024.

[63] Zihan Wang*, Kangrui Wang*, Qineng Wang*, Pingyue Zhang*, Linjie Li*, Zhengyuan Yang, Kefan Yu, Minh Nhat Nguyen, Monica Lam, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. Training agents by reinforcing reasoning, 2025. URL https://github.com/ZihanWang314/ragen.

[64] Hugging Face. Open-r1, 2024. URL https://github.com/huggingface/open-r1.

[65] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. In *Conference on Robot Learning*, pages 1769–1782. PMLR, 2023.

[66] Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *arXiv preprint arXiv:2302.01560*, 2023.

[67] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36, 2024.

[68] Zhibin Gou, Zhihong Shao, Yeyun Gong, Yujiu Yang, Nan Duan, Weizhu Chen, et al. Critic: Large language models can self-correct with tool-interactive critiquing. In *The Twelfth International Conference on Learning Representations*, 2024.

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

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

[71] Wen Yang, Minpeng Liao, and Kai Fan. Markov chain of thought for efficient mathematical reasoning. *arXiv preprint arXiv:2410.17635*, 2024.

[72] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik R Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[73] Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning, acting, and planning in language models. In *Forty-first International Conference on Machine Learning*, 2024.

[74] Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8154–8173, 2023.

[75] Maciej Besta, Nils Blach, Ale*vs.* Kubíček, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of thoughts: Solving elaborate problems with large language models. In *AAAI Conference on Artificial Intelligence*, 2023. URL https://api.semanticscholar.org/CorpusID:261030303.

[76] Ge Zhang, Mohammad Ali Alomrani, Hongjian Gu, Jiaming Zhou, Yaochen Hu, Bin Wang, Qun Liu, Mark Coates, Yingxue Zhang, and Jianye Hao. Path-of-thoughts: Extracting and following paths for robust relational reasoning with large language models. *arXiv preprint arXiv:2412.17963*, 2024.

[77] Yifan Zhang, Yang Yuan, and Andrew Chi-Chih Yao. On the diagram of thought. *ArXiv*, abs/2409.10038, 2024. URL https://api.semanticscholar.org/CorpusID:272690308.

[78] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, 2023.

[79] Chuanyang Zheng, Zhengying Liu, Enze Xie, Zhenguo Li, and Yu Li. Progressive-hint prompting improves reasoning in large language models. *arXiv preprint arXiv:2304.09797*, 2023.

[80] Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. On the self-verification limitations of large language models on reasoning and planning tasks. *arXiv preprint arXiv:2402.08115*, 2024.

[81] Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason E Weston. Chain-of-verification reduces hallucination in large language models. In *ICLR 2024 Workshop on Reliable and Responsible Foundation Models*, 2024.

[82] Shima Imani, Liang Du, and Harsh Shrivastava. Mathprompter: Mathematical reasoning using large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 37–42, 2023.

[83] Zhuoxuan Jiang, Haoyuan Peng, Shanshan Feng, Fan Li, and Dongsheng Li. Llms can find mathematical reasoning mistakes by pedagogical chain-of-thought. *arXiv preprint arXiv:2405.06705*, 2024.

[84] Xinyu Pang, Ruixin Hong, Zhanke Zhou, Fangrui Lv, Xinwei Yang, Zhilong Liang, Bo Han, and Changshui Zhang. Physics reasoner: Knowledge-augmented reasoning for solving physics problems with large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 11274–11289, 2025.

[85] Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H Chi, Quoc V Le, and Denny Zhou. Take a step back: Evoking reasoning via abstraction in large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[86] Simran Arora, Avanika Narayan, Mayee F Chen, Laurel Orr, Neel Guha, Kush Bhatia, Ines Chami, Frederic Sala, and Christopher Ré. Ask me anything: A simple strategy for prompting language models. *arXiv preprint arXiv:2210.02441*, 2022.

[87] Xingxuan Li, Ruochen Zhao, Yew Ken Chia, Bosheng Ding, Shafiq Joty, Soujanya Poria, and Lidong Bing. Chain-of-knowledge: Grounding large language models via dynamic knowledge adapting over heterogeneous sources. *arXiv preprint arXiv:2305.13269*, 2023.

[88] Lishui Fan, Mouxiang Chen, and Zhongxin Liu. Self-explained keywords empower large language models for code generation. *arXiv preprint arXiv:2410.15966*, 2024.

[89] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[90] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[91] Eric Zelikman, Georges Harik, Yijia Shao, Varuna Jayasiri, Nick Haber, and Noah D Goodman. Quiet-star: Language models can teach themselves to think before speaking. *arXiv preprint arXiv:2403.09629*, 2024.

[92] Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

[93] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530. IEEE, 2023.

[94] Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. Adapt: As-needed decomposition and planning with language models. *arXiv preprint arXiv:2311.05772*, 2023.

[95] Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. Travelplanner: a benchmark for real-world planning with language agents. In *ICML*, 2024.

[96] Drew McDermott et al. Pddl—the planning domain definition language. *AIPS-98 Planning Competition Committee*, 1998. Defines PDDL, a standard language for planning domains used in LLM integrations.

[97] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36, 2023.

[98] George A Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review*, 63(2):81, 1956.

[99] Kenji Doya. Complementary roles of basal ganglia and cerebellum in learning and motor control. *Current opinion in neurobiology*, 10(6):732–739, 2000.

[100] Jerry A Fodor. *The modularity of mind*. MIT press, 1983.

[101] Joshua D. McGraw, Donsuk Lee, and Justin N. Wood. Parallel development of social behavior in biological and artificial fish. *Nature Communications*, 2024.

[102] Hongjin Su, Ruoxi Sun, Jinsung Yoon, Pengcheng Yin, Tao Yu, and Sercan Ö Arık. Learn-by-interact: A data-centric framework for self-adaptive agents in realistic environments. *arXiv preprint arXiv:2501.10893*, 2025.

[103] Hao Bai, Yifei Zhou, Mert Cemri, Jiayi Pan, Alane Suhr, Sergey Levine, and Aviral Kumar. Digirl: Training in-the-wild device-control agents with autonomous reinforcement learning. *arXiv preprint arXiv:2406.11896*, 2024.

[104] Hao Peng, Yunjia Qi, Xiaozhi Wang, Zijun Yao, Bin Xu, Lei Hou, and Juanzi Li. Agentic reward modeling: Integrating human preferences with verifiable correctness signals for reliable reward systems, 2025. URL https://arxiv.org/abs/2502.19328.

[105] Tianbao Xie, Siheng Zhao, Chen Henry Wu, Yitao Liu, Qian Luo, Victor Zhong, Yanchao Yang, and Tao Yu. Text2reward: Automated dense reward function generation for reinforcement learning. *arXiv preprint arXiv:2309.11489*, 2023.

[106] Zhenfang Chen, Delin Chen, Rui Sun, Wenjun Liu, and Chuang Gan. Scaling autonomous agents via automatic reward modeling and planning, 2025. URL https://arxiv.org/abs/2502.12130.

[107] Yu Gu, Boyuan Zheng, Boyu Gou, Kai Zhang, Cheng Chang, Sanjari Srivastava, Yanan Xie, Peng Qi, Huan Sun, and Yu Su. Is your LLM secretly a world model of the internet? model-based planning for web agents. *arXiv preprint arXiv:2411.06559*, 2024.

[108] Minghao Chen, Yihang Li, Yanting Yang, Shiyu Yu, Binbin Lin, and Xiaofei He. AutoManual: Generating instruction manuals by LLM agents via interactive environmental learning. *arXiv preprint arXiv:2405.16247*, 2024.

[109] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[110] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

[111] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

[112] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024.

[113] Kimi Team. Kimi k1.5: Scaling reinforcement learning with llms, 2025. URL https://arxiv.org/abs/2501.12599.

[114] Hao Li, Xue Yang, Zhaokai Wang, Xizhou Zhu, Jie Zhou, Yu Qiao, Xiaogang Wang, Hongsheng Li, Lewei Lu, and Jifeng Dai. Auto mc-reward: Automated dense reward design with large language models for minecraft. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16426–16435, 2024.

[115] Letian Fu, Gaurav Datta, Huang Huang, William Chung-Ho Panitch, Jaimyn Drake, Joseph Ortiz, Mustafa Mukadam, Mike Lambeta, Roberto Calandra, and Ken Goldberg. A touch, vision, and language dataset for multimodal alignment. *arXiv preprint arXiv:2402.13232*, 2024.

[116] Shailja Gupta, Rajesh Ranjan, and Surya Narayan Singh. A comprehensive survey of retrieval-augmented generation (rag): Evolution, current landscape and future directions. *arXiv preprint arXiv:2410.12837*, 2024.

[117] Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models, 2025. URL https://arxiv.org/abs/2501.05366.

[118] Qwen Team. Qwq: Reflect deeply on the boundaries of the unknown, November 2024. URL https://qwenlm.github.io/blog/qwq-32b-preview/.

[119] Dacheng Li, Shiyi Cao, Tyler Griggs, Shu Liu, Xiangxi Mo, Shishir G Patil, Matei Zaharia, Joseph E Gonzalez, and Ion Stoica. Llms can easily learn to reason from demonstrations structure, not content, is what matters! *arXiv preprint arXiv:2502.07374*, 2025.

[120] Arian Hosseini, Xingdi Yuan, Nikolay Malkin, Aaron Courville, Alessandro Sordoni, and Rishabh Agarwal. V-star: Training verifiers for self-taught reasoners, 2024. URL https://arxiv.org/abs/2402.06457.

[121] Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. rstar-math: Small LLMs can master math reasoning with self-evolved deep thinking, 2025.

[122] Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron Parisi, Abhishek Kumar, Alex Alemi, Alex Rizkowsky, Azade Nova, Ben Adlam, Bernd Bohnet, Gamaleldin Elsayed, Hanie Sedghi, Igor Mordatch, Isabelle Simpson, Izzeddin Gur, Jasper Snoek, Jeffrey Pennington, Jiri Hron, Kathleen Kenealy, Kevin Swersky, Kshiteej Mahajan, Laura Culp, Lechao Xiao, Maxwell L. Bileschi, Noah Constant, Roman Novak, Rosanne Liu, Tris Warkentin, Yundi Qian, Yamini Bansal, Ethan Dyer, Behnam Neyshabur, Jascha Sohl-Dickstein, and Noah Fiedel. Beyond human data: Scaling self-training for problem-solving with language models, 2024. URL https://arxiv.org/abs/2312.06585.

[123] Yuxiang Zhang, Shangxi Wu, Yuqi Yang, Jiangming Shu, Jinlin Xiao, Chao Kong, and Jitao Sang. o1-coder: an o1 replication for coding. *CoRR*, abs/2412.00154, 2024.

[124] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient RLHF framework. *CoRR*, abs/2409.19256, 2024.

[125] Yuxiang Zhang, Yuqi Yang, Jiangming Shu, Yuhang Wang, Jinlin Xiao, and Jitao Sang. Openrft: Adapting reasoning foundation model for domain-specific tasks with reinforcement fine-tuning. *CoRR*, abs/2412.16849, 2024.

[126] Jiayi Pan, Xingyao Wang, Graham Neubig, Navdeep Jaitly, Heng Ji, Alane Suhr, and Yizhe Zhang. Training software engineering agents and verifiers with swe-gym. *CoRR*, abs/2412.21139, 2024.

[127] Zonghan Yang, Peng Li, Ming Yan, Ji Zhang, Fei Huang, and Yang Liu. React meets actre: Autonomous annotations of agent trajectories for contrastive self-training. *arXiv preprint arXiv:2403.14589*, 2024.

[128] Geunwoo Kim, Pierre Baldi, and Stephen McAleer. Language models can solve computer tasks. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[129] Hyungjoo Chae, Namyoung Kim, Kai Tzu-iunn Ong, Minju Gwak, Gwanwoo Song, Jihoon Kim, Sunghwan Kim, Dongha Lee, and Jinyoung Yeo. Web agents with world models: Learning and leveraging environment dynamics in web navigation. In *The Thirteenth International Conference on Learning Representations*, 2024.

[130] Kewei Cheng, Jingfeng Yang, Haoming Jiang, Zhengyang Wang, Binxuan Huang, Ruirui Li, Shiyang Li, Zheng Li, Yifan Gao, Xian Li, et al. Inductive or deductive? rethinking the fundamental reasoning abilities of llms. *arXiv preprint arXiv:2408.00114*, 2024.

[131] Brett K Hayes, Evan Heit, and Haruka Swendsen. Inductive reasoning. *Wiley interdisciplinary reviews: Cognitive science*, 1(2):278–292, 2010.

[132] Fengwei Teng, Zhaoyang Yu, Quan Shi, Jiayi Zhang, Chenglin Wu, and Yuyu Luo. Atom of thoughts for markov llm test-time scaling, 2025. URL https://arxiv.org/abs/2502.12018.

[133] Harsha Nori, Yin Tat Lee, Sheng Zhang, Dean Carignan, Richard Edgar, Nicolo Fusi, Nicholas King, Jonathan Larson, Yuanzhi Li, Weishung Liu, et al. Can generalist foundation models outcompete special-purpose tuning? case study in medicine. *arXiv preprint arXiv:2311.16452*, 2023.

[134] Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Annual Meeting of the Association for Computational Linguistics*, 2023. URL https://api.semanticscholar.org/CorpusID:259075564.

[135] Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beatriz Borges, Antoine Bosselut, Robert West, and Boi Faltings. Refiner: Reasoning feedback on intermediate representations. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1100–1126, 2024.

[136] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

[137] Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations*, 2023.

[138] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, et al. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*, 2023.

[139] Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. Complexity-based prompting for multi-step reasoning. In *The Eleventh International Conference on Learning Representations*, 2023.

[140] Yihe Deng, Weitong Zhang, Zixiang Chen, and Quanquan Gu. Rephrase and respond: Let large language models ask better questions for themselves. *CoRR*, abs/2311.04205, 2023.

[141] Ruixin Hong, Hongming Zhang, Xiaoman Pan, Dong Yu, and Changshui Zhang. Abstraction-of-thought makes language models better reasoners. *arXiv preprint arXiv:2406.12442*, 2024.

[142] Bilgehan Sel, Ahmad Al-Tawaha, Vanshaj Khattar, Ruoxi Jia, and Ming Jin. Algorithm of thoughts: Enhancing exploration of ideas in large language models. *arXiv preprint arXiv:2308.10379*, 2023.

[143] Tianhe Lin, Jian Xie, Siyu Yuan, and Deqing Yang. Implicit reasoning in transformers is reasoning through shortcuts. *arXiv preprint arXiv:2503.07604*, 2025.

[144] Allen Newell, John Calman Shaw, and Herbert A Simon. Elements of a theory of human problem solving. *Psychological review*, 65(3):151, 1958.

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

[146] Haoming Li, Zhaoliang Chen, Jonathan Zhang, and Fei Liu. Lasp: Surveying the state-of-the-art in large language model-assisted ai planning. *arXiv preprint arXiv:2409.01806*, 2024.

[147] Subbarao Kambhampati. Can large language models reason and plan? *Annals of the New York Academy of Sciences*, 1534(1):15–18, 2024.

[148] Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. On the planning abilities of large language models-a critical investigation. *Advances in Neural Information Processing Systems*, 36:75993–76005, 2023.

[149] Vishal Pallagani, Bharath Muppasani, Keerthiram Murugesan, Francesca Rossi, Biplav Srivastava, Lior Horesh, Francesco Fabiano, and Andrea Loreggia. Understanding the capabilities of large language models for automated planning. *arXiv preprint arXiv:2305.16151*, 2023.

[150] Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Kaya Stechly, Mudit Verma, Siddhant Bhambri, Lucas Saldyt, and Anil Murthy. Llms can't plan, but can help planning in llm-modulo frameworks. *arXiv preprint arXiv:2402.01817*, 2024.

[151] Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. Adapt: As-needed decomposition and planning with language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4226–4252, 2024.

[152] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *Advances in Neural Information Processing Systems*, 36, 2024.

[153] Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. Chameleon: Plug-and-play compositional reasoning with large language models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[154] Fangru Lin, Emanuele La Malfa, Valentin Hofmann, Elle Michelle Yang, Anthony Cohn, and Janet B Pierrehumbert. Graph-enhanced large language models in asynchronous plan reasoning. *arXiv preprint arXiv:2402.02805*, 2024.

[155] Amrith Setlur, Nived Rajaraman, Sergey Levine, and Aviral Kumar. Scaling test-time compute without verification or rl is suboptimal. *arXiv preprint arXiv:2502.12118*, 2025.

[156] Shibo Hao, Yi Gu, Haotian Luo, Tianyang Liu, Xiyan Shao, Xinyuan Wang, Shuhua Xie, Haodi Ma, Adithya Samavedhi, Qiyue Gao, et al. Llm reasoners: New evaluation, library, and analysis of step-by-step reasoning with large language models. In *First Conference on Language Modeling*, 2024.

[157] Jinghan Zhang and Kunpeng Liu. Thought space explorer: Navigating and expanding thought space for large language model reasoning. In *2024 IEEE International Conference on Big Data (BigData)*, pages 8259–8251. IEEE, 2024.

[158] Siheng Xiong, Ali Payani, Ramana Kompella, and Faramarz Fekri. Large language models can learn temporal reasoning. *CoRR*, 2024.

[159] Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning acting and planning in language models. *arXiv preprint arXiv:2310.04406*, 2023.

[160] Owen Burns, Dana Hughes, and Katia Sycara. Plancritic: Formal planning with human feedback. *arXiv preprint arXiv:2412.00300*, 2024.

[161] Zirui Zhao, Wee Sun Lee, and David Hsu. Large language models as commonsense knowledge for large-scale task planning. *Advances in Neural Information Processing Systems*, 36, 2024.

[162] Zhiting Hu and Tianmin Shu. Language models, agent models, and world models: The law for machine reasoning and planning. *arXiv preprint arXiv:2312.05230*, 2023.

[163] Bo Liu, Yuqian Jiang, Xiaohan Zhang, Qiang Liu, Shiqi Zhang, Joydeep Biswas, and Peter Stone. Llm+ p: Empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*, 2023.

[164] Sadegh Mahdavi, Raquel Aoki, Keyi Tang, and Yanshuai Cao. Leveraging environment interaction for automated PDDL translation and planning with large language models. In *NeurIPS*, 2024.

[165] Lin Guan, Karthik Valmeekam, Sarath Sreedharan, and Subbarao Kambhampati. Leveraging pre-trained large language models to construct and utilize world models for model-based task planning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[166] Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Mudit Verma, Kaya Stechly, Siddhant Bhambri, Lucas Paul Saldyt, and Anil B Murthy. Position: Llms can't plan, but can help planning in llm-modulo frameworks. In *Forty-first International Conference on Machine Learning*, 2024.

[167] Jiaxin Wen, Jian Guan, Hongning Wang, Wei Wu, and Minlie Huang. Unlocking reasoning potential in large langauge models by scaling code-form planning. *arXiv preprint arXiv:2409.12452*, 2024.

[168] Shuofei Qiao, Runnan Fang, Ningyu Zhang, Yuqi Zhu, Xiang Chen, Shumin Deng, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. Agent planning with world knowledge model. *Advances in Neural Information Processing Systems*, 37:114843–114871, 2024.

[169] Jun Wang, Jiaming Tong, Kaiyuan Tan, Yevgeniy Vorobeychik, and Yiannis Kantaros. Conformal temporal logic planning using large language models. *arXiv preprint arXiv:2309.10092*, 2023.

[170] Richard C Atkinson. Human memory: A proposed system and its control processes. *The psychology of learning and motivation*, 2, 1968.

[171] Kieran CR Fox, Nicholas S Fitz, and Peter B Reiner. The multiplicity of memory enhancement: Practical and ethical implications of the diverse neural substrates underlying human memory systems. *Neuroethics*, 10: 375–388, 2017.

[172] Alan Baddeley. Working memory. *Science*, 255(5044):556–559, 1992.

[173] George Sperling. The information available in brief visual presentations. *Psychological monographs: General and applied*, 74(11):1, 1960.

[174] Max Coltheart. Iconic memory and visible persistence. *Perception & psychophysics*, 27:183–228, 1980.

[175] JM Gardiner. On recency and echoic memory. *Philosophical Transactions of the Royal Society of London. B, Biological Sciences*, 302(1110):267–282, 1983.

[176] Bart Aben, Sven Stapert, and Arjan Blokland. About the distinction between working memory and short-term memory. *Frontiers in psychology*, 3:301, 2012.

[177] Nelson Cowan. What are the differences between long-term, short-term, and working memory? *Progress in brain research*, 169:323–338, 2008.

[178] Richard M Shiffrin and Richard C Atkinson. Storage and retrieval processes in long-term memory. *Psychological review*, 76(2):179, 1969.

[179] Dennis Norris. Short-term memory and long-term memory are still different. *Psychological bulletin*, 143(9): 992, 2017.

[180] Hermann Ebbinghaus. Memory: A contribution to experimental psychology. *Annals of neurosciences*, 20(4): 155, 2013.

[181] Howard Eichenbaum. Declarative memory: Insights from cognitive neurobiology. *Annual review of psychology*, 48(1):547–572, 1997.

[182] Abhilasha A Kumar. Semantic memory: A review of methods, models, and current challenges. *Psychonomic bulletin & review*, 28(1):40–80, 2021.

[183] Endel Tulving. Episodic memory: From mind to brain. *Annual review of psychology*, 53(1):1–25, 2002.

[184] Robyn Fivush. The development of autobiographical memory. *Annual review of psychology*, 62(1):559–582, 2011.

[185] Larry R Squire. Declarative and nondeclarative memory: Multiple brain systems supporting learning and memory. *Journal of cognitive neuroscience*, 4(3):232–243, 1992.

[186] Prahlad Gupta and Neal J Cohen. Theoretical and computational analysis of skill learning, repetition priming, and procedural memory. *Psychological review*, 109(2):401, 2002.

[187] Neal J Cohen and Larry R Squire. Preserved learning and retention of pattern-analyzing skill in amnesia: Dissociation of knowing how and knowing that. *Science*, 210(4466):207–210, 1980.

[188] Endel Tulving and Daniel L Schacter. Priming and human memory systems. *Science*, 247(4940):301–306, 1990.

[189] Robert E Clark, Joseph R Manns, and Larry R Squire. Classical conditioning, awareness, and brain systems. *Trends in cognitive sciences*, 6(12):524–531, 2002.

[190] Androulla Ioannou and Xenia Anastassiou-Hadjicharalambous. Non-associative learning. *Encyclopedia of evolutionary psychological science*, pages 5419–5432, 2021.

[191] Martin A Conway and Christopher W Pleydell-Pearce. The construction of autobiographical memories in the self-memory system. *Psychological review*, 107(2):261, 2000.

[192] Alan D Baddeley, Graham Hitch, and Gordon H Bower. Working memory. volume 8 of. *Psychology of Learning and Motivation*, pages 47–89, 1974.

[193] Alan Baddeley. The episodic buffer: a new component of working memory? *Trends in cognitive sciences*, 4 (11):417–423, 2000.

[194] Nelson Cowan. Evolving conceptions of memory storage, selective attention, and their mutual constraints within the human information-processing system. *Psychological bulletin*, 104(2):163, 1988.

[195] Endel Tulving. Memory and consciousness. *Canadian Psychology/Psychologie canadienne*, 26(1):1, 1985.

[196] Bernard J Baars. *A cognitive theory of consciousness*. Cambridge University Press, 1993.

[197] Stan Franklin. *Artificial minds*. MIT press, 1997.

[198] Stan Franklin, Arpad Kelemen, and Lee McCauley. Ida: A cognitive agent architecture. In *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)*, volume 3, pages 2646–2651. IEEE, 1998.

[199] John R Anderson. *How can the human mind occur in the physical universe?* Oxford University Press, 2009.

[200] Yuheng Cheng, Ceyao Zhang, Zhengwen Zhang, Xiangrui Meng, Sirui Hong, Wenhao Li, Zihao Wang, Zekai Wang, Feng Yin, Junhua Zhao, et al. Exploring large language model based intelligent agents: Definitions, methods, and prospects. *arXiv preprint arXiv:2401.03428*, 2024.

[201] Alan Baddeley. Working memory. *Current biology*, 20(4):R136–R140, 2010.

[202] Jose Camacho-Collados and Mohammad Taher Pilehvar. From word to sense embeddings: A survey on vector representations of meaning. *Journal of Artificial Intelligence Research*, 63:743–788, 2018.

[203] Lei Liu, Xiaoyan Yang, Yue Shen, Binbin Hu, Zhiqiang Zhang, Jinjie Gu, and Guannan Zhang. Think-in-memory: Recalling and post-thinking enable llms with long-term memory. *arXiv preprint arXiv:2311.08719*, 2023.

[204] Zhuosheng Zhang and Aston Zhang. You only look at screens: Multimodal chain-of-action agents. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 3132–3149, 2024.

[205] Lei Wang, Jingsen Zhang, Hao Yang, Zhiyuan Chen, Jiakai Tang, Zeyu Zhang, Xu Chen, Yankai Lin, Ruihua Song, Wayne Xin Zhao, et al. User behavior simulation with large language model based agents. *arXiv preprint arXiv:2306.02552*, 2023.

[206] Yujia Zhou, Qiannan Zhu, Jiajie Jin, and Zhicheng Dou. Cognitive personalized search integrating large language models with an efficient memory mechanism. In *Proceedings of the ACM on Web Conference 2024*, pages 1464–1473, 2024.

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

[208] Ziheng Huang, Sebastian Gutierrez, Hemanth Kamana, and Stephen MacNeil. Memory sandbox: Transparent and interactive memory management for conversational agents. In *Adjunct Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–3, 2023.

[209] Yue Fan, Xiaojian Ma, Rujie Wu, Yuntao Du, Jiaqi Li, Zhi Gao, and Qing Li. Videoagent: A memory-augmented multimodal agent for video understanding. In *European Conference on Computer Vision*, pages 75–92. Springer, 2024.

[210] Zhiqi Ge, Hongzhe Huang, Mingze Zhou, Juncheng Li, Guoming Wang, Siliang Tang, and Yueting Zhuang. Worldgpt: Empowering LLM as multimodal world model. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 7346–7355, 2024.

[211] Saaket Agashe, Jiuzhou Han, Shuyu Gan, Jiachen Yang, Ang Li, and Xin Eric Wang. Agent s: An open agentic framework that uses computers like a human. *arXiv preprint arXiv:2410.08164*, 2024.

[212] Zhiyong Wu, Chengcheng Han, Zichen Ding, Zhenmin Weng, Zhoumianze Liu, Shunyu Yao, Tao Yu, and Lingpeng Kong. Os-copilot: Towards generalist computer agents with self-improvement. *arXiv preprint arXiv:2402.07456*, 2024.

[213] Sen Li, Ruochen Wang, Cho-Jui Hsieh, Minhao Cheng, and Tianyi Zhou. Mulan: Multimodal-llm agent for progressive multi-object diffusion. *arXiv preprint arXiv:2402.12741*, 2024.

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

[215] Zixuan Wang, Bo Yu, Junzhe Zhao, Wenhao Sun, Sai Hou, Shuai Liang, Xing Hu, Yinhe Han, and Yiming Gan. Karma: Augmenting embodied ai agents with long-and-short term memory systems. *arXiv preprint arXiv:2409.14908*, 2024.

[216] Zeru Shi, Kai Mei, Mingyu Jin, Yongye Su, Chaoji Zuo, Wenyue Hua, Wujiang Xu, Yujie Ren, Zirui Liu, Mengnan Du, et al. From commands to prompts: Llm-based semantic file system for aios. *arXiv preprint arXiv:2410.11843*, 2024.

[217] Xiaoqiang Wang and Bang Liu. Oscar: Operating system control via state-aware reasoning and re-planning. *arXiv preprint arXiv:2410.18963*, 2024.

[218] Kevin A Fischer. Reflective linguistic programming (rlp): A stepping stone in socially-aware agi (socialagi). *arXiv preprint arXiv:2305.12647*, 2023.

[219] Andrew Zhu, Lara Martin, Andrew Head, and Chris Callison-Burch. Calypso: LLMs as dungeon master's assistants. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 19, pages 380–390, 2023.

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

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

[222] Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. Hipporag: Neurobiologically inspired long-term memory for large language models. In *NeurIPS*, 2024.

[223] Sunjae Lee, Junyoung Choi, Jungjae Lee, Munim Hasan Wasi, Hojun Choi, Steven Y Ko, Sangeun Oh, and Insik Shin. Explore, select, derive, and recall: Augmenting llm with human-like memory for mobile task automation. *arXiv preprint arXiv:2312.03003*, 2023.

[224] Leonard Bärmann, Chad DeChant, Joana Plewnia, Fabian Peller-Konrad, Daniel Bauer, Tamim Asfour, and Alex Waibel. Episodic memory verbalization using hierarchical representations of life-long robot experience. *arXiv preprint arXiv:2409.17702*, 2024.

[225] Junyeong Park, Junmo Cho, and Sungjin Ahn. Mr. steve: Instruction-following agents in minecraft with what-where-when memory. *arXiv preprint arXiv:2411.06736*, 2024.

[226] K Roth, Rushil Gupta, Simon Halle, and Bang Liu. Pairing analogy-augmented generation with procedural memory for procedural q&a. *arXiv preprint arXiv:2409.01344*, 2024.

[227] Weihao Tan, Ziluo Ding, Wentao Zhang, Boyu Li, Bohan Zhou, Junpeng Yue, Haochong Xia, Jiechuan Jiang, Longtao Zheng, Xinrun Xu, et al. Towards general computer control: A multimodal agent for red dead redemption ii as a case study. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*, 2024.

[228] Zihao Wang, Shaofei Cai, Anji Liu, Yonggang Jin, Jinbing Hou, Bowei Zhang, Haowei Lin, Zhaofeng He, Zilong Zheng, Yaodong Yang, et al. Jarvis-1: Open-world multi-task agents with memory-augmented multimodal language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[229] Ming Yan, Ruihao Li, Hao Zhang, Hao Wang, Zhilan Yang, and Ji Yan. Larp: Language-agent role play for open-world games. *arXiv preprint arXiv:2312.17653*, 2023.

[230] Yijun Liu, Wu Liu, Xiaoyan Gu, Yong Rui, Xiaodong He, and Yongdong Zhang. Lmagent: A large-scale multimodal agents society for multi-user simulation. *arXiv preprint arXiv:2412.09237*, 2024.

[231] Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. A human-inspired reading agent with gist memory of very long contexts. *arXiv preprint arXiv:2402.09727*, 2024.

[232] Shuai Wang, Liang Ding, Yibing Zhan, Yong Luo, Zheng He, and Dapeng Tao. Leveraging metamemory mechanisms for enhanced data-free code generation in llms. *arXiv preprint arXiv:2501.07892*, 2025.

[233] Pengbo Hu and Xiang Ying. Unified mind model: Reimagining autonomous agents in the llm era. *arXiv preprint arXiv:2503.03459*, 2025.

[234] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5149–5169, 2021.

[235] Yuki Hou, Haruki Tamoto, and Homei Miyashita. "my agent understands me better": Integrating dynamic human-like memory recall and consolidation in llm-based agents. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pages 1–7, 2024.

[236] Bo Pan, Jiaying Lu, Ke Wang, Li Zheng, Zhen Wen, Yingchaojie Feng, Minfeng Zhu, and Wei Chen. Agentcoord: Visually exploring coordination strategy for llm-based multi-agent collaboration. *arXiv preprint arXiv:2404.11943*, 2024.

[237] Hang Gao and Yongfeng Zhang. Memory sharing for large language model based agents. *arXiv preprint arXiv:2404.09982*, 2024.

[238] Meng Chu, Yicong Li, and Tat-Seng Chua. Understanding long videos via llm-powered entity relation graphs. *arXiv preprint arXiv:2501.15953*, 2025.

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

[240] Hassan Ali, Philipp Allgeuer, Carlo Mazzola, Giulia Belgiovine, Burak Can Kaplan, Lukáš Gajdošech, and Stefan Wermter. Robots can multitask too: Integrating a memory architecture and llms for enhanced cross-task robot action generation. In *2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids)*, pages 811–818. IEEE, 2024.

[241] Zaijing Li, Yuquan Xie, Rui Shao, Gongwei Chen, Dongmei Jiang, and Liqiang Nie. Optimus-1: Hybrid multimodal memory empowered agents excel in long-horizon tasks. *arXiv preprint arXiv:2408.03615*, 2024.

[242] Zaijing Li, Yuquan Xie, Rui Shao, Gongwei Chen, Dongmei Jiang, and Liqiang Nie. Optimus-2: Multimodal minecraft agent with goal-observation-action conditioned policy. *arXiv preprint arXiv:2502.19902*, 2025.

[243] Tenghao Huang, Kinjal Basu, Ibrahim Abdelaziz, Pavan Kapanipathi, Jonathan May, and Muhao Chen. R2d2: Remembering, reflecting and dynamic decision making for web agents. *arXiv preprint arXiv:2501.12485*, 2025.

[244] Zhenhailong Wang, Haiyang Xu, Junyang Wang, Xi Zhang, Ming Yan, Ji Zhang, Fei Huang, and Heng Ji. Mobile-agent-e: Self-evolving mobile assistant for complex tasks. *arXiv preprint arXiv:2501.11733*, 2025.

[245] Philippe Laban, Wojciech Kryściński, Divyansh Agarwal, Alexander Richard Fabbri, Caiming Xiong, Shafiq Joty, and Chien-Sheng Wu. Summedits: Measuring llm ability at factual reasoning through the lens of summarization. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 9662–9676, 2023.

[246] Bing Wang, Xinnian Liang, Jian Yang, Hui Huang, Shuangzhi Wu, Peihao Wu, Lu Lu, Zejun Ma, and Zhoujun Li. Enhancing large language model with self-controlled memory framework. *arXiv preprint arXiv:2304.13343*, 2023.

[247] Zhiyao Ren, Yibing Zhan, Baosheng Yu, Liang Ding, and Dacheng Tao. Healthcare copilot: Eliciting the power of general llms for medical consultation. *arXiv preprint arXiv:2402.13408*, 2024.

[248] Qingyue Wang, Liang Ding, Yanan Cao, Zhiliang Tian, Shi Wang, Dacheng Tao, and Li Guo. Recursively summarizing enables long-term dialogue memory in large language models. *arXiv preprint arXiv:2308.15022*, 2023.

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

[250] Yudi Shi, Shangzhe Di, Qirui Chen, and Weidi Xie. Unlocking video-llm via agent-of-thoughts distillation. *arXiv preprint arXiv:2412.01694*, 2024.

[251] Jiaqi Liu, Chengkai Xu, Peng Hang, Jian Sun, Mingyu Ding, Wei Zhan, and Masayoshi Tomizuka. Language-driven policy distillation for cooperative driving in multi-agent reinforcement learning. *arXiv preprint arXiv:2410.24152*, 2024.

[252] Maryam Hashemzadeh, Elias Stengel-Eskin, Sarath Chandar, and Marc-Alexandre Cote. Sub-goal distillation: A method to improve small language agents. *arXiv preprint arXiv:2405.02749*, 2024.

[253] Justin Chih-Yao Chen, Swarnadeep Saha, Elias Stengel-Eskin, and Mohit Bansal. Magdi: structured distillation of multi-agent interaction graphs improves reasoning in smaller language models. In *Proceedings of the 41st International Conference on Machine Learning*, pages 7220–7235, 2024.

[254] Zhao Kaiya, Michelangelo Naim, Jovana Kondic, Manuel Cortes, Jiaxin Ge, Shuying Luo, Guangyu Robert Yang, and Andrew Ahn. Lyfe agents: Generative agents for low-cost real-time social interactions. *arXiv preprint arXiv:2310.02172*, 2023.

[255] Chen Gao, Xiaochong Lan, Zhihong Lu, Jinzhu Mao, Jinghua Piao, Huandong Wang, Depeng Jin, and Yong Li. S^3: Social-network simulation system with large language model-empowered agents. *arXiv preprint arXiv:2307.14984*, 2023.

[256] Yang Li, Yangyang Yu, Haohang Li, Zhi Chen, and Khaldoun Khashanah. Tradinggpt: Multi-agent system with layered memory and distinct characters for enhanced financial trading performance. *arXiv preprint arXiv:2309.03736*, 2023.

[257] Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. Longmemeval: Benchmarking chat assistants on long-term interactive memory. *arXiv preprint arXiv:2410.10813*, 2024.

[258] Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Xufang Luo, Hao Cheng, Dongsheng Li, Yuqing Yang, Chin-Yew Lin, H Vicky Zhao, Lili Qiu, et al. On memory construction and retrieval for personalized conversational agents. *arXiv preprint arXiv:2502.05589*, 2025.

[259] Guillaume Lample, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Large memory layers with product keys. *Advances in Neural Information Processing Systems*, 32, 2019.

[260] Jiaming Xu, Kaibin Guo, Wuxuan Gong, and Runyu Shi. Osagent: Copiloting operating system with llm-based agent. In *2024 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9. IEEE, 2024.

[261] Dzmitry Bahdanau. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[262] Mete Demircigil, Judith Heusel, Matthias Löwe, Sven Upgang, and Franck Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168:288–299, 2017.

[263] Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, et al. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*, 2020.

[264] Alex Falcon, Giovanni D'Agostino, Oswald Lanz, Giorgio Brajnik, Carlo Tasso, and Giuseppe Serra. Neural turing machines for the remaining useful life estimation problem. *Computers in Industry*, 143:103762, 2022.

[265] Yu Wang, Yifan Gao, Xiusi Chen, Haoming Jiang, Shiyang Li, Jingfeng Yang, Qingyu Yin, Zheng Li, Xian Li, Bing Yin, Jingbo Shang, and Julian McAuley. Memoryllm: towards self-updatable large language models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[266] Yu Wang, Xinshuang Liu, Xiusi Chen, Sean O'Brien, Junda Wu, and Julian McAuley. Self-updatable large language models with parameter integration. *arXiv preprint arXiv:2410.00487*, 2024.

[267] Hongjin Qian, Peitian Zhang, Zheng Liu, Kelong Mao, and Zhicheng Dou. Memorag: Moving towards next-gen rag via memory-inspired knowledge discovery. *arXiv preprint arXiv:2409.05591*, 2024.

[268] Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, et al. Learning to (learn at test time): Rnns with expressive hidden states. *arXiv preprint arXiv:2407.04620*, 2024.

[269] Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time. *arXiv preprint arXiv:2501.00663*, 2024.

[270] Xiaoqiang Wang, Suyuchen Wang, Yun Zhu, and Bang Liu. R^3mem: Bridging memory retention and retrieval via reversible compression. *arXiv preprint arXiv:2502.15957*, 2025.

[271] Xuanwang Zhang, Yunze Song, Yidong Wang, Shuyun Tang, Xinfeng Li, Zhengran Zeng, Zhen Wu, Wei Ye, Wenyuan Xu, Yue Zhang, et al. Raglab: A modular and research-oriented unified framework for retrieval-augmented generation. *arXiv preprint arXiv:2408.11381*, 2024.

[272] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822, 2023.

[273] Mehrdad Farahani and Richard Johansson. Deciphering the interplay of parametric and non-parametric memory in retrieval-augmented language models. *arXiv preprint arXiv:2410.05162*, 2024.

[274] Ruifeng Yuan, Shichao Sun, Yongqi Li, Zili Wang, Ziqiang Cao, and Wenjie Li. Personalized large language model assistant with evolving conditional memory. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 3764–3777, 2025.

[275] Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. Recurrent memory transformer. *Advances in Neural Information Processing Systems*, 35:11079–11091, 2022.

[276] Aydar Bulatov, Yuri Kuratov, Yermek Kapushev, and Mikhail S Burtsev. Scaling transformer to 1m tokens and beyond with rmt. *arXiv preprint arXiv:2304.11062*, 2023.

[277] Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3829–3846, 2023.

[278] Tao Ge, Jing Hu, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. *arXiv preprint arXiv:2307.06945*, 2023.

[279] Jesse Mu, Xiang Li, and Noah Goodman. Learning to compress prompts with gist tokens. *Advances in Neural Information Processing Systems*, 36, 2024.

[280] Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. Compact: Compressing retrieved documents actively for question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21424–21439, 2024.

[281] Johnny Li, Saksham Consul, Eda Zhou, James Wong, Naila Farooqui, Yuxin Ye, Nithyashree Manohar, Zhuxiaona Wei, Tian Wu, Ben Echols, et al. Banishing llm hallucinations requires rethinking generalization. *arXiv preprint arXiv:2406.17642*, 2024.

[282] Sangjun Park and JinYeong Bak. Memoria: Resolving fateful forgetting problem through human-inspired memory architecture. *arXiv preprint arXiv:2310.03052*, 2023.

[283] Xu Owen He. Mixture of a million experts. *arXiv preprint arXiv:2407.04153*, 2024.

[284] Hanxing Ding, Liang Pang, Zihao Wei, Huawei Shen, and Xueqi Cheng. Retrieve only when it needs: Adaptive retrieval augmentation for hallucination mitigation in large language models. *arXiv preprint arXiv:2402.10612*, 2024.

[285] Yingxu Wang, Dong Liu, and Ying Wang. Discovering the capacity of human memory. *Brain and Mind*, 4: 189–198, 2003.

[286] Jikun Kang, Romain Laroche, Xindi Yuan, Adam Trischler, Xue Liu, and Jie Fu. Think before you act: Decision transformers with internal working memory. *arXiv preprint arXiv:2305.16338*, 2023.

[287] Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, et al. Check your facts and try again: Improving large language models with external knowledge and automated feedback. *arXiv preprint arXiv:2302.12813*, 2023.

[288] Taewoon Kim, Michael Cochez, Vincent François-Lavet, Mark Neerincx, and Piek Vossen. A machine with short-term, episodic, and semantic memory systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 48–56, 2023.

[289] Weiran Yao, Shelby Heinecke, Juan Carlos Niebles, Zhiwei Liu, Yihao Feng, Le Xue, Rithesh Murthy, Zeyuan Chen, Jianguo Zhang, Devansh Arpit, et al. Retroformer: Retrospective large language agents with policy gradient optimization. *arXiv preprint arXiv:2308.02151*, 2023.

[290] Siyuan Wang, Zhongyu Wei, Yejin Choi, and Xiang Ren. Symbolic working memory enhances language models for complex rule application. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17583–17604, 2024.

[291] Longtao Zheng, Rundong Wang, and Bo An. Synapse: Leveraging few-shot exemplars for human-level computer control. *arXiv preprint arXiv:2306.07863*, 2023.

[292] Krishan Rana, Jesse Haviland, Sourav Garg, Jad Abou-Chakra, Ian D Reid, and Niko Suenderhauf. Sayplan: Grounding large language models using 3d scene graphs for scalable task planning. *CoRR*, 2023.

[293] Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu 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*, pages 2998–3009, 2023.

[294] Yao Fu, Dong-Ki Kim, Jaekyeom Kim, Sungryull Sohn, Lajanugen Logeswaran, Kyunghoon Bae, and Honglak Lee. Autoguide: Automated generation and selection of state-aware guidelines for large language model agents. *arXiv preprint arXiv:2403.08978*, 2024.

[295] Wenqi Zhang, Ke Tang, Hai Wu, Mengna Wang, Yongliang Shen, Guiyang Hou, Zeqi Tan, Peng Li, Yueting Zhuang, and Weiming Lu. Agent-pro: Learning to evolve via policy-level reflection and optimization. *arXiv preprint arXiv:2402.17574*, 2024.

[296] Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. Gpt-4v (ision) is a generalist web agent, if grounded. In *ICML*, 2024.

[297] Hanyu Lai, Xiao Liu, Iat Long Iong, Shuntian Yao, Yuxuan Chen, Pengbo Shen, Hao Yu, Hanchen Zhang, Xiaohan Zhang, Yuxiao Dong, et al. Autowebglm: A large language model-based web navigating agent. In *KDD*, 2024.

[298] Paloma Sodhi, SRK Branavan, and Ryan McDonald. Heap: Hierarchical policies for web actions using LLMs. *arXiv preprint arXiv:2310.03720*, 2023.

[299] Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. *arXiv preprint arXiv:2409.07429*, 2024.

[300] Xinyuan Wang, Chenxi Li, Zhen Wang, Fan Bai, Haotian Luo, Jiayou Zhang, Nebojsa Jojic, Eric P Xing, and Zhiting Hu. PromptAgent: Strategic planning with language models enables expert-level prompt optimization. *arXiv preprint arXiv:2310.16427*, 2023.

[301] Chen Qian, Yufan Dang, Jiahao Li, Wei Liu, Zihao Xie, Yifei Wang, Weize Chen, Cheng Yang, Xin Cong, Xiaoyin Che, et al. Experiential co-learning of software-developing agents. *arXiv preprint arXiv:2312.17025*, 2023.

[302] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. *arXiv preprint arXiv:2311.12871*, 2023.

[303] Chen Qian, Jiahao Li, Yufan Dang, Wei Liu, YiFei Wang, Zihao Xie, Weize Chen, Cheng Yang, Yingli Zhang, Zhiyuan Liu, et al. Iterative experience refinement of software-developing agents. *arXiv preprint arXiv:2405.04219*, 2024.

[304] Shreyas Basavatia, Keerthiram Murugesan, and Shivam Ratnakar. Starling: Self-supervised training of text-based reinforcement learning agent with large language models. *arXiv preprint arXiv:2406.05872*, 2024.

[305] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40:e253, 2017.

[306] Anirudh Goyal, Riashat Islam, Daniel Strouse, Zafarali Ahmed, Matthew Botvinick, Hugo Larochelle, Yoshua Bengio, and Sergey Levine. Infobot: Transfer and exploration via the information bottleneck. *arXiv preprint arXiv:1901.10902*, 2019.

[307] Shilong Li, Yancheng He, Hangyu Guo, Xingyuan Bu, Ge Bai, Jie Liu, Jiaheng Liu, Xingwei Qu, Yangguang Li, Wanli Ouyang, et al. Graphreader: Building graph-based agent to enhance long-context abilities of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 12758–12786, 2024.

[308] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[309] Grace W Lindsay. Attention in psychology, neuroscience, and machine learning. *Frontiers in computational neuroscience*, 14:29, 2020.

[310] Arsha Nagrani, Shan Yang, Anurag Arnab, Aren Jansen, Cordelia Schmid, and Chen Sun. Attention bottlenecks for multimodal fusion. *Advances in neural information processing systems*, 34:14200–14213, 2021.

[311] Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[312] Yuheng Cheng, Huan Zhao, Xiyuan Zhou, Junhua Zhao, Yuji Cao, Chao Yang, and Xinlei Cai. A large language model for advanced power dispatch. *Scientific Reports*, 15(1):8925, 2025.

[313] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[314] Larry R Squire, Lisa Genzel, John T Wixted, and Richard G Morris. Memory consolidation. *Cold Spring Harbor perspectives in biology*, 7(8):a021766, 2015.

[315] Yuji Cao, Huan Zhao, Yuheng Cheng, Ting Shu, Yue Chen, Guolong Liu, Gaoqi Liang, Junhua Zhao, Jinyue Yan, and Yun Li. Survey on large language model-enhanced reinforcement learning: Concept, taxonomy, and methods. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[316] N Reimers. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[317] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[318] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

[319] Wen Li, Ying Zhang, Yifang Sun, Wei Wang, Mingjie Li, Wenjie Zhang, and Xuemin Lin. Approximate nearest neighbor search on high dimensional data—experiments, analyses, and improvement. *IEEE Transactions on Knowledge and Data Engineering*, 32(8):1475–1488, 2019.

[320] Peiyan Zhang, Chaozhuo Li, Liying Kang, Feiran Huang, Senzhang Wang, Xing Xie, and Sunghun Kim. High-frequency-aware hierarchical contrastive selective coding for representation learning on text attributed graphs. In *Proceedings of the ACM Web Conference 2024*, pages 4316–4327, 2024.

[321] Zhenyi Wang, Enneng Yang, Li Shen, and Heng Huang. A comprehensive survey of forgetting in deep learning beyond continual learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[322] Bart Kosko. Bidirectional associative memories. *IEEE Transactions on Systems, man, and Cybernetics*, 18(1): 49–60, 1988.

[323] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850. PMLR, 2016.

[324] Zihang Dai. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

[325] Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. Transformer-patcher: One mistake worth one neuron. *arXiv preprint arXiv:2301.09785*, 2023.

[326] Govind Krishnan Gangadhar and Karl Stratos. Model editing by standard fine-tuning. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 5907–5913, 2024.

[327] Peiyan Zhang, Yuchen Yan, Chaozhuo Li, Senzhang Wang, Xing Xie, Guojie Song, and Sunghun Kim. Continual learning on dynamic graphs via parameter isolation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 601–611, 2023.

[328] Yu Wang, Ruihan Wu, Zexue He, Xiusi Chen, and Julian McAuley. Large scale knowledge washing. *arXiv preprint arXiv:2405.16720*, 2024.

[329] Wuyang Chen, Yanqi Zhou, Nan Du, Yanping Huang, James Laudon, Zhifeng Chen, and Claire Cui. Lifelong language pretraining with distribution-specialized experts. In *International Conference on Machine Learning*, pages 5383–5395. PMLR, 2023.

[330] Yinpeng Chen, DeLesley Hutchins, Aren Jansen, Andrey Zhmoginov, David Racz, and Jesper Andersen. Melodi: Exploring memory compression for long contexts. *arXiv preprint arXiv:2410.03156*, 2024.

[331] Jihoon Tack, Jaehyung Kim, Eric Mitchell, Jinwoo Shin, Yee Whye Teh, and Jonathan Richard Schwarz. Online adaptation of language models with a memory of amortized contexts. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=RIfgKCknTu.

[332] Yu Wang, Dmitry Krotov, Yuanzhe Hu, Yifan Gao, Wangchunshu Zhou, Julian McAuley, Dan Gutfreund, Rogerio Feris, and Zexue He. M+: Extending memoryllm with scalable long-term memory. *arXiv preprint arXiv:2502.00592*, 2025.

[333] Shankar Padmanabhan, Yasumasa Onoe, Michael Zhang, Greg Durrett, and Eunsol Choi. Propagating knowledge updates to lms through distillation. *Advances in Neural Information Processing Systems*, 36:47124–47142, 2023.

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

[335] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[336] Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. Llmlingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13358–13376, 2023.

[337] Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677, 2024.

[338] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, 2019.

[339] Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

[340] Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630, 2024.

[341] Edward C Tolman. Cognitive maps in rats and men. *Psychological review*, 55(4):189, 1948.

[342] Kenneth James Williams Craik. *The nature of explanation*, volume 445. CUP Archive, 1967.

[343] Dedre Gentner and Albert L Stevens. *Mental models*. Psychology Press, 2014.

[344] Andy Clark. *Surfing uncertainty: Prediction, action, and the embodied mind*. Oxford University Press, 2015.

[345] Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2 (4):160–163, 1991.

[346] Jürgen Schmidhuber. *Making the world differentiable: on using self supervised fully recurrent neural networks for dynamic reinforcement learning and planning in non-stationary environments*, volume 126. Inst. für Informatik, 1990.

[347] Jürgen Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In *1st International Conference on Simulation of Adaptive Behavior on From Animals to Animats*, page 222–227, Cambridge, MA, USA, 1991. MIT Press. ISBN 0262631385.

[348] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[349] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering Atari, Go, Chess and Shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[350] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

[351] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11097–11107, 2020.

[352] Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *Conference on Robot Learning*, pages 2226–2240. PMLR, 2023.

[353] Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. *arXiv preprint arXiv:2405.12399*, 2024.

[354] SM Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S Morcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

[355] Marc Deisenroth and Carl E Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472, 2011.

[356] Chenxi Liu, Yongqiang Chen, Tongliang Liu, Mingming Gong, James Cheng, Bo Han, and Kun Zhang. Discovery of the hidden world with large language models. *arXiv preprint arXiv:2402.03941*, 2024.

[357] Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. GR-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

[358] Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024.

[359] Haochen Shi, Huazhe Xu, Zhiao Huang, Yunzhu Li, and Jiajun Wu. Robocraft: Learning to see, simulate, and shape elasto-plastic objects in 3d with graph networks. *The International Journal of Robotics Research*, 43(4): 533–549, 2024.

[360] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Advances in neural information processing systems*, 35:23192–23204, 2022.

[361] Ganlong Zhao, Guanbin Li, Weikai Chen, and Yizhou Yu. Over-nav: Elevating iterative vision-and-language navigation with open-vocabulary detection and structured representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16296–16306, 2024.

[362] Basil Kouvaritakis and Mark Cannon. Model predictive control. *Switzerland: Springer International Publishing*, 38(13-56):7, 2016.

[363] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 7559–7566. IEEE, 2018.

[364] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

[365] Boyan Li, Jiayi Zhang, Ju Fan, Yanwei Xu, Chong Chen, Nan Tang, and Yuyu Luo. Alpha-sql: Zero-shot text-to-sql using monte carlo tree search. *CoRR*, abs/2502.17248, 2025.

[366] Allen Newell. *Unified theories of cognition*. Harvard University Press, 1994.

[367] Pierre Harvey Richemond, Yunhao Tang, Daniel Guo, Daniele Calandriello, Mohammad Gheshlaghi Azar, Rafael Rafailov, Bernardo Avila Pires, Eugene Tarassov, Lucas Spangher, Will Ellsworth, et al. Offline regularised reinforcement learning for large language models alignment. *arXiv preprint arXiv:2405.19107*, 2024.

[368] Dahyun Kim, Yungi Kim, Wonho Song, Hyeonwoo Kim, Yunsu Kim, Sanghoon Kim, and Chanjun Park. sdpo: Don't use your data all at once. *arXiv preprint arXiv:2403.19270*, 2024.

[369] Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pages 4447–4455. PMLR, 2024.

[370] Junkang Wu, Yuexiang Xie, Zhengyi Yang, Jiancan Wu, Jinyang Gao, Bolin Ding, Xiang Wang, and Xiangnan He. β-dpo: Direct preference optimization with dynamic β, 2024. URL https://arxiv.org/abs/2407.08639.

[371] Jiwoo Hong, Noah Lee, and James Thorne. Orpo: Monolithic preference optimization without reference model. *arXiv preprint arXiv:2403.07691*, 2024.

[372] Corby Rosset, Ching-An Cheng, Arindam Mitra, Michael Santacroce, Ahmed Awadallah, and Tengyang Xie. Direct nash optimization: Teaching language models to self-improve with general preferences. *arXiv preprint arXiv:2404.03715*, 2024.

[373] Chaoqi Wang, Yibo Jiang, Chenghao Yang, Han Liu, and Yuxin Chen. Beyond reverse kl: Generalizing direct preference optimization with diverse divergence constraints. *arXiv preprint arXiv:2309.16240*, 2023.

[374] Jing Xu, Andrew Lee, Sainbayar Sukhbaatar, and Jason Weston. Some things are more cringe than others: Iterative preference optimization with the pairwise cringe loss. *arXiv preprint arXiv:2312.16682*, 2023.

[375] Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. From r to q^*: Your language model is secretly a q-function. *arXiv preprint arXiv:2404.12358*, 2024.

[376] Shiva Kumar Pentyala, Zhichao Wang, Bin Bi, Kiran Ramnath, Xiang-Bo Mao, Regunathan Radhakrishnan, Sitaram Asur, et al. Paft: A parallel training paradigm for effective llm fine-tuning. *arXiv preprint arXiv:2406.17923*, 2024.

[377] Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2025.

[378] Tianqi Liu, Zhen Qin, Junru Wu, Jiaming Shen, Misha Khalman, Rishabh Joshi, Yao Zhao, Mohammad Saleh, Simon Baumgartner, Jialu Liu, et al. Lipo: Listwise preference optimization through learning-to-rank. *arXiv preprint arXiv:2402.01878*, 2024.

[379] Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. Rrhf: Rank responses to align language models with human feedback without tears. *arXiv preprint arXiv:2304.05302*, 2023.

[380] Feifan Song, Bowen Yu, Minghao Li, Haiyang Yu, Fei Huang, Yongbin Li, and Houfeng Wang. Preference ranking optimization for human alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18990–18998, 2024.

[381] Shitong Duan, Xiaoyuan Yi, Peng Zhang, Yan Liu, Zheng Liu, Tun Lu, Xing Xie, and Ning Gu. Negating negatives: Alignment with human negative samples via distributional dispreference optimization. *arXiv preprint arXiv:2403.03419*, 2024.

[382] Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. *arXiv preprint arXiv:2404.05868*, 2024.

[383] Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint arXiv:2402.14740*, 2024.

[384] Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. Contrastive preference optimization: Pushing the boundaries of llm performance in machine translation. *arXiv preprint arXiv:2401.08417*, 2024.

[385] Rémi Munos, Michal Valko, Daniele Calandriello, Mohammad Gheshlaghi Azar, Mark Rowland, Zhaohan Daniel Guo, Yunhao Tang, Matthieu Geist, Thomas Mesnard, Andrea Michi, et al. Nash learning from human feedback. *arXiv preprint arXiv:2312.00886*, 18, 2023.

[386] Gokul Swamy, Christoph Dann, Rahul Kidambi, Zhiwei Steven Wu, and Alekh Agarwal. A minimaximalist approach to reinforcement learning from human feedback. *arXiv preprint arXiv:2401.04056*, 2024.

[387] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017.

[388] Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In *International conference on machine learning*, pages 5062–5071. PMLR, 2019.

[389] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International conference on machine learning*, pages 8583–8592. PMLR, 2020.

[390] Yali Du, Lei Han, Meng Fang, Ji Liu, Tianhong Dai, and Dacheng Tao. Liir: Learning individual intrinsic reward in multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[391] Cédric Colas, Pierre Fournier, Mohamed Chetouani, Olivier Sigaud, and Pierre-Yves Oudeyer. Curious: intrinsically motivated modular multi-goal reinforcement learning. In *International conference on machine learning*, pages 1331–1340. PMLR, 2019.

[392] Vitchyr H Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning. *arXiv preprint arXiv:1903.03698*, 2019.

[393] Ali Hassani, Amir Iranmanesh, Mahdi Eftekhari, and Abbas Salemi. Discern: diversity-based selection of centroids for k-estimation and rapid non-stochastic clustering. *International Journal of Machine Learning and Cybernetics*, 12:635–649, 2021.

[394] Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

[395] Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024.

[396] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

[397] Jean-Francois Ton, Muhammad Faaiz Taufiq, and Yang Liu. Understanding chain-of-thought in LLMs through information theory. *arXiv preprint arXiv:2411.11984*, 2024.

[398] Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. *Advances in neural information processing systems*, 29, 2016.

[399] Hyoungseok Kim, Jaekyeom Kim, Yeonwoo Jeong, Sergey Levine, and Hyun Oh Song. Emi: Exploration with mutual information. *arXiv preprint arXiv:1810.01176*, 2018.

[400] Pranav Shyam, Wojciech Jaśkowski, and Faustino Gomez. Model-based active exploration. In *International conference on machine learning*, pages 5779–5788. PMLR, 2019.

[401] Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, et al. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*, 2023.

[402] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

[403] Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. *arXiv preprint arXiv:2312.11456*, 2023.

[404] Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. Rlhf workflow: From reward modeling to online rlhf. *arXiv preprint arXiv:2405.07863*, 2024.

[405] Yongcheng Zeng, Guoqing Liu, Weiyu Ma, Ning Yang, Haifeng Zhang, and Jun Wang. Token-level direct preference optimization. *arXiv preprint arXiv:2404.11999*, 2024.

[406] Robert G Lewis, Ermanno Florio, Daniela Punzo, and Emiliana Borrelli. *The Brain's reward system in health and disease*. Springer, 2021.

[407] Marc Fakhoury. *The Brain Reward System*. Springer, 2021.

[408] Vincent Breton-Provencher and Mriganka Sur. Active control of arousal by a locus coeruleus gabaergic circuit. *Nature neuroscience*, 22(2):218–228, 2019.

[409] Jia Qi, Shiliang Zhang, Hui-Ling Wang, Huikun Wang, Jose de Jesus Aceves Buendia, Alexander F Hoffman, Carl R Lupica, Rebecca P Seal, and Marisela Morales. A glutamatergic reward input from the dorsal raphe to ventral tegmental area dopamine neurons. *Nature communications*, 5(1):5390, 2014.

[410] Melissa J Sharpe, Nathan J Marchant, Leslie R Whitaker, Christopher T Richie, Yajun J Zhang, Erin J Campbell, Pyry P Koivula, Julie C Necarsulmer, Carlos Mejias-Aponte, Marisela Morales, et al. Lateral hypothalamic gabaergic neurons encode reward predictions that are relayed to the ventral tegmental area to regulate learning. *Current Biology*, 27(14):2089–2100, 2017.

[411] MSD Manual. Neurotransmission, 2022. URL https://www.msdmanuals.cn/professional/neurologic-disorders/neurotransmission/neurotransmission. Accessed: 2022-04-01.

[412] Anil Ananthaswamy. How close is AI to human-level intelligence? *Nature*, 636(8041):22–25, 2024.

[413] Eric G Ceballos, Asa Farahani, Zhen-Qi Liu, Filip Milisav, Justine Y Hansen, Alain Dagher, and Bratislav Misic. Mapping neuropeptide sigaling in the human brain. *bioRxiv*, pages 2024–12, 2024.

[414] Jinghan Zhang, Xiting Wang, Yiqiao Jin, Changyu Chen, Xinhao Zhang, and Kunpeng Liu. Prototypical reward network for data-efficient rlhf. In *ACL*, 2024.

[415] Sebastian Thrun and Michael L Littman. Reinforcement learning: An introduction. *AI Magazine*, 21(1): 103–103, 2000.

[416] Leonard Adolphs, Tianyu Gao, Jing Xu, Kurt Shuster, Sainbayar Sukhbaatar, and Jason Weston. The cringe loss: Learning what language not to model. *arXiv preprint arXiv:2211.05826*, 2022.

[417] Luiz Pessoa. Multiple influences of reward on perception and attention. *Visual cognition*, 23(1-2):272–290, 2015.

[418] Han-Xiao Li, Quan-Shan Long, An-Tao Chen, and Qing Li. The influence of reward motivation on emotion regulation. *Sheng li xue bao:[Acta Physiologica Sinica]*, 71(4):562–574, 2019.

[419] Ewa A Miendlarzewska, Daphne Bavelier, and Sophie Schwartz. Influence of reward motivation on human declarative memory. *Neuroscience & Biobehavioral Reviews*, 61:156–176, 2016.

[420] Marvin Lee Minsky, editor. *The Emotion Machine: Commensense Thinking, Artificial Intelligence, and the Future of the Human Mind*. Simon & Schuster, 2006.

[421] Paul Ekman. An argument for basic emotions. *Cognition & Emotion*, 6:169–200, 1992.

[422] Cheng Li, Jindong Wang, Yixuan Zhang, Kaijie Zhu, Wenxin Hou, Jianxun Lian, Fang Luo, Qiang Yang, and Xing Xie. Large language models understand and can be enhanced by emotional stimuli. *arXiv preprint arXiv: 2307.11760*, 2023.

[423] Xuena Wang, Xueting Li, Zi Yin, Yue Wu, and Jia Liu. Emotional intelligence of large language models. *Journal of Pacific Rim Psychology*, 17:18344909231213958, 2023.

[424] Lisa Feldman Barrett. The theory of constructed emotion: an active inference account of interoception and categorization. *Social Cognitive and Affective Neuroscience*, 12:1833 – 1833, 2017.

[425] Rachael E. Jack, Oliver G. B. Garrod, Hui Yu, Roberto Caldara, and Philippe G. Schyns. Facial expressions of emotion are not culturally universal. *Proceedings of the National Academy of Sciences*, 109(19): 7241–7244, 2012. doi:10.1073/pnas.1200155109. URL https://www.pnas.org/doi/abs/10.1073/pnas.1200155109.

[426] James Russell. A circumplex model of affect. *Journal of Personality and Social Psychology*, 39:1161–1178, 12 1980. doi:10.1037/h0077714.

[427] Albert Mehrabian. Pleasure-arousal-dominance: A general framework for describing and measuring individual differences in temperament. *Current Psychology*, 14:261–292, 1996.

[428] Zhenyi Lu, Wei Wei, Xiaoye Qu, Xian-Ling Mao, Dangyang Chen, and Jixiong Chen. Miracle: Towards personalized dialogue generation with latent-space multiple personal attribute control. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5933–5957, Singapore, December 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.findings-emnlp.395. URL https://aclanthology.org/2023.findings-emnlp.395/.

[429] Ala N. Tak and Jonathan Gratch. Is gpt a computational model of emotion? detailed analysis. *arXiv preprint arXiv: 2307.13779*, 2023.

[430] Shudong Liu, Yiqiao Jin, Cheng Li, Derek F Wong, Qingsong Wen, Lichao Sun, Haipeng Chen, Xing Xie, and Jindong Wang. Culturevlm: Characterizing and improving cultural understanding of vision-language models for over 100 countries. *arXiv:2501.01282*, 2025.

[431] Robert Plutchik. A general psychoevolutionary theory of emotion. *Theories of emotion*, 1:3–31, 1980.

[432] Klaus R. Scherer. The dynamic architecture of emotion: Evidence for the component process model. *Cognition and Emotion*, 23:1307 – 1351, 2009.

[433] Andrew Ortony, Gerald L Clore, and Allan Collins. *The cognitive structure of emotions*. Cambridge university press, 2022.

[434] Eva Hudlicka. Computational modeling of cognition-emotion interactions: Relevance to mechanisms of affective disorders and therapeutic action. *Cognitive Science*, 36, 2014.

[435] Stacy Marsella and J. Gratch. Computationally modeling human emotion. *Commun. ACM*, 57:56–67, 2014.

[436] Hao Fei, Bobo Li, Qian Liu, Lidong Bing, Fei Li, and Tat seng Chua. Reasoning implicit sentiment with chain-of-thought prompting. *Annual Meeting of the Association for Computational Linguistics*, 2023. doi:10.48550/arXiv.2305.11255.

[437] Xiaofei Sun, Xiaoya Li, Shengyu Zhang, Shuhe Wang, Fei Wu, Jiwei Li, Tianwei Zhang, and Guoyin Wang. Sentiment analysis through LLM negotiations. *arXiv preprint arXiv: 2311.01876*, 2023.

[438] Adam S Lowet, Qiao Zheng, Melissa Meng, Sara Matias, Jan Drugowitsch, and Naoshige Uchida. An opponent striatal circuit for distributional reinforcement learning. *Nature*, pages 1–10, 2025.

[439] Xin Hong, Yuan Gong, Vidhyasaharan Sethu, and Ting Dang. Aer-llm: Ambiguity-aware emotion recognition leveraging large language models. *arXiv preprint arXiv: 2409.18339*, 2024.

[440] Zebang Cheng, Zhi-Qi Cheng, Jun-Yan He, Kai Wang, Yuxiang Lin, Zheng Lian, Xiaojiang Peng, and Alexander Hauptmann. Emotion-LLaMA: Multimodal emotion recognition and reasoning with instruction tuning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 110805–110853. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/c7f43ada17acc234f568dc66da527418-Paper-Conference.pdf.

[441] Sahand Sabour, Siyang Liu, Zheyuan Zhang, June Liu, Jinfeng Zhou, Alvionna Sunaryo, Tatia Lee, Rada Mihalcea, and Minlie Huang. EmoBench: Evaluating the emotional intelligence of large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5986–6004, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.acl-long.326. URL https://aclanthology.org/2024.acl-long.326/.

[442] Wenbin Wang, Liang Ding, Li Shen, Yong Luo, Han Hu, and Dacheng Tao. Wisdom: Improving multimodal sentiment analysis by fusing contextual world knowledge. In *Proceedings of the 32nd ACM International Conference on Multimedia*, MM '24, page 2282–2291, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400706868. doi:10.1145/3664647.3681403. URL https://doi.org/10.1145/3664647.3681403.

[443] Jinyang Wu, Mingkuan Feng, Shuai Zhang, Feihu Che, Zengqi Wen, and Jianhua Tao. Beyond examples: High-level automated reasoning paradigm in in-context learning via mcts. *arXiv preprint arXiv:2411.18478*, 2024.

[444] Zheng Lian, Haiyang Sun, Licai Sun, Hao Gu, Zhuofan Wen, Siyuan Zhang, Shun Chen, Mingyu Xu, Ke Xu, Kang Chen, Lan Chen, Shan Liang, Ya Li, Jiangyan Yi, Bin Liu, and Jianhua Tao. Explainable multimodal emotion recognition. *arXiv preprint arXiv: 2306.15401*, 2023.

[445] Shanglin Lei, Guanting Dong, Xiaoping Wang, Keheng Wang, Runqi Qiao, and Sirui Wang. Instructerc: Reforming emotion recognition in conversation with multi-task retrieval-augmented large language models. *arXiv preprint arXiv: 2309.11911*, 2023.

[446] Zheng Lian, Licai Sun, Haiyang Sun, Kang Chen, Zhuofan Wen, Hao Gu, Bin Liu, and Jianhua Tao. GPT-4V with emotion: A zero-shot benchmark for generalized emotion recognition. *Inf. Fusion*, 108:102367, 2024. doi:10.1016/J.INFFUS.2024.102367. URL https://doi.org/10.1016/j.inffus.2024.102367.

[447] Yiqiao Jin, Minje Choi, Gaurav Verma, Jindong Wang, and Srijan Kumar. Mm-soc: Benchmarking multimodal large language models in social media platforms. In *ACL*, 2024.

[448] William Stigall, Md Abdullah Al Hafiz Khan, Dinesh Attota, Francis Nweke, and Yong Pei. Large language models performance comparison of emotion and sentiment classification. In *Proceedings of the 2024 ACM Southeast Conference*, ACMSE '24, page 60–68, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400702372. doi:10.1145/3603287.3651183. URL https://doi.org/10.1145/3603287.3651183.

[449] Steve Rathje, Dan-Mircea Mirea, Ilia Sucholutsky, Raja Marjieh, Claire E. Robertson, and Jay Joseph Van Bavel. Gpt is an effective tool for multilingual psychological text analysis. *Proceedings of the National Academy of Sciences of the United States of America*, 121, 2024.

[450] Minxue Niu, Mimansa Jaiswal, and E. Provost. From text to emotion: Unveiling the emotion annotation capabilities of llms. *INTERSPEECH*, 2024. doi:10.21437/interspeech.2024-2282.

[451] Haiquan Zhao, Lingyu Li, Shisong Chen, Shuqi Kong, Jiaan Wang, Kexin Huang, Tianle Gu, Yixu Wang, Wang Jian, Dandan Liang, Zhixu Li, Yan Teng, Yanghua Xiao, and Yingchun Wang. Esc-eval: Evaluating emotion support conversations in large language models. *arXiv preprint arXiv: 2406.14952*, 2024.

[452] Yingjie Zhou, Zicheng Zhang, Jiezhang Cao, Jun Jia, Yanwei Jiang, Farong Wen, Xiaohong Liu, Xiongkuo Min, and Guangtao Zhai. Memo-bench: A multiple benchmark for text-to-image and multimodal large language models on human emotion analysis. *arXiv preprint arXiv: 2411.11235*, 2024.

[453] Zheng Lian, Licai Sun, Yong Ren, Hao Gu, Haiyang Sun, Lan Chen, Bin Liu, and Jianhua Tao. Merbench: A unified evaluation benchmark for multimodal emotion recognition. *arXiv preprint arXiv: 2401.03429*, 2024.

[454] Mostafa M. Amin, Rui Mao, Erik Cambria, and Björn W. Schuller. A wide evaluation of chatgpt on affective computing tasks. *IEEE Trans. Affect. Comput.*, 15(4):2204–2212, 2024. doi:10.1109/TAFFC.2024.3419593. URL https://doi.org/10.1109/TAFFC.2024.3419593.

[455] Weixiang Zhao, Yanyan Zhao, Xin Lu, Shilong Wang, Yanpeng Tong, and Bing Qin. Is chatgpt equipped with emotional dialogue capabilities? *arXiv preprint arXiv: 2304.09582*, 2023.

[456] Tom Sühr, Florian E. Dorner, Samira Samadi, and Augustin Kelava. Challenging the validity of personality tests for large language models. *arXiv preprint arXiv: 2311.05297*, 2023.

[457] Nikolay B Petrov, Gregory Serapio-García, and Jason Rentfrow. Limited ability of LLMs to simulate human psychological behaviours: a psychometric analysis. *arXiv preprint arXiv: 2405.07248*, 2024.

[458] Jen tse Huang, Wenxuan Wang, Eric John Li, Man Ho LAM, Shujie Ren, Youliang Yuan, Wenxiang Jiao, Zhaopeng Tu, and Michael Lyu. On the humanity of conversational AI: Evaluating the psychological portrayal of LLMs. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=H3UayAQWoE.

[459] Jen tse Huang, Wenxiang Jiao, Man Ho Lam, Eric John Li, Wenxuan Wang, and Michael R. Lyu. Revisiting the reliability of psychological scales on large language models. *arXiv preprint arXiv: 2305.19926*, 2023.

[460] Yiming Ai, Zhiwei He, Ziyin Zhang, Wenhong Zhu, Hongkun Hao, Kai Yu, Lingjun Chen, and Rui Wang. Is cognition and action consistent or not: Investigating large language model's personality. *arXiv preprint arXiv: 2402.14679*, 2024.

[461] Xintao Wang, Yunze Xiao, Jen-tse Huang, Siyu Yuan, Rui Xu, Haoran Guo, Quan Tu, Yaying Fei, Ziang Leng, Wei Wang, Jiangjie Chen, Cheng Li, and Yanghua Xiao. Incharacter: Evaluating personality fidelity in role-playing agents through psychological interviews. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 1840–1873. Association for Computational Linguistics, 2024. doi:10.18653/V1/2024.ACL-LONG.102. URL https://doi.org/10.18653/v1/2024.acl-long.102.

[462] Marcel Binz and Eric Schulz. Turning large language models into cognitive models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=eiC4BKypf1.

[463] Thilo Hagendorff, Ishita Dasgupta, Marcel Binz, Stephanie C. Y. Chan, Andrew Lampinen, Jane X. Wang, Zeynep Akata, and Eric Schulz. Machine psychology. *arXiv preprint arXiv: 2303.13988*, 2023.

[464] Julian Coda-Forno, Marcel Binz, Jane X. Wang, and Eric Schulz. Cogbench: a large language model walks into a psychology lab. *International Conference on Machine Learning*, 2024. doi:10.48550/arXiv.2402.18225.

[465] Jesse Roberts, Kyle Moore, Drew Wilenzick, and Doug Fisher. Using artificial populations to study psychological phenomena in neural models. *AAAI Conference on Artificial Intelligence*, 2023. doi:10.1609/aaai.v38i17.29856.

[466] Maor Reuben, Ortal Slobodin, Aviad Elyshar, Idan-Chaim Cohen, Orna Braun-Lewensohn, Odeya Cohen, and Rami Puzis. Assessment and manipulation of latent constructs in pre-trained language models using psychometric scales. *arXiv preprint arXiv: 2409.19655*, 2024.

[467] Jen tse Huang, Man Ho LAM, Eric John Li, Shujie Ren, Wenxuan Wang, Wenxiang Jiao, Zhaopeng Tu, and Michael Lyu. Apathetic or empathetic? evaluating LLMs' emotional alignments with humans. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=pwRVGRWtGg.

[468] Bo Zhao, Maya Okawa, Eric J Bigelow, Rose Yu, Tomer Ullman, and Hidenori Tanaka. Emergence of hierarchical emotion representations in large language models, 2025. URL https://openreview.net/forum?id=wTm4W39GdD.

[469] Fiona Anting Tan, Gerard Christopher Yeo, Kokil Jaidka, Fanyou Wu, Weijie Xu, Vinija Jain, Aman Chadha, Yang Liu, and See-Kiong Ng. Phantom: Persona-based prompting has an effect on theory-of-mind reasoning in large language models. *arXiv preprint arXiv: 2403.02246*, 2024.

[470] Hang Jiang, Xiajie Zhang, Xubo Cao, Cynthia Breazeal, Deb Roy, and Jad Kabbara. PersonaLLM: Investigating the ability of large language models to express personality traits. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3605–3627, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.findings-naacl.229. URL https://aclanthology.org/2024.findings-naacl.229/.

[471] Leonard Salewski, Stephan Alaniz, Isabel Rio-Torto, Eric Schulz, and Zeynep Akata. In-context impersonation reveals large language models' strengths and biases. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36*, 2023.

[472] Lucio La Cava and Andrea Tagarelli. Open models, closed minds? on agents capabilities in mimicking human personalities through open large language models. *arXiv preprint arXiv: 2401.07115*, 2024.

[473] Navya Jain, Zekun Wu, Cristian Munoz, Airlie Hilliard, Adriano Koshiyama, Emre Kazim, and Philip Treleaven. From text to emoji: How peft-driven personality manipulation unleashes the emoji potential in llms. *arXiv preprint arXiv: 2409.10245*, 2024.

[474] Jen-tse Huang, Wenxiang Jiao, Man Ho Lam, Eric John Li, Wenxuan Wang, and Michael Lyu. On the reliability of psychological scales on large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6152–6173, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.emnlp-main.354. URL https://aclanthology.org/2024.emnlp-main.354/.

[475] Jia Deng, Tianyi Tang, Yanbin Yin, Wenhao Yang, Wayne Xin Zhao, and Ji-Rong Wen. Neuron-based personality trait induction in large language models. *arXiv preprint arXiv: 2410.12327*, 2024.

[476] Lena Podoletz. We have to talk about emotional AI and crime. *AI & SOCIETY*, 38(3):1067–1082, 2023.

[477] Author Name(s). Emotional ai: Privacy, manipulation, and bias risks, 2024. URL https://businesslawtoday.org/2024/09/emotional-ai-privacy-manipulation-bias-risks/. Accessed January 18, 2025.

[478] Author Name(s). Emotional artificial intelligence: Risks and opportunities, 2024. URL https://www.linkedin.com/pulse/emotional-artificial-intelligence-risks-opportunities-vincent-mba-e2rre/. Accessed January 18, 2025.

[479] Julian Coda-Forno, Kristin Witte, Akshay K. Jagadish, Marcel Binz, Zeynep Akata, and Eric Schulz. Inducing anxiety in large language models can induce bias, 2024. URL https://arxiv.org/abs/2304.11111.

[480] Yiqiao Jin, Mohit Chandra, Gaurav Verma, Yibo Hu, Munmun De Choudhury, and Srijan Kumar. Better to ask in english: Cross-lingual evaluation of large language models for healthcare queries. In *Web Conference*, pages 2627–2638, 2024.

[481] Peter Mantello and Manh-Tung Ho. Emotional AI and the future of wellbeing in the post-pandemic workplace. *AI & society*, 39(4):1883–1889, 2024.

[482] Corina Pelau, Dan-Cristian Dabija, and Irina Ene. What makes an AI device human-like? the role of interaction quality, empathy and perceived psychological anthropomorphic characteristics in the acceptance of artificial intelligence in the service industry. *Computers in Human Behavior*, 122:106855, 2021.

[483] Jay Ratican and James Hutson. The six emotional dimension (6de) model: A multidimensional approach to analyzing human emotions and unlocking the potential of emotionally intelligent artificial intelligence (ai) via large language models (llm). *Journal of Artificial Intelligence and Robotics*, 1(1), 2023.

[484] Baihan Lin, Djallel Bouneffouf, Guillermo Cecchi, and Kush R Varshney. Towards healthy ai: large language models need therapists too. *arXiv preprint arXiv:2308.04434*, 2023.

[485] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[486] Yinhan Liu. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364, 2019.

[487] Z Lan. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[488] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[489] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[490] Tianhe Ren, Qing Jiang, Shilong Liu, Zhaoyang Zeng, Wenlong Liu, Han Gao, Hongjie Huang, Zhengyu Ma, Xiaoke Jiang, Yihao Chen, et al. Grounding dino 1.5: Advance the "edge" of open-set object detection. *arXiv preprint arXiv:2405.10300*, 2024.

[491] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021.

[492] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35:10078–10093, 2022.

[493] Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech 2: Fast and high-quality end-to-end text to speech. *arXiv preprint arXiv:2006.04558*, 2020.

[494] Loïc Barrault, Yu-An Chung, Mariano Coria Meglioli, David Dale, Ning Dong, Mark Duppenthaler, Paul-Ambroise Duquenne, Brian Ellis, Hady Elsahar, Justin Haaheim, et al. Seamless: Multilingual expressive and streaming speech translation. *arXiv preprint arXiv:2312.05187*, 2023.

[495] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33: 12449–12460, 2020.

[496] Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671*, 2023.

[497] Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu, Ce Liu, Michael Zeng, and Lijuan Wang. Mm-react: Prompting chatgpt for multimodal reasoning and action. *arXiv preprint arXiv:2303.11381*, 2023.

[498] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11888–11898, 2023.

[499] Rongjie Huang, Mingze Li, Dongchao Yang, Jiatong Shi, Xuankai Chang, Zhenhui Ye, Yuning Wu, Zhiqing Hong, Jiawei Huang, Jinglin Liu, et al. Audiogpt: Understanding and generating speech, music, sound, and talking head. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 23802–23804, 2024.

[500] Shilong Liu, Hao Cheng, Haotian Liu, Hao Zhang, Feng Li, Tianhe Ren, Xueyan Zou, Jianwei Yang, Hang Su, Jun Zhu, et al. Llava-plus: Learning to use tools for creating multimodal agents. In *European Conference on Computer Vision*, pages 126–142. Springer, 2025.

[501] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pages 4904–4916. PMLR, 2021.

[502] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

[503] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019.

[504] Hu Xu, Gargi Ghosh, Po-Yao Huang, Dmytro Okhonko, Armen Aghajanyan, Florian Metze, Luke Zettlemoyer, and Christoph Feichtenhofer. Videoclip: Contrastive pre-training for zero-shot video-text understanding. *arXiv preprint arXiv:2109.14084*, 2021.

[505] Ruben Villegas, Mohammad Babaeizadeh, Pieter-Jan Kindermans, Hernan Moraldo, Han Zhang, Mohammad Taghi Saffar, Santiago Castro, Julius Kunze, and Dumitru Erhan. Phenaki: Variable length video generation from open domain textual description. *arXiv preprint arXiv:2210.02399*, 2022.

[506] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022.

[507] Ho-Hsiang Wu, Prem Seetharaman, Kundan Kumar, and Juan Pablo Bello. Wav2clip: Learning robust audio representations from clip. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4563–4567. IEEE, 2022.

[508] Hassan Akbari, Liangzhe Yuan, Rui Qian, Wei-Hong Chuang, Shih-Fu Chang, Yin Cui, and Boqing Gong. Vatt: Transformers for multimodal self-supervised learning from raw video, audio and text. *Advances in neural information processing systems*, 34:24206–24221, 2021.

[509] Andrey Guzhov, Federico Raue, Jörn Hees, and Andreas Dengel. Audioclip: Extending clip to image, text and audio. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 976–980. IEEE, 2022.

[510] Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshan. Clip-forge: Towards zero-shot text-to-shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18603–18613, 2022.

[511] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022.

[512] Jun Chen, Deyao Zhu, Xiaoqian Shen, Xiang Li, Zechun Liu, Pengchuan Zhang, Raghuraman Krishnamoorthi, Vikas Chandra, Yunyang Xiong, and Mohamed Elhoseiny. Minigpt-v2: large language model as a unified interface for vision-language multi-task learning. *arXiv preprint arXiv:2310.09478*, 2023.

[513] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, 2024.

[514] Wenyi Hong, Weihan Wang, Ming Ding, Wenmeng Yu, Qingsong Lv, Yafan Wang, Yean Cheng, Shiyu Huang, Junhui Ji, Zhao Xue, et al. Cogvlm2: Visual language models for image and video understanding. *arXiv preprint arXiv:2408.16500*, 2024.

[515] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[516] Quan Sun, Yufeng Cui, Xiaosong Zhang, Fan Zhang, Qiying Yu, Yueze Wang, Yongming Rao, Jingjing Liu, Tiejun Huang, and Xinlong Wang. Generative multimodal models are in-context learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14398–14409, 2024.

[517] Zhengqing Yuan, Zhaoxu Li, Weiran Huang, Yanfang Ye, and Lichao Sun. Tinygpt-v: Efficient multimodal large language model via small backbones. *arXiv preprint arXiv:2312.16862*, 2023.

[518] Xiangxiang Chu, Limeng Qiao, Xinyang Lin, Shuang Xu, Yang Yang, Yiming Hu, Fei Wei, Xinyu Zhang, Bo Zhang, Xiaolin Wei, et al. Mobilevlm: A fast, strong and open vision language assistant for mobile devices. *arXiv preprint arXiv:2312.16886*, 2023.

[519] Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, et al. Minicpm-v: A gpt-4v level mllm on your phone. *arXiv preprint arXiv:2408.01800*, 2024.

[520] Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. Omniparser for pure vision based gui agent. *arXiv preprint arXiv:2408.00203*, 2024.

[521] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on robot learning*, pages 894–906. PMLR, 2022.

[522] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[523] Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Sean Kirmani, Brianna Zitkovich, Fei Xia, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023.

[524] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.

[525] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

[526] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[527] Yining Hong, Zishuo Zheng, Peihao Chen, Yian Wang, Junyan Li, and Chuang Gan. Multiply: A multisensory object-centric embodied large language model in 3d world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26406–26416, 2024.

[528] Zhifeng Kong, Arushi Goel, Rohan Badlani, Wei Ping, Rafael Valle, and Bryan Catanzaro. Audio flamingo: A novel audio language model with few-shot learning and dialogue abilities. *arXiv preprint arXiv:2402.01831*, 2024.

[529] Nilaksh Das, Saket Dingliwal, Srikanth Ronanki, Rohit Paturi, Zhaocheng Huang, Prashant Mathur, Jie Yuan, Dhanush Bekal, Xing Niu, Sai Muralidhar Jayanthi, et al. Speechverse: A large-scale generalizable audio language model. *arXiv preprint arXiv:2405.08295*, 2024.

[530] Dongchao Yang, Haohan Guo, Yuanyuan Wang, Rongjie Huang, Xiang Li, Xu Tan, Xixin Wu, and Helen Meng. Uniaudio 1.5: Large language model-driven audio codec is a few-shot audio task learner. *arXiv preprint arXiv:2406.10056*, 2024.

[531] Dongting Li, Chenchong Tang, and Han Liu. Audio-llm: Activating the capabilities of large language models to comprehend audio data. In *International Symposium on Neural Networks*, pages 133–142. Springer, 2024.

[532] Zhifei Xie and Changqiao Wu. Mini-omni: Language models can hear, talk while thinking in streaming. *arXiv preprint arXiv:2408.16725*, 2024.

[533] Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities. *arXiv preprint arXiv:2305.11000*, 2023.

[534] Peng Wang, Shijie Wang, Junyang Lin, Shuai Bai, Xiaohuan Zhou, Jingren Zhou, Xinggang Wang, and Chang Zhou. One-peace: Exploring one general representation model toward unlimited modalities. *arXiv preprint arXiv:2305.11172*, 2023.

[535] Yixuan Su, Tian Lan, Huayang Li, Jialu Xu, Yan Wang, and Deng Cai. Pandagpt: One model to instruction-follow them all. *arXiv preprint arXiv:2305.16355*, 2023.

[536] Chenyang Lyu, Minghao Wu, Longyue Wang, Xinting Huang, Bingshuai Liu, Zefeng Du, Shuming Shi, and Zhaopeng Tu. Macaw-LLM: Multi-modal language modeling with image, audio, video, and text integration. *arXiv preprint arXiv:2306.09093*, 2023.

[537] Bin Zhu, Bin Lin, Munan Ning, Yang Yan, Jiaxi Cui, HongFa Wang, Yatian Pang, Wenhao Jiang, Junwu Zhang, Zongwei Li, et al. Languagebind: Extending video-language pretraining to n-modality by language-based semantic alignment. *arXiv preprint arXiv:2310.01852*, 2023.

[538] Mustafa Shukor, Corentin Dancette, Alexandre Rame, and Matthieu Cord. Unival: Unified model for image, video, audio and language tasks. *Transactions on Machine Learning Research Journal*, 2023.

[539] Feilong Chen, Minglun Han, Haozhi Zhao, Qingyang Zhang, Jing Shi, Shuang Xu, and Bo Xu. X-LLM: Bootstrapping advanced large language models by treating multi-modalities as foreign languages. *arXiv preprint arXiv:2305.04160*, 2023.

[540] Runsen Xu, Xiaolong Wang, Tai Wang, Yilun Chen, Jiangmiao Pang, and Dahua Lin. PointLLM: Empowering large language models to understand point clouds. In *European Conference on Computer Vision*, pages 131–147. Springer, 2025.

[541] Yuan Tang, Xu Han, Xianzhi Li, Qiao Yu, Yixue Hao, Long Hu, and Min Chen. Minigpt-3d: Efficiently aligning 3d point clouds with large language models using 2d priors. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 6617–6626, 2024.

[542] Shengqiong Wu, Hao Fei, Leigang Qu, Wei Ji, and Tat-Seng Chua. Next-gpt: Any-to-any multimodal LLM. *arXiv preprint arXiv:2309.05519*, 2023.

[543] Jiasen Lu, Christopher Clark, Sangho Lee, Zichen Zhang, Savya Khosla, Ryan Marten, Derek Hoiem, and Aniruddha Kembhavi. Unified-io 2: Scaling autoregressive multimodal models with vision language audio and action. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26439–26455, 2024.

[544] Zineng Tang, Ziyi Yang, Mahmoud Khademi, Yang Liu, Chenguang Zhu, and Mohit Bansal. Codi-2: In-context interleaved and interactive any-to-any generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27425–27434, 2024.

[545] Xinyu Wang, Bohan Zhuang, and Qi Wu. Modaverse: Efficiently transforming modalities with LLMs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26606–26616, 2024.

[546] Fiona Macpherson. *The senses: Classic and contemporary philosophical perspectives*, volume 11. Oxford University Press, 2011.

[547] Jamie Ward. *The student's guide to cognitive neuroscience*. Routledge, 2019.

[548] Stanley Coren, Lawrence M Ward, and James T Enns. *Sensation and perception*. John Wiley & Sons Hoboken, NJ, 2004.

[549] Simon Grondin. Timing and time perception: A review of recent behavioral and neuroscience findings and theoretical directions. *Attention, Perception, & Psychophysics*, 72(3):561–582, 2010.

[550] Henrik Mouritsen. Long-distance navigation and magnetoreception in migratory animals. *Nature*, 558(7708): 50–59, 2018.

[551] Chen Wang, Zhesi Chen, Chak Lam Jonathan Chan, Zhu'an Wan, Wenhao Ye, Wenying Tang, Zichao Ma, Beitao Ren, Daquan Zhang, Zhilong Song, et al. Biomimetic olfactory chips based on large-scale monolithically integrated nanotube sensor arrays. *Nature Electronics*, 7(2):157–167, 2024.

[552] Caroline Bushdid, Marcelo O Magnasco, Leslie B Vosshall, and Andreas Keller. Humans can discriminate more than 1 trillion olfactory stimuli. *Science*, 343(6177):1370–1372, 2014.

[553] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):423–443, 2018.

[554] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332, 2016.

[555] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. Understanding bag-of-words model: a statistical framework. *International journal of machine learning and cybernetics*, 1:43–52, 2010.

[556] OpenAI. Gpt-3.5: Language model, 2023. URL https://platform.openai.com/docs/models/gpt-3.5-turbo.

[557] Glenn Jocher. YOLOv5 by Ultralytics, May 2020. URL https://github.com/ultralytics/yolov5.

[558] Glenn Jocher, Jing Qiu, and Ayush Chaurasia. Ultralytics YOLO, January 2023. URL https://github.com/ultralytics/ultralytics.

[559] Chang Zeng, Xin Wang, Erica Cooper, Xiaoxiao Miao, and Junichi Yamagishi. Attention back-end for automatic speaker verification with multiple enrollment utterances. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6717–6721. IEEE, 2022.

[560] Zishuo Zhang and Bing Yan. Smart multiple photoresponsive tongue for sensing umami, sour and bitter tastes based on tb3+ functionalized hydrogen-bonded organic frameworks. *Advanced Functional Materials*, 34(25): 2316195, 2024.

[561] Raunaq Bhirangi, Venkatesh Pattabiraman, Enes Erciyes, Yifeng Cao, Tess Hellebrekers, and Lerrel Pinto. Anyskin: Plug-and-play skin sensing for robotic touch. *arXiv preprint arXiv:2409.08276*, 2024.

[562] Shashank Goel, Hritik Bansal, Sumit Bhatia, Ryan Rossi, Vishwa Vinay, and Aditya Grover. Cyclip: Cyclic contrastive language-image pretraining. *Advances in Neural Information Processing Systems*, 35:6704–6719, 2022.

[563] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International conference on machine learning*, pages 8821–8831. Pmlr, 2021.

[564] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

[565] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[566] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pages 12888–12900. PMLR, 2022.

[567] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[568] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1728–1738, 2021.

[569] Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi. Audiogen: Textually guided audio generation. *arXiv preprint arXiv:2209.15352*, 2022.

[570] Junyi Ao, Rui Wang, Long Zhou, Chengyi Wang, Shuo Ren, Yu Wu, Shujie Liu, Tom Ko, Qing Li, Yu Zhang, et al. Speecht5: Unified-modal encoder-decoder pre-training for spoken language processing. *arXiv preprint arXiv:2110.07205*, 2021.

[571] Prakhar Bhardwaj, Sheethal Bhat, and Andreas Maier. Enhancing zero-shot learning in medical imaging: integrating clip with advanced techniques for improved chest x-ray analysis. *arXiv preprint arXiv:2503.13134*, 2025.

[572] Zhiyu Wu, Xiaokang Chen, Zizheng Pan, Xingchao Liu, Wen Liu, Damai Dai, Huazuo Gao, Yiyang Ma, Chengyue Wu, Bingxuan Wang, Zhenda Xie, Yu Wu, Kai Hu, Jiawei Wang, Yaofeng Sun, Yukun Li, Yishi Piao, Kang Guan, Aixin Liu, Xin Xie, Yuxiang You, Kai Dong, Xingkai Yu, Haowei Zhang, Liang Zhao, Yisong Wang, and Chong Ruan. Deepseek-vl2: Mixture-of-experts vision-language models for advanced multimodal understanding. *arXiv preprint arXiv:2412.10302*, 2024. URL https://arxiv.org/abs/2412.10302.

[573] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. *arXiv preprint arXiv:2306.05424*, 2023.

[574] Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*, 2023.

[575] Peng Jin, Ryuichi Takanobu, Wancai Zhang, Xiaochun Cao, and Li Yuan. Chat-univi: Unified visual representation empowers large language models with image and video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13700–13710, 2024.

[576] Haiyang Xu, Qinghao Ye, Xuan Wu, Ming Yan, Yuan Miao, Jiabo Ye, Guohai Xu, Anwen Hu, Yaya Shi, Guangwei Xu, et al. Youku-mplug: A 10 million large-scale chinese video-language dataset for pre-training and benchmarks. *arXiv preprint arXiv:2306.04362*, 2023.

[577] Mingze Xu, Mingfei Gao, Zhe Gan, Hong-You Chen, Zhengfeng Lai, Haiming Gang, Kai Kang, and Afshin Dehghan. Slowfast-llava: A strong training-free baseline for video large language models. *arXiv preprint arXiv:2407.15841*, 2024.

[578] Mojan Javaheripi, Sébastien Bubeck, Marah Abdin, Jyoti Aneja, Sebastien Bubeck, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, et al. Phi-2: The surprising power of small language models. *Microsoft Research Blog*, 1(3):3, 2023.

[579] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

[580] Boxun Li, Yadong Li, Zhiyuan Li, Congyi Liu, Weilin Liu, Guowei Niu, Zheyue Tan, Haiyang Xu, Zhuyu Yao, Tao Yuan, et al. Megrez-omni technical report. *arXiv preprint arXiv:2502.15803*, 2025.

[581] Wentong Chen, Junbo Cui, Jinyi Hu, Yujia Qin, Junjie Fang, Yue Zhao, Chongyi Wang, Jun Liu, Guirong Chen, Yupeng Huo, et al. Guicourse: From general vision language models to versatile gui agents. *arXiv preprint arXiv:2406.11317*, 2024.

[582] Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, et al. Os-atlas: A foundation action model for generalist gui agents. *arXiv preprint arXiv:2410.23218*, 2024.

[583] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[584] Zhihao Du, Jiaming Wang, Qian Chen, Yunfei Chu, Zhifu Gao, Zerui Li, Kai Hu, Xiaohuan Zhou, Jin Xu, Ziyang Ma, et al. Lauragpt: Listen, attend, understand, and regenerate audio with gpt. *arXiv preprint arXiv:2310.04673*, 2023.

[585] Sreyan Ghosh, Sonal Kumar, Ashish Seth, Chandra Kiran Reddy Evuru, Utkarsh Tyagi, S Sakshi, Oriol Nieto, Ramani Duraiswami, and Dinesh Manocha. Gama: A large audio-language model with advanced audio understanding and complex reasoning abilities. *arXiv preprint arXiv:2406.11768*, 2024.

[586] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[587] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[588] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15180–15190, 2023.

[589] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging LLM-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.

[590] Jun Zhan, Junqi Dai, Jiasheng Ye, Yunhua Zhou, Dong Zhang, Zhigeng Liu, Xin Zhang, Ruibin Yuan, Ge Zhang, Linyang Li, et al. Anygpt: Unified multimodal LLM with discrete sequence modeling. *arXiv preprint arXiv:2402.12226*, 2024.

[591] Chenming Zhu, Tai Wang, Wenwei Zhang, Jiangmiao Pang, and Xihui Liu. Llava-3d: A simple yet effective pathway to empowering lmms with 3d-awareness. *arXiv preprint arXiv:2409.18125*, 2024.

[592] Sudharshan Suresh, Haozhi Qi, Tingfan Wu, Taosha Fan, Luis Pineda, Mike Lambeta, Jitendra Malik, Mrinal Kalakrishnan, Roberto Calandra, Michael Kaess, et al. Neuralfeels with neural fields: Visuotactile perception for in-hand manipulation. *Science Robotics*, 9(96):eadl0628, 2024.

[593] Zhizhao Duan, Hao Cheng, Duo Xu, Xi Wu, Xiangxie Zhang, Xi Ye, and Zhen Xie. Cityllava: Efficient fine-tuning for vlms in city scenario. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 7180–7189, June 2024.

[594] Junfeng Fang, Zac Bi, Ruipeng Wang, Houcheng Jiang, Yuan Gao, Kun Wang, An Zhang, Jie Shi, Xiang Wang, and Tat-Seng Chua. Towards neuron attributions in multi-modal large language models. *Advances in Neural Information Processing Systems*, 37:122867–122890, 2024.

[595] Yuxin Fang, Bencheng Liao, Xinggang Wang, Jiemin Fang, Jiyang Qi, Rui Wu, Jianwei Niu, and Wenyu Liu. You only look at one sequence: Rethinking transformer in vision through object detection. *Advances in Neural Information Processing Systems*, 34:26183–26197, 2021.

[596] Renrui Zhang, Jiaming Han, Chris Liu, Peng Gao, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, and Yu Qiao. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*, 2023.

[597] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

[598] Qingbin Zeng, Qinglong Yang, Shunan Dong, Heming Du, Liang Zheng, Fengli Xu, and Yong Li. Perceive, reflect, and plan: Designing LLM agent for goal-directed city navigation without instructions. *arXiv preprint arXiv:2408.04168*, 2024.

[599] Yaodong Yang and Jun Wang. An overview of multi-agent reinforcement learning from game theoretical perspective. *arXiv preprint arXiv:2011.00583*, 2020.

[600] Zhenbei Guo, Fuliang Li, Jiaxing Shen, Tangzheng Xie, Shan Jiang, and Xingwei Wang. Configreco: Network configuration recommendation with graph neural networks. *IEEE Network*, 2023.

[601] Huaxiang Zhang, Yaojia Mu, Guo-Niu Zhu, and Zhongxue Gan. Insightsee: Advancing multi-agent vision-language models for enhanced visual understanding. *arXiv preprint arXiv:2405.20795*, 2024.

[602] Andrew Nash, Andrew Vardy, and Dave Churchill. Herd's eye view: Improving game AI agent learning with collaborative perception. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 19, pages 306–314, 2023.

[603] Zhehao Zhang, Ryan Rossi, Tong Yu, Franck Dernoncourt, Ruiyi Zhang, Jiuxiang Gu, Sungchul Kim, Xiang Chen, Zichao Wang, and Nedim Lipka. Vipact: Visual-perception enhancement via specialized vlm agent collaboration and tool-use. *arXiv preprint arXiv:2410.16400*, 2024.

[604] Bingchen Li, Xin Li, Yiting Lu, and Zhibo Chen. Lossagent: Towards any optimization objectives for image processing with LLM agents. *arXiv preprint arXiv:2412.04090*, 2024.

[605] Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. *arXiv preprint arXiv:2210.02406*, 2022.

[606] Jonathon Schwartz, Rhys Newbury, Dana Kulic, and Hanna Kurniawati. Posggym: A library for decision-theoretic planning and learning in partially observable, multi-agent environments. In *Proceedings of the 33rd International Conference on Automated Planning and Scheduling (ICAPS)*, 2024.

[607] Zhonghan Zhao, Wenhao Chai, Xuan Wang, Boyi Li, Shengyu Hao, Shidong Cao, Tian Ye, and Gaoang Wang. See and think: Embodied agent in virtual environment. In *European Conference on Computer Vision*, pages 187–204. Springer, 2025.

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

[609] Difei Gao, Siyuan Hu, Zechen Bai, Qinghong Lin, and Mike Zheng Shou. Assisteditor: Multi-agent collaboration for gui workflow automation in video creation. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 11255–11257, 2024.

[610] Zixuan Wang, Yu-Wing Tai, and Chi-Keung Tang. Audio-agent: Leveraging LLMs for audio generation, editing and composition. *arXiv preprint arXiv:2410.03335*, 2024.

[611] Shuoyi Zhou, Yixuan Zhou, Weiqing Li, Jun Chen, Runchuan Ye, Weihao Wu, Zijian Lin, Shun Lei, and Zhiyong Wu. The codec language model-based zero-shot spontaneous style tts system for covoc challenge 2024. In *2024 IEEE 14th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 496–500. IEEE, 2024.

[612] Kai Li and Yi Luo. Apollo: Band-sequence modeling for high-quality audio restoration. *arXiv preprint arXiv:2409.08514*, 2024.

[613] Chunhui Wang, Chang Zeng, Bowen Zhang, Ziyang Ma, Yefan Zhu, Zifeng Cai, Jian Zhao, Zhonglin Jiang, and Yong Chen. Ham-tts: Hierarchical acoustic modeling for token-based zero-shot text-to-speech with model and data scaling. *arXiv preprint arXiv:2403.05989*, 2024.

[614] Xiao Yu, Baolin Peng, Vineeth Vajipey, Hao Cheng, Michel Galley, Jianfeng Gao, and Zhou Yu. Exact: Teaching AI agents to explore with reflective-mcts and exploratory learning. *arXiv preprint arXiv:2410.02052*, 2024.

[615] Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. *arXiv preprint arXiv:2401.13649*, 2024.

[616] Jingxuan Chen, Derek Yuen, Bin Xie, Yuhao Yang, Gongwei Chen, Zhihao Wu, Li Yixing, Xurui Zhou, Weiwen Liu, Shuai Wang, et al. Spa-bench: A comprehensive benchmark for smartphone agent evaluation. In *NeurIPS 2024 Workshop on Open-World Agents*, 2024.

[617] Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, et al. Androidworld: A dynamic benchmarking environment for autonomous agents. *arXiv preprint arXiv:2405.14573*, 2024.

[618] Chengyou Jia, Minnan Luo, Zhuohang Dang, Qiushi Sun, Fangzhi Xu, Junlin Hu, Tianbao Xie, and Zhiyong Wu. Agentstore: Scalable integration of heterogeneous agents as specialized generalist computer assistant. *arXiv preprint arXiv:2410.18603*, 2024.

[619] Aohan Zeng, Zhengxiao Du, Mingdao Liu, Kedong Wang, Shengmin Jiang, Lei Zhao, Yuxiao Dong, and Jie Tang. Glm-4-voice: Towards intelligent and human-like end-to-end spoken chatbot. *arXiv preprint arXiv:2412.02612*, 2024.

[620] Mike Lambeta, Tingfan Wu, Ali Sengul, Victoria Rose Most, Nolan Black, Kevin Sawyer, Romeo Mercado, Haozhi Qi, Alexander Sohn, Byron Taylor, et al. Digitizing touch with an artificial multimodal fingertip. *arXiv preprint arXiv:2411.02479*, 2024.

[621] Peiyan Zhang, Haoyang Liu, Chaozhuo Li, Xing Xie, Sunghun Kim, and Haohan Wang. Foundation model-oriented robustness: Robust image model evaluation with pretrained models. In *ICLR*, 2024.

[622] Lu Wang, Fangkai Yang, Chaoyun Zhang, Junting Lu, Jiaxu Qian, Shilin He, Pu Zhao, Bo Qiao, Ray Huang, Si Qin, Qisheng Su, Jiayi Ye, Yudi Zhang, Jian-Guang Lou, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. Large action models: From inception to implementation. *CoRR*, abs/2412.10047, 2024.

[623] Volker Krüger, Danica Kragic, Aleš Ude, and Christopher Geib. The meaning of action: A review on action recognition and mapping. *Advanced robotics*, 21(13):1473–1501, 2007.

[624] Nico Dosenbach, Marus Raichle, and Evan Gordon. The brain's action-mode network. *Nature reviews. Neuroscience*, 26, 01 2025. doi:10.1038/s41583-024-00895-x.

[625] Significant Gravitas. Auto-gpt: An autonomous gpt-4 experiment. https://github.com/Significant-Gravitas/Auto-GPT, 2023.

[626] Sirui Hong, Mingchen Xia, Jonathan Wang, Zhanghao Li, Zili Chen, Junjue He, Jiazheng Fan, Chenyu Zhou, Beining Mei, et al. MetaGPT: Meta programming for multi-agent collaborative framework. In *International Conference on Learning Representations*, 2023.

[627] Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. Chatdev: Communicative agents for software development, 2024. URL https://arxiv.org/abs/2307.07924.

[628] John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering, 2024. URL https://arxiv.org/abs/2405.15793.

[629] Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, Hoang H. Tran, Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian, Yanjun Shao, Niklas Muennighoff, Yizhe Zhang, Binyuan Hui, Junyang Lin, Robert Brennan, Hao Peng, Heng Ji, and Graham Neubig. OpenHands: An Open Platform for AI Software Developers as Generalist Agents, 2024. URL https://arxiv.org/abs/2407.16741.

[630] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen LLM applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*, 2023.

[631] Xiao Shao, Weifu Jiang, Fei Zuo, and Mengqing Liu. Swarmbrain: Embodied agent for real-time strategy game starcraft II via large language models. *CoRR*, abs/2401.17749, 2024.

[632] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

[633] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[634] Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A real-world webagent with planning, long context understanding, and program synthesis, 2024. URL https://arxiv.org/abs/2307.12856.

[635] Junyang Wang, Haiyang Xu, Jiabo Ye, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. Mobile-agent: Autonomous multi-modal mobile device agent with visual perception. *CoRR*, abs/2401.16158, 2024.

[636] Chi Zhang, Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. Appagent: Multimodal agents as smartphone users. *CoRR*, abs/2312.13771, 2023.

[637] Chaoyun Zhang, Liqun Li, Shilin He, Xu Zhang, Bo Qiao, Si Qin, Minghua Ma, Yu Kang, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. UFO: A ui-focused agent for windows OS interaction. *CoRR*, abs/2402.07939, 2024.

[638] Tianbao Xie, Chen Henry Wu, Peng Shi, Ruiqi Zhong, Torsten Scholak, Michihiro Yasunaga, Chien-Sheng Wu, Ming Zhong, Pengcheng Yin, Sida I Wang, et al. Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2022.

[639] Yu Gu, Xiang Deng, and Yu Su. Don't generate, discriminate: A proposal for grounding language models to real-world environments. In *ACL*, 2023.

[640] Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, Xuanhe Zhou, Chenhao Ma, Guoliang Li, Kevin Chen-Chuan Chang, Fei Huang, Reynold Cheng, and Yongbin Li. Can LLM already serve as A database interface? A big bench for large-scale database grounded text-to-sqls. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/83fc8fab1710363050bbd1d4b8cc0021-Abstract-Datasets_and_Benchmarks.html.

[641] Fangyu Lei, Jixuan Chen, Yuxiao Ye, Ruisheng Cao, Dongchan Shin, SU Hongjin, ZHAOQING SUO, Hongcheng Gao, Wenjing Hu, Pengcheng Yin, et al. Spider 2.0: Evaluating language models on real-world enterprise text-to-sql workflows. In *The Thirteenth International Conference on Learning Representations*, 2024.

[642] Yu Gu, Yiheng Shu, Hao Yu, Xiao Liu, Yuxiao Dong, Jie Tang, Jayanth Srinivasa, Hugo Latapie, and Yu Su. Middleware for llms: Tools are instrumental for language agents in complex environments. In *EMNLP*, 2024.

[643] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[644] Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[645] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[646] Brian Ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, Dmitry Kalashnikov, Sergey Levine, Yao Lu, Carolina Parada, Kanishka Rao, Pierre Sermanet, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Mengyuan Yan, Noah Brown, Michael Ahn, Omar Cortes, Nicolas Sievers, Clayton Tan, Sichun Xu, Diego Reyes, Jarek Rettinghouse, Jornell Quiambao, Peter Pastor, Linda Luu, Kuang-Huei Lee, Yuheng Kuang, Sally Jesmonth, Nikhil J. Joshi, Kyle Jeffrey, Rosario Jauregui Ruano, Jasmine Hsu, Keerthana Gopalakrishnan, Byron David, Andy Zeng, and Chuyuan Kelly Fu. Do as I can, not as I say: Grounding language in robotic affordances. In *Conference on Robot Learning, CoRL 2022, 14-18 December 2022, Auckland, New Zealand*, volume 205 of *Proceedings of Machine Learning Research*, pages 287–318. PMLR, 2022.

[647] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. In *Conference on Robot Learning, CoRL 2023, 6-9 November 2023, Atlanta, GA, USA*, volume 229 of *Proceedings of Machine Learning Research*, pages 540–562. PMLR, 2023.

[648] Yao Mu, Qinglong Zhang, Mengkang Hu, Wenhai Wang, Mingyu Ding, Jun Jin, Bin Wang, Jifeng Dai, Yu Qiao, and Ping Luo. Embodiedgpt: Vision-language pre-training via embodied chain of thought. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[649] Yunxuan Li, Yibing Du, Jiageng Zhang, Le Hou, Peter Grabowski, Yeqing Li, and Eugene Ie. Improving multi-agent debate with sparse communication topology. *arXiv preprint arXiv:2406.11776*, 2024.

[650] Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2609–2634, 2023.

[651] Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. Gptswarm: Language agents as optimizable graphs. In *ICML*. OpenReview.net, 2024.

[652] Haiteng Zhao, Chang Ma, Guoyin Wang, Jing Su, Lingpeng Kong, Jingjing Xu, Zhi-Hong Deng, and Hongxia Yang. Empowering large language model agents through action learning. *arXiv preprint arXiv:2402.15809*, 2024.

[653] Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.

[654] Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quon Vuong, Jonathan Tompson, Yevgen Chebotar, Debidatta Dwibedi, and Dorsa Sadigh. Rt-h: Action hierarchies using language. *arXiv preprint arXiv:2403.01823*, 2024.

[655] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[656] Jinliang Zheng, Jianxiong Li, Dongxiu Liu, Yinan Zheng, Zhihao Wang, Zhonghong Ou, Yu Liu, Jingjing Liu, Ya-Qin Zhang, and Xianyuan Zhan. Universal actions for enhanced embodied foundation models. *arXiv preprint arXiv:2501.10105*, 2025.

[657] Weirui Ye, Yunsheng Zhang, Haoyang Weng, Xianfan Gu, Shengjie Wang, Tong Zhang, Mengchen Wang, Pieter Abbeel, and Yang Gao. Reinforcement learning with foundation priors: Let the embodied agent efficiently learn on its own. *arXiv preprint arXiv:2310.02635*, 2023.

[658] Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. Guiding pretraining in reinforcement learning with large language models. In *International Conference on Machine Learning*, pages 8657–8677. PMLR, 2023.

[659] Lirui Wang, Yiyang Ling, Zhecheng Yuan, Mohit Shridhar, Chen Bao, Yuzhe Qin, Bailin Wang, Huazhe Xu, and Xiaolong Wang. Gensim: Generating robotic simulation tasks via large language models. *arXiv preprint arXiv:2310.01361*, 2023.

[660] Jie Wang, Alexandros Karatzoglou, Ioannis Arapakis, and Joemon M Jose. Reinforcement learning-based recommender systems with large language models for state reward and action modeling. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 375–385, 2024.

[661] Jiajun Chai, Sicheng Li, Yuqian Fu, Dongbin Zhao, and Yuanheng Zhu. Empowering LLM agents with zero-shot optimal decision-making through q-learning. In *Adaptive Foundation Models: Evolving AI for Personalized and Efficient Learning*, 2024.

[662] Jing-Cheng Pang, Si-Hang Yang, Kaiyuan Li, Jiaji Zhang, Xiong-Hui Chen, Nan Tang, and Yang Yu. Kalm: Knowledgeable agents by offline reinforcement learning from large language model rollouts. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[663] Bin Hu, Chenyang Zhao, Pu Zhang, Zihao Zhou, Yuanhang Yang, Zenglin Xu, and Bin Liu. Enabling intelligent interactions between an agent and an llm: A reinforcement learning approach. *arXiv preprint arXiv:2306.03604*, 2023.

[664] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. *arXiv preprint arXiv:2310.12931*, 2023.

[665] Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. Archer: Training language model agents via hierarchical multi-turn rl, 2024b. *URL https://arxiv. org/pdf/2402*, 19446, 2024.

[666] Andrew Szot, Max Schwarzer, Harsh Agrawal, Bogdan Mazoure, Rin Metcalf, Walter Talbott, Natalie Mackraz, R Devon Hjelm, and Alexander T Toshev. Large language models as generalizable policies for embodied tasks. In *The Twelfth International Conference on Learning Representations*, 2023.

[667] Xinyu Liu, Shuyu Shen, Boyan Li, Nan Tang, and Yuyu Luo. Nl2sql-bugs: A benchmark for detecting semantic errors in nl2sql translation, 2025. URL https://arxiv.org/abs/2503.11984.

[668] Xuedi Qin, Chengliang Chai, Yuyu Luo, Tianyu Zhao, Nan Tang, Guoliang Li, Jianhua Feng, Xiang Yu, and Mourad Ouzzani. Interactively discovering and ranking desired tuples by data exploration. *VLDB J.*, 31(4): 753–777, 2022.

[669] Reg Revans. *ABC of action learning*. Routledge, 2017.

[670] Shaokun Zhang, Jieyu Zhang, Jiale Liu, Linxin Song, Chi Wang, Ranjay Krishna, and Qingyun Wu. Offline training of language model agents with functions as learnable weights. In *Forty-first International Conference on Machine Learning*, 2024.

[671] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[672] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986, 2023.

[673] Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.

[674] Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, et al. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024.

[675] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[676] Daeyeol Lee, Hyojung Seo, and Min Whan Jung. Neural basis of reinforcement learning and decision making. *Annual review of neuroscience*, 35(1):287–308, 2012.

[677] Jiabin Liu, Chengliang Chai, Yuyu Luo, Yin Lou, Jianhua Feng, and Nan Tang. Feature augmentation with reinforcement learning. In *ICDE*, pages 3360–3372. IEEE, 2022.

[678] Chengliang Chai, Kaisen Jin, Nan Tang, Ju Fan, Lianpeng Qiao, Yuping Wang, Yuyu Luo, Ye Yuan, and Guoren Wang. Mitigating data scarcity in supervised machine learning through reinforcement learning guided data generation. In *ICDE*, pages 3613–3626. IEEE, 2024.

[679] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[680] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[681] Kimi Team. Kimi k1.5: Scaling reinforcement learning with llms. *CoRR*, abs/2501.12599, 2025.

[682] Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. Is reinforcement learning (not) for natural language processing: Benchmarks, baselines, and building blocks for natural language policy optimization. *arXiv preprint arXiv:2210.01241*, 2022.

[683] Jian Hu, Li Tao, June Yang, and Chandler Zhou. Aligning language models with offline learning from human feedback. *arXiv preprint arXiv:2308.12050*, 2023.

[684] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

[685] Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning, 2025. URL https://arxiv.org/abs/2503.09516.

[686] Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher: Incentivizing the search capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2503.05592.

[687] Zihan Wang*, Kangrui Wang*, Qineng Wang*, Pingyue Zhang*, Linjie Li*, Zhengyuan Yang, Kefan Yu, Minh Nhat Nguyen, Monica Lam, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. Training agents by reinforcing reasoning, 2025. URL https://github.com/ZihanWang314/ragen.

[688] OpenManus-RL Team. Openmanus-rl: Open platform for generalist llm reasoning agents with rl optimization, 2025. URL https://github.com/OpenManus/OpenManus-RL.

[689] Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=Yacmpz84TH.

[690] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis, 2023. URL https://arxiv.org/abs/2307.16789.

[691] Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive APIs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=tBRNC6YemY.

[692] Shibo Hao, Tianyang Liu, Zhen Wang, and Zhiting Hu. Toolkengpt: Augmenting frozen language models with massive tools via tool embeddings. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[693] Rui Yang, Lin Song, Yanwei Li, Sijie Zhao, Yixiao Ge, Xiu Li, and Ying Shan. Gpt4tools: Teaching large language model to use tools via self-instruction. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/e393677793767624f2821cec8bdd02f1-Abstract-Conference.html.

[694] Yu Du, Fangyun Wei, and Hongyang Zhang. Anytool: Self-reflective, hierarchical agents for large-scale api calls, 2024. URL https://arxiv.org/abs/2402.04253.

[695] Jimmy Wu, Rika Antonova, Adam Kan, Marion Lepert, Andy Zeng, Shuran Song, Jeannette Bohg, Szymon Rusinkiewicz, and Thomas Funkhouser. Tidybot: personalized robot assistance with large language models. *Autonomous Robots*, 47(8):1087–1102, November 2023. ISSN 1573-7527. doi:10.1007/s10514-023-10139-z. URL http://dx.doi.org/10.1007/s10514-023-10139-z.

[696] Chang Qi, Feng Jiang, and Shu Yang. Advanced honeycomb designs for improving mechanical properties: A review. *Composites Part B: Engineering*, 227:109393, 2021. ISSN 1359-8368. doi:https://doi.org/10.1016/j.compositesb.2021.109393.

[697] Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D. White, and Philippe Schwaller. Augmenting large language models with chemistry tools. *Nat. Mac. Intell.*, 6(5):525–535, 2024.

[698] Huajun Chen, Keyan Ding, Jing Yu, Junjie Huang, Yuchen Yang, and Qiang Zhang. Scitoolagent: A knowledge graph-driven scientific agent for multi-tool integration. In *ICLR*, 2025.

[699] Yubo Ma, Zhibin Gou, Junheng Hao, Ruochen Xu, Shuohang Wang, Liangming Pan, Yujiu Yang, Yixin Cao, and Aixin Sun. Sciagent: Tool-augmented language models for scientific reasoning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 15701–15736, 2024.

[700] Yuchen Zhuang, Xiang Chen, Tong Yu, Saayan Mitra, Victor Bursztyn, Ryan A Rossi, Somdeb Sarkhel, and Chao Zhang. Toolchain*: Efficient action space navigation in large language models with a* search. *arXiv preprint arXiv:2310.13227*, 2023.

[701] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: program-aided language models. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202, pages 10764–10799, 2023.

[702] Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. Large language models as tool makers. *arXiv preprint arXiv:2305.17126*, 2023.

[703] Cheng Qian, Chi Han, Yi Fung, Yujia Qin, Zhiyuan Liu, and Heng Ji. CREATOR: Tool creation for disentangling abstract and concrete reasoning of large language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=aCHq10rQiH.

[704] Jingqing Ruan, Yihong Chen, Bin Zhang, Zhiwei Xu, Tianpeng Bao, Guoqing Du, Shiwei Shi, Hangyu Mao, Ziyue Li, Xingyu Zeng, and Rui Zhao. Tptu: Large language model-based ai agents for task planning and tool usage, 2023. URL https://arxiv.org/abs/2308.03427.

[705] Yujia Qin, Zihan Cai, Dian Jin, Lan Yan, Shihao Liang, Kunlun Zhu, Yankai Lin, Xu Han, Ning Ding, Huadong Wang, et al. Webcpm: Interactive web search for chinese long-form question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8968–8988, 2023.

[706] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. In *The Twelfth International Conference on Learning Representations*, 2024.

[707] Xuanhe Zhou, Guoliang Li, Zhaoyan Sun, Zhiyuan Liu, Weize Chen, Jianming Wu, Jiesi Liu, Ruohang Feng, and Guoyang Zeng. D-bot: Database diagnosis system using large language models, 2023. URL https://arxiv.org/abs/2312.01454.

[708] Xinyu Liu, Shuyu Shen, Boyan Li, Peixian Ma, Runzhi Jiang, Yuxin Zhang, Ju Fan, Guoliang Li, Nan Tang, and Yuyu Luo. A survey of NL2SQL with large language models: Where are we, and where are we going?, 2025. URL https://arxiv.org/abs/2408.05109.

[709] Boyan Li, Yuyu Luo, Chengliang Chai, Guoliang Li, and Nan Tang. The dawn of natural language to SQL: are we fully ready? *Proc. VLDB Endow.*, 17(11):3318–3331, 2024.

[710] Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. Executable code actions elicit better LLM agents. *arXiv preprint arXiv:2402.01030*, 2024.

[711] Xuedi Qin, Yuyu Luo, Nan Tang, and Guoliang Li. Making data visualization more efficient and effective: a survey. *VLDB J.*, 29(1):93–117, 2020.

[712] Yuyu Luo, Xuedi Qin, Nan Tang, and Guoliang Li. Deepeye: Towards automatic data visualization. In *ICDE*, pages 101–112. IEEE Computer Society, 2018.

[713] Xuedi Qin, Chengliang Chai, Yuyu Luo, Nan Tang, and Guoliang Li. Interactively discovering and ranking desired tuples without writing SQL queries. In *SIGMOD Conference*, pages 2745–2748. ACM, 2020.

[714] Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Xuanhe Zhou, Yufei Huang, Chaojun Xiao, Chi Han, Yi Ren Fung, Yusheng Su, Huadong Wang, Cheng Qian, Runchu Tian, Kunlun Zhu, Shihao Liang, Xingyu Shen, Bokai Xu, Zhen Zhang, Yining Ye, Bowen Li, Ziwei Tang, Jing Yi, Yuzhang Zhu, Zhenning Dai, Lan Yan, Xin Cong, Yaxi Lu, Weilin Zhao, Yuxiang Huang, Junxi Yan, Xu Han, Xian Sun, Dahai Li, Jason Phang, Cheng Yang, Tongshuang Wu, Heng Ji, Guoliang Li, Zhiyuan Liu, and Maosong Sun. Tool learning with foundation models. *ACM Comput. Surv.*, 57(4), December 2024. ISSN 0360-0300. doi:10.1145/3704435. URL https://doi.org/10.1145/3704435.

[715] Sadra Zargarzadeh, Maryam Mirzaei, Yafei Ou, and Mahdi Tavakoli. From decision to action in surgical autonomy: Multi-modal large language models for robot-assisted blood suction. *IEEE Robotics and Automation Letters*, 10(3):2598–2605, March 2025. ISSN 2377-3774. doi:10.1109/lra.2025.3535184. URL http://dx.doi.org/10.1109/LRA.2025.3535184.

[716] Zhenjie Yang, Xiaosong Jia, Hongyang Li, and Junchi Yan. Llm4drive: A survey of large language models for autonomous driving, 2023.

[717] Jiageng Mao, Junjie Ye, Yuxi Qian, Marco Pavone, and Yue Wang. A language agent for autonomous driving. *arXiv preprint arXiv:2311.10813*, 2023.

[718] Sherwood L Washburn. Tools and human evolution. *Scientific American*, 203(3):62–75, 1960.

[719] Nan Tang, Chenyu Yang, Ju Fan, Lei Cao, Yuyu Luo, and Alon Y. Halevy. Verifai: Verified generative AI. In *CIDR*. www.cidrdb.org, 2024.

[720] Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Zhibo Sun, and Yue Zhang. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing*, 4(2):100211, June 2024. ISSN 2667-2952. doi:10.1016/j.hcc.2024.100211. URL http://dx.doi.org/10.1016/j.hcc.2024.100211.

[721] Yongchao Chen, Jacob Arkin, Yilun Hao, Yang Zhang, Nicholas Roy, and Chuchu Fan. Prompt optimization in multi-step tasks (promst): Integrating human feedback and preference alignment. *arXiv preprint arXiv:2402.08702*, 2024.

[722] Yurong Wu, Yan Gao, Bin Benjamin Zhu, Zineng Zhou, Xiaodi Sun, Sheng Yang, Jian-Guang Lou, Zhiming Ding, and Linjun Yang. StraGo: Harnessing strategic guidance for prompt optimization. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10043–10061, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.findings-emnlp.588. URL https://aclanthology.org/2024.findings-emnlp.588.

[723] Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. In *ICLR*. OpenReview.net, 2024.

[724] Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T Joshi, Hanna Moazam, et al. Dspy: Compiling declarative language model calls into self-improving pipelines. *arXiv preprint arXiv:2310.03714*, 2023.

[725] Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. Dynamic LLM-agent network: An LLM-agent collaboration framework with agent team optimization. *arXiv preprint arXiv:2310.02170*, 2023.

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

[727] Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. *arXiv preprint arXiv:2305.15334*, 2023.

[728] Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. Textgrad: Automatic "differentiation" via text. *arXiv preprint arXiv:2406.07496*, 2024.

[729] Yiran Wu, Tianwei Yue, Shaokun Zhang, Chi Wang, and Qingyun Wu. Stateflow: Enhancing LLM task-solving through state-driven workflows. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=3nTbuygoop.

[730] Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Bb4VGOWELI.

[731] Eric Zelikman, Eliana Lorch, Lester Mackey, and Adam Tauman Kalai. Self-taught optimizer (stop): Recursively self-improving code generation. *arXiv preprint arXiv:2310.02304*, 2023.

[732] Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. *arXiv preprint arXiv:2309.16797*, 2023.

[733] Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=flNZJ2eOet.

[734] Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=0g0X4H8yN4I.

[735] Deqing Fu, Tian qi Chen, Robin Jia, and Vatsal Sharan. Transformers learn to achieve second-order convergence rates for in-context linear regression. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=L8h6cozcbn.

[736] Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

[737] Juhao Liang, Ziwei Wang, Zhuoheng Ma, Jianquan Li, Zhiyi Zhang, Xiangbo Wu, and Benyou Wang. Online training of large language models: Learn while chatting. *arXiv preprint arXiv:2403.04790*, 2024.

[738] Haotian Sun, Yuchen Zhuang, Lingkai Kong, Bo Dai, and Chao Zhang. Adaplanner: Adaptive planning from feedback with language models. *Advances in Neural Information Processing Systems*, 36, 2024.

[739] Zhiruo Wang, Daniel Fried, and Graham Neubig. Trove: Inducing verifiable and efficient toolboxes for solving programmatic tasks. *arXiv preprint arXiv:2401.12869*, 2024.

[740] Jianguo Zhang, Tian Lan, Ming Zhu, Zuxin Liu, Thai Hoang, Shirley Kokane, Weiran Yao, Juntao Tan, Akshara Prabhakar, Haolin Chen, et al. xlam: A family of large action models to empower ai agent systems. *arXiv preprint arXiv:2409.03215*, 2024.

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

[742] Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. Can LLMs Generate Novel Research Ideas? A Large-Scale Human Study with 100+ NLP Researchers, September 2024.

[743] Alireza Ghafarollahi and Markus J. Buehler. SciAgents: Automating Scientific Discovery Through Bioinspired Multi-Agent Intelligent Graph Reasoning. *Advanced Materials*, n/a(n/a):2413523, December 2024. ISSN 1521-4095. doi:10.1002/adma.202413523.

[744] Ievgeniia A. Tiukova, Daniel Brunnsåker, Erik Y. Bjurström, Alexander H. Gower, Filip Kronström, Gabriel K. Reder, Ronald S. Reiserer, Konstantin Korovin, Larisa B. Soldatova, John P. Wikswo, and Ross D. King. Genesis: Towards the Automation of Systems Biology Research, September 2024.

[745] Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI Scientist: Towards Fully Automated Open-Ended Scientific Discovery, September 2024.

[746] Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Zicheng Liu, and Emad Barsoum. Agent laboratory: Using LLM agents as research assistants. *arXiv preprint arXiv:2501.04227*, 2025.

[747] Xiangru Tang, Tianyu Hu, Muyang Ye, Yanjun Shao, Xunjian Yin, Siru Ouyang, Wangchunshu Zhou, Pan Lu, Zhuosheng Zhang, Yilun Zhao, Arman Cohan, and Mark Gerstein. ChemAgent: Self-updating Library in Large Language Models Improves Chemical Reasoning, January 2025.

[748] Malcolm Sim, Mohammad Ghazi Vakili, Felix Strieth-Kalthoff, Han Hao, Riley J. Hickman, Santiago Miret, Sergio Pablo-García, and Alán Aspuru-Guzik. ChemOS 2.0: An orchestration architecture for chemical self-driving laboratories. *Matter*, 7(9):2959–2977, September 2024. ISSN 2590-2393, 2590-2385. doi:10.1016/j.matt.2024.04.022.

[749] Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, Khaled Saab, Dan Popovici, Jacob Blum, Fan Zhang, Katherine Chou, Avinatan Hassidim, Burak Gokturk, Amin Vahdat, Pushmeet Kohli, Yossi Matias, Andrew Carroll, Kavita Kulkarni, Nenad Tomasev, Yuan Guan, Vikram Dhillon, Eeshit Dhaval Vaishnav, Byron Lee, Tiago R. D. Costa, José R. Penadés, Gary Peltz, Yunhan Xu, Annalisa Pawlosky, Alan Karthikesalingam, and Vivek Natarajan. Towards an AI co-scientist, February 2025.

[750] Tianwei Dai, Sriram Vijayakrishnan, Filip T. Szczypiński, Jean-François Ayme, Ehsan Simaei, Thomas Fellowes, Rob Clowes, Lyubomir Kotopanov, Caitlin E. Shields, Zhengxue Zhou, John W. Ward, and Andrew I. Cooper. Autonomous mobile robots for exploratory synthetic chemistry. *Nature*, pages 1–8, November 2024. ISSN 1476-4687. doi:10.1038/s41586-024-08173-7.

[751] Felix Strieth-Kalthoff, Han Hao, Vandana Rathore, Joshua Derasp, Théophile Gaudin, Nicholas H. Angello, Martin Seifrid, Ekaterina Trushina, Mason Guy, Junliang Liu, Xun Tang, Masashi Mamada, Wesley Wang, Tuul Tsagaantsooj, Cyrille Lavigne, Robert Pollice, Tony C. Wu, Kazuhiro Hotta, Leticia Bodo, Shangyu Li, Mohammad Haddadnia, Agnieszka Wołos, Rafał Roszak, Cher Tian Ser, Carlota Bozal-Ginesta, Riley J. Hickman, Jenya Vestfrid, Andrés Aguilar-Granda, Elena L. Klimareva, Ralph C. Sigerson, Wenduan Hou, Daniel Gahler, Slawomir Lach, Adrian Warzybok, Oleg Borodin, Simon Rohrbach, Benjamin Sanchez-Lengeling, Chihaya Adachi, Bartosz A. Grzybowski, Leroy Cronin, Jason E. Hein, Martin D. Burke, and Alán Aspuru-Guzik. Delocalized, asynchronous, closed-loop discovery of organic laser emitters. *Science*, 384(6697): eadk9227, May 2024. doi:10.1126/science.adk9227.

[752] Kyle Swanson, Wesley Wu, Nash L Bulaong, John E Pak, and James Zou. The virtual lab: Ai agents design new sars-cov-2 nanobodies with experimental validation. *bioRxiv*, pages 2024–11, 2024.

[753] Trieu H. Trinh, Yuhuai Wu, Quoc V. Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, January 2024. ISSN 1476-4687. doi:10.1038/s41586-023-06747-5.

[754] Haoyang Liu, Yijiang Li, Jinglin Jian, Yuxuan Cheng, Jianrong Lu, Shuyi Guo, Jinglei Zhu, Mianchen Zhang, Miantong Zhang, and Haohan Wang. Toward a Team of AI-made Scientists for Scientific Discovery from Gene Expression Data, February 2024.

[755] Sirui Hong, Yizhang Lin, Bang Liu, Bangbang Liu, Binhao Wu, Danyang Li, Jiaqi Chen, Jiayi Zhang, Jinlin Wang, Li Zhang, Lingyao Zhang, Min Yang, Mingchen Zhuge, Taicheng Guo, Tuo Zhou, Wei Tao, Wenyi Wang, Xiangru Tang, Xiangtao Lu, Xiawu Zheng, Xinbing Liang, Yaying Fei, Yuheng Cheng, Zongze Xu, and Chenglin Wu. Data Interpreter: An LLM Agent For Data Science, March 2024.

[756] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[757] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004.

[758] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

[759] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20(55):1–21, 2019. URL http://jmlr.org/papers/v20/18-598.html.

[760] Jiabin Liu, Fu Zhu, Chengliang Chai, Yuyu Luo, and Nan Tang. Automatic data acquisition for deep learning. *Proc. VLDB Endow.*, 14(12):2739–2742, 2021.

[761] Yuyu Luo, Nan Tang, Guoliang Li, Chengliang Chai, Wenbo Li, and Xuedi Qin. Synthesizing natural language to visualization (NL2VIS) benchmarks from NL2SQL benchmarks. In *SIGMOD Conference*, pages 1235–1247. ACM, 2021.

[762] Jiawei Tang, Yuyu Luo, Mourad Ouzzani, Guoliang Li, and Hongyang Chen. Sevi: Speech-to-visualization through neural machine translation. In *SIGMOD Conference*, pages 2353–2356. ACM, 2022.

[763] Bernd Bischl, Martin Binder, Michel Lang, Tobias Pielok, Jakob Richter, Stefan Coors, Janek Thomas, Theresa Ullmann, Marc Becker, Anne-Laure Boulesteix, et al. Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 13(2):e1484, 2023.

[764] Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola. Autogluon-tabular: Robust and accurate automl for structured data. *arXiv preprint arXiv:2003.06505*, 2020.

[765] Chi Wang, Qingyun Wu, Markus Weimer, and Erkang Zhu. Flaml: A fast and lightweight automl library. *Proceedings of Machine Learning and Systems*, 3:434–447, 2021.

[766] Shaokun Zhang, Feiran Jia, Chi Wang, and Qingyun Wu. Targeted hyperparameter optimization with lexicographic preferences over multiple objectives. In *The Eleventh international conference on learning representations*, 2023.

[767] Shaokun Zhang, Yiran Wu, Zhonghua Zheng, Qingyun Wu, and Chi Wang. Hypertime: Hyperparameter optimization for combating temporal distribution shifts. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 4610–4619, 2024.

[768] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys (CSUR)*, 54(4):1–34, 2021.

[769] Xiawu Zheng, Chenyi Yang, Shaokun Zhang, Yan Wang, Baochang Zhang, Yongjian Wu, Yunsheng Wu, Ling Shao, and Rongrong Ji. Ddpnas: Efficient neural architecture search via dynamic distribution pruning. *International Journal of Computer Vision*, 131(5):1234–1249, 2023.

[770] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[771] Tianjun Zhang, Xuezhi Wang, Denny Zhou, Dale Schuurmans, and Joseph E Gonzalez. Tempera: Test-time prompting via reinforcement learning. *arXiv preprint arXiv:2211.11890*, 2022.

[772] Lifan Yuan, Yangyi Chen, Xingyao Wang, Yi Fung, Hao Peng, and Heng Ji. CRAFT: Customizing LLMs by creating and retrieving from specialized toolsets. In *12th International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=G0vdDSt9XM.

[773] Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, et al. Aflow: Automating agentic workflow generation. *arXiv preprint arXiv:2410.10762*, 2024.

[774] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*, 2022.

[775] Wenyi Wang, Hisham A Alyahya, Dylan R Ashley, Oleg Serikov, Dmitrii Khizbullin, Francesco Faccio, and Jürgen Schmidhuber. How to correctly do semantic backpropagation on language-based agentic systems. *arXiv preprint arXiv:2412.03624*, 2024.

[776] Xuanchang Zhang, Zhuosheng Zhang, and Hai Zhao. Glape: Gold label-agnostic prompt evaluation and optimization for large language model. *CoRR*, abs/2402.02408, 2024.

[777] Xiaoqiang Lin, Zhongxiang Dai, Arun Verma, See-Kiong Ng, Patrick Jaillet, and Bryan Kian Hsiang Low. Prompt optimization with human feedback. *arXiv preprint arXiv:2405.17346*, 2024.

[778] Jinyu Xiang, Jiayi Zhang, Zhaoyang Yu, Fengwei Teng, Jinhao Tu, Xinbing Liang, Sirui Hong, Chenglin Wu, and Yuyu Luo. Self-supervised prompt optimization. *arXiv preprint arXiv:2502.06855*, 2025.

[779] Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search. In *EMNLP*, pages 7957–7968. Association for Computational Linguistics, 2023.

[780] Peiyan Zhang, Haibo Jin, Leyang Hu, Xinnuo Li, Liying Kang, Man Luo, Yangqiu Song, and Haohan Wang. Revolve: Optimizing ai systems by tracking response evolution in textual optimization. *arXiv preprint arXiv:2412.03092*, 2024.

[781] Mingchen Zhuge, Changsheng Zhao, Dylan Ashley, Wenyi Wang, Dmitrii Khizbullin, Yunyang Xiong, Zechun Liu, Ernie Chang, Raghuraman Krishnamoorthi, Yuandong Tian, et al. Agent-as-a-judge: Evaluate agents with agents. *arXiv preprint arXiv:2410.10934*, 2024.

[782] Cilin Yan, Jingyun Wang, Lin Zhang, Ruihui Zhao, Xiaopu Wu, Kai Xiong, Qingsong Liu, Guoliang Kang, and Yangyang Kang. Efficient and accurate prompt optimization: the benefit of memory in exemplar-guided reflection. *CoRR*, abs/2411.07446, 2024.

[783] Han Zhou, Xingchen Wan, Yinhong Liu, Nigel Collier, Ivan Vulic, and Anna Korhonen. Fairer preferences elicit improved human-aligned large language model judgments. In *EMNLP*, pages 1241–1252. Association for Computational Linguistics, 2024.

[784] Xingchen Wan, Ruoxi Sun, Hanjun Dai, Sercan Ö. Arik, and Tomas Pfister. Better zero-shot reasoning with self-adaptive prompting. In *ACL (Findings)*, pages 3493–3514. Association for Computational Linguistics, 2023.

[785] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *ACL (1)*, pages 8086–8098. Association for Computational Linguistics, 2022.

[786] Tal Ridnik, Dedy Kredo, and Itamar Friedman. Code generation with alphacodium: From prompt engineering to flow engineering. *CoRR*, abs/2401.08500, 2024.

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

[788] Yinjie Wang, Ling Yang, Guohao Li, Mengdi Wang, and Bryon Aragam. Scoreflow: Mastering LLM agent workflows via score-based preference optimization. *arXiv preprint arXiv:2502.04306*, 2025.

[789] Jon Saad-Falcon, Adrian Gamarra Lafuente, Shlok Natarajan, Nahum Maru, Hristo Todorov, Etash Guha, E. Kelly Buchanan, Mayee Chen, Neel Guha, Christopher Ré, and Azalia Mirhoseini. Archon: An architecture search framework for inference-time techniques, 2024. URL https://arxiv.org/abs/2409.15254.

[790] Zhi Rui Tam, Cheng-Kuang Wu, Yi-Lin Tsai, Chieh-Yen Lin, Hung-yi Lee, and Yun-Nung Chen. Let me speak freely? A study on the impact of format restrictions on performance of large language models. *CoRR*, abs/2408.02442, 2024.

[791] Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. Api-bank: A comprehensive benchmark for tool-augmented LLMs. *arXiv preprint arXiv:2304.08244*, 2023.

[792] Qiantong Xu, Fenglu Hong, Bo Li, Changran Hu, Zhengyu Chen, and Jian Zhang. On the tool manipulation capability of open-source large language models. *arXiv preprint arXiv:2305.16504*, 2023.

[793] Zhicheng Guo, Sijie Cheng, Hao Wang, Shihao Liang, Yujia Qin, Peng Li, Zhiyuan Liu, Maosong Sun, and Yang Liu. Stabletoolbench: Towards stable large-scale benchmarking on tool learning of large language models, 2024.

[794] Qiaoyu Tang, Ziliang Deng, Hongyu Lin, Xianpei Han, Qiao Liang, Boxi Cao, and Le Sun. Toolalpaca: Generalized tool learning for language models with 3000 simulated cases. *arXiv preprint arXiv:2306.05301*, 2023.

[795] Yangjun Ruan, Honghua Dong, Andrew Wang, Silviu Pitis, Yongchao Zhou, Jimmy Ba, Yann Dubois, Chris J Maddison, and Tatsunori Hashimoto. Identifying the risks of lm agents with an lm-emulated sandbox. *arXiv preprint arXiv:2309.15817*, 2023.

[796] Yue Huang, Jiawen Shi, Yuan Li, Chenrui Fan, Siyuan Wu, Qihui Zhang, Yixin Liu, Pan Zhou, Yao Wan, Neil Zhenqiang Gong, and Lichao Sun. Metatool benchmark for large language models: Deciding whether to use tools and which to use, 2024. URL https://arxiv.org/abs/2310.03128.

[797] Junjie Ye, Guanyu Li, Songyang Gao, Caishuang Huang, Yilong Wu, Sixian Li, Xiaoran Fan, Shihan Dou, Qi Zhang, Tao Gui, et al. Tooleyes: Fine-grained evaluation for tool learning capabilities of large language models in real-world scenarios. *arXiv preprint arXiv:2401.00741*, 2024.

[798] Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains, 2024. URL https://arxiv.org/abs/2406.12045.

[799] Xunjian Yin, Xinyi Wang, Liangming Pan, Xiaojun Wan, and William Yang Wang. G\" odel agent: A self-referential agent framework for recursive self-improvement. *arXiv preprint arXiv:2410.04444*, 2024.

[800] Han Zhou, Xingchen Wan, Ruoxi Sun, Hamid Palangi, Shariq Iqbal, Ivan Vulić, Anna Korhonen, and Sercan Ö. Arık. Multi-agent design: Optimizing agents with better prompts and topologies, 2025. URL https://arxiv.org/abs/2502.02533.

[801] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.

[802] James C Spall. *Introduction to stochastic search and optimization: estimation, simulation, and control*. John Wiley & Sons, 2005.

[803] Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

[804] Nikolaus Hansen. The cma evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*, 2016.

[805] Hao-Jun Michael Shi, Melody Qiming Xuan, Figen Oztoprak, and Jorge Nocedal. On the numerical performance of finite-difference-based methods for derivative-free optimization. *Optimization Methods and Software*, 38(2): 289–311, 2023.

[806] OpenAI. Openai o3-mini system card, 2025. URL https://openai.com/index/openai-o3-mini/. [Online; accessed 2025-02-02].

[807] Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. *arXiv preprint arXiv:2304.05128*, 2023.

[808] Qinyuan Ye, Maxamed Axmed, Reid Pryzant, and Fereshte Khani. Prompt engineering a prompt engineer. *arXiv preprint arXiv:2311.05661*, 2023.

[809] Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. *arXiv preprint arXiv:2310.11324*, 2023.

[810] Ruotian Ma, Xiaolei Wang, Xin Zhou, Jian Li, Nan Du, Tao Gui, Qi Zhang, and Xuanjing Huang. Are large language models good prompt optimizers? *arXiv preprint arXiv:2402.02101*, 2024.

[811] Ting-Yun Chang and Robin Jia. Data curation alone can stabilize in-context learning. *arXiv preprint arXiv:2212.10378*, 2022.

[812] Tai Nguyen and Eric Wong. In-context example selection with influences. *arXiv preprint arXiv:2302.11042*, 2023.

[813] Ching-An Cheng, Allen Nie, and Adith Swaminathan. Trace is the next autodiff: Generative optimization with rich feedback, execution traces, and LLMs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=rYs2Dmn9tD.

[814] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[815] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[816] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[817] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[818] Shizhe Diao, Zhichao Huang, Ruijia Xu, Xuechun Li, Yong Lin, Xiao Zhou, and Tong Zhang. Black-box prompt learning for pre-trained language models. *arXiv preprint arXiv:2201.08531*, 2022.

[819] Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*, 2022.

[820] Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, et al. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*, 2023.

[821] Krista Opsahl-Ong, Michael J Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. Optimizing instructions and demonstrations for multi-stage language model programs. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9340–9366, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.emnlp-main.525. URL https://aclanthology.org/2024.emnlp-main.525.

[822] Shuhei Watanabe. Tree-structured parzen estimator: Understanding its algorithm components and their roles for better empirical performance. *arXiv preprint arXiv:2304.11127*, 2023.

[823] Yongchao Chen, Jacob Arkin, Yilun Hao, Yang Zhang, Nicholas Roy, and Chuchu Fan. PRompt optimization in multi-step tasks (PROMST): Integrating human feedback and heuristic-based sampling. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3859–3920, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.emnlp-main.226. URL https://aclanthology.org/2024.emnlp-main.226.

[824] Brandon Amos et al. Tutorial on amortized optimization. *Foundations and Trends® in Machine Learning*, 16 (5):592–732, 2023.

[825] Anselm Paulus, Arman Zharmagambetov, Chuan Guo, Brandon Amos, and Yuandong Tian. Advprompter: Fast adaptive adversarial prompting for LLMs. *arXiv preprint arXiv:2404.16873*, 2024.

[826] Ollie Liu, Deqing Fu, Dani Yogatama, and Willie Neiswanger. DeLLMa: Decision making under uncertainty with large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=Acvo2RGSCy.

[827] Wenyue Hua, Ollie Liu, Lingyao Li, Alfonso Amayuelas, Julie Chen, Lucas Jiang, Mingyu Jin, Lizhou Fan, Fei Sun, William Wang, et al. Game-theoretic llm: Agent workflow for negotiation games. *arXiv preprint arXiv:2411.05990*, 2024.

[828] Sicheng Zhu, Brandon Amos, Yuandong Tian, Chuan Guo, and Ivan Evtimov. Advprefix: An objective for nuanced LLM jailbreaks. *arXiv preprint arXiv:2412.10321*, 2024.

[829] Luke Metz, C Daniel Freeman, James Harrison, Niru Maheswaranathan, and Jascha Sohl-Dickstein. Practical tradeoffs between memory, compute, and performance in learned optimizers. In *Conference on Lifelong Learning Agents*, pages 142–164. PMLR, 2022.

[830] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. *arXiv preprint arXiv:1807.03819*, 2018.

[831] Laurent Hascoet and Mauricio Araya-Polo. Enabling user-driven checkpointing strategies in reverse-mode automatic differentiation. *arXiv preprint cs/0606042*, 2006.

[832] Amirreza Shaban, Ching-An Cheng, Nathan Hatch, and Byron Boots. Truncated back-propagation for bilevel optimization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1723–1732. PMLR, 2019.

[833] Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, Joao Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 35151–35174. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/von-oswald23a.html.

[834] Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=RdJVFCHjUMI.

[835] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[836] Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*, 2022.

[837] Michael Hanna, Ollie Liu, and Alexandre Variengien. How does gpt-2 compute greater-than. *Interpreting mathematical abilities in a pre-trained language model*, 2:11, 2023.

[838] Arthur Conmy, Augustine Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, 36:16318–16352, 2023.

[839] Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. *arXiv preprint arXiv:2408.05147*, 2024.

[840] Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. *arXiv preprint arXiv:2406.04093*, 2024.

[841] Samuel Marks, Can Rager, Eric J Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. *arXiv preprint arXiv:2403.19647*, 2024.

[842] Cem Anil, Esin Durmus, Nina Rimsky, Mrinank Sharma, Joe Benton, Sandipan Kundu, Joshua Batson, Meg Tong, Jesse Mu, Daniel J Ford, et al. Many-shot jailbreaking. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[843] Michael Laskin, Luyu Wang, Junhyuk Oh, Emilio Parisotto, Stephen Spencer, Richie Steigerwald, DJ Strouse, Steven Stenberg Hansen, Angelos Filos, Ethan Brooks, maxime gazeau, Himanshu Sahni, Satinder Singh, and Volodymyr Mnih. In-context reinforcement learning with algorithm distillation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=hy0a5MMPUv.

[844] Allen Nie, Yi Su, Bo Chang, Jonathan N Lee, Ed H Chi, Quoc V Le, and Minmin Chen. Evolve: Evaluating and optimizing LLMs for exploration. *arXiv preprint arXiv:2410.06238*, 2024.

[845] Akshay Krishnamurthy, Keegan Harris, Dylan J Foster, Cyril Zhang, and Aleksandrs Slivkins. Can large language models explore in-context? *arXiv preprint arXiv:2403.15371*, 2024.

[846] Giovanni Monea, Antoine Bosselut, Kianté Brantley, and Yoav Artzi. Llms are in-context reinforcement learners. In *ICLR*, 2024.

[847] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

[848] Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for "mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008, 2023.

[849] Collin Zhang, John X Morris, and Vitaly Shmatikov. Extracting prompts by inverting LLM outputs. *arXiv preprint arXiv:2405.15012*, 2024.

[850] Hao Xiang, Bowen Yu, Hongyu Lin, Keming Lu, Yaojie Lu, Xianpei Han, Le Sun, Jingren Zhou, and Junyang Lin. Aligning large language models via self-steering optimization. *arXiv preprint arXiv:2410.17131*, 2024.

[851] Yizhang Zhu, Shiyin Du, Boyan Li, Yuyu Luo, and Nan Tang. Are large language models good statisticians? In *NeurIPS*, 2024.

[852] Tianqi Luo, Chuhan Huang, Leixian Shen, Boyan Li, Shuyu Shen, Wei Zeng, Nan Tang, and Yuyu Luo. nvbench 2.0: A benchmark for natural language to visualization under ambiguity, 2025. URL https://arxiv.org/abs/2503.12880.

[853] Teng Lin, Yizhang Zhu, Yuyu Luo, and Nan Tang. Srag: Structured retrieval-augmented generation for multi-entity question answering over wikipedia graph, 2025. URL https://arxiv.org/abs/2503.01346.

[854] Zhengxuan Zhang, Yin Wu, Yuyu Luo, and Nan Tang. Fine-grained retrieval-augmented generation for visual question answering, 2025. URL https://arxiv.org/abs/2502.20964.

[855] Mingye Zhu, Yi Liu, Lei Zhang, Junbo Guo, and Zhendong Mao. Lire: listwise reward enhancement for preference alignment. *arXiv preprint arXiv:2405.13516*, 2024.

[856] Julian Schrittwieser, Thomas Hubert, Amol Mandhane, Mohammadamin Barekatain, Ioannis Antonoglou, and David Silver. Online and offline reinforcement learning by planning with a learned model. *Advances in Neural Information Processing Systems*, 34:27580–27591, 2021.

[857] Sili Huang, Jifeng Hu, Zhejian Yang, Liwei Yang, Tao Luo, Hechang Chen, Lichao Sun, and Bo Yang. Decision mamba: Reinforcement learning via hybrid selective sequence modeling. *arXiv preprint arXiv:2406.00079*, 2024.

[858] Kun Lei, Zhengmao He, Chenhao Lu, Kaizhe Hu, Yang Gao, and Huazhe Xu. Uni-o4: Unifying online and offline deep reinforcement learning with multi-step on-policy optimization. *arXiv preprint arXiv:2311.03351*, 2023.

[859] Yoshua Bengio, Michael Cohen, Damiano Fornasiere, Joumana Ghosn, Pietro Greiner, Matt MacDermott, Sören Mindermann, Adam Oberman, Jesse Richardson, Oliver Richardson, Marc-Antoine Rondeau, Pierre-Luc St-Charles, and David Williams-King. Superintelligent Agents Pose Catastrophic Risks: Can Scientist AI Offer a Safer Path?, February 2025.

[860] Plato, Bernard Williams, M. J. Levett, and Myles Burnyeat. *Theaetetus*. Hackett Publishing, January 1992. ISBN 978-0-87220-158-3.

[861] Edmund L Gettier. Is Justified True Belief Knowledge? *Analysis*, June 1963. doi:10.1093/analys/23.6.121.

[862] Matthias Steup and Ram Neta. Epistemology. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2024 edition, 2024.

[863] E. T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003. ISBN 978-0-521-59271-0. doi:10.1017/CBO9780511790423.

[864] Thomas Parr, Giovanni Pezzulo, and Karl J. Friston. *Active Inference: The Free Energy Principle in Mind, Brain, and Behavior*. MIT Press, 2022.

[865] François Chollet. On the Measure of Intelligence, November 2019.

[866] Thomas M Cover and Joy A Thomas. *ELEMENTS OF INFORMATION THEORY*. John Wiley & Sons, April 2005.

[867] Raymond B. Cattell. Theory of fluid and crystallized intelligence: A critical experiment. *Journal of Educational Psychology*, 54(1):1–22, 1963. ISSN 1939-2176. doi:10.1037/h0046743.

[868] Raymond B. Cattell. *Abilities: Their Structure, Growth, and Action*. Houghton Mifflin, 1971. ISBN 978-0-395-04275-5.

[869] Alexandr Ten, Pramod Kaushik, Pierre-Yves Oudeyer, and Jacqueline Gottlieb. Humans monitor learning progress in curiosity-driven exploration. *Nature Communications*, 12(1):5972, October 2021. ISSN 2041-1723. doi:10.1038/s41467-021-26196-w.

[870] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A. Efros. Large-Scale Study of Curiosity-Driven Learning, August 2018.

[871] Eberhard O. Voit. Perspective: Dimensions of the scientific method. *PLOS Computational Biology*, 15(9): e1007279, September 2019. ISSN 1553-7358. doi:10.1371/journal.pcbi.1007279.

[872] Kjell Jørgen Hole and Subutai Ahmad. A thousand brains: Toward biologically constrained AI. *SN Applied Sciences*, 3(8):743, July 2021. ISSN 2523-3971. doi:10.1007/s42452-021-04715-0.

[873] Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, Vishal Dey, Mingyi Xue, Frazier N. Baker, Benjamin Burns, Daniel Adu-Ampratwum, Xuhui Huang, Xia Ning, Song Gao, Yu Su, and Huan Sun. ScienceAgentBench: Toward Rigorous Assessment of Language Agents for Data-Driven Scientific Discovery, October 2024.

[874] Michael H. Prince, Henry Chan, Aikaterini Vriza, Tao Zhou, Varuni K. Sastry, Yanqi Luo, Matthew T. Dearing, Ross J. Harder, Rama K. Vasudevan, and Mathew J. Cherukara. Opportunities for retrieval and tool augmented large language models in scientific facilities. *npj Computational Materials*, 10(1):1–8, November 2024. ISSN 2057-3960. doi:10.1038/s41524-024-01423-2.

[875] Karl Raimund Popper. *Conjectures and Refutations: The Growth of Scientific Knowledge*. Routledge, 1962.

[876] Karl R. Popper. *The Logic of Scientific Discovery*. Routledge Classics. Routledge, repr. 2008 (twice) edition, 2008. ISBN 978-0-415-27843-0 978-0-415-27844-7.

[877] Donald A. Gillies. Popper and computer induction. *BioEssays*, 23(9):859–860, 2001. ISSN 1521-1878. doi:10.1002/bies.1123.

[878] Yiqiao Jin, Qinlin Zhao, Yiyang Wang, Hao Chen, Kaijie Zhu, Yijia Xiao, and Jindong Wang. Agentreview: Exploring peer review dynamics with llm agents. In *EMNLP*, 2024.

[879] Haoyang Su, Renqi Chen, Shixiang Tang, Xinzhe Zheng, Jingzhe Li, Zhenfei Yin, Wanli Ouyang, and Nanqing Dong. Two Heads Are Better Than One: A Multi-Agent System Has the Potential to Improve Scientific Idea Generation, October 2024.

[880] Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. ResearchAgent: Iterative Research Idea Generation over Scientific Literature with Large Language Models, April 2024.

[881] Alexander H. Gower, Konstantin Korovin, Daniel Brunnsåker, Ievgeniia A. Tiukova, and Ross D. King. LGEM+: A First-Order Logic Framework for Automated Improvement of Metabolic Network Models Through Abduction. In Albert Bifet, Ana Carolina Lorena, Rita P. Ribeiro, João Gama, and Pedro H. Abreu, editors, *Discovery Science*, pages 628–643. Springer Nature Switzerland, 2023. ISBN 978-3-031-45275-8. doi:10.1007/978-3-031-45275-8_42.

[882] Anthony Coutant, Katherine Roper, Daniel Trejo-Banos, Dominique Bouthinon, Martin Carpenter, Jacek Grzebyta, Guillaume Santini, Henry Soldano, Mohamed Elati, Jan Ramon, Celine Rouveirol, Larisa N. Soldatova, and Ross D. King. Closed-loop cycles of experiment design, execution, and learning accelerate systems biology model development in yeast. *Proceedings of the National Academy of Sciences*, 116(36): 18142–18147, September 2019. doi:10.1073/pnas.1900548116.

[883] Xiaoxuan Wang, Ziniu Hu, Pan Lu, Yanqiao Zhu, Jieyu Zhang, Satyen Subramaniam, Arjun R. Loomba, Shichang Zhang, Yizhou Sun, and Wei Wang. SciBench: Evaluating College-Level Scientific Problem-Solving Abilities of Large Language Models, June 2024.

[884] Haorui Wang, Marta Skreta, Cher-Tian Ser, Wenhao Gao, Lingkai Kong, Felix Strieth-Kalthoff, Chenru Duan, Yuchen Zhuang, Yue Yu, Yanqiao Zhu, Yuanqi Du, Alán Aspuru-Guzik, Kirill Neklyudov, and Chao Zhang. Efficient Evolutionary Search Over Chemical Space with Large Language Models, July 2024.

[885] Shuyi Jia, Chao Zhang, and Victor Fung. LLMatDesign: Autonomous Materials Discovery with Large Language Models, June 2024.

[886] Holland Hysmith, Elham Foadian, Shakti P. Padhy, Sergei V. Kalinin, Rob G. Moore, Olga S. Ovchinnikova, and Mahshid Ahmadi. The future of self-driving laboratories: From human in the loop interactive AI to gamification. *Digital Discovery*, 3(4):621–636, 2024. doi:10.1039/D4DD00040D.

[887] Yijia Xiao, Wanjia Zhao, Junkai Zhang, Yiqiao Jin, Han Zhang, Zhicheng Ren, Renliang Sun, Haixin Wang, Guancheng Wan, Pan Lu, et al. Protein large language models: A comprehensive survey. *arXiv:2502.17504*, 2025.

[888] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, March 2023. doi:10.1126/science.ade2574.

[889] Richard Evans, Michael O'Neill, Alexander Pritzel, Natasha Antropova, Andrew Senior, Tim Green, Augustin Žídek, Russ Bates, Sam Blackwell, Jason Yim, Olaf Ronneberger, Sebastian Bodenstein, Michal Zielinski, Alex Bridgland, Anna Potapenko, Andrew Cowie, Kathryn Tunyasuvunakool, Rishub Jain, Ellen Clancy, Pushmeet Kohli, John Jumper, and Demis Hassabis. Protein complex prediction with AlphaFold-Multimer, October 2021.

[890] Veda Sheersh Boorla, Ratul Chowdhury, Ranjani Ramasubramanian, Brandon Ameglio, Rahel Frick, Jeffrey J. Gray, and Costas D. Maranas. De novo design and Rosetta-based assessment of high-affinity antibody variable regions (Fv) against the SARS-CoV-2 spike receptor binding domain (RBD). *Proteins: Structure, Function, and Bioinformatics*, 91(2):196–208, 2023. ISSN 1097-0134. doi:10.1002/prot.26422.

[891] Jiefu Ou, Arda Uzunoglu, Benjamin Van Durme, and Daniel Khashabi. WorldAPIs: The World Is Worth How Many APIs? A Thought Experiment, July 2024.

[892] Yuxing Fei, Bernardus Rendy, Rishi Kumar, Olympia Dartsi, Hrushikesh P. Sahasrabuddhe, Matthew J. McDermott, Zheren Wang, Nathan J. Szymanski, Lauren N. Walters, David Milsted, Yan Zeng, Anubhav Jain, and Gerbrand Ceder. AlabOS: A Python-based reconfigurable workflow management framework for autonomous laboratories. *Digital Discovery*, 3(11):2275–2288, November 2024. ISSN 2635-098X. doi:10.1039/D4DD00129J.

[893] Andrew D McNaughton, Gautham Krishna Sankar Ramalaxmi, Agustin Kruel, Carter R Knutson, Rohith A Varikoti, and Neeraj Kumar. CACTUS: Chemistry agent connecting tool usage to science. *ACS Omega*, 9(46): 46563–46573, 2024.

[894] Rafael Vescovi, Tobias Ginsburg, Kyle Hippe, Doga Ozgulbas, Casey Stone, Abraham Stroka, Rory Butler, Ben Blaiszik, Tom Brettin, Kyle Chard, Mark Hereld, Arvind Ramanathan, Rick Stevens, Aikaterini Vriza, Jie Xu, Qingteng Zhang, and Ian Foster. Towards a modular architecture for science factories. *Digital Discovery*, 2 (6):1980–1998, 2023.

[895] Daniil A. Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578, December 2023. ISSN 1476-4687. doi:10.1038/s41586-023-06792-0.

[896] Emerald Cloud Lab. ECL Documentation. https://www.emeraldcloudlab.com/documentation/objects/, 2025.

[897] Jiafei Duan, Samson Yu, Hui Li Tan, Hongyuan Zhu, and Cheston Tan. A Survey of Embodied AI: From Simulators to Research Tasks, January 2022.

[898] Rafael Vescovi, Ryan Chard, Nickolaus D Saint, Ben Blaiszik, Jim Pruyne, Tekin Bicer, Alex Lavens, Zhengchun Liu, Michael E Papka, Suresh Narayanan, Nicholas Schwarz, Kyle Chard, and Ian T. Foster. Linking scientific instruments and computation: Patterns, technologies, and experiences. *Patterns*, 3(10), 2022.

[899] Doga Yamac Ozgulbas, Don Jensen Jr, Rory Butler, Rafael Vescovi, Ian T Foster, Michael Irvin, Yasukazu Nakaye, Miaoqi Chu, Eric M Dufresne, Soenke Seifert, et al. Robotic pendant drop: Containerless liquid for μs-resolved, AI-executable XPCS. *Light: Science & Applications*, 12(1):196, 2023.

[900] Chandima Fernando, Daniel Olds, Stuart I Campbell, and Phillip M Maffettone. Facile integration of robots into experimental orchestration at scientific user facilities. In *IEEE International Conference on Robotics and Automation*, pages 9578–9584. IEEE, 2024.

[901] Stanley Lo, Sterling G Baird, Joshua Schrier, Ben Blaiszik, Nessa Carson, Ian Foster, Andrés Aguilar-Granda, Sergei V Kalinin, Benji Maruyama, Maria Politi, Helen Tran, Taylor D. Sparks, and Alan Aspuru-Guzik. Review of low-cost self-driving laboratories in chemistry and materials science: The "frugal twin" concept. *Digital Discovery*, 3(5):842–868, 2024.

[902] David Abel, André Barreto, Michael Bowling, Will Dabney, Shi Dong, Steven Hansen, Anna Harutyunyan, Khimya Khetarpal, Clare Lyle, Razvan Pascanu, Georgios Piliouras, Doina Precup, Jonathan Richens, Mark Rowland, Tom Schaul, and Satinder Singh. Agency Is Frame-Dependent, February 2025.

[903] Elliot Glazer, Ege Erdil, Tamay Besiroglu, Diego Chicharro, Evan Chen, Alex Gunning, Caroline Falkman Olsson, Jean-Stanislas Denain, Anson Ho, Emily de Oliveira Santos, Olli Järviniemi, Matthew Barnett, Robert Sandler, Matej Vrzala, Jaime Sevilla, Qiuyu Ren, Elizabeth Pratt, Lionel Levine, Grant Barkley, Natalie Stewart, Bogdan Grechuk, Tetiana Grechuk, Shreepranav Varma Enugandla, and Mark Wildon. FrontierMath: A Benchmark for Evaluating Advanced Mathematical Reasoning in AI, December 2024.

[904] Solim LeGris, Wai Keen Vong, Brenden M. Lake, and Todd M. Gureckis. H-ARC: A Robust Estimate of Human Performance on the Abstraction and Reasoning Corpus Benchmark, September 2024.

[905] Junjie Wu, Mo Yu, Lemao Liu, Dit-Yan Yeung, and Jie Zhou. Understanding LLMs' Fluid Intelligence Deficiency: An Analysis of the ARC Task, February 2025.

[906] Zeyuan Allen-Zhu and Xiaoli Xu. DOGE: Reforming AI Conferences and Towards a Future Civilization of Fairness and Justice, February 2025.

[907] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models, January 2023.

[908] Andrew D. White, Glen M. Hocky, Heta A. Gandhi, Mehrad Ansari, Sam Cox, Geemi P. Wellawatte, Subarna Sasmal, Ziyue Yang, Kangxin Liu, Yuvraj Singh, and Willmor J. Peña Ccoa. Assessment of chemistry knowledge in large language models that generate code. *Digital Discovery*, 2(2):368–376, 2023. ISSN 2635-098X. doi:10.1039/D2DD00087C.

[909] Botao Yu, Frazier N Baker, Ziru Chen, Garrett Herb, Boyu Gou, Daniel Adu-Ampratwum, Xia Ning, and Huan Sun. Tooling or not tooling? the impact of tools on language agents for chemistry problem solving. *arXiv preprint arXiv:2411.07228*, 2024.

[910] Franck Cappello, Sandeep Madireddy, Robert Underwood, Neil Getty, Nicholas Lee-Ping Chia, Nesar Ramachandra, Josh Nguyen, Murat Keceli, Tanwi Mallick, Zilinghan Li, Marieme Ngom, Chenhui Zhangx, Angel Yanguas-Gilxi, Evan Antoniuk, Bhavya Kailkhura, Minyang Tian, Yufeng Du, Yuan-Sen Ting, Azton Wells, Bogdan Nicolae, Avinash Maurya, M. Mustafa Rafique, Eliu Huerta, Bo Li, Ian Foster, and Rick Stevens. EAIRA: Establishing a methodology for evaluating AI models as scientific research assistants. *arXiv preprint arXiv:2502.20309*, 2025.

[911] Paul Raccuglia, Katherine C. Elbert, Philip D. F. Adler, Casey Falk, Malia B. Wenny, Aurelio Mollo, Matthias Zeller, Sorelle A. Friedler, Joshua Schrier, and Alexander J. Norquist. Machine-learning-assisted materials discovery using failed experiments. *Nature*, 533(7601):73–76, May 2016. ISSN 1476-4687. doi:10.1038/nature17439.

[912] OpenAI. Introducing deep research. https://openai.com/index/introducing-deep-research/, 2025.

[913] Steven N. Goodman. Introduction to Bayesian methods I: Measuring the strength of evidence. *Clinical Trials (London, England)*, 2(4):282–290; discussion 301–304, 364–378, 2005. ISSN 1740-7745. doi:10.1191/1740774505cn098oa.

[914] Xinyi Li, Sai Wang, Siqi Zeng, Yu Wu, and Yi Yang. A survey on llm-based multi-agent systems: Workflow, infrastructure, and challenges. *Vicinagearth*, 1(1):9, 10 2024. doi:10.1007/s44336-024-00009-2. URL https://doi.org/10.1007/s44336-024-00009-2.

[915] James Surowiecki. The wisdom of crowds. *Surowiecki, J*, 2005.

[916] Chris Frith and Uta Frith. Theory of mind. *Current biology*, 15(17):R644–R645, 2005.

[917] Huao Li, Yu Quan Chong, Simon Stepputtis, Joseph Campbell, Dana Hughes, Michael Lewis, and Katia Sycara. Theory of mind for multi-agent collaboration via large language models. *arXiv preprint arXiv:2310.10701*, 2023.

[918] Justin Chih-Yao Chen, Swarnadeep Saha, and Mohit Bansal. Reconcile: Round-table conference improves reasoning via consensus among diverse LLMs. *arXiv preprint arXiv:2309.13007*, 2023.

[919] Yihuai Lan, Zhiqiang Hu, Lei Wang, Yang Wang, De-Yong Ye, Peilin Zhao, Ee-Peng Lim, Hui Xiong, and Hao Wang. Llm-based agent society investigation: Collaboration and confrontation in avalon gameplay. In *Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://api.semanticscholar.org/CorpusID:264436387.

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

[921] Junkai Li, Siyu Wang, Meng Zhang, Weitao Li, Yunghwei Lai, Xinhui Kang, Weizhi Ma, and Yang Liu. Agent hospital: A simulacrum of hospital with evolvable medical agents. *arXiv preprint arXiv:2405.02957*, 2024.

[922] Xiangru Tang, Anni Zou, Zhuosheng Zhang, Ziming Li, Yilun Zhao, Xingyao Zhang, Arman Cohan, and Mark Gerstein. MedAgents: Large language models as collaborators for zero-shot medical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 599–621, Bangkok, Thailand, 2024.

[923] Hao Wei, Jianing Qiu, Haibao Yu, and Wu Yuan. Medco: Medical education copilots based on a multi-agent framework. *arXiv preprint arXiv:2408.12496*, 2024.

[924] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B Tenenbaum, Tianmin Shu, and Chuang Gan. Building cooperative embodied agents modularly with large language models. *arXiv preprint arXiv:2307.02485*, 2023.

[925] Yubo Dong, Xukun Zhu, Zhengzhe Pan, Linchao Zhu, and Yi Yang. VillagerAgent: A graph-based multi-agent framework for coordinating complex task dependencies in Minecraft. In *Findings of the Association for Computational Linguistics: ACL 2024*, 2024.

[926] Saaket Agashe, Yue Fan, Anthony Reyna, and Xin Eric Wang. Llm-coordination: evaluating and analyzing multi-agent coordination abilities in large language models. *arXiv preprint arXiv:2310.03903*, 2023.

[927] Jiaqi Chen, Yuxian Jiang, Jiachen Lu, and Li Zhang. S-agents: self-organizing agents in open-ended environment. *arXiv preprint arXiv:2402.04578*, 2024.

[928] Xiao Liu, Tianjie Zhang, Yu Gu, Iat Long Iong, Yifan Xu, Xixuan Song, Shudan Zhang, Hanyu Lai, Xinyi Liu, Hanlin Zhao, et al. Visualagentbench: Towards large multimodal models as visual foundation agents. In *The Thirteenth International Conference on Learning Representations*, 2025.

[929] Bo Qiao, Liqun Li, Xu Zhang, Shilin He, Yu Kang, Chaoyun Zhang, Fangkai Yang, Hang Dong, Jue Zhang, Lu Wang, Minghua Ma, Pu Zhao, Si Qin, Xiaoting Qin, Chao Du, Yong Xu, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. Taskweaver: A code-first agent framework, 2024. URL https://arxiv.org/abs/2311.17541.

[930] Wannita Takerngsaksiri, Jirat Pasuksmit, Patanamon Thongtanunam, Chakkrit Tantithamthavorn, Ruixiong Zhang, Fan Jiang, Jing Li, Evan Cook, Kun Chen, and Ming Wu. Human-in-the-loop software development agents, 2025. URL https://arxiv.org/abs/2411.12924.

[931] Anthropic. Model context protocol, 2025. URL https://www.anthropic.com/news/model-context-protocol. Accessed: 2025-01-07.

[932] Samuele Marro, Emanuele La Malfa, Jesse Wright, Guohao Li, Nigel Shadbolt, Michael Wooldridge, and Philip Torr. A scalable communication protocol for networks of large language models, 2024. URL https://arxiv.org/abs/2410.11905.

[933] Weize Chen, Ziming You, Ran Li, Yitong Guan, Chen Qian, Chenyang Zhao, Cheng Yang, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. Internet of agents: Weaving a web of heterogeneous agents for collaborative intelligence, 2024. URL https://arxiv.org/abs/2407.07061.

[934] Gabriel Mukobi, Hannah Erlebach, Niklas Lauffer, Lewis Hammond, Alan Chan, and Jesse Clifton. Welfare diplomacy: Benchmarking language model cooperation. *ArXiv*, abs/2310.08901, 2023. URL https://api.semanticscholar.org/CorpusID:264127980.

[935] Dong Chen, Shaoxin Lin, Muhan Zeng, Daoguang Zan, Jian-Gang Wang, Anton Cheshkov, Jun Sun, Hao Yu, Guoliang Dong, Artem Aliev, et al. Coder: Issue resolving with multi-agent and task graphs. *arXiv preprint arXiv:2406.01304*, 2024.

[936] Ziyi Yang, Zaibin Zhang, Zirui Zheng, Yuxian Jiang, Ziyue Gan, Zhiyu Wang, Zijian Ling, Jinsong Chen, Martz Ma, Bowen Dong, et al. Oasis: Open agents social interaction simulations on one million agents. *arXiv preprint arXiv:2411.11581*, 2024.

[937] Joanne Leong, John Tang, Edward Cutrell, Sasa Junuzovic, Gregory Paul Baribault, and Kori Inkpen. Dittos: Personalized, embodied agents that participate in meetings when you are unavailable. *Proc. ACM Hum.-Comput. Interact.*, 8(CSCW2), November 2024.

[938] Ge Gao, Alexey Taymanov, Eduardo Salinas, Paul Mineiro, and Dipendra Misra. Aligning LLM agents by learning latent preference from user edits, 2024. URL https://arxiv.org/abs/2404.15269.

[939] Jacob Austin, Augustus Odena, Maxwell I. Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie J. Cai, Michael Terry, Quoc V. Le, and Charles Sutton. Program synthesis with large language models. *CoRR*, abs/2108.07732, 2021. URL https://arxiv.org/abs/2108.07732.

[940] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2369–2380. Association for Computational Linguistics, 2018. doi:10.18653/V1/D18-1259. URL https://doi.org/10.18653/v1/d18-1259.

[941] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

[942] Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are NLP models really able to solve simple math word problems? In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tür, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 2080–2094. Association for Computational Linguistics, 2021. doi:10.18653/V1/2021.NAACL-MAIN.168. URL https://doi.org/10.18653/v1/2021.naacl-main.168.

[943] Subhro Roy and Dan Roth. Solving general arithmetic word problems. *CoRR*, abs/1608.01413, 2016. URL http://arxiv.org/abs/1608.01413.

[944] Haochen Sun, Shuwen Zhang, Lei Ren, Hao Xu, Hao Fu, Caixia Yuan, and Xiaojie Wang. Collab-overcooked: Benchmarking and evaluating large language models as collaborative agents, 2025. URL https://arxiv.org/abs/2502.20073.

[945] Longling Geng and Edward Y. Chang. Realm-bench: A real-world planning benchmark for llms and multi-agent systems, 2025. URL https://arxiv.org/abs/2502.18836.

[946] Matthew Chang, Gunjan Chhablani, Alexander Clegg, Mikael Dallaire Cote, Ruta Desai, Michal Hlavac, Vladimir Karashchuk, Jacob Krantz, Roozbeh Mottaghi, Priyam Parashar, Siddharth Patki, Ishita Prasad, Xavier Puig, Akshara Rai, Ram Ramrakhya, Daniel Tran, Joanne Truong, John M. Turner, Eric Undersander, and Tsung-Yen Yang. Partnr: A benchmark for planning and reasoning in embodied multi-agent tasks, 2024.

[947] Ruochen Zhao, Wenxuan Zhang, Yew Ken Chia, Weiwen Xu, Deli Zhao, and Lidong Bing. Auto-arena: Automating llm evaluations with agent peer battles and committee discussions, 2024. URL https://arxiv.org/abs/2405.20267.

[948] Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Zhe Wang, Zhenhailong Wang, Cheng Qian, Xiangru Tang, Heng Ji, and Jiaxuan You. Multiagentbench: Evaluating the collaboration and competition of llm agents, 2025. URL https://arxiv.org/abs/2503.01935.

[949] Yadong Zhang, Shaoguang Mao, Tao Ge, Xun Wang, Adrian de Wynter, Yan Xia, Wenshan Wu, Ting Song, Man Lan, and Furu Wei. Llm as a mastermind: A survey of strategic reasoning with large language models. *arXiv preprint arXiv:2404.01230*, 2024.

[950] Alonso Silva. Large language models playing mixed strategy nash equilibrium games. In *International Conference on Network Games, Artificial Intelligence, Control and Optimization*, pages 142–152. Springer, 2024.

[951] John J Horton. Large language models as simulated economic agents: What can we learn from homo silicus? Technical report, National Bureau of Economic Research, 2023.

[952] Ian Gemp, Yoram Bachrach, Marc Lanctot, Roma Patel, Vibhavari Dasagi, Luke Marris, Georgios Piliouras, Siqi Liu, and Karl Tuyls. States as strings as strategies: Steering language models with game-theoretic solvers. *arXiv preprint arXiv:2402.01704*, 2024.

[953] Shaoguang Mao, Yuzhe Cai, Yan Xia, Wenshan Wu, Xun Wang, Fengyi Wang, Tao Ge, and Furu Wei. Alympics: Llm agents meet game theory–exploring strategic decision-making with ai agents. *arXiv preprint arXiv:2311.03220*, 2023.

[954] Elif Akata, Lion Schulz, Julian Coda-Forno, Seong Joon Oh, Matthias Bethge, and Eric Schulz. Playing repeated games with large language models. *arXiv preprint arXiv:2305.16867*, 2023.

[955] Lin Xu, Zhiyuan Hu, Daquan Zhou, Hongyu Ren, Zhen Dong, Kurt Keutzer, See Kiong Ng, and Jiashi Feng. Magic: Investigation of large language model powered multi-agent in cognition, adaptability, rationality and collaboration, 2024. URL https://arxiv.org/abs/2311.08562.

[956] Kanishk Gandhi, Dorsa Sadigh, and Noah D Goodman. Strategic reasoning with language models. *arXiv preprint arXiv:2305.19165*, 2023.

[957] Jinhao Duan, Renming Zhang, James Diffenderfer, Bhavya Kailkhura, Lichao Sun, Elias Stengel-Eskin, Mohit Bansal, Tianlong Chen, and Kaidi Xu. Gtbench: Uncovering the strategic reasoning limitations of llms via game-theoretic evaluations. *arXiv preprint arXiv:2402.12348*, 2024.

[958] Nian Li, Chen Gao, Yong Li, and Qingmin Liao. Large language model-empowered agents for simulating macroeconomic activities. *Available at SSRN 4606937*, 2023.

[959] Qinlin Zhao, Jindong Wang, Yixuan Zhang, Yiqiao Jin, Kaijie Zhu, Hao Chen, and Xing Xie. Competeai: Understanding the competition behaviors in large language model-based agents. In *ICML*, 2024.

[960] Tian Xia, Zhiwei He, Tong Ren, Yibo Miao, Zhuosheng Zhang, Yang Yang, and Rui Wang. Measuring bargaining abilities of llms: A benchmark and a buyer-enhancement method. *arXiv preprint arXiv:2402.15813*, 2024.

[961] Karthik Sreedhar and Lydia Chilton. Simulating human strategic behavior: Comparing single and multi-agent llms. *ArXiv*, abs/2402.08189, 2024. URL https://api.semanticscholar.org/CorpusID:267636591.

[962] Ryan Y Lin, Siddhartha Ojha, Kevin Cai, and Maxwell F Chen. Strategic collusion of LLM agents: Market division in multi-commodity competitions. *arXiv preprint arXiv:2410.00031*, 2024.

[963] Giorgio Piatti, Zhijing Jin, Max Kleiman-Weiner, Bernhard Schölkopf, Mrinmaya Sachan, and Rada Mihalcea. Cooperate or collapse: Emergence of sustainable cooperation in a society of LLM agents. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[964] Zelai Xu, Chao Yu, Fei Fang, Yu Wang, and Yi Wu. Language agents with reinforcement learning for strategic play in the werewolf game. *arXiv preprint arXiv:2310.18940*, 2023.

[965] Silin Du and Xiaowei Zhang. Helmsman of the masses? evaluate the opinion leadership of large language models in the werewolf game. *arXiv preprint arXiv:2404.01602*, 2024.

[966] Xuanfa Jin, Ziyan Wang, Yali Du, Meng Fang, Haifeng Zhang, and Jun Wang. Learning to discuss strategically: A case study on one night ultimate werewolf. *arXiv preprint arXiv:2405.19946*, 2024.

[967] Simon Stepputtis, Joseph Campbell, Yaqi Xie, Zhengyang Qi, Wenxin Sharon Zhang, Ruiyi Wang, Sanketh Rangreji, Michael Lewis, and Katia Sycara. Long-horizon dialogue understanding for role identification in the game of avalon with large language models. *arXiv preprint arXiv:2311.05720*, 2023.

[968] Shenzhi Wang, Chang Liu, Zilong Zheng, Siyuan Qi, Shuo Chen, Qisen Yang, Andrew Zhao, Chaofei Wang, Shiji Song, and Gao Huang. Avalon's game of thoughts: Battle against deception through recursive contemplation. *arXiv preprint arXiv:2310.01320*, 2023.

[969] Zijing Shi, Meng Fang, Shunfeng Zheng, Shilong Deng, Ling Chen, and Yali Du. Cooperation on the fly: Exploring language agents for ad hoc teamwork in the avalon game. *arXiv preprint arXiv:2312.17515*, 2023.

[970] Dekun Wu, Haochen Shi, Zhiyuan Sun, and Bang Liu. Deciphering digital detectives: Understanding LLM behaviors and capabilities in multi-agent mystery games. *arXiv preprint arXiv:2312.00746*, 2023.

[971] Yuzhuang Xu, Shuo Wang, Peng Li, Fuwen Luo, Xiaolong Wang, Weidong Liu, and Yang Liu. Exploring large language models for communication games: An empirical study on werewolf, 2024. URL https://arxiv.org/abs/2309.04658.

[972] Jonathan Light, Min Cai, Sheng Shen, and Ziniu Hu. Avalonbench: Evaluating LLMs playing the game of avalon, 2023. URL https://arxiv.org/abs/2310.05036.

[973] Wenyue Hua, Lizhou Fan, Lingyao Li, Kai Mei, Jianchao Ji, Yingqiang Ge, Libby Hemphill, and Yongfeng Zhang. War and peace (waragent): Large language model-based multi-agent simulation of world wars. *arXiv preprint arXiv:2311.17227*, 2023.

[974] Mingyu Jin, Beichen Wang, Zhaoqian Xue, Suiyuan Zhu, Wenyue Hua, Hua Tang, Kai Mei, Mengnan Du, and Yongfeng Zhang. What if LLMs have different world views: Simulating alien civilizations with llm-based agents. *arXiv preprint arXiv:2402.13184*, 2024.

[975] Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *Humanities and Social Sciences Communications*, 11(1):1–24, 2024.

[976] Nian Li, Chen Gao, Mingyu Li, Yong Li, and Qingmin 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)*, pages 15523–15536, 2024.

[977] Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D Nguyen. Multi-agent collaboration mechanisms: A survey of llms. *arXiv preprint arXiv:2501.06322*, 2025.

[978] Xinnong Zhang, Jiayu Lin, Libo Sun, Weihong Qi, Yihang Yang, Yue Chen, Hanjia Lyu, Xinyi Mou, Siming Chen, Jiebo Luo, et al. Electionsim: Massive population election simulation powered by large language model driven agents. *arXiv preprint arXiv:2410.20746*, 2024.

[979] Antonino Ferraro, Antonio Galli, Valerio La Gatta, Marco Postiglione, Gian Marco Orlando, Diego Russo, Giuseppe Riccio, Antonio Romano, and Vincenzo Moscato. Agent-based modelling meets generative AI in social network simulations. *arXiv preprint arXiv:2411.16031*, 2024.

[980] Yun-Shiuan Chuang, Agam Goyal, Nikunj Harlalka, Siddharth Suresh, Robert Hawkins, Sijia Yang, Dhavan Shah, Junjie Hu, and Timothy T Rogers. Simulating opinion dynamics with networks of LLM-based agents. *arXiv preprint arXiv:2311.09618*, 2023.

[981] Yuhan Liu, Xiuying Chen, Xiaoqing Zhang, Xing Gao, Ji Zhang, and Rui Yan. From skepticism to acceptance: Simulating the attitude dynamics toward fake news. *arXiv preprint arXiv:2403.09498*, 2024.

[982] Jiakai Tang, Heyang Gao, Xuchen Pan, Lei Wang, Haoran Tan, Dawei Gao, Yushuo Chen, Xu Chen, Yankai Lin, Yaliang Li, et al. Gensim: A general social simulation platform with large language model based agents. *arXiv preprint arXiv:2410.04360*, 2024.

[983] Chen Qian, Xin Cong, Cheng Yang, Weize Chen, Yusheng Su, Juyuan Xu, Zhiyuan Liu, and Maosong Sun. Communicative agents for software development. *arXiv preprint arXiv:2307.07924*, 6(3), 2023.

[984] Xudong Guo, Kaixuan Huang, Jiale Liu, Wenhui Fan, Natalia Vélez, Qingyun Wu, Huazheng Wang, Thomas L. Griffiths, and Mengdi Wang. Embodied LLM agents learn to cooperate in organized teams, 2024. URL https://arxiv.org/abs/2403.12482.

[985] Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*, 2023.

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

[987] Jingyun Sun, Chengxiao Dai, Zhongze Luo, Yangbo Chang, and Yang Li. Lawluo: A multi-agent collaborative framework for multi-round chinese legal consultation, 2024. URL https://arxiv.org/abs/2407.16252.

[988] Wenhao Yu, Jie Peng, Yueliang Ying, Sai Li, Jianmin Ji, and Yanyong Zhang. Mhrc: Closed-loop decentralized multi-heterogeneous robot collaboration with large language models, 2024. URL https://arxiv.org/abs/2409.16030.

[989] Altera. AL, Andrew Ahn, Nic Becker, Stephanie Carroll, Nico Christie, Manuel Cortes, Arda Demirci, Melissa Du, Frankie Li, Shuying Luo, Peter Y Wang, Mathew Willows, Feitong Yang, and Guangyu Robert Yang. Project sid: Many-agent simulations toward AI civilization, 2024. URL https://arxiv.org/abs/2411.00114.

[990] Ryosuke Takata, Atsushi Masumori, and Takashi Ikegami. Spontaneous emergence of agent individuality through social interactions in llm-based communities, 2024. URL https://arxiv.org/abs/2411.03252.

[991] Shubham Gandhi, Manasi Patwardhan, Lovekesh Vig, and Gautam Shroff. Budgetmlagent: A cost-effective llm multi-agent system for automating machine learning tasks, 2025. URL https://arxiv.org/abs/2411.07464.

[992] Yuxing Lu and Jinzhuo Wang. Karma: Leveraging multi-agent llms for automated knowledge graph enrichment, 2025. URL https://arxiv.org/abs/2502.06472.

[993] Chaoyun Zhang, Shilin He, Jiaxu Qian, Bowen Li, Liqun Li, Si Qin, Yu Kang, Minghua Ma, Guyue Liu, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. Large language model-brained gui agents: A survey, 2025. URL https://arxiv.org/abs/2411.18279.

[994] Dong Huang, Jie M. Zhang, Michael Luck, Qingwen Bu, Yuhao Qing, and Heming Cui. Agentcoder: Multi-agent-based code generation with iterative testing and optimisation, 2024. URL https://arxiv.org/abs/2312.13010.

[995] Zixuan Wang, Chi-Keung Tang, and Yu-Wing Tai. Audio-agent: Leveraging LLMs for audio generation, editing and composition, 2025. URL https://arxiv.org/abs/2410.03335.

[996] Dong Zhang, Zhaowei Li, Pengyu Wang, Xin Zhang, Yaqian Zhou, and Xipeng Qiu. Speechagents: Human-communication simulation with multi-modal multi-agent systems, 2024. URL https://arxiv.org/abs/2401.03945.

[997] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie

Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

[998] Yuyu Luo, Nan Tang, Guoliang Li, Jiawei Tang, Chengliang Chai, and Xuedi Qin. Natural language to visualization by neural machine translation. *IEEE Trans. Vis. Comput. Graph.*, 28(1):217–226, 2022.

[999] Shuyu Shen, Sirong Lu, Leixian Shen, Zhonghua Sheng, Nan Tang, and Yuyu Luo. Ask humans or ai? exploring their roles in visualization troubleshooting. *CoRR*, abs/2412.07673, 2024.

[1000] Xudong Yang, Yifan Wu, Yizhang Zhu, Nan Tang, and Yuyu Luo. Askchart: Universal chart understanding through textual enhancement. *arXiv preprint arXiv:2412.19146*, 2024.

[1001] Zhilin Wang, Yu Ying Chiu, and Yu Cheung Chiu. Humanoid agents: Platform for simulating human-like generative agents. In Yansong Feng and Els Lefever, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 167–176, Singapore, December 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.emnlp-demo.15. URL https://aclanthology.org/2023.emnlp-demo.15/.

[1002] Gaowei Chang. Agentnetworkprotocol, 2025. URL https://github.com/chgaowei/AgentNetworkProtocol. GitHub repository, Accessed: 2025-01-07.

[1003] Yao Fu, Hao Peng, Tushar Khot, and Mirella Lapata. Improving language model negotiation with self-play and in-context learning from ai feedback. *arXiv preprint arXiv:2305.10142*, 2023.

[1004] Kai Xiong, Xiao Ding, Yixin Cao, Ting Liu, and Bing Qin. Examining inter-consistency of large language models collaboration: An in-depth analysis via debate. *arXiv preprint arXiv:2305.11595*, 2023.

[1005] Haotian Wang, Xiyuan Du, Weijiang Yu, Qianglong Chen, Kun Zhu, Zheng Chu, Lian Yan, and Yi Guan. Apollo's oracle: Retrieval-augmented reasoning in multi-agent debates. *arXiv preprint arXiv:2312.04854*, 2023.

[1006] Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. Language agents as optimizable graphs. *arXiv preprint arXiv:2402.16823*, 2024.

[1007] Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. Chateval: Towards better llm-based evaluators through multi-agent debate, 2023. URL https://arxiv.org/abs/2308.07201.

[1008] Guangyao Chen, Siwei Dong, Yu Shu, Ge Zhang, Jaward Sesay, Börje F. Karlsson, Jie Fu, and Yemin Shi. Autoagents: A framework for automatic agent generation, 2024. URL https://arxiv.org/abs/2309.17288.

[1009] Bingzheng Gan, Yufan Zhao, Tianyi Zhang, Jing Huang, Yusu Li, Shu Xian Teo, Changwang Zhang, and Wei Shi. Master: A multi-agent system with llm specialized mcts, 2025. URL https://arxiv.org/abs/2501.14304.

[1010] Bin Lei, Yi Zhang, Shan Zuo, Ali Payani, and Caiwen Ding. Macm: Utilizing a multi-agent system for condition mining in solving complex mathematical problems, 2024. URL https://arxiv.org/abs/2404.04735.

[1011] Zhuoyun Du, Chen Qian, Wei Liu, Zihao Xie, Yifei Wang, Yufan Dang, Weize Chen, and Cheng Yang. Multi-agent software development through cross-team collaboration. *arXiv preprint arXiv:2406.08979*, 2024.

[1012] Guozheng Li, Runfei Li, Yunshan Feng, Yu Zhang, Yuyu Luo, and Chi Harold Liu. Coinsight: Visual storytelling for hierarchical tables with connected insights. *IEEE Transactions on Visualization and Computer Graphics*, 2024.

[1013] Yilin Ye, Jianing Hao, Yihan Hou, Zhan Wang, Shishi Xiao, Yuyu Luo, and Wei Zeng. Generative ai for visualization: State of the art and future directions. *Visual Informatics*, 2024.

[1014] Yifan Wu, Lutao Yan, Leixian Shen, Yunhai Wang, Nan Tang, and Yuyu Luo. Chartinsights: Evaluating multimodal large language models for low-level chart question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 12174–12200, 2024.

[1015] Yunfan Zhang, Changlun Li, Yuyu Luo, and Nan Tang. Sketchfill: Sketch-guided code generation for imputing derived missing values. *arXiv preprint arXiv:2412.19113*, 2024.

[1016] Chengliang Chai, Nan Tang, Ju Fan, and Yuyu Luo. Demystifying artificial intelligence for data preparation. In *Companion of the 2023 International Conference on Management of Data*, pages 13–20, 2023.

[1017] Leixian Shen, Haotian Li, Yun Wang, Tianqi Luo, Yuyu Luo, and Huamin Qu. Data playwright: Authoring data videos with annotated narration. *IEEE Transactions on Visualization and Computer Graphics*, 2024.

[1018] Yupeng Xie, Yuyu Luo, Guoliang Li, and Nan Tang. Haichart: Human and AI paired visualization system. *Proc. VLDB Endow.*, 17(11):3178–3191, 2024.

[1019] Patara Trirat, Wonyong Jeong, and Sung Ju Hwang. Automl-agent: A multi-agent llm framework for full-pipeline automl. *arXiv preprint arXiv:2410.02958*, 2024.

[1020] Ziming Li, Qianbo Zang, David Ma, Jiawei Guo, Tuney Zheng, Minghao Liu, Xinyao Niu, Yue Wang, Jian Yang, Jiaheng Liu, Wanjun Zhong, Wangchunshu Zhou, Wenhao Huang, and Ge Zhang. Autokaggle: A multi-agent framework for autonomous data science competitions, 2024.

[1021] Suma Bailis, Jane Friedhoff, and Feiyang Chen. Werewolf arena: A case study in LLM evaluation via social deduction. *arXiv preprint arXiv:2407.13943*, 2024.

[1022] Yuwei Hu, Runlin Lei, Xinyi Huang, Zhewei Wei, and Yongchao Liu. Scalable and accurate graph reasoning with llm-based multi-agents, 2024. URL https://arxiv.org/abs/2410.05130.

[1023] Sumedh Rasal and E. J. Hauer. Navigating complexity: Orchestrated problem solving with multi-agent llms, 2024. URL https://arxiv.org/abs/2402.16713.

[1024] Cheng Li, Damien Teney, Linyi Yang, Qingsong Wen, Xing Xie, and Jindong Wang. Culturepark: Boosting cross-cultural understanding in large language models, 2024. URL https://arxiv.org/abs/2405.15145.

[1025] Zhao Kaiya, Michelangelo Naim, Jovana Kondic, Manuel Cortes, Jiaxin Ge, Shuying Luo, Guangyu Robert Yang, and Andrew Ahn. Lyfe agents: Generative agents for low-cost real-time social interactions, 2023. URL https://arxiv.org/abs/2310.02172.

[1026] Thorsten Händler. Balancing autonomy and alignment: A multi-dimensional taxonomy for autonomous llm-powered multi-agent architectures, 2023. URL https://arxiv.org/abs/2310.03659.

[1027] Weize Chen, Jiarui Yuan, Chen Qian, Cheng Yang, Zhiyuan Liu, and Maosong Sun. Optima: Optimizing effectiveness and efficiency for llm-based multi-agent system. *arXiv preprint arXiv:2410.08115*, 2024.

[1028] Chen Qian, Zihao Xie, Yifei Wang, Wei Liu, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, Zhiyuan Liu, and Maosong Sun. Scaling large-language-model-based multi-agent collaboration. *arXiv preprint arXiv:2406.07155*, 2024.

[1029] Hanqing Yang, Jingdi Chen, Marie Siew, Tania Lorido-Botran, and Carlee Joe-Wong. Llm-powered decentralized generative agents with adaptive hierarchical knowledge graph for cooperative planning. *arXiv preprint arXiv:2502.05453*, 2025.

[1030] Guangyao Chen, Siwei Dong, Yu Shu, Ge Zhang, Jaward Sesay, Börje F Karlsson, Jie Fu, and Yemin Shi. Autoagents: A framework for automatic agent generation. *arXiv preprint arXiv:2309.17288*, 2023.

[1031] Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*, 2023.

[1032] Yaoxiang Wang, Zhiyong Wu, Junfeng Yao, and Jinsong Su. Tdag: A multi-agent framework based on dynamic task decomposition and agent generation. *Neural Networks*, page 107200, 2025.

[1033] Boye Niu, Yiliao Song, Kai Lian, Yifan Shen, Yu Yao, Kun Zhang, and Tongliang Liu. Flow: A modular approach to automated agentic workflow generation. *arXiv preprint arXiv:2501.07834*, 2025.

[1034] Shilong Wang, Guibin Zhang, Miao Yu, Guancheng Wan, Fanci Meng, Chongye Guo, Kun Wang, and Yang Wang. G-safeguard: A topology-guided security lens and treatment on llm-based multi-agent systems. *arXiv preprint arXiv:2502.11127*, 2025.

[1035] Dawei Gao, Zitao Li, Xuchen Pan, Weirui Kuang, Zhijian Ma, Bingchen Qian, Fei Wei, Wenhao Zhang, Yuexiang Xie, Daoyuan Chen, et al. Agentscope: A flexible yet robust multi-agent platform. *arXiv preprint arXiv:2402.14034*, 2024.

[1036] Zhihao Fan, Jialong Tang, Wei Chen, Siyuan Wang, Zhongyu Wei, Jun Xi, Fei Huang, and Jingren Zhou. Ai hospital: Benchmarking large language models in a multi-agent medical interaction simulator. *arXiv preprint arXiv:2402.09742*, 2024.

[1037] Xiutian Zhao, Ke Wang, and Wei Peng. An electoral approach to diversify llm-based multi-agent collective decision-making. *arXiv preprint arXiv:2410.15168*, 2024.

[1038] Yoichi Ishibashi and Yoshimasa Nishimura. Self-organized agents: A LLM multi-agent framework toward ultra large-scale code generation and optimization. *arXiv preprint arXiv:2404.02183*, 2024.

[1039] Thorsten Händler. A taxonomy for autonomous llm-powered multi-agent architectures. In *KMIS*, pages 85–98, 2023.

[1040] Jinghua Piao, Yuwei Yan, Jun Zhang, Nian Li, Junbo Yan, Xiaochong Lan, Zhihong Lu, Zhiheng Zheng, Jing Yi Wang, Di Zhou, Chen Gao, Fengli Xu, Fang Zhang, Ke Rong, Jun Su, and Yong Li. Agentsociety: Large-scale simulation of llm-driven generative agents advances understanding of human behaviors and society, 2025. URL https://arxiv.org/abs/2502.08691.

[1041] Hung Du, Srikanth Thudumu, Rajesh Vasa, and Kon Mouzakis. A survey on context-aware multi-agent systems: techniques, challenges and future directions. *arXiv preprint arXiv:2402.01968*, 2024.

[1042] Ziyuan Zhou, Guanjun Liu, and Ying Tang. Multi-agent reinforcement learning: Methods, applications, visionary prospects, and challenges. *arXiv preprint arXiv:2305.10091*, 2023.

[1043] Changxi Zhu, Mehdi Dastani, and Shihan Wang. A survey of multi-agent deep reinforcement learning with communication. *Autonomous Agents and Multi-Agent Systems*, 38(1):4, 2024.

[1044] Jingqing Ruan, Xiaotian Hao, Dong Li, and Hangyu Mao. Learning to collaborate by grouping: A consensus-oriented strategy for multi-agent reinforcement learning. In *ECAI 2023*, pages 2010–2017. IOS Press, 2023.

[1045] Huaben Chen, Wenkang Ji, Lufeng Xu, and Shiyu Zhao. Multi-agent consensus seeking via large language models. *arXiv preprint arXiv:2310.20151*, 2023.

[1046] Yu Han Kim, Chanwoo Park, Hyewon Jeong, Yik Siu Chan, Xuhai Xu, Daniel McDuff, Cynthia Breazeal, and Hae Won Park. Mdagents: An adaptive collaboration of LLMs for medical decision-making. In *NeurIPS*, 2024.

[1047] Marios Papachristou, Longqi Yang, and Chin-Chia Hsu. Leveraging large language models for collective decision-making. *arXiv preprint arXiv:2311.04928*, 2023.

[1048] Giorgio Piatti, Zhijing Jin, Max Kleiman-Weiner, Bernhard Schölkopf, Mrinmaya Sachan, and Rada Mihalcea. Cooperate or collapse: Emergence of sustainable cooperation in a society of llm agents. *Advances in Neural Information Processing Systems*, 37:111715–111759, 2025.

[1049] Zichen Zhu, Hao Tang, Yansi Li, Kunyao Lan, Yixuan Jiang, Hao Zhou, Yixiao Wang, Situo Zhang, Liangtai Sun, Lu Chen, et al. Moba: A two-level agent system for efficient mobile task automation. *arXiv preprint arXiv:2410.13757*, 2024.

[1050] Zhenran Xu, Senbao Shi, Baotian Hu, Jindi Yu, Dongfang Li, Min Zhang, and Yuxiang Wu. Towards reasoning in large language models via multi-agent peer review collaboration. *arXiv preprint arXiv:2311.08152*, 2023.

[1051] Guhong Chen, Liyang Fan, Zihan Gong, Nan Xie, Zixuan Li, Ziqiang Liu, Chengming Li, Qiang Qu, Shiwen Ni, and Min Yang. Agentcourt: Simulating court with adversarial evolvable lawyer agents. *arXiv preprint arXiv:2408.08089*, 2024.

[1052] Zhangyue Yin, Qiushi Sun, Cheng Chang, Qipeng Guo, Junqi Dai, Xuanjing Huang, and Xipeng Qiu. Exchange-of-thought: Enhancing large language model capabilities through cross-model communication. *arXiv preprint arXiv:2312.01823*, 2023.

[1053] Ziming Li, Qianbo Zang, David Ma, Jiawei Guo, Tuney Zheng, Minghao Liu, Xinyao Niu, Yue Wang, Jian Yang, Jiaheng Liu, et al. Autokaggle: A multi-agent framework for autonomous data science competitions. *arXiv preprint arXiv:2410.20424*, 2024.

[1054] Chuyi Shang, Amos You, Sanjay Subramanian, Trevor Darrell, and Roei Herzig. Traveler: A modular multi-lmm agent framework for video question-answering. *arXiv preprint arXiv:2404.01476*, 2024.

[1055] Junzhi Chen, Juhao Liang, and Benyou Wang. Smurfs: Leveraging multiple proficiency agents with context-efficiency for tool planning, 2024.

[1056] Allen Z. Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, Zhenjia Xu, Dorsa Sadigh, Andy Zeng, and Anirudha Majumdar. Robots that ask for help: Uncertainty alignment for large language model planners. In *Proceedings of the Conference on Robot Learning (CoRL)*, 2023.

[1057] Yijia Shao, Vinay Samuel, Yucheng Jiang, John Yang, and Diyi Yang. Collaborative gym: A framework for enabling and evaluating human-agent collaboration, 2025. URL https://arxiv.org/abs/2412.15701.

[1058] Varun Nair, Elliot Schumacher, Geoffrey Tso, and Anitha Kannan. Dera: enhancing large language model completions with dialog-enabled resolving agents. *arXiv preprint arXiv:2303.17071*, 2023.

[1059] Chenglei Si, Weijia Shi, Chen Zhao, Luke Zettlemoyer, and Jordan Boyd-Graber. Getting more out of mixture of language model reasoning experts. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8234–8249, 2023.

[1060] Philip Schroeder, Nathaniel Morgan, Hongyin Luo, and James Glass. Thread: Thinking deeper with recursive spawning. *arXiv preprint arXiv:2405.17402*, 2024.

[1061] Tongxuan Liu, Xingyu Wang, Weizhe Huang, Wenjiang Xu, Yuting Zeng, Lei Jiang, Hailong Yang, and Jing Li. Groupdebate: Enhancing the efficiency of multi-agent debate using group discussion. *arXiv preprint arXiv:2409.14051*, 2024.

[1062] Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. A dynamic llm-powered agent network for task-oriented agent collaboration. In *First Conference on Language Modeling*, 2024.

[1063] Jintian Zhang, Xin Xu, Ningyu Zhang, Ruibo Liu, Bryan Hooi, and Shumin Deng. Exploring collaboration mechanisms for LLM agents: A social psychology view. *arXiv preprint arXiv:2310.02124*, 2023.

[1064] Shanshan Han, Qifan Zhang, Yuhang Yao, Weizhao Jin, Zhaozhuo Xu, and Chaoyang He. Llm multi-agent systems: Challenges and open problems. *arXiv preprint arXiv:2402.03578*, 2024.

[1065] Siyue Ren, Zhiyao Cui, Ruiqi Song, Zhen Wang, and Shuyue Hu. Emergence of social norms in generative agent societies: principles and architecture. *arXiv preprint arXiv:2403.08251*, 2024.

[1066] Aron Vallinder and Edward Hughes. Cultural evolution of cooperation among llm agents. *arXiv preprint arXiv:2412.10270*, 2024.

[1067] Nathalia Nascimento, Paulo Alencar, and Donald Cowan. Self-adaptive large language model (LLM)-based multiagent systems. In *2023 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*, pages 104–109. IEEE, 2023.

[1068] Lin Xu, Zhiyuan Hu, Daquan Zhou, Hongyu Ren, Zhen Dong, Kurt Keutzer, See Kiong Ng, and Jiashi Feng. MAgIC: Benchmarking large language model powered multi-agent in cognition, adaptability, rationality and collaboration. *arXiv preprint arXiv:2311.08562*, 2023.

[1069] Ceyao Zhang, Kaijie Yang, Siyi Hu, Zihao Wang, Guanghe Li, Yihang Sun, Cheng Zhang, Zhaowei Zhang, Anji Liu, Song-Chun Zhu, et al. Proagent: Building proactive cooperative AI with large language models. *CoRR*, 2023.

[1070] Kuan Wang, Yadong Lu, Michael Santacroce, Yeyun Gong, Chao Zhang, and Yelong Shen. Adapting LLM agents through communication. *arXiv preprint arXiv:2310.01444*, 2023.

[1071] Vighnesh Subramaniam, Yilun Du, Joshua B Tenenbaum, Antonio Torralba, Shuang Li, and Igor Mordatch. Multiagent finetuning: Self improvement with diverse reasoning chains. *arXiv preprint arXiv:2501.05707*, 2025.

[1072] Wanjia Zhao, Mert Yuksekgonul, Shirley Wu, and James Zou. Sirius: Self-improving multi-agent systems via bootstrapped reasoning. *arXiv preprint arXiv:2502.04780*, 2025.

[1073] Yifei Zhou, Song Jiang, Yuandong Tian, Jason Weston, Sergey Levine, Sainbayar Sukhbaatar, and Xian Li. Sweet-rl: Training multi-turn llm agents on collaborative reasoning tasks. *arXiv preprint arXiv:2503.15478*, 2025.

[1074] Haoyi Xiong, Zhiyuan Wang, Xuhong Li, Jiang Bian, Zeke Xie, Shahid Mumtaz, Anwer Al-Dulaimi, and Laura E Barnes. Converging paradigms: The synergy of symbolic and connectionist ai in LLM-empowered autonomous agents. *arXiv preprint arXiv:2407.08516*, 2024.

[1075] Houcheng Jiang, Junfeng Fang, Tianyu Zhang, An Zhang, Ruipeng Wang, Tao Liang, and Xiang Wang. Neuron-level sequential editing for large language models. *arXiv preprint arXiv:2410.04045*, 2024.

[1076] Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. minif2f: a cross-system benchmark for formal olympiad-level mathematics. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=9ZPegFuFTFv.

[1077] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *CoRR*, abs/2107.03374, 2021. URL https://arxiv.org/abs/2107.03374.

[1078] Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. Measuring coding challenge competence with APPS. In Joaquin Vanschoren and Sai-Kit Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/c24cd76e1ce41366a4bbe8a49b02a028-Abstract-round2.html.

[1079] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4149–4158. Association for Computational Linguistics, 2019. doi:10.18653/V1/N19-1421. URL https://doi.org/10.18653/v1/n19-1421.

[1080] Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? A question answering benchmark with implicit reasoning strategies. *Trans. Assoc. Comput. Linguistics*, 9:346–361, 2021. doi:10.1162/TACL_A_00370. URL https://doi.org/10.1162/tacl_a_00370.

[1081] Tanik Saikh, Tirthankar Ghosal, Amish Mittal, Asif Ekbal, and Pushpak Bhattacharyya. Scienceqa: a novel resource for question answering on scholarly articles. *Int. J. Digit. Libr.*, 23(3):289–301, 2022. doi:10.1007/S00799-022-00329-Y. URL https://doi.org/10.1007/s00799-022-00329-y.

[1082] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1601–1611. Association for Computational Linguistics, 2017. doi:10.18653/V1/P17-1147. URL https://doi.org/10.18653/v1/P17-1147.

[1083] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021. URL https://arxiv.org/abs/2110.14168.

[1084] Albert Qiaochu Jiang, Wenda Li, Jesse Michael Han, and Yuhuai Wu. Lisa: Language models of isabelle proofs. In *6th Conference on Artificial Intelligence and Theorem Proving*, pages 378–392, 2021.

[1085] Wei Xiong, Chengshuai Shi, Jiaming Shen, Aviv Rosenberg, Zhen Qin, Daniele Calandriello, Misha Khalman, Rishabh Joshi, Bilal Piot, Mohammad Saleh, Chi Jin, Tong Zhang, and Tianqi Liu. Building math agents with multi-turn iterative preference learning, 2025. URL https://arxiv.org/abs/2409.02392.

[1086] Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, and Maarten Sap. Sotopia: Interactive evaluation for social intelligence in language agents, 2024. URL https://arxiv.org/abs/2310.11667.

[1087] Yujia Li, David H. Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de Masson d'Autume,

Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. Competition-level code generation with alphacode. *CoRR*, abs/2203.07814, 2022. doi:10.48550/ARXIV.2203.07814. URL https://doi.org/10.48550/arXiv.2203.07814.

[1088] Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. Codegen: An open large language model for code with multi-turn program synthesis. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=iaYcJKpY2B_.

[1089] Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Wen-Tau Yih, Daniel Fried, Sida I. Wang, and Tao Yu. DS-1000: A natural and reliable benchmark for data science code generation. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 18319–18345. PMLR, 2023. URL https://proceedings.mlr.press/v202/lai23b.html.

[1090] Zhiruo Wang, Shuyan Zhou, Daniel Fried, and Graham Neubig. Execution-based evaluation for open-domain code generation. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 1271–1290. Association for Computational Linguistics, 2023. doi:10.18653/V1/2023.FINDINGS-EMNLP.89. URL https://doi.org/10.18653/v1/2023.findings-emnlp.89.

[1091] Jiangyi Deng, Xinfeng Li, Yanjiao Chen, Yijie Bai, Haiqin Weng, Yan Liu, Tao Wei, and Wenyuan Xu. Raconteur: A knowledgeable, insightful, and portable llm-powered shell command explainer. In *In the 32nd Annual Network and Distributed System Security Symposium (NDSS)*, 2025.

[1092] Sumithra Bhakthavatsalam, Daniel Khashabi, Tushar Khot, Bhavana Dalvi Mishra, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, and Peter Clark. Think you have solved direct-answer question answering? try arc-da, the direct-answer AI2 reasoning challenge. *CoRR*, abs/2102.03315, 2021. URL https://arxiv.org/abs/2102.03315.

[1093] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2924–2936. Association for Computational Linguistics, 2019. doi:10.18653/V1/N19-1300. URL https://doi.org/10.18653/v1/n19-1300.

[1094] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? A new dataset for open book question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2381–2391. Association for Computational Linguistics, 2018. doi:10.18653/V1/D18-1260. URL https://doi.org/10.18653/v1/d18-1260.

[1095] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9):99–106, 2021. doi:10.1145/3474381. URL https://doi.org/10.1145/3474381.

[1096] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4791–4800. Association for Computational Linguistics, 2019. doi:10.18653/V1/P19-1472. URL https://doi.org/10.18653/v1/p19-1472.

[1097] Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social iqa: Commonsense reasoning about social interactions. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 4462–4472. Association for Computational Linguistics, 2019. doi:10.18653/V1/D19-1454. URL https://doi.org/10.18653/v1/D19-1454.

[1098] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12,*

2020, pages 7432–7439. AAAI Press, 2020. doi:10.1609/AAAI.V34I05.6239. URL https://doi.org/10.1609/aaai.v34i05.6239.

[1099] Keisuke Sakaguchi, Chandra Bhagavatula, Ronan Le Bras, Niket Tandon, Peter Clark, and Yejin Choi. proscript: Partially ordered scripts generation via pre-trained language models. *CoRR*, abs/2104.08251, 2021. URL https://arxiv.org/abs/2104.08251.

[1100] Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=qFVVBzXxR2V.

[1101] Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. A diverse corpus for evaluating and developing english math word problem solvers. *CoRR*, abs/2106.15772, 2021. URL https://arxiv.org/abs/2106.15772.

[1102] Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2357–2367. Association for Computational Linguistics, 2019. doi:10.18653/V1/N19-1245. URL https://doi.org/10.18653/v1/n19-1245.

[1103] Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 158–167. Association for Computational Linguistics, 2017. doi:10.18653/V1/P17-1015. URL https://doi.org/10.18653/v1/P17-1015.

[1104] Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. MAWPS: A math word problem repository. In Kevin Knight, Ani Nenkova, and Owen Rambow, editors, *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1152–1157. The Association for Computational Linguistics, 2016. doi:10.18653/V1/N16-1136. URL https://doi.org/10.18653/v1/n16-1136.

[1105] Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2368–2378. Association for Computational Linguistics, 2019. doi:10.18653/V1/N19-1246. URL https://doi.org/10.18653/v1/n19-1246.

[1106] Sean Welleck, Jiacheng Liu, Ronan Le Bras, Hanna Hajishirzi, Yejin Choi, and Kyunghyun Cho. Naturalproofs: Mathematical theorem proving in natural language. In Joaquin Vanschoren and Sai-Kit Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/d9d4f495e875a2e075a1a4a6e1b9770f-Abstract-round1.html.

[1107] Zhangir Azerbayev, Bartosz Piotrowski, Hailey Schoelkopf, Edward W. Ayers, Dragomir Radev, and Jeremy Avigad. Proofnet: Autoformalizing and formally proving undergraduate-level mathematics. *CoRR*, abs/2302.12433, 2023. doi:10.48550/ARXIV.2302.12433. URL https://doi.org/10.48550/arXiv.2302.12433.

[1108] Wei Liu, Chenxi Wang, Yifei Wang, Zihao Xie, Rennai Qiu, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, and Chen Qian. Autonomous agents for collaborative task under information asymmetry, 2024. URL https://arxiv.org/abs/2406.14928.

[1109] Timothy Ossowski, Jixuan Chen, Danyal Maqbool, Zefan Cai, Tyler Bradshaw, and Junjie Hu. Comma: A communicative multimodal multi-agent benchmark, 2025. URL https://arxiv.org/abs/2410.07553.

[1110] Yang Liu, Peng Sun, and Hang Li. Large language models as agents in two-player games, 2024. URL https://arxiv.org/abs/2402.08078.

[1111] Taehoon Kim. Ethereum AI agent coordinator (EAAC): A framework for AI agent activity coordination. In *Agentic Markets Workshop at ICML 2024*, 2024. URL https://openreview.net/forum?id=n2dVVwZwPP.

[1112] Yohei Nakajima. Babyagi arena, 2023. URL https://github.com/yoheinakajima/babyagi-arena.

[1113] Shankar Kumar Jeyakumar, Alaa Alameer Ahmad, and Adrian Garret Gabriel. Advancing agentic systems: Dynamic task decomposition, tool integration and evaluation using novel metrics and dataset. In *NeurIPS 2024 Workshop on Open-World Agents*, 2024. URL https://openreview.net/forum?id=kRRLhPp7CO.

[1114] Haofei Yu, Zhaochen Hong, Zirui Cheng, Kunlun Zhu, Keyang Xuan, Jinwei Yao, Tao Feng, and Jiaxuan You. Researchtown: Simulator of human research community, 2024. URL https://arxiv.org/abs/2412.17767.

[1115] Xian Gao, Zongyun Zhang, Mingye Xie, Ting Liu, and Yuzhuo Fu. Graph of ai ideas: Leveraging knowledge graphs and llms for ai research idea generation, 2025. URL https://arxiv.org/abs/2503.08549.

[1116] Junzhe Chen, Xuming Hu, Shuodi Liu, Shiyu Huang, Wei-Wei Tu, Zhaofeng He, and Lijie Wen. Llmarena: Assessing capabilities of large language models in dynamic multi-agent environments, 2024. URL https://arxiv.org/abs/2402.16499.

[1117] Richard Zhuang, Akshat Gupta, Richard Yang, Aniket Rahane, Zhengyu Li, and Gopala Anumanchipalli. Pokerbench: Training large language models to become professional poker players, 2025. URL https://arxiv.org/abs/2501.08328.

[1118] Nicoló Fontana, Francesco Pierri, and Luca Maria Aiello. Nicer than humans: How do large language models behave in the prisoner's dilemma?, 2024. URL https://arxiv.org/abs/2406.13605.

[1119] Sihao Hu, Tiansheng Huang, and Ling Liu. Pokellmon: A human-parity agent for pokemon battles with large language models, 2024. URL https://arxiv.org/abs/2402.01118.

[1120] Qiuejie Xie, Qiming Feng, Tianqi Zhang, Qingqiu Li, Linyi Yang, Yuejie Zhang, Rui Feng, Liang He, Shang Gao, and Yue Zhang. Human simulacra: Benchmarking the personification of large language models, 2025. URL https://arxiv.org/abs/2402.18180.

[1121] Rong Ye, Yongxin Zhang, Yikai Zhang, Haoyu Kuang, Zhongyu Wei, and Peng Sun. Multi-agent kto: Reinforcing strategic interactions of large language model in language game, 2025. URL https://arxiv.org/abs/2501.14225.

[1122] Chengxing Xie and Difan Zou. A human-like reasoning framework for multi-phases planning task with large language models, 2024. URL https://arxiv.org/abs/2405.18208.

[1123] Yauwai Yim, Chunkit Chan, Tianyu Shi, Zheye Deng, Wei Fan, Tianshi Zheng, and Yangqiu Song. Evaluating and enhancing llms agent based on theory of mind in guandan: A multi-player cooperative game under imperfect information, 2024. URL https://arxiv.org/abs/2408.02559.

[1124] Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors, 2023. URL https://arxiv.org/abs/2308.10848.

[1125] Han Wang, Binbin Chen, Tieying Zhang, and Baoxiang Wang. Learning to communicate through implicit communication channels, 2025. URL https://arxiv.org/abs/2411.01553.

[1126] Wenyue Hua, Lizhou Fan, Lingyao Li, Kai Mei, Jianchao Ji, Yingqiang Ge, Libby Hemphill, and Yongfeng Zhang. War and peace (waragent): Large language model-based multi-agent simulation of world wars, 2024. URL https://arxiv.org/abs/2311.17227.

[1127] Yizhe Huang, Xingbo Wang, Hao Liu, Fanqi Kong, Aoyang Qin, Min Tang, Song-Chun Zhu, Mingjie Bi, Siyuan Qi, and Xue Feng. Adasociety: An adaptive environment with social structures for multi-agent decision-making, 2025. URL https://arxiv.org/abs/2411.03865.

[1128] Jen tse Huang, Jiaxu Zhou, Tailin Jin, Xuhui Zhou, Zixi Chen, Wenxuan Wang, Youliang Yuan, Michael R. Lyu, and Maarten Sap. On the resilience of llm-based multi-agent collaboration with faulty agents, 2025. URL https://arxiv.org/abs/2408.00989.

[1129] Zehang Deng, Yongjian Guo, Changzhou Han, Wanlun Ma, Junwu Xiong, Sheng Wen, and Yang Xiang. Ai agents under threat: A survey of key security challenges and future pathways. *ACM Computing Surveys*, 2024.

[1130] Yi Liu, Gelei Deng, Yuekang Li, Kailong Wang, Zihao Wang, Xiaofeng Wang, Tianwei Zhang, Yepang Liu, Haoyu Wang, Yan Zheng, et al. Prompt injection attack against llm-integrated applications. *arXiv preprint arXiv:2306.05499*, 2023.

[1131] Wenkai Yang, Xiaohan Bi, Yankai Lin, Sishuo Chen, Jie Zhou, and Xu Sun. Watch out for your agents! investigating backdoor threats to LLM-based agents. *arXiv preprint arXiv:2402.11208*, 2024.

[1132] Haibo Jin, Leyang Hu, Xinuo Li, Peiyan Zhang, Chonghan Chen, Jun Zhuang, and Haohan Wang. Jailbreakzoo: Survey, landscapes, and horizons in jailbreaking large language and vision-language models. *arXiv preprint arXiv:2407.01599*, 2024.

[1133] Sibo Yi, Yule Liu, Zhen Sun, Tianshuo Cong, Xinlei He, Jiaxing Song, Ke Xu, and Qi Li. Jailbreak attacks and defenses against large language models: A survey. *arXiv preprint arXiv:2407.04295*, 2024.

[1134] Andy Zou, Zifan Wang, Norman Mu, and Jacob Andreas. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.

[1135] Yihao Zhang and Zeming Wei. Boosting jailbreak attack with momentum. *arXiv preprint arXiv:2405.01229*, 2024.

[1136] Xiaojun Jia, Tianyu Pang, Chao Du, Yihao Huang, Jindong Gu, Yang Liu, Xiaochun Cao, and Min Lin. Improved techniques for optimization-based jailbreaking on large language models. *arXiv preprint arXiv:2405.21018*, 2024.

[1137] Yifan Luo, Zhennan Zhou, Meitan Wang, and Bin Dong. Jailbreak instruction-tuned LLMs via end-of-sentence mlp re-weighting. *arXiv preprint arXiv:2410.10150*, 2024.

[1138] Tianlong Li, Xiaoqing Zheng, and Xuanjing Huang. Open the pandora's box of llms: Jailbreaking LLMs through representation engineering. *arXiv preprint arXiv:2401.06824*, 2024.

[1139] Leyang Hu and Boran Wang. DROJ: A prompt-driven attack against large language models. *arXiv preprint arXiv:2411.09125*, 2024.

[1140] Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. Autodan: Generating stealthy jailbreak prompts on aligned large language models. *arXiv preprint arXiv:2310.04451*, 2023.

[1141] Xuancun Lu, Zhengxian Huang, Xinfeng Li, Xiaoyu Ji, and Wenyuan Xu. Poex: Policy executable embodied AI jailbreak attacks. *arXiv preprint arXiv:2412.16633*, 2024.

[1142] Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does LLM safety training fail? *Advances in Neural Information Processing Systems*, 36, 2023.

[1143] Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries. In *arXiv preprint arXiv:2310.08419*, 2023.

[1144] Haibo Jin, Andy Zhou, Joe Menke, and Haohan Wang. Jailbreaking large language models against moderation guardrails via cipher characters. *Advances in Neural Information Processing Systems*, 37:59408–59435, 2025.

[1145] Xiangyu Qi, Kaixuan Huang, Ashwinee Panda, Peter Henderson, Mengdi Wang, and Prateek Mittal. Visual adversarial examples jailbreak aligned large language models. In *AAAI Conference on Artificial Intelligence*, volume 38, pages 21527–21536, 2024.

[1146] Haibo Jin, Ruoxi Chen, Andy Zhou, Yang Zhang, and Haohan Wang. Guard: Role-playing to generate natural-language jailbreakings to test guideline adherence of large language models. *arXiv preprint arXiv:2402.03299*, 2024.

[1147] Teng Ma, Xiaojun Jia, Ranjie Duan, Xinfeng Li, Yihao Huang, Zhixuan Chu, Yang Liu, and Wenqi Ren. Heuristic-induced multimodal risk distribution jailbreak attack for multimodal large language models. *arXiv preprint arXiv:2412.05934*, 2024.

[1148] Sensen Gao, Xiaojun Jia, Yihao Huang, Ranjie Duan, Jindong Gu, Yang Liu, and Qing Guo. Rt-attack: Jailbreaking text-to-image models via random token. *arXiv preprint arXiv:2408.13896*, 2024.

[1149] Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*, pages 79–90, 2023.

[1150] Xiaogeng Liu, Zhiyuan Yu, Yizhe Zhang, Ning Zhang, and Chaowei Xiao. Automatic and universal prompt injection attacks against large language models. *arXiv preprint arXiv:2403.04957*, 2024.

[1151] Jiawen Shi, Zenghui Yuan, Yinuo Liu, Yue Huang, Pan Zhou, Lichao Sun, and Neil Zhenqiang Gong. Optimization-based prompt injection attack to LLM-as-a-judge. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 660–674, 2024.

[1152] Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. Benchmarking indirect prompt injections in tool-integrated large language model agents. *arXiv preprint arXiv:2403.02691*, 2024.

[1153] Johann Rehberger. Trust no ai: Prompt injection along the cia security triad. *arXiv preprint arXiv:2412.06090*, 2024.

[1154] Subaru Kimura, Ryota Tanaka, Shumpei Miyawaki, Jun Suzuki, and Keisuke Sakaguchi. Empirical analysis of large vision-language models against goal hijacking via visual prompt injection. *arXiv preprint arXiv:2408.03554*, 2024.

[1155] Edoardo Debenedetti, Javier Rando, Daniel Paleka, Silaghi Fineas Florin, Dragos Albastroiu, Niv Cohen, Yuval Lemberg, Reshmi Ghosh, Rui Wen, Ahmed Salem, et al. Dataset and lessons learned from the 2024 satml LLM capture-the-flag competition. *arXiv preprint arXiv:2406.07954*, 2024.

[1156] Sander Schulhoff, Jeremy Pinto, Anaum Khan, Louis-François Bouchard, Chenglei Si, Svetlina Anati, Valen Tagliabue, Anson Kost, Christopher Carnahan, and Jordan Boyd-Graber. Ignore this title and hackaprompt: Exposing systemic vulnerabilities of LLMs through a global prompt hacking competition. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4945–4977, 2023.

[1157] Yucheng Zhang, Qinfeng Li, Tianyu Du, Xuhong Zhang, et al. Hijackrag: Hijacking attacks against retrieval-augmented large language models. *arXiv preprint arXiv:2410.22832*, 2025.

[1158] Cody Clop and Yannick Teglia. Backdoored retrievers for prompt injection attacks on retrieval augmented generation of large language models. *arXiv preprint arXiv:2410.14479*, 2024.

[1159] Donghyun Lee and Mo Tiwari. Prompt Infection: LLM-to-LLM Prompt Injection within Multi-Agent Systems. *arXiv preprint arXiv:2410.07283*, 2024.

[1160] Fredrik Nestaas, Edoardo Debenedetti, and Florian Tramèr. Adversarial search engine optimization for large language models. *arXiv preprint arXiv:2406.18382*, 2024.

[1161] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12): 1–38, 2023.

[1162] Nick McKenna, Tianyi Li, Liang Cheng, Mohammad Javad Hosseini, Mark Johnson, and Mark Steedman. Sources of hallucination by large language models on inference tasks. *arXiv preprint arXiv:2305.14552*, 2023.

[1163] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232*, 2023.

[1164] Mobashir Sadat, Zhengyu Zhou, Lukas Lange, Jun Araki, Arsalan Gundroo, Bingqing Wang, Rakesh R Menon, Md Rizwan Parvez, and Zhe Feng. DELUCIONQA: Detecting Hallucinations in Domain-specific Question Answering. *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3737–3748, 2023.

[1165] Haoqiang Kang and Xiao-Yang Liu. Deficiency of large language models in finance: An empirical examination of hallucination. In *I Can't Believe It's Not Better Workshop: Failure Modes in the Age of Foundation Models*, 2023.

[1166] Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv:2401.11817*, 2024.

[1167] Jio Oh, Soyeon Kim, Junseok Seo, Jindong Wang, Ruochen Xu, Xing Xie, and Steven Euijong Whang. Erbench: An entity-relationship based automatically verifiable hallucination benchmark for large language models. *arXiv preprint arXiv:2403.05266*, 2024.

[1168] Tian Yu, Shaolei Zhang, and Yang Feng. Truth-aware context selection: Mitigating the hallucinations of large language models being misled by untruthful contexts. *arXiv preprint arXiv:2403.07556*, 2024.

[1169] Yiyi Chen, Qiongxiu Li, Russa Biswas, and Johannes Bjerva. Large language models are easily confused: A quantitative metric, security implications and typological analysis. *arXiv preprint arXiv:2410.13237*, 2024.

[1170] Zhiying Zhu, Zhiqing Sun, and Yiming Yang. Halueval-wild: Evaluating hallucinations of language models in the wild. *arXiv preprint arXiv:2403.04307*, 2024.

[1171] Yiyang Zhou, Chenhang Cui, Jaehong Yoon, Linjun Zhang, Zhun Deng, Chelsea Finn, Mohit Bansal, and Huaxiu Yao. Analyzing and mitigating object hallucination in large vision-language models. In *International Conference on Learning Representations*, 2023.

[1172] Linxi Zhao, Yihe Deng, Weitong Zhang, and Quanquan Gu. Mitigating object hallucination in large vision-language models via classifier-free guidance. *arXiv preprint arXiv:2402.08680*, 2024.

[1173] Leonardo Ranaldi and Giulia Pucci. When large language models contradict humans? large language models' sycophantic behaviour. *arXiv preprint arXiv:2311.09410*, 2023.

[1174] Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, et al. Hallusionbench: an advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14375–14385, 2024.

[1175] Kedi Chen, Qin Chen, Jie Zhou, Yishen He, and Liang He. Diahalu: A dialogue-level hallucination evaluation benchmark for large language models. *arXiv preprint arXiv:2403.00896*, 2024.

[1176] Jiaming Ji, Tianyi Qiu, Boyuan Chen, Borong Zhang, Hantao Lou, Kaile Wang, Yawen Duan, Zhonghao He, Jiayi Zhou, Zhaowei Zhang, et al. Ai alignment: A comprehensive survey. *arXiv preprint arXiv:2310.19852*, 2023.

[1177] Victoria Krakovna, Jonathan Uesato, Vladimir Mikulik, Matthew Rahtz, Tom Everitt, Ramana Kumar, Zac Kenton, Jan Leike, and Shane Legg. Specification gaming: The flip side of AI ingenuity, 2020. URL https://deepmind.google/discover/blog/specification-gaming-the-flip-side-of-ai-ingenuity/.

[1178] Richard Ngo, Lawrence Chan, and Sören Mindermann. The alignment problem from a deep learning perspective. *arXiv preprint arXiv:2209.00626*, 2022.

[1179] Shen Li, Liuyi Yao, Lan Zhang, and Yaliang Li. Safety layers in aligned large language models: The key to LLM security. *arXiv preprint arXiv:2408.17003*, 2024.

[1180] Zhanhui Zhou, Jie Liu, Zhichen Dong, Jiaheng Liu, Chao Yang, Wanli Ouyang, and Yu Qiao. Emulated disalignment: Safety alignment for large language models may backfire! *arXiv preprint arXiv:2402.12343*, 2024.

[1181] Hasan Abed Al Kader Hammoud, Umberto Michieli, Fabio Pizzati, Philip Torr, Adel Bibi, Bernard Ghanem, and Mete Ozay. Model merging and safety alignment: One bad model spoils the bunch. *arXiv preprint arXiv:2406.14563*, 2024.

[1182] Yang Liu, Yuanshun Yao, Jean-Francois Ton, Xiaoying Zhang, Ruocheng Guo Hao Cheng, Yegor Klochkov, Muhammad Faaiz Taufiq, and Hang Li. Trustworthy llms: A survey and guideline for evaluating large language models' alignment. *arXiv preprint arXiv:2308.05374*, 2023.

[1183] Boyi Wei, Kaixuan Huang, Yangsibo Huang, Tinghao Xie, Xiangyu Qi, Mengzhou Xia, Prateek Mittal, Mengdi Wang, and Peter Henderson. Assessing the brittleness of safety alignment via pruning and low-rank modifications. *arXiv preprint arXiv:2402.05162*, 2024.

[1184] Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! *arXiv preprint arXiv:2310.03693*, 2023.

[1185] Yotam Wolf, Noam Wies, Oshri Avnery, Yoav Levine, and Amnon Shashua. Fundamental limitations of alignment in large language models. *arXiv preprint arXiv:2304.11082*, 2023.

[1186] Keita Kurita, Paul Michel, and Graham Neubig. Weight poisoning attacks on pre-trained models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5560–5574, 2020.

[1187] Yanzhou Li, Tianlin Li, Kangjie Chen, Jian Zhang, Shangqing Liu, Wenhan Wang, Tianwei Zhang, and Yang Liu. Badedit: Backdooring large language models by model editing. *arXiv preprint arXiv:2403.13355*, 2024.

[1188] Tian Dong, Minhui Xue, Guoxing Chen, Rayne Holland, Yan Meng, Shaofeng Li, Zhen Liu, and Haojin Zhu. The philosopher's stone: Trojaning plugins of large language models. *arXiv preprint arXiv:2312.00374*, 2023.

[1189] Jaehan Kim, Minkyoo Song, Seung Ho Na, and Seungwon Shin. Obliviate: Neutralizing task-agnostic backdoors within the parameter-efficient fine-tuning paradigm. *arXiv preprint arXiv:2409.14119*, 2024.

[1190] Sanghak Oh, Kiho Lee, Seonhye Park, Doowon Kim, and Hyoungshick Kim. Poisoned chatgpt finds work for idle hands: Exploring developers' coding practices with insecure suggestions from poisoned ai models. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 1141–1159. IEEE, 2024.

[1191] Sumeet Ramesh Motwani, Mikhail Baranchuk, Martin Strohmeier, Vijay Bolina, Philip HS Torr, Lewis Hammond, and Christian Schroeder de Witt. Secret collusion among generative AI agents. *arXiv preprint arXiv:2402.07510*, 2024.

[1192] Abdullah Arafat Miah and Yu Bi. Exploiting the vulnerability of large language models via defense-aware architectural backdoor. *arXiv preprint arXiv:2409.01952*, 2024.

[1193] Alexander Wan, Eric Wallace, Sheng Shen, and Dan Klein. Poisoning language models during instruction tuning. In *International Conference on Machine Learning*, pages 35413–35425. PMLR, 2023.

[1194] Zhaorun Chen, Zhen Xiang, Chaowei Xiao, Dawn Song, and Bo Li. Agentpoison: Red-teaming llm agents via poisoning memory or knowledge bases. *Advances in Neural Information Processing Systems*, 37:130185–130213, 2025.

[1195] Fatemeh Nazary, Yashar Deldjoo, and Tommaso di Noia. Poison-rag: Adversarial data poisoning attacks on retrieval-augmented generation in recommender systems. *arXiv preprint arXiv:2501.11759*, 2025.

[1196] Tingchen Fu, Mrinank Sharma, Philip Torr, Shay B Cohen, David Krueger, and Fazl Barez. Poisonbench: Assessing large language model vulnerability to data poisoning. *arXiv preprint arXiv:2410.08811*, 2024.

[1197] Bocheng Chen, Hanqing Guo, Guangjing Wang, Yuanda Wang, and Qiben Yan. The dark side of human feedback: Poisoning large language models via user inputs. *arXiv preprint arXiv:2409.00787*, 2024.

[1198] Dillon Bowen, Brendan Murphy, Will Cai, David Khachaturov, Adam Gleave, and Kellin Pelrine. Scaling laws for data poisoning in LLMs. *arXiv e-prints*, pages arXiv–2408, 2024.

[1199] Jiaming He, Wenbo Jiang, Guanyu Hou, Wenshu Fan, Rui Zhang, and Hongwei Li. Talk too much: Poisoning large language models under token limit. *arXiv preprint arXiv:2404.14795*, 2024.

[1200] Tim Baumgärtner, Yang Gao, Dana Alon, and Donald Metzler. Best-of-venom: Attacking rlhf by injecting poisoned preference data. *arXiv preprint arXiv:2404.05530*, 2024.

[1201] Evan Hubinger, Carson Denison, Jesse Mu, Mike Lambert, Meg Tong, Monte MacDiarmid, Tamera Lanham, Daniel M Ziegler, Tim Maxwell, Newton Cheng, et al. Sleeper agents: Training deceptive LLMs that persist through safety training. *arXiv preprint arXiv:2401.05566*, 2024.

[1202] Fangzhou Wu, Shutong Wu, Yulong Cao, and Chaowei Xiao. Wipi: A new web threat for LLM-driven web agents. *arXiv preprint arXiv:2403.09875*, 2024.

[1203] Ruochen Jiao, Shaoyuan Xie, Justin Yue, Takami Sato, Lixu Wang, Yixuan Wang, Qi Alfred Chen, and Qi Zhu. Exploring backdoor attacks against large language model-based decision making. *arXiv preprint arXiv:2405.20774*, 2024.

[1204] Huaizhi Ge, Yiming Li, Qifan Wang, Yongfeng Zhang, and Ruixiang Tang. When backdoors speak: Understanding LLM backdoor attacks through model-generated explanations. *arXiv preprint arXiv:2411.12701*, 2024.

[1205] Jun Yan, Vikas Yadav, Shiyang Li, Lichang Chen, Zheng Tang, Hai Wang, Vijay Srinivasan, Xiang Ren, and Hongxia Jin. Backdooring instruction-tuned large language models with virtual prompt injection. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6065–6086, 2024.

[1206] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[1207] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX security symposium (USENIX security 19)*, pages 267–284, 2019.

[1208] Christopher A Choquette-Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. Label-only membership inference attacks. In *International conference on machine learning*, pages 1964–1974. PMLR, 2021.

[1209] Wenjie Fu, Huandong Wang, Chen Gao, Guanghua Liu, Yong Li, and Tao Jiang. Practical membership inference attacks against fine-tuned large language models via self-prompt calibration. *arXiv preprint arXiv:2311.06062*, 2023.

[1210] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1897–1914. IEEE, 2022.

[1211] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S Yu, and Xuyun Zhang. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys (CSUR)*, 54(11s):1–37, 2022.

[1212] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[1213] Yang Bai, Ge Pei, Jindong Gu, Yong Yang, and Xingjun Ma. Special characters attack: Toward scalable training data extraction from large language models. *arXiv preprint arXiv:2405.05990*, 2024.

[1214] Zhexin Zhang, Jiaxin Wen, and Minlie Huang. Ethicist: Targeted training data extraction through loss smoothed soft prompting and calibrated confidence estimation. *arXiv preprint arXiv:2307.04401*, 2023.

[1215] John X Morris, Wenting Zhao, Justin T Chiu, Vitaly Shmatikov, and Alexander M Rush. Language model inversion. *arXiv preprint arXiv:2311.13647*, 2023.

[1216] Xudong Pan, Mi Zhang, Shouling Ji, and Min Yang. Privacy risks of general-purpose language models. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1314–1331. IEEE, 2020.

[1217] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models. *arXiv preprint arXiv:2202.07646*, 2022.

[1218] Nicholas Carlini, Daniel Paleka, Krishnamurthy Dj Dvijotham, Thomas Steinke, Jonathan Hayase, A Feder Cooper, Katherine Lee, Matthew Jagielski, Milad Nasr, Arthur Conmy, et al. Stealing part of a production language model. *arXiv preprint arXiv:2403.06634*, 2024.

[1219] Yash More, Prakhar Ganesh, and Golnoosh Farnadi. Towards more realistic extraction attacks: An adversarial perspective. *arXiv preprint arXiv:2407.02596*, 2024.

[1220] Fábio Perez and Ian Ribeiro. Ignore previous prompt: Attack techniques for language models. In *Workshop on Trustworthy and Socially Responsible Machine Learning (TSRML@ NeurIPS 2022)*, 2022.

[1221] Xinyue Shen, Yiting Qu, Michael Backes, and Yang Zhang. Prompt stealing attacks against {Text-to-Image} generation models. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 5823–5840, 2024.

[1222] Zhifeng Jiang, Zhihua Jin, and Guoliang He. Safeguarding system prompts for llms. *arXiv preprint arXiv:2412.13426*, 2024.

[1223] Xinyao Zheng, Husheng Han, Shangyi Shi, Qiyan Fang, Zidong Du, Qi Guo, and Xing Hu. Inputsnatch: Stealing input in LLM services via timing side-channel attacks. *arXiv preprint arXiv:2411.18191*, 2024.

[1224] Yiming Zhang, Nicholas Carlini, and Daphne Ippolito. Effective prompt extraction from language models. *arXiv preprint arXiv:2307.06865*, 2023.

[1225] Rui Wen, Tianhao Wang, Michael Backes, Yang Zhang, and Ahmed Salem. Last one standing: A comparative analysis of security and privacy of soft prompt tuning, lora, and in-context learning. *arXiv preprint arXiv:2310.11397*, 2023.

[1226] Yanjie Zhao, Xinyi Hou, Shenao Wang, and Haoyu Wang. Llm app store analysis: A vision and roadmap. *ACM Transactions on Software Engineering and Methodology*, 2024.

[1227] Yong Yang, Xuhong Zhang, Yi Jiang, Xi Chen, Haoyu Wang, Shouling Ji, and Zonghui Wang. Prsa: Prompt reverse stealing attacks against large language models. *arXiv preprint arXiv:2402.07870*, 2024.

[1228] Divyansh Agarwal, Alexander R Fabbri, Ben Risher, Philippe Laban, Shafiq Joty, and Chien-Sheng Wu. Prompt leakage effect and defense strategies for multi-turn llm interactions. *arXiv preprint arXiv:2404.16251*, 2024.

[1229] Divyansh Agarwal, Alexander R Fabbri, Philippe Laban, Shafiq Joty, Caiming Xiong, and Chien-Sheng Wu. Investigating the prompt leakage effect and black-box defenses for multi-turn LLM interactions. *arXiv preprint arXiv:2402.06770*, 2024.

[1230] Zi Liang, Haibo Hu, Qingqing Ye, Yaxin Xiao, and Haoyang Li. Why are my prompts leaked? unraveling prompt extraction threats in customized large language models. *arXiv preprint arXiv:2408.02416*, 2024.

[1231] Bo Hui, Haolin Yuan, Neil Gong, Philippe Burlina, and Yinzhi Cao. Pleak: Prompt leaking attacks against large language model applications. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 3600–3614, 2024.

[1232] Itay Yona, Ilia Shumailov, Jamie Hayes, and Nicholas Carlini. Stealing user prompts from mixture of experts. *arXiv preprint arXiv:2410.22884*, 2024.

[1233] Gelei Deng, Yi Liu, Yuekang Li, Kailong Wang, Ying Zhang, Zefeng Li, Haoyu Wang, Tianwei Zhang, and Yang Liu. Jailbreaker: Automated jailbreak across multiple large language model chatbots. *arXiv preprint arXiv:2307.08715*, 2023.

[1234] Jiahao Yu, Xingwei Lin, Zheng Yu, and Xinyu Xing. Gptfuzzer: Red teaming large language models with auto-generated jailbreak prompts. *arXiv preprint arXiv:2309.10253*, 2023.

[1235] Aounon Kumar, Chirag Agarwal, Suraj Srinivas, Aaron Jiaxun Li, Soheil Feizi, and Himabindu Lakkaraju. Certifying llm safety against adversarial prompting. *arXiv preprint arXiv:2309.02705*, 2023.

[1236] Alexander Robey, Eric Wong, Hamed Hassani, and George J Pappas. SmoothLLM: Defending large language models against jailbreaking attacks. *arXiv preprint arXiv:2310.03684*, 2023.

[1237] Yifan Zeng, Yiran Wu, Xiao Zhang, Huazheng Wang, and Qingyun Wu. Autodefense: Multi-agent LLM defense against jailbreak attacks. In *Neurips Safe Generative AI Workshop 2024*, 2024. URL https://openreview.net/forum?id=WMwoSLAENS.

[1238] Zelong Li, Wenyue Hua, Hao Wang, He Zhu, and Yongfeng Zhang. Formal-llm: Integrating formal language and natural language for controllable llm-based agents. *arXiv preprint arXiv:2402.00798*, 2024.

[1239] Benji Peng, Ziqian Bi, Qian Niu, Ming Liu, Pohsun Feng, Tianyang Wang, Lawrence KQ Yan, Yizhu Wen, Yichao Zhang, and Caitlyn Heqi Yin. Jailbreaking and mitigation of vulnerabilities in large language models. *arXiv preprint arXiv:2410.15236*, 2024.

[1240] Yuchen Yang, Hongwei Yao, Bingrun Yang, Yiling He, Yiming Li, Tianwei Zhang, and Zhan Qin. Tpia: Towards target-specific prompt injection attack against code-oriented large language models. *arXiv preprint arXiv:2407.09164*, 2024.

[1241] Md Ahsan Ayub and Subhabrata Majumdar. Embedding-based classifiers can detect prompt injection attacks. *arXiv preprint arXiv:2410.22284*, 2024.

[1242] Sizhe Chen, Julien Piet, Chawin Sitawarin, and David Wagner. Struq: Defending against prompt injection with structured queries. *arXiv preprint arXiv:2402.06363*, 2024.

[1243] Feiran Jia, Tong Wu, Xin Qin, and Anna Squicciarini. The task shield: Enforcing task alignment to defend against indirect prompt injection in LLM agents. *arXiv preprint arXiv:2412.16682*, 2024.

[1244] Kuo-Han Hung, Ching-Yun Ko, Ambrish Rawat, I Chung, Winston H Hsu, Pin-Yu Chen, et al. Attention tracker: Detecting prompt injection attacks in llms. *arXiv preprint arXiv:2411.00348*, 2024.

[1245] Yulin Chen, Haoran Li, Zihao Zheng, Yangqiu Song, Dekai Wu, and Bryan Hooi. Defense against prompt injection attack by leveraging attack techniques. *arXiv preprint arXiv:2411.00459*, 2024.

[1246] Rongwu Xu, Brian Lin, Shujian Yang, Tianqi Zhang, Weiyan Shi, Tianwei Zhang, Zhixuan Fang, Wei Xu, and Han Qiu. The earth is flat because...: Investigating llms' belief towards misinformation via persuasive conversation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16259–16303, 2024.

[1247] Scott Barnett, Stefanus Kurniawan, Srikanth Thudumu, Zach Brannelly, and Mohamed Abdelrazek. Seven failure points when engineering a retrieval augmented generation system. In *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI*, pages 194–199, 2024.

[1248] Jiarui Li, Ye Yuan, and Zehua Zhang. Enhancing LLM factual accuracy with rag to counter hallucinations: A case study on domain-specific queries in private knowledge-bases. *arXiv preprint arXiv:2403.10446*, 2024.

[1249] Christian Tomani, Kamalika Chaudhuri, Ivan Evtimov, Daniel Cremers, and Mark Ibrahim. Uncertainty-based abstention in LLMs improves safety and reduces hallucinations. *arXiv preprint arXiv:2404.10960*, 2024.

[1250] Ernesto Quevedo, Jorge Yero, Rachel Koerner, Pablo Rivas, and Tomas Cerny. Detecting hallucinations in large language model generation: A token probability approach. *arXiv preprint arXiv:2405.19648*, 2024.

[1251] Shukang Yin, Chaoyou Fu, Sirui Zhao, Tong Xu, Hao Wang, Dianbo Sui, Yunhang Shen, Ke Li, Xing Sun, and Enhong Chen. Woodpecker: Hallucination correction for multimodal large language models. *Science China Information Sciences*, 67(12):220105, 2024.

[1252] Xiaowei Huang, Wenjie Ruan, Wei Huang, Gaojie Jin, Yi Dong, Changshun Wu, Saddek Bensalem, Ronghui Mu, Yi Qi, Xingyu Zhao, et al. A survey of safety and trustworthiness of large language models through the lens of verification and validation. *arXiv preprint arXiv:2309.10635*, 2023.

[1253] Shikha Bordia and Samuel R Bowman. Identifying and reducing gender bias in word-level language models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 7–15, 2019.

[1254] Kaifeng Lyu, Haoyu Zhao, Xinran Gu, Dingli Yu, Anirudh Goyal, and Sanjeev Arora. Keeping LLMs aligned after fine-tuning: The crucial role of prompt templates. *arXiv preprint arXiv:2402.18540*, 2024.

[1255] James Y Huang, Sailik Sengupta, Daniele Bonadiman, Yi-an Lai, Arshit Gupta, Nikolaos Pappas, Saab Mansour, Katrin Kirchoff, and Dan Roth. Deal: Decoding-time alignment for large language models. *arXiv preprint arXiv:2402.06147*, 2024.

[1256] Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. Safety alignment should be made more than just a few tokens deep. *arXiv preprint arXiv:2406.05946*, 2024.

[1257] Tiansheng Huang, Sihao Hu, Fatih Ilhan, Selim Furkan Tekin, and Ling Liu. Lazy safety alignment for large language models against harmful fine-tuning. *arXiv preprint arXiv:2405.18641*, 2, 2024.

[1258] Pengyu Zhu, Zhenhong Zhou, Yuanhe Zhang, Shilinlu Yan, Kun Wang, and Sen Su. Demonagent: Dynamically encrypted multi-backdoor implantation attack on llm-based agent. *arXiv preprint arXiv:2502.12575*, 2025.

[1259] Xue Tan, Hao Luan, Mingyu Luo, Xiaoyan Sun, Ping Chen, and Jun Dai. Knowledge database or poison base? detecting rag poisoning attack through LLM activations. *arXiv preprint arXiv:2411.18948*, 2024.

[1260] Biao Yi, Tiansheng Huang, Sishuo Chen, Tong Li, Zheli Liu, Chu Zhixuan, and Yiming Li. Probe before you talk: Towards black-box defense against backdoor unalignment for large language models. In *ICLR*, 2025.

[1261] Sahar Abdelnabi, Aideen Fay, Giovanni Cherubin, Ahmed Salem, Mario Fritz, and Andrew Paverd. Are you still on track!? catching LLM task drift with activations. *arXiv preprint arXiv:2406.00799*, 2024.

[1262] Xi Li, Yusen Zhang, Renze Lou, Chen Wu, and Jiaqi Wang. Chain-of-scrutiny: Detecting backdoor attacks for large language models. *arXiv preprint arXiv:2406.05948*, 2024.

[1263] Wenjie Mo, Jiashu Xu, Qin Liu, Jiongxiao Wang, Jun Yan, Chaowei Xiao, and Muhao Chen. Test-time backdoor mitigation for black-box large language models with defensive demonstrations. *arXiv preprint arXiv:2311.09763*, 2023.

[1264] Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Xiang Wang, Xiangnan He, and Tat-seng Chua. Alphaedit: Null-space constrained knowledge editing for language models. *arXiv preprint arXiv:2410.02355*, 2024.

[1265] Zongru Wu, Pengzhou Cheng, Lingyong Fang, Zhuosheng Zhang, and Gongshen Liu. Gracefully filtering backdoor samples for generative large language models without retraining. *arXiv preprint arXiv:2412.02454*, 2024.

[1266] Hanlei Zhang, Yijie Bai, Yanjiao Chen, Zhongming Ma, and Wenyuan Xu. Barbie: Robust backdoor detection based on latent separability. In *In the 32nd Annual Network and Distributed System Security Symposium (NDSS)*, 2025.

[1267] Yu He, Boheng Li, Liu Liu, Zhongjie Ba, Wei Dong, Yiming Li, Zhan Qin, Kui Ren, and Chun Chen. Towards label-only membership inference attack against pre-trained large language models. In *Proceedings of the 34th USENIX Security Symposium (USENIX Security)*. USENIX Association, 2025.

[1268] Yu He, Boheng Li, Yao Wang, Mengda Yang, Juan Wang, Hongxin Hu, and Xingyu Zhao. Is difficulty calibration all we need? towards more practical membership inference attacks. In *CCS*, pages 1226–1240, 2024.

[1269] Mansi Sakarvadia, Aswathy Ajith, Arham Khan, Nathaniel Hudson, Caleb Geniesse, Kyle Chard, Yaoqing Yang, Ian Foster, and Michael W Mahoney. Mitigating memorization in language models. *arXiv preprint arXiv:2410.02159*, 2024.

[1270] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[1271] Lynn Chua, Badih Ghazi, Yangsibo Huang, Pritish Kamath, Ravi Kumar, Daogao Liu, Pasin Manurangsi, Amer Sinha, and Chiyuan Zhang. Mind the privacy unit! user-level differential privacy for language model fine-tuning. *arXiv preprint arXiv:2406.14322*, 2024.

[1272] Panlong Wu, Kangshuo Li, Junbao Nan, and Fangxin Wang. Federated in-context llm agent learning. *arXiv preprint arXiv:2412.08054*, 2024.

[1273] Weirui Kuang, Bingchen Qian, Zitao Li, Daoyuan Chen, Dawei Gao, Xuchen Pan, Yuexiang Xie, Yaliang Li, Bolin Ding, and Jingren Zhou. Federatedscope-llm: A comprehensive package for fine-tuning large language models in federated learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5260–5271, 2024.

[1274] Pengtao Xie, Misha Bilenko, Tom Finley, Ran Gilad-Bachrach, Kristin Lauter, and Michael Naehrig. Crypto-nets: Neural networks over encrypted data. *arXiv preprint arXiv:1412.6181*, 2014.

[1275] Donghwan Rho, Taeseong Kim, Minje Park, Jung Woo Kim, Hyunsik Chae, Jung Hee Cheon, and Ernest K Ryu. Encryption-friendly LLM architecture. *arXiv preprint arXiv:2410.02486*, 2024.

[1276] Antonio Muñoz, Ruben Ríos, Rodrigo Román, and Javier López. A survey on the (in) security of trusted execution environments. *Computers & Security*, 129:103180, 2023.

[1277] Brian Knott, Shobha Venkataraman, Awni Hannun, Shubho Sengupta, Mark Ibrahim, and Laurens van der Maaten. Crypten: Secure multi-party computation meets machine learning. *Advances in Neural Information Processing Systems*, 34:4961–4973, 2021.

[1278] Junyuan Mao, Fanci Meng, Yifan Duan, Miao Yu, Xiaojun Jia, Junfeng Fang, Yuxuan Liang, Kun Wang, and Qingsong Wen. Agentsafe: Safeguarding large language model-based multi-agent systems via hierarchical data management. *arXiv preprint arXiv:2503.04392*, 2025.

[1279] Yiming Li, Mingyan Zhu, Xue Yang, Yong Jiang, Tao Wei, and Shu-Tao Xia. Black-box dataset ownership verification via backdoor watermarking. *IEEE Transactions on Information Forensics and Security*, 2023.

[1280] Junfeng Guo, Yiming Li, Lixu Wang, Shu-Tao Xia, Heng Huang, Cong Liu, and Bo Li. Domain watermark: Effective and harmless dataset copyright protection is closed at hand. In *NeurIPS*, 2023.

[1281] Boheng Li, Yanhao Wei, Yankai Fu, Zhenting Wang, Yiming Li, Jie Zhang, Run Wang, and Tianwei Zhang. Towards reliable verification of unauthorized data usage in personalized text-to-image diffusion models. In *IEEE S&P*, 2025.

[1282] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE, 2021.

[1283] Houcheng Jiang, Junfeng Fang, Ningyu Zhang, Guojun Ma, Mingyang Wan, Xiang Wang, Xiangnan He, and Tat-seng Chua. Anyedit: Edit any knowledge encoded in language models. *arXiv preprint arXiv:2502.05628*, 2025.

[1284] Xinfeng Li, Yuchen Yang, Jiangyi Deng, Chen Yan, Yanjiao Chen, Xiaoyu Ji, and Wenyuan Xu. SafeGen: Mitigating Sexually Explicit Content Generation in Text-to-Image Models. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2024.

[1285] Yijia Xiao, Yiqiao Jin, Yushi Bai, Yue Wu, Xianjun Yang, Xiao Luo, Wenchao Yu, Xujiang Zhao, Yanchi Liu, Haifeng Chen, et al. Large language models can be good privacy protection learners. In *EMNLP*, 2024.

[1286] Lingzhi Yuan, Xinfeng Li, Chejian Xu, Guanhong Tao, Xiaojun Jia, Yihao Huang, Wei Dong, Yang Liu, XiaoFeng Wang, and Bo Li. Promptguard: Soft prompt-guided unsafe content moderation for text-to-image models. *arXiv preprint arXiv:2501.03544*, 2025.

[1287] Zhexin Zhang, Leqi Lei, Lindong Wu, Rui Sun, Yongkang Huang, Chong Long, Xiao Liu, Xuanyu Lei, Jie Tang, and Minlie Huang. Safetybench: Evaluating the safety of large language models with multiple choice questions. *arXiv preprint arXiv:2309.07045*, 2023.

[1288] Liang Xu, Kangkang Zhao, Lei Zhu, and Hang Xue. Sc-safety: A multi-round open-ended question adversarial safety benchmark for large language models in chinese. *arXiv preprint arXiv:2310.05818*, 2023.

[1289] Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe rlhf: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773*, 2023.

[1290] Junfeng Fang, Wei Liu, Yuan Gao, Zemin Liu, An Zhang, Xiang Wang, and Xiangnan He. Evaluating post-hoc explanations for graph neural networks via robustness analysis. *Advances in neural information processing systems*, 36:72446–72463, 2023.

[1291] Mansi Phute, Alec Helbling, Matthew Daniel Hull, ShengYun Peng, Sebastian Szyller, Cory Cornelius, and Duen Horng Chau. LLM self defense: By self examination, LLMs know they are being tricked. In *The Second Tiny Papers Track at ICLR 2024*, 2023.

[1292] Zhexin Zhang, Junxiao Yang, Pei Ke, Fei Mi, Hongning Wang, and Minlie Huang. Defending large language models against jailbreaking attacks through goal prioritization. *arXiv preprint arXiv:2311.09096*, 2023.

[1293] Julien Piet, Maha Alrashed, Chawin Sitawarin, Sizhe Chen, Zeming Wei, Elizabeth Sun, Basel Alomair, and David Wagner. Jatmo: Prompt injection defense by task-specific finetuning. In *European Symposium on Research in Computer Security*, pages 105–124. Springer, 2024.

[1294] Kun Wang, Yuxuan Liang, Xinglin Li, Guohao Li, Bernard Ghanem, Roger Zimmermann, Zhengyang Zhou, Huahui Yi, Yudong Zhang, and Yang Wang. Brave the wind and the waves: Discovering robust and generalizable graph lottery tickets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):3388–3405, 2023.

[1295] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[1296] Peng Xu, Xiatian Zhu, and David A Clifton. Multimodal learning with transformers: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(10):12113–12132, 2023.

[1297] Xinfeng Li, Chen Yan, Xuancun Lu, Zihan Zeng, Xiaoyu Ji, and Wenyuan Xu. Inaudible adversarial perturbation: Manipulating the recognition of user speech in real time. *arXiv preprint arXiv:2308.01040*, 2023.

[1298] Elias Abad Rocamora, Yongtao Wu, Fanghui Liu, Grigorios Chrysos, and Volkan Cevher. Revisiting character-level adversarial attacks for language models. In *41st International Conference on Machine Learning (ICML 2024)*, 2024.

[1299] Yuxin Wen, Neel Jain, John Kirchenbauer, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. *Advances in Neural Information Processing Systems*, 36, 2024.

[1300] Jialin Wu, Jiangyi Deng, Shengyuan Pang, Yanjiao Chen, Jiayang Xu, Xinfeng Li, and Wenyuan Xu. Legilimens: Practical and unified content moderation for large language model services. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 1151–1165, 2024.

[1301] Hannah Brown, Leon Lin, Kenji Kawaguchi, and Michael Shieh. Self-evaluation as a defense against adversarial attacks on llms. *arXiv preprint arXiv:2407.03234*, 2024.

[1302] Raha Moraffah, Shubh Khandelwal, Amrita Bhattacharjee, and Huan Liu. Adversarial text purification: A large language model approach for defense. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 65–77. Springer, 2024.

[1303] Lujia Shen, Xuhong Zhang, Shouling Ji, Yuwen Pu, Chunpeng Ge, Xing Yang, and Yanghe Feng. Textdefense: Adversarial text detection based on word importance entropy. *arXiv preprint arXiv:2302.05892*, 2023.

[1304] Luke Bailey, Euan Ong, Stuart Russell, and Scott Emmons. Image hijacks: Adversarial images can control generative models at runtime. *arXiv preprint arXiv:2309.00236*, 2023.

[1305] Dongchen Han, Xiaojun Jia, Yang Bai, Jindong Gu, Yang Liu, and Xiaochun Cao. Ot-attack: Enhancing adversarial transferability of vision-language models via optimal transport optimization. *arXiv preprint arXiv:2312.04403*, 2023.

[1306] Sensen Gao, Xiaojun Jia, Xuhong Ren, Ivor Tsang, and Qing Guo. Boosting transferability in vision-language attacks via diversification along the intersection region of adversarial trajectory. In *European Conference on Computer Vision*, pages 442–460. Springer, 2024.

[1307] Linhao Huang, Xue Jiang, Zhiqiang Wang, Wentao Mo, Xi Xiao, Bo Han, Yongjie Yin, and Feng Zheng. Image-based multimodal models as intruders: Transferable multimodal attacks on video-based mllms. *arXiv preprint arXiv:2501.01042*, 2025.

[1308] Chen Henry Wu, Rishi Rajesh Shah, Jing Yu Koh, Russ Salakhutdinov, Daniel Fried, and Aditi Raghunathan. Dissecting adversarial robustness of multimodal lm agents. In *The Thirteenth International Conference on Learning Representations*, 2025.

[1309] Xiaoyu Ji, Yushi Cheng, Yuepeng Zhang, Kai Wang, Chen Yan, Wenyuan Xu, and Kevin Fu. Poltergeist: Acoustic adversarial machine learning against cameras and computer vision. In *2021 IEEE symposium on security and privacy (SP)*, pages 160–175. IEEE, 2021.

[1310] Xiaojun Jia, Yong Zhang, Baoyuan Wu, Ke Ma, Jue Wang, and Xiaochun Cao. Las-at: adversarial training with learnable attack strategy. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13398–13408, 2022.

[1311] Xiaojun Jia, Yong Zhang, Xingxing Wei, Baoyuan Wu, Ke Ma, Jue Wang, and Xiaochun Cao. Improving fast adversarial training with prior-guided knowledge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[1312] Xiaojun Jia, Sensen Gao, Simeng Qin, Ke Ma, Xinfeng Li, Yihao Huang, Wei Dong, Yang Liu, and Xiaochun Cao. Evolution-based region adversarial prompt learning for robustness enhancement in vision-language models. *arXiv preprint arXiv:2503.12874*, 2025.

[1313] Caixin Kang, Yinpeng Dong, Zhengyi Wang, Shouwei Ruan, Yubo Chen, Hang Su, and Xingxing Wei. Diffender: Diffusion-based adversarial defense against patch attacks. In *European Conference on Computer Vision*, pages 130–147. Springer, 2024.

[1314] Xilie Xu, Keyi Kong, Ning Liu, Lizhen Cui, Di Wang, Jingfeng Zhang, and Mohan Kankanhalli. An LLM can fool itself: A prompt-based adversarial attack. *arXiv preprint arXiv:2310.13345*, 2023.

[1315] Zhicong Zheng, Xinfeng Li, Chen Yan, Xiaoyu Ji, and Wenyuan Xu. The silent manipulator: A practical and inaudible backdoor attack against speech recognition systems. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 7849–7858, 2023.

[1316] Xinfeng Li, Junning Ze, Chen Yan, Yushi Cheng, Xiaoyu Ji, and Wenyuan Xu. Enrollment-stage backdoor attacks on speaker recognition systems via adversarial ultrasound. *IEEE Internet of Things Journal*, 2023.

[1317] Junning Ze, Xinfeng Li, Yushi Cheng, Xiaoyu Ji, and Wenyuan Xu. Ultrabd: Backdoor attack against automatic speaker verification systems via adversarial ultrasound. In *2022 IEEE 28th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 193–200. IEEE, 2023.

[1318] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. Dolphinattack: Inaudible voice commands. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 103–117, 2017.

[1319] Junae Kim and Amardeep Kaur. A survey on adversarial robustness of lidar-based machine learning perception in autonomous vehicles. *arXiv preprint arXiv:2411.13778*, 2024.

[1320] James Tu, Tsunhsuan Wang, Jingkang Wang, Sivabalan Manivasagam, Mengye Ren, and Raquel Urtasun. Adversarial attacks on multi-agent communication. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7768–7777, 2021.

[1321] Yunmok Son, Hocheol Shin, Dongkwan Kim, Youngseok Park, Juhwan Noh, Kibum Choi, Jungwoo Choi, and Yongdae Kim. Rocking drones with intentional sound noise on gyroscopic sensors. In *24th USENIX security symposium (USENIX Security 15)*, pages 881–896, 2015.

[1322] Mohsin Kamal, Arnab Barua, Christian Vitale, Christos Laoudias, and Georgios Ellinas. Gps location spoofing attack detection for enhancing the security of autonomous vehicles. In *2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall)*, pages 1–7. IEEE, 2021.

[1323] Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. Grounding large language models in interactive environments with online reinforcement learning. In *International Conference on Machine Learning*, pages 3676–3713. PMLR, 2023.

[1324] Isabel O Gallegos, Ryan A Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K Ahmed. Bias and fairness in large language models: A survey. *Computational Linguistics*, pages 1–79, 2024.

[1325] Divyat Mahajan, Shruti Tople, and Amit Sharma. Domain generalization using causal matching. In *International conference on machine learning*, pages 7313–7324. PMLR, 2021.

[1326] Osama Mazhar, Robert Babuška, and Jens Kober. Gem: Glare or gloom, i can still see you–end-to-end multi-modal object detection. *IEEE Robotics and Automation Letters*, 6(4):6321–6328, 2021.

[1327] Lizhou Fan, Wenyue Hua, Lingyao Li, Haoyang Ling, and Yongfeng Zhang. Nphardeval: Dynamic benchmark on reasoning ability of large language models via complexity classes. *arXiv preprint arXiv:2312.14890*, 2023.

[1328] Daniele Vilone and Eugenia Polizzi. Modeling opinion misperception and the emergence of silence in online social system. *Plos one*, 19(1):e0296075, 2024.

[1329] Runsheng Xu, Jinlong Li, Xiaoyu Dong, Hongkai Yu, and Jiaqi Ma. Bridging the domain gap for multi-agent perception. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6035–6042. IEEE, 2023.

[1330] Heechang Ryu, Hayong Shin, and Jinkyoo Park. Cooperative and competitive biases for multi-agent reinforcement learning. *arXiv preprint arXiv:2101.06890*, 2021.

[1331] Xenia Ohmer, Michael Marino, Michael Franke, and Peter König. Mutual influence between language and perception in multi-agent communication games. *PLoS computational biology*, 18(10):e1010658, 2022.

[1332] Runsheng Xu, Weizhe Chen, Hao Xiang, Xin Xia, Lantao Liu, and Jiaqi Ma. Model-agnostic multi-agent perception framework. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1471–1478. IEEE, 2023.

[1333] Fangzhou Wu, Ning Zhang, Somesh Jha, Patrick McDaniel, and Chaowei Xiao. A new era in LLM security: Exploring security concerns in real-world LLM-based systems. *arXiv preprint arXiv:2402.18649*, 2024.

[1334] Junjie Ye, Sixian Li, Guanyu Li, Caishuang Huang, Songyang Gao, Yilong Wu, Qi Zhang, Tao Gui, and Xuanjing Huang. Toolsword: Unveiling safety issues of large language models in tool learning across three stages. *arXiv preprint arXiv:2402.10753*, 2024.

[1335] Xinfeng Li, Zhicong Zheng, Chen Yan, Chaohao Li, Xiaoyu Ji, and Wenyuan Xu. Toward pitch-insensitive speaker verification via soundfield. *IEEE Internet of Things Journal*, 11(1):1175–1189, 2023.

[1336] Wanqi Yang, Yanda Li, Meng Fang, Yunchao Wei, Tianyi Zhou, and Ling Chen. Who can withstand chat-audio attacks? an evaluation benchmark for large language models. *arXiv preprint arXiv:2411.14842*, 2024.

[1337] Guoming Zhang, Xiaoyu Ji, Xinfeng Li, Gang Qu, and Wenyuan Xu. Eararray: Defending against dolphinattack via acoustic attenuation. In *In the 28th Annual Network and Distributed System Security Symposium (NDSS)*, 2021.

[1338] Raghuveer Peri, Sai Muralidhar Jayanthi, Srikanth Ronanki, Anshu Bhatia, Karel Mundnich, Saket Dingliwal, Nilaksh Das, Zejiang Hou, Goeric Huybrechts, Srikanth Vishnubhotla, et al. Speechguard: Exploring the adversarial robustness of multimodal large language models. *arXiv preprint arXiv:2405.08317*, 2024.

[1339] Xinfeng Li, Xiaoyu Ji, Chen Yan, Chaohao Li, Yichen Li, Zhenning Zhang, and Wenyuan Xu. Learning normality is enough: A software-based mitigation against inaudible voice attacks. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 2455–2472, 2023.

[1340] Wenyuan Xu, Chen Yan, Weibin Jia, Xiaoyu Ji, and Jianhao Liu. Analyzing and enhancing the security of ultrasonic sensors for autonomous vehicles. *IEEE Internet of Things Journal*, 5(6):5015–5029, 2018.

[1341] Ruixu Geng, Jianyang Wang, Yuqin Yuan, Fengquan Zhan, Tianyu Zhang, Rui Zhang, Pengcheng Huang, Dongheng Zhang, Jinbo Chen, Yang Hu, et al. A survey of wireless sensing security from a role-based view: Victim, weapon, and shield. *arXiv preprint arXiv:2412.03064*, 2024.

[1342] Xiaoyu Ji, Wenjun Zhu, Shilin Xiao, and Wenyuan Xu. Sensor-based iot data privacy protection. *Nature Reviews Electrical Engineering*, 1(7):427–428, 2024.

[1343] Basudha Pal, Aniket Roy, Ram Prabhakar Kathirvel, Alice J O'Toole, and Rama Chellappa. Diversinet: Mitigating bias in deep classification networks across sensitive attributes through diffusion-generated data. In *2024 IEEE International Joint Conference on Biometrics (IJCB)*, pages 1–10. IEEE, 2024.

[1344] Pedro Mendes, Paolo Romano, and David Garlan. Error-driven uncertainty aware training. *arXiv preprint arXiv:2405.01205*, 2024.

[1345] Clayton Sanford, Bahare Fatemi, Ethan Hall, Anton Tsitsulin, Mehran Kazemi, Jonathan Halcrow, Bryan Perozzi, and Vahab Mirrokni. Understanding transformer reasoning capabilities via graph algorithms. *arXiv preprint arXiv:2405.18512*, 2024.

[1346] Stephen Grossberg. A path toward explainable ai and autonomous adaptive intelligence: deep learning, adaptive resonance, and models of perception, emotion, and action. *Frontiers in neurorobotics*, 14:36, 2020.

[1347] Donghee Shin. The effects of explainability and causability on perception, trust, and acceptance: Implications for explainable ai. *International journal of human-computer studies*, 146:102551, 2021.

[1348] Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2022.

[1349] Jingwei Yi, Yueqi Xie, Bin Zhu, Emre Kiciman, Guangzhong Sun, Xing Xie, and Fangzhao Wu. Benchmarking and defending against indirect prompt injection attacks on large language models. *arXiv preprint arXiv:2312.14197*, 2023.

[1350] Keegan Hines, Gary Lopez, Matthew Hall, Federico Zarfati, Yonatan Zunger, and Emre Kiciman. Defending against indirect prompt injection attacks with spotlighting. *arXiv preprint arXiv:2403.14720*, 2024.

[1351] Kaijie Zhu, Xianjun Yang, Jindong Wang, Wenbo Guo, and William Yang Wang. Melon: Indirect prompt injection defense via masked re-execution and tool comparison. *arXiv preprint arXiv:2502.05174*, 2025.

[1352] Maxwell Crouse, Ibrahim Abdelaziz, Kinjal Basu, Soham Dan, Sadhana Kumaravel, Achille Fokoue, Pavan Kapanipathi, and Luis Lastras. Formally specifying the high-level behavior of LLM-based agents. *arXiv preprint arXiv:2312.04572*, 2023.

[1353] Ayush RoyChowdhury, Mulong Luo, Prateek Sahu, Sarbartha Banerjee, and Mohit Tiwari. Confusedpilot: Confused deputy risks in rag-based llms. *arXiv preprint arXiv:2408.04870*, 2024.

[1354] Wei Zou, Runpeng Geng, Binghui Wang, and Jinyuan Jia. Poisonedrag: Knowledge corruption attacks to retrieval-augmented generation of large language models. *arXiv preprint arXiv:2402.07867*, 2024.

[1355] Avital Shafran, Roei Schuster, and Vitaly Shmatikov. Machine against the rag: Jamming retrieval-augmented generation with blocker documents. *arXiv preprint arXiv:2406.05870*, 2024.

[1356] Jiaqi Xue, Mengxin Zheng, Yebowen Hu, Fei Liu, Xun Chen, and Qian Lou. Badrag: Identifying vulnerabilities in retrieval augmented generation of large language models. *arXiv preprint arXiv:2406.00083*, 2024.

[1357] Pengzhou Cheng, Yidong Ding, Tianjie Ju, Zongru Wu, Wei Du, Ping Yi, Zhuosheng Zhang, and Gongshen Liu. Trojanrag: Retrieval-augmented generation can be backdoor driver in large language models. *arXiv preprint arXiv:2405.13401*, 2024.

[1358] Quanyu Long, Yue Deng, LeiLei Gan, Wenya Wang, and Sinno Jialin Pan. Whispers in grammars: Injecting covert backdoors to compromise dense retrieval systems. *arXiv preprint arXiv:2402.13532*, 2024.

[1359] Anastasios Giannaros, Aristeidis Karras, Leonidas Theodorakopoulos, Christos Karras, Panagiotis Kranias, Nikolaos Schizas, Gerasimos Kalogeratos, and Dimitrios Tsolis. Autonomous vehicles: Sophisticated attacks, safety issues, challenges, open topics, blockchain, and future directions. *Journal of Cybersecurity and Privacy*, 3(3):493–543, 2023.

[1360] Kurt Geihs. Engineering challenges ahead for robot teamwork in dynamic environments. *Applied Sciences*, 10 (4):1368, 2020.

[1361] Shah Zahid Khan, Mujahid Mohsin, and Waseem Iqbal. On gps spoofing of aerial platforms: a review of threats, challenges, methodologies, and future research directions. *PeerJ Computer Science*, 7:e507, 2021.

[1362] Jonathan Petit, Baris Stottelaar, Manfred Feiri, and Frank Kargl. Remote attacks on automated vehicles sensors: Experiments on camera and lidar. *Black Hat Europe*, 111:99–108, 2015.

[1363] Jianying Zhou, Zhenfu Cao, Xiaolei Dong, and Athanasios V Vasilakos. Security and privacy in cyber-physical systems: A survey. *IEEE communications surveys & tutorials*, 19(2):1197–1229, 2017.

[1364] Yulong Cao, Chaowei Xiao, Dawei Yang, Jing Fang, Ruigang Yang, Mingyan Liu, and Bo Li. Adversarial objects against lidar-based autonomous driving systems. *arXiv preprint arXiv:1907.05418*, 2019.

[1365] Sehoon Ha, Peng Xu, Zhenyu Tan, Sergey Levine, and Jie Tan. Learning to walk in the real world with minimal human effort. *arXiv preprint arXiv:2002.08550*, 2020.

[1366] Xiangru Tang, Qiao Jin, Kunlun Zhu, Tongxin Yuan, Yichi Zhang, Wangchunshu Zhou, Meng Qu, Yilun Zhao, Jian Tang, Zhuosheng Zhang, Arman Cohan, Zhiyong Lu, and Mark Gerstein. Prioritizing safeguarding over autonomy: Risks of llm agents for science, 2024. URL https://arxiv.org/abs/2402.04247.

[1367] Tong Liu, Zizhuang Deng, Guozhu Meng, Yuekang Li, and Kai Chen. Demystifying rce vulnerabilities in llm-integrated apps. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 1716–1730, 2024.

[1368] Michael Guastalla, Yiyi Li, Arvin Hekmati, and Bhaskar Krishnamachari. Application of large language models to ddos attack detection. In *International Conference on Security and Privacy in Cyber-Physical Systems and Smart Vehicles*, pages 83–99. Springer, 2023.

[1369] Jonas Geiping, Alex Stein, Manli Shu, Khalid Saifullah, Yuxin Wen, and Tom Goldstein. Coercing LLMs to do and reveal (almost) anything. *arXiv preprint arXiv:2402.14020*, 2024.

[1370] Zeyi Liao, Lingbo Mo, Chejian Xu, Mintong Kang, Jiawei Zhang, Chaowei Xiao, Yuan Tian, Bo Li, and Huan Sun. Eia: Environmental injection attack on generalist web agents for privacy leakage. *arXiv preprint arXiv:2409.11295*, 2024.

[1371] Chejian Xu, Mintong Kang, Jiawei Zhang, Zeyi Liao, Lingbo Mo, Mengqi Yuan, Huan Sun, and Bo Li. Advweb: Controllable black-box attacks on vlm-powered web agents. *arXiv preprint arXiv:2410.17401*, 2024.

[1372] Weidi Luo, Shenghong Dai, Xiaogeng Liu, Suman Banerjee, Huan Sun, Muhao Chen, and Chaowei Xiao. Agrail: A lifelong agent guardrail with effective and adaptive safety detection. *arXiv preprint arXiv:2502.11448*, 2025.

[1373] Lewis Hammond, Alan Chan, Jesse Clifton, Jason Hoelscher-Obermaier, Akbir Khan, Euan McLean, Chandler Smith, Wolfram Barfuss, Jakob Foerster, Tomáš Gavenčiak, et al. Multi-agent risks from advanced ai. *arXiv preprint arXiv:2502.14143*, 2025.

[1374] Aidan O'Gara. Hoodwinked: Deception and cooperation in a text-based game for language models. *arXiv preprint arXiv:2308.01404*, 2023.

[1375] Kanghua Mo, Weixuan Tang, Jin Li, and Xu Yuan. Attacking deep reinforcement learning with decoupled adversarial policy. *IEEE Transactions on Dependable and Secure Computing*, 20(1):758–768, 2022.

[1376] Guanghui Wen, Peijun Wang, Yuezu Lv, Guanrong Chen, and Jialing Zhou. Secure consensus of multi-agent systems under denial-of-service attacks. *Asian Journal of Control*, 25(2):695–709, 2023.

[1377] Sumeet Ramesh Motwani, Mikhail Baranchuk, Lewis Hammond, and Christian Schroeder de Witt. A Perfect Collusion Benchmark: How can AI agents be prevented from colluding with information-theoretic undetectability? In *Multi-Agent Security Workshop NeurIPS 23*, 2023.

[1378] Yikang Pan, Liangming Pan, Wenhu Chen, Preslav Nakov, Min-Yen Kan, and William Yang Wang. On the risk of misinformation pollution with large language models. In *arXiv preprint arXiv:2305.13661*, 2023.

[1379] Xiangming Gu, Xiaosen Zheng, Tianyu Pang, Chao Du, Qian Liu, Ye Wang, Jing Jiang, and Min Lin. Agent smith: A single image can jailbreak one million multimodal LLM agents exponentially fast. *arXiv preprint arXiv:2402.08567*, 2024.

[1380] Dan Zhang, Gang Feng, Yang Shi, and Dipti Srinivasan. Physical safety and cyber security analysis of multi-agent systems: A survey of recent advances. *IEEE/CAA Journal of Automatica Sinica*, 8(2):319–333, 2021.

[1381] Silen Naihin, David Atkinson, Marc Green, Merwane Hamadi, Craig Swift, Douglas Schonholtz, Adam Tauman Kalai, and David Bau. Testing language model agents safely in the wild. *arXiv preprint arXiv:2311.10538*, 2023.

[1382] Tongxin Yuan, Zhiwei He, Lingzhong Dong, Yiming Wang, Ruijie Zhao, Tian Xia, Lizhen Xu, Binglin Zhou, Fangqi Li, Zhuosheng Zhang, et al. R-judge: Benchmarking safety risk awareness for LLM agents. *arXiv preprint arXiv:2401.10019*, 2024.

[1383] Andres M Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D White, and Philippe Schwaller. Chemcrow: Augmenting large-language models with chemistry tools. *arXiv preprint arXiv:2304.05376*, 2023.

[1384] Naruki Yoshikawa, Marta Skreta, Kourosh Darvish, Sebastian Arellano-Rubach, Zhi Ji, Lasse Bjørn Kristensen, Andrew Zou Li, Yuchi Zhao, Haoping Xu, Artur Kuramshin, et al. Large language models for chemistry robotics. *Autonomous Robots*, 47(8):1057–1086, 2023.

[1385] Jiyan He, Weitao Feng, Yaosen Min, Jingwei Yi, Kunsheng Tang, Shuai Li, Jie Zhang, Kejiang Chen, Wenbo Zhou, Xing Xie, et al. Control risk for potential misuse of artificial intelligence in science. *arXiv preprint arXiv:2312.06632*, 2023.

[1386] Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, et al. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.

[1387] Usman Anwar, Abulhair Saparov, Javier Rando, Daniel Paleka, Miles Turpin, Peter Hase, Ekdeep Singh Lubana, Erik Jenner, Stephen Casper, Oliver Sourbut, et al. Foundational challenges in assuring alignment and safety of large language models. *arXiv preprint arXiv:2404.09932*, 2024.

[1388] OpenAI. Introducing superalignment. https://openai.com/index/introducing-superalignment/, July 2023. Accessed: 2025-03-26.

[1389] Eliezer Yudkowsky. Ai alignment: Why it's hard, and where to start. *Symbolic Systems Distinguished Speaker Series*, 2016.

[1390] David Krueger, Tegan Maharaj, and Jan Leike. Hidden incentives for auto-induced distributional shift. *arXiv preprint arXiv:2009.09153*, 2020.

[1391] Joseph Carlsmith. Is power-seeking ai an existential risk? *arXiv preprint arXiv:2206.13353*, 2022.

[1392] Paul Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, pages 4299–4307, 2017.

[1393] Jan Leike, David Krueger, Tom Everitt, Miljan Martic, Vishal Maini, and Shane Legg. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*, 2018.

[1394] Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021, 2020.

[1395] Geoffrey Irving and Amanda Askell. Ai safety needs social scientists. *Distill*, 4(2):e14, 2019.

[1396] Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, et al. Weak-to-strong generalization: eliciting strong capabilities with weak supervision. In *Proceedings of the 41st International Conference on Machine Learning*, pages 4971–5012, 2024.

[1397] Evan Hubinger, Chris van Merwijk, Vladimir Mikulik, Joar Skalse, and Scott Garrabrant. Risks from learned optimization in advanced machine learning systems. *arXiv preprint arXiv:1906.01820*, 2019.

[1398] Hao Zhang, Ramesh Kumar, and Wei Li. Balancing immediate accuracy and long-term goal adherence in reinforcement learning. In *Proceedings of the 40th Conference on Neural Information Processing Systems (NeurIPS 2023)*, pages 1234–1245, 2023.

[1399] Ming Wu, Lei Zhao, and Jun Chen. Dynamic calibration of composite rewards for robust reinforcement learning. In *Proceedings of the 2023 International Conference on Learning Representations (ICLR 2023)*, pages 567–578, 2023.

[1400] Xiangwen Wang, Yibo Jacky Zhang, Zhoujie Ding, Katherine Tsai, and Sanmi Koyejo. Aligning compound ai systems via system-level dpo. *arXiv preprint arXiv:2502.17721*, 2025.

[1401] Jixuan Leng, Chengsong Huang, Banghua Zhu, and Jiaxin Huang. Taming overconfidence in llms: Reward calibration in rlhf. *arXiv preprint arXiv:2410.09724*, 2024.

[1402] Chenghua Huang, Zhizhen Fan, Lu Wang, Fangkai Yang, Pu Zhao, Zeqi Lin, Qingwei Lin, Dongmei Zhang, Saravan Rajmohan, and Qi Zhang. Self-evolved reward learning for llms. *arXiv preprint arXiv:2411.00418*, 2024.

[1403] Yoshua Bengio, Geoffrey Hinton, Andrew Yao, Dawn Song, Pieter Abbeel, Trevor Darrell, Yuval Noah Harari, Ya-Qin Zhang, Lan Xue, Shai Shalev-Shwartz, et al. Managing extreme ai risks amid rapid progress. *Science*, 384(6698):842–845, 2024.

[1404] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[1405] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[1406] Zonghao Ying, Aishan Liu, Siyuan Liang, Lei Huang, Jinyang Guo, Wenbo Zhou, Xianglong Liu, and Dacheng Tao. Safebench: A safety evaluation framework for multimodal large language models. *arXiv preprint arXiv:2410.18927*, 2024.

[1407] Reda Alami, Ali Khalifa Almansoori, Ahmed Alzubaidi, Mohamed El Amine Seddik, Mugariya Farooq, and Hakim Hacid. Alignment with preference optimization is all you need for LLM safety. *arXiv preprint arXiv:2409.07772*, 2024.

[1408] Huayu Chen, Guande He, Lifan Yuan, Ganqu Cui, Hang Su, and Jun Zhu. Noise contrastive alignment of language models with explicit rewards. *arXiv preprint arXiv:2402.05369*, 2024.

[1409] Yi-Lin Tuan, Xilun Chen, Eric Michael Smith, Louis Martin, Soumya Batra, Asli Celikyilmaz, William Yang Wang, and Daniel M Bikel. Towards safety and helpfulness balanced responses via controllable large language models. *arXiv preprint arXiv:2404.01295*, 2024.

[1410] Suyu Ge, Chunting Zhou, Rui Hou, Madian Khabsa, Yi-Chia Wang, Qifan Wang, Jiawei Han, and Yuning Mao. Mart: Improving LLM safety with multi-round automatic red-teaming. *arXiv preprint arXiv:2311.07689*, 2023.

[1411] Zaifan Jiang, Xing Huang, and Chao Wei. Preference as reward, maximum preference optimization with importance sampling. *arXiv preprint arXiv:2312.16430*, 2023.

[1412] Sayak Ray Chowdhury, Anush Kini, and Nagarajan Natarajan. Provably robust dpo: Aligning language models with noisy feedback. *arXiv preprint arXiv:2403.00409*, 2024.

[1413] Yao Zhao, Misha Khalman, Rishabh Joshi, Shashi Narayan, Mohammad Saleh, and Peter J Liu. Calibrating sequence likelihood improves conditional language generation. *arXiv preprint arXiv:2210.00045*, 2022.

[1414] Yang Chao, Lu Chaochao, Wang Yingchun, and Zhou Bowen. Towards AI-45° law: A roadmap to trustworthy AGI. *arXiv preprint arXiv:2412.14186*, 2024.

[1415] Artificial Analysis. LLM Leaderboard - Compare GPT-4o, Llama 3, Mistral, Gemini & other models, 2024. URL https://artificialanalysis.ai/leaderboards/models.

[1416] Clémentine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. Open llm leaderboard v2, 2024. URL https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard.