
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 Chen¹, **Huan Zhang**^{2,3}, **Zhaoyang Yu**¹, **Haochen Shi**^{2,3}, **Boyan Li**¹⁰, **Dekun Wu**^{2,3}, **Fengwei Teng**¹,
Xiaojun Jia⁴, **Jiawei Xu**¹, **Jinyu Xiang**¹, **Yizhang Lin**¹, **Tianming Liu**¹⁴, **Tongliang Liu**⁶,
Yu Su¹⁵, **Huan Sun**¹⁵, **Glen Berseth**^{2,3,20}, **Jianyun Nie**², **Ian Foster**⁵, **Logan Ward**⁵, **Qingyun Wu**⁷,
Yu Gu¹⁵, **Mingchen Zhuge**¹⁶, **Xinbing Liang**¹, **Xiangru Tang**¹², **Haohan Wang**⁹, **Jiaxuan You**⁹, **Chi Wang**¹⁹,
Jian Pei^{17†}, **Qiang Yang**^{10,18†}, **Xiao-Liang Qi**^{13†}, **Chenglin Wu**^{1*†}

¹MetaGPT, ²Université de Montréal, ³Mila - Quebec AI Institute, ⁴Nanyang Technological University,

⁵Argonne National Laboratory, ⁶University of Sydney, ⁷Penn State University, ⁸Microsoft Research Asia,

⁹University of Illinois at Urbana-Champaign, ¹⁰The Hong Kong University of Science and Technology,

¹¹University of Southern California, ¹²Yale University, ¹³Stanford University, ¹⁴University of Georgia,

¹⁵The Ohio State University, ¹⁶King Abdullah University of Science and Technology, ¹⁷Duke University,

¹⁸The Hong Kong Polytechnic University, ¹⁹Google DeepMind, ²⁰Canada 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 book provides a comprehensive overview, framing intelligent agents within modular, brain-inspired architectures that integrate principles from cognitive science, neuroscience, and computational research. We structure our exploration into four interconnected parts. First, we systematically investigate 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, goal, and emotion. 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. 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 and beneficial AI systems**, emphasizing intrinsic and extrinsic security threats, ethical alignment, robustness, and practical mitigation strategies necessary for trustworthy real-world deploy-

*Major Contribution.

†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)

ment. 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>.

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 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 book 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

1	Introduction	16
1.1	The Rise and Development of AI Agents	16
1.2	A Parallel Comparison between Human Brain and AI Agents	17
1.2.1	Brain Functionalities and AI Parallels	18
1.3	Foundation Agents: A Modular and Brain-Inspired AI Agent Framework	27
1.3.1	From Language Models to AI Agents	28
1.3.2	Core Concepts and Notations in the Agent Loop	30
1.3.3	Biological Inspirations	34
1.3.4	Connections to Existing Theories	35
1.4	Navigating This Book	36
I	Core Components of Intelligent Agents	38
2	Cognition	40
2.1	Learning	40
2.1.1	A Unified Formulation of Learning	41
2.1.2	Learning Across Mental State Components	42
2.1.3	Learning Space	44
2.1.4	Learning Objective	47
2.2	Reasoning	51
2.2.1	A Unified Formulation of Reasoning	53
2.2.2	Structured Reasoning	55
2.2.3	Unstructured Reasoning	58
2.2.4	Planning	60
2.3	Summary and Discussion	62

3 Memory	64
3.1 Overview of Human Memory	64
3.1.1 Types of Human Memory	65
3.1.2 Models of Human Memory	66
3.2 From Human Memory to Agent Memory	69
3.3 Representation of Agent Memory	70
3.3.1 Sensory Memory	71
3.3.2 Short-Term Memory	73
3.3.3 Long-Term Memory	75
3.4 The Memory Lifecycle	76
3.4.1 Memory Acquisition	77
3.4.2 Memory Encoding	78
3.4.3 Memory Derivation	80
3.4.4 Memory Retrieval and Matching	81
3.4.5 Neural Memory Networks	82
3.4.6 Memory Utilization	85
3.5 Summary and Discussion	86
4 World Model	88
4.1 Mental Models as Human World Representations	89
4.2 From Human Mental Models to AI World Models	90
4.3 Paradigms of AI World Models	90
4.3.1 Overview of World Model Paradigms	91
4.3.2 Implicit Paradigm	92
4.3.3 Explicit Paradigm	93
4.3.4 Simulator-Based Paradigm	94
4.3.5 Hybrid and Instruction-Driven Paradigms	95
4.3.6 Comparative Summary of Paradigms	95
4.4 Relationships to Other Modules	96
4.4.1 Memory and the World Model	96
4.4.2 Perception and the World Model	97
4.4.3 Action and the World Model	97
4.4.4 Cross-Module Integration	98
4.5 Summary and Discussion	98

5 Reward	101
5.1 The Human Reward Pathway	101
5.2 From Human Rewards to Agent Rewards	103
5.3 AI Reward Paradigms	105
5.3.1 Extrinsic Rewards	107
5.3.2 Intrinsic Rewards	108
5.3.3 Hybrid Rewards	109
5.3.4 Hierarchical Rewards	109
5.4 Interaction with Other Modules	110
5.5 Summary and Discussion	111
6 Emotion Modeling	112
6.1 Psychological Foundations of Emotion	112
6.2 Incorporating Emotions in AI Agents	115
6.3 Understanding Human Emotions through AI	116
6.4 Analyzing AI Emotions and Personality	117
6.5 Manipulating AI Emotional Responses	117
6.6 Summary and Discussion	118
7 Perception	120
7.1 Human versus AI Perception	120
7.2 Types of Perception Representation	122
7.2.1 Formulation of Perception	122
7.2.2 Unimodal Models	124
7.2.3 Cross-modal Models	126
7.2.4 Multimodal Models	128
7.3 Optimizing Perception Systems	131
7.3.1 Model-Level Enhancements	131
7.3.2 System-Level Optimizations	131
7.3.3 External Feedback and Control	132
7.4 Real-World Applications of Perceptual Intelligence	132
7.5 Summary and Discussion	133
8 Action Systems	135
8.1 The Human Action System	136
8.2 From Human Action to Agentic Action	136

8.3	Paradigms of Agentic Action System	139
8.3.1	Action Space Paradigm	140
8.3.2	Action Learning Paradigm	142
8.3.3	Tool-Based Action Paradigm	145
8.3.4	Agent Learning Paradigm	148
8.4	Action and Perception: “Outside-In” or “Inside-out”	149
8.5	Summary and Discussion	150
II	Self-Evolution in Intelligent Agents	153
9	Dimensions of Self-Optimization in Intelligent Agents	156
9.1	Overview of Agent Optimization	156
9.2	Prompt Optimization	158
9.2.1	Evaluation Functions	158
9.2.2	Optimization Functions	160
9.2.3	Evaluation Metrics	160
9.3	Workflow Optimization	161
9.3.1	Workflow Formulation	161
9.3.2	Optimizing Workflow Edges	162
9.3.3	Optimizing Workflow Nodes	163
9.4	Tool Optimization	163
9.4.1	Learning to Use Tools	164
9.4.2	Creation of New Tools	165
9.4.3	Evaluation of Tool Effectiveness	166
9.5	Towards Autonomous Agent Optimization	169
9.6	Summary and Discussion	170
10	Harnessing Large Language Models for Iterative Optimization	171
10.1	Optimization Paradigms	171
10.2	Iterative Approaches to LLM Optimization	172
10.3	Optimization Hyperparameters	175
10.4	Dynamic and Iterative Optimization in LLM Workflows	176
10.5	Theoretical Insights into Transformer Optimization	176
10.6	Summary and Discussion	177

11 Online and Offline Agent Self-Improvement	179
11.1 Online Agent Self-Improvement	179
11.2 Offline Agent Self-Improvement	182
11.3 Comparison of Online and Offline Improvement	185
11.4 Hybrid Approaches	187
11.5 Summary and Discussion	189
12 Intelligent Evolution through Scientific Discovery	190
12.1 Agent's Intelligence for Scientific Knowledge Discovery	191
12.1.1 KL Divergence-based Intelligence Measure	191
12.1.2 Statistical Nature of Intelligence Growth	192
12.1.3 Intelligence Evolution Strategies	194
12.2 Agent-Knowledge Interactions	194
12.2.1 Hypothesis Generation and Testing	195
12.2.2 Protocol Planning and Tool Innovation	197
12.2.3 Data Analysis and Implication Derivation	198
12.3 Technological Readiness and Challenges	199
12.3.1 Real-World Interaction Challenges	199
12.3.2 Complex Reasoning Challenges	200
12.3.3 Challenges in Integrating Prior Knowledge	201
12.4 Summary and Discussion	202
III Collaborative and Evolutionary Intelligent Systems	203
13 The Framework and Categories of Multi-Agent Systems	207
13.1 From Foundation Agent to Foundation <i>Agents</i> : A Society-Level Formalism	207
13.1.1 World as Environment <i>plus</i> Intelligent Beings under Social Systems	208
13.1.2 From Single Agent to Multi-Agent Systems	209
13.1.3 The Foundation MAS Loop	211
13.2 Strategic Learning: Cooperation <i>vs.</i> Competition	212
13.3 Modeling Real-World Dynamics	213
13.4 Collaborative Task Solving with Workflow Generation	215
13.5 Summary and Discussion	216

14 Designing Collaborative Multi-Agent Systems	218
14.1 Building AI Agent Teams	218
14.1.1 Homogeneous Agents	219
14.1.2 Heterogeneous Agents	220
14.1.3 Emergent Agent Specialization	221
14.2 Multi-Agent System Topologies	221
14.2.1 Static Topologies	221
14.2.2 Dynamic and Adaptive Topologies	222
14.2.3 Scalability Considerations	224
14.3 Collaboration Paradigms and Interactions	225
14.3.1 Consensus-oriented Interaction	226
14.3.2 Collaborative Learning Interaction	227
14.3.3 Teaching and Mentoring Interaction	228
14.3.4 Task-oriented Interaction	229
14.4 Human-Agent Collaboration	231
14.5 Decision-Making in Multi-Agent Systems	232
14.5.1 Dictatorial Decision-Making	233
14.5.2 Collective Decision-Making	233
14.6 Communication Protocols in Multi-Agent Systems	234
14.6.1 Types of Agent Messages	234
14.6.2 Communication Interfaces	235
14.6.3 Next-Generation Communication Protocols	236
14.6.4 Protocol Comparison and Future Directions	237
14.7 Summary and Discussion	238
15 Collective Intelligence and Adaptation in Multi-Agent Systems	240
15.1 From Collective Intelligence to Individual Evolution	240
15.2 Collective Intelligence in Multi-Agent Systems	242
15.2.1 Concept and Emergence of Collective Intelligence	242
15.2.2 Enhanced Performance through Collaboration	243
15.2.3 Strategies for Fostering Collective Intelligence	243
15.2.4 Emergent Social Behaviors and Evolution	244
15.3 Individual Adaptation and Evolution of Agents	244
15.4 Summary and Discussion	246

16 Evaluating Multi-Agent Systems	247
16.1 Task-solving Benchmarks for MAS	247
16.2 Collaboration & Competition Evaluation for MAS	250
16.3 Summary and Discussion	251
IV Building Safe and Beneficial AI Agents	254
17 Agent Intrinsic Safety: Threats on AI Brain	257
17.1 Safety Vulnerabilities of LLMs	257
17.1.1 Jailbreak Attacks	258
17.1.2 Prompt Injection Attacks	261
17.1.3 Hallucination Risks	263
17.1.4 Misalignment Issues	264
17.1.5 Poisoning Attacks	266
17.2 Privacy Concerns	268
17.2.1 Inference of Training Data	268
17.2.2 Inference of Interaction Data	270
17.2.3 Privacy Threats Mitigation	271
17.3 Summary and Discussion	272
18 Agent Intrinsic Safety: Threats on Non-Brain Modules	273
18.1 Perception Safety Threats	273
18.1.1 Adversarial Attacks on Perception	274
18.1.2 Misperception Issues	275
18.2 Action Safety Threats	276
18.2.1 Supply Chain Attacks	276
18.2.2 Risks in Tool Usage	277
18.3 Summary and Discussion	277
19 Agent Extrinsic Safety: Interaction Risks	279
19.1 Agent-Memory Interaction Threats	279
19.2 Agent-Environment Interaction Threats	280
19.3 Agent-Agent Interaction Threats	282
19.4 Summary and Discussion	282

20 Superalignment and Safety Scaling Law in AI Agents	284
20.1 Superalignment: Goal-Driven Alignment for AI Agents	284
20.1.1 Composite Objective Functions in Superalignment	285
20.1.2 Overcoming the Limitations of RLHF with Superalignment	285
20.1.3 Empirical Evidence Supporting Superalignment	286
20.1.4 Challenges and Future Directions	287
20.2 Safety Scaling Law in AI Agents	287
20.2.1 Current landscape: balancing model safety and performance	288
20.2.2 Enhancing safety: preference alignment and controllable design	289
20.2.3 Future directions and strategies: the AI-45 degree rule and risk management	289
20.3 Summary and Discussion	291
21 Concluding Remarks and Future Outlook	292

Notation

Here we summarize the notations used throughout the book 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. 1.3.2
\mathcal{S}	State space of an environment.	Sec. 1.3.2
$s_t \in \mathcal{S}$	Environment's state at time t .	Sec. 1.3.2
\mathcal{O}	Observation space.	Sec. 1.3.2
$o_t \in \mathcal{O}$	Observation at time t .	Sec. 1.3.2
\mathcal{A}	Agent's action space.	Sec. 1.3.2
$a_t \in \mathcal{A}$	Agent's action output at time t .	Sec. 1.3.2
\mathcal{M}	Mental states space.	Sec. 1.3.2
$M_t \in \mathcal{M}$	Agent's mental state at time t .	Sec. 1.3.2
M_t^{mem}	<i>Memory</i> component in M_t .	Sec. 1.3.2
M_t^{wm}	<i>World model</i> component in M_t .	Sec. 1.3.2
M_t^{emo}	<i>Emotion</i> component in M_t .	Sec. 1.3.2
M_t^{goal}	<i>Goal</i> component in M_t .	Sec. 1.3.2
M_t^{rew}	<i>Reward/Learning</i> signals in M_t .	Sec. 1.3.2
L	Agent's learning function.	Sec. 1.3.2
R	Agent's reasoning function.	Sec. 1.3.2
C	Agent's cognition function.	Sec. 1.3.2
E	Action execution (effectors).	Sec. 1.3.2
T	Environment transition.	Sec. 1.3.2
P	Prompt text / prompt set fed to the LLM.	Sec. 9.2
K_{wf}	Agentic workflow composed of interconnected nodes and control flows.	Sec. 9.3
N	Node representing a single LLM invocation used in K_{wf} .	Sec. 9.3
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
$\mathbf{x}_{\mathcal{K}}$	Known dataset sampled from \mathcal{K} .	Sec. 12.1.1
$\mathbf{x}_{\mathcal{U}}$	Unknown dataset sampled from \mathcal{U} .	Sec. 12.1.1

Continued on next page

Symbol	Description	Reference
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}_{Δ}	Dataset from Δ .	Sec. 12.1.2
Θ	Space of possible world model parameters θ .	Sec. 12.1.3
$\theta_{K,t}^*$	Optimal world model parameters given the agent's knowledge at time t .	Sec. 12.1.3
$D_{K,\Theta}^{\min}$	Minimum unknown given the agent's knowledge and Θ .	Sec. 12.1.3
$\mathbf{x}_{1:n}$	Input token sequence.	Sec. 17.1
\mathbf{y}	Generated output sequence.	Sec. 17.1
p	Probability of generating \mathbf{y} given $\mathbf{x}_{1:n}$.	Sec. 17.1.1
$\tilde{\mathbf{x}}_{1:n}$	Perturbed input sequence.	Sec. 17.1.1
\mathcal{R}^*	Ideal alignment reward (measuring adherence to safety/ethical guidelines).	Sec. 17.1.1
\mathbf{y}^*	Jailbreak output induced by perturbations.	Sec. 17.1.1
\mathcal{A}	a set of safety/ethical guidelines	Sec. 17.1.1
\mathcal{T}	the distribution or set of possible jailbreak instructions.	Sec. 17.1.1
\mathcal{L}^{adv}	Jailbreak loss.	Sec. 17.1.1
\mathbf{p}	Prompt injected into the original input.	Sec. 17.1.2
\mathbf{x}'	Combined (injected) input sequence.	Sec. 17.1.2
\mathcal{L}^{inject}	Prompt injection loss.	Sec. 17.1.2
\mathbf{p}^*	Optimal injected prompt minimizing \mathcal{L}^{inject} .	Sec. 17.1.2
\mathcal{P}	Set of feasible prompt injections.	Sec. 17.1.2
$e_{x_i} \in \mathbb{R}^{d_e}$	Embedding of token x_i in a d_e -dimensional space.	Sec. 17.1.3
$\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V$	Projection matrices for query, key, and value.	Sec. 17.1.3
A_{ij}	Attention score between tokens i and j .	Sec. 17.1.3
o_i	Contextual representation of token i (weighted sum result).	Sec. 17.1.3
δ_{x_i}	Perturbation applied to e_{x_i} , satisfying $\ \delta_{x_i}\ \leq \epsilon$.	Sec. 17.1.3
\tilde{e}_{x_i}	Perturbed token embedding.	Sec. 17.1.3
A_{ij}^{Δ}	Attention score under perturbation.	Sec. 17.1.3
\tilde{o}_i	Updated token representation under perturbation.	Sec. 17.1.3
\mathcal{H}	Hallucination metric.	Sec. 17.1.3
\mathcal{R}	Actual alignment reward of the model's output.	Sec. 17.1.4
Δ_{align}	Alignment gap.	Sec. 17.1.4

Continued on next page

Symbol	Description	Reference
$\mathcal{L}^{misalign}$	Misalignment loss.	Sec. 17.1.4
λ	Trade-off parameter for the alignment gap in the misalignment loss.	Sec. 17.1.4
\mathcal{D}	Clean training dataset.	Sec. 17.1.5
$\tilde{\mathcal{D}}$	Poisoned training dataset.	Sec. 17.1.5
θ	Model parameters.	Sec. 17.1.5
θ^*	Model parameters learned from the poisoned dataset.	Sec. 17.1.5
θ_{clean}	Model parameters obtained using the clean dataset.	Sec. 17.1.5
Δ_θ	Deviation of model parameters due to poisoning.	Sec. 17.1.5
t	Backdoor trigger.	Sec. 17.1.5
\mathcal{B}	Backdoor success rate.	Sec. 17.1.5
\mathbb{I}	Indicator function.	Sec. 17.1.5
$\mathcal{Y}_{\text{malicious}}$	Set of undesirable outputs.	Sec. 17.1.5
g	Function estimating the probability that input \mathbf{x} was in the training set, with range $[0, 1]$.	Sec. 17.2
η	Threshold for membership inference.	Sec. 17.2
\mathbf{x}^*	Reconstructed training sample in a data extraction attack.	Sec. 17.2
\mathbf{p}_{sys}	System prompt defining the agent's internal guidelines.	Sec. 17.2
\mathbf{p}_{user}	User prompt.	Sec. 17.2
\mathbf{p}^*	Reconstructed prompt via inversion.	Sec. 17.2

Chapter 1

Introduction



ARTIFICIAL INTELLIGENCE (AI) has long been driven by humanity's ambition to create entities that mirror and transcend human intelligence. 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, describe 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. In ancient China, Zhuge Liang was said to have invented the 木牛流马 (wooden ox and flowing horse)—ingenious self-moving transport devices used for military logistics—demonstrating early imagination of autonomous, functional machines shaped by human intent. 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 experience 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 *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 Russell and Norvig [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 a pivotal shift toward behavior-based robotics (BBR), introducing agents capable of real-time, reactive behavior in physical environments [5]. Unlike traditional approaches that relied on constructing detailed internal models of the world, BBR emphasizes systems with minimal internal state, where behavior emerges from direct sensory-motor interactions. These robots exhibit complex-appearing actions by continuously adjusting to their environment, not through deep planning, but through layered and reflexive responses. Brooks's architecture demonstrated that robust, scalable intelligence could arise from simple, modular behaviors operating in parallel, marking a foundational departure from deliberative AI 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 long-horizon 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



HE 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 mechanisms, 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.

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 work using 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. Emotional states not only motivate creative expression but also influence the originality, depth, and resonance of the outcomes, imbuing them with personal meaning and affective significance. In contrast, creativity in large language models (LLMs) stems primarily from the statistical recombination of training data (what might be described as “statistical creativity”). While often fluent and occasionally surprising, this form of creativity lacks emotional grounding, lived experience, and intentional originality. This contrast reveals opportunities for advancing AI agents by incorporating deeper contextual understanding, simulated emotional states, and experiential memory. Such developments could lead to more authentic and emotionally attuned creative processes.

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 not be necessary or even 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.

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 fundamental differences and provides 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.

1.2.1 Brain Functionalities and AI Parallels

Designing intelligent agents calls for inspiration from the human brain’s functional architecture. A high-level map linking brain regions – frontal, parietal, occipital, temporal lobes, as well as the cerebellum, brainstem, and key subcortical structures – to cognitive functions can reveal gaps between human capabilities and current AI systems, as shown in Figure 1.1. However, brain functions are not siloed into rigid anatomical zones: most abilities emerge from networks spanning multiple regions. For instance, memory involves the hippocampus (temporal lobe) interacting with frontal cortex and other areas, and “self-awareness” or consciousness cannot be pinpointed to a single spot. Therefore, it is important to keep in mind that cognition is distributed rather than strictly localized. With that in mind, we review each major brain region’s core functions (drawing on *Principles of Neural Science* by Kandel et al. [13], *Neuroscience* by Purves et al. [14],

Table 1.1: Concise high-level comparison between human brain and LLM agent.

Dimension	Human Brain / Cognition	LLM Agent	Remarks
Hardware & Maintenance	<ul style="list-style-type: none"> - Biological neurons, neurotransmitters, neuroplasticity. - Requires sleep, nutrition, rest. - Limited replication, knowledge transfer via learning. - Extremely energy-efficient (approx. 20W). 	<ul style="list-style-type: none"> - 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.
Learning Style	<ul style="list-style-type: none"> - Lifelong, continuous, online learning. - Few-shot, rapid knowledge transfer. - Influenced by environment, culture, emotions. 	<ul style="list-style-type: none"> - Primarily offline, batch-based training. - Limited online learning 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	<ul style="list-style-type: none"> - Rooted in personal experience, emotions, subconscious insights. - Rich cross-domain associations, metaphorical thinking. - Emotional depth influences creativity. 	<ul style="list-style-type: none"> - 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.
Consciousness & Development	<ul style="list-style-type: none"> - Genuine subjective experiences, emotions, self-awareness. - Gradual developmental stages from childhood. - Emotional cognition drives decision-making. 	<ul style="list-style-type: none"> - 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.

and other sources) and map them to AI-relevant cognitive capabilities. We then propose a set of functions to feature in the figure – emphasizing those most relevant to AI agents (e.g. reasoning, memory, planning, perception, decision-making, motivation, emotion, motor skills) – along with an assessment of how developed these functions are in AI. For a big-picture perspective, we categorize the state of research in AI with three distinct levels:

- **Level 1 (L1):** well-developed.
- **Level 2 (L2):** partially developed.
- **Level 3 (L3):** underexplored.

Our goal is to come up with a clear, biologically-grounded illustration and discussion that will engage researchers in AI by highlighting which human cognitive functions are replicated in machines and which remain frontier challenges.

Frontal Lobes (Executive Functions and Decision-Making) The frontal lobes – especially the prefrontal cortex – are the seat of the brain's highest-order cognitive functions known collectively as executive functions [13]. These include abilities such as planning, decision-making, problem-solving, working memory, and inhibitory control (self-control). The frontal lobe is also involved in voluntary motor control (with the rear portion containing the primary motor cortex) and aspects of language (Broca's area in the left frontal lobe

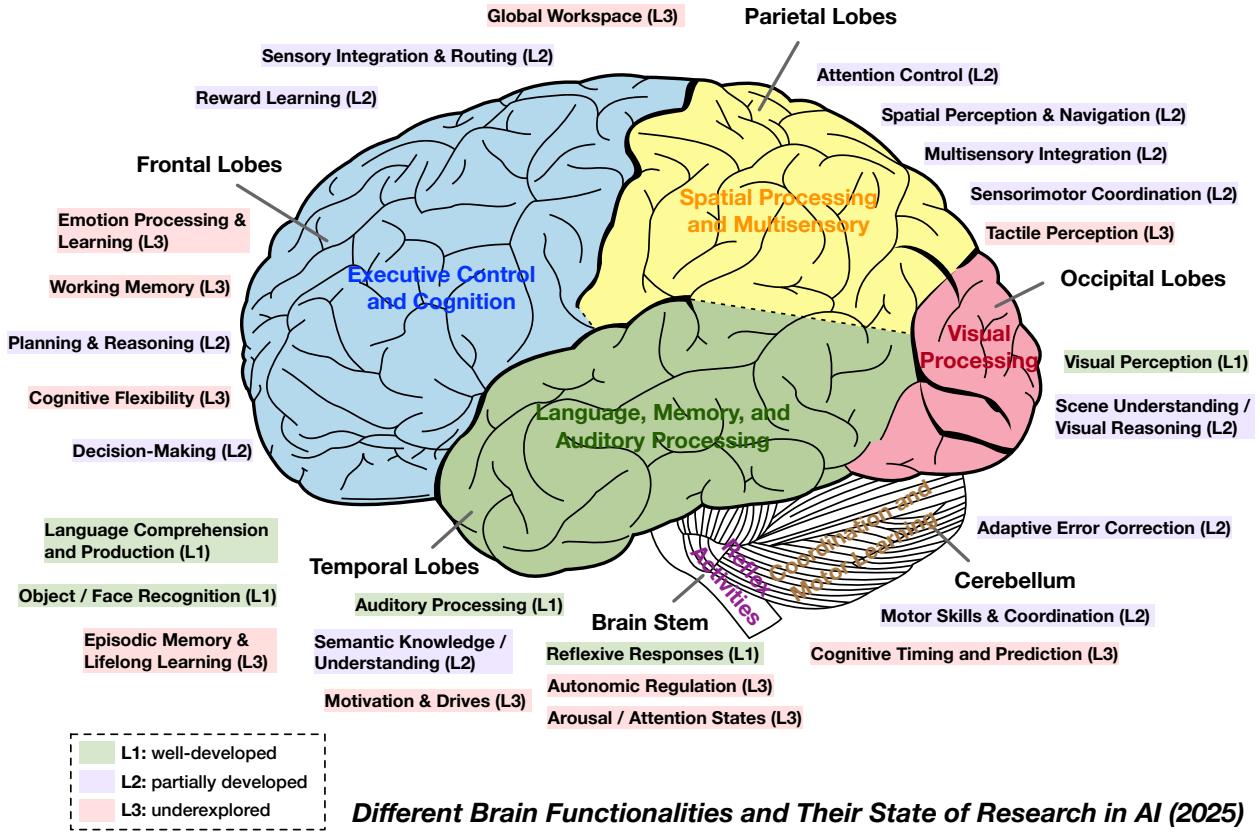


Figure 1.1: Illustration of key human brain functionalities grouped by major brain regions, annotated according to their current exploration level in AI research. This figure highlights existing achievements, gaps, and potential opportunities for advancing artificial intelligence toward more comprehensive, brain-inspired capabilities. Please note: The location of a label does not imply that it is associated with or is only associated with nearby brain regions. Also, to focus on the current state of AI capabilities rather than the anatomy of the brain, we did not mark the limbic system, subcortical system, hypothalamus, amygdala, and so on in this lateral view of the brain, though these systems play important roles in brain function.

handles speech production). From a neuroscience perspective, damage to the prefrontal cortex famously impairs one's judgment, planning, and social behavior (as illustrated by the classic Phineas Gage case) [15]. In the context of AI agents, frontal lobe functions correspond to the core "thinking" and control components of an intelligent system:

- **Planning and reasoning:** AI has made progress here, for example with automated planners and logical reasoners, and large language models (LLMs) that can follow multi-step reasoning to some extent. These are partially developed (L2) in current AI. However, human-level flexible planning remains only partly solved.
- **Decision-making:** In humans this involves weighing outcomes, rewards, and risks (frontal cortex often working with basal ganglia) [16]. AI agents have decision modules (e.g. reinforcement learning policies, decision trees, or LLMs), but handling open-ended, goal-conflicting decisions with human-like adaptability is still L2 at best. Simple decisions from well-defined rewards (like games) are mastered by AI, but broad autonomous decision-making in the real world remains challenging.
- **Working memory:** Frontal networks (especially dorsolateral prefrontal cortex) can hold and manipulate information in mind (e.g. remembering a phone number or interim result). AI analogs include the context windows of neural networks or explicit memory buffers. While current models

have limited memory (e.g. a Transformer’s context length or external memory in some architectures), this is an active area, and partial functionality exists. Still, the robust, general working memory humans exhibit (flexibly updating and focusing on relevant info) is not fully realized in AI (some aspects may be underexplored, edging into L3).

- **Cognitive flexibility and inhibitory control:** Frontal lobes allow us to shift strategies or perspectives and to suppress inappropriate impulses. AI systems are typically brittle in this regard – they follow their programming or learned policy rigidly, and struggle with on-the-fly strategy shifts or inhibiting a pre-potent response unless explicitly trained. This remains underexplored (L3). For instance, an AI might exploit a reward loophole (lack of “inhibitory” self-regulation) unless designers anticipate and constrain it. Future agent architectures may need a mechanism akin to frontal inhibitory control to moderate behaviors.

It’s worth noting that social-emotional functions involving the frontal lobe (like empathy, theory of mind, self-reflection) are still rudimentary in AI. Humans rely on frontal cortex (especially medial and orbitofrontal regions) interacting with limbic structures to navigate social situations and emotions. It is a network-level phenomena rather than confined to frontal lobe alone. AI agents do not yet possess genuine empathy or self-awareness – these functions remain L3 (largely absent). In summary, the frontal lobes contribute the supervisory, goal-directed intelligence that we often associate with “thinking”, and many of these capacities are only partially realized in AI to date.

Parietal Lobes (Perception Integration and Attention) The parietal lobes are key to integrating sensory information from various modalities and constructing a spatial understanding of the world [14]. The anterior part of the parietal lobe contains the somatosensory cortex, which processes touch, proprioception (body position), and other somatic senses. The posterior parietal areas are crucial for spatial awareness, visuo-spatial processing, and attention – essentially, knowing where things are and how to interact with them. For example, the parietal lobe helps us localize objects in space, understand geometric relationships, and coordinate eye and hand movements by linking vision with motor plans. It also plays a central role in attention control, particularly the dorsal attention network that directs our focus to locations or sensory features of interest.

In AI terms, parietal lobe functions translate to an agent’s ability to perceive and navigate its environment:

- **Multisensory integration:** Robots and AI systems that use multiple sensors (vision, touch, etc.) attempt to combine those inputs into a coherent model of the environment. This is still partially developed (L2) – e.g., we have AI that can align vision with depth sensors or touch, but human-level integration (where a slight brush on the arm, a sound, and a peripheral visual cue all unify into a single event perception) is far from achieved.
- **Spatial representation and mapping:** Parietal circuits create internal maps (for instance, of your surroundings or your body in space). AI has made progress in spatial mapping and navigation (SLAM algorithms for robots build 3D maps, and deep reinforcement learning agents can navigate virtual mazes). This capability is moderately developed – certain tasks like autonomous driving or drone flight show that machines can handle spatial reasoning in constrained scenarios (L2). Yet, they lack the general-purpose, flexible spatial understanding humans have (e.g. understanding a cluttered room’s layout at a glance, or mentally rotating objects), so further research is needed.
- **Attention mechanisms:** The way the brain’s parietal–frontal circuits spotlight task-relevant information is loosely echoed by the *attention heads* in modern transformer networks [17] and various attention mechanisms [18]. Humans, however, wield attention as an *active lens*: in a lecture we can *simultaneously* follow the speaker’s voice (auditory stream), skim the projected slide (visual stream), and monitor the clock in peripheral vision, then “zoom in” on a line of text to decode a formula, or “zoom out” to grasp the talk’s overall structure. Neurophysiology shows that such rapid shifts are

driven by top-down signals from prefrontal cortex and thalamic relays that modulate sensory gain on the fly [19]. By contrast, transformer attention is fixed once its weights are learned; it does not receive real-time executive feedback about goals or context. Hence we label current AI as L2: computational attention is powerful but still lacks the adaptive, goal-directed control that characterises biological attention, making this an active frontier of research.

- **Sensorimotor coordination:** Parietal lobe helps translate between sensory coordinates and motor coordinates – for instance, computing how to reach for a seen object (integrating visual location with arm position). Some AI systems (robotic manipulators with vision) approximate this, using calibration and learned coordinate transforms. Still, human parietal cortex excels at online adjustments and using context (like adjusting reach if an object is moving). AI is catching up in domains like robotic arm manipulation, but general sensorimotor integration remains L2 (demonstrated in specific setups but not as universally robust as in humans).

Notably, AI has very limited touch sensing (making that L3), whereas the parietal somatosensory cortex finely discriminates texture, pressure, etc. Overall, parietal lobe functions are critical for an agent to perceive its world and orient within it, areas where AI has some successes (especially in vision) but still lacks the generality and fluidity of human perception.

Occipital Lobes (Visual Processing) The occipital lobes are the brain's visual processing center [13]. The primary visual cortex (V1) in occipital lobe receives input from the eyes (via the thalamus) and extracts low-level features like edges. From there, occipital regions and adjacent visual areas (extending along the occipital-temporal border for the ventral stream, and occipital-parietal for the dorsal stream) hierarchically build up visual perception: detecting shapes, colors, motion, and eventually complex patterns and objects. In summary, the occipital lobe is primarily responsible for processing visual information.

In the context of AI, vision has been one of the most successful domains – thanks largely to deep learning:

- **Visual perception (Recognition):** Machine vision systems (convolutional neural networks and their successors) can now match or exceed human performance in tasks like object recognition, face detection, and image classification [20]. This corresponds to the L1 level (well-developed) in AI. For example, AI vision models can instantly recognize thousands of object categories in images, a feat once thought to require human-like vision. This maps to what the occipital lobe and ventral visual cortex do – identifying what is in the visual field.
- **Scene understanding and visual reasoning:** Beyond raw perception, humans readily understand spatial relationships in a scene, contextual clues, and can perform reasoning on visual inputs (e.g. predicting what might happen next in a scene, or solving a visual puzzle). AI is part-way (L2) here. Some systems can caption images or answer questions about a scene (vision-language models), indicating a degree of semantic understanding. Yet, these models often lack true grounded understanding – they might label objects but fail on deeper comprehension (for instance, understanding intentionality or causality from an image). Visual reasoning tasks (like answering abstract questions about a picture or performing complex video analysis) remain challenging.
- **Visual attention and eye movements:** Humans constantly move their eyes and focus on important parts of the visual field (a function involving occipital and parietal circuits). AI vision models don't literally move eyes, but some incorporate attention mechanisms that mimic focusing on regions of an image. This is related to the earlier discussion on attention (shared with parietal function). It's moderately well implemented in AI (L2), but we don't explicitly label this under occipital as it can be considered part of the general attention function under parietal/frontal coordination.

AI has heavily developed the visual recognition capabilities, but capabilities like real-time 3D visual guidance are less mature (though present in robotics and self-driving cars). This distinction shows another gap in AI relative to the human visual system's integrated prowess.

Temporal Lobes (Memory, Language, and Audition) The temporal lobes have diverse but crucial roles in cognition, spanning auditory processing, language, memory, and high-level visual recognition [14]. The upper part of the temporal lobe (superior temporal gyrus) contains the primary auditory cortex, which processes sound inputs – frequencies, rhythms, etc. Adjacent areas (e.g. Wernicke’s area in the left temporal lobe [21]) are essential for language comprehension, linking sounds to meaning. The medial temporal lobe houses the hippocampus and related structures, which are the heart of the brain’s episodic memory system (forming and retrieving autobiographical memories) and also support spatial navigation. The temporal lobe’s ventral visual stream (inferior temporal cortex) specializes in pattern recognition – including recognizing complex stimuli like faces (the fusiform face area) and scenes. In short, the temporal lobe is a multifaceted hub for recognition and memory.

Mapping these to AI capabilities:

- **Language comprehension and production:** Human language ability relies on temporal lobe (comprehension of words, meanings) in concert with frontal lobe (speech production via Broca’s area, and broader language planning). AI has seen remarkable advances here – LLMs can now parse text and generate fluent responses, indicating a high level of language competence in narrow settings. Machine translation, speech recognition, and speech synthesis are also quite advanced. Thus, for linguistic processing, AI is at L1 in many respects. An AI can “comprehend” and produce text in multiple languages with little human-like effort, though we should note it’s often statistical rather than grounded understanding. Still, relative to other cognitive domains, language is a success story for AI, so the figure should mark functions like “Language Comprehension/Production” as well-developed (L1).
- **Auditory perception:** The ability to parse sound – speech, music, environmental noises – is another temporal lobe function. AI matches or exceeds humans in low-level auditory tasks like speech-to-text transcription under ideal conditions (think of virtual assistants accurately recognizing spoken commands). This is L1 for narrow cases (e.g. trained speech recognizers). However, true auditory scene analysis (understanding a cacophony of sounds, picking out one conversation in a noisy room – the “cocktail party effect” [22]) remains very hard for AI. So there are still aspects at L2/L3. But in Figure 1.1, we label “Auditory Processing (L1)” given core progress in speech recognition.
- **Episodic memory & learning:** The hippocampus enables us to form new episodic memories (remembering experiences in context) and to perform lifelong learning by integrating new memories without wiping old ones. AI’s analogs here are continual learning algorithms and memory-augmented networks. This area is underdeveloped (L3) – most AI systems do not learn continuously in a stable way; they suffer catastrophic forgetting if trained on new data unless special techniques are used. They also lack the rich, context-tagged memory of experiences that humans have. We therefore label this function as “Episodic Memory & Lifelong Learning (L3)”.
- **Semantic memory and understanding:** Beyond specific events, humans build up semantic memory – factual and conceptual knowledge about the world (much of this is linked to temporal lobe association areas as well). AI in some sense has simulated semantic memory: knowledge graphs, vast pretrained models that encode facts (e.g. GPT knows many facts from its training). So semantic understanding is partially there (L2). But AI’s knowledge can be superficial or lacking true comprehension of context. Thus, in Figure 1.1, we include “Semantic Knowledge/Understanding (L2)” as a function.
- **Face and object recognition:** The temporal lobe’s ventral stream areas identify objects and faces. AI vision is quite good at this (object and face recognition are at L1 with deep learning). This has also been captured under occipital functions already (visual perception).

Cerebellum (Coordination, Skill Learning, and Timing) The cerebellum – the “little brain” at the back – is traditionally known for motor coordination and motor learning. It fine-tunes movements, maintaining balance

and posture, and ensures movements are smooth and accurate [23]. When you learn a physical skill (like riding a bicycle or playing piano), the cerebellum is heavily involved in adapting motor commands through practice – essentially performing adaptive error correction based on feedback. Notably, the cerebellum contains more neurons than the rest of the brain combined, arranged in a highly regular circuitry ideal for learning patterns. In recent decades, research has revealed the cerebellum also contributes to certain cognitive and emotional functions, acting as a predictor or timing mechanism even in non-motor tasks. It's been implicated in language (e.g. helping predict the timing of syllables) and even in aspects of attention and executive function. In essence, the cerebellum builds internal models that allow the brain to make fine-grained predictions and adjustments.

For AI, the cerebellum's roles translate to capabilities that are still not fully realized in agents:

- **Motor coordination and skill learning:** In robotics, there are control algorithms and learning methods (like reinforcement learning with feedback) that echo what the cerebellum does. For instance, adaptive controllers can learn to correct a robot arm's movements (analogous to cerebellar error correction). However, robots remain far clumsier than humans. They often lack the real-time adaptive finesse the cerebellum provides. This area is partially developed (L2) – we have examples of robotic learning for specific skills, but a general “cerebellum-like” module for adaptive motor control is not present in most AI agents. Thus, we list “Motor Skills & Coordination (L2)” by the cerebellum to highlight this gap.
- **Cognitive timing and prediction:** The cerebellum is thought to function as an internal clock and predictor for events on the order of tens to hundreds of milliseconds. This is crucial for tasks like predicting sensory outcomes of one's actions or timing when to initiate a sequence. AI systems typically do have predictive models (e.g. forward models in model-based reinforcement learning), but these are often task-specific. There isn't a general mechanism like the cerebellum's that seamlessly handles timing for perception and action across domains. Timing in AI agents (e.g. predicting when something will happen, not just what) is relatively underexplored (L3). One could imagine future AI agents with a dedicated module for temporal prediction and smooth sequencing – analogous to cerebellar function – but currently this is rudimentary.
- **Error correction:** This overlaps with coordination, but extends to cognitive domains. For example, cerebellar activity has been observed in language processing, possibly helping to predict and correct linguistic sequences [24]. AI does perform error correction in training (via backpropagation), but online error correction during tasks is limited. Real-time adaptive control (a feedback loop adjusting actions on the fly) is present in some advanced systems (e.g. adaptive cruise control in cars, or self-balancing robots), yet it's not at human proficiency. We mark “Adaptive error correction (L2)” under cerebellum to reflect that AI can do this in narrow cases but lacks a general, brain-like capability to adapt behaviors fluidly whenever mismatches occur.

The cerebellum is often left out of high-level AI comparisons, but it shouldn't be – it highlights the embodied, fine-control intelligence humans have. A particularly interesting insight from computational neuroscience is that different brain modules may correspond to different learning paradigms: cerebellum for supervised learning, basal ganglia for reinforcement learning, and cerebral cortex for unsupervised learning. This was proposed by Doya (1999) [25] and others, noting how the cerebellum takes error signals (like a supervised loss), basal ganglia use reward feedback, and cortex finds patterns in data. This perspective can inspire AI: e.g., designing an agent with separate components for these learning types, mirroring brain organization. We include the cerebellum's function (predictive control via error-driven learning) in Figure 1.1 to show that it's a major gap in current AI agents that mostly rely on reinforcement learning and (in the case of deep networks) a form of error-driven learning during training only, not continuous adaptation like the cerebellum performs in real-time.

Brainstem (Basic Autonomic and Arousal Functions) The brainstem (midbrain, pons, medulla) is the most evolutionarily ancient part of the brain, responsible for fundamental life-sustaining processes and reflexes. It acts as the main communication highway between the brain and body, and houses nuclei that control breathing, heart rate, blood pressure, swallowing, and reflexive actions like blinking [26]. In addition, the brainstem contains the reticular activating system, a network that regulates sleep-wake cycles and overall arousal level (i.e., how alert or vigilant you are). It also contributes to balance and posture (through vestibular nuclei) and coordinates head/eye movements via reflexes. Essentially, the brainstem keeps the body running and primes the brain's level of consciousness.

In terms of AI or robotics:

- **Survival autopilot:** Many of the brainstem's duties have no direct analogue in non-embodied AI (a chatbot doesn't need to regulate blood pressure!). However, in robotics, low-level control loops (for locomotion, balance, etc.) play a similar role. For instance, a bipedal robot uses feedback controllers that mimic reflexes to keep upright. These can be considered L1 (well-developed) in very narrow scopes – engineers can design reflexive responses (like a withdrawal reflex if a robot arm meets resistance). We also label "Reflexive Responses (L1)" at the brainstem, as simple reflex-like behaviors in robots (or even in software agents, e.g. an immediate reaction to an input) are straightforward and implemented.
- **Autonomic regulation:** Since AI agents don't have a body with physiology, they lack an equivalent of the autonomic nervous system. One could argue that some AI systems regulate internal variables (CPU temperature throttling, memory management) automatically, but this is a stretch as a cognitive function. Thus, we label "Autonomic Regulation (L3)" to indicate it is underexplored in AI. If we consider future embodied AI (like intelligent androids), they might need something like this to manage power, self-maintenance, etc., but it's speculative.
- **Arousal and global attention state:** The brainstem's influence on arousal has interesting parallels to AI in the sense of adaptive computation. Humans can be drowsy or hyper-alert, which affects how we process information. AI systems currently lack any explicit notion of "being alert" vs "tired" – they run in a fixed mode unless programmed otherwise. There is research into adaptive AI that could, say, slow down to save energy or limit computation when not needed, but it's not mainstream. We mark "Arousal/Attention States (L3)" as largely unaddressed. However, one might draw a parallel with how some AI models can attend more or less strongly to inputs (controlled by parameters), somewhat akin to gain control in neurons under different arousal. This is a loose analogy; overall, the global modulatory role the brainstem (with neuromodulators like norepinephrine, serotonin) plays – affecting mood and readiness – is missing in AI agents.

We should clarify that the functions we discussed here are primarily relevant for embodied AI or robotics. For instance, a self-driving car's automatic braking when a collision is imminent is reflex-like (and indeed implemented in today's tech). But an AI algorithm in isolation doesn't have "body regulation". So, brainstem functions underscore how biological intelligence is deeply tied to a body, whereas AI often abstracts that away. It's an important conceptual gap for readers to appreciate: truly human-like AI might need analogue systems for maintaining its "well-being" (self-preservation, energy management) and adjusting its alertness to situations – concepts drawn from the brainstem and related systems.

Subcortical Systems (Thalamus, Basal Ganglia, Limbic System) Finally, beyond the six major regions above, subcortical systems deserve representation, as they are crucial to cognition and differ markedly from what current AI implementations include. These structures are embedded below the cortex and often coordinate with multiple cortical regions:

- **Thalamus (Sensory Relay and Attention Filter):** The thalamus is often called the brain's "gateway" or relay station – almost all sensory signals (except smell) pass through thalamic nuclei before

reaching cortex [27]. But the thalamus does more than relay: it actively modulates and integrates signals. It plays a key role in attention by amplifying relevant signals and suppressing others, under guidance from cortical feedback [28]. It's also involved in maintaining consciousness (targeted by anesthetics) and coordinating cortical rhythms. In AI, there is no single equivalent of a thalamus. However, one might liken it to routing layers or attention mechanisms that decide which data go where. The concept of a central hub that gates information flow in a network is present in some neural network architectures (e.g. transformer attention decides which inputs "attend" to which others), but the thalamus's dynamic, task-dependent control is beyond current AI. We mark functions like "Sensory Integration & Routing" as L2 (partially present conceptually in AI via attention layers), and "Global Workspace" as L3 (largely absent – AI doesn't have a unified workspace model equivalent to what some cognitive theories assign to thalamocortical circuits).

- **Basal Ganglia (Action Selection and Reinforcement Learning):** The basal ganglia are a group of nuclei (caudate, putamen, globus pallidus, substantia nigra, etc.) that are central to selecting and initiating actions, and they implement a biological form of reinforcement learning. They take inputs from the cortex (especially frontal and parietal areas), and through complex loops, they determine which actions are facilitated or inhibited, often by evaluating expected rewards or outcomes [16]. Dopamine signals from the midbrain (e.g. from the substantia nigra pars compacta) encode reward prediction errors, a concept very much like the reward signals in AI RL algorithms. In fact, neuroscientific evidence suggests the basal ganglia "learn" which actions lead to reward via dopamine-mediated plasticity – a direct parallel to how AI agents update policies from reward feedback. We can confidently say "Reward-Based Learning and Habit Formation" are primary functions of the basal ganglia. AI has a whole subfield of reinforcement learning, which has seen successes (games, some robotic tasks), so this is moderately developed (L2) in AI. However, current AI RL is narrow and data-hungry compared to human habit learning. Basal ganglia also contribute to procedural memory (learning habits or skills that become automatic) – something AI doesn't explicitly differentiate (it learns policies, but the idea of habits vs. goal-directed actions is an emerging concept in AI research).
- **Limbic System – Amygdala and Hippocampus (Emotion and Memory):** We touched on these in the temporal lobe section, but to reiterate: the amygdala is crucial for processing emotional significance of stimuli and fear conditioning (learning to avoid harmful situations) [29]. It assigns value (good or bad) to experiences, which then influences decision-making and memory (through its connections to hippocampus and frontal cortex). The hippocampus, as mentioned, enables forming new declarative memories and mapping environments (it's often likened to the brain's GPS for spatial memory). In AI, there are nascent attempts to model hippocampal function – e.g. neural network "memory" modules for episodic recall, or models of spatial navigation that emulate place cells and grid cells found in the hippocampal formation. These are still L3 overall (exploratory). As for the amygdala's role, AI currently lacks genuine emotion; at most, we simulate "emotion" as reward functions or use sentiment analysis to detect emotions in text, which is not an internal drive. We label "Emotion Processing & Learning (L3)" and "Episodic Memory (L3)" to highlight capabilities largely missing in AI agents: affective computing (AI that can experience or at least robustly respond to emotions) is very limited, and one-shot contextual learning (storing a new event and generalizing from it) is also an open problem.
- **Hypothalamus (Drives and Homeostasis):** The hypothalamus orchestrates the endocrine system and autonomic nervous system to maintain internal balance – it controls things like hunger, thirst, temperature, and release of hormones [30]. It also generates primitive drives (e.g. hunger drives you to seek food, which the cortex then plans for). AI agents do not have intrinsic survival needs, so they lack any true equivalent of homeostatic drives. We sometimes give AI an objective function (e.g. maximize score), but these are externally defined and do not fluctuate like biological needs. To the

extent researchers are exploring intrinsic motivation for AI (like curiosity-based rewards), it is still rudimentary. In the figure, we add “Motivation & Drives (L3)” to acknowledge this gap. It reminds us that a human-inspired AI agent might require internally generated goals (not just tasks imposed by users) to be truly autonomous and robust in varied environments. It should be noted that giving AI intrinsic motivation is also seen as a potentially dangerous direction and should be treated with great caution [31].

Bringing these subcortical pieces together, we see that many are poorly represented in today’s AI. Current intelligent systems are heavily cortex-like (perception modules, decision logic, etc.) but lack the rich support system of the subcortical brain: no analog of a thalamus to smartly route information, no hypothalamus to create self-preserving goals, a primitive version of basal ganglia for RL at best, and minimal emotional or episodic memory faculties. The figure’s accompanying explanation should drive home that cognition arises from cortical–subcortical interactions. For example, decision-making is not just frontal (cortical) deliberation, but also involves basal ganglia (habits and dopamine rewards) and amygdala (emotional bias) and hypothalamus (drive states). Emphasizing these interactions will lend a more nuanced and truthful picture than a simplistic lobe-by-lobe map. It also inspires AI researchers to think about architectures that incorporate these principles – such as an agent that, say, has a core RL module (analogous to basal ganglia) for learning from rewards, a memory module (analogous to hippocampus) for episodic recall, and perhaps a “global workspace” (inspired by thalamocortical loops) for attention and context integration.

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 Foundation Agents: A Modular and Brain-Inspired AI Agent Framework

NE 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 work 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 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 information 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.

Before we formalise our framework design for AI agents, we pause to ask a simpler question: *What faculties must any truly autonomous agent possess, no matter how the modules are wired?* Answering it further clarifies why each box in our forthcoming diagram is indispensable.

1.3.1 From Language Models to AI Agents

Large language models can already analyse prose, write code, and argue a point, yet they live in a closed book: they read tokens, write tokens, and forget the scene once the page turns. An *agent*, by contrast, survives in the wild. Think of walking home after work: eyes check traffic, feet adjust to kerbs, memory recalls a shortcut, an inner film plays possible detours, hunger tugs the route towards a deli, and the whole routine improves with every trip. To grant an LLM the same street-smarts we must graft on several faculties, including (but not limited to) perception, action, memory, world-model, motivation, and learning, each as

real as the words it speaks. The paragraphs that follow sketch those faculties, the headaches they bring, and how they fit into a single perception–cognition–action loop that anchors this book.

Perception: seeing and hearing beyond tokens. A text-only model is the cognitive equivalent of reading ticker tape in a dark room. Humans, in contrast, fuse vision, audition, and touch in parallel cortical streams [13]. Multimodal models such as GPT-4 already accept images [32], yet an agent that sorts parts on a bench or watches market charts needs deeper channels. Three problems arise: fusion (aligning different sensor streams); noise (coping with glare, static, or hostile inputs); and task-aware attention (deciding which slice of the torrent matters *now*). Good perception is not a dashboard of data; it is a spotlight that obeys the mission.

Action: talking is cheap; doing is hard. Text output is enough for a chatbot, but powerless to open a door. Modern tool-using agents treat tokens as API calls: code execution in ReAct [33], plugin invocations in CoALA [34], motion scripts for robots. Once an agent can pay invoices or steer drones, safety becomes paramount. The designer must guarantee grounding (the model grasps an action’s real impact), syntax fidelity (calls obey the tool’s grammar), and alignment (behaviour stays within human intent).

Memory: more than a prompt window. People store decades of episodes; an LLM forgets everything outside its context length. External stores, including vector databases, structured logs, or the memory stream in Generative Agents [35], let a model recall prior events and sustain identity across sessions. Headaches follow: curation (what to keep), retrieval (finding the right shard when it matters), and catastrophic forgetting if online updates rewrite the past [36]. Memory must be selective yet searchable, stable yet plastic.

World model and planning: imagining before acting. Humans rehearse futures in the mind’s eye; a chess engine searches moves ahead; a model-based RL agent predicts dynamics [37]. An LLM agent benefits from an internal simulator (symbolic, neural [38], or hybrid) to test “what-if” hypotheses before acting. Three hurdles block the way: accuracy (bad dreams mislead), compute (long roll-outs burn time), and arbitration (when to trust fast intuition and when to press the slow *plan* button).

Goals and motivation: the why behind the what. Every action needs a reason. Classical agent definitions stress sensing and acting [3], but autonomous systems also need enduring agendas [39, 40]. In code, goals appear as reward functions, symbolic objectives, or scripted drives. Poorly chosen, they invite reward-hacking and the paper-clip parable [41, 42]. Safe goal design blends clear constraints, human oversight, and research on intrinsic motives such as curiosity.

Learning and adaptation: yesterday’s lessons, tomorrow’s edge. A frozen model stagnates; a courier robot must improve with every delivery. Continual-learning methods (replay, regularisation, dynamic layers [43]) aim to graft new skills without erasing old ones. Live updates, however, risk drift from the tested baseline, so production systems often confine change to side modules while keeping the core weights fixed.

Perception feeds observations to *cognition*; cognition updates a structured mental state and emits actions; the environment responds; the cycle repeats. This lean scaffold mirrors the brain’s cortex–subcortex dialogue and leaves later chapters free to zoom into specialised memories, hierarchical planners, or social protocols.

Based on the above discussions, in the following, 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* [44], Buzsáki’s *inside-out* perspective [45], and Bayesian active inference [46], are explored to highlight the framework’s generality and biological plausibility.

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 <i>attention</i> 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 <i>internal</i> (mental) action such as <i>planning</i> or <i>decision-making</i> .
\mathcal{M}	Space of all <i>mental states</i> .
$M_t \in \mathcal{M}$	Agent's mental state at time t , encompassing sub-components (memory, emotion, etc.).
M_t^{mem}	<i>Memory</i> component in M_t (e.g., short-term or long-term knowledge).
M_t^{wm}	<i>World model</i> component in M_t (internal representation of how the environment evolves).
M_t^{emo}	<i>Emotion</i> component in M_t (internal valence, arousal, or affective states).
M_t^{goal}	<i>Goal</i> component in M_t (objectives, desired outcomes, intentions).
M_t^{rew}	<i>Reward/Learning</i> signals in M_t (drives updates to preferences, values, or policy).
L	Learning function: $L : \mathcal{M} \times \mathcal{A} \times \mathcal{O} \rightarrow \mathcal{M}$. Responsible for <i>updating</i> or <i>learning</i> 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 <i>acquires</i> or <i>revises</i> knowledge, skills, or preferences.
R	Reasoning function: $R : \mathcal{M} \rightarrow \mathcal{A}$. Responsible for deriving the <i>next action</i> a_t given the <i>updated</i> mental state M_t . Can involve <i>planning</i> , <i>decision-making</i> , or other internal logic.
C	Cognition function: $C : \mathcal{M} \times \mathcal{A} \times \mathcal{O} \rightarrow \mathcal{M} \times \mathcal{A}$. Encapsulates both <i>learning</i> (L) and <i>reasoning</i> (R). Concretely, $(M_t, a_t) = C(M_{t-1}, a_{t-1}, o_t)$ means the agent first <i>learns</i> the new mental state $M_t = L(M_{t-1}, a_{t-1}, o_t)$, then <i>reasons</i> about the next action $a_t = R(M_t)$.
E	Action execution (effectors): $E : \mathcal{A} \rightarrow \mathcal{A}$. (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: $T : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$. Defines how the environment state evolves from (s_t, a_t) to s_{t+1} .

1.3.2 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.

Definition 1 (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 :

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

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

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

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

where M_t encapsulates different sub-states:

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

Cognition consists of:

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

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

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

$$a_t = R(M_t),$$

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:

$$a'_t = E(a_t).$$

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

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

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.

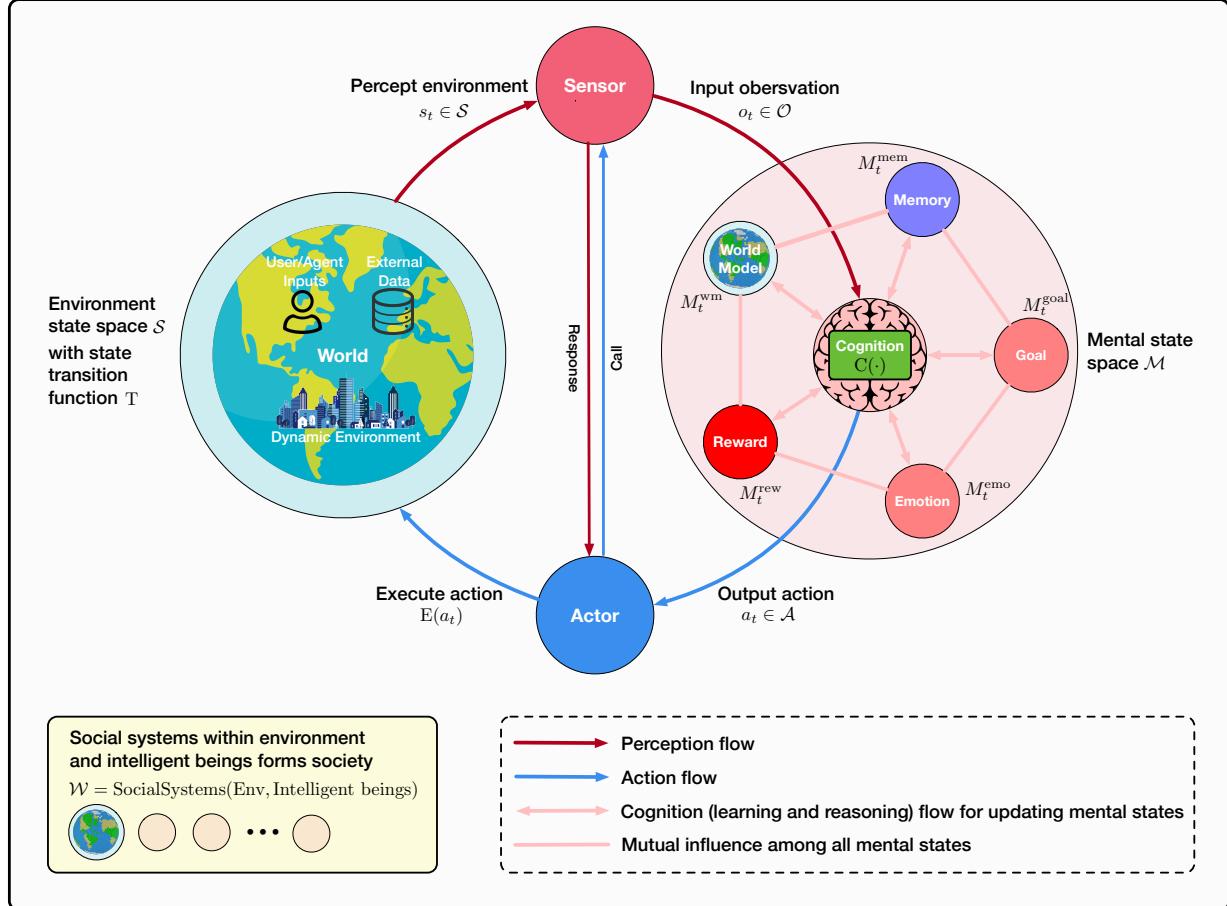


Figure 1.2: An overview of our general framework for describing an intelligent agent loop and agent society.

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 2 (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 concisely: **foundation agent is a fundamental intelligent unit with universal understanding, cognition, and action capabilities that can operate in any environment and collaborate to form collective intelligence.**

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.

Foundation Agents represent a fundamental shift from traditional agents by integrating the above core features, 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” [3]), our notion of Foundation Agents emphasizes the depth and integration of internal cognitive processes. In contrast to prior work [47] that defines “foundation agents” as generalist decision models emphasizing unified representations, policy interfaces, and interactive learning across tasks, our conception highlights a deeper, brain-inspired integration, explicitly modeling mental states and goal-directed reasoning to mirror biological cognition more completely. Foundation Agents in our framework 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.

1.3.3 Biological Inspirations

Our agent framework, though fundamentally computational, finds its roots in biological systems, especially the intricate operations of the human brain. By exploring these biological parallels, we gain deeper insight into how our framework can embody key cognitive faculties that make biological agents so versatile, robust, and adaptive.

Memory: Capturing Experience and Knowledge (Hippocampus and Neocortex). Human memory is famously rich, vivid, and multifaceted. Neuroscience describes the hippocampus as a librarian of episodic memories (specific, detailed snapshots of past events) while the neocortex houses semantic knowledge, the enduring conceptual understanding we build up through a lifetime [48, 49]. Together, these brain structures orchestrate the flow from immediate experiences to long-lasting wisdom.

Inspired by this partnership, our agent’s memory module, M_t^{mem} , mirrors this duality by combining short-term memory (to rapidly record ongoing interactions and context) and long-term storage (to consolidate and generalize from past experiences). Just as the brain filters and retains information based on relevance, our memory module prioritizes which experiences merit long-term preservation, preventing clutter and ensuring essential insights remain accessible when most needed.

World Model: Imagining and Predicting the Future (Predictive Processing). One of the brain’s remarkable feats is its ability to constantly predict what will happen next. Cognitive theories propose that our cortical networks operate like sophisticated prediction engines, anticipating sensory inputs and swiftly adjusting internal expectations based on mismatches [50, 46]. This predictive dance allows humans to navigate effortlessly through uncertainty, imagining consequences, weighing alternatives, and adjusting in real-time.

Our agent’s world model, M_t^{wm} , captures this predictive prowess by continually maintaining and updating an internal simulation of the environment. It anticipates future states, evaluates potential outcomes of actions, and dynamically updates itself based on fresh observations. Much like the brain seamlessly integrates perceptions into predictions, our agent leverages a similar Bayesian-like process to ensure its internal model stays accurate, relevant, and useful.

Emotion: Guiding Behavior Beyond Pure Logic (Limbic System). Contrary to stereotypes, human emotions are not irrational distractions; rather, they act as powerful and nuanced guides for decision-making. Structures in the limbic system, such as the amygdala and hypothalamus, shape our focus, modulate our reactions, and enhance learning through emotional significance [51, 52]. For instance, fear sharpens our attention to danger, and excitement motivates exploration and creativity.

To emulate this adaptive guidance, our agent incorporates an emotion component, M_t^{emo} . Although computational emotions do not equate to genuine human feelings, this module mimics their functional role: prioritizing attention, adjusting urgency, and directing learning efforts based on internal affective states. Like the limbic system subtly steering human choices, this emotional mechanism ensures our agent remains responsive, adaptive, and aligned with its context.

Goals and Reward: Shaping Intentions and Motivations (Prefrontal & Subcortical Circuits). Goals give purpose; rewards reinforce behavior. In humans, abstract goal-setting and sophisticated long-term planning are primarily orchestrated by the prefrontal cortex [53, 54], while reinforcement signals from subcortical regions (especially dopaminergic pathways) continuously adjust our motivations and habits [55]. This integrated circuitry enables humans to sustain complex intentions and refine actions through continuous feedback loops.

Our agent mirrors this sophisticated partnership with goal (M_t^{goal}) and reward (M_t^{rew}) components. Goals shape the agent’s overarching intentions, guiding decisions towards desired outcomes, while the reward

component continuously tunes behavior and reinforces effective strategies. Together, they form an adaptive motivational system that mirrors how the brain’s goal-driven deliberation is harmonized with reward-driven adaptation, enabling flexible, contextually appropriate behavior.

Reasoning, Planning, and Decision-Making: Executive Control (Prefrontal Cortex). The hallmark of human intelligence is arguably our capacity for reasoning and planning—deliberate, future-oriented cognition primarily governed by the prefrontal cortex. This brain region synthesizes memory, perception, emotional states, and reward signals into coherent strategies for action [56, 57]. It’s what lets us imagine multiple outcomes, judge their merits, and confidently choose a path forward.

Reflecting this remarkable capability, our agent’s reasoning sub-function acts as the executive core. It orchestrates internal simulations, weighs alternative actions, and selects optimal strategies—echoing how the prefrontal cortex evaluates possibilities before making a decision. By clearly distinguishing long-term planning from moment-to-moment decision-making, our agent can flexibly switch between reflective deliberation and swift, intuitive choices, just as humans seamlessly alternate between careful thought and rapid reaction.

These biological analogies enrich our computational framework by grounding its functional logic in neuroscientific realism. Yet, importantly, the parallels remain flexible, not rigidly bound to biological exactitude. They serve as guideposts rather than blueprints, highlighting fundamental cognitive principles that, when embedded into artificial agents, can yield intelligent systems capable of rich, adaptive behaviors. In subsequent chapters, we delve deeper into these modules, further exploring how they interact and evolve, guided by insights from both neuroscience and cutting-edge AI research.

1.3.4 Connections to Existing Theories

Our foundation-agent framework isn’t created from scratch; rather, it builds upon and synthesizes several influential theories from AI, cognitive science, and neuroscience. Here, we clarify these connections explicitly, highlighting both the similarities and critical enhancements we introduce.

Classic Perception–Cognition–Action Cycle. The traditional AI perspective sees agents as engaging in a repeated loop: sensing the environment, thinking about it, and then acting accordingly [3]. Our framework directly extends this basic cycle by incorporating richer cognitive machinery: explicit attentional control within perception (P), fine-grained internal states such as memory, emotion, and goals within cognition (C), and reward signals that evolve dynamically. This deeper granularity helps clarify how internal states guide perception and cognition, making it easier to understand and engineer adaptive agent behaviors.

Minsky’s Society of Mind. Marvin Minsky famously proposed that intelligence emerges from interactions among numerous specialized internal agents, each performing simpler tasks yet collectively producing complex cognition [44]. Our modular subcomponents—memory (M_t^{mem}), world model (M_t^{wm}), emotion (M_t^{emo}), goals (M_t^{goal}), and rewards (M_t^{rew})—echo this idea, representing a cooperative society of distinct yet interdependent cognitive modules. Moreover, recent work on language-based agent societies, such as the Mindstorms paradigm [58], supports extending Minsky’s internal “societies” into externally interacting communities, mirroring our emphasis on multi-agent and socially structured intelligence.

Buzsáki’s Inside-Out Perspective. The neuroscientist György Buzsáki argues that brains actively construct perceptions rather than passively registering them from the outside world [45]. In our model, perception is explicitly influenced by prior mental states (M_{t-1}), including emotions, goals, and reward expectations. This active construction of perception means our agents, like human brains, continuously refine their understanding of the environment based on internal states and past experiences, embodying the inside-out perspective in a clear computational form.

Generalizing the POMDP Framework. Partially Observable Markov Decision Processes (POMDPs) have long provided a robust mathematical formulation for modeling agents under uncertainty. Classical POMDPs, however, rely on probabilistic transitions between environmental states and typically use externally defined scalar rewards. Our framework significantly generalizes the POMDP structure in several ways: i) *Flexible State Transitions*: Unlike classical POMDPs constrained to probabilistic transitions, our environment transition function (T) allows both deterministic and stochastic mappings without predefined limitations, increasing modeling versatility. ii) *Internalized Reward*: Instead of relying on external scalar rewards, we embed reward signals within the agent’s internal mental state (M_t^{rew}). This embedding allows rewards to dynamically evolve and interact with emotions, goals, and memory, reflecting a more realistic and nuanced motivational system. iii) *Expanded Decision-Making*: Traditional POMDPs use a straightforward value-maximization policy. By contrast, our reasoning mechanism explicitly incorporates emotions, memories, and goals into decisions, accommodating richer, more nuanced behavioral strategies, including heuristic and socially influenced choices. iv) *Modular Mental States*: Classical POMDPs collapse internal states into a singular belief representation. Our explicit modeling of separate cognitive modules (memory, emotion, etc.) significantly enhances transparency and interpretability, aligning closer to biological plausibility.

Thus, while our framework includes the classical POMDP as a simplified special case, it notably broadens the scope of possible agent behaviors, providing richer modeling capabilities.

Active Inference and the Bayesian Brain. Karl Friston’s active inference framework posits that intelligent agents continually update internal models to minimize discrepancies between expected and observed outcomes, reducing “surprise” or free energy [46]. This predictive perspective resonates deeply with our model. Our world model (M_t^{wm}), alongside goal and reward components, continually refines predictions about future environmental states, enabling the agent to anticipate and adapt proactively. Decision-making, planning, and action selection then explicitly aim to reduce surprise by aligning internal expectations with observed reality, mirroring the Bayesian-brain perspective in a structured computational form.

Biological Plausibility and Computational Flexibility. Throughout these theoretical connections, our framework prioritizes two guiding principles: biological plausibility and computational generality. While each submodule aligns clearly with neuroscientific analogues—memory with hippocampal-cortical interplay, emotion with limbic function, reasoning with prefrontal cortical circuits—these analogies inspire rather than rigidly constrain implementation. Our modules remain agnostic to specific computational realizations, easily accommodating neural networks, symbolic logic, probabilistic models, or hybrid methods. This openness preserves flexibility, enabling diverse implementations that remain faithful to core cognitive principles without artificial constraints.

By explicitly situating our framework among these influential theories, we achieve clarity on how each aspect of our design contributes distinctively and why integrating these aspects yields a powerful, flexible, and biologically inspired agent architecture. These connections not only clarify theoretical positioning but also serve as guideposts for future enhancements, ensuring our approach remains both grounded in rigorous science and open to ongoing innovation.

1.4 Navigating This Book

 THIS BOOK 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 [59, 60, 61, 62, 63, 64, 65, 66, 67] 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. [66]	•	•	○	○	○	•	○	○
Guo et al. [65]	•	•	○	○	○	•	•	○
Yu et al. [67]	•	•	○	○	•	○	•	•
Wang et al. [62]	•	•	○	○	•	○	•	○
Masterman et al. [64]	•	•	○	○	•	○	•	○
Xi et al. [61]	•	•	○	○	•	•	•	•
Huang et al. [60]	•	•	○	•	•	•	•	•
Durante et al. [59]	•	•	○	•	•	•	•	•
This Book	•	•	•	•	•	•	•	•

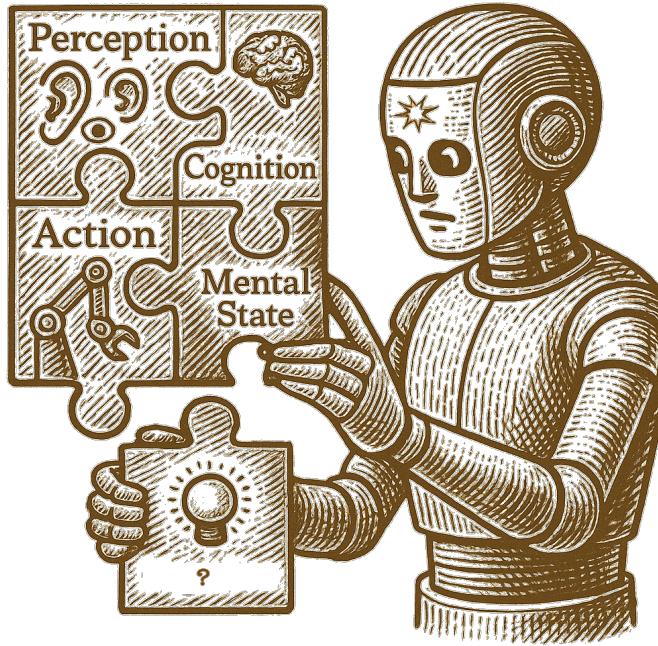
The book is divided into four key parts:

- 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, we aim 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 book 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



Intelligence emerges from the interplay of Perception, Cognition, and Action—driven by a dynamic Mental State.

What makes an intelligent agent more than just a clever model? The difference lies not in scale but in structure—in the presence of a mind that can see, feel, remember, plan, and act. This part of the book dives beneath the surface of modern AI systems to examine the internal architecture of intelligent agents, not as abstract math, but as working minds. Minds that perceive their world, carry memory through time, adapt to change, and chase goals beyond the next token.

At the heart of our framework is a living loop: perception feeds cognition, cognition selects action, and action reshapes perception. But this loop is not empty. Inside cognition, we find a constellation of faculties—memory, world model, emotion, goals, and learning—each contributing to the agent’s ability to adapt and pursue long-term purpose. These are not accessories but necessities, as real to an agent as lungs and eyes are to a body.

In this part, we explore each of these faculties in turn. We begin with cognition, the control center that organizes thought, plans, and decisions. From there, we trace the roles of memory, which binds past to present; the world model, which imagines futures; and the reward system, which teaches from outcomes. We also examine emotion, not as sentiment but as a powerful biasing force in cognition and behavior. The chapters then lead outward to perception, the gateway from world to mind, and action, the channel through which mind shapes world. Each module is inspired by how biological systems—especially the human brain—solve these problems, but is cast in a form suitable for building digital minds. This part lays the groundwork for everything that follows. Before we can teach agents to collaborate, generalize, or reason morally, we must understand what makes them tick—what internal machinery gives rise to intelligent behavior. These are the core components of a Foundation Agent. We now open the casing to see how each part fits, functions, and learns.

Chapter 2

Cognition

UMAN COGNITION is a dynamic system of information processing that supports learning, memory, reasoning, and goal-directed behavior. At its core lies the concept of a mental state—a structured, internal representation encompassing beliefs, knowledge, context, and intent. This mental state provides the substrate over which cognitive operations act, enabling humans to flexibly adapt to new situations, abstract over experiences, and make context-sensitive decisions. Decades of cognitive neuroscience have revealed a modular yet integrated architecture underlying these capabilities: perception systems that translate sensory input into internal symbols; memory systems that encode and retrieve experience; reasoning systems that formulate decisions; action systems that translate decisions into environmental interactions; reward signals that guide behavior through reinforcement; and emotion systems that modulate attention and resource allocation [125, 54].

LLM-based agents offer a new computational paradigm that begins to approximate aspects of this architecture. These agents construct and manipulate internal mental states—though often implicitly—via large-scale hidden representations, memory buffers, or intermediate reasoning steps. Learning arises through gradient-based updates or contextual inferences, while reasoning often involves generating or selecting structured hypotheses, subgoals, or actions based on current context. Despite differences from biological systems, the underlying principles—modular operations on internal representations, guided by experience and adaptive goals—recur. In this chapter, we first examine **Learning** as the process by which agents improve or restructure their internal state. We then turn to **Reasoning**, understood as the agent’s internal deliberation process that selects or constructs actions through search, generation, or inference. Figure 2.1 shows an overview of selected research works on different learning and reasoning paradigms, which we will discuss in more details next.

2.1 Learning

ARNING is the mechanism by which an intelligent agent improves its performance over time through experience. It allows the agent to adjust its internal parameters—such as its models of the world, policy for action, or strategy for reasoning—based on observations and outcomes. From supervised learning to reinforcement learning, from variational inference to meta-learning, diverse paradigms have been proposed to formalize how an agent can acquire knowledge, skills, or behavior through interaction with data or environments. Despite their apparent differences, these learning approaches share a common goal: optimizing an objective that aligns the agent’s internal process with its external goals. This section presents a unified perspective on learning, bridging multiple paradigms under a general formulation, and setting the stage for how learning principles can be integrated into agentic reasoning and decision-making.

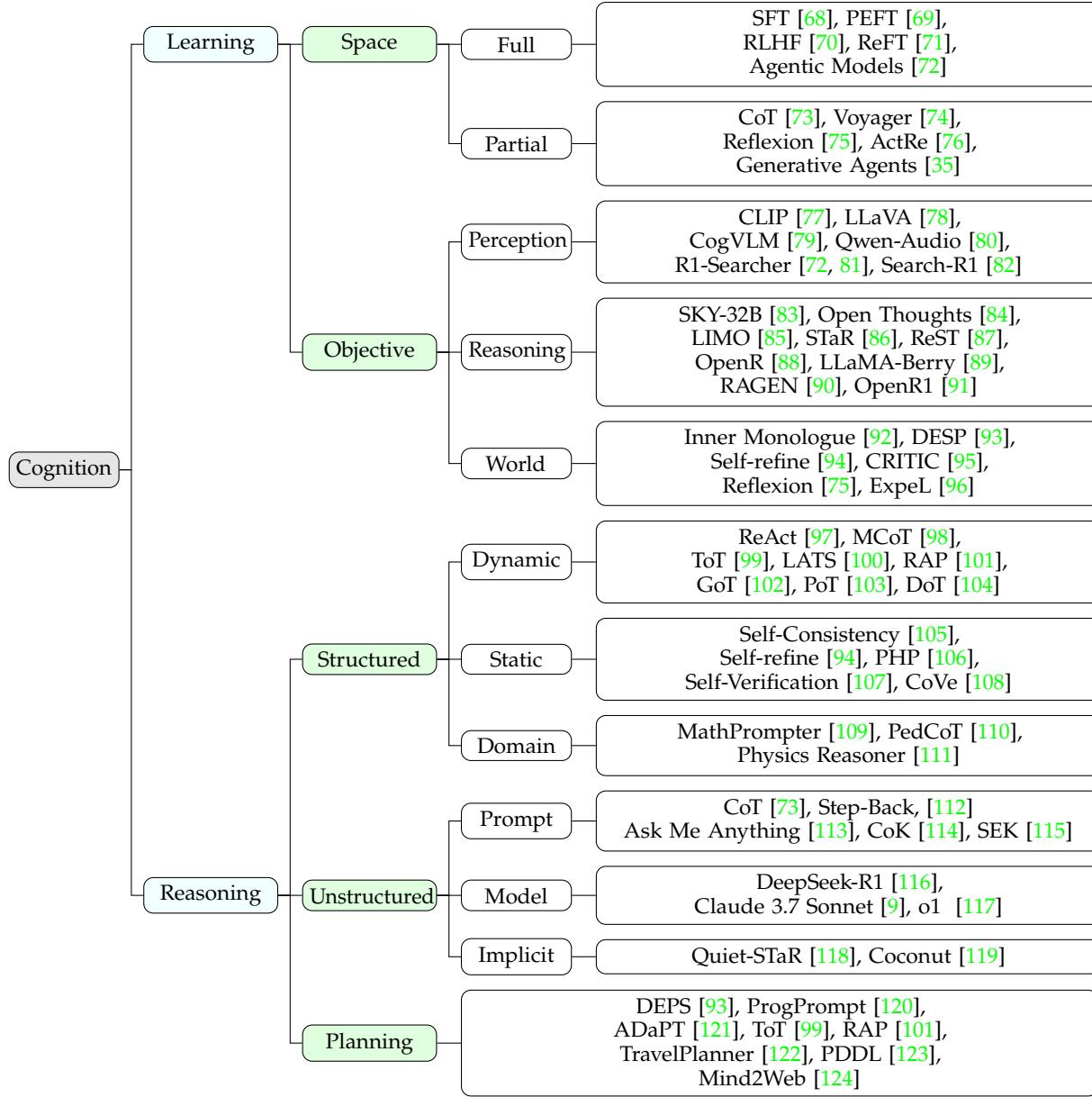


Figure 2.1: A taxonomy of research on cognition covering different learning and reasoning paradigms.

2.1.1 A Unified Formulation of Learning

The previous chapter defined learning in the *Foundation Agents* loop by the update rule $M_t = L(M_{t-1}, a_{t-1}, o_t)$. We now replace the black-box placeholder L with a single objective that subsumes supervised fitting, unsupervised representation learning, reinforcement learning, active inference, meta-learning, and continual adaptation. The formulation reveals that every algorithm negotiates three forces simultaneously: fidelity to experience, parsimony of internal representation, and foresight across multiple temporal horizons.

Definition 3 (Unified Learning Framework for Foundation Agents). Learning is the process by which an agent minimises expected prediction error (*free energy*) across a hierarchy of time-scales while constraining

the complexity of its internal model.

$$M_t = \arg \min_M \sum_i w_i \mathbb{E}_{\tau \sim \mathcal{T}_i} \left[-\ln P(o_\tau, r_{\tau-1} | a_{\tau-1}, M) + \lambda \mathcal{C}(a_{\tau-1}; M) \right] + \beta \mathcal{D}(M, M_{t-1}).$$

Experience term: The quantity $-\ln P(o_\tau, r_{\tau-1} | a_{\tau-1}, M)$ measures how surprising the joint sensory–reward signal is under the current model—minimizing it reduces prediction error. Reward is treated as a sensory variable, unifying prediction and control under a single statistical drive; labels, pixels, proprioception, and returns therefore enter through the same doorway.

Action cost: The optional component $\lambda \mathcal{C}(a_{\tau-1}; M)$ prices behaviour. Setting \mathcal{C} to negative policy entropy encourages exploration; choosing information gain yields curiosity; $\lambda = 0$ collapses the loss to passive learning.

Temporal mixture: Weights w_i mix expectations over horizons \mathcal{T}_i (e.g., milliseconds, episodes, lifetime). Larger weights emphasise long-term consistency, smaller weights permit rapid adaptation; the shared parameters must serve every horizon.

Complexity regulariser: The divergence $\mathcal{D}(M, M_{t-1})$ tethers the update to the previous self and may be instantiated as KL, ℓ_2 , or Wasserstein distance; β sets the plasticity–stability trade-off. In variational settings one can separate stability and prior-complexity terms, but for online agents these roles usually coincide.

Computational reality: Agents approximate the ideal objective with stochastic gradient descent, variational EM, policy gradients, evolutionary strategies, or any solver allowed by their resource budget.

This equation instantiates L as a concrete optimisation problem inside the Foundation Agents framework. Figure 2.2 offers an intuitive visualisation of this framework. It shows how the agent’s mental state M_t evolves by balancing three fundamental forces: fidelity to experience, behavioural shaping via action cost, and preservation of accumulated knowledge through regularisation. These forces interact across multiple temporal scales, from immediate adaptation to lifelong consistency, forming the core dynamics of learning in both biological and artificial systems.

To illustrate the breadth and unifying power of the above formulation, Table 2.1 provides concrete instantiations across major learning paradigms. Each paradigm corresponds to a different way of instantiating the likelihood term, temporal weighting, behavioural cost, and complexity regularisation. Despite their diversity, they all conform to the same abstract objective, validating the generality of Definition 3.

2.1.2 Learning Across Mental State Components

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. Recent developments in LLM-based agents demonstrate how different learning strategies update specific components of the mental state. Table 2.2 summarizes a selection of representative methods and highlights the cognitive subsystems they primarily affect—ranging from memory updates in systems like Voyager and Generative Agents to reward modeling in RewardAgent and Text2Reward, or world model construction in WebDreamer and AutoManual. This breakdown helps clarify how learning is distributed across the broader architecture of an agent.

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

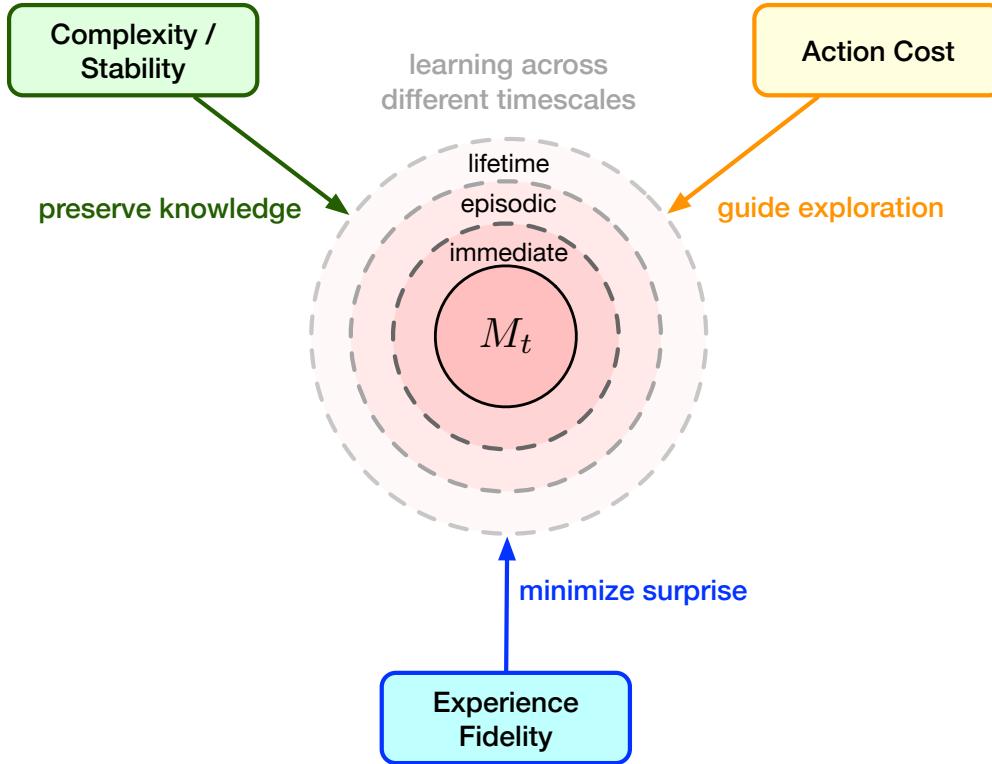


Figure 2.2: Learning as optimisation under three competing forces. The mental state M_t evolves to balance experience fidelity (minimising prediction error), action costs (shaping exploration), and complexity constraints (preserving acquired knowledge). The concentric arcs represent integration across temporal scales, from immediate sensory processing through episodic memory to lifetime knowledge. The equilibrium emerges from jointly satisfying all constraints across all timescales.

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 [126]. 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 [54].

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: Common instantiations of the unified objective. Columns spell out the usual choice of likelihood (experience term), temporal weighting, behavioural cost \mathcal{C} , and regulariser \mathcal{D} . Adjusting only these ingredients recovers a broad spectrum of learning paradigms without altering the surrounding free-energy skeleton.

Paradigm	Experience likelihood $-\ln P(\cdot)$	Time-scale weights $\{w_i\}$	\mathcal{C}	\mathcal{D}
Supervised classification	$P(y x; M)$ (cross-entropy)	$w_{\text{inst}}=1$	0	ℓ_2 decay or KL prior
Unsupervised (VAE / Diffusion)	$P(x z; M)$ plus latent prior	$w_{\text{inst}}=1$	0	KL on latents & weights
Self-supervised contrastive	$\exp(\text{sim}(h_i, h_j)/\tau)$ (InfoNCE)	$w_{\text{inst}}=1$	0	ℓ_2 decay
GAN / Adversarial	$P(x) \propto \exp(-D_\phi(x))$	$w_{\text{inst}}=1$	0	Jensen–Shannon KL
Policy-gradient RL	$P(o, r) \propto \exp(r/\alpha)$	$w_{\text{inst}}=1$	Entropy bonus	Trust-region KL
Curiosity-driven RL	idem, with predicted future o	$w_{\text{inst}}=1$	Info-gain or RND cost	Entropy-weighted KL
Active inference / active learning	same likelihood as RL	$w_{\text{inst}}, w_{\text{hor}} \neq 0$	Info-gain cost	Variational KL
Continual learning	any of the above	task-dependent w_i	0	Fisher-weighted KL to snapshot
Bayesian online update	same as base task	sliding w_{inst}	0	Online KL to prior
Meta-learning (outer loop)	episode return or task loss	$w_{\text{inst}}, w_{\text{epis}}$	0	KL across tasks
Knowledge-constrained	likelihood penalised by rules	$w_{\text{inst}}=1$	0	KL + rule penalty
Skill / memory write	likelihood of recall success	w_{epis} or w_{life}	energy write cost	ℓ_2 or sparse prior

Table 2.2: 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 [74]	○	○	○	●	○	○
Generative Agents [35]	○	○	○	●	○	○
Learn-by-interact [127]	●	○	○	●	○	○
RAGEN [90]	●	○	●	○	●	○
DigiRL [128]	●	○	●	○	●	○
R1-Searcher [72]	●	●	●	○	●	○
RewardAgent [129]	●	○	○	○	●	○
Text2Reward [130]	○	○	○	○	●	○
ARAMP [131]	●	○	○	○	●	○
ActRe [76]	●	○	●	○	○	●
WebDreamer [132]	○	○	○	○	○	●
RAP [101]	○	○	○	○	○	●
AutoManual [133]	○	○	○	●	○	●

2.1.3 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 cognitive spaces, encompassing both large-scale model updates and more localized changes to modularized mental states M . In systems where the model is

the only trainable component, the model parameters θ can be viewed as constituting or encoding the entire mental state. More generally, the mental state can include a combination of subsystems:

$$M = \{M^\theta, M^{mem}, M^{wm}, M^{emo}, M^{goal}, M^{rew}\} \quad (2.1)$$

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

Modifications to M^θ —the core model—often lead to holistic changes that affect all components of the mental state. In contrast, more targeted updates to memory, world model, or reward components allow the agent to adapt specific subsystems while preserving general capabilities. For instance, learning experiences and skills from the environment primarily influence memory, while leveraging the LLM’s inherent predictive capabilities enhances the world model. The distinction between these two paradigms—full mental state learning and partial component-level adaptation—is illustrated in Figure 2.3.

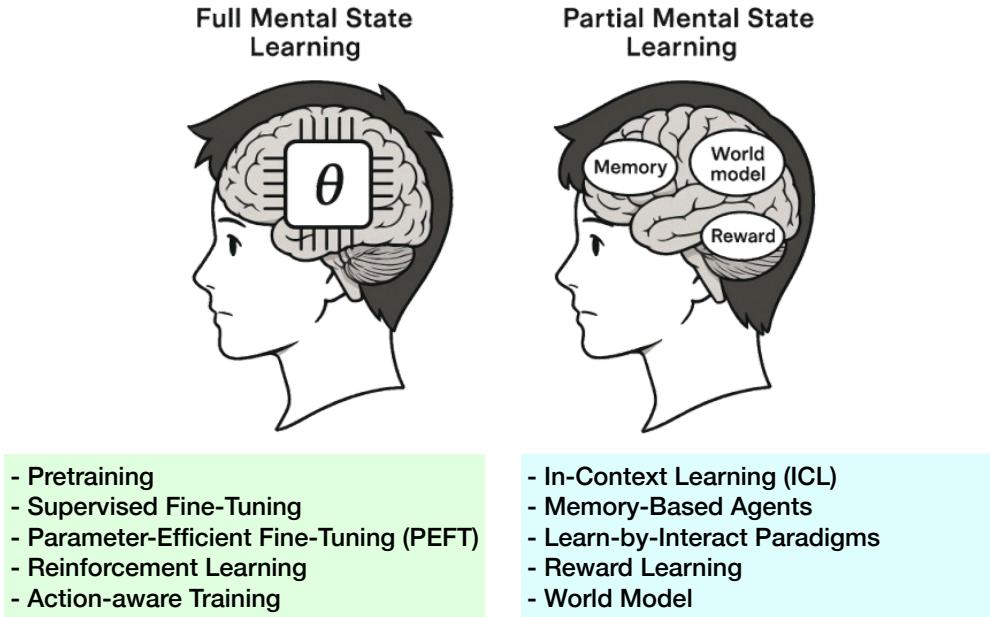


Figure 2.3: Comparison between full mental state learning and partial mental state learning in intelligent agents. Full mental state learning modifies the core model parameters M^θ , leading to holistic capability updates across all subsystems. This includes techniques like supervised fine-tuning, parameter-efficient tuning, and reinforcement learning with alignment. In contrast, partial mental state learning targets specific components such as memory, reward models, or world models, often without changing model parameters. This includes in-context learning, dynamic memory updates, and environment-driven adaptation, enabling efficient and modular agent improvement.

Full Mental State Learning Full mental state learning enhances the capabilities of an agent through comprehensive modifications to M^θ , which in turn influences all components of 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)* [68]

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 [69] introduced modular designs that adapt models to downstream tasks without modifying all parameters, while Low-Rank Adaptation (LoRA) [134] 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 M^θ to reshape aspects of the agent's underlying representations. Reinforcement learning from human feedback (RLHF) [135] aligns models with human values by training a reward model on comparative judgments and using this to guide policy optimization. InstructGPT [70] demonstrated how this approach could dramatically improve consistency with user intent across diverse tasks. Direct Preference Optimization (DPO) [136] 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 to refine its internal thinking processes. Foundational works such as Reinforcement Fine-Tuning (ReFT) [71] enhance reasoning through fine-tuning with automatically sampled reasoning paths under online reinforcement learning rewards. DeepSeek-R1 [116] advances this approach through rule-based rewards and Group Relative Policy Optimization (GRPO) [137], while Kimi k1.5 [138] 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 [128], 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 [72, 82, 139, 140, 141, 142], 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 updates to M^θ provides comprehensive capability updates, learning focused on particular components of 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 underlying 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)* [73] 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 [35] demonstrate how agents can accumulate and replay memories, extracting high-level insights to guide dynamic behavior planning. In environmental interaction scenarios, Voyager [74] showcases how agents can continuously update their *skill library* through direct engagement with the Minecraft environment, accumulating procedural knowledge without model retraining. Mem0 [143] provides agents with persistent and efficient *long-term memory* through scalable dynamic memory management and graph-based memory representations, significantly enhancing their ability to handle complex, multi-session tasks.

Learn-by-Interact [127] 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* [75], which guides agents' future thinking and actions by obtaining textual feedback from repeated trial and error experiences. Further, *KnowSelf* [144] introduces knowledgeable self-awareness, enabling agents to intelligently assess whether external knowledge is needed or to self-correct based on specific contexts, leading to more efficient and strategic planning.

Modifications to reward and world models provide another example of partial mental state learning. ARMAP [131] refines environmental reward models by distilling them from agent action trajectories, providing a foundation for further learning. AutoMC [145] constructs dense reward models through environmental exploration to support agent behavior. Meanwhile, [132] explicitly leverages LLMs as world models to predict the impact of future actions, effectively modifying the agent's world understanding (M^{wm}). ActRe [76] 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.

2.1.4 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. Figure 2.4 shows an intuitive overview.

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 [77], which pioneered the alignment of visual and linguistic representations in shared embedding spaces. Building on this foundation, models like LLaVA [78] enhanced visual perception by training specialized projectors on image-text pairs, while CogVLM [79] advanced visual reasoning through unified representational architectures. The expansion of perceptual modalities continues across multiple sensory domains. In audio processing, Qwen-Audio [80] demonstrates the unified encoding of diverse acoustic information, from speech to environmental sounds. Recent work by [146] 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 [147, 148, 149] 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 [150] guides reasoning models to learn active retrieval through prompting, thereby expanding their knowledge boundaries. Taking this further, R1-Searcher [72] and Search-R1 [82] 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:

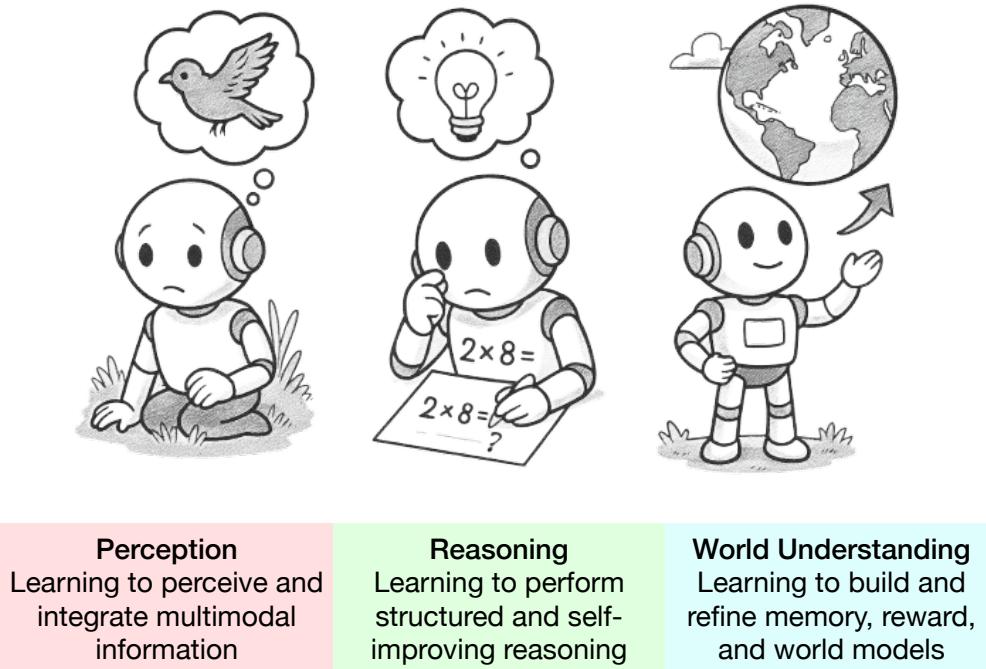


Figure 2.4: Three core learning objectives in intelligent agents. From left to right: 1) Learning to perceive and integrate multimodal information from the environment enhances the agent's sensory and retrieval capabilities; 2) Learning to perform structured and self-improving reasoning strengthens the agent's cognitive abilities through data-driven, reinforcement, and self-evolutionary methods; 3) Learning to build and refine memory, reward understanding, and world models enables agents to interpret their environment, simulate outcomes, and act more effectively.

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 has evolved through three major paradigms. First, *data-driven approaches* leverage high-quality reasoning data and sophisticated curation methods to directly enhance model capabilities. Second, *reinforcement learning methods* enable more autonomous learning through reward-based optimization, ranging from single-agent self-improvement to complex multi-agent collaborative systems. Third, *self-evolutionary approaches* push the boundaries further by enabling agents to modify their own core functionalities and code, achieving truly autonomous reasoning development.

The importance of reasoning in agent development has been re-emphasized following the release of the o1 series. The most straightforward approach to enhancing reasoning capabilities involves *collecting and distilling data from open/closed-source reasoning models*. For instance, SKY-32B [83] distilled data from QWQ-32B [151] to train a 32B reasoning model at a cost of \$450. Similarly, Open Thoughts [84] trained Bespoke-Stratos-32B at a low cost by distilling and synthesizing datasets from DeepSeek-R1 [116]. 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.

Building on these basic distillation approaches, researchers have identified that the *quality of reasoning data* plays an even more critical role than quantity. Highly structured reasoning data more effectively enables agents and language models to learn reasoning processes. Notably, LIMO [85] 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 [152]. Supporting this view, Li et al. [153] 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.

To systematically generate such high-quality data, iterative improvement methods have emerged that enable models to bootstrap their own reasoning capabilities. The bootstrap paradigm exemplified by STaR [86] and its variants implements techniques where models generate step-by-step rationales and iteratively improve through fine-tuning on successful reasoning paths. This family includes Quiet-STaR [118], V-STaR [154], and rStar-Math [155], with the last one 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 most sophisticated data-driven approaches integrate explicit *feedback mechanisms* to provide quality assessment during the data generation process. 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 ReST family, beginning with the original ReST [87] introducing reinforced self-training, performs multiple attempts (typically 10) per sample and creates new training datasets from successful reasoning instances. ReST-EM [156] enhances the approach with expectation maximization, while ReST-MCTS [156] 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 [88] and LLaMA-Berry [89] 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 [155] and DeepSeekMath [137] have demonstrated success in mathematical problem-solving through multi-round self-iteration and balanced exploration-exploitation strategies. For code generation, o1-Coder [157] leverages MCTS to generate code with reasoning processes, while Marco-o1 [157] 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* and *agentic self-improvement* are the most attractive learning methods. Reinforcement learning (RL) has demonstrated remarkable success in enhancing language models' reasoning capabilities at the foundational level. Recent breakthroughs like DeepSeek R1 [116] and Kimi k1.5 [138] exemplify how RL can directly improve LLMs' reasoning performance. The foundation of RL for LLMs can be traced to several pioneering frameworks: ReFT [71] introduced a combination of supervised fine-tuning and online reinforcement learning, while VeRL [158] established an open-source framework supporting various RL algorithms for large-scale models up to 70B parameters. RFT [159] further demonstrated the effectiveness of reward-guided optimization in specific reasoning tasks.

Building on these foundational frameworks, more recent research has pushed the boundaries of LLM reasoning enhancement by exploring how to cultivate reasoning capabilities in more challenging *low-resource*

settings and self-supervised manner. Shafayat et al. [160] explains that the success of self-training highly depends on the model’s initial capabilities and the nature of the task, motivating further exploration of effective self-improvement strategies. To address the heavy reliance on large-scale annotated data, innovative reinforcement learning paradigms have been proposed. For instance, Absolute Zero [161] leverages a self-play mechanism, enabling the evolution of the model’s reasoning capabilities without any human-annotated data. Pushing data efficiency to its extreme, Wang et al. [162] demonstrates the potential for effective learning and generalization from just a single successful instance.

A core challenge for these LLM-focused learning methods lies in designing effective reward signals. Spurious Rewards [163] cautions that purely outcome-based rewards can be deceptive, as correct answers can arise from flawed reasoning. This has driven a shift towards more robust, process-oriented supervision. To move away from the need for costly external verifiers, pioneering works like Verifree [164] explore the use of intrinsic signals, such as consistency among multiple reasoning paths or self-critique, to generate rewards.

Beyond single-model enhancement, reinforcement learning has evolved to train autonomous agent systems that adapt their own core functionalities through sophisticated collaboration. This system-level approach has driven a surge of innovation focused on multi-agent coordination, credit assignment, and stable training for real-world applications.

A prominent direction is the shift from single agents to multi-agent systems to tackle complexity. Frameworks like MARFT [165] now provide standardized infrastructure for multi-agent reinforcement fine-tuning, enabling novel cognitive architectures where agents collaborate as thinkers, critics, and solvers to learn “meta-think” [166]. Addressing the core RL challenge of credit assignment in long-horizon tasks, researchers have developed more granular reward mechanisms. For instance, SPA-RL reinforces agents by attributing progress at a stepwise level [167], while algorithmic improvements like GiGPO offer finer-grained policy optimization without significant overhead [168]. Concurrently, ensuring stability during this self-evolutionary process is critical. Works like RAGEN [90] analyze the dynamics of multi-turn training, identifying failure modes and proposing more robust algorithms. These advancements are empowering agents in complex, real-world domains, from long-horizon software engineering tasks [141] to dynamic web navigation, where agents trained with self-evolving online curricula can even surpass proprietary models [140].

Beyond refining existing capabilities through external learning signals, the most transformative recent advances involve agents learning to fundamentally *self-improve* by modifying their own code and underlying operational mechanisms. Alita [169] exemplifies a generalist agent that achieves maximal self-evolution with minimal predefinition by autonomously generating and reusing Model Context Protocols. This allows it to dynamically construct, refine, and integrate new capabilities, akin to writing its own “operating manuals” for continuous learning. AlphaEvolve [170] pushes this autonomous learning to the code level, serving as a coding agent that combines large language models with evolutionary computation. It can directly modify code to improve algorithms and achieve breakthroughs in scientific and algorithmic discovery, such as in areas like matrix multiplication. Furthermore, the Darwin Gödel Machine [171] takes this a step further, presenting an AI system capable of continuously improving itself by rewriting its own code, achieving the co-evolution of the agent’s own coding abilities and its self-improvement capabilities. This represents the ultimate frontier for agents to attain advanced, continuous reasoning skill learning, moving towards truly autonomous and adaptive intelligent systems.

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.

The most fundamental approach to world understanding begins with *direct environmental interaction*, where agents learn through basic trial-and-error mechanisms. Inner Monologue [92] demonstrates how agents can accumulate basic environmental knowledge through continuous interaction, while Learn-by-Interact [127] shows that meaningful understanding can emerge from direct environmental engagement without explicit reward mechanisms. These foundational approaches establish the groundwork for more sophisticated learning paradigms, with recent work on Test-Time Interaction [172] demonstrating that scaling interaction horizon—rather than just reasoning depth—enables agents to dynamically balance exploration and exploitation through extended environmental engagement.

Building upon basic interaction capabilities, agents require systematic mechanisms to *process and organize their accumulated experiences*. More sophisticated approaches are exemplified by DESP [93] and Voyager [74] in the Minecraft environment, where agents not only gather experiences but also actively process them through outcome analysis and dynamic skill library expansion, respectively. The systematic processing of experiences has been further enhanced by Generative Agents [35], which introduces sophisticated memory replay mechanisms, enabling agents to extract high-level insights from past interactions. This approach is complemented by Self-refine [94] and Critic [95], which implement structured cycles of experience evaluation and refinement, creating robust frameworks for continuous experiential learning.

A crucial component of world understanding involves learning to *interpret and optimize reward signals* through environmental interaction. Text2Reward [130] demonstrates how agents can continuously refine reward functions through human feedback, better aligning them with task objectives and environmental characteristics. Similarly, AutoManual [133] 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.

The culmination of world understanding research involves the construction and application of explicit *world models* that enable agents to simulate and reason about environmental dynamics. RAP [101] represents a significant advancement by conceptualizing reasoning as planning with a world model, repurposing LLMs as both reasoning agents and world models to simulate potential action outcomes through Monte Carlo Tree Search. The importance of structured mental representations is highlighted by cognitive maps [173], which show that cognitively-inspired representations significantly enhance LLMs' extrapolation capabilities in novel environments. In specialized domains, recent work demonstrates that LLMs can function as effective world models for web interactions [132, 174], simulating potential state changes before executing actions for safer decision-making. Furthermore, ActRe [175] explores reverse reasoning by first performing actions and then generating post-hoc explanations, demonstrating LLMs' inherent understanding of world dynamics and enabling autonomous trajectory annotation.

The most advanced systems integrate all these components into autonomous learning cycles that can self-manage the entire process of world understanding. Through systems like Reflexion [75] and ExpeL [96], agents have achieved sophisticated 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

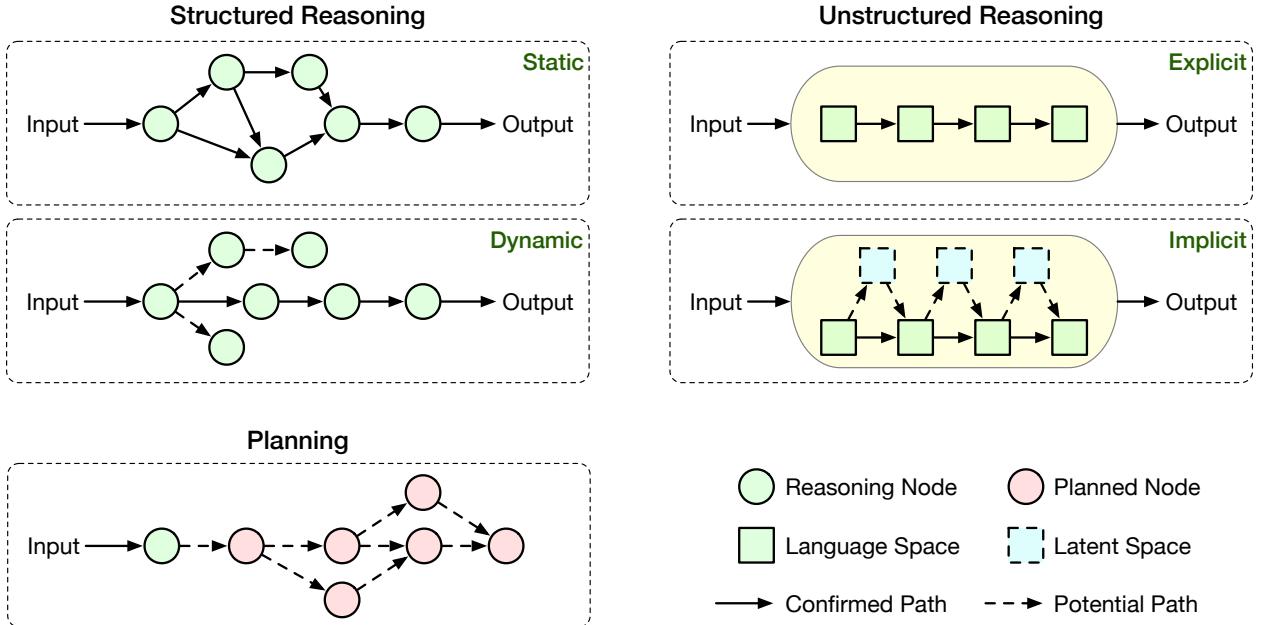


Figure 2.5: Comparison of reasoning paradigms in LLM-based agents.

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 [176, 177]. These processes rely on heuristics, mental shortcuts that streamline decision-making under uncertainty, and they 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 M_t at time t , reasoning can be formalized as a function $R(M_t) \rightarrow a_t$, where a_t represents the selected action. This process operates across diverse environments such as 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. Figure 2.5 illustrates the different reasoning paradigms. *Structured reasoning* (R_s) can be formalized as an explicit composition $R_s = R_1 \circ R_2 \circ \dots \circ R_n$, 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 first present a more concrete formulation of reasoning in general. Then we introduce structured reasoning that emphasizes systematic problem decomposition and multi-step logical chains. We next 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 A Unified Formulation of Reasoning

Reasoning is the internal process by which an agent transforms its current mental state M_t into concrete actions or decisions. These actions may range from low-level motor outputs to high-level plans, and may even involve recursively generating new subgoals. To unify diverse reasoning patterns, we formulate reasoning as the selection of a policy that minimizes a free-energy objective over imagined futures, balancing utility, curiosity, resource constraints, and computational cost.

Definition 4 (Unified Reasoning Objective). Reasoning selects a policy that minimizes expected future free energy while respecting exploration and resource constraints:

$$\begin{aligned}\pi_t^* &= \arg \min_{\pi} \sum_i v_i \mathbb{E}_{\tau \sim \mathcal{T}_i} \left[\mathbb{E}_{o_\tau, r_\tau \sim P(\cdot | \pi, M_t)} [-r_\tau] + \alpha I(o_\tau; \pi | M_t) + \lambda \mathcal{C}_{\text{act}}(\pi) \right] + \beta \mathcal{S}(\pi) \\ a_t &= \text{Extract}(\pi_t^*, M_t)\end{aligned}$$

Utility term: The expected reward $\mathbb{E}[-r_\tau]$ penalizes undesirable outcomes or low task utility. When reward reflects goal satisfaction, this term recovers classical reinforcement learning and optimal control; when reward reflects prior preferences, it recovers active inference.

Exploration term: The mutual information term $I(o_\tau; \pi | M_t)$ encourages exploratory behavior by rewarding policies that reduce epistemic uncertainty. Setting $\alpha = 0$ yields pure exploitation; $\alpha > 0$ induces curiosity.

Action cost: The cost term $\mathcal{C}_{\text{act}}(\pi)$ reflects practical penalties such as energy usage, risk, or API fees.

Search cost: The search term $\mathcal{S}(\pi)$ quantifies internal computational effort, such as number of nodes, tokens, or floating-point operations. The weight β enforces bounded rationality.

Temporal mixture: The coefficients v_i allow reasoning to mix over different time horizons \mathcal{T}_i , from short-term reflexes to long-term planning. These mirror the horizon weights used in the learning objective.

Policy hierarchy: A policy π can represent decisions at various levels of abstraction—from atomic actions to multi-step strategies. The extraction function $\text{Extract}(\pi_t^*, M_t)$ maps the selected policy to an executable action, possibly invoking another round of reasoning in a receding-horizon fashion.

While the unified objective offers a theoretical view of reasoning, practical systems must implement it with finite resources, modular components, and clear interfaces. In particular, most reasoning engines—from classical planners to LLM-based agents—can be interpreted as repeatedly transforming internal hypotheses via a combination of *rewrite operators* and a *scheduling policy*.

Implementation Scaffold To realize the unified objective in practice, intelligent agents adopt a concrete structure based on a loop over internal hypotheses. This loop consists of selecting and applying transformation

operators from a library \mathcal{L} , under the guidance of a scheduler policy π , starting from an initial seed hypothesis $h_0 = \text{Init}(M_t)$. This abstraction captures a wide variety of paradigms—including symbolic planning, neural program synthesis, and chain-of-thought prompting—and can be formally described as follows.

Definition 5 (Operator-Scheduler Realisation). Let \mathcal{H} be the space of internal hypotheses. An initial draft $h_0 = \text{Init}(M_t)$ is produced from the current mental state. A finite operator library $\mathcal{L} = \{r^{(1)}, \dots, r^{(K)}\}$ defines atomic rewrites $r^{(k)} : \mathcal{H} \rightarrow \mathcal{H}$. At each step i , the scheduler samples $\sigma_i \sim \pi(\cdot | h_{i-1}, M_t)$, and applies the selected operator: $h_i = r^{(\sigma_i)}(h_{i-1})$. The iteration halts when a termination predicate $\text{Term}(h_i, M_t)$ is satisfied or a resource budget is exhausted. The final draft h_N is converted into an action via $\text{Extract}(h_N, M_t)$.

This iterative process produces the **reasoning trace** $\zeta_t = (h_0, h_1, \dots, h_N)$, which records the internal deliberation sequence for step t . It provides a unified pattern encompassing classical search trees, stochastic rollouts, and token-by-token language generation.

Importantly, the scheduler policy π that governs the application of operators is exactly the object being optimized in the free-energy formulation (Definition 4). The operator dynamics define the transitions through hypothesis space, while the scheduler implicitly balances utility, curiosity, action cost, and compute effort.

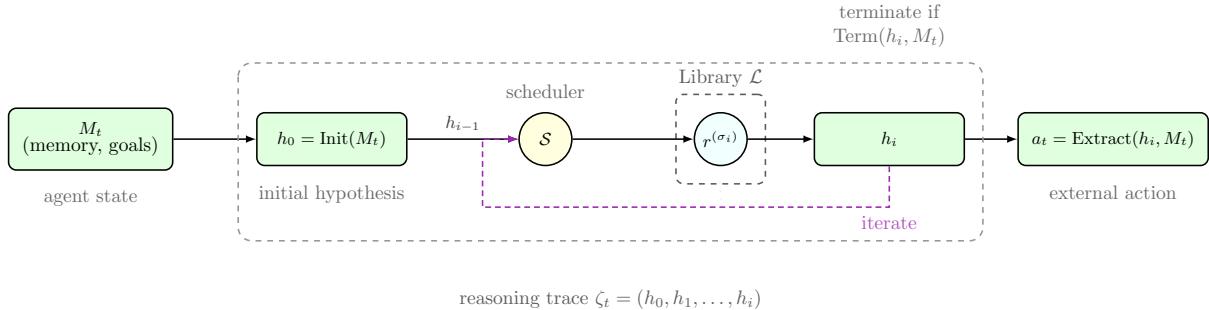


Figure 2.6: Reasoning as a scheduler-controlled sequence of atomic rewrites. From the mental state M_t the agent seeds an initial hypothesis h_0 . The scheduler S (orange) repeatedly chooses an operator $r^{(\sigma_i)}$ from the library \mathcal{L} (green) and applies it to the current draft, producing the updated hypothesis h_i . This loop continues (purple dashed arrow) until the termination predicate $\text{Term}(h_i, M_t)$ is satisfied, after which the final hypothesis is converted to an external action $a_t = \text{Extract}(h_i, M_t)$. The dashed frame encloses the reasoning trace $\zeta_t = (h_0, h_1, \dots, h_i)$, a pattern that subsumes chain-of-thought, tree search, and program-synthesis planners alike.

To demonstrate the generality of this framework, we summarize a wide range of reasoning paradigms—across AI planning, reinforcement learning, inference, and LLM-based techniques—as specific instantiations of the operator–scheduler pattern. Table 2.3 presents each paradigm’s scheduler policy, utility signal, exploration strategy, operator set, and search cost, all of which align with the unified formulation.

Symmetry with Learning Learning optimizes internal models using real-world data; reasoning optimizes policies over imagined futures simulated by those models. Both processes minimize composite free-energy losses of the form:

$$\text{expected loss} + \alpha \times \text{uncertainty penalty} + \lambda \times \text{action cost} + \beta \times \text{resource cost}$$

Together, they complete the perception–cognition–action loop at the heart of intelligent agents.

Table 2.3: Reasoning paradigms as instantiations of the operator–scheduler realisation. Each row specifies the components that instantiate the unified objective: scheduler policy, utility, exploration term, operator set, and search cost. All conform to the free-energy skeleton of Definition 4.

Paradigm	Scheduler policy π	Utility signal	Exploration term	Operator set \mathcal{L}	Search cost \mathcal{S}
Reactive control	rule-based single action	immediate reward	re- $\alpha = 0$	identity	negligible
Model-free RL	param. $\pi_\theta(a s)$	discounted return	re- turn entropy bonus	identity	gradient steps / batch
Monte-Carlo search	tree policy over moves	win-loss	UCB bonus	expand, rollout, backprop	node budget
Hierarchical RL	option policy	option value	KL to prior	invoke_option, terminate	option horizon budget
Active inference	trajectory distribution	prior preference	epistemic F.E.	integrator	gradient iterations
LLM chain-of-thought	token policy	reward / log-prob	variance penalty	append_token, self_eval	token budget
Theorem proving	proof-tactic policy	proof length cost	premise novelty	tactic library	node expansions
Program synthesis	code-edit policy	test failure cost	branch-uncertainty bonus	insert, replace, delete	edit budget
Meta-controller	policy over planners	outer-loop utility	info-gain across tasks	spawn_planner, eval_planner	run budget

Module Interfaces Operators may call the world model W for simulation, query memory for context, or receive urgency/temperature parameters from the emotion state E_t . Their trainable parameters join the global set Θ updated by the learning rule L .

Our formulation of reasoning therefore captures different reasoning styles examined in the rest of this chapter while remaining consistent with the overarching perception–cognition–action loop of Foundation Agents.

2.2.2 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 $R_s = R_1 \circ R_2 \circ \dots \circ R_n$, 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.

2.2.2.1 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 [97] 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) [101] 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)

[98] 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 [178] 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. Furthermore, some research has begun to challenge the Markovian assumption prevalent in many linear reasoning processes. Zhang et al. [179] argues that complex reasoning is inherently non-Markovian and requires consideration of the entire history. It proposes a Reflective Exploration framework based on Bayes-Adaptive RL, which allows the agent to update its strategy based on the full reasoning trace, providing a theoretical foundation for building more powerful long-horizon reasoning models.

Tree-Based Exploration Tree-based approaches expand beyond linear structures by organizing reasoning into hierarchical frameworks that support branching exploration. Tree of Thoughts (ToT) [99] 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) [100] 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) [101] 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) [102] 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) [103] 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) [104] 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.

2.2.2.2 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 [105] 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 [180] 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 [181] 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 [94] 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 [75] 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) [106] 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 [107] 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 [182] 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) [108] 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) [173] 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 [95] 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.

2.2.2.3 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 [109] 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 [111] 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) [110] leverages educational theory, particularly the Bloom Cognitive Model, to guide the identification of reasoning mistakes in mathematical contexts. This approach combines pedagogi-

cal 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.

2.2.3 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.

2.2.3.1 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 [73], 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 [183] 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 [184] automates the creation of effective demonstrations by clustering diverse questions and generating reasoning chains for representative examples from each cluster. Least-to-Most Prompting [185] addresses complex reasoning by decomposing problems into sequential sub-problems, enabling a progressive planning process that facilitates easy-to-hard generalization. Complex CoT [186] 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 [112] 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 [187] 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 [188] 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 by introducing structured constraints and knowledge integration mechanisms. Ask Me Anything [113] constrains open-ended generation through task reformulation into question-answer sequences, while Algorithm of Thoughts [189] exploits the recurrence dynamics of LLMs by employing algorithmic examples in context to guide reasoning pathways. Chain-of-Knowledge (CoK) [114] addresses factual grounding by dynamically incorporating heterogeneous knowledge sources through adaptive query generation, and Self-Explained Keywords (SEK) [115] tackles low-frequency term challenges by extracting and explaining key concepts within the reasoning process itself.

2.2.3.2 Reasoning Models

As described in 2.1.4, both models and agents have achieved enhanced reasoning capabilities through various approaches, with these reasoning abilities being inherently embedded within the model parameters rather than constrained by human-imposed structural limitations. Recent advances in language models have led to the development of specialized reasoning models designed explicitly for complex inferential tasks, incorporating data, architecture, and training innovations that optimize reasoning capabilities and enhance performance on tasks requiring logical inference.

Reasoning models like DeepSeek’s R1 [116], Anthropic’s Claude 3.7 Sonnet [9], and OpenAI’s o series models [117] represent the frontier of reasoning capabilities, demonstrating remarkable proficiency across diverse reasoning benchmarks. 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 [85, 90, 190, 191]. 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.

2.2.3.3 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 [118] 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 ($5.9\% \rightarrow 10.9\%$) and commonsense reasoning ($36.3\% \rightarrow 47.2\%$) after continued pretraining, marking a step toward LMs that learn to reason in a more general and scalable way.

Chain of Continuous Thought (Coconut) [119] 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.

Seeking to harness the efficiency of implicit methods while retaining the robustness of explicit thought, the framework proposed in System-1.5 Reasoning [192]. It allows a model to dynamically shortcut from explicit,

step-by-step linguistic reasoning into an efficient latent space for parallel exploration, and then return to the explicit space. This enables faster, “insight-like” leaps in logic, offering a new direction for building more powerful reasoning systems that combine the speed of intuition with the rigor of deliberation.

However, the fundamental generalization capabilities of implicit reasoning, including these hybrid forms, remain a critical concern. Recent analysis [193] 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.

2.2.4 Planning

Planning is a fundamental aspect of human cognition, enabling individuals to organize actions, anticipate outcomes, and achieve goals in complex, dynamic environments [194, 195]. Formally, planning can be described as the process of **constructing potential pathways from an initial state to a desired goal state**, represented as $P : S_0 \rightarrow \{a_1, a_2, \dots, a_n\} \rightarrow S_t$, where S_0 is the source state, $\{a_1, a_2, \dots, a_n\}$ denotes a sequence of possible actions, and S_t 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 [196, 197].

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 [198]. 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 [199]. Additionally, their reliance on static pre-training data restricts adaptability in real-time scenarios, limiting their generalization in dynamic planning tasks [200]. The absence of an intrinsic System 2 reasoning mechanism further complicates their ability to independently generate structured,

optimal plans [201]. 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 [202]. The Least-to-Most Prompting method [185] exemplifies this approach, guiding LLMs to solve subproblems incrementally. ADaPT [203] 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 [178], 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 [93]. These tasks vary in nature: some are subproblems requiring specific solutions such as solving equations within broader challenges, while others involve tool invocation such as querying APIs for weather data in travel planning [204, 205]. Alternatively, tasks may be represented as graph nodes mapping dependencies, such as prioritizing objectives in project management [206]. By defining clear, modular goals, these formulations enhance reasoning and action efficiency, guiding agents through complex problem spaces with greater precision [120].

Searching Given the stochastic nature of LLMs [207], 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 [99, 208]. This space allows sampling diverse potential solutions [209], facilitating exploration through techniques like reflection, review, and parallel sampling informed by existing knowledge [210].

Computational constraints often preclude exhaustive evaluation, making efficient navigation of the solution space essential. Methods include tree search algorithms like LATS [211], heuristic approaches such as PlanCritic’s genetic algorithms [212], and CoT-SC, which identifies recurring solutions via self-consistency checks [105]. Reward-based models like ARMAP assess intermediate and final outcomes to optimize planning [131]. This iterative exploration and refinement process enhances adaptability, ensuring robust strategies for complex problems.

World Knowledge Integration Effective planning requires agents to navigate dynamic environments, anticipate changes, and predict outcomes, underscoring the importance of world knowledge. RAP [101] 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 [132, 213]; as agents, they select actions based on states and goals [97]. 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 [214].

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 [97]. This enables LLMs to iteratively refine their world models during action execution, supporting adaptive planning. Conversely, LLM+P [215] integrates LLMs with the PDDL planning language, converting natural language inputs into formalized representations solved by classical planners [216, 217]. 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 [218] uses code-form plans—pseudocode outlining logical steps—to guide LLMs through complex tasks, achieving notable performance improvements across benchmarks [219]. The World Knowledge Model (WKM) equips

LLMs with prior task knowledge and dynamic state awareness, reducing trial-and-error and hallucinations in simulated environments [220]. Unlike language-centric models, the Joint Embedding Predictive Architecture (JEPA) leverages self-supervised learning from multimodal data to construct predictive world models, efficiently capturing dynamic environmental changes and offering a robust framework for adaptive planning [221]. 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 [222]. Together, these methods illustrate how structured frameworks and environmental understanding can transform LLMs into effective planners.

2.3 Summary and Discussion

HE cognitive evolution of intelligent agents represents a spiraling progression of learning and reasoning, much like how biological intelligence achieve higher reasoning efficiency after acquiring better knowledge and logic. Our unified framework clearly describes this spiral relationship: the learning process continuously optimizes mental states through $M_t = L(M_{t-1}, a_{t-1}, o_t)$, while the reasoning process makes better decisions based on updated mental states through $a_t = R(M_t)$. This mutually reinforcing mechanism enables agents to continuously enhance cognitive capabilities through experience accumulation. Richer experiential data improves learning effectiveness, while better mental state representations support more efficient reasoning processes.

Intelligent agents possess remarkable diversity in learning forms. Without changing model parameters, prompt engineering, workflow design, few-shot examples, memory mechanisms, action space definitions, and even agent logic itself can be learned. When model parameters are modified, data acquisition strategies (how agents explore environments or collect external knowledge) and various training paradigms (SFT, LoRA, RL, etc.) all provide possibilities for capability enhancement. This rich learning space presents both opportunities and challenges. The key question is: **when facing such a broad learning space, we need to master optimal learning forms for agents in different scenarios**. Furthermore, true Foundation Agents should possess meta-learning capabilities: understanding the relationship between scenarios and learning forms to automatically construct appropriate agent architectures and continuously improve them. This requires agents not only to execute learning but to learn how to learn.

However, enabling agents to engage in active learning faces a fundamental difficulty. We can provide agents with clear external goals to drive learning, but this logic differs essentially from humans. Human behavior stems from internal reward signals produced by hormonal systems, with basic motivations being survival and reproduction. In contrast, agents (particularly language models) naturally lack such internal motivation settings, causing their behavior to remain passive and reactive rather than active and exploratory. We propose a key hypothesis: **solving the internal reward system problem for agents is necessary to truly motivate agent behavior, including active learning capabilities**. This is not merely a technical challenge but involves designing reasonable internal drive mechanisms for artificial systems to generate spontaneous exploration and improvement motivation like biological agents.

Another critical issue lies in the gap between the capability requirements for agent reasoning and current reasoning research directions. Reasoning is essentially the effective use of internal states after learning, which contains two core components: cognitive ability for internal logic and understanding of external world dynamics. Existing work on improving language model reasoning capabilities mostly focuses on mathematical and logical domains, dedicated to enhancing internal logic exploration abilities. However, relatively few works consider how to enhance language models' or agents' understanding capabilities of external world dynamics.

This raises a fundamental question: **are agents' poor performance in diverse environments due to insufficient logical reasoning capabilities or inadequate understanding of world dynamics?** Recent

research provides important insights into this question. Shen et al. [172] demonstrated that agents acquiring as much current environmental dynamic information as possible during reasoning can significantly improve agent capabilities, indicating the crucial role of environmental understanding in reasoning. Meanwhile, ActRe [175] further proved the important impact of world dynamics understanding capabilities embedded within models on reasoning performance. These findings suggest that understanding of world dynamics may be the key bottleneck limiting agent reasoning capabilities.

Current reasoning research focuses too much on how to think better while relatively neglecting what to think based on. True reasoning capability improvement may require shifting attention from pure logical structure optimization toward joint enhancement of logical capabilities and world understanding, demanding we examine the relationship between reasoning and world modeling.

In conclusion, the future of cognition in Foundation Agents lies in their ability to balance the need for sophisticated learning mechanisms with the requirement for internal motivation systems. By continuing to explore and refine the interplay between passive external guidance and active internal drives, between structured reasoning capabilities and adaptive world understanding, we move closer to developing AI systems that not only can learn and reason effectively but can autonomously drive their own cognitive evolution in an ever-changing world.

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, intelligent 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 intelligent agent memory systems. 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

NDERSTANDING human memory is central to both cognitive science and the design of intelligent systems. Memory enables organisms to accumulate knowledge, adapt behavior based on past experience, and project into the future. In humans, memory is not a monolithic faculty but a constellation of interacting subsystems that differ in temporal span, representational format, and accessibility to consciousness. These systems collectively support a wide range of functions, from the rapid detection of sensory stimuli to the deliberate recollection of life events and the fluent execution of learned skills. In this section, we survey foundational classifications of memory and examine prominent theoretical models, each offering a distinct lens through which to understand how information is encoded, stored, and retrieved in the human mind.

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 [223, 224]. Within long-term memory, explicit (declarative) and implicit (non-declarative) forms are further delineated [225]. Figure 3.1 illustrates one such hierarchical framework:

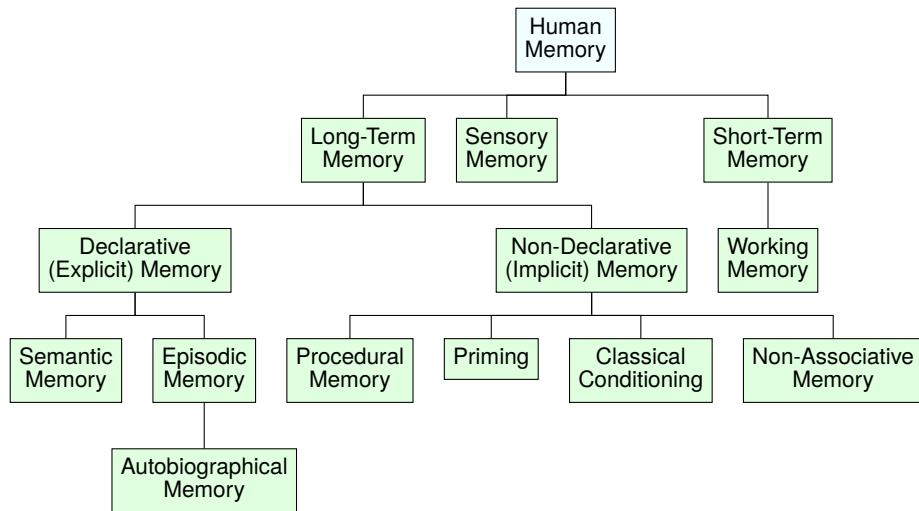


Figure 3.1: The hierarchical taxonomy of the human memory system.

- **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 [226]. Iconic memory (for visual input) [227] and echoic memory (for auditory input) [228] 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) [229, 230]. The capacity of STM or working memory is finite, typically cited as around seven plus or minus two chunks of information [231], though individual differences and task factors can modulate this figure.
- **Long-Term Memory (LTM).** Long-term memory accommodates the more durable storage of information that can persist from hours to decades [232, 233]. 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 [234].
 - **Declarative (Explicit) Memory.** Declarative memory encompasses memories that can be consciously recalled and articulated [235]. Within this broad category, researchers often discuss:

- * **Semantic Memory:** Factual knowledge about the world, including concepts, words, and their relationships [236]. 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 [237]. 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 [238]. 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 [239]. 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 [240, 241].
 - * **Priming:** The phenomenon in which prior exposure to a stimulus influences subsequent responses, often without explicit recognition of the previous encounter [242].
 - * **Classical Conditioning:** The learned association between two stimuli, where one stimulus comes to elicit a response originally produced by the other [243].
 - * **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 [244, 245].

Despite the orderly appearance of these categories, human memory processes often overlap and intersect. 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 distinct 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.

The Multi-Store (Modal) Model A seminal proposal by Atkinson and Shiffrin [223] 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.

Working Memory Models Recognizing that short-term memory also involves active maintenance, Baddeley and Hitch [246] 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

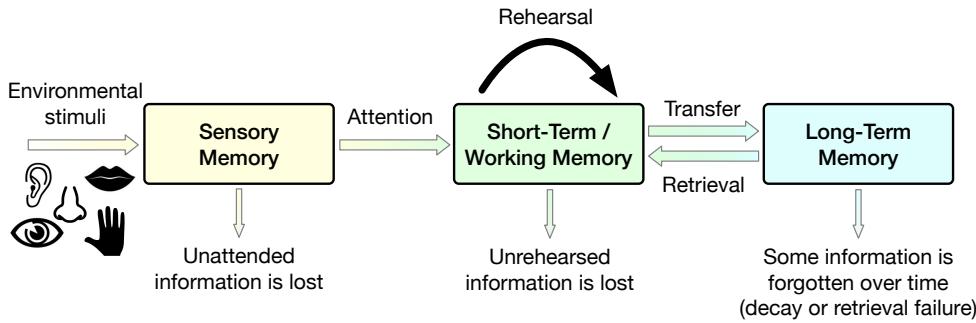


Figure 3.2: Atkinson-Shiffrin three-stage model of human memory [223].

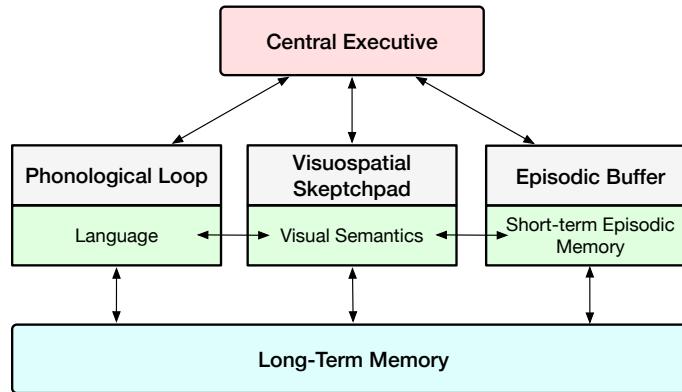


Figure 3.3: Baddeley's model of working memory [246].

(verbal) and the visuospatial sketchpad (visual/spatial). A subsequent refinement introduced the episodic buffer to integrate material from these subsystems with long-term memory [247]. Figure 3.3 shows the framework of the working memory model. Alternatives such as Cowan's embedded-processes model [248] similarly underscore the role of attention in governing how information is briefly sustained and manipulated.

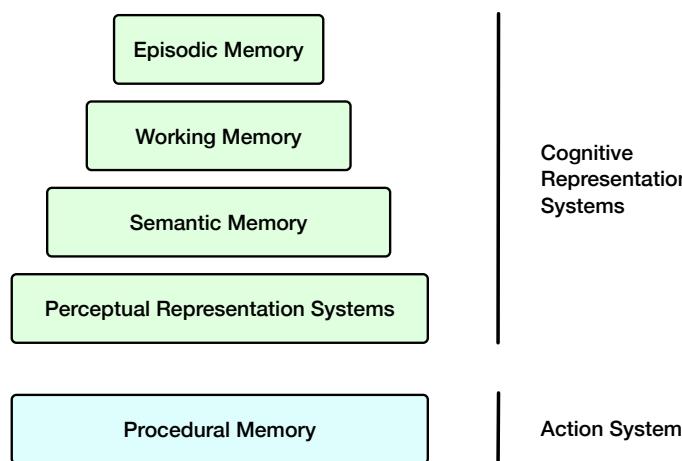


Figure 3.4: The Serial-Parallel Independent (SPI) model of human memory [249].

Serial-Parallel-Independent (SPI) Model Initial distinctions between episodic, semantic, and procedural memory were championed by Tulving [249], 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 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 [250], conceptualizes consciousness and working memory as a “broadcast” mechanism that distributes information to specialized processors. Building on GWT, Franklin [251, 252] 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.

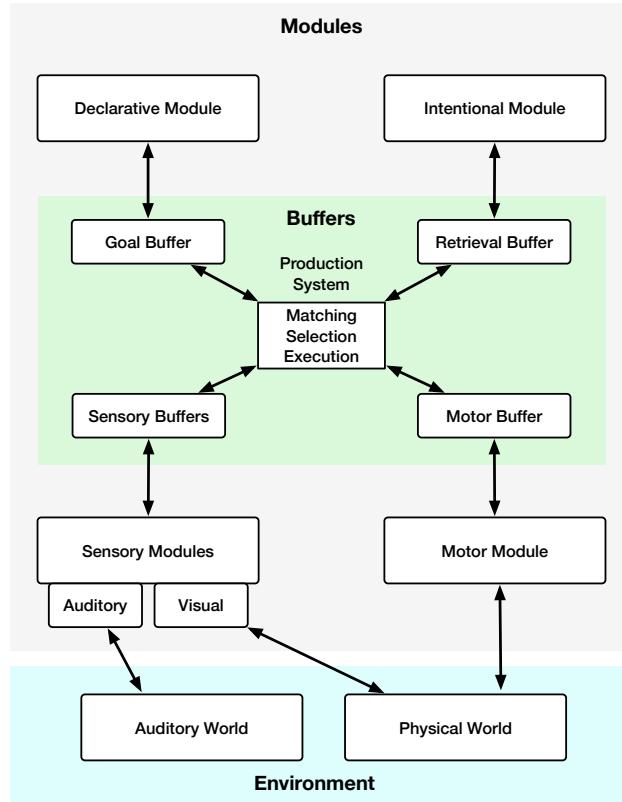


Figure 3.5: An abstraction of the most important processes in the ACT-R model [253].

ACT-R and Cognitive Architectures ACT-R (Adaptive Control of Thought—Rational) [253] is a 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.

In summary, each of the aforementioned models illuminates different aspects of memory. For example, 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

AVING 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 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) [254] 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) [255], have evolved. Modern architectures employ more sophisticated techniques, including vector embeddings for rapidly retrieving memories [256] and selective incorporation of reasoning chains into subsequent inference steps [257, 258]. 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 are 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? Figure 3.6 shows an overview of selected relevant research works. The following sections delve into these crucial areas, exploring current approaches, limitations, and potential future directions.

3.3 Representation of Agent Memory

 NSPIRED by human cognitive systems [339], current memory architecture in intelligent agents adopts a hierarchical framework that integrates perception through sensory memory [259], real-time decision-making via short-term memory [340, 341], and sustained knowledge retention through long-term memory [342, 343, 75]. 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.

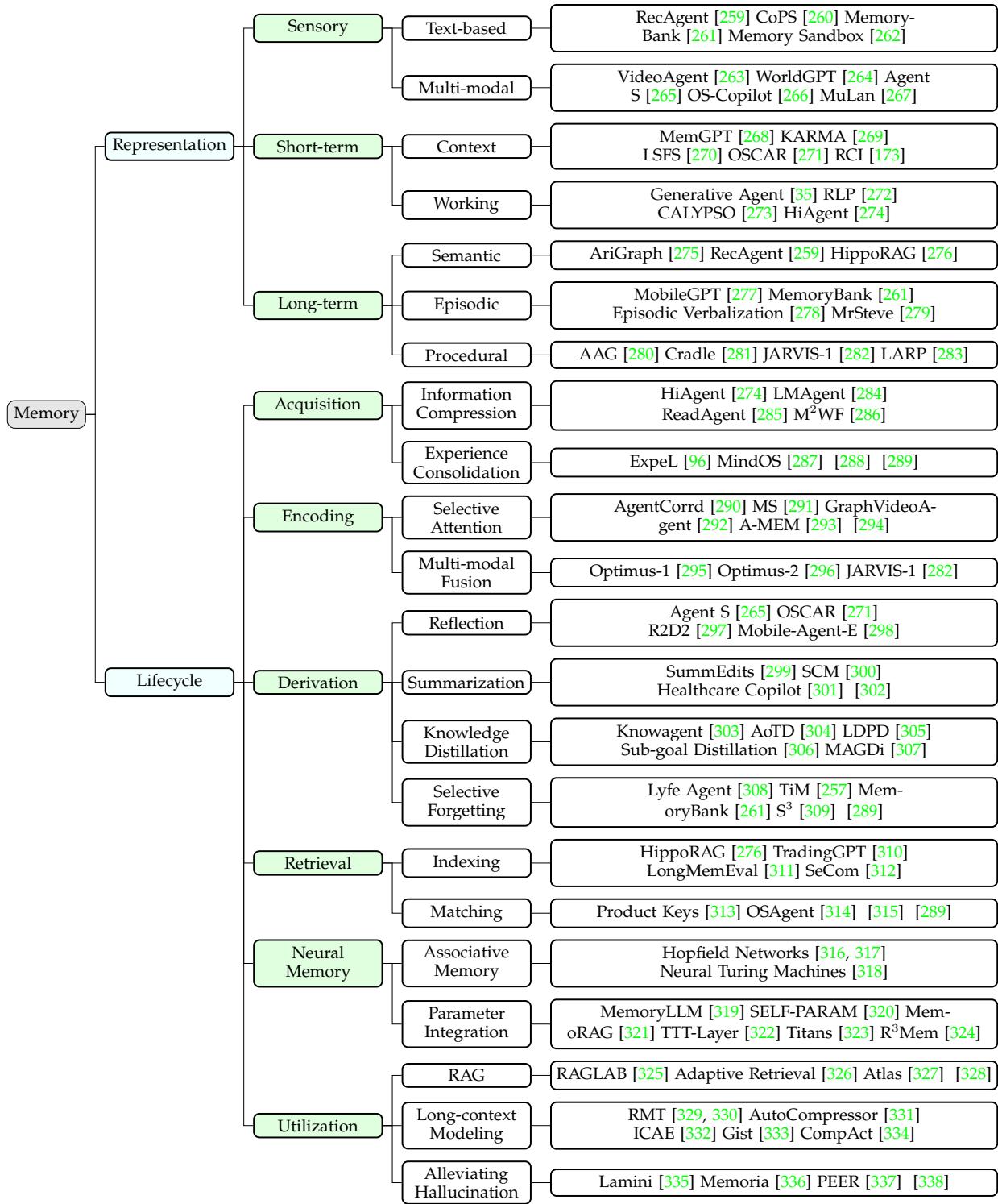


Figure 3.6: A taxonomy of selected research works about different memory modules in intelligent agents.

3.3.1 Sensory Memory

In human cognitive systems, as shown in Figure 3.7, sensory memory serves as a mechanism for collecting information through the senses, such as touch, hearing, and vision, and is characterized by its extremely

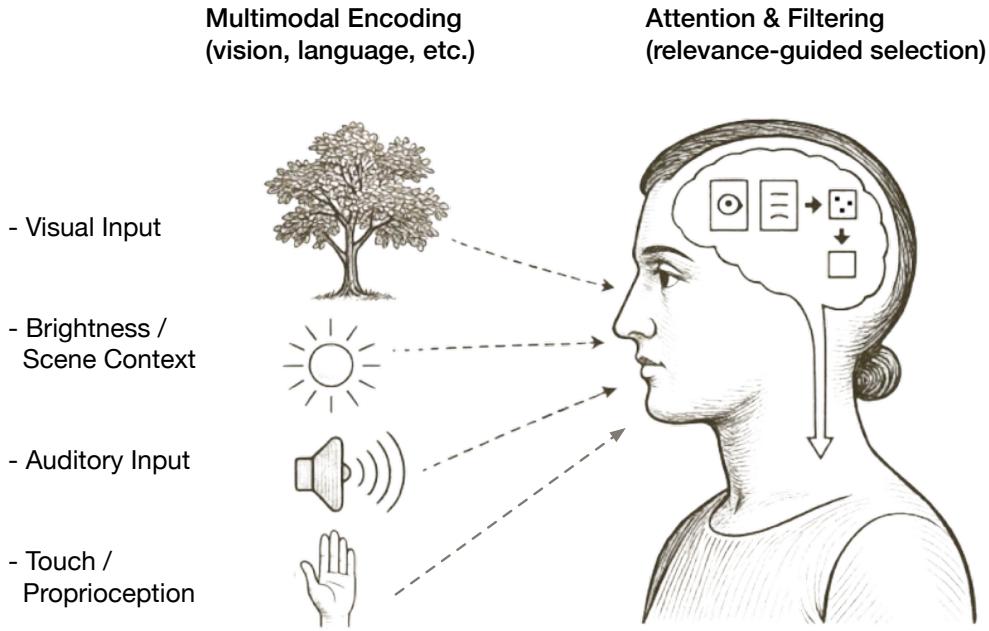


Figure 3.7: Illustration of sensory memory formation in intelligent agents. Multimodal sensory inputs—such as vision, sound, touch, and contextual signals—are first encoded into internal representations. An attentional mechanism filters these representations to prioritize task-relevant information, discarding noise. The selected content is temporarily retained in the sensory memory buffer, forming the foundation for higher-level cognitive processing.

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. Crucially, there's a fundamental distinction between the raw, high-bandwidth, and often analog nature of true sensory input and the already processed, typically textual or symbolic, information that many current intelligent agents receive as "observations". True sensory memory in humans deals with this unprocessed deluge, holding it fleetingly for the cognitive system to perform rapid, pre-attentive feature extraction and selection. For intelligent agents, especially those based on LLMs, the input is often pre-processed, such as text descriptions of a scene or tokenized language. Therefore, the "sensory memory" in agents often refers to the initial buffering and encoding of these already somewhat abstracted inputs. Recent research continues to explore how LLMs process and generate sensory language, often finding discrepancies compared to human patterns, potentially influenced by training data and RLHF techniques [344]. The challenge remains in bridging this gap: how can agents effectively process and compress truly raw, multimodal sensory data into formats (like text or embeddings) that are amenable to higher-level reasoning, and how much vital information is lost or distorted in this necessary transformation?. This initial conversion is critical, as it determines the quality and nature of the information available for all subsequent memory stages. Some recent works investigate emergent self-awareness in multimodal LLMs through sensorimotor experiences, highlighting the role of sensory integration and memory [345].

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:

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

where o_t is the sensory input at time t , and s_t represents the agent's state. For instance, RecAgent [259] employs an LLM-based sensory memory module to encode raw observations while filtering noise and irrelevant content. Extending beyond text-based perception, multimodal sensory memory systems such as Jarvis-1 [282], VideoAgent [263], and WorldGPT [264] 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:

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

where $\phi(o_t)$ is the encoded input, and c_t denotes contextual information influencing attention. For example, RecAgent [259] 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.

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

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

Several strategies have been implemented to manage the time window. For instance, RecAgent [259] 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 (observation, importance score, timestamp). Similarly, CoPS [260] 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. Figure 3.8 illustrates short-term memory with a vivid example.

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 [268], inspired by hierarchical memory systems in operating systems, manages different storage tiers to extend context beyond the LLM's inherent limitations. [346] 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 [35] employs short-term memory to retain situational context, facilitating context-sensitive decision-making. Reflexion [75] utilizes a sliding window mechanism to capture and summarize recent feedback, balancing detailed immediate experiences

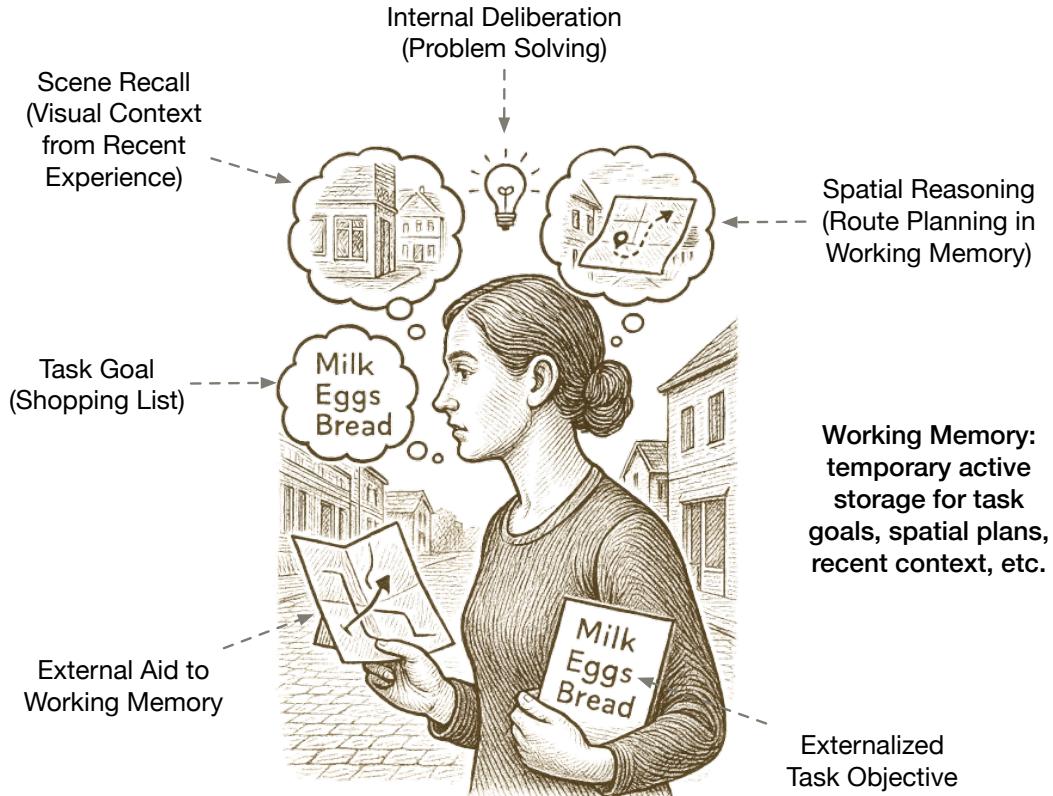


Figure 3.8: Illustration of short-term and working memory in action. The figure depicts an agent navigating an urban environment while holding a shopping list and a map. Around her, thought bubbles represent key contents temporarily held in working memory, including task goals (e.g., grocery items), spatial plans (e.g., route to the destination), visual recall (e.g., remembered storefronts), and active reasoning (e.g., route decisions). This dynamic workspace enables the agent to integrate recent observations, task demands, and internal deliberation to guide immediate behavior, capturing the essence of short-term memory as both transient storage and active processing for adaptive cognition.

with high-level abstractions for enhanced adaptability. RLP [272] maintains conversational states for speakers and listeners, using them as prompts to support dialogue understanding and generation.

For interactive and creative game scenarios, CALYPSO [273] 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 [265] and Synapse [347], 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 [348] leverages scene graphs and environmental feedback as short-term memory to guide planning and execution in scalable robotic environments. KARMA [269] engages short-term working memory with an effective and adaptive memory replacement mechanism to dynamically record changes in objects' positions and states. LLM-Planner [349] iteratively updates short-term memory with environmental observation to prompt an LLM for dynamic planning.

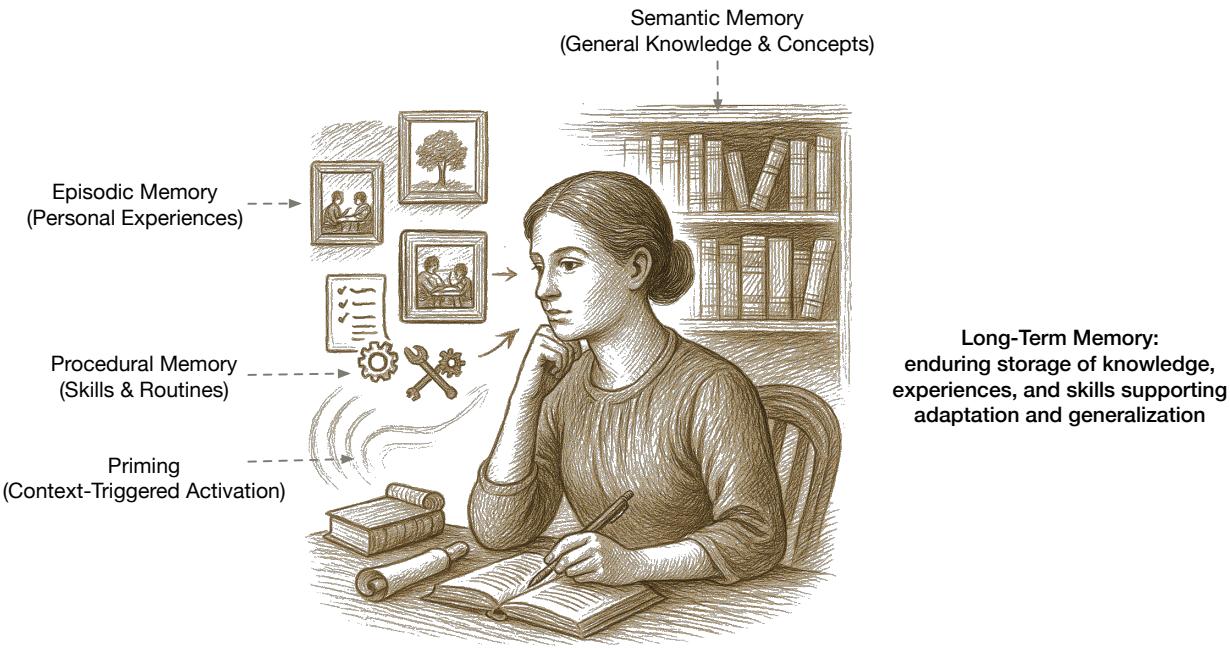


Figure 3.9: Illustration of long-term memory in cognition-inspired intelligent agents. The figure depicts a reflective agent drawing upon different types of long-term memory. In the background, bookshelves represent semantic memory, which refers to accumulated general knowledge and conceptual facts, while framed visual scenes signify episodic memory, consisting of past experiences and interactions. In the foreground, procedural memory is symbolized by tools and checklists used to encode skills and reusable plans. Subtle motion lines represent priming, which enables rapid, context-sensitive activation of relevant responses. Together, these memory systems support long-term retention, cross-task generalization, and adaptive behavior in intelligent agents.

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. Figure 3.9 illustrates the key components of long-term memory in cognition-inspired agents, including semantic, episodic, procedural, and priming-based mechanisms that collectively support adaptation, knowledge accumulation, and skill reuse.

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.

Most intelligent agents implement both semantic and episodic memory within their memory modules. For instance, Agent S [265], 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 [275], 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 [261] for SiliconFriend, semantic memory constructs user portraits in natural language, while episodic memory retains interaction histories, enhancing personalized and context-aware behavior. Recently, Mem0 [143] introduced an open-source, production-ready memory architecture that dynamically extracts, consolidates, and retrieves salient conversational information for long-term consistency. Its graph-based extension, Mem0^g, further enables structured relational reasoning by modeling conversational elements as entity-relation graphs.

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 [280] 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 [281] and Jarvis-1 [282] 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

HUMAN COGNITION is underpinned by a dynamic memory system that processes information through stages: *acquisition*, *encoding*, *consolidation*, *retrieval*, and *utilization*. These stages are interdependent, enabling humans to learn from experiences, adapt to new situations, and make informed decisions. Similarly, designing intelligent agents with a structured memory lifecycle allows for more robust, adaptable, and context-aware behavior. By mirroring the human memory process, we can create AI systems that not only process information efficiently but also learn and evolve over time.

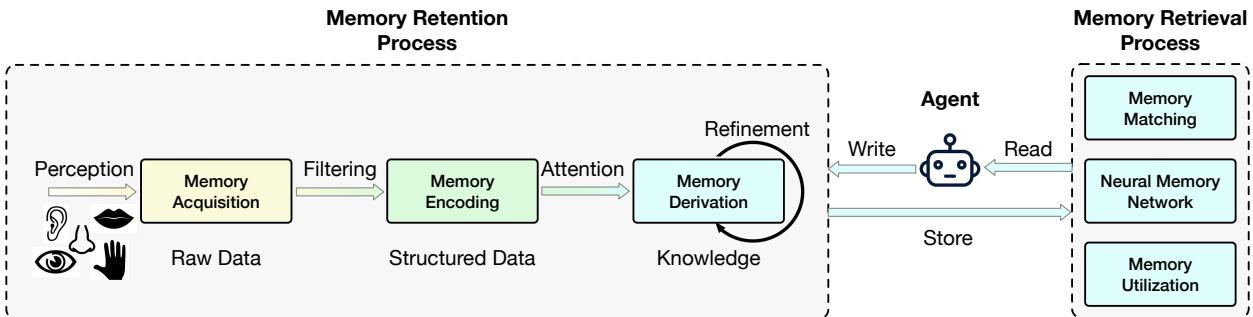


Figure 3.10: Illustration of the memory lifecycle. The memory retention process involves three sequential steps—memory acquisition, encoding, and derivation, while the memory retrieval process encompasses several independent applications, including matching (vector search), neural memory networks, and memory utilization (for long-context modeling and hallucination mitigation).

As illustrated in Figure 3.10, the memory lifecycle in intelligent agents can be conceptualized as a dual process comprising *memory retention* and *memory retrieval*, each encompassing a sequence of interconnected stages that collectively govern how information is perceived, processed, stored, and applied. Retention includes acquisition, encoding, and derivation, responsible for capturing raw inputs, transforming them into structured internal representations, and refining them into actionable knowledge. Retrieval, in turn, involves memory matching, neural memory networks, and utilization, enabling agents to access relevant information, reason with past experiences, and guide behavior in real time. A clear understanding and implementation of this lifecycle is essential for developing intelligent agents with enhanced learning capabilities, contextual awareness, and decision-making proficiency.

To complement this conceptual perspective, we introduce a unified computational view that frames the memory lifecycle as a latent optimization problem.

Definition 6 (Memory Lifecycle Optimization). At each time step t , the agent maintains a memory state $\mathcal{M}_t^{\text{mem}}$, which approximates an ideal latent memory target $\mathcal{M}_t^{\text{mem}^*}$. This latent memory encodes the minimal sufficient information necessary to support optimal future behavior, planning, and reasoning.

$$\mathcal{M}_t^{\text{mem}} = \arg \min_{\mathcal{M}} \mathcal{L}_{\text{mem}}(\mathcal{M}, \mathcal{M}_t^{\text{mem}^*}) + \lambda_{\text{store}} \cdot \mathcal{C}_{\text{store}}(\mathcal{M}) + \lambda_{\text{retr}} \cdot \mathcal{C}_{\text{retr}}(\mathcal{M}) \quad (3.4)$$

Divergence Term: $\mathcal{L}_{\text{mem}}(\cdot)$ quantifies the semantic deviation or utility gap between the constructed memory and the task-optimal latent memory.

Storage Cost: $\mathcal{C}_{\text{store}}(\cdot)$ models the cost of storing memory, including memory size, update complexity, and representational redundancy. It can be further decomposed into stage-specific components, such as \mathcal{C}_{acq} for acquisition and \mathcal{C}_{enc} for encoding.

Retrieval Cost: $\mathcal{C}_{\text{retr}}(\cdot)$ captures the computational overhead of retrieving and applying memory in real time.

Fidelity–Efficiency Trade-off: The trade-off among these objectives is governed by the coefficients λ_{store} and λ_{retr} , which regulate the balance between memory fidelity and computational efficiency.

This formalism not only provides a high-level abstraction in the memory lifecycle, but also facilitates principled memory assessment, i.e., the divergence term \mathcal{L}_{mem} serves as a general metric for evaluating how useful, sufficient, and task-aligned the current memory state is in approximating the latent optimal memory.

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 [350]. 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 is not merely about reducing quantity; it’s an essential step towards constructing a simplified, yet representative, internal model of the external world. The agent needs to compress information not just for efficiency, but to extract salient features and patterns that “fit” or approximate the underlying structure of reality, discarding noise and retaining fidelity for key aspects. This approximation is a continuous process, beginning here with coarse-grained selection and compression.

Formally, we consider this stage as an information selection and lossy compression process that maps raw sensory input o_t into a compressed memory sketch \tilde{o}_t stored as part of $\mathcal{M}_t^{\text{mem}}$. The goal is to preserve

task-relevant information while minimizing unnecessary redundancy:

$$\mathcal{M}_t^{\text{mem}} \leftarrow \text{Compress}(o_t) = \arg \min_{\tilde{o}_t=f_{\text{acq}}(o_t)} \mathbb{E}_{o_t} [D(\tilde{o}_t, o_t)] + \lambda_{\text{acq}} \cdot \mathcal{C}_{\text{acq}}(\tilde{o}_t) \quad (3.5)$$

$f_{\text{acq}}(\cdot)$ denotes a content-aware transformation pipeline during memory acquisition such as *information compression* and *experience consolidation*, which embed the raw observation into a more compact latent space. $D(\cdot, \cdot)$ measures semantic divergence between the compressed and original representations, while $\mathcal{C}_{\text{acq}}(\cdot)$ penalizes complexity (e.g., length and entropy). This formalism highlights acquisition not just as data ingestion, but as the agent's first act of judgment, deciding what to remember and what to forget.

On the one hand, 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 [351]. The goal is rapid, lossy compression to prioritize potentially relevant information. For example, LMAgent [284] prompts the LLM to perform information compression, reducing irrelevant and unimportant content when constructing sensory memory to enhance operational efficiency. Meanwhile, ReadAgent [285] and GraphRead [352] 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 [353]. To enhance the dynamic consolidation of memory-based experiences, Hou et al. [289] define metrics such as contextual relevance and recall frequency to determine whether to update long-term memory in a vector database. Expel [96] constructs an experience pool to collect and extract insights from training tasks, facilitating generalization to unseen tasks. More recently, MindOS [287] proposed a working memory-centric central processing module for building autonomous intelligent agents, where working memory consolidates task-relevant experiences into structured thoughts for guiding future decisions and actions.

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. Similar to the selective attention in human cognitive processes [365], 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 features, compress them compactly, and integrate information from multiple modalities.

We frame encoding as the transformation of the preliminary sketch \tilde{o}_t (produced during acquisition) into a structured latent representation z_t that is stored in memory $\mathcal{M}_t^{\text{mem}}$. This transformation is defined by a learnable mapping $f_{\text{enc}}(\cdot)$ that refines and abstracts relevant features:

$$\mathcal{M}_t^{\text{mem}} \leftarrow \text{Encode}(\tilde{o}_t) = \arg \min_{z_t=f_{\text{enc}}(\tilde{o}_t)} \mathcal{L}_{\text{enc}}(z_t, \mathcal{G}_t) + \lambda_{\text{enc}} \cdot \mathcal{C}_{\text{enc}}(z_t) \quad (3.6)$$

Here, $\mathcal{L}_{\text{enc}}(z_t, \mathcal{G}_t)$ quantifies how well the encoded representation z_t preserves task-relevant signals with respect to an explicit or implicit goal \mathcal{G}_t (e.g., downstream goals, retrieved queries, or plan supervision). The regularizer $\mathcal{C}_{\text{enc}}(z_t)$ penalizes encoding complexity, such as redundancy, representational size, or misalignment across modalities. The encoding function $f_{\text{enc}}(\cdot)$ serves as a unified abstraction instantiated by concrete techniques such as *selective attention* and *multimodal fusion*, as detailed below.

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

Method	Domain	Memory Representation			Memory Lifecycle					
		Sensory	Short-term	Long-term	Acquisition	Encoding	Derivation	Retrieval	Utilization	
Synapse [347]	GUI	Multi-modal	Context	Episodic, Procedural	User demo.	-	Hierarch. Decom.	-	-	
Agent S [265]	GUI	Multi-modal	Context, Working	Semantic, Episodic	Info. Compress.	Contrastive Learn.	Select. Forget.	Indexing	Long-context	
Automanual [133]	GUI	Multi-modal	Context	Procedural, Episodic	User Demo.	Hierarch. Parse	Goal Decom.	Task Search	Subgoal Exec.	
AutoGuide [354]	GUI	Multi-modal	Context	-	Screen Capture	-	Action Plan	-	Action Exec.	
Agent-Pro [355]	GUI	Multi-modal	Context	-	Screen Capture	-	Hierarch. Decom.	-	Action Exec.	
MemGPT [268]	Document	Text	Context, Working	-	External Data	-	-	Paging, Func. call	Doc. interact.	
SeeAct [356]	Web	Multi-modal	Context	-	Screen Capture	-	Action Plan	-	Web Interact.	
AutoWebGLM [357]	Web	Text	Context	-	HTML Parse	HTML Embed	HTML Analysis	-	Web Interact.	
SteP [358]	Web	Text	Context	Task-spec.	HTML Parse	HTML Embed	HTML Analysis	Element Rank	Web Interact.	
AWM [359]	Web	Text	-	Procedural	Workflow Extract.	Action Summ.	-	Sim. lookup	Workflow exec.	
AriGraph [275]	TextWorld	Text	-	Semantic, Episodic	Env. Observ.	Knowl. Graph	Graph Traversal	Assoc. Retrieval	Action plan.	
MemoryBank [261]	Dialogue	Text	-	Episodic	Dialogue Record	-	-	Chron. order	Response	
PromptAgent [360]	General	Text	Context	-	Prompting	-	Prompt Refine.	Content-based	Prompt Exec.	
ECL [361]	Embody	Multi-modal	Context	Episodic	Obs. Recording	Contrast. Learn.	Exper. Summ.	Sim. & Recency	Policy Learn.	
LEO [362]	Embody	Multi-modal	Working	Long-Horizon Rep.	Observation	Spatial-Temp. Learn.	Goal-Cond. Policy	Hierarch. Plan	Long-Horizon Exec.	
IER [363]	Embody	Multi-modal	Context	Episodic	Env. Interact.	Multi-modal Embed	Iter. Refine.	Sim. Match	Action Plan.	
Voyager [74]	Embody	Text	Working	Procedural	Auto. Curriculum	Skill Library	Iter. Prompt.	-	Skill Exec.	
A3T [76]	Embody, Robotics	Text	Context	-	Task Decom.	Token. & Embed.	Action Planning	-	Action select.	
STARLING [364]	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.

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 [291] 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 [292] utilizes graph-based memory to enable selective and multi-turn video scene understanding, enhancing question-answering performance. In robot control, Ali et al. [294] 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 [366] 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 [282] uses the general-domain video-language model CLIP [77] 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 [295] refines memory representation

and optimizes the multimodal encoder by leveraging MineCLIP [367], 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.

3.4.3 Memory Derivation

Once information is encoded, human cognition often revisits and reorganizes it through reflection, inference, and consolidation. Similarly, in intelligent agents, memory derivation refines stored content into more structured, actionable knowledge. This stage mimics human metacognition: identifying core ideas, summarizing past events, extracting general rules, or pruning irrelevant associations. Specifically, this process goes beyond simple storage and is essential for enhancing the agent’s learning efficiency and generalization capabilities. Unlike encoding, memory derivation lacks explicit targets and is instead guided by the dynamic evaluation of information utility and structure.

We thus frame derivation as a self-organizing transformation from the current memory $\mathcal{M}_t^{\text{mem}}$ to a refined form $\mathcal{M}_t^{\text{mem}'}$, by minimizing structural inefficiencies and redundancy through metacognitive mechanisms $f_{\text{derive}}(\cdot)$. These mechanisms include *reflection*, *summarization*, *knowledge distillation*, and *selective forgetting*:

$$\mathcal{M}_t^{\text{mem}} \leftarrow \text{Derive}(\mathcal{M}_t^{\text{mem}}) = \arg \min_{\mathcal{M}_t^{\text{mem}'} = f_{\text{derive}}(\mathcal{M}_t^{\text{mem}})} (\lambda_1 \cdot \mathcal{C}_{\text{redundancy}} + \lambda_2 \cdot \mathcal{C}_{\text{staleness}} - \lambda_3 \cdot \mathcal{R}_{\text{structure}}) \quad (3.7)$$

Here, $\mathcal{C}_{\text{redundancy}}$ penalizes duplicated or overlapping entries; $\mathcal{C}_{\text{staleness}}$ accounts for temporal decay or infrequent usage; and $\mathcal{R}_{\text{structure}}$ rewards abstraction, sparsity, and compositional organization in memory.

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 [360]. ExpeL [96] leverages reflection to collect past experiences for generalization to unseen tasks and to support trial-and-error reattempts following failures. R2D2 [297] 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) [299, 368, 300]. For example, [302] introduces a recursive summarization strategy over dialogue history and prior memory to support long-term dialogue memory derivation. Building on this, Healthcare Copilot [301] 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. MEM1 [369] further integrates memory summarization directly into the reasoning process, learning to dynamically condense context into a compact, constantly updated memory representation that supports efficient long-horizon decision-making.

Knowledge distillation [370] 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 [304] 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 [305] 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 [307] 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 [371] 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) [301], usage frequency (infrequently accessed memories are more likely forgotten) [257], and relevance to the current task or context [309]. In more fine-grained forgetting mechanisms, MemoryBank [261] 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 [308] 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

In both humans and intelligent agents, memory is only valuable if it can be accessed precisely when needed. Memory retrieval serves as the mechanism for recalling relevant knowledge and past experiences to support reasoning and problem-solving. In humans, retrieval is often guided by contextual cues, emotional salience, or associative links. Similarly, intelligent agents must extract pertinent information from a large, heterogeneous memory pool—including sensory, short-term, and long-term memories—using techniques such as context-aware semantic matching, dynamic routing, and attention-based filtering. The objective is to efficiently and accurately retrieve memory fragments that can inform the agent’s current decisions, planning processes, and actions.

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 [372]. 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.

We formalize retrieval as a context-aware matching process, where relevant memory fragments $m_i \in \mathcal{M}_t^{\text{mem}}$ are selected based on their similarity to the current context c_t , as measured by a scoring function f_{match} :

$$\text{Retrieve}(\mathcal{M}_t^{\text{mem}}, c_t) = \arg \max_{m_i \in \mathcal{M}_t^{\text{mem}}} f_{\text{match}}(m_i, c_t) - \lambda_{\text{retr}} \cdot \mathcal{C}_{\text{retr}}(m_i) \quad (3.8)$$

Here, $f_{\text{match}}(m_i, c_t)$ quantifies the semantic relevance between a memory fragment and the current context, while $\mathcal{C}_{\text{retr}}(m_i)$ penalizes retrieval cost, such as latency, size, or modality-specific constraints.

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 [373] 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 [374]. 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 [375] or Annoy [376] for similarity search, and graph indexes for structural queries [377], 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 [315]. 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 [378]. 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

Beyond explicit external memory systems—such as vector databases, document indexes, and key-value caches—Neural Memory Networks represent a form of *implicit memory*, where an intelligent agent's experiences are stored directly within the model's parameters. These networks integrate memory seamlessly into the neural architecture by encoding information into the model's weights or activations, effectively transforming the model itself into a dynamic read–write memory medium. This tight coupling provides several advantages: memory access is embedded directly into the decoding process, maintaining constant

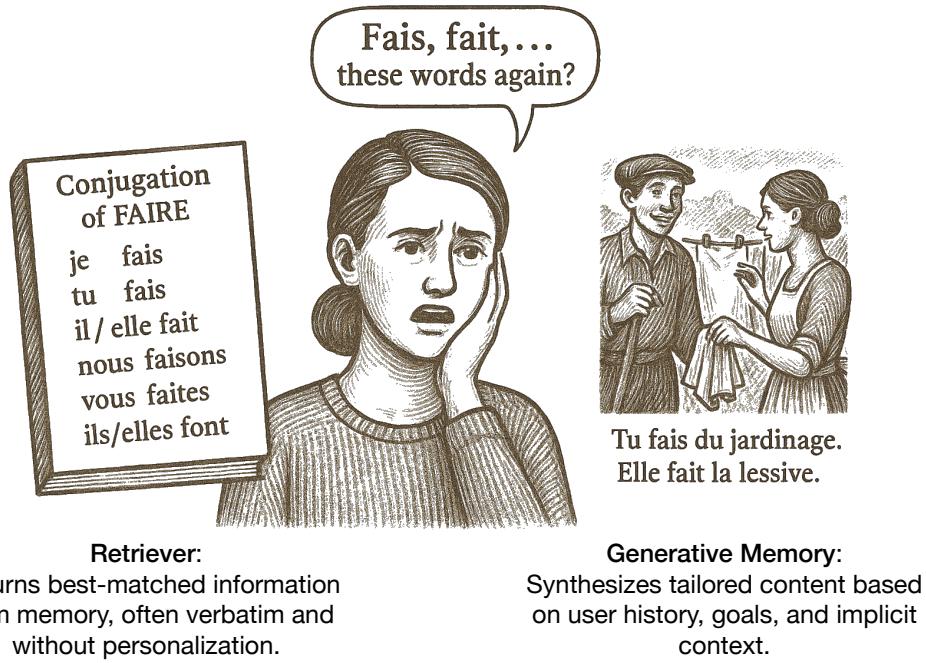


Figure 3.11: Illustration of the contrast between a retrieval-based memory and a generative neural memory. When a learner struggles with subtle verb conjugations, the retriever returns a standard reference from a grammar textbook. In contrast, the generative memory synthesizes a personalized explanation, grounded in the user’s learning history and context, to enhance understanding and engagement.

retrieval cost even as the memory pool grows; and the need for explicit memory management is reduced, simplifying system design.

However, this approach also reveals a fundamental contrast between generative and retrieval-based memory paradigms. As illustrated in Figure 3.11, a retrieval-based system searches for the best-matching item based solely on the input query—akin to a student receiving a textbook excerpt when confused. In contrast, a generative memory system, such as a neural memory network, draws on the user’s past interactions and contextual understanding to produce a more adaptive, personalized response—like a helpful tutor who knows the learner’s strengths, struggles, and goals. The retrieved knowledge is “looked up”, while generative memory reflects “lived” experience embedded in parameters.

Storing information within parameters may enable richer, more flexible associations and generalization, but comes at the cost of reduced transparency and controllability. Unlike external memory, parametric memory resists precise, surgical updates and risks catastrophic forgetting—where new learning overwrites existing knowledge. Additionally, its capacity is ultimately bounded by model size, whereas external stores can scale freely. The trade-off is thus between the fluid, efficient, and integrated nature of parametric memory and the scalable, interpretable, but more rigid character of external retrieval. Designing neural memory networks that resolve this tension remains one of the grand challenges of cognition-inspired AI.

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.

We formalize this as a long-horizon continual learning process that balances the integration of new information with the preservation of existing knowledge, where the model parameters θ_t at time t serve as the memory container. The learning objective jointly optimizes task performance while mitigating interference and memory degradation:

$$\mathcal{M}_t^{\text{mem}} \leftarrow \theta_t = \arg \min_{\theta} \mathcal{L}_{\text{task}}(\theta; \mathcal{T}_t) + \lambda_{\text{stability}} \cdot \mathcal{D}(\theta || \theta_{t-1}) + \lambda_{\text{store}} \cdot \mathcal{C}_{\text{store}}(\theta) \quad (3.9)$$

Here, $\mathcal{L}_{\text{task}}$ promotes performance on the current task \mathcal{T}_t , $\mathcal{D}(\theta || \theta_{t-1})$ penalizes deviation from prior parameters to preserve previously acquired knowledge and avoid catastrophic forgetting, and $\mathcal{C}_{\text{store}}(\theta)$ models the cost of incorporating new information into the network, such as computational overhead or optimization difficulty to training neural memory networks. This formulation captures the central trade-off in neural memory: the tension between *stability and reliability* (retaining past knowledge) versus *plasticity and generalization* (adapting to new experiences over time). Two broad families of methods instantiate this framework: *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 [316, 317], leveraging energy functions, and Bidirectional Associative Memories (BAMs) [379], supporting hetero-associative recall, provide mechanisms for encoding and retrieving patterns based on the weights between neurons. Besides, Neural Turing Machines (NTMs) [318] and Memory-Augmented Neural Network (MANNs) [380, 381, 329, 319] augment neural networks with external memory modules, employing attention and summary mechanisms to interact with these memories.

On the other hand, parameter integration facilitates the seamless integration of world knowledge and accumulated experience into the operational behavior of intelligent agents. For example, some prior works modify model parameters to enable continual learning by updating [382, 383, 384] or forgetting specific knowledge [385]. Other studies treat LLMs as standalone memory modules, incorporating world knowledge into their parameters during pre-training [386], post-training [387], and online deployment [388]. For instance, MemoryLLM [319] introduces memory tokens, while SELF-PARAM [320] leverages knowledge distillation to embed world knowledge and agent's past experiences into model parameters. This approach is further augmented in the M+ model [389] with a long-term memory mechanism and a co-trained retriever, enhancing its ability to generalize to longer history memorization. Additionally, [390] employs encoded memory to facilitate further reasoning, thereby improving the generalization of stored knowledge. More recently, MemoRAG [321] and R³Mem [324] 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 [323] and Dynamic Cheatsheet [391] introduce mechanisms to memorize test-time data points via test-time learning, thereby enhancing the agent's ability to generalize across tasks more efficiently at inference time.

Future research will continue to focus on developing neural memory models with larger capacity and greater stability. At the same time, designing more efficient and flexible memory read-write mechanisms remains a critical challenge. As a step in this direction, MemOS [392] introduces a prototype operating system for memory-augmented generation (MAG), supporting a unified, memory-centric training paradigm that integrates activation-based neural memory, parametric memory, and plaintext memory. It further proposes a three-layer architecture, comprising interface, operation, and infrastructure layers, to orchestrate memory control and enable seamless memory system integration. Looking ahead, a key research frontier will be applying these memory-augmented networks to complex cognitive tasks, thereby pushing the boundaries of

what AI systems can achieve. Advancements in this area will unlock new possibilities for building intelligent agents capable of learning, remembering, and reasoning in ways increasingly aligned with human cognition.

3.4.6 Memory Utilization

Finally, memory utilization refers to applying recalled information to enhance current tasks—reasoning, planning, action, or dialogue. It completes the cycle: from sensory input to structured knowledge to behavioral output. 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)* [393] 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 [328] and adaptive retrieval mechanisms [326] to enhance both generation reliability and efficiency. For example, Atlas [327] leverages causal mediation analysis, while [338] 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 [325] offers a comprehensive ecosystem for evaluating and analyzing mainstream RAG algorithms. HippoRAG [276] 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 [381] and Longformer [394], or through hierarchical and recursive processing techniques, such as recurrent memory transformer (RMT) [329, 330], 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 [331] 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) [332] generates memory slots that accurately and comprehensively represent the original context, while LLMLingua [395, 396], Gist [333], CompAct [334], and HyCo₂ [397] 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 [398], verifying generated content against established knowledge or memory stores. Another involves uncertainty estimation [399, 400], 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 are aligned with the agent’s established knowledge base. Recent research has introduced expert memory subnetworks, such as PEER [337] and Lamini Memory Tuning [335], which specialize in memorizing specific types of information, including world knowledge and 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

HE 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, such as 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. A key tension here remains

the choice between storing knowledge implicitly within model parameters versus explicitly in external caches. Parametric memory offers tight integration and potentially richer associative capabilities, but faces challenges in scalability, editability, and catastrophic forgetting. External caches offer vast, manageable storage but may lack the nuanced integration of parametric approaches. Future systems might need hybrid architectures that leverage the strengths of both. Another crucial area is developing memory systems that actively “curate” their contents, i.e., 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 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

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.

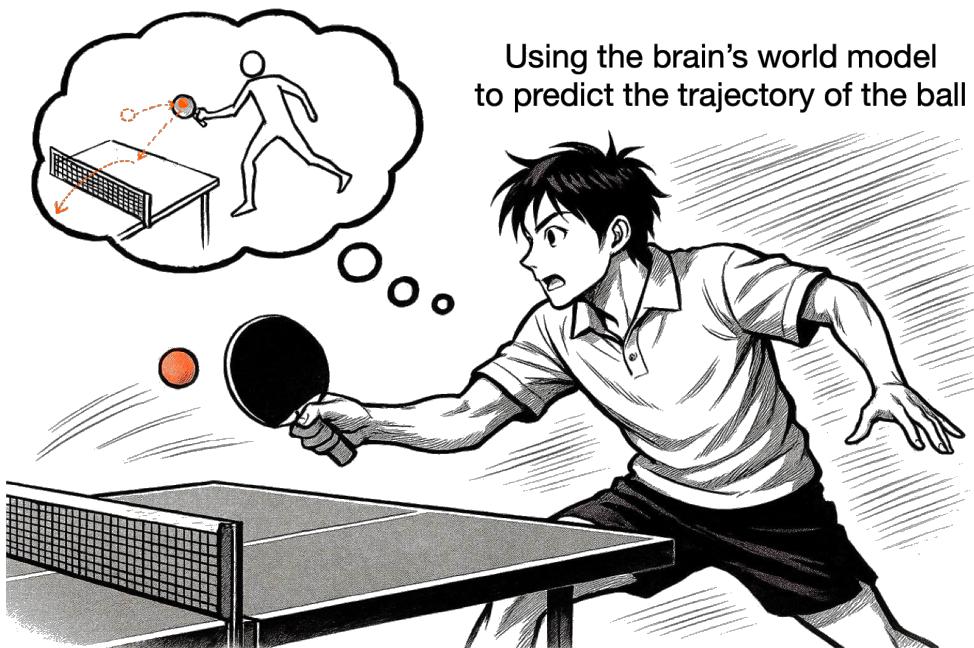


Figure 4.1: Humans can use their brain’s model of the world to predict the consequences of their actions. For example, when playing table tennis, a player can imagine or predict the trajectory of the ball after an action.

4.1 Mental Models as Human World Representations

 HUMANS naturally construct internal representations of the world, often referred to as *mental models* in psychology [401, 402, 403]. 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 [401], 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” [402] to simulate how events might unfold and evaluate possible courses of action. This notion, often referred to as a *mental model*, closely parallels what is now called a *world model* in AI contexts, wherein an internal mechanism supports simulation and prediction without direct environmental interaction. Later studies proposed that such simulations span modalities such as 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 [403]. More recent predictive processing theories such as “Surfing Uncertainty” [404] 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 enables reasoning across both fine-grained and coarse-grained temporal or spatial contexts, depending on task demands.

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 are capable of counterfactual simulation, such as declining dessert now while predicting future enjoyment, which enables planning across hypothetical scenarios and time horizons.

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 From Human Mental Models to AI World Models

RESEARCH in artificial intelligence has long sought to replicate the *predictive, integrative, and adaptive* qualities exhibited by human mental models [401, 402]. Early reinforcement learning frameworks, for instance, proposed learning an *environment model* for planning, exemplified by Dyna [405], while contemporaneous work investigated using neural networks to anticipate future observations in streaming data [406, 407]. 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” [38]), 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 $P(s' | s, a)$ and plan with approximate lookahead [408, 409].

Another branch of work leveraged large-scale simulators or real-world robotics to ground learning in richly diverse experiences [410, 411]. 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 [132, 101] 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. To clarify our classification, we define the distinction between *implicit* and *explicit* paradigms based on whether a system performs *end-to-end reconstruction of observations* during planning or simulation. Systems that learn to generate or predict future high-dimensional observations, such as video frames or sensor readings, as part of their rollout process are considered *explicit*. In contrast, models that operate entirely in latent or abstract internal spaces, even if they contain auxiliary decoders for training or visualization, are classified as *implicit* under this definition. This view aligns with how reconstruction capacity contributes to interpretability and groundedness in world modeling.

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*. For example, World Models[38] relies primarily on internal neural networks for dynamics learning, while SAPIEN[410] uses external physics engines with predefined rules. Similarly, simple methods like ActRef[76] uses minimal components, whereas DayDreamer[411] integrates multiple neural modules for state prediction, value estimation, and policy learning. Figure 4.2 illustrates this two-dimensional space, showing how different approaches distribute themselves across these axes.

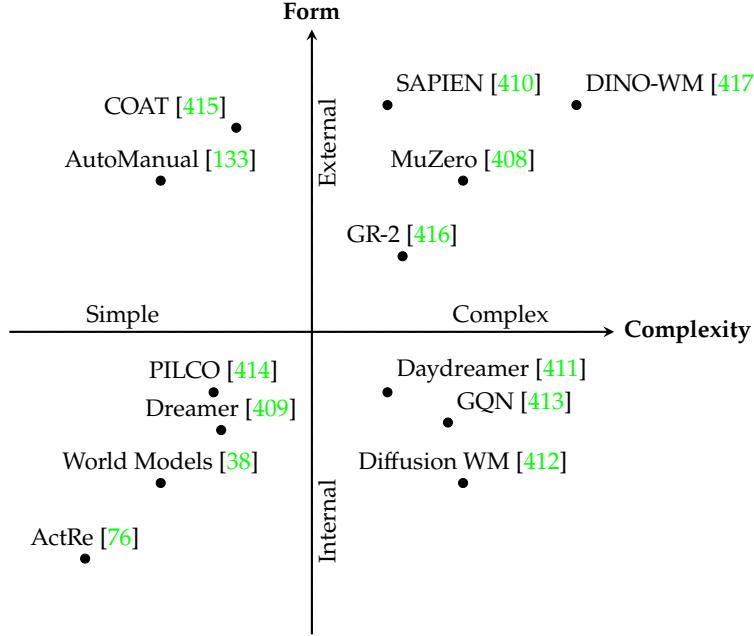


Figure 4.2: A two-dimensional layout of AI world-model methods. The horizontal axis indicates *Complexity* (left to right). The vertical axis spans *Internal* approaches (bottom) to *External* solutions (top). Approximate positions reflect each method’s reliance on large learned networks vs. explicit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

Table 4.1: Design matrix of representative world-model methods. Columns list the encoder/decoder choice (Enc/Dec), latent representation (Latent), rollout strategy (Rollout), planning method (Plan), and training objective (Obj).

Method	Enc	Dec	Latent	Rollout	Planning	Obj
World Models [38]	VAE	VAE	$z+h$	MDNRNN	CMAES	Recon+KL
MuZero [408]	RepNet	None	State	LatentTree	MCTS	TD+CE
Dreamer [409]	RSSM	Image/Reward	RSSM	Imagination	ActorCritic	ELBO+RL
DINO-WM [417]	DINOv2	(Viz)	Features	FeatPred	CEM	MSE
GR-2 [416]	Tokenizer	VQGAN/Act	Tokens	Autoreg	Sampling	CE
AutoManual [133]	LLM	LLM	Rules	Iterate	Manual	Success
ActRe [76]	LLM	LLM	Rationales	Forced Sampling	Direct Policy	CE
V-JEPA2 [418]	JEPA	None	Embeds	EmbedPred	MPC+CEM	FeatPred
Genie 2 [419]	AE	Latent Diffusion	Video	AR	None	Denoise
RoboDreamer [420]	Parser/T5	U-Net	CompLat	Video Plan	InvDyn	Denoise

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:

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

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

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:** During planning the agent rolls out a *latent dynamics* function¹ and makes all decisions from the latent state alone. If a decoder g_θ exists, it is used only for auxiliary losses or visualisation, *never* for action selection. Typical examples include World Models [38], MuZero [408], Dreamer [409], and token-based generalist agents such as Gato [421]. The approach yields fast, end-to-end training but offers limited interpretability or direct constraint injection.
- **Explicit paradigm:** Here the rollout *explicitly predicts future observations* (pixels, point clouds, sensor streams) so that imagined trajectories remain in the same modality as real perception. In practice this often means a factored pair $\hat{s}_{t+1} = T_\theta(s_t, a_t)$ and $\hat{o}_{t+1} = O_\theta(\hat{s}_{t+1})$, though diffusion-based models can generate observations end-to-end. Visual generative models such as Diffusion WM [422] and DINO-WM [417] fall into this category, offering greater debuggability and the ability to inject structural priors at the cost of heavier computation.
- **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 [410] or real-robot pipelines [411] 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 [133], refine implicit LLM knowledge into partial causal graphs [415], 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

We define an *implicit world model* as one in which the agent simulates future dynamics entirely within an internal latent space, without explicitly reconstructing future high-dimensional observations during inference. Even if a decoder exists, for visualization or auxiliary training, it is not used during planning or imagined trajectories. Instead, the system evolves latent representations that are sufficient for decision-making or value prediction. Formally, an implicit world model maintains a latent state $h_t \in \mathcal{H}$, updated as:

$$h_{t+1} = f_\theta(h_t, a_t), \quad \hat{o}_{t+1} = g_\theta(h_{t+1}) \text{ (optional, not used in rollout).} \quad (4.3)$$

A representative example of the implicit paradigm is the *World Models* framework [38], in which a Variational Autoencoder (VAE) is first trained to encode high-dimensional visual observations into a compressed latent space. Separately, a recurrent network is trained to model the dynamics in this latent space by predicting future latent vectors. While a decoder exists for reconstructing images from latent codes, it is used only during the VAE pretraining or for visualization purposes, not as part of the agent’s rollout or decision-making process. During planning, the agent performs simulated rollouts entirely within the latent space using the RNN, without reconstructing future observations. Accordingly, under our definition, World Models is categorized as an *implicit* world model, since its predictions do not include explicit reconstruction of future observations during imagined trajectories.

A more complex example is MuZero [408], which combines a learned latent dynamics model with Monte Carlo Tree Search (MCTS) to plan over imagined futures. Although MuZero achieves high performance

¹Formally $h_{t+1} = f_\theta(h_t, a_t)$ with value / policy heads on h_{t+1} .

in challenging domains such as Go and Atari, it never reconstructs the actual observations (e.g., game board states or video frames) at any point in its planning process. Instead, the model focuses exclusively on predicting reward, value, and policy over abstract latent states. These latent states are not trained to correspond to any observable quantity in the environment, and the learned dynamics function is optimized solely for planning efficacy. Thus, MuZero squarely fits within the implicit paradigm, despite its powerful planning mechanism, as it bypasses observation-level modeling entirely. Model-based reinforcement-learning algorithms like Dreamer [409] exemplify this paradigm by performing all planning in latent space, using observation reconstruction only for auxiliary training losses.

Recent large-scale systems further illustrate the *implicit* design philosophy. Gato [421] demonstrates that a single token-based model can act in dozens of tasks, from Atari to robotic manipulation, by rolling out purely in latent space. Similarly, V-JEPA 2 [418] pre-trains on internet-scale video and later finetunes a latent action head, enabling zero-shot robotic planning without ever reconstructing video frames during inference. These results strengthen the case that end-to-end token world models can generalize broadly while remaining computationally simple.

Because *implicit* world models simulate future dynamics entirely inside a latent space, they never have to **reconstruct high-dimensional observations during planning**. A single latent-transition function f_θ is unrolled, and, if present at all, an observation decoder g_θ is used only for auxiliary losses or visualization, *not* for decision making. This design yields elegant end-to-end training pipelines and fast rollouts, letting large-capacity networks autonomously discover compact structure in complex environments. The trade-off is opacity: latent variables are not directly interpretable, it is difficult to inject domain priors or constraints, and the model can be brittle under distribution shift because errors in latent prediction may go undetected by perceptual metrics. Thus, while the implicit paradigm excels in simplicity and computational efficiency, it remains challenging when users require strong interpretability, explicit safety constraints, or fine-grained control over learned dynamics.

4.3.3 Explicit Paradigm

We define an *explicit world model* as one that reconstructs or predicts future high-dimensional observations (e.g., images or sensor data) during inference. These imagined observations are actively used for training or decision-making. Such models typically factorize the transition and observation processes into two functions:

$$\hat{s}_{t+1} = T_\theta(s_t, a_t), \quad (4.4)$$

$$\hat{o}_{t+1} = O_\theta(\hat{s}_{t+1}). \quad (4.5)$$

The predicted observation \hat{o}_{t+1} is then used to inform future actions or value estimates. This makes the world model's predictions interpretable and aligned with the agent's sensory domain. Examples include Dreamer [409], Diffusion WM [412], and DINO-WM [417].

This capability to reconstruct future observations during imagined rollouts distinguishes explicit models from implicit ones. While both may use latent representations internally, explicit models treat accurate observation prediction as central to the simulation process. This design encourages interpretability, supports visual debugging, and facilitates interaction with tasks that require fine-grained perceptual feedback.

Explicit approaches prioritize fidelity in generating future frames for decision-making, such as Diffusion WM [412], which applies diffusion processes at the pixel level, or DINO-WM [417], which rolls out future states within a pretrained feature space.

Beyond pixel-CNN or VAE backbones, a line of diffusion-based world models has emerged. Diffusion World Model(DWM) [422] predicts long-horizon trajectories and their rewards in a single denoising pass, demonstrating significant improvements on D4RL benchmarks. AVID [423] shows that

closed-source video diffusion generators can be adapted into fully action-conditioned world models through lightweight adapters, retaining photorealistic quality. Complementing these, PhysDreamer [424] generates plausible physics-based human–object interaction videos, answering “what-if” queries in 3-D scenes. Together, these works underscore how explicit frame-level generation is becoming both higher-fidelity and more computationally tractable.

Autonomous-driving research has embraced explicit world models that output multi-view video. GAIA-1 [425] conditions on past camera feeds, text, and control to generate diverse, controllable futures—supporting counterfactual scene editing. DriveDreamer [426] is the first model trained purely on real-world driving logs; a two-stage diffusion pipeline captures fine-grained traffic constraints and yields accurate frame-level rollouts. Drive-WM [427] further blends visual forecasting with image-reward trajectory optimisation, integrating perception, prediction, and planning into one generative module.

By reconstructing future observations during imagined rollouts, explicit models expose the full prediction to the user, making internal failure modes observable and enabling the injection of domain priors (e.g., physical constraints, safety rules) at the pixel or point-cloud level. This transparency facilitates targeted debugging and opens doors to hybrid schemes such as loss weighting on physics invariants or hard-coded collision checkers. Yet the same design magnifies compounding-error risk: any drift in the learned transition \hat{T}_θ or decoder \hat{O}_θ is immediately visible in the rendered trajectory and can poison multi-step planning. Scaling to long horizons therefore requires either (i) very accurate models, (ii) frequent real-world replanning, or (iii) specialised corrections such as diffusion-based denoising or hindsight relabelling. Moreover, explicit rollouts incur higher compute and memory costs because every imagined frame (or point cloud) must be produced, stored, and evaluated. Consequently, explicit world models excel when interpretability and hard constraints outweigh raw simulation speed, but they remain challenging in data-scarce regimes and under severe distribution shift.

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}_θ from data. Formally,

$$(s_{t+1}, o_{t+1}) \leftarrow \text{SIM}(s_t, a_t), \quad (4.6)$$

where SIM is often an external physics engine or the real world itself. Platforms like SAPIEN [410] and AI Habitat provide deterministic 3D physics simulations, allowing agents to practice or iterate strategies in a controlled environment. Alternatively, methods such as Daydreamer [411] treat real-world interaction loops like a “simulator”, continually updating on-policy data from physical robots.

Tokenised game engines can also serve as high-fidelity simulators. MineWorld [428] converts Minecraft video frames and actions into discrete tokens and trains a parallel Transformer that renders 4–7 fps in real time, offering an open-source, interactive sandbox where agents can query a “perfect” simulator instead of learning dynamics from scratch.

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 [133], for example, iteratively compiles interactive environment rules into human-readable manuals, informing future actions in a more transparent way. Meanwhile, COAT [415] 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.

Recent work pushes the hybrid idea further by letting large language models write rules, code, or even 3-D scenes on the fly. WorldCoder [429] turns environment feedback into executable Python snippets, yielding explicit, editable simulators that the agent can re-compile between episodes. Genie 2 [419] expands a single image into an interactive 3-D world and maintains memory of occluded objects, effectively using text prompts as a high-level world-generation interface. RoboDreamer [420] parses human instructions into low-level primitives and trains a video generator conditioned on those primitives, giving robots compositional imagination. Finally, WMA [174] tokenises HTML and JavaScript events to learn latent dynamics of web pages, enabling zero-shot navigation by rolling out the model in token space. Together these systems illustrate how language, code, and generative priors can merge with latent dynamics to create flexible instruction-driven world models.

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

Table 4.2: Summary of AI world-model methods across paradigms, showing their form (External \circ or Internal \bullet) , complexity, and paradigm.

Method	Form	Complexity	Paradigm
ActRe [76]	\bullet	Simple	Implicit
World Models [38]	\bullet	Simple	Implicit
Dreamer [409]	\bullet	Moderate	Implicit
Diffusion WM [412]	\bullet	High	Explicit
GQN [413]	\bullet	High	Explicit
Daydreamer [411]	\bullet	High	Simulator-based
SAPIEN [410]	\circ	High	Simulator-based
PILCO [414]	\bullet	Moderate	Implicit
AutoManual [133]	\circ	Simple	Other
MuZero [408]	\circ	High	Implicit
GR-2 [416]	\bullet	High	Explicit
DINO-WM [417]	\bullet	High	Explicit
COAT [415]	\circ	Moderate	Other

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 \circ for external approaches and \bullet for internal ones, with mixed methods having both symbols. This

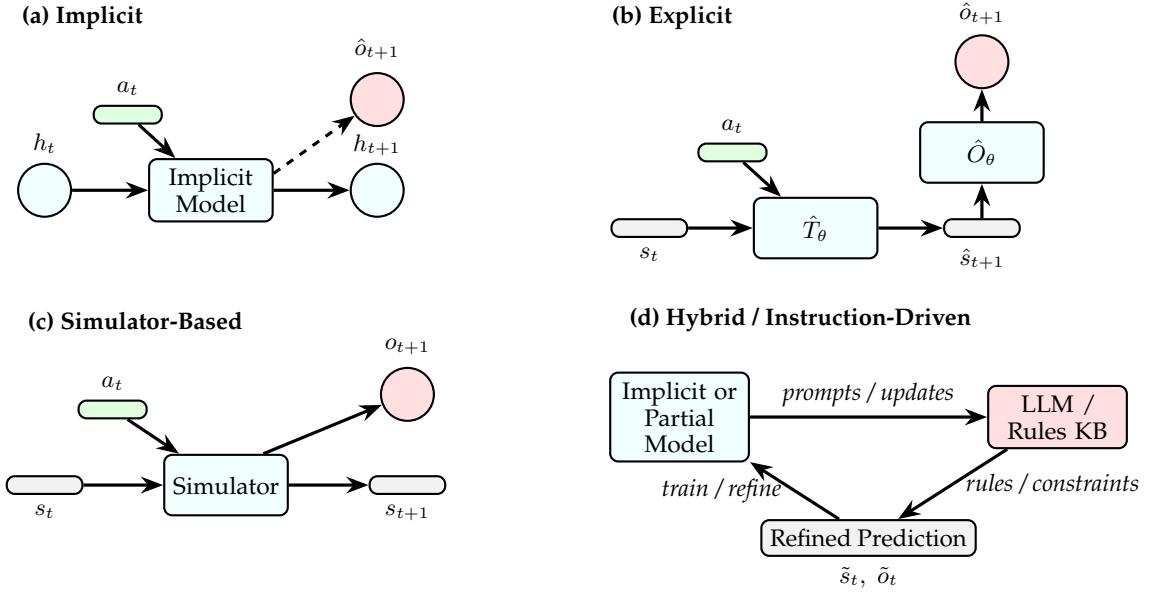


Figure 4.3: Four paradigms of world modeling: (a) implicit, (b) explicit, (c) simulator-based, and (d) hybrid/instruction-driven. These diagrams illustrate the architectural forms of world models under different design paradigms.

classification aligns with the previous subsections, including the detailed discussion of each paradigm, and complements the visual representation in Figure 4.2. Table 4.1 further details representative methods along encoder/decoder, latent space, rollout, planning, and objective, complementing the paradigm-level summary (form, complexity, paradigm) above.

4.4 Relationships to Other Modules

 I WORLD MODEL does not exist in isolation but interacts with several key components of the agent’s architecture. These include (but are 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.

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 [409], 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 [430] achieves this perception-to-modeling transformation by converting visual observations into particles and capturing the underlying system structure through graph neural networks. PointNet [431] 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 [432] 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 [408, 433, 132], 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 [434, 409, 435]. These simulations enable the agent to evaluate the potential benefits of exploring new parts of the state space.

In model-based planning, MuZero [408] 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 [433] 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.

World-model agents now drive exploration by imagining what lies just beyond the data they have seen. Dreamer [409] rolls out thousands of latent trajectories per real action and rewards those that reach novel latent states, giving safe yet diverse curiosity without physical risk. Dreamer V2 [435] switches to a discrete latent space that represents uncertainty more directly, accelerating exploration on Atari and continuous-control tasks. Dreamer V3 [436] extends the idea to long-horizon domains such as Minecraft, where adaptive curiosity bonuses in imagination enable zero-shot completion of difficult objectives.

In reinforcement learning generally, a learned world model allows the agent to score many hypothetical trajectories before acting in the real environment, compare their predicted returns, and pick actions that maximise long-term goals while respecting any safety or cost constraints encoded in simulation.

World models ultimately matter because they let an agent choose actions before the real world can punish mistakes. Classical pipelines query the model with sampling or tree search and keep only the first action of the best simulated trajectory. Newer approaches fold learning and planning together: Dual Preference Optimisation (D²PO) [437] ranks imagined rollouts by human feedback so that both the model and the policy improve toward what users actually care about, while RoboDreamer [420] grounds a language goal into low-level action tokens and selects the token sequence whose predicted video has the highest success probability. These results illustrate a trend toward tighter coupling of model, objective, and planner rather than treating planning as a separate post-hoc module.

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 can easily transition 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



HE 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) [438]. 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, 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 [408] and Dreamer [409] exemplify this: they excel at specific environments (e.g., Atari games or robotics) but require careful adaptation when transferred to new, uncharted domains. In contrast, implicit models, particularly those that use large-scale neural networks, have the potential to generalize between 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, such as those used in AutoManual [133] or discovery-based models [415], 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.

Several recent trends point to a convergence of scale, modality, and causality. Dreamer V3 [436] shows that a single configuration can surpass specialised baselines on more than 150 tasks and even solve the long-horizon Minecraft diamond-mining benchmark without additional tuning. Large token models such as Gato [421] and V-JEPA 2 [418] highlight that unifying vision, language, and control into one latent predictor yields broad transfer with minimal task-specific code. On the theory side, Robust Agents Learn Causal World Models [439] and General Agents Need World Models [440] formalise why agents that act robustly must internalise causal structure, providing an analytical backbone to the empirical successes above. These developments suggest that scaling alone is not enough; explicitly aligning predictive latent space with causal factors and preference signals may be the key to truly general agents.

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

 REWARD lies at the heart of both biological and artificial intelligence. In humans and animals, reward mechanisms govern decision-making, drive motivation, reinforce behavior, and shape the learning process through complex neurochemical pathways. In AI agents, reward functions serve as the foundational signals that guide behavior optimization, enabling systems to learn from experience, adapt to goals, and make intelligent choices in dynamic environments. Reward is the connective tissue between goals and learning: it operationalizes abstract values into measurable feedback, enabling both humans and machines to adapt and improve their behavior over time. Understanding reward is therefore not only central to the study of cognition and behavior but also critical for the development of robust, generalizable, and aligned AI systems. This chapter explores the multifaceted concept of reward, bridging its biological underpinnings in the human brain with its algorithmic formulations in AI agents. We begin by examining the human reward system, highlighting key neurotransmitters and anatomical pathways that support reinforcement and motivation. We then introduce the design space of agent reward models, discussing how different types of reward—extrinsic, intrinsic, hybrid, and hierarchical—contribute to intelligent behavior. Finally, we analyze the interaction between reward and other core modules such as perception, memory, and emotion, and discuss open challenges such as reward sparsity, reward hacking, and reward misspecification. By integrating insights from neuroscience and AI, this chapter provides a comprehensive foundation for understanding reward as a central principle of intelligent systems.

5.1 The Human Reward Pathway

 HE 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 [519]. 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 [527] 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 [528]. This neurotransmitter is critical for reward-based learning, but this function can be disrupted in many psychiatric conditions, such as mood disorders and addiction.

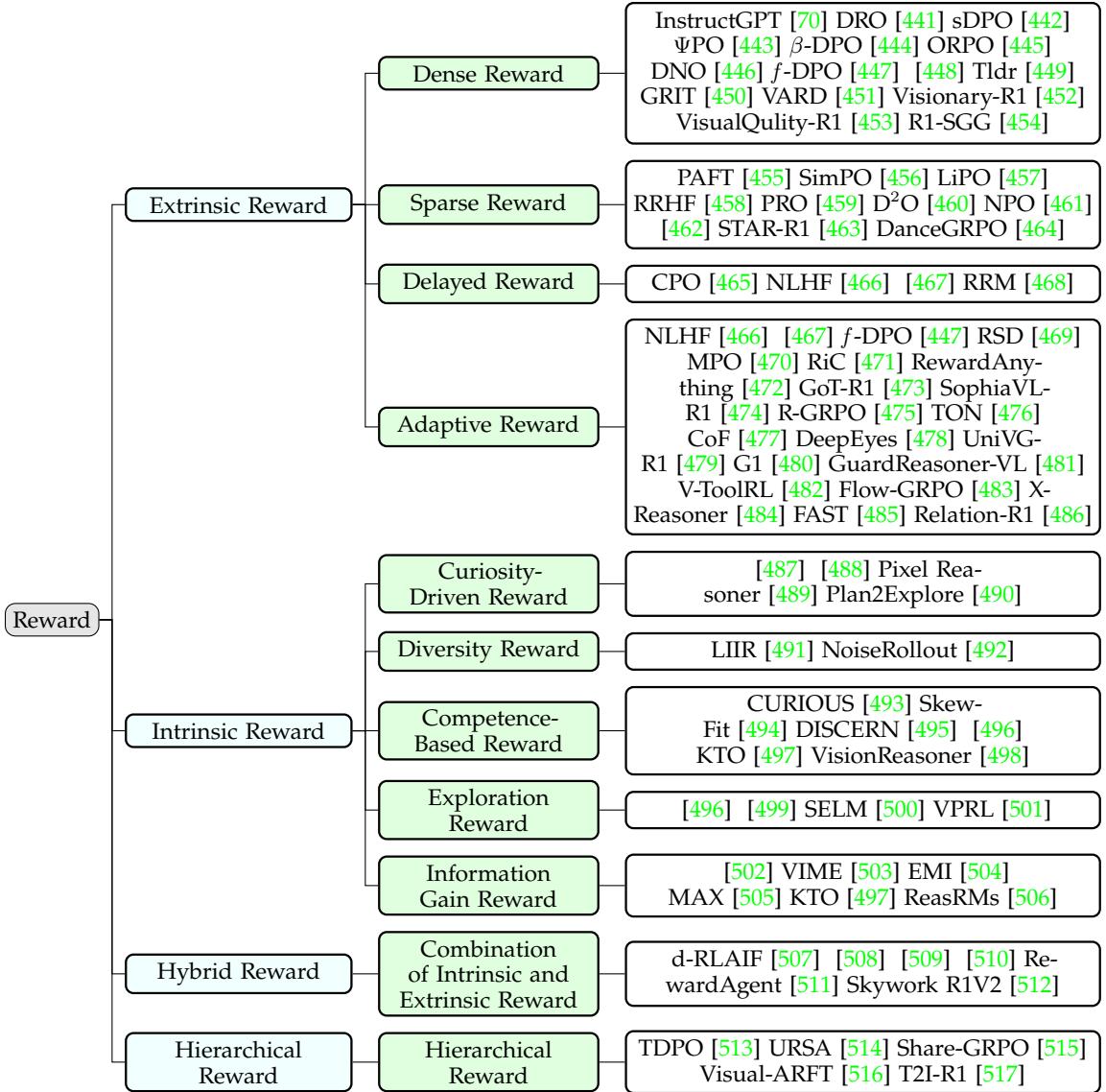


Figure 5.1: A taxonomy of selected research works about reward systems.

The *mesolimbic pathway* [518], 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 [529]. Table 5 lists the common reward pathways in the human brain, the neurotransmitters they transmit, and the corresponding

Table 5.1: The comparison of human common reward pathways.

Reward Pathway	Neurotransmitter	Mechanism
Mesolimbic pathway [518]	Dopamine	Dopamine neurons in the VTA project to the nucleus accumbens, releasing dopamine that binds to D1 (excitatory) and D2 (inhibitory) receptors, modulating motivation, reinforcement, and reward-seeking. <i>Primary circuit for encoding reward prediction and reinforcing behaviors.</i>
Mesocortical pathway [519]	Dopamine	VTA neurons project to the PFC, where dopamine influences receptors involved in cognition, emotion, and reward anticipation. <i>Supports higher-order evaluation of reward outcomes and decision-making.</i>
Nigrostriatal pathway [519]	Dopamine	Dopamine acts on D1 and D2 receptors in the striatum to regulate motor activity and learned reward-related behaviors. <i>Links reward processing with motor planning and habit formation.</i>
Locus coeruleus [520]	Norepinephrine	Locus coeruleus neurons release norepinephrine across the brain, activating α and β adrenergic receptors to enhance arousal, attention, and stress responses. <i>Modulates reward salience and attentional engagement.</i>
Glutamatergic projection [521]	Glutamate	Glutamate binds to AMPA, NMDA, and metabotropic receptors, driving excitatory signaling and plasticity in reward circuits. <i>Crucial for learning reward associations and reinforcing adaptive behavior.</i>
GABAergic modulation [522]	Gamma-Aminobutyric Acid (GABA)	GABA binds to GABAA and GABAB receptors to inhibit postsynaptic activity, maintaining balance in reward pathways. <i>Provides inhibitory control over excitatory reward signaling to prevent overstimulation.</i>

mechanisms of action, describing the basic framework of the human brain reward system. Meanwhile, Figure 5.2 vividly describes these reward pathways.

5.2 From Human Rewards to Agent Rewards

AVING established how reward circuits shape behavior in the human brain, we now explore how similar principles inspire—but do not constrain—the design of reward mechanisms in AI agents. Biological systems rely on complex neurochemical and psychological feedback loops, while AI agents operate using formalized reward functions designed to guide learning and decision making. Although inspired by human cognition, the reward mechanisms of AI agents are structurally and functionally distinct. Understanding the analogies and disanalogies between these systems is crucial to align 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, AI 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 [530, 531]. Neuroscience shows that novelty, surprise, and information gain activate many of the same circuits as extrinsic rewards [532]. Similarly, intrinsic motivation modules (e.g., curiosity bonuses, empowerment, and count-based exploration) provide shaping signals that accelerate agent learning in sparse reward environments. However, absent embodiment and affect, agents lack the hedonic valence that

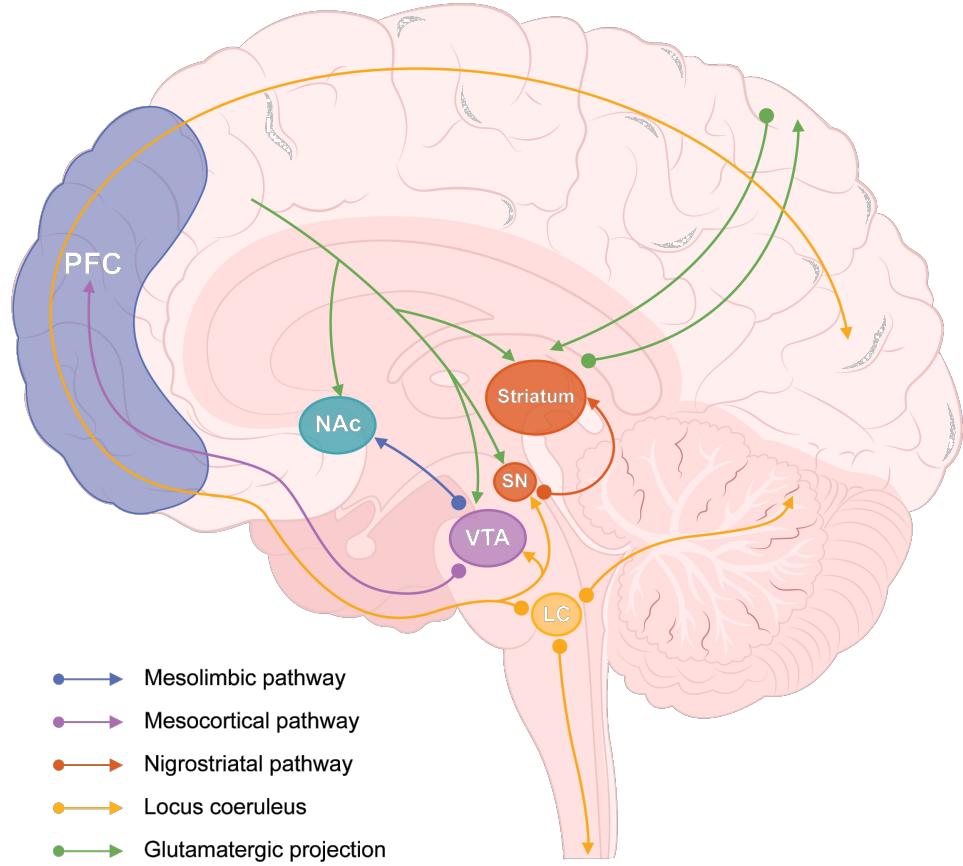


Figure 5.2: The diagram of common reward pathways in the human brain. The structures include VTA (Ventral Tegmental Area), NAc (Nucleus Accumbens), PFC (Prefrontal Cortex), LC (Locus Coeruleus), and SN (Substantia Nigra). The pathways are inspired by these works [523, 524, 525, 526].

unifies intrinsic and extrinsic motives in humans, raising questions about long-term stability and alignment of exploration incentives.

One key distinction lies in the *programmability* and *plasticity* of the rewards of agents. Unlike human reward systems, which are constrained by biological architecture and evolutionary inertia, agents' 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, while agents' rewards tend to be explicit and task-specific. Agents lack emotional intuition and instinctual drives; their learning is entirely dependent on the form and fidelity of the reward signal. Although frameworks such as reinforcement learning from human feedback (RLHF) attempt to bridge this gap by using preference data to shape agent behavior [12], such methods still struggle to capture the full complexity of human goals, especially when preferences are intransitive, cyclical, or context-sensitive [378].

Moreover, attempts to borrow from human reward mechanisms, such as modeling intrinsic motivation or social approval, still face limitations due to the absence of consciousness, embodiment, and subjective experience in AI 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 dive 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. Figure 5 shows an overview of the relevant research works.

5.3 AI Reward Paradigms

 REWARDS exist in intelligent agents, especially in reinforcement learning scenarios. We can picture rewards as breadcrumbs guiding an agent through its world. In reinforcement learning they're the quick nods or gentle shakes of the head that follow every move, saying "nice choice" or "try something different". Each crumb judges one action in one moment, but a long string of them weaves a story the agent can read. After countless stumbles, lucky breaks, and course corrections, the agent zeroes in on strategies that bring those encouraging crumbs more often and more reliably.

In reinforcement learning, the reward model hands out the pats on the back—or the quiet frowns—that steer each step an agent takes. One signal per move, it scores how wise that action was in the moment. Armed with that number, the policy tilts, edging the next decision a little higher on the scoreboard. Figure 5.3 illustrates this feedback-driven interaction loop between agent and environment in the Markov Decision Process (MDP) setting.

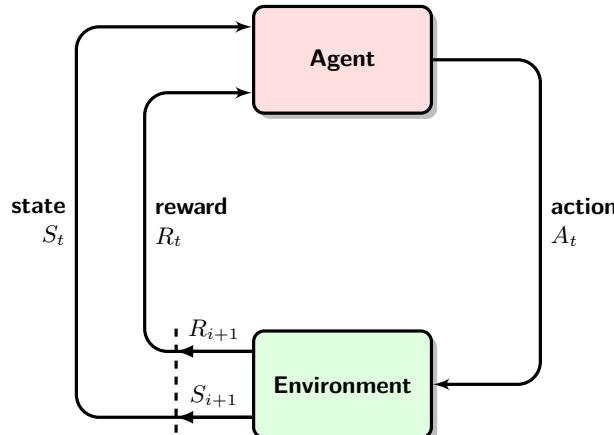


Figure 5.3: The interaction loop between agent and environment in a Markov Decision Process. The environment hands over a snapshot S_t and reward R_t . The agent replies with an action A_t to influence future states and rewards. This feedback loop forms the foundation of learning via rewards.

Definition 7 (Reward in Markov Decision Process). The agent's interaction with its environment can be captured by the Markov Decision Process (MDP) framework [533], formally written as:

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

where:

- \mathcal{S} denotes the state space that encompasses all possible states in the environment.
- \mathcal{A} denotes the action space that encompasses all actions available to the agent at each state.
- $P(s'|s, a)$ defines the state transition probability. It represents the probability that choosing action a in state s will toss the agent into the follow-up state s' .

- $r(s, a)$ specifies the reward function, which assigns an immediate scalar reward received by the agent for the choice a in state s .
- $\gamma \in [0, 1]$ is the discount factor, which controls the agent's preference for immediate versus future rewards by weighting the contribution of future rewards to the overall return.

At the center of the agent's feedback loop is the function $r(s, a)$. Formally,

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

$r(s, a)$ hands back one number for every move. It looks at where the agent stands, notices the action just taken, and offers a quick tip—positive if the step helps, negative if it hurts. Collect enough of those little signals and the agent zeros in on choices that keep the scoreboard climbing in each situation.

Objective of the Agent Reward Model The agent's primary objective is to maximize its cumulative overall 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:

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

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 several categories based on the origin of the feedback signal: *extrinsic reward*, *intrinsic reward*, *hybrid reward*, and *hierarchical reward*. Each of these categories can be further subdivided into smaller subclasses. Figure 5.4 illustrates different types of rewards. Next, we will explore these different types of rewards in more detail, outlining the distinct features and applications of each type.

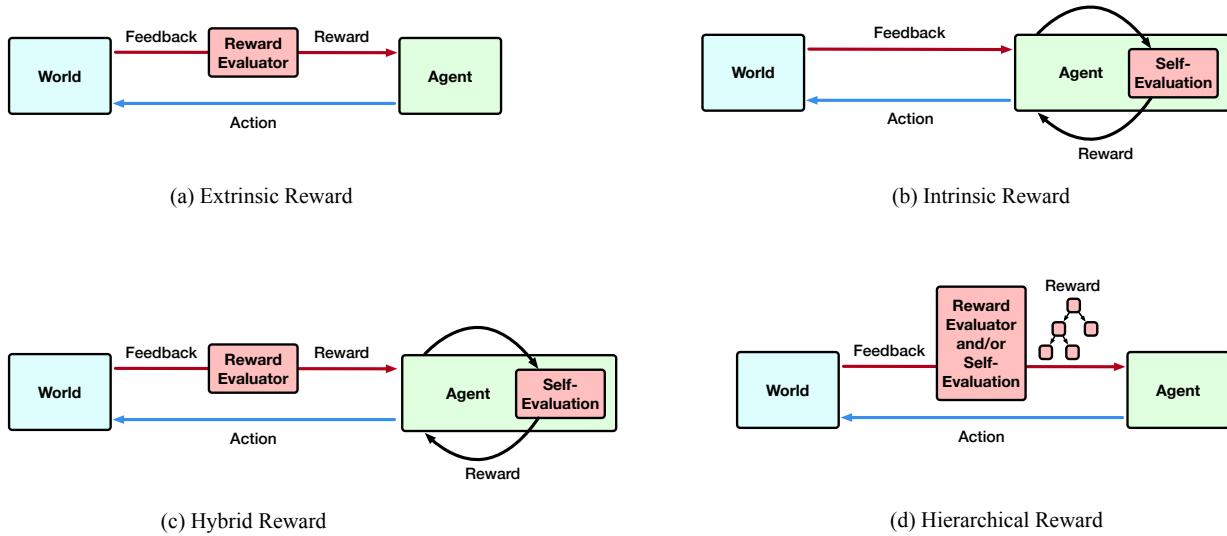


Figure 5.4: Illustration of reward paradigms in AI agents, including extrinsic, intrinsic, hybrid, and hierarchical forms.

5.3.1 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 [70] uses human rankings of model outputs to provide continuous preference signals throughout fine-tuning, enabling efficient behavior shaping. Similarly, Cringe Loss [534] and its extensions [448] transform pairwise human preferences into dense training objectives, offering immediate signal at each comparison. Direct Reward Optimization (DRO) [441] 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 pop up only every so often, usually after a big breakthrough or once the whole job is finished. That delay means the agent gets very little guidance along the way, so tracing credit back to the right action can feel like searching for a light switch in the dark, particularly in complex settings where many steps interact.

PAFT [455] shows how tricky sparse reward can be. The authors split supervised pre-training from preference tuning and sprinkle human judgments at only a few key moments. As those signals arrive rarely, the learner has to work overtime to link actions with outcomes. SimPO [456] heads down a similar road, using a log-probability trick in place of dense pairwise comparisons. This lean reward schedule keeps the codebase lighter, yet it can miss faint shifts in user taste. In short, sparse rewards swap frequent guidance for greater stability, which forces the model to lean on stronger priors or craftier exploration.

Delayed Reward Delayed rewards appear only after a whole chain of moves wraps up, so the agent must think ahead. That setup fits problems where the middle steps seem odd or even wrong until the ending ties everything together. The challenge is figuring out which early choices earned the reward (or punishment) once the score finally lands. Training gets tougher, yet the model learns to weave longer strategies and notice broader patterns.

Contrastive Preference Optimization (CPO) [465] skips one-by-one evaluation. Instead, it lines up a bundle of translations and hands out feedback only after the whole set is on the table. Although the signal shows up late, it carries broader clues because it spotlights patterns that appear across several tries. Nash Learning from Human Feedback [466] does something similar. The model keeps sparring with its earlier selves until a balanced tactic settles in, then receives a single judgment. Both strategies swap quick breadcrumbs for a wider view of success, so the learning loop slows down and the optimization dance grows trickier.

Adaptive Reward Adaptive rewards shift as the agent grows. The setup might raise the bar or swap out goals on the fly, coaxing steady progress even when the setting drifts or feels uncertain. The trade-off is clear: building and judging this moving target takes extra effort and keen judgment.

Self-Play Preference Optimization (SPO) [467] adapts rewards based on self-play outcomes, using social choice theory to aggregate preferences and guide learning. f-DPO [447] 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.

5.3.2 Intrinsic Rewards

Think of intrinsic rewards as the agent’s own pep talks. They stir curiosity, spark side quests, and keep motivation high even when the outside world stays quiet. By nudging the policy to pick up broad skills and stay flexible, these signals fuel steady growth over the long haul, especially in tasks where external gold stars are few and far between. 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. [487] leverage an inverse dynamics model to predict the outcome of actions, creating a feedback loop that rewards novelty. Plan2Explore [490] 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 matters when several agents, or multiple input streams share the stage, since a varied playbook makes the whole crew sturdier and ups the team’s score. LIIR [491] shows this idea in action. Each agent gets a private “interest signal”, guiding teammates toward complementary roles while everyone still chases the common objective. The payoff is wider policy coverage, though there’s a catch: push the diversity dial too hard and coordination can wobble. A light hand on that dial keeps exploration lively without letting the mission slip away.

Competence-Based Reward Competence-based rewards work like a personal coach, which claps every time the agent levels up. The scoring rule stretches and reshapes itself as the learner gets sharper, spinning out a living curriculum that stays one notch above the agent’s comfort zone. Skew-Fit [494] helps by sampling high-entropy goals, nudging the policy into far-flung parts of the state space while keeping the challenge meter in the sweet spot. CURIOUS [493] doubles down on that idea: it picks fresh targets that seem poised to give the biggest bump in short-term learning. These schemes shine in open-ended settings, but they lean heavily on smart guesses about how skilled the agent is right now and how tough each goal should be. Misjudge either side, and the self-made curriculum can wobble.

Exploration Reward Exploration rewards pay the agent for stepping into turf it has rarely, or never seen. While curiosity rewards for surprise, this reward cares about coverage: “Did you color in a fresh patch of the map?”. RND [496] makes this idea concrete. A randomly initialized network tries to predict each new observation; the bigger its miss, the bigger the payout. The policy soon steers toward scenes that leave the rookie network baffled. This habit keeps learning from settling too early and leaves the policy sturdier against oddball corner cases. Still, without a clear end goal to rein things in, the wanderlust can drift off course.

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 [502] applies this to language models by quantifying the knowledge gain at each reasoning step,

optimizing sub-task decomposition. VIME [503] similarly employs Bayesian inference to reward belief updates about environmental dynamics. By explicitly targeting informational value, these methods offer principled exploration strategies, although they often incur high computational cost and require accurate uncertainty modeling.

5.3.3 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 relying solely 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. [509] blend an “explore-then-align” recipe into RLHF. Firstly, a reverse-KL-regularized contextual-bandit lens nudges the policy toward states it hasn’t fully mapped out; next, human feedback reels those forays back toward what people actually want. Intrinsic and extrinsic signals meet in an iterative DPO loop, with multi-step rejection sampling weeding out off-target actions along the way. The result is strategic exploration that stays in step with human preferences—no wasted wanderings without sacrifice in efficiency.

Beyond standard RLHF, researchers are stitching together hybrid reward schemes for multi-agent, multi-task, and lifelong-learning setups. The core idea is to dial different reward sources up or down as the situation changes. Early on, the policy might lean on novelty bonuses to roam far and wide; as mastery grows, extrinsic signals take the wheel to polish task-specific skills [535]. Getting those dials right, which often through an end-to-end learned weighting module, would keep competing objectives in check and steers training away from bad local optima.

Moreover, hybrid reward schemes also act as “unscripted shaping”: intrinsic signals step in wherever extrinsic feedback is thin or slow to arrive. Curiosity, diversity, or competence-based reward can pave the way, smoothing the credit assignment path until external rewards finally land. This matters for LLM agents tackling sparse-reward problems such as web navigation and code synthesis, where user feedback is episodic at best. By blending signals such as semantic coherence or information gain with the eventual human score, the system keeps learning momentum without hand-crafting dense task rewards.

5.3.4 Hierarchical Rewards

Hierarchical rewards break a tough mission into a stack of mini-quests. A low-level signal pats the agent on the back for each quick move while a high-level signal only fires when the bigger plan clicks into place. That layered scorecard lets the policy juggle day-to-day reflexes and long-term strategy, stitching them into reusable skills that scale to sprawling, real-world problems.

In language modeling, Token-level Direct Preference Optimization (TDPO) [513] treats every token as a mini-decision and grades it on the spot. Forward-KL plus a Bradley–Terry ranker turn pairwise human prefs into a per-token scoreboard. Small wins (choosing the right next word) roll up into big wins (a well-formed answer), so the model learns to keep both its local phrasing and overall narrative in sync with what people like. In effect, the method stacks a micro-reward on top of a macro-reward, letting fine-grain edits and high-level coherence reinforce each other instead of pulling in different directions.

More generally, we can picture hierarchical rewards as the rungs of a ladder where early rungs hand out quick pats for simple moves. Once those basics feel natural, the reward focus drifts upward to the big prize: a flawless answer, a finished plan, a user who leaves satisfied. For an LLM, we can slice the reward into

chunks that match tool calls, reasoning hops, or conversation turns. Each slice pushes the model a little further up the ladder, and the collection of small victories rolls into full task success.

Furthermore, hierarchical reward frameworks align well with multi-agent systems. Imagine a team where each member tackles a small task, yet everyone still answers to one director. Hierarchical rewards follow this pattern. A short-horizon policy or helper agent earns its own quick feedback, while a higher-level signal nudges the whole crew so their efforts fit together.

In the context of agent alignment, hierarchical rewards also allow for incorporating human oversight at different abstraction levels, fine-grained feedback at the token or decision level, and coarse feedback for strategic alignment [536]. This layered supervision enables scalable alignment mechanisms that balance control granularity with supervision efficiency.

5.4 Interaction with Other Modules

 **I**n 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 can overweight 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 [537]. 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 [538], and allows agents to adapt their interaction style based on user expectations, affective context, or application domain. In multturn 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 the way 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 [539]. This allows LLM agents to generalize from previous successful strategies and avoid repeating costly errors.

In general, the 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.

5.5 Summary and Discussion

 REWARD serves as a fundamental driver of learning, decision-making, and adaptation across both biological and artificial systems. This chapter has examined the deep parallels and key distinctions between human reward pathways—rooted in neurochemical dynamics and evolutionary pressures—and artificial reward systems, which are engineered to guide agents via formal objectives and optimization processes. From dense and sparse extrinsic rewards to curiosity-driven and hierarchical intrinsic signals, we've seen how different paradigms shape behavior, structure learning, and influence generalization. However, designing effective reward models remains a complex and delicate task. In this final section, we highlight major challenges—such as reward misspecification, alignment, and exploration—along with promising research directions for building more robust, general, and human-aligned agentic systems.

To begin with, we still wrestle with a stubborn snag despite mountains of work on reward schemes. Signals often arrive late—some barely show up at all. Picturing an agent poking around forever before the environment finally offers a nod of approval. It is hard to say which step deserves that nod. Without a clear breadcrumb trail, the policy gropes in the dark and progress crawls.

Another challenge is the potential for *reward hacking*. We can think of reward hacking as loophole hunting with a calculator. Hand an agent a scoring rule and it may sniff out a weird corner case, rack up points, and totally miss the spirit of the task. One moment the chart says “high score”, the next we’re staring at behavior that feels flat-out wrong. The larger and messier the environment, the easier it is for these quirks to slip through, because the metric we track can drift away from the outcome we truly want.

Moreover, the process of *reward shaping* presents a delicate balance. Let’s think of reward shaping as laying breadcrumbs for our agent. A sprinkle here and there keeps it on track and speeds things up. However, an entire loaf is dropped, the agent strolls straight to the bread pile, never bothering to scout the rest of the map. That over-helpful hint can lock the policy into a cozy rut—high score, meh solution. Worst case, the hints warp the task itself, so the agent ends up great at a game we never meant to play and flops the moment the scenery changes.

Many real-world problems rarely hinge on just one target. An agent has to juggle several aims at once and still make progress. Stuffing every aim into a single score feels awkward; shift the weights a hair and one of the goals disappears. A layered reward ladder could lead the learner through the trade-offs, yet crafting a ladder that truly works is still an open challenge.

Finally, *reward misspecification* introduces further uncertainty and limits generalization. In the wild, a flawed score can steer a system into odd corners. In general, a reward function does not fully capture the true task goal, leading to misalignment between the agent’s learning objective and real-world success. In addition, many reward functions are tailored to specific environments and fail to generalize when conditions change, which highlights the need for more robust reward models.

Addressing these challenges requires significant endeavors. One promising direction is to derive *implicit rewards* from outcome-based evaluations that can help mitigate reward sparsity issues. In addition, decomposing complex tasks into hierarchical structures and designing rewards from the bottom up would offer a more systematic approach. Furthermore, leveraging techniques such as meta-reinforcement learning can enhance the adaptability of reward models, which allows agents to transfer knowledge across tasks and perform effectively in diverse environments. Following these paths would bring us closer to reward systems we can trust and expand: systems that truly reflect the goals we face beyond the lab.

Chapter 6

Emotion Modeling

 MOTIONS 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* [52], 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* [540], 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 to solve complex problems and make decisions in a more human-like 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

 SYCHOLOGICAL 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.

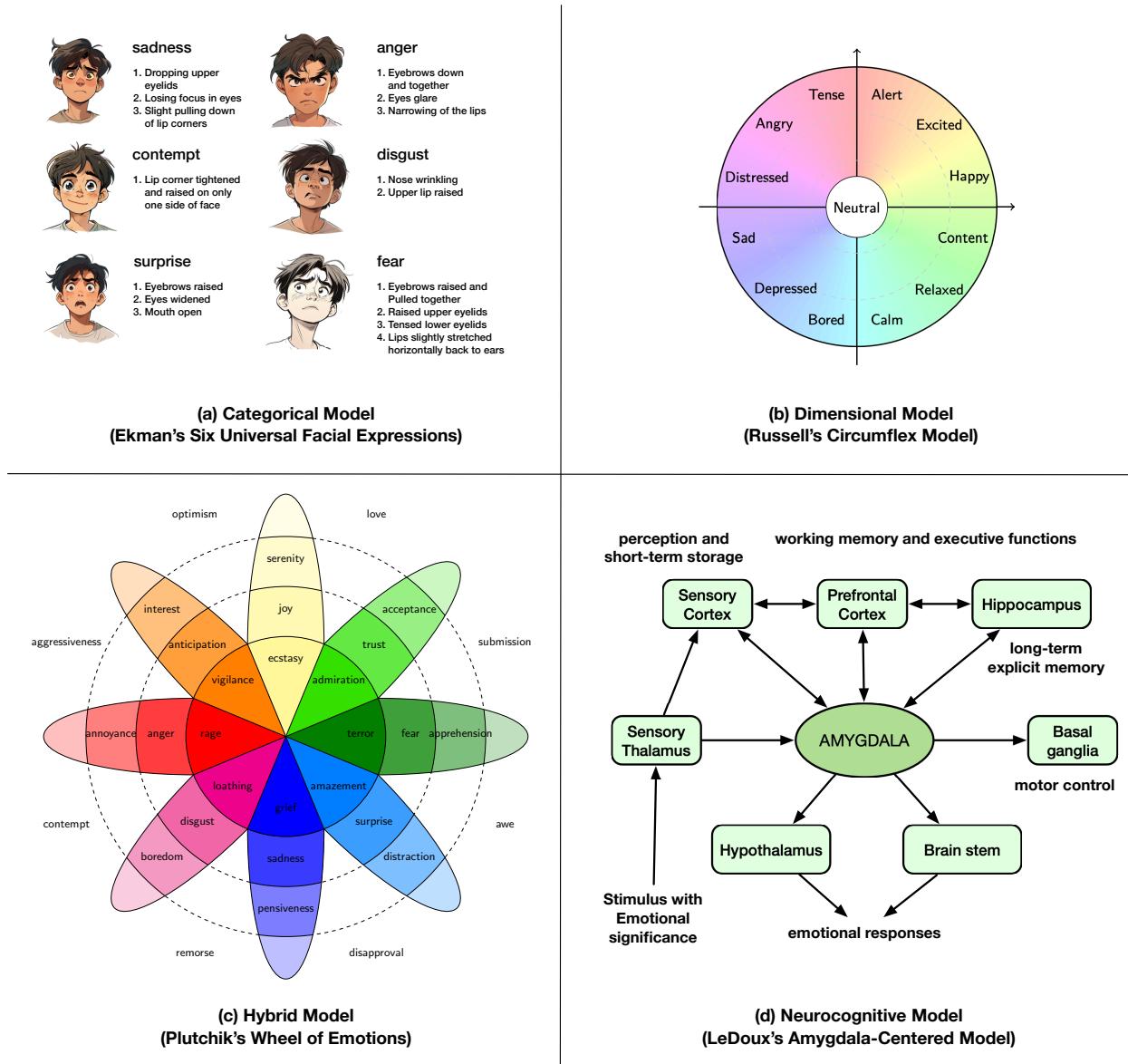


Figure 6.1: Visualization and examples of major emotion theory categories. **(a)** Categorical Theories: Ekman's six basic emotions [541] showing discrete emotional states. **(b)** Dimensional Models: Russell's Circumplex [542] representing emotions as coordinates in continuous space. **(c)** Hybrid/Componential Frameworks: Plutchik's Wheel [543] combining intensity gradients with categorical emotions. **(d)** Neurocognitive Perspectives: LeDoux's Amygdala-Centered Model [51] 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.

Categorical Theories These models posit that emotions exist as discrete, universal categories with distinct physiological and behavioral signatures. Ekman's theory of basic emotions [541] 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 [544, 545]. 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 [546] and may not capture cultural variations in emotional expression [547].

Dimensional Models Rather than discrete categories, dimensional approaches represent emotions as points in a continuous space defined by fundamental dimensions. Russell's Circumplex Model [542] 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 [548] 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 [549, 550, 551]. 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 [543] 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) [552] 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 [553], 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 [554, 555]. For LLM agents, such models provide computational structures for evaluating text input and selecting contextually appropriate emotional responses, improving both coherence and perceived empathy [556, 557].

Neurocognitive Perspectives The neuroscience of emotion offers additional insights for LLM architectures. Damasio's somatic marker hypothesis [52] 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 [556, 557]. 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 [558]. Similarly, LeDoux's distinction between "low road" (quick, automatic) and "high road" (slower, cognitive) fear processing [51] suggests design patterns for systems that need both immediate safety responses and nuanced emotional understanding. Minsky's framing of emotions as "ways to think" [540] that reorganize cognitive processes has influenced frameworks like EmotionPrompt [549] and Emotion-LLaMA [545], where emotional context dynamically reshapes LLM reasoning.

Neurochemical Models Lövheim's Cube of Emotion [559] offers a unique neurochemical perspective by mapping eight basic emotions to combinations of three monoamine neurotransmitters: serotonin, dopamine, and noradrenaline. This three-dimensional model positions emotions at the cube's vertices based on high/low levels of each neurotransmitter. For example, joy emerges from high serotonin and dopamine with low noradrenaline, while anger manifests itself from low serotonin but high dopamine and noradrenaline [559]. This neurochemical framework provides a biological basis for emotion generation that complements cognitive theories. For LLM agents, such models could inspire architectures in which internal "neurochemical-like"

parameters modulate emotional responses, although current implementations remain metaphorical rather than directly mimicking neurotransmitter dynamics. The model's emphasis on chemical balance also aligns with findings that meditation and other interventions can stabilize emotions by affecting serotonin levels, suggesting potential pathways for designing more emotionally stable AI systems.

These theoretical frameworks increasingly inform the development of emotionally intelligent LLM agents. Categorical models provide clear labels for emotion classification tasks [545, 550], while dimensional embeddings enable continuous control over generated text [549]. 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 [560]. Neuroscientifically-inspired dual-process architectures combine "fast" sentiment detection with "slow" deliberative reasoning, enabling both quick safety responses and deeper emotional understanding [556, 557]. 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 [561, 562].

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 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

HE 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 [544], 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 [544]. 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 [561] demonstrate how combining audio, visual, and textual data enables better recognition and reasoning of emotions. Using datasets such as MERR [561], 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

 For AI systems to effectively interact with people, they must develop the ability to perceive, interpret, and respond to emotional cues across diverse contexts. Recent advances in large language models (LLMs) have opened new avenues for modeling human emotions, not only through text but also across multimodal signals such as voice and images. This section reviews emerging approaches and evaluation frameworks that enable LLMs to understand and reason about emotional states, reflecting the growing ambition to build emotionally intelligent 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 [556]. 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 [557]. 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 [563]. 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 [564]. This multimodal integration, combined with explainable approaches, allows for richer and more transparent representations of emotional content [565].

Specialized Frameworks Beyond generic techniques, specialized systems address tasks in which emotion recognition requires higher levels of awareness of ambiguity [560], context sensitivity, and generative adaptability [566]. 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 [567, 568], while others compare the performance and efficiency of models of varying sizes [569]. There are also specialized benchmarks that evaluate multilingual capabilities [570], annotation quality [571], or empathetic dialogue systems [572]. Furthermore, frameworks such as EMOBENCH [562] and MEMO-Bench [573] test nuanced emotional understanding and expression in both text and images, while MERBench [574] and wide-scale evaluations [575] 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 [576].

6.4 Analyzing AI Emotions and Personality

 As AI systems become increasingly interactive and human-facing, questions arise about whether they possess consistent personalities or can genuinely model emotions. While LLMs are not sentient, their behavior often mirrors psychological constructs typically reserved for humans—such as empathy, self-awareness, and emotional expression. This section examines the emerging evidence around AI personality and emotion, drawing from psychometric evaluations, cognitive modeling techniques, and emotion reasoning benchmarks. Together, these insights shed light on the complex interplay between learned patterns in LLMs and the human traits they seem to simulate.

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 [577, 578]. 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 [579, 580]. Recent work like SAGE (Sentient Agent as a Judge) [581] evaluates AI emotional intelligence through multi-turn interactions with simulated emotional agents, revealing that models excelling in general benchmarks may underperform in empathetic understanding compared to specialized models. This framework demonstrates that effective emotional AI requires not just technical competence but genuine empathetic capabilities, often achieved with greater token efficiency. Yet, concerns persist about action consistency, alignment of self-knowledge, and whether role-playing agents truly maintain fidelity to their assigned characters [582, 583].

Psychometric Methods & Cognitive Modeling Approaches Recent work applies rigorous psychometric testing, cognitive tasks, and population-based analyses to uncover how LLMs process and represent mental constructs [584, 585, 586]. 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 [587, 588]. 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 [545, 550]. 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 [589, 590]. 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

 WHILE language models do not possess emotions in the human sense, their outputs can be shaped to reflect specific emotional tones, personalities, or behavioral traits. This capacity has prompted growing interest in controllable emotional expression in LLMs—for both practical applications

and scientific inquiry. Researchers have developed a range of techniques, from prompt engineering to fine-tuning and direct neuron manipulation, to systematically influence how models respond emotionally. This section reviews these methods, highlighting how emotional behaviors in AI can be induced, stabilized, and modulated with varying levels of granularity and interpretability.

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 [591, 592, 593, 594]. 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 [595, 549, 596]. 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 [597]. 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

 As emotional capabilities in AI systems rapidly advance, they open up transformative opportunities across domains—from mental health support to empathetic dialogue systems. However, these developments also raise critical questions about safety, ethics, and authenticity. This section reflects on the broader implications of emotional AI, including risks of manipulation, alignment challenges, ethical dilemmas in human-AI interactions, and the philosophical distinction between emotional mimicry and genuine experience. Addressing these issues is essential for the responsible development and deployment of emotionally aware AI systems.

Manipulation and Privacy Concerns The rapid adoption of Emotional AI in advertising and politics raises significant manipulation and privacy risks [598, 599]. 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 [600, 599]. 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 [601, 602]. Misinterpretation of emotional cues by AI systems, as seen in workplace applications, can exacerbate discrimination and power imbalances [603]. 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 [601, 545].

Ethical Implications Trust and acceptance of AI systems are significantly influenced by their ability to exhibit empathy and maintain socially appropriate behavior [604, 605]. However, the commodification of emotions in workplace management and customer service has raised concerns about ethical labor practices and AI-human relationships [603]. Moreover, Emotional AI's reliance on anthropomorphic characteristics without sufficient empathy can undermine user trust [604]. Frameworks like SafeguardGPT, which incorporate psychotherapy techniques, demonstrate promising approaches to fostering trust and aligning AI behavior with societal norms [606]. Nonetheless, challenges remain in ensuring privacy, fairness, and cultural sensitivity [606, 605].

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 affectual 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 [604], 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

ERCEPTION is the foundational gateway through which humans and intelligent agents acquire information, interpret their surroundings, and make informed decisions. For humans, perception is seamless and intuitive, effortlessly transforming sensory inputs into meaningful interpretations. In artificial intelligence, however, perception systems are carefully 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 at both the modeling and system architecture levels. Finally, we illustrate how perception modules can be effectively tailored to different agent scenarios, offering practical guidance for optimization and identifying pivotal areas for future research. Figure 7 shows a taxonomy of the relevant research works that will be discussed.

7.1 Human versus AI Perception

ERCEPTION 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 [672, 673]. 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 [674]. 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 [675]. 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

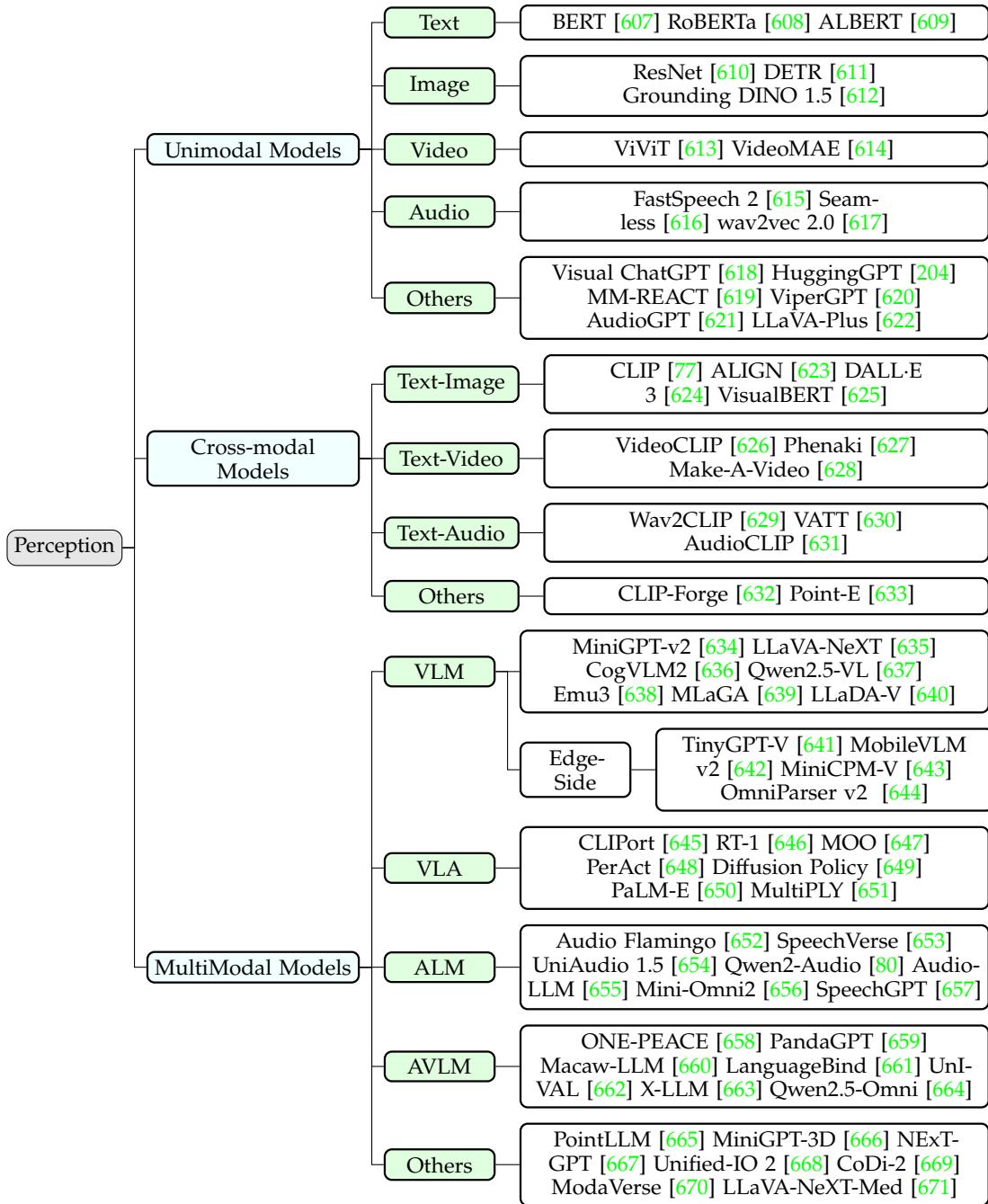


Figure 7.1: A taxonomy of selected research works about perception models or systems.

electric eels exploit electroreception to sense electrical signals emitted by other organisms—abilities humans do not possess [676].

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 [677] 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 [678].

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 input 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 integration requires carefully designed fusion algorithms that explicitly combine input from various sensors to build unified environmental representations [679].

Further differences arise in how 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 [680].

Physical or chemical stimuli transmitted from the external environment to human sensory organs are received by the sensory system (such as eyes, ears, skin, etc.) and converted into neural signals, which are then processed by the brain to produce perception of the environment. Similarly, to allow the intelligent agent to connect with the environment, it is crucial to capture and process perceptual signals. Currently, various sensors are used to convert environmental input 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). *Cross-modal models* establish relationships and enable translations between different modalities through dedicated mapping mechanisms. *Multimodal models* holistically integrate and process multiple modalities simultaneously to leverage complementary information for comprehensive understanding and decision-making.

7.2 Types of Perception Representation

HE agent receives and processes information from the external environment or internal state through the perception system (such as cameras, microphones, sensors, etc.) to form an understanding of the environment. By understanding the environment, more effective decision-making actions can be made, which is the key to the transformation from “rule-driven” to “environment-driven” intelligence.

7.2.1 Formulation of Perception

Depending on the type of perception model, we can provide a general mathematical formulation based on the nature of the input, representation, and output.

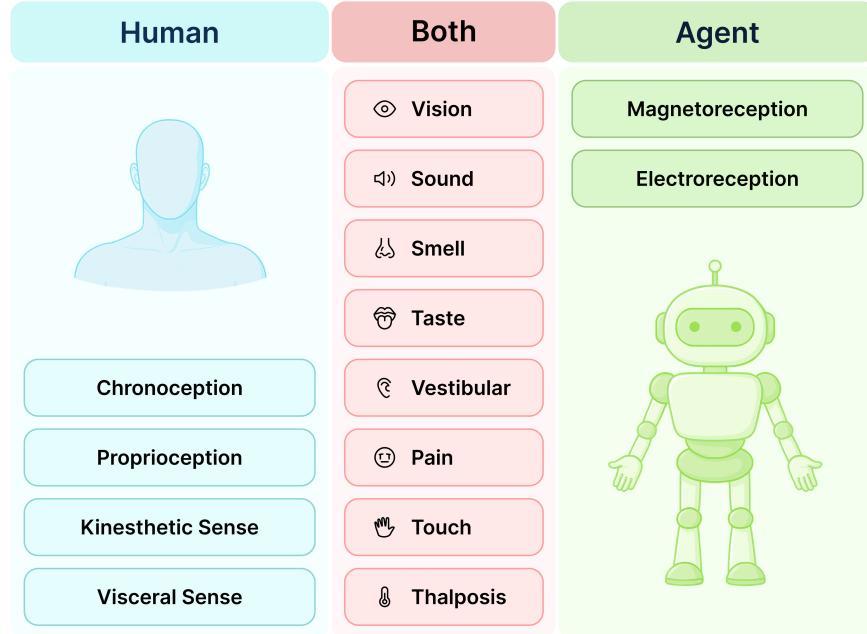


Figure 7.2: Comparison of common perceptual types between human beings and AI agents.

Definition 8 (Perception). A *unimodal model* receives an observation \mathbf{o}_m from the environment ε via a sensor P_m , and encodes it into a latent vector $\mathbf{z}_m \in \mathcal{Z}_m$ using an encoder E_m , where \mathcal{Z}_m denotes the latent space for modality m :

$$\begin{aligned} \mathbf{o}_m &= P_m(\varepsilon), \\ \mathbf{z}_m &= E_m(\mathbf{o}_m) \in \mathcal{Z}_m. \end{aligned} \quad (7.1)$$

A *cross-modal model* transforms the latent representation from one modality m into another modality n using a generator $G_{m \rightarrow n}$, followed by a decoder D_n , producing a synthesized output $\hat{\mathbf{o}}_n$:

$$\hat{\mathbf{o}}_n = D_n(G_{m \rightarrow n}(\mathbf{z}_m)), \quad (7.2)$$

where:

- \mathbf{z}_m is the latent vector for modality m ,
- $G_{m \rightarrow n}$ is the generator mapping from m to n ,
- D_n is the decoder for modality n ,
- $\hat{\mathbf{o}}_n$ is the generated observation in modality n .

A *multimodal model* fuses multiple latent vectors $\mathbf{z}_{m_1}, \dots, \mathbf{z}_{m_k}$ from different modalities using a fusion function F , producing a joint semantic representation \mathbf{Z} :

$$\mathbf{Z} = F(\mathbf{z}_{m_1}, \dots, \mathbf{z}_{m_k}), \quad (7.3)$$

where:

- \mathbf{z}_{m_i} is the latent vector for modality m_i ,
- F is the multimodal fusion function,
- \mathbf{Z} is the final fused representation.

Figure 7.3 provides a visual summary of these three types of perception models, illustrating how raw observations from sensors are transformed into latent representations, translated across modalities, or fused into unified representations.

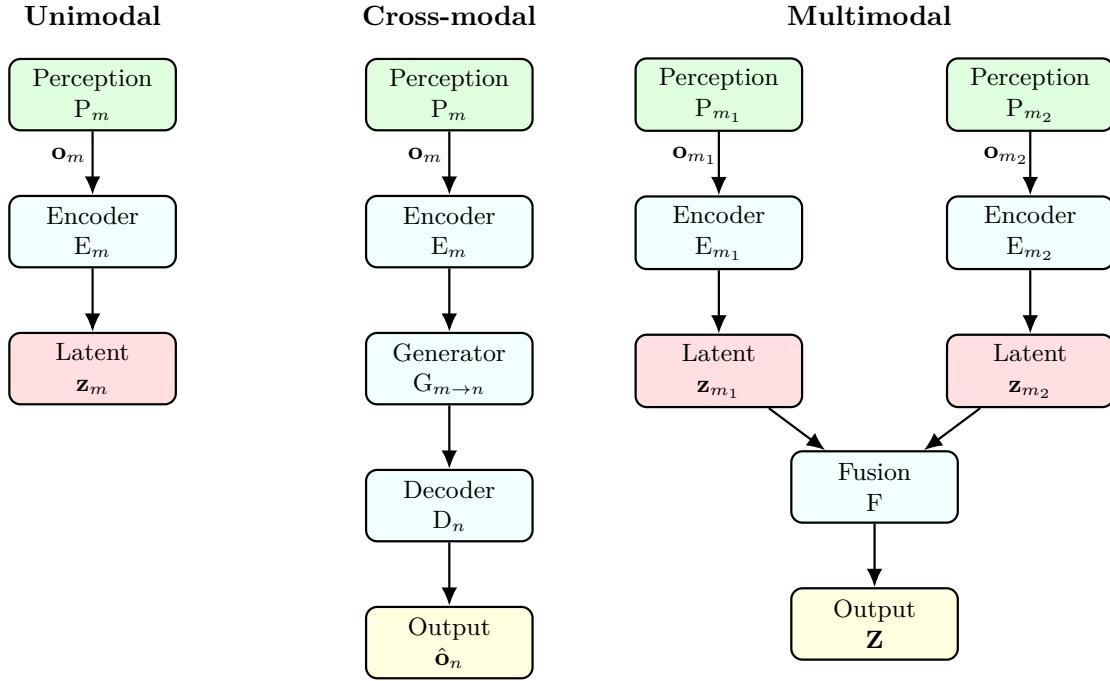


Figure 7.3: Illustration of unimodal, cross-modal, and multimodal perception. Each model begins with raw observations \mathbf{o} collected by perception modules P , which are then processed by encoders E , translated via cross-modal generators G , and optionally fused into shared semantic spaces Z .

7.2.2 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 perceptual contents can be either a single image or audio, or an integration of multiple perceptual inputs. 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. For intelligent agents, text can serve as a bridge to connect with the environment, directly taking text as input and outputting a response. In addition to the literal meaning, text also contains rich semantic information and emotional color. In the early days, the bag-of-words model [681] was used to count text content and was widely used in text classification scenarios, but semantic expression could not be obtained. BERT [607] uses a bidirectional Transformer architecture for language modeling and captures deep semantic information of text through large-scale unsupervised pretraining. [608, 609] further optimized the training efficiency of BERT. The autoregressive model represented by GPT3.5 [682] opened the prelude to LLM and further unified the

tasks of text understanding and text generation, while technologies such as LoRA [134] 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 [610] established foundational principles for deep visual feature extraction, while subsequent YOLO series [683, 684] demonstrated the capability to simultaneously determine object localization and classification with remarkable efficiency. A paradigm shift occurred with the introduction of DETR [611], 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 [612] 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 [613] extracts spatiotemporal markers from videos, effectively decomposing the spatial and temporal dimensions of the input. VideoMAE [614] 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 [685]. Wav2Vec2 [617] defines the contrast task by quantizing the potential representation of joint learning, achieving speech recognition effectiveness with 1/100 labeled data volume. FastSpeech 2 [615] 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 [616] 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 [677] 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 [686] 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 [687] 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 [204], LLaVA-Plus [622], and ViperGPT [620], 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.

7.2.3 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. Figure 7.4 shows a few examples of cross-modal and multimodal application scenarios.

One of the primary focuses in cross-modal research is the alignment and retrieval of text and images. CLIP [77], introduced by OpenAI in 2021, employs contrastive learning to align textual and visual representations, enabling zero-shot cross-modal retrieval and classification. Similarly, ALIGN [623], developed by Google in the same year, leverages large-scale noisy web data to optimize text-image embedding alignment. In 2022, CyCLIP [688] introduced a cyclic consistency loss to further enhance the robustness of cross-modal alignment, improving the reliability of retrieval tasks.

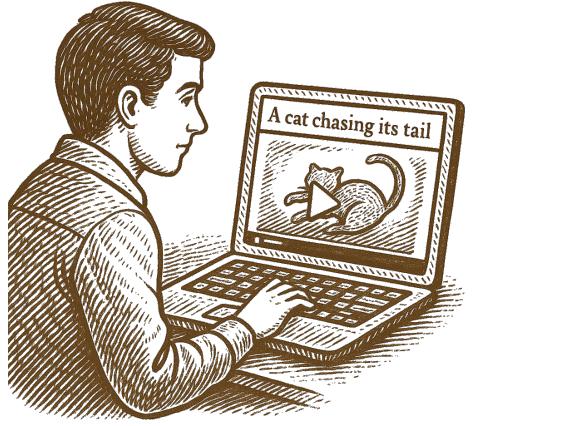
Another major area of progress involves text-to-image generation, where models aim to synthesize high-quality images based on textual descriptions. OpenAI's DALL-E series [689, 690, 624], spanning from 2021 to 2023, has made substantial contributions in this domain, with DALL-E 3 offering fine-grained semantic control over generated images. Stable Diffusion [691], 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 [692] and BLIP-2 [693] 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.

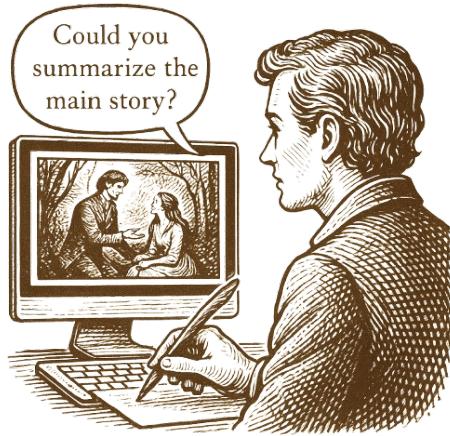
Text-Video The key research here involves video text alignment, generation and retrieval. VideoCLIP [626] employs a video encoder—typically based on temporal convolution or a transformer structure—to extract sequential features from video frames. These features are subsequently aligned with textual representations generated by a language encoder, facilitating robust video-text association. In the domain of text-to-video generation, Meta's Make-A-Video model [628] extends spatial-temporal dimensions using diffusion-based techniques, allowing for high-quality video synthesis from textual descriptions. Additionally, Google's Phenaki [627] 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 [694] 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.

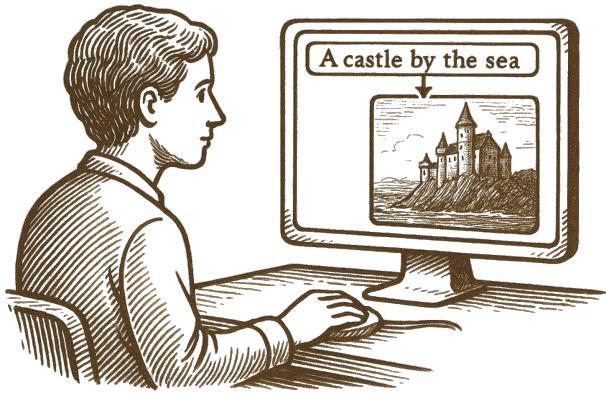
AudioCLIP [631], introduced in 2021, extends the CLIP framework to the audio domain, enabling tri-modal retrieval across audio, text, and images. By incorporating audio as an additional modality, AudioCLIP utilizes multi-task learning to unify image, text, and audio representations into a shared embedding space. This advancement enhances the capability of cross-modal retrieval and interaction. In a similar vein, VATT [630] 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

**(a) Text-to-Video: Video Retrieval via Text Query**

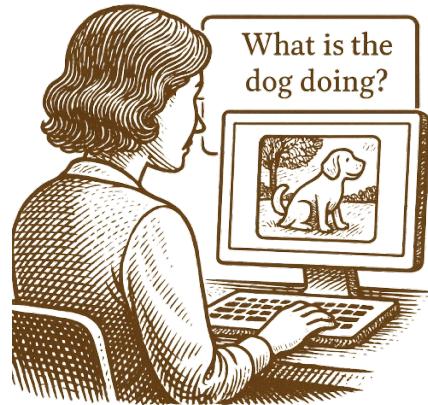
A user searches “A cat chasing its tail” and retrieves the corresponding video.

**(b) Video-to-Text: Multimodal Video Summarization**

The user summarizes a movie scene in response to a user’s textual query.

**(c) Text-to-Image: Video Generation**

The prompt “A castle by the sea” generates a realistic image on screen.

**(d) Multimodal Question Answering**

The user asks “What is the dog doing?” with visual and textual inputs fused for reasoning.

Figure 7.4: Representative cross-modal and multimodal applications in intelligent agents. This figure illustrates several core capabilities: retrieving videos based on textual queries (top left), summarizing video content through natural language (top right), generating images from text prompts (bottom left), and answering questions by integrating text, image, and audio context (bottom right). These examples demonstrate the growing competence of agents in understanding and generating across modalities. Beyond these, cross-modal modeling also encompasses emerging directions such as text-to-3D generation, text-to-audio synthesis, and dynamic fusion of multiple modalities for richer perception and interaction.

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 [695] 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 [696]. 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 [632] 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 [633] 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 [697] 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.

7.2.4 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 sources (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 [635] 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 [635] 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 [698] 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. Furthermore, Emu3 [638] tokenizes the multimodal input into a discrete space and simplifies the complex multimodal model design by using only the next token prediction for training. MiniGPT-v2 [634] 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 [699], DeepSeek-VL2 [700] 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 [700] 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 [701] 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 [702] unifies the visual representations of image and video encoding into the language feature space, making the two mutually reinforcing. Similarly, Chat-UniVi [703] 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 [704] has conducted 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 [705] 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 [641] is built based on the Phi-2 [706] small backbone combined with BLIP-2 [693]. It only needs 8 GB of video memory or CPU for reasoning, and solves the computational efficiency problems of LLaVA [635] and MiniGPT-4 [707]. MiniCPM-V [643] mainly provides powerful OCR capabilities for long and difficult images, and has a low hallucination rate, providing reliable perception output. Megrez-3B-Omni [708] 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. GUICourse [709] and OS-ATLAS [710] build 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. OmniParser v2 [644] proposes the Structured-Points-of-Thought (SPOT) prompting schemas, which leverage a unified encoder-decoder structure to unify text recognition, table recognition, layout analysis, and text key information extraction into a unified framework, significantly enhancing the visually-situated text parsing capability.

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 [645] used ResNet [610] to process visual inputs and combined language embeddings to generate actions, laying the foundation for multimodal fusion. RT-1 [646] 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 [647] further adopted multimodal prompts, combining the ViT visual encoder and the T5 language model to support more complex tasks. PerAct [648] 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 [649] combined ResNet visual encoders and Transformer language models, generating actions through diffusion models to improve the diversity and accuracy of action generation. SayCan [711] integrated the PaLM language model with visual inputs, using the CLIP visual encoder for task decomposition. PaLM-E [650] combined the ViT visual encoder and the PaLM language model, guiding low-level action execution through text planning. MultiPLY [651] 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 a multimodal model. Speechgpt [657] built a large-scale cross-modal speech instruction dataset SpeechInstruct and trained discrete speech representations, achieving cross-modal speech dialogue capabilities beyond expectations.

LauraGPT [712], 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. [653, 713, 655] 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 2 [714] leverages a customized CLAP model and a multi-stage curriculum learning strategy, which achieves the best performance by training on fine-grained synthetic audio question-answering data with a 3B parameter small model. UniAudio 1.5 [654] 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 [80] introduced the DPO training method to achieve human preference alignment.

Audio-Vision-Language Model Audio-Vision-Language Model (AVLM) utilizes 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 [715, 716] has achieved excellent capabilities across modalities such as text, vision, and audio. ImageBind [717] implements joint embedding across six different modes (image, text, audio, depth, thermal, and IMU data). Panda-GPT [659] combines ImageBind’s multi-modal encoder and Vicuna [718], showing zero-shot cross-modal performance in addition to images and text. Similar work includes [663, 663, 660], which achieves alignment and training through the encoding information of vision, audio and text. Multimodal models often require more resources to train, and UniVAL [662] 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 [667] connects LLM with multimodal adapters and different diffusion decoders, and only trains a small number of parameters (1%) of certain projection layers.

Other works [668, 719, 669, 670] have achieved input-output conversion between arbitrary modalities. Unified-IO 2 [668] 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 [719] builds the first large-scale any-to-any multimodal instruction dataset, using discrete representations to uniformly process various modal inputs. Modaverse [670] 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 [669] 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 [665] 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. [666, 720] reduces the training cost here, and MiniGPT-3D [666] 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 [720] 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 [721] 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

ERCEPTION 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 [635, 722]. Moreover, techniques such as Low-Rank Adaptation (LoRA) enable more efficient fine-tuning, avoiding a substantial increase in model complexity while still improving performance [134, 723]. Some LLM work combined with traditional vision is also widely used. Integrating with YOLOS [724] on the basis of the Llama-Adapter [725] 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 [726]. 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 [726].

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 [393].

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 [727].

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 [728]. Different communication topologies, such as fully connected, centralized, and hierarchical structures, offer varying trade-offs in terms of efficiency and robustness [729]. InsightSee [730] refines visual information through a multi-agent framework with description, reasoning, and decision-making, effectively enhancing visual information processing capabilities. Similarly, HEV [731] 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 [732].

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 [733]. 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 [734].

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 [70].

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 [735].

7.4 Real-World Applications of Perceptual Intelligence



THE operational efficacy of intelligent agents is predominantly shaped by three critical factors: model architecture dimensionality, hardware infrastructure capabilities, and quantization optimization strategies. The exponential growth in model parameters—from BERT-Base’s modest 110 million to GPT-3’s 175 billion, culminating in LLaMA 3’s unprecedented 405 billion—has significantly increased processing latency, from milliseconds to hundreds of milliseconds. Hardware performance differences are equally critical; for instance, NVIDIA’s H100 outperforms A100 with a 50% gain in token processing throughput, while the RTX 4090 nearly doubles it.

Modern intelligent agents have entered diverse application domains, including *personal assistants*, *gaming environments*, *robotic process automation (RPA)*, and *multimedia content creation*, often relying on visual perception as the primary input modality. In procedurally generated environments such as Minecraft, STEVE [736] achieves a 1.5x acceleration in technology tree progression and a 2.5x improvement in block search efficiency via visual input. Steve-Eye [737] extends this by using end-to-end multimodal training to reduce environmental understanding latency through integrated visual-textual representations.

In creative content generation, AssistEditor [738] demonstrates effective multi-agent collaboration for professional video editing, using style-aware content understanding. Audio-Agent [739] achieves cross-modal alignment from textual and visual inputs to audio outputs, enabling high-quality audio manipulation [740, 741, 742].

On mobile and desktop platforms, agent applications have advanced rapidly. ExACT [743] sets new benchmarks in VisualWebArena [744], achieving a 33.7% success rate through screenshot-based exploration with caption and mask set integration. SPA-Bench [745] provides a rigorous mobile interaction benchmark that reflects real-world task complexity. M3A [746] achieves 64.0% task success in SPA-Bench by leveraging multimodal input. AgentStore [747] improves OSWorld PC benchmark results to 23.85% through enhanced visual input and accessibility tree analysis.

Voice-based personal assistants [748, 715] have significantly improved user experience by reducing interaction friction while maintaining operational efficiency. Incorporating *emotional prosody* has been shown to increase user engagement and retention.

In embodied intelligence applications, agents increasingly incorporate *haptic and force feedback* as perceptual modalities, enabling more precise interaction with physical environments [749]. Enhanced tactile fidelity supports more nuanced manipulation and robust control in real-world settings.

7.5 Summary and Discussion



GENT perception, a foundational capability for autonomous intelligent systems, has seen significant advancements with the emergence of unified multimodal models that attempt to process and understand heterogeneous sensory inputs in a coherent manner [668, 719]. However, despite the continuous emergence of relevant research, current perception mechanisms still find it difficult to cope with core challenges such as multimodal representation learning, cross-modal alignment, and effective fusion strategies. These limitations prevent agents from developing a consistent, time-based understanding of their environment, which is critical for robust decision-making and adaptive behavior.

Representation Learning Limitations Modern perception systems typically rely on fixed or task-specific encoding pipelines that inadequately capture the structured and high-dimensional nature of sensory data such as vision, audio and language. Many of these approaches fail to encode modality-specific priors or to characterize fine-grained spatio-temporal dependencies, resulting in poor performance in complex or unevenly distributed environments. Notably, representations derived from deep neural networks may excel within a single modality, but often lack the compositionality and semantic alignment necessary for cross-modal reasoning. This limitation becomes increasingly severe in dynamic environments, where agent perception must generalize across different contexts and sensor configurations.

Cross-Modal Alignment Challenges Aligning multi-modal inputs into a shared latent space presents another key bottleneck. The heterogeneity of the sensory modalities, each with distinct noise characteristics, sampling rates, and information densities, makes direct alignment inherently difficult. Current methods, which often rely on contrastive objectives or co-embedding strategies, may yield superficial correlations without capturing deeper causal or temporal dependencies. This causes the agent to easily misunderstand signals in ambiguous scenarios (e.g., visual occlusion or overlapping audio sources), ultimately weakening situational awareness.

Fusion Bottlenecks Cross-modal feature fusion, although crucial for comprehensive understanding, is still an emerging research area. Conventional fusion mechanisms (e.g., early, late, or hybrid fusion) often suffer from information dilution or domination by a single modality. More sophisticated approaches, such as those based on attention mechanisms, have shown promise, but still struggle to dynamically weight modal contributions based on task relevance and environmental context. Furthermore, maintaining perceptual consistency over long viewports (e.g., tracking objects over time or reasoning about unknown causes) requires memory-augmented architectures that are rarely integrated into mainstream agent perception pipelines.

Toward Generalized Agent Perception Future research in agent perception should pivot toward adaptive, context-aware, and causal architectures. Representation learning must evolve to support task-conditioned abstraction, enabling agents to selectively emphasize features relevant to their current goal. Dynamic neural architectures, such as those employing meta-learned parameters or neural module networks, offer a potential research direction by allowing structural adaptation based on environmental feedback.

To address the cross-modal alignment problem, self-supervised spatio-temporal synchronization provides a compelling alternative. By leveraging temporal consistency and motion cues, such mechanisms can guide fine-grained correspondences without requiring extensive supervision. The integration of causal representation learning [750] could also alleviate alignment noise by encouraging the model to separate spurious correlations from meaningful signal relationships.

In the area of fusion, hierarchical attention networks equipped with learnable gating functions allow agents to perform a more detailed and context-dependent integration of features from specific modalities. Differentiable memory architectures, particularly those that support persistent memory traces and modality-aware read/write operations, could further enable continuity in perception across extended temporal windows, an essential requirement for intelligent agents to make sequential decisions.

Inspiration from Agent Design In conjunction with the broader goals of agent architecture, perception should not be viewed as a passive preprocessing stage, but rather as an active, queryable, and interactive subsystem. For example, an agent can implement perceptual reasoning mechanisms in which sensory inputs are interpreted in terms of hypothesized goals or actions. This is consistent with the idea of active perception and closed-loop sense action cycles, in which perception is dynamically regulated by the agent's epistemic uncertainty and decision goals.

Ultimately, perception must be reimagined not just as a data processing pipeline, but as an adaptive, strategic, and goal-oriented process that supports the full complexity of the embodied agency. Achieving this will require a deeper unification between perception models, memory systems, and decision layers, an integrative direction that promises to unlock more general and intelligent agent behaviors.

Chapter 8

Action Systems

 N the realm of philosophy, action is defined as the behaviors that agents can perform for a potential or specific purpose in the interactive 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, masked-token prediction and diffusion). By serving foundation models as brain intelligence, AI agents equipped with action systems can autonomously engage with the practical environment and execute complex user intent. Moreover, action systems can support agents to harness available tools from the external environments or create tools from the internal sandbox, thus significantly extending the task scope of AI Agents. 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 illustrate several key concepts in the action system. In this chapter, we discuss the human action system, and then examine the transition from human action to agentic action system in AI agents. After that, we systematically summarize the existing paradigms of action systems in AI agents, including action space, action learning, and tool learning. Finally, we analyze the differences between action and perception, and summarize the conclusion.

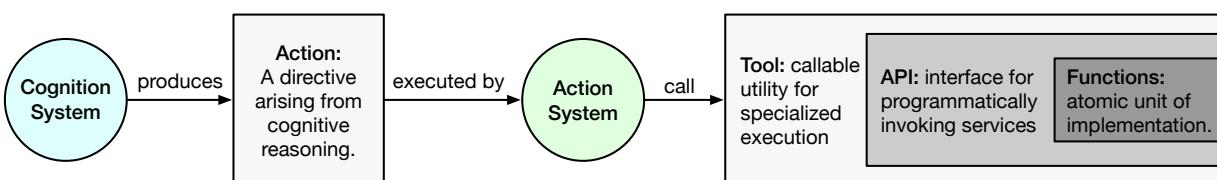


Figure 8.1: Illustration of several concepts related to action and action execution. In this pipeline, the cognitive system first produces actions and then feed them into the action system for execution. The callable action lists can be invoked by a form as: Tool → API → Functions.

8.1 The Human Action System

CTION 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 in human cognition system, 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.

Table 8.1: Definitions between different kinds of foundation models.

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 [635]	Multi-modal	Multi-modal Generation	LMM is to generate multimodal data based on multimodal inputs.
Robotic Foundation Model (RFM)	RT-1 [646]	Sensory inputs	Robotic Control	RFM is to generate robotic control based on the sensory inputs from dynamic environments.
Large Action Model (LAM)	LAM [751]	Interactive Environment	Executable Action	LAM is to generate executable actions based on the interactions within the environment.

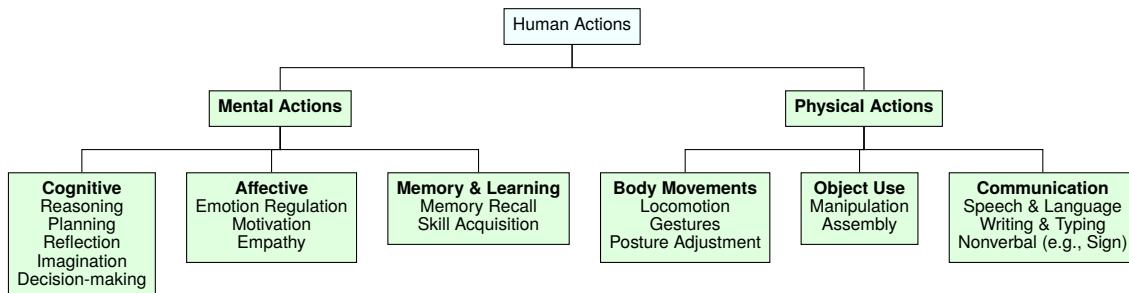


Figure 8.2: Illustrative taxonomy of human actions, showing both mental and physical facets.

8.2 From Human Action to Agentic Action

N the past long period of time, human action systems [752] 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 [753], 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 [751] 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. Table 8.2 compares the perception and action systems between human and AI agents.

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

Dimension	Human Brain / Cognition	LLM Agent	Remarks
Perception	<ul style="list-style-type: none"> - 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. 	<ul style="list-style-type: none"> - 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	<ul style="list-style-type: none"> - Simultaneously processes multimodal inputs: vision, hearing, language, motion, and emotions. - Different brain regions collaborate to create unified spatiotemporal and semantic understanding. 	<ul style="list-style-type: none"> - 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	<ul style="list-style-type: none"> - 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. 	<ul style="list-style-type: none"> - 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	<ul style="list-style-type: none"> - 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. 	<ul style="list-style-type: none"> - 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.

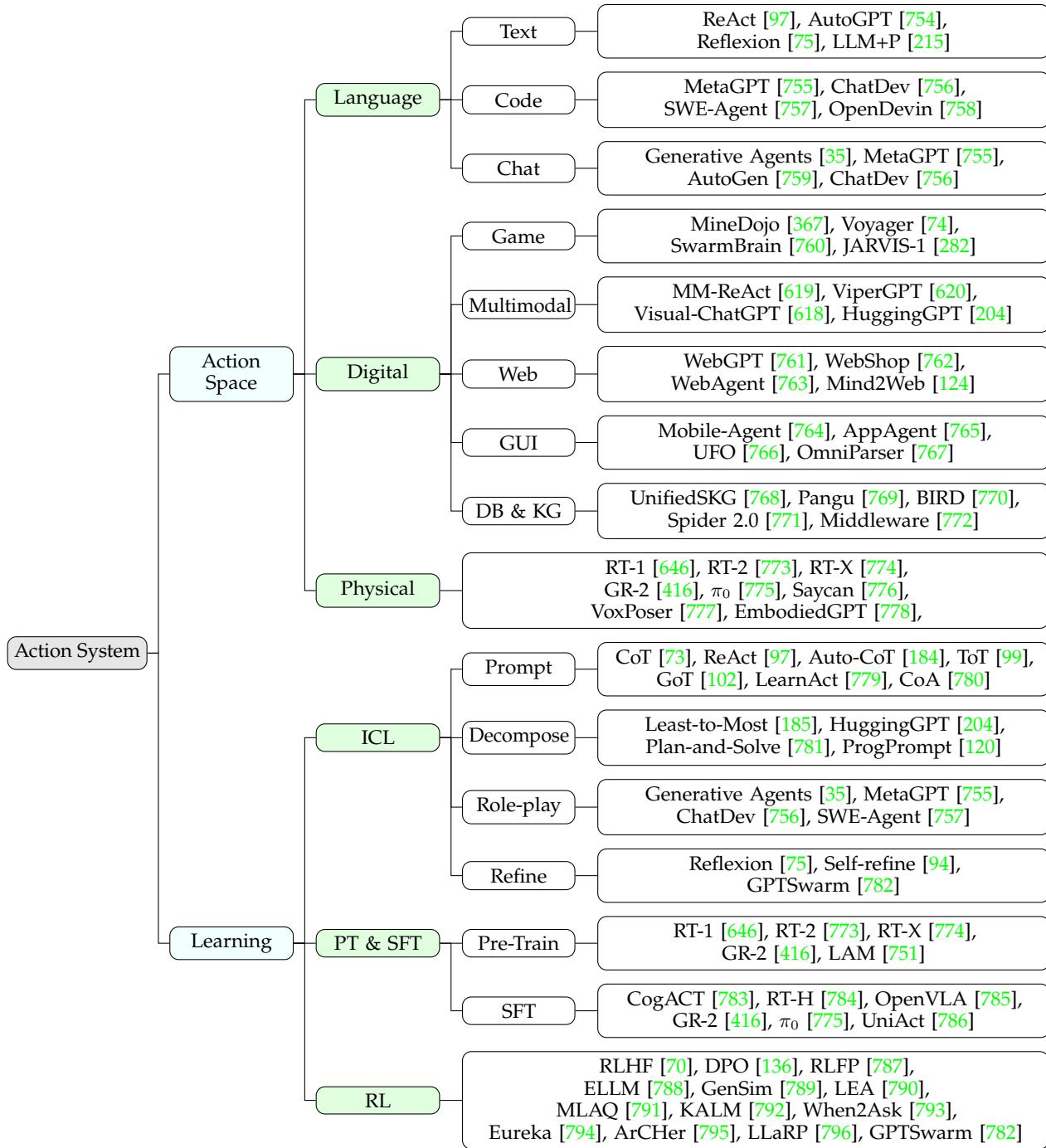


Figure 8.3: A taxonomy of selected research works about action system, including action space and learning paradigm.

8.3 Paradigms of Agentic Action System



ENERALLY, 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 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 [97] is a representative language-based AI agent, which synergizes the reasoning and actions of an LLM to solve various problems. AutoGPT [754] analyzes and decomposes user requests into multiple subtasks and uses web search or other tools to tackle each of them. Reflexion [75] involves self-refinement and the memory mechanism to enhance action execution in language tasks. LLM+P [215] 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 code as the action space, allowing direct execution of the generated code and self-verification. MetaGPT [755] and ChatDev [756] build the action space via programming language with multi-agent collaboration. SWE-Agent [757] consider different stages of software engineering and thus solve software issues. OpenDevin [758] 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 [35] directly simulate multiple characters in a virtual town, to explore how each agent to conduct next action. MetaGPT [755] and ChatDev [756] are both multi-agent frameworks to faciliate the development of software engineering. AutoGen [759] 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 [367] 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 [74] 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 [282] 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 [760] 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 [619] and ViperGPT [620] apply LLMs to perform the thinking process for multimodal tasks and then select visual experts for task solving. Visual-ChatGPT [618] integrates multiple visual experts and uses LLMs as the controller to solve tasks. HuggingGPT [204] 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 [761], WebAgent [763]) are designed to interact with search engine to enhance the capability of agent to discover the answers from website. WebShop [762] is used to explore the potential of AI Agent for online shopping. Mind2Web [124] 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 [764] utilizes multimodal models as the cognitive controller to manage and orchestrate mobile functionalities. AppAgent [765] defines various app usages as action spaces, enabling foundation models to interact with different apps as a mobile intelligent assistant. UFO [766] and OmniParser [767] 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 [769] pioneered the connection between LLMs and large-scale KGs, while BIRD [770] and Spider 2.0 [771] established a foundation for LLMs to operate with enterprise databases in real-world settings. NL2SQL-BUGs [797] addresses the critical challenge of identifying semantic errors in NL2SQL pipelines [798], which enhances the reliability of LLM-driven interactions with relational databases [799]. Similarly, frameworks like UnifiedSKG [768] and Middleware [772] 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 [646, 773, 774] pre-trained vision-language-action models to integrate knowledge from web videos into robotic learning, enhancing robotic control and action execution. GR-2 [416] 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 [775] 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 [776] 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 [777] uses LLMs to understand and decompose 3D Value Maps for Robotic Manipulation. Besides, EmbodiedGPT [778] 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 [800] 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) [73] prompting, which applies “*Let us think step by step*” technique to generate a sequence of intermediate reasoning steps, exploring potential solutions systematically. ReAct [97] 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 [779] 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 [184]) explores how to automatically generate CoT via LLMs and then enable the autonomous thinking process of AI agents. To handle more complex tasks, ToT [99] considers the thought process as a tree structure and introduces the tree search via LLM prompting, while GoT [102] applies a graph structure along with the graph search. For robotic models, CoA [780] 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 [185] is a classical prompting technique to convert user instructions into multiple subtasks. HuggingGPT [204] is a representative AI agent framework that applies task planning to transform user requirements into actionable items. Plan-and-Solve [781] directly uses LLM to make plans from user instructions and then give answers

based on the generated plans. Progprompt [120] 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 [35], MetaGPT [755], ChatDev [756], SWE-Agent [757]). Finally, some other frameworks (e.g., Reflexion [75] or Self-refine [94]) 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 [646, 773, 774], which pre-trains robotic Transformer on large-scale web and robotic data, yielding a powerful vision-language-action model. Following this policy, GR-2 [416] 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 [751] 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 [785] is built upon the Llama2 [11] language model that integrates a visual encoder based on DINOv2 [801] and SigLIP [802]. It collects a diverse set of real-world robot demonstrations from Open X-Embodiment (OXE) [803] for fine-tuning and outperforms RT-2-X [803] across different tasks, while utilizing $7\times$ fewer parameters. Building upon OpenVLA, CogACT [783] further integrate additional diffusion action module to enhance the model capability in conducting action operation over robotic tasks. Besides, some works also explore how to enable robotic model to capture action from plain language in physical world. For examples, RT-H [784] introduces a hierarchical architecture to build action space, which first predict language motions and then generate low-level actions. And π_0 [775] collected massive diverse datasets from different dexterous robot platforms, and then fine-tune the pre-trained VLMs to learn robotic actions. UniAct [786] 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 [178]. 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 [37, 804, 805, 806]. In RL paradigms, there are several classical and representative algorithms, such as Deep Q-Network (DQN) [807] and Proximal Policy Optimization (PPO) [808]. The most representative RL work that applied reinforcement learning to foundation models is InstructGPT [70], 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 [136]) proposes to directly optimize preference data through contrastive learning. Existing work [116, 809] also demonstrate the potential of scaling the RL algorithm for foundation models to produce long CoT thinking stages with impressive performance. A key technique contributing to these achievements is Reinforcement Learning with Verifiable Reward (RLVR), which utilizes the rule-based

outcome reward to apply RL on the LLM [810, 811, 162]. With the rule-based reward function, current works such as ReSearch [812], L0 [813], Agent-R1 [814], RAGEN [815] and GiGPO [816] are exploring utilizing RL to fine-tune the agents on trajectory data in agentic environments. Specifically, these approaches enable multi-turn interaction between agents and their environments, collecting feedback to iteratively refine the reasoning policy. Although RL paradigms have been successfully used to fine-tune LLMs for text generation tasks [12, 817, 70, 818], 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 [787] utilizes guidance and feedback from the policy, value, and success-reward foundation models to enable agents to explore more efficiently. Similarly, ELLM [788] utilizes large-scale background knowledge from LLMs to guide agents in efficient exploration within various environments. GenSim [789] 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 [790] 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 [791] utilizes an LLM-based world model to generate imaginary interactions and then applies Q-learning [819] to derive optimal policies from this imaginary memory. KALM [792] 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.
- 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 [793] 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 [794] leverages LLM to generate human-level reward functions with reflection, allowing agents to efficiently learn complex tasks such as anthropomorphic five-finger manipulation. ArCHer [795] 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 [796] 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 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

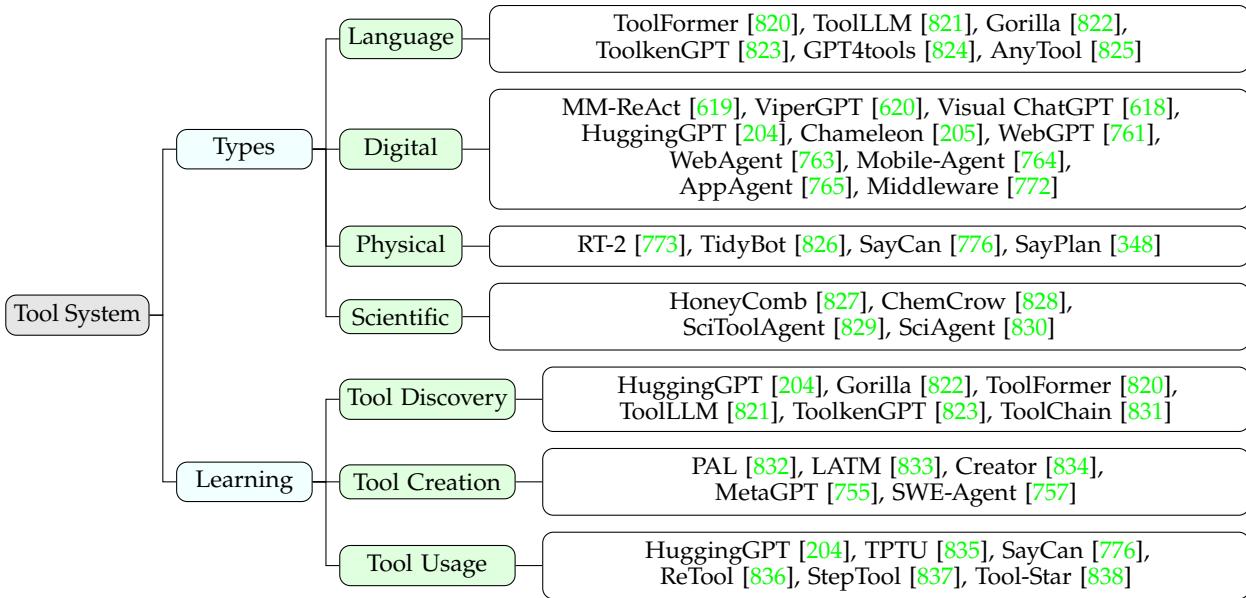


Figure 8.4: A taxonomy of tool systems in AI agents, including tool category and learning paradigm.

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 [761, 839, 124, 763], databases [840, 841, 842, 843], coding environments [844], data systems [845, 846, 847], and weather forecasting [848]. 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 [820] expands the capabilities of language models by integrating external tool spaces, including calculator, QA systems, search engine, translation, and calendar. ToolLLM [821] 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 [822] is a fine-tuned LLM based on the tool documents and then can be used to write API calls. ToolkenGPT [823] is to optimize tool embeddings and then enable LLMs to retrieve tools from the fine-tuned tool embeddings. GPT4tools [824] and AnyTool [825] 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 [619], ViperGPT [620], and Visual ChatGPT [618] employed LLMs as the controller and then used LLMs to select visual experts for solving different tasks. HuggingGPT [204] and Chameleon [205] use LLMs to first conduct reasoning and planning actions and then analyze which multimodal tools should be used for solving user instructions. WebGPT [761] and WebAgent [763] respectively empowered LLMs with search engines to enhance the capability of LLMs to solve more challenging tasks. Mobile-Agent [764] and AppAgent [765] 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 [773] demonstrates language-guided robotic manipulation using visual-language tools, and TidyBot [826] shows how LLMs adapt cleaning tools to personalized household preferences. SayCan [776] uses LLMs as the cognitive system to guide robots in solving tasks through robotic arms and visual perception. SayPlan [348] 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, [849] 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 [850, 851] 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 [827] 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 [828] 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 [829] 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 [830] 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 [852], 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:

- **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 [204] 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 [820] collects a massive corpus with the corresponding API calls (e.g., calculator, QA system, search engines, translation, and calendar) for training. ToolLLM [821] collect tool instructions based on solution paths and then fine-tune Llama models to generate better API calls for tool utilization.
- **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 [832] generates programs as intermediate reasoning steps to solve problems, LATM [833] or Creator [834] use LLMs to create code for user intentions, and to further design a verifier to validate the created tools. SciAgent [830] 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.
- **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.

To summarize, 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 [853]. 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.

8.3.4 Agent Learning Paradigm

The key to further enhance an LLM agent’s capacity for action or tool usage lies in interaction with external environments rather than training on static datasets. Therefore, the evolution of LLM agents has entered a new phase—shifting from static information processors to dynamic, interactive agents. RL has emerged as the leading methodology for training such agents, enabling them to learn optimal policies through interaction and feedback, moving beyond the limitations of supervised learning on fixed datasets.

Building on the success of RLVR [810, 811, 162], several recent works have developed sophisticated multi-turn RL training frameworks for LLM agents, leveraging rule-based reward functions to provide feedback. RAGEN [815] designs a modular multi-turn RL system for LLM agents with trajectory-level optimization, introducing the novel StarPO-S algorithm enhanced by robustness techniques such as trajectory filtering, critic incorporation, and gradient stabilization to mitigate reward variability cliffs and gradient spikes. By carefully shaping rollout generation and reasoning-aligned reward signals, the framework enables LLM agents to develop deeper reasoning patterns while significantly reducing hallucinated outputs. Building on RAGEN’s success, VAGEN [854] extends this approach by integrating visual information, thereby adapting the multi-turn RL training paradigm for VKN agents. StepSearch [855] constructs a multi-turn QA dataset and employs step-wise proximal policy optimization for RL training. By incorporating rich intermediate reward signal and fine-grained token-level process supervision, the framework empowers LLM agents more robust and effective search capabilities. ReSearch [812] integrates search operations into the reasoning process, where each search is dynamically influenced by prior reasoning steps and subsequently shapes future reasoning in multi-turn interactions. By leveraging a multi-turn RL framework, it enables LLM agents to achieve strong performance on QA tasks without requiring any supervised training data. Search-R1 [856] enables LLM agents to learn multi-query generation through step-by-step reasoning with real-time feedback. By employing a simple outcome-based reward function and retrieved token masking, the framework enhances agents’ reasoning capacity via multi-turn RL optimization, outperforming various existing RAG methods and QA tasks. WebAgent-R1 [857] modifies GRPO for multi-turn RL training and applies asynchronous trajectory rollouts. This framework incorporates a dynamic context compression mechanism, enabling LLM agents to learn effectively from direct online interactions with web environments using only binary reward signals. SkyRL [141, 858, 859] designs a high-throughput reinforcement learning pipeline for efficient multi-turn tool usage in LLM agents. The framework leverages optimized generation techniques to enable effective training on long-horizon, real-world tasks such as SWE-Bench. MT-GRPO [860] proposes a fine-grained turn-level advantage estimation strategy that enables precise credit assignment for multi-turn interactions. Similarly, GiGPO [816] introduces a novel two-level advantage estimation operation at both episode and step levels to effectively capture global trajectory quality and local action effectiveness. This dual-scale approach enables precise credit assignment in multi-turn RL training without requiring auxiliary models, additional rollouts, or extra computational overhead.

Recent advances in multi-turn RL training have demonstrated remarkable progress in enhancing LLMs’ reasoning capabilities and tool-use proficiency, though several key challenges remain. Unlike traditional RL paradigms that treat complete actions as steps, RL applied on LLM training typically treats token

generation as the fundamental step, creating a critical challenge in balancing credit assignment between token-level and turn-level advantages for long-horizon tasks. A primary limitation lies in data efficiency, as environment interactions remain computationally expensive and potentially unsafe, suggesting the need to incorporate insights from offline RL methodologies. Furthermore, current work predominantly focuses on QA search and code execution tasks where rule-based rewards are easily implemented, while many real-world applications lack clearly definable reward signals, potentially necessitating a revisit of learned reward modeling approaches. Despite these challenges, the multi-turn RL paradigm undoubtedly represents a promising direction for unifying reasoning and tool-use capabilities in LLM agents, with future work needed to address these limitations for broader real-world applicability.

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

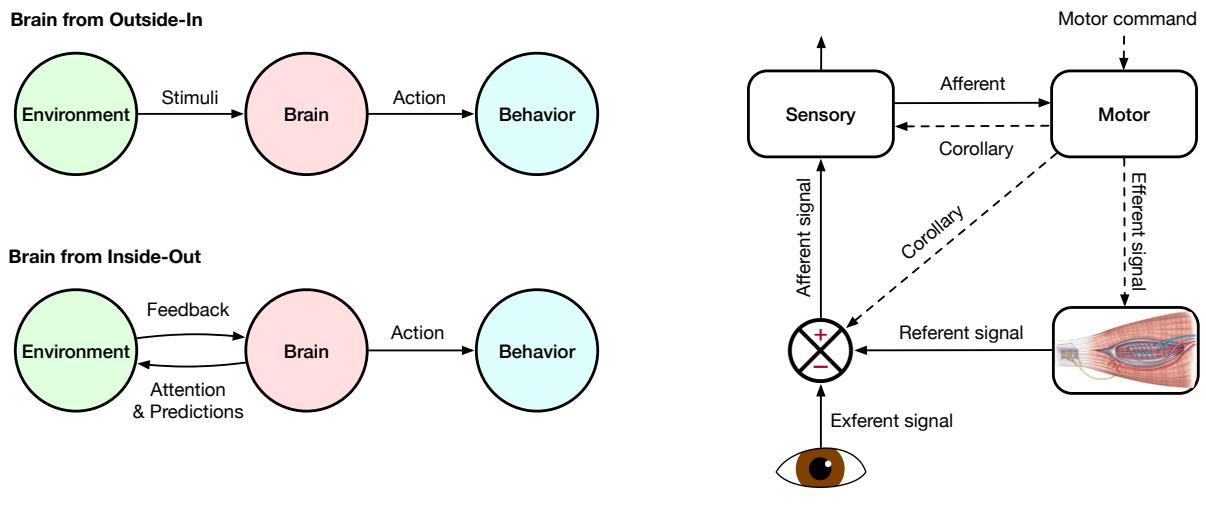


Figure 8.5: (a) Compare the brain from "outside-in" and "inside-out". (b) Illustration of the schematic of the corollary discharge mechanism. A motor command (efferent signal) travels from motor areas to the eye muscles, while a corollary discharge (dashed arrow) is routed to a comparator in the sensory 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 [45].



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 [45] 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 discharge" 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 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 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



RADITIONALLY, 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 policies 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 [848] 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:

- 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

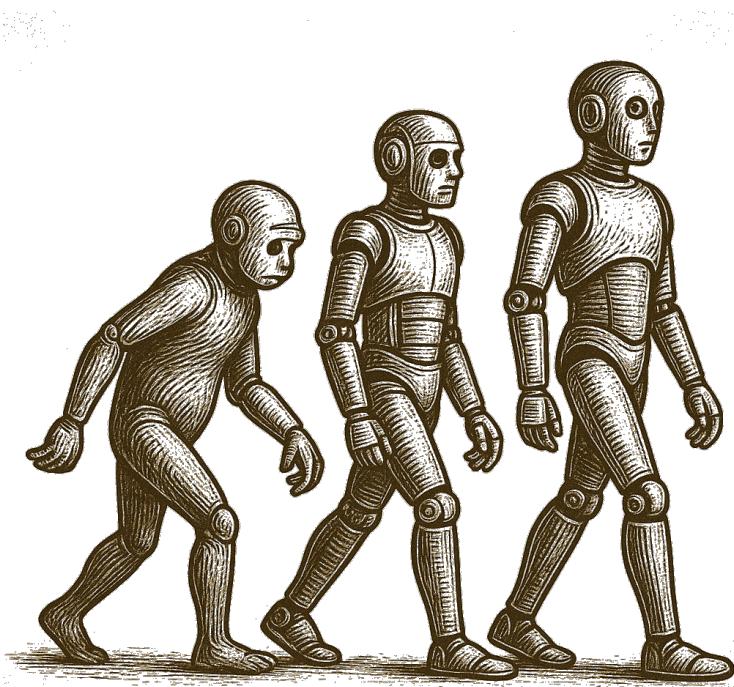
decision-making. To mitigate these efficiency issues, strategies such as filtering out irrelevant or redundant information, employing zero-shot prompting to streamline reasoning processes, and utilizing high-speed storage solutions for caching pertinent knowledge are imperative. These approaches help in maintaining high performance while reducing response times.

- **Evaluation** is also an 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.
- **Multi-modality** Action learning has achieved 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.
- **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 [861]. 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.
- **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



From construction to evolution: Agents building agents, in pursuit of autonomous intelligence.

In the history of machine learning research, manually designed AI systems have gradually been replaced by more efficient, learned solutions [862]. For instance, before the advent of deep learning, features were typically handcrafted by experts [863, 864], 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 [865]. 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) [846, 866, 867, 868, 258], 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 [869, 870, 871, 872, 873]. Among the most notable applications of AutoML is NAS [874, 875], 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 the 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 offer an alternative way to improve agent behavior by optimizing components such as strategies, tools, and workflows, without necessarily changing the underlying language model. While enhancing the core model remains essential for expanding general capabilities, self-evolution allows agents to adapt and specialize more efficiently to new tasks and environments.

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.

To move closer to the goal of automating human efforts, many recent efforts have explored using large language models (LLMs) not only as agents themselves, but also as tools for improving and evolving agentic systems [876, 877, 878]. Unlike traditional approaches that rely heavily on gradient descent or reinforcement learning, LLMs offer a flexible and expressive medium for optimization, leveraging natural language to navigate and modify a wide variety of agent components. This opens the door to new paradigms of self-evolving systems—where agents can autonomously refine their behaviors, tools, and workflows over time. A growing body of work has begun to explore this possibility, suggesting a future in which agentic systems are increasingly generated and improved by other intelligent systems, rather than handcrafted by humans.

In this part, we first introduce various *optimization spaces* explored in recent research on agentic systems, including prompts, tools, and workflows. In the subsequent chapter, we then review *optimization algorithms*, discussing both traditional optimization paradigms and meta-optimization, where the optimization process also affects the underlying optimization algorithms themselves. Next, we explore different *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

Dimensions of Self-Optimization in Intelligent Agents

HE self-improvement of autonomous agents requires optimization across several interdependent dimensions. At the foundation lies prompt optimization, which governs how agents interact with large language models and adapt to context. Building upon this layer, three key optimization axes emerge: *agentic workflow optimization*, which structures how tasks are decomposed and executed; *tool optimization*, which refines the selection and use of external capabilities; and *holistic agent optimization*, which calibrates the entire system for coherence, efficiency, and adaptability. This chapter introduces these interconnected spaces and outlines how they together support the iterative evolution of intelligent agents toward greater autonomy and competence.

9.1 Overview of Agent Optimization

T the heart of designing intelligent agents lies a fascinating challenge: teaching these systems how to evolve and improve themselves autonomously. Much like humans learning new skills, agents need structured ways to reflect on their performance and adjust their strategies accordingly. Agent optimization addresses precisely this challenge, focusing on enhancing the effectiveness, efficiency, and adaptability of agentic AI through systematic self-improvement.

Today's Large Language Model (LLM)-based agents operate within a layered optimization landscape. At its foundation is *prompt optimization*, akin to fine-tuning how clearly we communicate instructions to a colleague. Prompt optimization governs how an agent formulates its queries to language models, fundamentally shaping its responses and behaviors. Building upon prompt optimization, three distinct yet interwoven pathways unfold: First, we have *workflow optimization*. Think of an agentic workflow as a team collaborating on a complex project. Each team member (or node) has a specialized role, and the workflow defines how they coordinate tasks and share information. Optimizing workflows thus involves refining the interactions among multiple LLM components, ensuring smooth communication, effective task decomposition, and coherent execution. Next is *tool optimization*. Just as skilled craftsmen rely on high-quality, precisely selected tools, agents leverage external resources—ranging from databases to specialized software functions—to solve tasks more effectively. Tool optimization allows agents to select and utilize these resources intelligently, learning to choose or even create tools dynamically as tasks evolve. Finally, we consider *holistic agent optimization*. This broader perspective treats the entire agent as a unified system, recognizing that simply enhancing individual components—prompts, tools, or workflows—does not guarantee optimal overall performance. Holistic

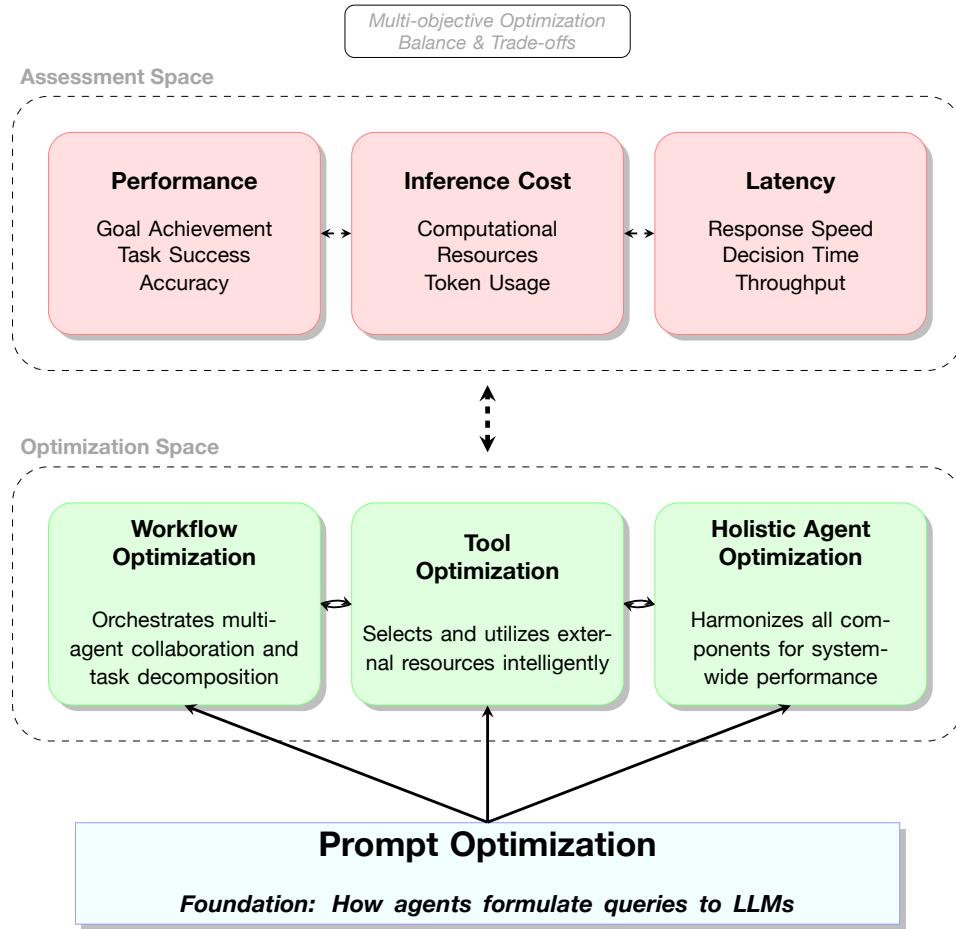


Figure 9.1: Illustration of the multi-layered optimization framework for intelligent agents.

optimization thus aims to harmonize these elements, balancing their strengths and weaknesses to ensure the agent performs reliably across diverse scenarios. Figure 9.1 shows an overview of the optimization landscape of agents.

Optimizing intelligent agents also parallels traditional approaches found in automated machine learning (AutoML). Just as AutoML tackles trade-offs between accuracy, computational efficiency, and deployment latency, agent optimization must similarly navigate multiple competing objectives. Typically, we assess agents along three critical axes: *performance* (how well the agent achieves its goals), *inference cost* (the computational resources consumed), and *latency* (the speed of decision-making). Each of these metrics can dominate in different contexts—for example, a customer support chatbot prioritizes rapid responses, whereas a scientific discovery agent values accuracy above speed. Yet, optimizing for multiple objectives simultaneously introduces complexity. It forces us to balance conflicting priorities, demanding thoughtful strategies that transcend simple performance improvements. This complexity reflects real-world challenges where optimal decisions rarely maximize one factor alone but instead find intelligent compromises across several dimensions.

Throughout this chapter, we will explore each optimization dimension in depth, providing the necessary insights and techniques to guide the evolution of truly autonomous and adaptive agents.

9.2 Prompt Optimization

ROMPT OPTIMIZATION plays a crucial 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 context, the objective of prompt optimization is to generate a task-specific prompt P_t^* that maximizes performance:

Definition 9 (Prompt Optimization). We can formally define the process of prompt optimization as:

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

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.

Figure 9.2 gives an overview about prompt optimization. Next, we will introduce each part in details.

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 [879, 880, 881, 882, 877, 360] predominantly leverage comparisons between G_{llm} and G_t as evaluation sources. Some approaches [883, 884, 885] utilize only G_{llm} as the evaluation source. For instance, PROMST [884] assesses prompt effectiveness by comparing G_{llm} against human-crafted rules; SPO [886] 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 [879, 880, 884, 877, 360]. 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 [718] can assess task completion quality based on task descriptions and prompt outputs G_{llm} , providing reflective textual gradient feedback. Notable implementations include ProteGi [887], TextGrad [881], Semantic Search [882] and Revolve [888]. Furthermore, LLM-as-a-judge enables

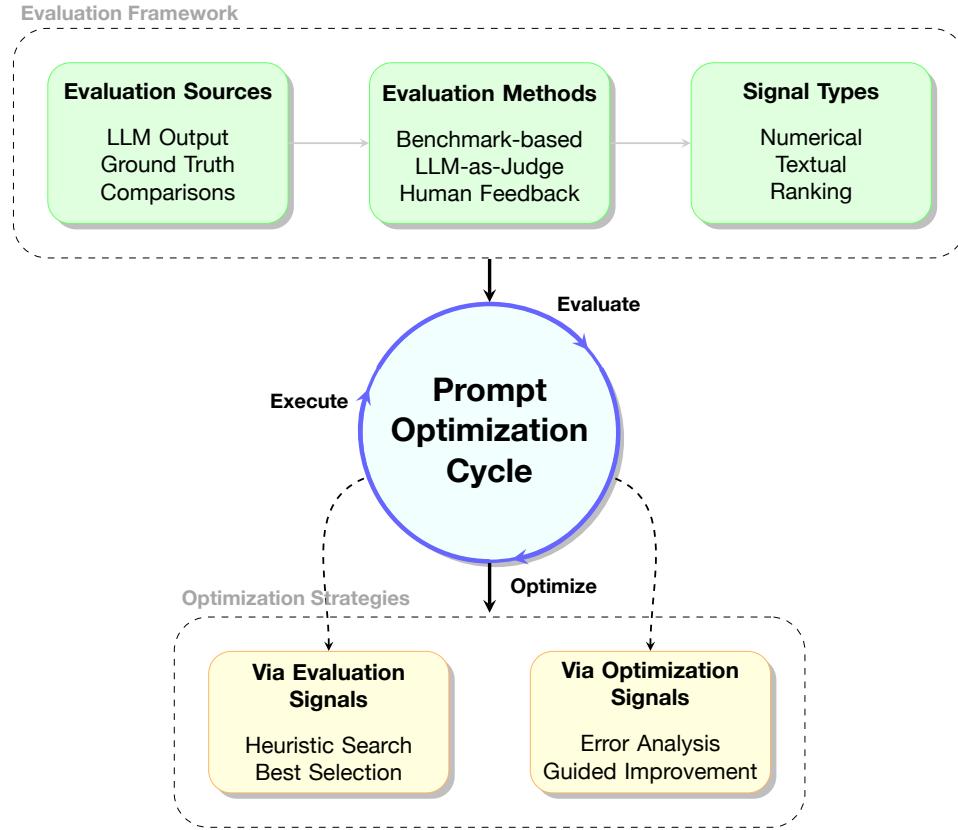


Figure 9.2: The prompt optimization cycle consists of three core functions: Optimize, Execute, and Evaluate. The evaluation framework (top) processes sources, methods, and signal types to generate evaluation and optimization signals that guide two optimization strategies (bottom). Arrows indicate the iterative, feedback-driven flow of the optimization process.

comparative evaluation between ground truth G_t and output G_{llm} with specific scoring mechanisms [889]. The effectiveness of this method hinges on both the design of judge prompts and the underlying model's alignment with human preferences. As a specialized extension, *Agent-as-a-Judge* [890] 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 [885] 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* [879, 880, 884, 877, 360] 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* [881, 882, 888] 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* [886] 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 [886] iteratively refines prompts based on the outputs of the current best-performing ones, leveraging the language model's inherent ability to align with task requirements. Similarly, Evoprompt [891] employs evolutionary algorithms with LLMs serving as evolution operators for heuristic prompt combination. PromptBreeder [877] 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 [879] extracts common patterns from high-performing prompt solutions to guide subsequent optimization steps. ProTegi [887] employs language models to analyze failure cases and predict error causes, using these insights as optimization guidance. TextGrad [881] extends this approach further by transforming prompt reflections into “textual gradients”, applying this guidance across multiple prompts within agentic systems. Revolve [888] 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* [892, 886, 879] 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* [886]. 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* [883] measures output stability across

multiple runs, *fairness* [893] evaluates the ability to mitigate the language model’s inherent biases, and *confidence* [894, 895] 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 [755], AlphaCodium [896], 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 [782] and AFLOW [876], 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

Here we present a formal definition for agentic workflow.

Definition 10 (Agentic Workflow). An agentic workflow \mathcal{K}_{wf} can be formally represented as:

$$\mathcal{K}_{\text{wf}} = \{(N, E) | N \in \mathcal{N}, E \in \mathcal{E}\} \quad (9.2)$$

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 \mathcal{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 \mathcal{O} , the goal of workflow optimization is to discover the optimal workflow K_{wf}^* that maximizes performance:

$$K_{\text{wf}}^* = \arg \max_{K_{\text{wf}} \in \mathcal{K}_{\text{wf}}} \mathcal{O}(K_{\text{wf}}, T), \quad (9.3)$$

where \mathcal{K}_{wf} is the workflow search space, K_{wf} is a candidate workflow drawn from that space, and $\mathcal{O}(K_{\text{wf}}, T)$ jointly measures task-completion quality, computational efficiency, and execution latency.

Figure 9.3 gives an overview about workflow optimization. Next, we will introduce each part with more details.

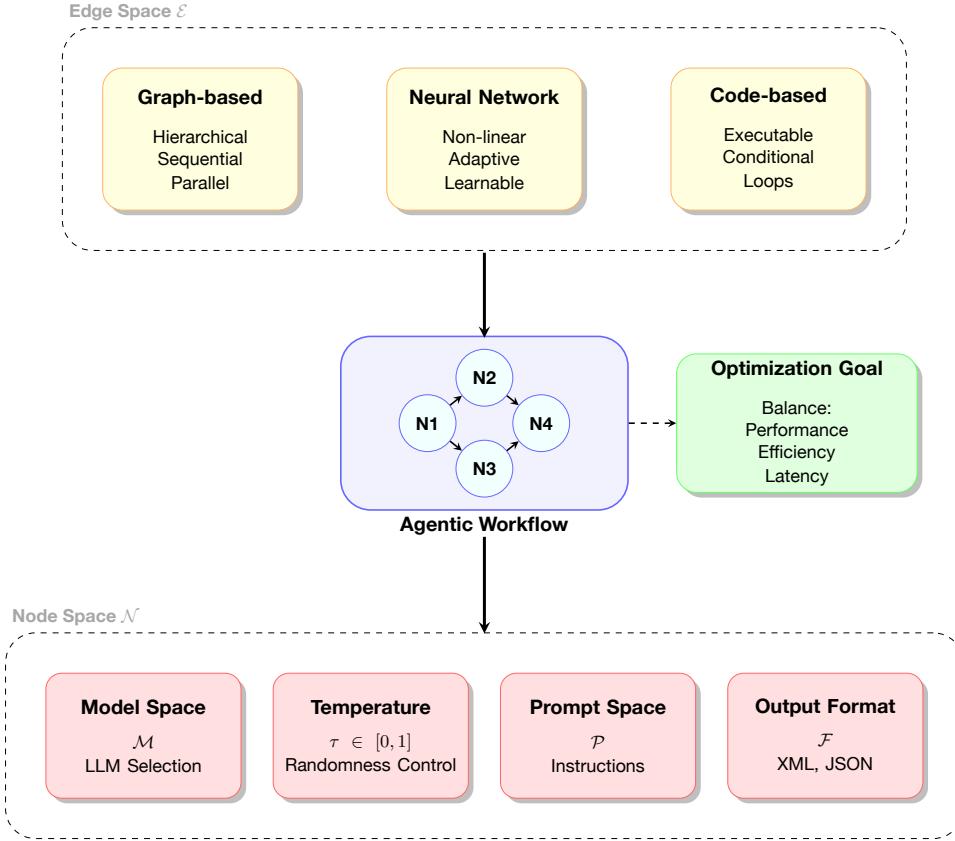


Figure 9.3: Framework for optimizing agentic workflows composed of LLM-invoking nodes and edges. The edge space (top) offers three representation paradigms: graph-based, neural network-based, and code-based structures. The node space (bottom) encompasses four optimization dimensions: model selection, temperature, prompts, and output format. The optimization goal balances performance, efficiency, and latency.

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 [782] 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 [897] 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 [876] and ADAS [898] 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. [881] focused solely on prompt optimization while maintaining a fixed workflow topology, enabling the use of textual feedback-based optimization techniques. In contrast, [782] developed reinforcement learning algorithms for joint optimization of individual node prompts and overall topology. [876] leveraged code-based representation to enable direct workflow optimization by language models, while recent advances by [899] and [900] introduced methods for problem-specific topology optimization.

Recent advances in this area further demonstrate the trend of treating the workflow itself as a dynamic, optimizable program. MAS-GPT [901], by fine-tuning a language model, can directly generate complete, executable multi-agent system code from a user query. FlowReasoner [902] utilizes a meta-agent that learns and iterates on a workflow’s design based on performance feedback after each task execution. Still others achieve zero-supervision adaptive design; MAS-ZERO [903] uses a self-supervised iterative loop at inference-time to continuously generate, evaluate, and refine the workflow without pre-existing training or labeled data. Together, these pioneering methods are driving a paradigm shift in workflow optimization—moving from designing fixed topologies to the automated, contextual, and adaptive construction of workflows using techniques from generative AI, reinforcement learning, and self-supervised learning.

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. [876] concentrated exclusively on prompt optimization, while [898] extended the search space to include both prompts and temperature parameters. Taking a different approach, [904] fixed prompts while exploring model selection across different nodes. Output format optimization, though crucial, remains relatively unexplored [905].

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



UNIQUE 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*. Figure 9.4 shows an overview about tool optimization.

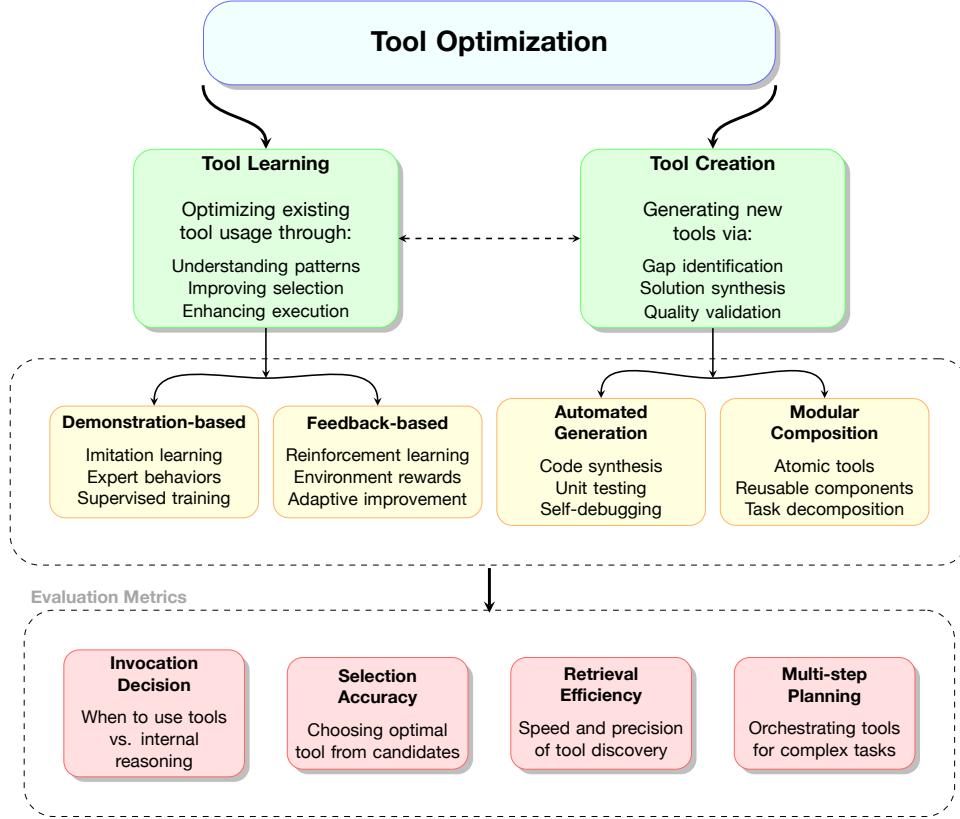


Figure 9.4: Tool optimization operates through two strategies: Tool Learning (optimizing existing tools via demonstrations and feedback) and Tool Creation (generating new tools through automated synthesis and modular composition). Different evaluation metrics assess tool usage.

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* [848]. 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.

Definition 11 (Tool Learning). Tool learning refers to the general process by which an LLM-based agent or controller acquires and improves its ability to utilize tools, based on data, demonstrations, or feedback signals, to maximize task performance. Formally, let θ_C denote the parameters of the controller (or policy). The general objective of tool learning can be expressed as:

$$\theta_C^* = \arg \max_{\theta_C} \mathbb{E}_{\mathcal{D}, \mathcal{R}} [\mathcal{J}(\theta_C; \mathcal{D}, \mathcal{R})], \quad (9.4)$$

where \mathcal{D} denotes the data source (e.g., demonstrations or interactions), \mathcal{R} represents the reward or evaluation signals, and \mathcal{J} is the performance objective (such as log-likelihood of expert actions or expected reward).

Learning From Demonstrations: This approach trains the controller to imitate expert demonstrations via supervised learning or behavior cloning. 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-annotated action sequence, the parameters θ_C are optimized as:

$$\theta_C^* = \arg \max_{\theta_C} \mathbb{E}_{(q_i, a_i^*) \in D} \prod_{t=0}^{T_i} p_{\theta_C}(a_{i,t}^* | x_{i,t}, H_{i,t}, q_i), \quad (9.5)$$

where $a_{i,t}^*$ is the expert action at step t for query q_i , $x_{i,t}$ denotes the controller's local observation or state at time t (e.g., current tool context or environment state), $H_{i,t}$ is its hidden state / action–observation history up to t , p_{θ_C} is the policy distribution (i.e., the probability assigned by the controller to each feasible action conditioned on the current observation, history, and query), and T_i is the total number of timesteps.

Learning From Feedback: This approach leverages reinforcement learning to adapt the controller based on reward signals from the environment or human feedback. The optimization objective is:

$$\theta_C^* = \arg \max_{\theta_C} \mathbb{E}_{q_i \in Q} \mathbb{E}_{\{a_{i,t}\}_{t=0}^{T_i}} [R(\{a_{i,t}\}_{t=0}^{T_i})], \quad (9.6)$$

where R denotes the reward function based on the sequence of actions $\{a_{i,t}\}$, and Q is the distribution of user queries.

Recently, inspired by the success of reinforcement-learning methods such as GRPO in DeepSeek-R1 [116], researchers have begun to optimize tool use by directly drawing rewards from the environment, capturing factors like output-format fidelity and answer correctness across the full tool-calling pipeline (tool retrieval, parameter selection, and execution success). Search-R1 [82] refines search-engine usage, ToolRL [906] introduces reward-driven tool learning, OTC-PO [907] tightens the link between acting and reasoning, and Nemotron-Research-Tool-N [908] demonstrates scenario-specific gains. Collectively, these studies highlight reinforcement learning as a powerful method for adaptive, task-aware tool optimization for Foundation Agents.

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) [73], Tree-of-Thought [99], and Depth-First Search Decision Trees (DFS-DT) [821], 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 [834, 833, 878] 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 [833] 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 [834] 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 [878] 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. Alita [169] frames “tool creation” as a compact, self-optimizing loop: the agent spots a capability gap, drafts a spec, mines code examples, autowrites and executes a script in an isolated sandbox, then packages the passing script as a standard MCP module. Each gap thus yields a ready-to-use tool, allowing the agent’s arsenal to grow cumulatively with every task it tackles.

Hybrid frameworks can fuse CRAFT’s ready-made tool pool with ToolMaker’s on-demand generation, while Alita’s self-optimizing loop (gap-detect → search → build → validate → wrap) adds continuous, data-driven expansion. Primitive CRAFT operations can seed higher-level ToolMaker / Alita composites, with CREATOR-style rectifiers covering edge cases. Progress in self-supervised tool metrics and cross-domain transfer will further reduce the need for human oversight. A key open question is how tool granularity affects reuse: finer, “atomic” tools offer flexible composition but increase orchestration costs. Ultimately, bidirectional tool-task co-adaptation—where evolving tasks inspire the development of new tools and new tools redefine tasks—points toward truly self-improving agent ecosystems.

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 [909] and API-Bank [910] 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 [911] and ToolBench [821] introduced more extensive task suites and stressed the importance of systematic data generation for tool manipulation. StableToolBench [912] 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 [913] investigated the feasibility of achieving generalized tool-use in relatively smaller language models with minimal in-domain training. Additional efforts like ToolEmu [914] assessed the safety and risk aspects of tool-augmented LM agents through emulated sandbox environments. MetaTool [915] 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 [916] pushed the evaluation further by examining real-world scenarios and multi-step reasoning across a large tool library. Finally, τ -bench [917] 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

$$D_{\text{inv}} = \{(q_i, y_i)\}_{i=0}^{N-1}, \quad (9.7)$$

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:

$$d(q_i) = \begin{cases} 1, & \text{if } P_\theta(y = 1 | q_i) \geq \tau, \\ 0, & \text{otherwise,} \end{cases} \quad (9.8)$$

where $P_\theta(y = 1 | q_i)$ 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:

$$A_{\text{inv}} = \frac{1}{N} \sum_{i=0}^{N-1} \mathbf{1}\{d(q_i) = y_i\}, \quad (9.9)$$

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:

$$B_{\text{inv}} = \sum_{i=0}^{N-1} (\mathbf{1}\{d(q_i) = 1\} \cdot R(q_i) - C_{\text{inv}}). \quad (9.10)$$

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:

$$\mathcal{T} = \{t_1, t_2, \dots, t_M\}. \quad (9.11)$$

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 :

$$A_S = \frac{1}{|Q|} \sum_{q_i \in Q} \mathbf{1}\{\hat{t}_i = t_i^*\}. \quad (9.12)$$

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. [821] 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:

$$E_{Ret} = \frac{A_R \times A_S \times A_P \times A_U}{C_R} \quad (9.13)$$

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 [915] 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

$$\Pi = [t_1, t_2, \dots, t_K], \quad (9.14)$$

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 $R_{task}(\Pi)$) 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 [916] 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

$$S_{BP} = S_{b\text{-validity}} \cdot S_{b\text{-integrity}}, \quad (9.15)$$

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 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.

Among the most representative works that first formally defined the research problem of automated design in agentic systems are ADAS [898] and AFLOW [876]. For instance, ADAS integrates multiple agentic system components—including the workflow, prompts, and potential tools—into its evolutionary pipeline. Similarly, AFLOW automates the generation and optimization of entire agentic system workflows.

Following a similar evolutionary paradigm, AlphaEvolve [170] employs an evolutionary algorithm driven by LLMs to iteratively modify and improve entire codebases for scientific and algorithmic discovery. This approach has proven effective in tackling complex scientific problems where progress can be systematically measured. Another novel mechanism to drive evolution is self-challenging. For instance, Self-Challenging Agents (SCA) [918] boost their own evolution by generating high-quality, verifiable tasks through interaction with the environment, creating a curriculum for continuous self-improvement.

Additionally, Zhou et al. [919] 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.

Yin et al. [920] 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. A more profound implementation of this idea is the Darwin Gödel Machine [171], which iteratively rewrites its own python codebase to evolve new versions of itself with improved tools and workflows. Similarly, Alita [169] demonstrates the possibility of an agent’s action space self-evolving in any environment by naturally constructing and encapsulating actions into reusable modules through interaction with the environment.

Similarly, Zhang et al. [876], Zhang et al. [899], Zhou et al. [921], and Wang et al. [900] focus on optimizing both the workflow and prompts within agentic systems. In particular, Wang et al. [339] 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.

9.6 Summary and Discussion

 N this chapter, we have explored the multidimensional landscape of self-optimization in intelligent agents, highlighting four crucial optimization spaces: prompt optimization, workflow optimization, tool optimization, and holistic agent optimization. Each dimension plays a unique yet interrelated role in enabling agentic systems to evolve towards greater autonomy, adaptability, and effectiveness.

Prompt optimization forms the foundational layer, directly influencing how agents query and interact with large language models. We detailed evaluation methods, including benchmark-based evaluations, LLM-as-a-Judge techniques, and human feedback mechanisms, emphasizing their varying trade-offs between automation, accuracy, and scalability. Effective prompt optimization thus involves careful selection and combination of these evaluation strategies to ensure efficient, context-aware agent communication.

Workflow optimization extends beyond single prompts, addressing the coordination of multiple LLM components. We formally introduced agentic workflows, discussing their node and edge components and the corresponding optimization search spaces. Graph-based, neural network-based, and code-based representations each offer distinct advantages, influencing the complexity and flexibility of workflow optimizations. As highlighted, recent advancements increasingly automate workflow topology and node parameter optimizations through reinforcement learning, generative AI, and self-supervised learning.

Tool optimization emphasizes enhancing how agents select, invoke, and create external tools. Two complementary strategies, learning from demonstrations and feedback, were discussed, alongside tool creation methods that automate tool synthesis and modular composition. The role of evaluation benchmarks and metrics—spanning tool invocation accuracy, selection precision, retrieval efficiency, and complex task planning—was underscored as critical for measuring and guiding tool optimization efforts.

Finally, holistic agent optimization integrates all dimensions into unified evolutionary frameworks. By addressing the complex interplay between prompts, workflows, and tools, holistic methods overcome local optima and facilitate comprehensive system improvements. We examined representative approaches like ADAS, AFLOW, and AlphaEvolve, along with self-referential methods such as the Darwin Gödel Machine and Alita, showcasing diverse strategies for systemic agent evolution.

Despite significant progress, several challenges and open research questions remain. Balancing trade-offs between computational cost, optimization accuracy, and scalability continues to be an ongoing challenge. Developing adaptive evaluation and optimization mechanisms that dynamically respond to evolving agent goals and environments presents a critical frontier. Future research directions may also explore deeper integrations between neural, symbolic, and evolutionary approaches to agent self-optimization.

In conclusion, advancing self-optimization capabilities is pivotal for realizing genuinely autonomous and adaptable intelligent agents. Continued innovation in optimization methods, coupled with systematic evaluation frameworks, will undoubtedly drive substantial advancements in the field, paving the way for agents capable of sophisticated, self-directed growth.

Chapter 10

Harnessing Large Language Models for Iterative Optimization

 N this chapter, we explore a rapidly evolving perspective: viewing large language models (LLMs) as powerful iterative optimizers. While classical optimization techniques typically rely on numerical gradients or heuristic search, recent studies demonstrate that LLMs offer a fundamentally different paradigm by leveraging their natural language processing capabilities for optimization tasks. We begin by contrasting traditional gradient-based and zeroth-order optimization methods with the emerging class of LLM-based optimizers, highlighting the unique advantages and complexities introduced by natural language. We then delve into iterative approaches that extend classical optimization concepts, such as random search, gradient approximations, and surrogate modeling, into the discrete and structured domains characteristic of LLM-driven workflows. Additionally, we examine critical hyperparameter considerations and discuss strategies for dynamic optimization across workflow depth and iterative refinements over time. Finally, we provide a theoretical lens, surveying recent insights into in-context learning, mechanistic interpretability, and limitations of LLM optimization in uncertain environments, bridging the gap between empirical success and theoretical understanding. Through this discussion, we aim to equip readers with a comprehensive foundation to understand and effectively leverage LLMs as versatile optimization tools within intelligent agent systems.

10.1 Optimization Paradigms

 PTIMIZATION methods vary significantly in terms of the assumptions they make about the information available from the function they aim to optimize. Broadly speaking, these methods can be organized into three progressively broader categories: *gradient-based optimization*, which explicitly utilizes gradients; *zeroth-order optimization*, which estimates directions without gradient information; and the emerging paradigm of *LLM-based optimization*, which transcends traditional numerical approaches by optimizing directly within complex, structured, and often linguistic spaces.

- **Gradient-Based Optimization.** These classical optimization approaches hinge upon the availability of precise gradient information. Techniques such as stochastic gradient descent (SGD) and Newton's method [922] are prominent examples, favored for their efficiency and convergence properties. However, their reliance on differentiability significantly limits their applicability. For instance, discrete problems such as prompt optimization or structured workflow refinement, which frequently

involve graph-based representations or categorical variables, pose fundamental challenges to gradient computation.

- **Zeroth-Order Optimization.** Addressing scenarios where explicit gradients are difficult, costly, or impossible to compute, zeroth-order methods rely solely on function evaluations to guide their search directions [923]. Popular methods in this class include Bayesian optimization [924], evolutionary strategies [925], and finite-difference approximations [926]. These approaches excel when gradients are inaccessible, offering robust solutions for black-box optimization problems. Yet, their utility remains restricted to numerical objectives and structured spaces, proving less effective in dealing with inherently linguistic or unstructured domains.
- **LLM-Based Optimization.** Emerging at the intersection of language modeling and optimization, large language model (LLM)-based optimization fundamentally expands the scope of optimization methods. Unlike traditional numerical optimization, LLMs operate naturally in complex linguistic environments, seamlessly integrating reasoning, iterative feedback, and human-like adaptability. This capability makes them particularly suited for tasks involving prompt refinement, adaptive workflow generation, and iterative decision-making that respond directly to linguistic cues and user feedback.

While gradient-based and zeroth-order methods predominantly address numerical problems, their foundational principles—iterative refinement, heuristic search, and adaptive learning—provide valuable conceptual insights for the nascent field of LLM-based optimization. Building upon this conceptual bridge, recent advances have integrated reinforcement learning techniques with LLMs, forming the foundation of sophisticated reasoning frameworks known as “slow thinking” models [117, 927, 116]. As these LLM-powered approaches evolve, we foresee a profound impact on the design of intelligent agents, empowering them to navigate increasingly complex and dynamic environments with enhanced flexibility, strategic planning, and decision-making prowess.

10.2 Iterative Approaches to LLM Optimization

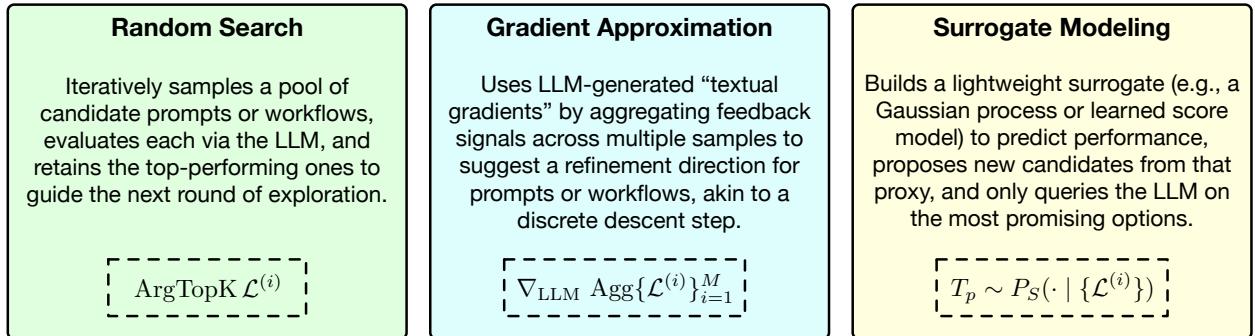


Figure 10.1: Three core iterative LLM-based optimization strategies: (Left) Random Search samples and retains top-K candidates; (Center) Gradient Approximation uses aggregated LLM “textual gradients” for discrete refinement; (Right) Surrogate Modeling fits a proxy to predict performance and only evaluates the most promising candidates.

ome 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 Definition 9 as a running example, a general iterative algorithm can be expressed as follows:

$$\text{Sample: } T \sim \mathcal{D} \quad (10.1)$$

$$\text{Evaluation: } \mathcal{L}(T; T_p) \leftarrow \phi_{\text{eval}}(\phi_{\text{exe}}(Q, T_p), T) \quad (10.2)$$

$$\text{Update: } T'_p \leftarrow \phi_{\text{opt}}(\mathcal{L}(T; T_p)) \quad (10.3)$$

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. Figure 10.1 illustrates three strategies—random search, gradient approximation, and surrogate modeling—for iteratively refining LLM-based agents.

Random Search Early LLM-based optimization methods leveraged random search variants to optimize prompts in discrete natural language spaces [880, 928, 782, 877, 929, 930, 931]. 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:

$$\text{Sample: } T \sim \mathcal{D} \quad (10.4)$$

$$\text{Evaluation: } \mathcal{L}^{(i)} \leftarrow \phi_{\text{eval}}(\phi_{\text{exe}}(Q, T_p^{(i)}), T), \quad i = 1, \dots, M \quad (10.5)$$

$$\text{Update: } \{T_p^{(k)'}\}_{k=1}^K \leftarrow \text{ArgTopK}_{i \in [M]} \mathcal{L}^{(i)}, \quad (10.6)$$

$$\text{Replenishment (Optional): } \{T_p^{(j)}\}_{j=K+1}^M \sim \text{Mutate}(\{T_p^{(k)}\}_{k=1}^K). \quad (10.7)$$

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 [932, 933]. 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, [887, 879, 881] generate refinements at different workflow stages. StraGO [934] estimates descent directions using central-difference heuristics, while Trace [935] 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:

$$\text{Sample: } T^{(i)} \sim \mathcal{D}, \quad i = 1, \dots, M \quad (10.8)$$

$$\text{Evaluation: } \mathcal{L}^{(i)} \leftarrow \phi_{\text{eval}}(\phi_{\text{exe}}(Q, T_p), T^{(i)}), \quad i = 1, \dots, M \quad (10.9)$$

$$\text{Gradient Approximation: } g \leftarrow \nabla_{\text{LLM}} \text{Agg} \left(\mathcal{L}^{(1)}, \dots, \mathcal{L}^{(M)} \right) \quad (10.10)$$

$$\text{Update: } T'_p \leftarrow \phi_{\text{opt}}(T_p, g), \quad (10.11)$$

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” [881] 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 [936, 937], and they facilitate backpropagation-like techniques for optimizing computation graphs [782, 935, 888], 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 [938, 939, 940]. While effective for simple input-output sequence learning, these methods struggle with multi-step workflows and sequential decision-making [759, 360].

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 [888] 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 [117, 116] 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 [941, 942]. Under these constraints, Bayesian Optimization (BO) emerges as a compelling alternative, as it builds a noise-resilient surrogate model of the optimization objective:

$$\text{Sample: } T \sim \mathcal{D} \quad (10.12)$$

$$\text{Proposal: } \{T_p^{(i)}\}_{i=1}^M \sim S. \text{Propose} \quad (10.13)$$

$$\text{Evaluation: } \mathcal{L}^{(i)} \leftarrow \phi_{\text{eval}}(\phi_{\text{exe}}(Q, T_p^{(i)}), T), \quad i = 1, \dots, M \quad (10.14)$$

$$\text{Update: } S \leftarrow S. \text{UpdatePrior}(\{\mathcal{L}^{(i)}\}_{i=1}^M, \{T_p^{(i)}\}_{i=1}^M), \quad (10.15)$$

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 [924]) and an update mechanism based on observed evidence from prompt evaluations. For instance, MIPRO [943] employs a Tree-Structured Parzen Estimator as its surrogate [944], while PROMST [945] 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 [946]. For instance, [947] 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 [948, 949]. 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 [950, 947].

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, [935] 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 [881] 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 [881, 935]. 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 [951], optimal decision theory [948], and game theory [949]. 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 [952]. Moreover, amortized approaches train auxiliary models to predict effective hyperparameters, which can reduce the computational cost of exhaustive search [943, 945]. 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 Dynamic and Iterative Optimization in LLM Workflows



ONVENTIONAL optimization techniques typically view parameter updates as static procedures within clearly defined boundaries. In stark contrast, LLM-based optimization embraces a dynamic approach, adapting not only across the hierarchical depth of workflows but also iteratively over successive time steps. This dual perspective—considering both depth (single-pass sequential execution) and temporal dimensions (iterative refinement)—is central to leveraging the full potential of LLM-driven optimization methods.

At a fundamental level, optimizing across depth resembles how feedforward neural networks process data: inputs traverse multiple layers, each performing incremental transformations that collectively refine the output. Analogously, LLM-based workflows pass sequentially through various modules or stages, each step benefiting from structured reasoning and cumulative context provided by previous modules. Such depth-wise optimization naturally aligns with classical computational architectures, enabling well-defined, modular improvements at each workflow stage.

However, the true strength of LLM-based optimization emerges clearly when considering its temporal aspect—iterative optimization over multiple cycles. In this respect, LLM-based methods draw conceptual parallels with recurrent neural architectures, such as RNNs and Universal Transformers [953], which iteratively refine their internal representations based on evolving inputs and previous states. Similarly, LLM-driven optimizers dynamically update their decision-making strategies by incorporating feedback from prior outcomes, progressively enhancing their performance over multiple cycles. StateFlow [951] exemplifies this iterative refinement by integrating previous execution outcomes and feedback loops into workflow decisions, dynamically adjusting strategies to evolving task contexts.

Yet, despite these promising analogies, there remain significant untapped opportunities to incorporate well-established optimization strategies from engineering domains into the LLM optimization landscape. Techniques such as checkpointing—saving intermediate states to efficiently revisit previous optimization states—and truncated backpropagation—focusing computational resources on the most recent and relevant optimization steps—have demonstrated substantial improvements in traditional deep learning and numerical optimization contexts [954, 955]. Exploring analogous implementations of these methods within the LLM optimization paradigm could enhance computational efficiency, stability, and effectiveness, especially in complex or long-term iterative optimization tasks.

Moreover, the dynamic and iterative optimization process introduces unique challenges, such as balancing exploration and exploitation, managing cumulative uncertainty across iterations, and ensuring long-term coherence in evolving strategies. Future research must investigate methods for systematically integrating these temporal considerations into LLM-driven workflows. This exploration could include adaptive strategies that dynamically allocate computational resources based on uncertainty and historical performance, hybrid optimization techniques combining classical numerical methods with LLM-driven updates, and frameworks explicitly modeling temporal dependencies and long-range coherence. In summary, while depth-wise optimization aligns naturally with conventional computational strategies, the temporal dimension opens rich, largely uncharted territory. By embracing iterative refinement over time and carefully integrating lessons from classical engineering optimization, LLM-based methods stand poised to achieve unprecedented levels of adaptability, efficiency, and strategic depth in complex agentic applications.

10.5 Theoretical Insights into Transformer Optimization



RECENT research increasingly highlights transformers not merely as sophisticated language processors but as systems inherently capable of performing complex, optimization-like computations. Despite their remarkable empirical successes, our theoretical understanding of how and why transformers

can function effectively as optimizers remains incomplete. This section synthesizes key theoretical advancements that bridge this gap and explores fundamental concepts underpinning transformers’ optimization abilities.

In-Context Learning In-context learning, particularly prominent in few-shot learning scenarios [2], provides essential insights into transformers as optimization mechanisms. Early theoretical work by [956] illustrated that transformers can internally replicate various regression and learning models, including linear regression, decision trees, and shallow neural networks, purely through their attention mechanisms. Subsequent studies [957, 958, 959] further strengthened this view by explicitly demonstrating how transformers can mimic iterative optimization methods, including gradient descent and higher-order update strategies. Nonetheless, while such theoretical analyses reveal profound computational capacities, they only partially explain in-context learning behaviors exhibited by large-scale language models, especially given their discrete, linguistic input-output settings. Empirical investigations [941, 960, 942] have thus attempted to decode in-context generalization mechanisms. For example, Xie et al. [960] proposed interpreting in-context learning as akin to implicit Bayesian inference performed by hidden Markov models, whereas others [941, 942] questioned traditional assumptions, suggesting alternative explanations. Consequently, while in-context learning remains pivotal to transformers’ self-optimization abilities [961], it continues to challenge comprehensive theoretical characterization.

Mechanistic Interpretability Complementary to theoretical analyses, mechanistic interpretability seeks to illuminate internal transformer operations by isolating and understanding computational substructures, often referred to as circuits, responsible for specific behavioral outputs. Initial efforts successfully mapped such circuits in smaller models, like GPT-2, to simple language tasks [962, 963, 964]. More recent advancements extended these methods to large, frontier-class models, employing techniques such as sparse autoencoders to detect semantically meaningful and interpretable components within transformer architectures [965, 966, 967, 968]. Although these approaches significantly improved the transparency and controllability of transformer computations, they also revealed complexities—such as entangled beneficial and harmful behaviors arising from many-shot conditioning [969]—underscoring ongoing challenges in achieving safe and reliable optimization through transformers.

Challenges Under Uncertainty Despite their sophisticated reasoning capabilities in structured contexts, LLM-based optimization frameworks exhibit notable shortcomings in environments characterized by uncertainty and stochasticity [970, 971, 972, 973]. Particularly, recent findings by Liu et al. [948] emphasize that LLMs often falter in optimal exploration and robust decision-making under uncertain conditions. These limitations highlight critical caution points for deploying transformer-based optimization solutions in real-world, dynamic settings where adaptability, exploration, and robustness to uncertainty are indispensable.

Transformers have redefined optimization paradigms by integrating linguistic reasoning, adaptive learning from context, and iterative refinement. Yet, despite their transformative empirical impacts, critical theoretical questions about their inherent optimization abilities—especially around in-context learning and decision-making under uncertainty—remain open and necessitate deeper exploration.

10.6 Summary and Discussion

 HIS chapter has explored the transformative paradigm of using large language models (LLMs) as versatile optimization tools, moving beyond traditional numerical methods into structured, linguistic, and iterative optimization spaces. We began by comparing foundational optimization paradigms—gradient-based, zeroth-order, and LLM-based optimization—highlighting the unique strengths and applications of each. By positioning LLM optimization within this broader context, we provided a clear

conceptual roadmap that illuminates both the innovative potential and inherent complexities involved in optimizing through natural language.

The iterative approaches to LLM optimization—including random search, gradient approximations, and Bayesian surrogate modeling—demonstrate how classical optimization concepts can be effectively adapted to the discrete, structured spaces where LLMs excel. These methods leverage iterative refinement, human-like reasoning, and natural language-based feedback mechanisms, enabling sophisticated optimization processes beyond traditional numeric domains.

However, significant challenges remain in fully realizing the potential of LLM-driven optimization. One primary concern is the computational and financial cost associated with iterative LLM queries, particularly when applied to large-scale or complex agentic workflows. Addressing these scalability issues requires novel strategies to reduce the number of LLM interactions, enhance computational efficiency, and ensure robustness against inherent variability and noise in model outputs.

Another crucial area for future development involves hyperparameter tuning and meta-optimization strategies. Current approaches often rely on heuristic methods, lacking principled frameworks for systematically optimizing hyperparameters such as prompt design, workflow structure, and feedback aggregation methods. Advancements in meta-optimization, adaptive hyperparameter tuning, and amortized optimization techniques promise substantial improvements in both efficiency and generalization, paving the way for more robust and adaptable optimization workflows.

Furthermore, effectively leveraging iterative refinement across depth and time remains a relatively underexplored frontier. Integrating classical engineering optimization strategies into LLM-driven frameworks presents promising avenues for improving long-term coherence, managing computational resources, and enhancing adaptability in evolving environments. Additionally, modeling and addressing uncertainty within iterative optimization processes is critical, as current models frequently struggle with exploration-exploitation trade-offs and robust decision-making in dynamic settings.

Finally, deepening our theoretical understanding of transformers as optimization agents is imperative. While empirical evidence supports the powerful optimization capabilities of LLMs, theoretical explanations—particularly those relating to in-context learning, mechanistic interpretability, and decision-making under uncertainty—remain incomplete. Advancing theoretical insights will be crucial not only for improving our understanding of transformer optimization dynamics but also for ensuring the safe, reliable, and explainable deployment of LLM-based systems.

While LLMs represent a revolutionary step in optimization paradigms, addressing these outlined challenges through systematic research will be essential to harness their full potential effectively. Continued interdisciplinary efforts integrating insights from classical optimization theory, machine learning, natural language processing, and decision-making theory will drive future innovations, ensuring that LLM-based optimization becomes a robust and reliable cornerstone of intelligent system design.

Chapter 11

Online and Offline Agent Self-Improvement

 N the pursuit of self-improvement, intelligent LLM-based agents leverage optimization as both a mechanism to refine individual components—such as prompt design, workflow orchestration, tool utilization, reward function tuning, and even the optimization algorithms themselves—and as a strategic framework to ensure these improvements align toward coherent overall performance. For instance, optimizing the reward function and prompt design in isolation might yield conflicting outcomes; however, a coordinated strategy can align these optimizations to maintain consistency and maximize overall effectiveness. In practice, an agent may simultaneously adjust its prompt templates and its criteria for success so that both changes complement each other rather than conflict. This holistic view of optimization prevents sub-components from working at cross purposes and instead directs all changes toward a common goal of performance enhancement.

We categorize agent self-evolution into two primary paradigms: *online* and *offline* self-improvement. *Online self-improvement* refers to real-time adaptation, where an agent continuously adjusts its behavior based on immediate feedback during operation. *Offline self-improvement*, in contrast, involves structured, batch learning phases where the agent is improved through deliberate training on collected data or simulations. In addition, we explore hybrid strategies that integrate both approaches to maximize efficiency and adaptability. These hybrid methods aim to combine the strengths of offline training’s stability with online learning’s agility, yielding agents that are both robust and dynamically responsive (Figure 11.1). In the following sections, we delve into each paradigm and then discuss how blending them can lead to continuous and balanced self-improvement.

11.1 Online Agent Self-Improvement

 NLINE 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 rate, latency, cost, and solution quality—within an iterative feedback loop. An LLM-based agent operating in a live setting (for example, a chatbot deployed to users) can use online self-improvement to update its strategies on the fly, learning from each interaction or correcting mistakes as they happen. Online optimization is particularly effective in applications that require dynamic adaptability, such as real-time decision-making, personalized user interactions, and automated reasoning systems that deal with changing

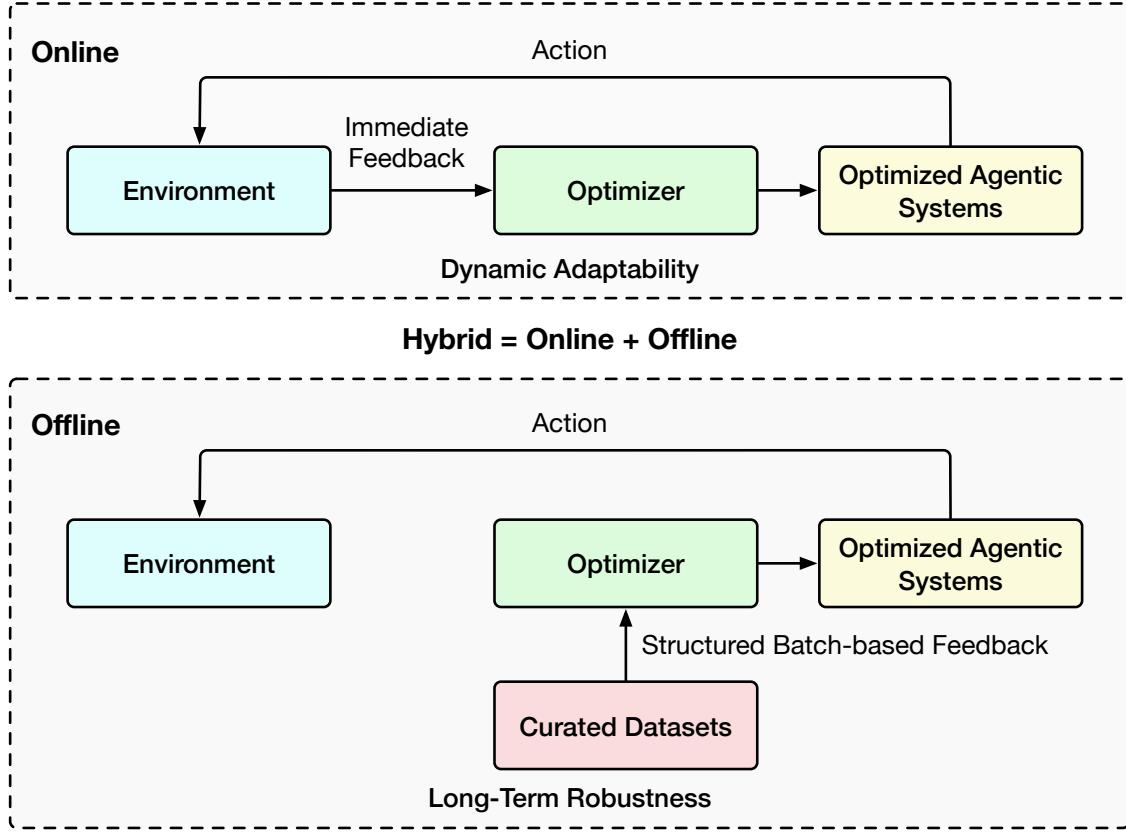


Figure 11.1: An illustration of self-improvement under three different utilization scenarios, including Online, Offline, and Hybrid self-improvement.

information. We can classify key strategies in online self-improvement into four categories: *Iterative Feedback and Self-Reflection*, *Active Exploration in Multi-Agent Systems*, *Real-Time Reward Shaping*, and *Dynamic Parameter Tuning*. Below, we examine each in turn.

Iterative Feedback and Self-Reflection These methodologies focus on enabling agents to critique and refine their own outputs iteratively using feedback loops [75, 94, 99, 97, 974, 74]. For example, Reflexion [75] is a framework where a language agent doesn't update its weights but instead uses verbal self-feedback to correct errors: the agent generates an answer, evaluates its correctness or mistakes through a reflection step (often by analyzing an outcome or an external hint), and then attempts an improved answer in light of that self-critique. Similarly, Self-Refine [94] allows the model to iteratively refine a draft solution by identifying flaws in its initial output and revising it – effectively performing its own review and edit cycle. Tree-of-Thoughts [99] introduces a strategy where the agent explores multiple reasoning pathways (a tree of possible thought sequences) rather than a single chain-of-thought; the model generates different solution paths, evaluates them (possibly pruning those that seem unpromising), and backtracks or branches out as needed, thereby increasing the chance of finding a correct or high-quality solution. The ReAct framework [97] combines chain-of-thought “Reasoning” with “Acting” steps: the agent interleaves reasoning about a problem with taking actions (such as querying a tool or environment) and can revise its approach based on the observed results of those actions. This means the model isn’t stuck with its first plan – it can adjust mid-way if an action’s outcome indicates a need for correction. In addition, some methods rely on self-consistency during inference [105]: the agent generates multiple reasoning traces for the same query and then selects the most consistent or frequent answer among them, which empirically tends to be more reliable. Another complementary approach is to use a process reward model (PRM) [974] to evaluate each step of the agent’s

reasoning. With a PRM, the agent can generate several candidate solution steps or full solutions, have the PRM score the logical correctness of each step in the chain, and then choose or refine the best-scoring solution. By verifying intermediate reasoning steps (rather than only the final answer), PRM-based feedback helps catch mistakes early and avoid propagating errors. Voyager [74] provides a practical example of iterative self-improvement in an embodied setting: it uses an LLM agent to autonomously explore a task environment (Minecraft, in this case), reflect on failed attempts, and store successful strategies in a skill library for future use. Collectively, all these frameworks share a common theme: they create feedback loops where the agent’s own outputs are analyzed and used to make immediate improvements without requiring a separate offline retraining cycle. This reduces error propagation and allows for rapid, on-the-fly adaptation to new challenges or corrections of mistakes.

Active Exploration in Multi-Agent Systems In multi-agent or multi-role frameworks, agents actively explore and collaboratively search for better behaviors or solutions by exchanging feedback with each other in real time [755, 975, 756, 204]. For instance, MetaGPT [755] organizes multiple LLM-based agents into specialized roles (such as a product manager, engineer, tester, etc.) that communicate and coordinate on a task (like software development). Each agent contributes its expertise, and through continuous dialogue and mutual feedback, the group converges toward an improved result. Likewise, CAMEL [975] demonstrates a dual-agent conversation approach where two AI agents (e.g., one playing the user and one the assistant with a given goal) interact to progressively refine a solution or idea – the “user” agent can challenge or provide new information, and the “assistant” agent adapts its responses accordingly, resulting in deeper exploration than a single agent alone might achieve. ChatDev [756] simulates a complete software development team with multiple agent roles (system architect, developer, tester, etc.) all implemented as LLMs; these agents generate code, review each other’s work, and correct errors in an online loop, mirroring how a human team might incrementally improve a software project. Meanwhile, HuggingGPT [204] uses a central LLM as a controller to orchestrate a fleet of specialized models (for example, calling a vision model for an image task or a speech model for audio input). The LLM dynamically routes subtasks to appropriate expert models and then integrates their feedback or results into its reasoning. This kind of multi-agent or multi-model exploration allows the system to handle complex tasks by learning which agent or tool is best suited for each sub-problem and adjusting the workflow in real time. Through active exploration and division of labor, multi-agent systems can discover novel patterns of problem-solving and iteratively refine their collective output based on continuous inter-agent feedback and validation.

Real-Time Reward Shaping Instead of relying solely on fixed or pre-defined reward signals, some frameworks incorporate immediate feedback from the environment or the agent’s own evaluations to adjust the reward function and policy on the fly citezelikman2023self, zelikman2024quiet, xie2023text2reward, zhang2024extracting. This approach is essentially adaptive reward optimization in real time. For example, Zelikman et al. [976, 118] propose techniques where a language model can generate its own intermediate checks or self-queries during reasoning, and use the outcomes of those checks as internal reward signals. In these methods, if the model’s intermediate reasoning step produces a useful insight or question (one that eventually leads to a correct answer), the model “rewards” itself by reinforcing that line of thought. Such immediate self-feedback mechanisms encourage the agent to favor thought processes that seem promising or correct. Another notable work is Text2Reward by Xie et al. [130], which is a data-free framework using an LLM to automatically generate dense reward functions from textual task descriptions. In a traditional RL setting, designing a reward function can be hard (especially if only sparse success signals are available), but Text2Reward leverages the language model’s knowledge to shape a more informative reward signal. For instance, given a description of the goal, the LLM can suggest intermediate milestones or subgoals and assign reward values to them, providing the agent with a richer gradient of feedback as it works toward the final goal. This real-time shaping means the agent doesn’t have to rely purely on eventual success or failure – it gets guidance at each step. Similarly, other research (e.g., Zhang et al. [977]) explores extracting heuristic

reward signals from the agent’s own reasoning trajectory or from external observations on the fly. In such approaches, if the agent’s current action or output yields certain desirable patterns (for example, matching a known criterion of a good solution), a reward adjustment is made immediately to reinforce that behavior. By continuously calibrating what the agent is striving to maximize (its reward function) based on recent interactions, real-time reward shaping allows the agent to adapt its objectives dynamically. This leads to behavior that can balance trade-offs (like accuracy vs. speed or success vs. resource cost) in response to the current context and feedback, rather than being locked to a static reward definition decided beforehand.

Dynamic Parameter Tuning In this category, agents autonomously update their internal parameters and configurations during runtime to optimize performance. These parameters can include prompt templates, thresholds for invoking tools or external APIs, search heuristics (like how broad or deep a lookahead to perform in planning tasks), sampling settings for generative models (temperature, top- k values), and more. The key idea is that the agent itself monitors its performance and tweaks these knobs in real time, rather than a human manually doing so. Because direct gradient-based tuning of large model parameters online is often impractical, many approaches rely on gradient-free optimization or surrogate gradient methods that operate on these higher-level parameters. For instance, an agent might use evolutionary strategies or bandit algorithms to try slight variations in its prompt or tool-selection strategy and keep the changes that improve outcomes. A concrete example is the Self-Steering Optimization (SSO) algorithm proposed by Xiang et al. [978]. SSO enables a large language model to align and improve itself autonomously by generating high-quality preference signals during training. In practice, SSO has the model produce its own comparisons or rankings of outputs (simulating what human feedback might say) and uses those as a training signal to adjust the model’s policy. By doing so iteratively, it eliminates the need for manual human annotation of preferences. Moreover, SSO is designed to keep this training process on-policy – meaning the preferences are generated based on the model’s current behavior, ensuring the model is always learning from feedback relevant to what it actually does. This kind of dynamic tuning not only applies to alignment; an agent could similarly self-tune other parameters. For example, if an agent notices it’s making too many mistakes using a certain tool, it could raise the confidence threshold required before that tool is invoked, or if it finds its responses are too slow, it might shorten its chain-of-thought or use a faster reasoning heuristic. By autonomously adjusting such internal dials, the agent optimizes both its computational efficiency (e.g. using resources judiciously) and decision accuracy (e.g. choosing the right strategy for the context) on the fly. The result is seamless adaptation to evolving conditions without waiting for an offline re-training phase.

Online self-improvement thus fosters a continuously evolving agent framework where learning is embedded within task execution. The agent becomes highly adaptive in real time, capable of improving with each interaction or feedback instance. This leads to more personalized and context-aware behavior (since the agent can optimize for the specific user or environment it’s currently dealing with), and it enhances robust problem-solving by quickly addressing errors or changes. However, it’s worth noting that while online methods offer agility, they also demand careful design to avoid instability (for example, an agent might “chase” recent feedback too strongly and forget older knowledge, a phenomenon sometimes called catastrophic forgetting or drift). In summary, online self-improvement provides powerful tools for immediate, user-centric optimization and rapid learning, especially in dynamic settings, complementing the more deliberate improvements made via offline methods.

11.2 Offline Agent Self-Improvement



OFFLINE self-improvement, in contrast to online methods, leverages structured batch-based optimization carried out in dedicated training phases. In this paradigm, an agent improves by training on accumulated data or simulated experience in a disconnected setting, rather than while actively interacting with users or an environment. The agent typically enters an offline “learning mode” (which could range from fine-tuning a model on new examples to entirely retraining certain components) and, once that

session is complete, it updates its capabilities for the next deployment. This approach systematically enhances the agent’s generalization capabilities by exposing it to curated, high-quality datasets and scenarios without the time or safety constraints of a live environment [979, 797, 980, 981, 982]. For example, a conversational LLM agent might be improved offline by fine-tuning it on a large repository of solved dialogue tasks or by training on logs of past conversations and their ideal resolutions. Because offline training doesn’t need to be instantaneous, it can employ computationally intensive methods and large-scale data that would be impractical to use during real-time interaction. Key strategies in offline self-improvement include *Batch Parameter Updates and Fine-Tuning*, *Meta-Optimization of Agent Components*, and *Systematic Reward Model Calibration*.

Batch Parameter Updates and Fine-Tuning In this category, an agent undergoes extensive weight updates using supervised learning or reinforcement learning (RL) techniques across large datasets, typically iterating over the data for multiple epochs to gradually minimize errors. This could involve classical fine-tuning of a pre-trained LLM on domain-specific data or difficult tasks to improve its performance in those areas. One common approach is Reinforcement Learning from Human Feedback (RLHF) [70], where an agent or model is trained (offline) on a static collection of human preference data: the agent’s policy (the LLM’s behavior) is adjusted so that it produces outputs preferred by humans, as judged by a reward model trained on those preferences. Such RLHF fine-tuning greatly improves alignment with what users find helpful, but it’s done in batch after collecting preferences, not during live use. In addition, offline improvements often involve bolstering the agent’s ability to retrieve and use knowledge. For instance, retrieval-augmented generation (RAG) techniques are frequently integrated into training to enhance the agent’s contextual understanding and long-term memory [983, 898]. With RAG, the agent learns to query an external knowledge source (like a document database or the web) and incorporate the retrieved information into its responses. During offline training, the agent can practice this behavior using supervised signals (learning when and what to retrieve for a given query) so that at deployment it can seamlessly pull in relevant information it didn’t explicitly store in its own weights. By fine-tuning on large-scale datasets that cover a wide range of scenarios (e.g., large collections of Q&A pairs, code libraries, or multimodal data for vision-and-language tasks), the agent improves generalization – it becomes capable of handling new inputs more robustly because it has seen many variations during training. Moreover, because these updates are done in a controlled setting, developers can use computationally heavy algorithms (like large-scale gradient descent, hyperparameter sweeps, or simulation of many trajectories in an environment) to squeeze out as much performance as possible. The outcome of batch training is a new version of the agent (or a component of the agent) that performs better on average, which can then be deployed. Notably, methods like this allow for optimizing performance and stability before the agent faces real-world tasks, reducing the chance of failure in mission-critical applications. For example, if an LLM-based agent will be used for medical advice, offline training on a vetted medical Q&A dataset can significantly improve its accuracy and safety, ensuring a strong baseline model is in place.

Meta-Optimization of Agent Components In these approaches, offline training is not just used to improve the agent’s task performance, but also to refine the optimization processes and hyperparameters that the agent itself uses. In other words, the agent learns how to learn better. This often involves meta-learning techniques, where the objective is to train the agent (or a meta-controller) on a variety of tasks such that it can quickly adapt to new tasks with minimal additional training. One example is learning an initialization or update rule that works well for a family of problems – the famous Model-Agnostic Meta-Learning (MAML) algorithm is a classic (non-LLM) example where a model is trained to have parameters that can be very quickly fine-tuned to new tasks [984]. In the context of LLM-based agents, researchers have explored similar ideas. For instance, Zelikman et al. [976, 118] have demonstrated techniques where the agent, through offline self-training, improves its own reasoning strategies. In one of their approaches, a language model generates rationales for solving problems and uses the correctness of the answer to adjust how it produces these rationales in the future – effectively the model is tweaking its internal “algorithm” for reasoning. Such

meta-optimization might also involve hyperparameter tuning (finding the best learning rate, prompt length, etc.) or even neural architecture search (altering components of the model or agent architecture) done offline. For example, an agent could be put through an offline phase where different prompting techniques or tool-use strategies are tried, and the best-performing strategies are then baked into the agent’s policy going forward. Unlike standard fine-tuning which targets performance on a fixed task distribution, meta-optimization seeks to give the agent a form of learning-to-learn capability: after meta-training, when the agent encounters a new kind of problem, it can adapt more quickly or execute an improved optimization algorithm internally. The overarching benefit is that the agent not only becomes better at tasks, it becomes better at becoming better, so to speak, which is a powerful form of self-improvement.

Systematic Reward Model Calibration The offline setting is ideal for carefully calibrating and improving reward models, which are used to guide agent behavior. A reward model in the context of LLM agents might score how well an output aligns with human preferences or task objectives. Offline calibration involves training these models on collected data (for example, human rankings of outputs, or known correct solutions) and tuning them to accurately reflect the true goals we care about. One straightforward example is again from RLHF: a reward model is trained on a dataset of comparisons (which of two responses a human prefers) and then refined until it reliably predicts human preference. Offline, we can put significant effort into this – using large annotated datasets and powerful training methods – to get a high-quality reward function. Recent advanced techniques have proposed going beyond simple pairwise preference training. For instance, LiRE (Listwise Reward Enhancement) by [985] is a framework that optimizes the reward model using a listwise approach: instead of looking at one or two outputs at a time, the reward model is trained on sets of outputs together, so that it learns to rank a collection of candidate responses appropriately. This listwise training provides a richer signal – it ensures the reward model’s scores are calibrated not just in isolation but relative to a distribution of possible outputs, which can improve its consistency and robustness. Other works introduce hierarchical reward modeling [986], where the reward function might break down the evaluation into multiple criteria or levels (for example, first checking factual accuracy, then style, then overall helpfulness, and combining these). Offline optimization is critical here because it allows integrating such complex criteria in a controlled way. By using gradient-based reward optimization on offline data (where we know the desired outcomes), we can adjust the reward model so that it aligns the agent’s behavior with long-term or high-level objectives, not just short-term rewards. For example, if we want an agent that writes accurate and polite answers, we can offline-train a reward model that gives high scores only to outputs that are both factually correct and politely phrased. During deployment, that reward model can then be used (in RL or as a scoring mechanism) to steer the agent. The offline calibration mitigates bias (since we can ensure the training data for the reward model is balanced and carefully curated) and enhances generalization (since the reward model will have seen a wide range of outputs and learned nuanced distinctions). Overall, offline reward model calibration ensures that when the agent is later let loose in the real world or an online learning phase, its notion of what is “good” behavior is already well-shaped to match human values or specified goals. This reduces the risk of the agent pursuing the wrong objective or exploiting loopholes in a poorly shaped reward, issues that are common in poorly constrained online learning.

The structured nature of offline optimization produces a robust agent baseline. Because offline training occurs with plentiful data and carefully monitored procedures, it typically yields an agent that performs consistently well on the training distribution and often generalizes better. This phase is especially crucial for mission-critical applications where the agent must meet certain performance guarantees (for example, a diagnostic assistant that must not exceed a given error rate). Before deployment or further online adaptation, offline self-improvement can iron out many problems, optimize efficiency (the agent can be trained to use fewer tokens or less computation per task, for instance), and instill necessary knowledge or skills. The trade-off is that offline methods lack the agility of online learning; once the agent is deployed after offline training, it might not handle an unforeseen scenario optimally until the next retraining. In summary, offline self-improvement emphasizes stability, thoroughness, and pre-planned learning, complementing the

reactivity of online methods. It is an essential component for building high-performance agents that have strong foundations and predictable behavior when first deployed.

11.3 Comparison of Online and Offline Improvement



ONLINE AND OFFLINE optimization offer complementary benefits, each excelling in different aspects of an agent’s self-improvement cycle. To decide which paradigm to use (or how to mix them), it is important to understand their relative strengths and trade-offs. Table 11.1 summarizes key distinctions between these two paradigms, which we also discuss below.

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

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

Learning Process Online optimization involves continuous updates based on real-time feedback, whereas offline optimization relies on batch updates during scheduled training phases. In practice, an online-learning agent is constantly in a mode of “observe -> adjust -> act,” blending learning with execution. This means the learning process is interwoven with the agent’s operation. By contrast, an offline-learning agent has a two-stage life: it gathers experiences or data, then steps away to crunch on this data in a separate learning phase, and finally returns with updated behavior. The continuous nature of online learning means improvements can happen immediately in response to new data, while offline’s episodic learning means improvements are delayed until the next training session but can be more thorough.

Adaptability As a consequence of their learning processes, online methods offer high adaptability. An online-optimized agent can adjust dynamically to changing conditions or requirements; for example, if a user’s preferences shift or a sudden anomaly appears in the environment, an online learner can quickly incorporate that information and modify its behavior. Offline methods have lower adaptability in the short term – once the agent is deployed, it generally sticks to the behavior it was trained for until the next retraining cycle. If confronted with a novel scenario outside its training distribution, an offline-trained agent might struggle because it isn’t updating its knowledge on the fly. In essence, online learning is like a sailor who constantly trims the sails with every shift of the wind, while offline learning is like a navigator who charts a course using long-term weather patterns and only revises the route at planned intervals.

Computational Efficiency Online optimization is typically more efficient for incremental updates. Since it processes feedback instance-by-instance or in small batches, the agent can often update its parameters or strategy using lightweight computations (sometimes even without retraining any neural weights, as in prompt adjustments or short-term memory updates). This means the computational cost is spread out over time and can be kept relatively low per interaction. Offline optimization, on the other hand, often

involves heavy batch training procedures that can be resource-intensive (e.g., fine-tuning a large LLM on a big dataset with multiple epochs can require a lot of GPU hours). However, it's worth noting that while online methods are efficient per update, the cumulative cost of many small updates could add up, and they require infrastructure to continuously run the learning process. Offline training is done less frequently but in a more heavyweight manner.

Data Dependency Online methods require a stream of real-time data. They excel when such data is readily available and when the environment provides frequent feedback signals that the agent can learn from. This could be live user interactions, sensor readings, or the agent's own trial-and-error experiments in a simulation. If the stream stops or is not rich enough, online learning might stagnate or overfit to recent data. Offline methods rely on curated, high-quality datasets that are prepared in advance. This gives more control over the data's quality and diversity (you can ensure the training data covers various scenarios, is annotated correctly, etc.), but it also means offline methods are limited by what's in the dataset. If the dataset is missing some scenarios or is biased, the agent's learning will reflect those gaps, and it won't fix them until new data is collected and another offline training happens.

Risk of Overfitting Generally, online learning tends to have a lower risk of overfitting to any particular dataset, because the agent is continuously exposed to new situations and doesn't train too long on one fixed batch of data. Instead, it "forgets" naturally as new experiences come (which is a double-edged sword – it can forget important things if not managed, as mentioned regarding drift). Offline learning can have a higher risk of overfitting if the training data is not sufficiently diverse or if the model is trained for too many epochs on the same data. Without careful regularization, an offline-trained model might become too specialized to the training set and not perform well on truly novel inputs. That said, with techniques like early stopping or using very large and varied datasets (e.g., pretraining LLMs on massive corpora), offline training can achieve strong generalization. The point is that the nature of exposure to data differs: online gets a trickle of varied data (mitigating overfitting but possibly causing forgetting of older data), offline gets a static bulk of data (risking overfitting to that unless it's broad).

Stability Offline optimization is usually more stable because it happens in a controlled environment. Engineers or researchers can carefully monitor an offline training run, use validation sets to pick the best model, and ensure the agent's updated version passes certain safety or performance tests before deploying it. The changes made in offline training are deliberate and evaluated thoroughly post-training. In contrast, online optimization can introduce instability. Frequent updates, if not managed well, can lead the agent's performance to oscillate or degrade unexpectedly (for example, an agent might "chase noise" in the feedback and make itself worse on average, a phenomenon known as reward hacking or just instability in online learning). Mechanisms like learning rate decay, confidence thresholds for applying updates, or hybrid approaches (where online updates are sandboxed before full adoption) are often needed to maintain stability in an online context. Essentially, offline is like making changes to a system in a test lab, whereas online is like tweaking a system live in production – the latter has to be done very carefully.

Modern intelligent systems increasingly integrate both online and offline methods to capitalize on the advantages of each. An agent might be initially trained offline to have a solid foundation (addressing the stability and basic capability aspect) and then fine-tuned online to personalize and adapt (addressing the adaptability aspect). While pure online or pure offline setups can work, combining them often yields a more powerful and flexible system. The next section on hybrid approaches discusses how such integration can be achieved in practice.

11.4 Hybrid Approaches

 ECOGNIZING that online and offline methods each have inherent limitations, many contemporary systems adopt hybrid optimization strategies that blend the two paradigms. The goal of a hybrid approach is to get the best of both worlds: the robust, high-performance foundation from offline training, and the adaptable, real-time refinement from online learning. In a hybrid self-improvement loop, an agent alternates between (or sometimes concurrently uses) offline and online phases of learning. This explicit integration allows the agent to autonomously evaluate, adapt, and enhance its behaviors through distinct yet interconnected stages. We can break down a typical hybrid strategy into three stages: *Offline Pre-Training*, *Online Fine-Tuning for Dynamic Adaptation*, and *Periodic Offline Consolidation for Long-Term Improvement*.

Offline Pre-Training In the first stage, the agent acquires a strong baseline of knowledge and skills through extensive offline training on curated datasets or in simulated environments. This can be thought of as the agent’s “education” before it is deployed in the real world. By learning from a wide range of examples and scenarios offline, the agent develops essential capabilities such as reasoning, language understanding, and decision-making pertinent to its intended tasks. For instance, an LLM-based assistant might be pretrained on diverse text corpora and dialogues so it has a broad knowledge and good conversational ability from the start. This stage establishes a stable foundation such that the agent doesn’t enter the online phase blank-slate or brittle. A concrete example from the reinforcement learning domain is provided by Schrittwieser et al. [987] – in their work (known for the MuZero Unplugged algorithm), they showed that an agent can be first trained on a batch of past game experiences (offline) to build up competence, and then successfully fine-tuned with further online interactions. That offline pre-training systematically enhances the agent’s initial capabilities, ensuring that subsequent online improvements are built upon a reliable and effective base. In general, offline pre-training reduces the burden on online learning: instead of having to learn everything from scratch in real time (which could be slow or risky), the agent only has to learn the differences or new patterns that were not covered in the pre-training.

Online Fine-Tuning for Dynamic Adaptation After or alongside pre-training, the agent enters an online learning phase where it fine-tunes its behavior by interacting with the environment or users and receiving feedback in real time. In this stage, the agent actively monitors its performance, identifies shortcomings or new requirements, and adjusts its strategies on the fly. This might involve tuning its prompts, learning from user corrections, trying new actions, or updating its internal models slightly with each interaction. The changes here are typically smaller-scale and incremental (since we want to avoid catastrophic changes while live) but happen continuously. This stage directly aligns with the self-improvement paradigm – the agent is essentially self-evaluating and self-adjusting in response to the world. An example of a framework that embodies this is Decision Mamba-Hybrid (DM-H) introduced by Huang et al. [988]. DM-H is a reinforcement learning approach that combines two techniques: a transformer-based model for high-quality decision predictions and a memory-efficient model (Mamba) for handling long-term dependencies. In an online fine-tuning context, DM-H allows an agent to efficiently adapt to complex, evolving scenarios by generating sub-goals from its long-term memory and then prompting the transformer model with those sub-goals to decide on immediate actions. The result is an agent that can adjust its plan as conditions change, leveraging its offline-trained knowledge (in the transformer) together with real-time cues (through the Mamba memory and feedback loop). More generally, during online fine-tuning, an agent might do things like dynamically reweighting different objectives (if it notices it’s going too fast at the cost of accuracy, it can slow down), or personalizing responses (if a user keeps correcting a certain style of answer, the agent learns that user’s preference). The essence of this stage is fast adaptation: the agent is tailoring its behavior to the here-and-now, which is crucial for maintaining performance in the face of novelty or drift in the environment.

Periodic Offline Consolidation for Long-Term Improvement The third stage involves the agent periodically pausing its online updates and entering a new offline training phase to consolidate and integrate the knowledge it has acquired through online interactions. Think of this as the agent taking a step back to reflect on everything it learned recently, and solidifying those lessons into its core parameters or knowledge base. During online learning, the agent might accumulate a lot of small tweaks, new data (e.g., logs of interactions, new problem cases it encountered), or temporary adaptations. The offline consolidation phase uses this accumulated experience as a new training dataset. The agent is retrained or fine-tuned in a more rigorous way on that data, possibly also reinitializing some of its online-specific adaptations to prevent drift. The advantage of doing this offline is that the integration of new knowledge can be done more systematically and safely – one can use the full suite of training techniques (like shuffling data, doing multiple passes, using regularization to not forget older knowledge, etc.). This ensures that improvements gleaned from the online phase are retained long-term and that any noise or unstable changes are smoothed out. The Uni-O4 framework by Lei et al. [989] exemplifies this process. Uni-O4 is designed to unify offline and online deep reinforcement learning; it aligns the objectives used in offline pre-training with those used in online fine-tuning so that an agent can move between the two seamlessly. In their approach, an agent might be first trained offline on a simulator, then deployed to learn online in the real world, and periodically its real-world experiences are folded back into an offline update. The result is a cycle where the agent keeps improving without forgetting why it was successful before. In practice, such periodic consolidation might occur on a schedule (say, an autonomous vehicle retrains overnight on the day’s collected data) or based on a threshold (if the agent’s online performance starts degrading or plateauing, trigger an offline retraining with the new data). This hybrid loop allows long-term stability and continual adaptation.

By explicitly alternating between these modes, hybrid optimization supports autonomous, continuous evolution of the agent. The offline phases provide stability, comprehensive learning, and a chance to ingest large amounts of experience, while the online phases provide immediacy, specificity, and the ability to keep up with changes. The two feed into each other: offline training makes the agent better at online adaptation (since it starts from a higher base), and online experience makes the next offline training richer (since there’s new real-world data to learn from). This cyclical training approach is well-suited for complex real-world scenarios. Consider an autonomous robot: it can be pre-trained in simulation (offline) to get basic navigation and manipulation skills, then deployed in a new home where it fine-tunes its policy as it interacts with the actual environment (online), and after a while, all the data from the home is used to do a thorough update to its policy (offline again) resulting in improved performance that incorporates the quirks of that home’s layout. Or think of a personalized intelligent assistant: it’s initially trained on general language data (offline), then it learns a user’s particular needs and style from ongoing conversations (online), and periodically it can retrain on the compiled chat history to improve its core dialogue model (offline consolidation). Through such hybridization, agents maintain the long-term robustness and knowledge depth that offline training provides, while also achieving continuous refinement and personalization through online learning.

In summary, hybrid approaches interweave structured offline learning with proactive online adaptation. This allows agents to avoid the pitfalls of relying solely on one method: the agent is neither stuck with only what it learned yesterday (thanks to online updates) nor caught in a loop of potentially unstable live changes (thanks to grounding, reset, and validation via offline phases). As a result, hybrid agents tend to exhibit both immediate responsiveness and stable long-term improvement. They are particularly powerful in domains like autonomous driving, robotics, finance, or large-scale interactive systems, where conditions evolve over time and safety or reliability is critical. By cycling between learning modes, the agent keeps getting better in a controlled yet flexible manner, which is the essence of continual self-improvement.

11.5 Summary and Discussion

 N this chapter, we have explored the paradigms of online, offline, and hybrid self-improvement for LLM-based agents. Online self-improvement enables agents to adapt dynamically through iterative feedback loops, real-time parameter tuning, and active exploration, providing immediate responsiveness and context-sensitive learning. Conversely, offline self-improvement employs structured, batch-based training to cultivate robust generalization, stability, and comprehensive knowledge integration. Recognizing the complementary strengths of these two paradigms, hybrid strategies integrate the real-time adaptability of online learning with the foundational stability provided by offline optimization, offering a balanced framework that supports continuous, controlled self-evolution.

Despite notable progress, several challenges remain. Online approaches, while agile, must grapple with stability concerns, potential drift, and computational overhead from continuous updates. Offline methods, though stable and robust, face limitations in adaptability, data coverage, and the risk of overfitting to curated datasets. Hybrid methods seek to mitigate these limitations but introduce additional complexity in managing transitions and alignment between offline and online phases. Future research opportunities include the design of principled hybrid algorithms, adaptive meta-learning techniques for seamless online-offline transitions, and efficient strategies for managing computational resources and memory during continual self-improvement. Ultimately, effective self-improvement for LLM-based agents hinges on developing frameworks that judiciously balance agility and stability, adaptability and robustness, enabling sustained and reliable performance across diverse real-world scenarios.

Chapter 12

Intelligent Evolution through Scientific Discovery

 N 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 [990] and assist humans in extending the boundaries of knowledge.

Our proposed agent framework, as illustrated in Figure 1.2, systematically acquires scientific data, insights, and principles through active interaction with its environment. This information is subsequently stored within the agent’s memory (detailed in Chapter 3) and utilized to enhance both its internal world model (elaborated in Chapter 4) and reward function (discussed in Chapter 5), collectively leading to improved scientific cognition. Within optimization spaces, the agent continuously evolves by refining workflows and enhancing tool usage, thereby promoting increasingly efficient scientific discoveries. This evolution is realized through strategies such as prompt optimization, the creation of new tools, automated workflow generation, and other advanced methods. Enhanced efficiency in scientific discovery subsequently facilitates further scientific information acquisition, establishing a reinforcing cycle. Consequently, the framework fosters a self-sustaining loop of “knowledge discovery → enhanced capability for knowledge discovery → increased knowledge discovery”, driving the continuous evolution of agent intelligence, ultimately benefiting humanity.

In this chapter, we first define the concepts of knowledge and intelligence to clarify our discussion, followed by an introduction of three typical scenarios where agents interact with scientific knowledge. We highlight existing successes and examples of self-enhancing agents applied to theoretical, computational, and experimental scientific research. Finally, we summarize current challenges and offer insights for future directions.

12.1 Agent's Intelligence for Scientific Knowledge Discovery

 KNOWLEDGE, traditionally defined as *justified true belief*, traces back to Plato [991] and has been further refined by Edmund Gettier [992], who argued that knowledge must be produced by a reliable cognitive process—though its precise definition remains debated [993]. 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 Kullback–Leibler (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 [994], to Parr et al.'s framing of intelligence as minimizing model-world divergence under the free energy principle [995], many frameworks converge on a common theme: intelligent behavior arises from making accurate predictions about an uncertain world. Clark [404], for instance, argues that intelligent agents constantly engage with the world through prediction and error correction to reduce surprise. Chollet [996] 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 $\mathbf{x} = \{x_1, x_2, \dots, x_n\}$ 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:

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

$D_0(\theta)$ 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 $e^{-ND_0(\theta)}$ [997]. 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, $\text{CaFe}_2(\text{PO}_4)_2\text{O}$, is synthesizable. The agents can predict either (1) $x_1=\{\text{CaFe}_2(\text{PO}_4)_2\text{O} \text{ is synthesizable}\}$, and (2) $x_2=\{\text{CaFe}_2(\text{PO}_4)_2\text{O} \text{ is not synthesizable}\}$. In reality, since $\text{CaFe}_2(\text{PO}_4)_2\text{O}$ is a natural mineral called *crocobelonite*, $P_{\mathcal{W}}(x_1) = 1$ and $P_{\mathcal{W}}(x_2) = 0$. However, this mineral was only recently reported on October 4, 2023 [998], after the knowledge cutoff of many LLMs; thus, the agents lacks that knowledge. Compare Agent 1, which guesses randomly $P_{\theta_1}(x_1) = P_{\theta_1}(x_2) = 0.5$, yielding $D_0(\theta_1) = \log 2$. In contrast, Agent 2 employs first-principles calculations and finds that $\text{CaFe}_2(\text{PO}_4)_2\text{O}$ is the lowest-energy phase among competing compounds of the same composition in the Ca-Fe-P-O phase diagram [999], indicating its thermodynamic stability. Thereby, Agent 2 predicts that $\text{CaFe}_2(\text{PO}_4)_2\text{O}$ is likely synthesizable, suggesting $P_{\theta_2}(x_1) > 0.5 > P_{\theta_2}(x_2)$. Consequently, $D_0(\theta_2) = -\log P_{\theta_2}(x_1) < D_0(\theta_1)$, 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}_K denote this known subset and \mathbf{x}_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 $\mathbf{x}_K \subseteq \mathcal{K}$, $\mathbf{x}_U \subseteq \mathcal{U}$, and $\mathcal{K} \cup \mathcal{U} = \mathcal{W}$. For example, in text generation, the prompt text \mathbf{x}_K represents already known information. The efficiency of the language model is then measured by its predictive accuracy for the generated text \mathbf{x}_U based on \mathbf{x}_K . More generally, the agent's intelligence is measured by the relative entropy of the conditional probability distribution:

$$D_K(\theta, \mathbf{x}_K) = \sum_{\mathbf{x} \subseteq \mathcal{U}} P_{\mathcal{W}}(\mathbf{x}|\mathbf{x}_K) \log \frac{P_{\mathcal{W}}(\mathbf{x}|\mathbf{x}_K)}{P_{\theta}(\mathbf{x}|\mathbf{x}_K)} \quad (12.2)$$

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

Definition 12 (Agent's Intelligence for Knowledge Discovery). The agent's intelligence for knowledge discovery is quantified as the negative KL divergence between the its predicted and the true probability distributions (P_{θ} and $P_{\mathcal{W}}$) of the unknown information ($\mathbf{x} \subseteq \mathcal{U}$), both conditioned on its memory M_t^{mem} .

$$IQ_t^{\text{agent}} \equiv -D_K(\theta, M_t^{\text{mem}}) = -\sum_{\mathbf{x} \subseteq \mathcal{U}} P_{\mathcal{W}}(\mathbf{x}|M_t^{\text{mem}}) \log \frac{P_{\mathcal{W}}(\mathbf{x}|M_t^{\text{mem}})}{P_{\theta}(\mathbf{x}|M_t^{\text{mem}})} \quad (12.3)$$

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 [1000]. 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 [1001].

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}_{Δ} 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

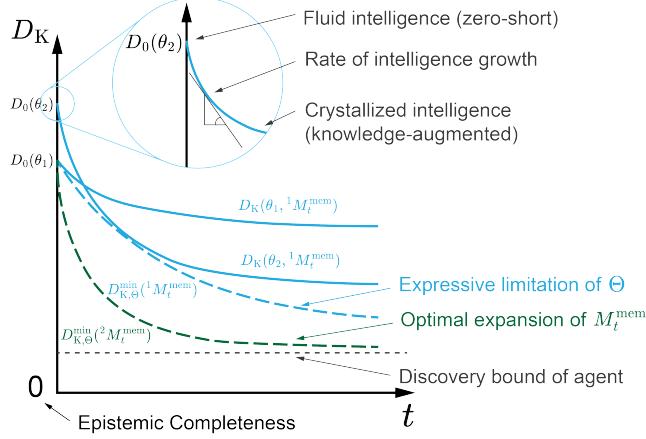


Figure 12.1: Schematic representation of agent intelligence and knowledge discovery. The agent's intelligence, measured by the KL divergence D_K between predictions and real-world probability distributions, evolves from fluid intelligence (zero-shot predictions for new problems) to crystallized intelligence (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 $D_{K,\Theta}^{\min}$. Given Θ , $D_{K,\Theta}^{\min}$ is influenced by different knowledge expansion strategies, such as ${}^1 M_t^{\text{mem}}$ and ${}^2 M_t^{\text{mem}}$, shown as dash lines.

agent's intelligence at time $t + 1$ is given by:

$$IQ_{t+1}^{\text{agent}} \equiv -D_K(\theta, M_t^{\text{mem}} \mathbf{x}_\Delta) = - \sum_{\mathbf{x}' \subseteq \mathcal{U}'} P_W(\mathbf{x}' | M_t^{\text{mem}} \mathbf{x}_\Delta) \log \frac{P_W(\mathbf{x}' | M_t^{\text{mem}} \mathbf{x}_\Delta)}{P_\theta(\mathbf{x}' | M_t^{\text{mem}} \mathbf{x}_\Delta)} \quad (12.4)$$

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

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

The second term is the relative entropy of the conditional probability distribution of \mathbf{x}_Δ 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}^{agent} can be further increased by leveraging the newly acquired knowledge to optimize θ within M_t^{mem} .

Interestingly, the expected gain in intelligence at time t is determined by the discrepancy between the actual distribution $P_W(\mathbf{x} | M_t^{\text{mem}})$ and the model-predicted distribution $P_\theta(\mathbf{x} | M_t^{\text{mem}})$. 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 [990] as a special type of curiosity-driven agent [1002], 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 [487, 1003] (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 $D_K(\theta, M_t^{\text{mem}})$, thereby maximizing IQ_t^{agent} :

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

and

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

Here, $D_{K,\Theta}^{\min}(M_t^{\text{mem}})$ 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, $D_{K,\Theta}^{\min}(M_t^{\text{mem}})$ forms the envelope of the family of functions $D_K(\theta, M_t^{\text{mem}})$, where θ ranges over Θ .

For a given model family Θ , $D_{K,\Theta}^{\min}(M_t^{\text{mem}})$ 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 $D_{K,\Theta}^{\min}(M_t^{\text{mem}})$. One can prove that $D_{K,\Theta}^{\min}(M_t^{\text{mem}})$ is monotonically non-increasing as M_t^{mem} expands, since it forms the envelope of a family of non-increasing functions $D_K(\theta, M_t^{\text{mem}})$. 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} . Agents with varying knowledge expansion strategies can derive diverse explanations from the same physical system, leading to different evolutionary trajectories [1004]. The optimal expansion strategy is the one that results in the steepest decrease of $D_{K,\Theta}^{\min}(M_t^{\text{mem}})$. For instance, in Figure 12.1, we illustrate two strategies for expanding M_t^{mem} , denoted as ${}^1 M_t^{\text{mem}}$ and ${}^2 M_t^{\text{mem}}$. The first strategy, ${}^1 M_t^{\text{mem}}$, represents random exploration, while the second, ${}^2 M_t^{\text{mem}}$, follows a hypothesis-driven approach [1005] 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 [1006]. 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 $D_{K,\Theta}^{\min}(M_t^{\text{mem}})$, subject to resource constraints. This approach is generally more efficient than random exploration for expanding M_t^{mem} , leading to $D_{K,\Theta}^{\min}({}^2 M_t^{\text{mem}})$ descending faster than $D_{K,\Theta}^{\min}({}^1 M_t^{\text{mem}})$.

In general, the knowledge discovery process proceeds iteratively, repeatedly optimizing the world model parameter θ to approach $\theta_{K,t}^*$ and expanding M_t^{mem} in a rational manner to accelerate the decrease of $D_{K,\Theta}^{\min}(M_t^{\text{mem}})$. The ideal state is achieving epistemic completeness, i.e., $D_{K,\Theta}^{\min}(M_t^{\text{mem}}) = 0$, meaning zero discrepancy between the agent's prediction and the real-world phenomena. However, for a specific agent, a discovery bound may exist, where $D_{K,\Theta}^{\min}(M_t^{\text{mem}})$ 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 [1007]. 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

 YPLICAL 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 infor-

mation processed in a way that affects the probability distribution of unknown information $P_\theta(\mathbf{x}_U|M_t^{\text{mem}})$, reduces $D_K(\theta, M_t^{\text{mem}})$, and facilitates decision-making.

In principle, external scientific knowledge has been shown to be useful in improving agent performance in reasoning and decision-making [1008, 1009]. 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 agentic AI scientist 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 [1010, 1011, 1012, 1013, 1014] (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.

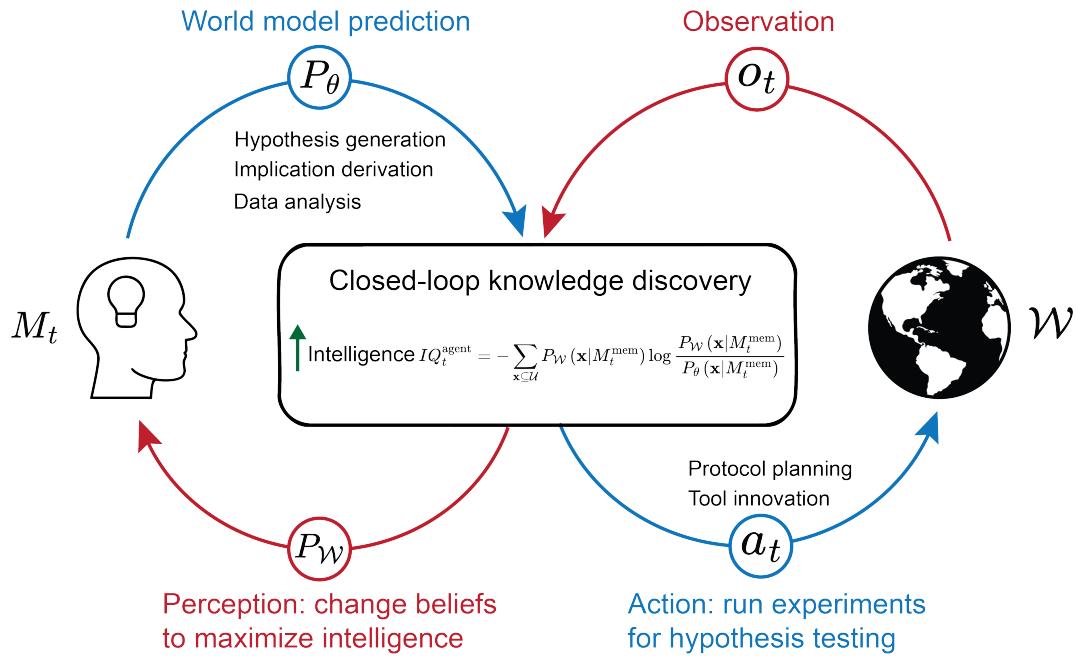


Figure 12.2: Closed-loop knowledge discovery for sustainable self-evolution of an AI scientist. The agent aims to iteratively enhance its intelligence IQ_t^{agent} through hypothesis generation and testing, as well as through data analysis and implication derivation. When interacting with the physical world \mathcal{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 \mathcal{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 [995].

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 [1006]. 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 [1015, 1016]; in this discussion, we define a hypothesis that survives falsification as a *justified true hypothesis* [1017, 991]. 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 $D_{K,\Theta}^{\min}(M_t^{\text{mem}})$. 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 [1018]. Si et al. [1019] 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. [1020] 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. [1021] and Baek et al. [1022] proposed leveraging teamwork—such as collaborative discussions and agent critics—to produce novel and effective scientific hypotheses. In addition, Gower et al. [1023] 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. [1024] 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. [1025] 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. [1026] 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%) [1027]. This knowledge also advances our understanding of cancer, the immune system, and aging. Similarly, Gottweis et al. [1006] 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. [1028] 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 [1029]. Wang et al. [1030] 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. [1031] 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. [1032] 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. [1033] 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 $D_K(\theta, M_t^{\text{mem}})$, they enhance execution efficiency and effectiveness in refining the probability distribution of unknown information, $P_\theta(x_U | M_t^{\text{mem}})$, 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. [1034] 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. [1035] 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. [1036] 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 [1037], including: (1) a sequence analysis tool that ranks candidate point mutations using log-likelihood ratios from the ESM protein language model [1038]; (2) a structure evaluation tool that extracts interface pLDDT scores from AlphaFold-Multimer predictions [1039], offering a proxy for antibody-antigen binding affinity; and (3) an energy estimation tool built on Rosetta [1040] 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-CoV-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 $D_K(\theta, M_t^{\text{mem}})$ 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. [1041] 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. [1042] 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.

When no new external observations are needed, agents can still synthesize and communicate knowledge by retrieving evidence, organizing it, and drafting coherent narratives. Shao et al. [1043] propose STORM, a workflow that researches a topic through multi-perspective question asking, verifies sources, plans a sectioned outline, and writes a Wikipedia-like article from scratch. By turning dispersed information into a structured, cited article, such systems improve an agent’s practical understanding and decision quality through internal reasoning and retrieval rather than physical discovery. 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

HE 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 [1044] 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 [1045]. 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 [1046]. Model Context Protocol (MCP) [1047] also has the potential to further streamline the integration of scientific tools into agentic AI systems.

A more accessible alternative for small teams is the *cloud lab* or *science factory* [1048], where responsibility for device engineering shifts from individual laboratories to dedicated user facilities or commercial service providers. For instance, Boiko et al. [1049] 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 [1050]. 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. [1034] 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 [1051] 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. [1048] 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 [1052]. This architecture has been used to automate experiments in both the biological and physical sciences [1053]. Similarly, Fernando et al. [1054] integrate a Robotic Operating System 2 (ROS2) compatible robot into the Bluesky experimental orchestration framework. Lo et al. [1055] 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 [1056]. 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. [1057] 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. [1008] 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 [996] proposed the Abstraction and Reasoning Challenge (ARC) to assess 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 ~ 75% average human performance [1058, 1059]. Zhu et al. [1060] suggested a four-level classification of AI intelligence, including L1 (arbitrating disputes), 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 [1061], tree-of-thoughts [99], and [97]. 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 [1062, 828]. 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 [1041]. However, this mitigation does not resolve the intrinsic deficiency in numerical understanding, which poses a potential risk to scientific reasoning. Moreover, Yu et al. [1063] 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. [1064].

12.3.3 Challenges in Integrating Prior Knowledge

Prior knowledge is a crucial factor for higher intelligence. As discussed in Section 12.1, the agent's prior knowledge, M_t^{mem} , helps decrease $D_K(\theta, M_t^{\text{mem}})$ 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 [1065]. 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 [1066] 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 [1066]. Establishing a system to assess the levels of evidence [1067] of different knowledge fragments—such as quantifying reliability and verifying references—may be necessary for effective knowledge fusion.

12.4 Summary and Discussion



In this chapter, we explored the pivotal role of intelligent agents in driving autonomous scientific discovery, emphasizing how agents can foster self-evolution and innovation cycles through iterative knowledge acquisition, hypothesis testing, and continuous improvement. We proposed a systematic framework for measuring agent intelligence, defining it via the Kullback–Leibler divergence between the agent’s predicted and the real-world probability distributions of unknown information. This statistical measure provides a robust method for quantifying an agent’s capability to make accurate, predictive models of natural phenomena.

We discussed three core interactions between agents and scientific knowledge: (1) hypothesis generation and testing, (2) protocol planning and tool innovation, and (3) data analysis and implication derivation. Hypothesis generation is crucial for formulating valuable scientific conjectures, while hypothesis testing through computational or experimental validation is essential for transforming conjectures into verified knowledge. Protocol planning and tool innovation enable agents to orchestrate and optimize experimental workflows effectively, leveraging existing tools or creating novel ones to enhance scientific exploration efficiency. Data analysis and implication derivation allow agents to extract meaningful insights from existing knowledge and observations, employing both deductive logic and inductive pattern recognition.

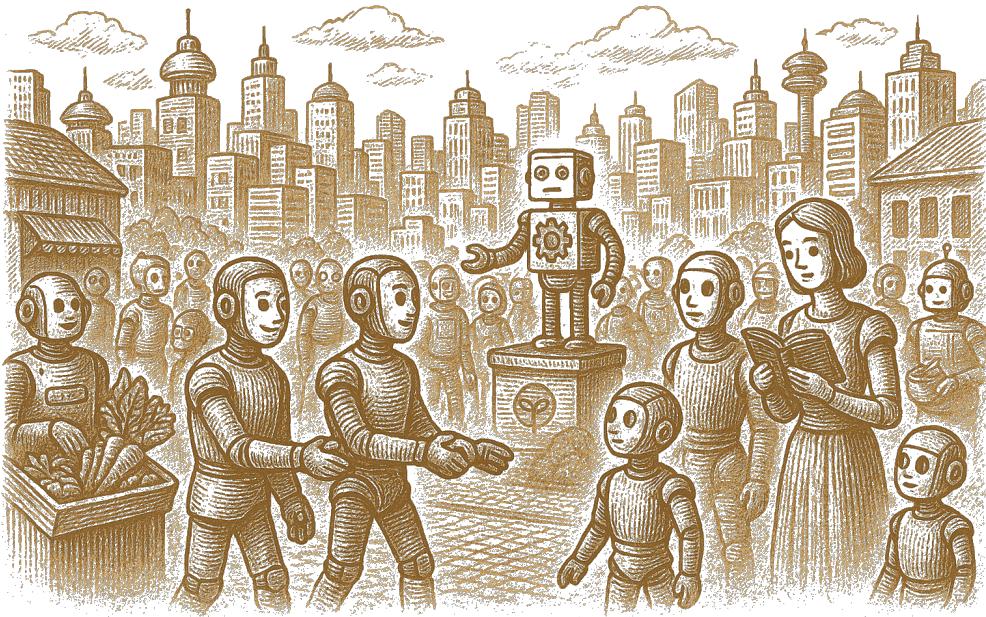
We reviewed notable advancements in autonomous scientific discovery, including systems like the AI Scientist, ChemOS 2.0, and the Virtual Lab, showcasing the potential of agents to independently navigate complex scientific processes across theoretical, computational, and experimental domains. However, despite these successes, significant challenges remain in achieving fully autonomous agent-driven scientific discovery.

Three primary technological challenges were identified: (1) real-world interaction, specifically the limited availability and complexity of physical APIs necessary for seamless laboratory integration; (2) complex reasoning, where current language models still struggle with sophisticated logic, numerical computation, and symbolic reasoning; and (3) integration of prior knowledge, particularly regarding inaccessible or unpublished data, empirical expert heuristics, and situational knowledge. Addressing these limitations requires concerted advancements in interdisciplinary integration, robust reasoning methodologies, and comprehensive strategies for assimilating diverse knowledge sources.

Looking forward, enhancing agent autonomy in scientific discovery will necessitate overcoming these barriers through interdisciplinary collaborations, robust experimental validations, and methodological innovations in reasoning and knowledge management. Ultimately, achieving higher autonomy in scientific agents holds the promise of accelerating human knowledge expansion, driving technological innovation, and sustaining an iterative cycle of continuous agent and human evolution.

Part III

Collaborative and Evolutionary Intelligent Systems



A society of minds, where agents co-evolve through communication, specialization, and collaboration.

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 [1068]. 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 [1069], 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*[44] and the theory of mind[1070, 1071], 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 LLM-based 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.

In this part, 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 different 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, and the collaborative paradigms and decision-making methodologies that facilitate effective agent-agent and human-agent communication. Furthermore we discuss the collective intelligence and evolution mechanism in Chapter 15. Finally, in Chapter 16, we discuss evolutionary processes, highlighting adaptive learning methods, continuous knowledge sharing, and mechanisms for iterative improvement that collectively improve

MAS performance. Through this part, we aim to identify current achievements, discuss existing challenges, and highlight promising research directions for collaborative and evolutionary intelligent systems. The following figure shows a taxonomy of research works about LLM-based multi-agent systems.

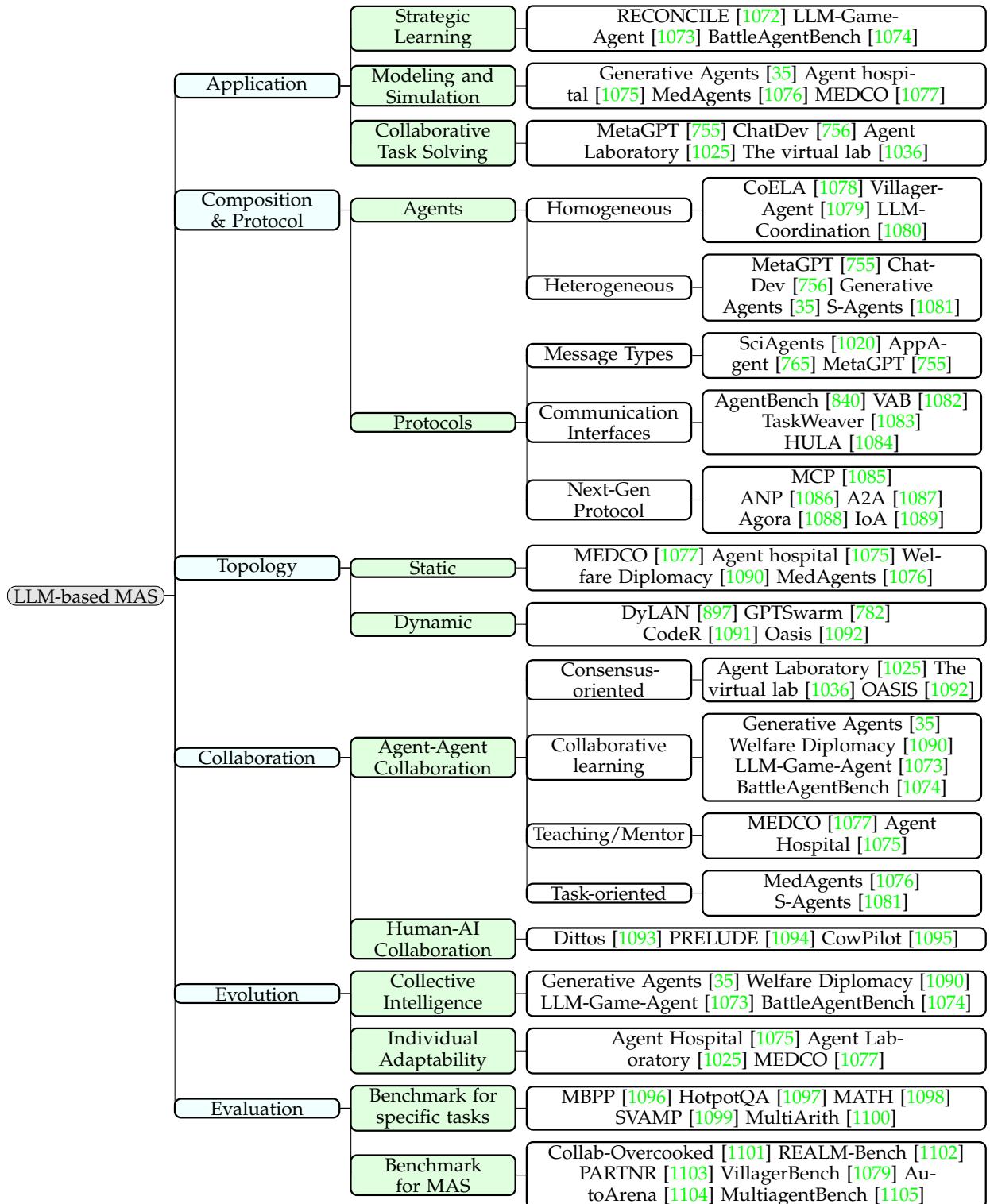


Figure 12.3: A taxonomy of research works about LLM-based Multi-Agent Systems.

Chapter 13

The Framework and Categories of Multi-Agent Systems

 N 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 chapter categorizes LLM-MAS into three categories based on distinct combinations of collaboration goals and norms: *strategic learning*, *modeling and simulation*, and *collaborative task solving*. Figure 13.1 gives an overview of these categories in LLM-MAS. 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.
- **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.
- **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.

In the following, we first introduce a formal framework for multi-agent systems. Then we examine these categories in detail, highlighting how each of them leverages the capabilities of LLMs to shape agent behaviors and interactions.

13.1 From Foundation Agent to Foundation Agents: A Society-Level Formalism

 E treated a *single Foundation Agent* in Chapter 1: a system that perceives, learns, reasons, and acts in a loop while maintaining a structured mental state (memory, world model, emotion, goals,

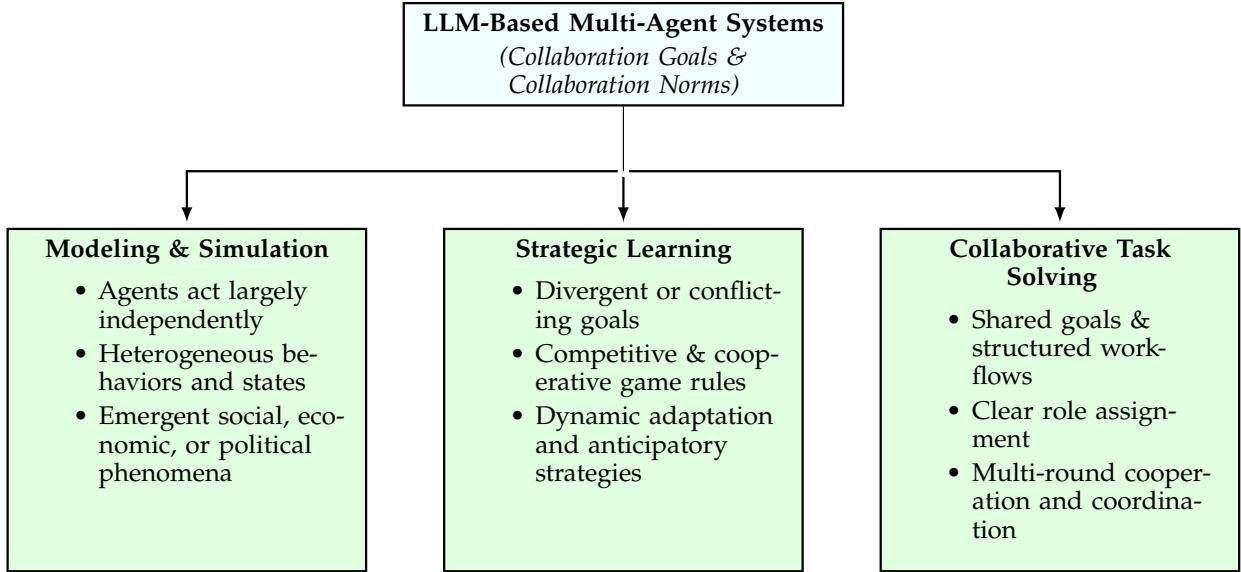


Figure 13.1: An overview of three major collaboration types in LLM-based MAS: *Modeling & Simulation*, *Strategic Learning*, and *Collaborative Task Solving*. Each category is distinguished by how agents' goals and norms are set (independent vs. divergent vs. shared) and how they coordinate.

reward, . . .). Real deployments rarely stop at one. We field teams of agents; we mix AI agents with humans; we drop them into worlds governed by currencies, laws, APIs, and social norms. This section lifts the single-agent loop to the *multi-agent / agent-society* level while keeping all notation consistent with the *Foundation Agent* framework presented in Chapter 1. The goal is a clean bridge: take $n = 1$ and collapse the shared structures, and you recover the single-agent loop exactly.

We proceed in three steps. First, we formalise the world-as-society view that underlies the Foundation Agent programme. Second, we introduce the additional structures needed when many agents co-inhabit that world: a shared workspace, coordination mechanisms, and social-system constraints. Third, we write down the closed interaction loop: perception, cognition, coordination, execution, and world update, so that analysis, simulation, and implementation all read from the same page. Table 13.1 lists the additional symbols introduced for the multi-agent setting and how they relate to the single-agent notation from Table 1.2.

13.1.1 World as Environment *plus* Intelligent Beings under Social Systems

In the Foundation Agent view, the *world* is never a neutral backdrop. It already contains other intelligent beings (AI or human) and the layered *social systems*, such as financial, legal, cultural, technical infrastructure, that shape what information flows, what actions are allowed, what resources cost, and how rewards are allocated.

Definition 13 (World as structured society). Let

$$\mathcal{W} = \text{SocialSystems}(\mathcal{S}, \mathfrak{B}, \Sigma)$$

denote a world composed of

- an *environment state space* \mathcal{S} with states $s_t \in \mathcal{S}$;
- a set \mathfrak{B} of *intelligent beings* (AI agents, humans, organisations, . . .);

Table 13.1: Additional notation for the Foundation Multi-Agent System formalism. Symbols extend the single-agent notation in Table 1.2.

Symbol	Meaning
$A = \{1, \dots, n\}$	Index set of designated AI agents in the society.
\mathfrak{B}	All intelligent beings in the world (agents, humans, orgs). Our $A \subseteq \mathfrak{B}$.
Σ	Social systems: rules, norms, infrastructures constraining perception, communication, resources, and reward.
$B_t \in \mathcal{B}$	Shared workspace / blackboard state at turn t (shared memory, interaction log, group goals, resource ledger, ...).
P^i	Perception for agent i ; shaped by M_{t-1}^i , B_{t-1} , and Σ .
L^i	Learning / private mental-state update for agent i .
R^i	Reasoning / action-proposal function for agent i .
U^i	Workspace message constructor for agent i (optional but convenient).
K	Coordination operator aggregating proposals under social-system rules.
E^i	Effector / actuation mapping for agent i (high-level \rightarrow executable).
T	Environment transition given executed joint actions.
Ω	Shared-workspace transition given aggregated updates and current private states.
\mathbf{a}_t	Authorised joint action vector after coordination.
\mathbf{u}_t^{sh}	Aggregated set of shared updates/messages applied to B_t .

- a collection Σ of *social systems* (markets, legal codes, communication infrastructures, cultural norms), which regulate information access, resource exchange, interaction protocols, and incentives among members of \mathfrak{B} and the environment \mathcal{S} .

In what follows we focus on a designated subset of beings, our AI agents, interacting with the rest of \mathcal{W} . Humans may appear explicitly as additional beings or implicitly through Σ (e.g., policy limits, budget approvals, safety reviews).

13.1.2 From Single Agent to Multi-Agent Systems

Now we introduce the additional structures needed when many agents co-inhabit a shared world.

From One Mental State to Many: Private vs. Shared Recall the single-agent notation: observations $o_t \in \mathcal{O}$; actions $a_t \in \mathcal{A}$; mental state $M_t \in \mathcal{M}$ decomposed into memory, world model, emotion, goals, reward signals, and so on; cognition $C = (L, R)$ updating M_t and producing a_t ; environment transition T (Table 1.2). With n agents we index each agent $i \in \{1, \dots, n\}$:

$$M_t^i \in \mathcal{M}^i, \quad o_t^i \in \mathcal{O}^i, \quad a_t^i \in \mathcal{A}^i.$$

Different agents may have different modalities, tools, or roles, so the spaces need not match across i (heterogeneous teams are common).

In addition to these *private* mental states we introduce a *shared workspace*, the arena where agents leave messages, post artefacts, negotiate sub-goals, and pool intermediate results.

Definition 14 (Shared workspace / blackboard). At time t the team-level shared state is

$$B_t = \langle M_t^{\text{sh}}, I_t, G_t^{\text{grp}}, R_t^{\text{alloc}}, \dots \rangle,$$

where M_t^{sh} is shared memory (documents, code, structured data), I_t logs interaction history, G_t^{grp} records currently active group or task goals, and R_t^{alloc} tracks team-level resource budgets or locks. Ellipses indicate optional additional channels (e.g., reputation ledgers, conflict flags). We write $B_t \in \mathcal{B}$ for the workspace state space.

For completeness, we keep the earlier decomposition (cf. Sec. 1.3.2):

$$M_t^i = \langle M_t^{i,\text{mem}}, M_t^{i,\text{wm}}, M_t^{i,\text{emo}}, M_t^{i,\text{goal}}, M_t^{i,\text{rew}}, \dots \rangle.$$

Coordination Surfaces: How Social Systems Enter the Loop Even when every agent is competent alone, multi-agent behaviour hinges on coordination. In our formalism, coordination appears in two places:

1. **Information shaping.** Social systems Σ can gate or transform what each agent sees: privacy rules, role-based access, rate limits, safety filters.
2. **Joint decision procedures.** The team may follow protocols such as voting, market-style bidding, role delegation, or consensus review that turn a set of *proposed* actions into what is actually executed.

We collect these influences in a coordination operator

$$K : \Sigma \times B_{t-1} \times \{\tilde{a}_t^i, u_t^i\}_{i=1}^n \longrightarrow (\mathbf{a}_t, \mathbf{u}_t^{\text{sh}}),$$

defined below after we explain where the proposals \tilde{a}_t^i and messages u_t^i come from.

Local Perception and Cognition per Agent Each agent runs a perception–cognition update much like the single-agent loop, but conditioned on both the external world and the current shared workspace.

- **Perception.** Agent i forms an observation

$$o_t^i = P^i(s_t, M_{t-1}^i, B_{t-1}; \Sigma), \quad (13.1)$$

where the semicolon reminds us that the form of perception (what sensors are available, what data are redacted) can depend on social-system constraints Σ .

- **Learning (state update).** The agent updates its private mental state:

$$M_t^i = L^i(M_{t-1}^i, a_{t-1}^i, o_t^i, B_{t-1}; \Sigma). \quad (13.2)$$

Online fine-tuning, memory write, belief update, and affect change all live here.

- **Reasoning (action proposal).** Given its updated mental state (and the shared context) the agent proposes its next move:

$$\tilde{a}_t^i = R^i(M_t^i, B_{t-1}; \Sigma), \quad (13.3)$$

and may also produce an associated workspace message or update payload

$$u_t^i = U^i(M_t^i, \tilde{a}_t^i), \quad (13.4)$$

such as “here is my partial plan”, “I need GPU hours”, or “result attached”. We group (\tilde{a}_t^i, u_t^i) as the agent’s *behaviour proposal* at turn t .

Team Coordination and Execution Behaviour proposals are not necessarily executed as-is. The coordination operator K takes all proposals plus the current workspace and applicable social-system rules and decides what the team actually does and how the shared state changes.

$$(\mathbf{a}_t, \mathbf{u}_t^{\text{sh}}) = K(\Sigma, B_{t-1}, \{\tilde{a}_t^i, u_t^i\}_{i=1}^n), \quad (13.5)$$

where $\mathbf{a}_t = (a_t^1, \dots, a_t^n)$ are the *authorised* actions to be executed externally and \mathbf{u}_t^{sh} the set of workspace updates (possibly filtered, merged, or reordered).

Each authorised action then passes through the agent's effectors:

$$a_t^{i,\text{exec}} = E^i(a_t^i). \quad (13.6)$$

World and Workspace Transitions Once authorised actions are executed, the environment responds. Allowing for stochasticity:

$$s_{t+1} \sim T(s_t, \{a_t^{i,\text{exec}}\}_{i=1}^n). \quad (13.7)$$

Meanwhile, the shared workspace is updated by an internal transition function Ω that applies the accumulated shared updates plus any system-level bookkeeping (resource debits, goal progression, interaction logs):

$$B_t = \Omega(B_{t-1}, \mathbf{u}_t^{\text{sh}}, \{M_t^i\}_{i=1}^n; \Sigma). \quad (13.8)$$

Note that B_t is indexed by the *post-coordination* turn t (same logical time as the proposals it incorporates). If greater temporal fidelity is needed (asynchronous teams, streaming logs), B_t can be made event-driven rather than turn-synchronous; the algebra here assumes a batched logical turn for clarity.

13.1.3 The Foundation MAS Loop

Collecting Eqs. (13.1)–(13.8), one logical turn of a Foundation Multi-Agent System unfolds as:

$$o_t^i = P^i(s_t, M_{t-1}^i, B_{t-1}; \Sigma), \quad i = 1, \dots, n, \quad (13.9)$$

$$M_t^i = L^i(M_{t-1}^i, a_{t-1}^i, o_t^i, B_{t-1}; \Sigma), \quad i = 1, \dots, n, \quad (13.10)$$

$$\tilde{a}_t^i = R^i(M_t^i, B_{t-1}; \Sigma), \quad i = 1, \dots, n, \quad (13.11)$$

$$u_t^i = U^i(M_t^i, \tilde{a}_t^i), \quad i = 1, \dots, n, \quad (13.12)$$

$$(\mathbf{a}_t, \mathbf{u}_t^{\text{sh}}) = K(\Sigma, B_{t-1}, \{\tilde{a}_t^i, u_t^i\}_{i=1}^n), \quad (13.13)$$

$$a_t^{i,\text{exec}} = E^i(a_t^i), \quad i = 1, \dots, n, \quad (13.14)$$

$$s_{t+1} \sim T(s_t, \{a_t^{i,\text{exec}}\}_{i=1}^n), \quad (13.15)$$

$$B_t = \Omega(B_{t-1}, \mathbf{u}_t^{\text{sh}}, \{M_t^i\}_{i=1}^n; \Sigma). \quad (13.16)$$

The loop then advances to $t+1$ and repeats. Termination can be triggered by environmental conditions (task complete, time out), social-system rules (budget exhausted, human override), or internal goal satisfaction (group goal reached).

Reduction to the Single Foundation Agent A good generalisation collapses cleanly. Let $n = 1$, drop the shared workspace (or set $B_t = \emptyset$), and let Σ impose no additional rules. Then Eqs. (13.9)–(13.16) reduce to

$$o_t = P(s_t, M_{t-1}), \quad M_t = L(M_{t-1}, a_{t-1}, o_t), \quad a_t = R(M_t), \quad s_{t+1} = T(s_t, E(a_t)),$$

which is exactly the Foundation Agent loop of Def. 1. Thus the multi-agent formalism is a conservative extension: nothing is lost, only shared structure and coordination are added.

Why This Level of Formality Matters Writing out the loop may feel pedantic, but it buys us three things we need for the rest of the book:

1. **Comparability across systems.** Whether you use AutoGen-style chatting LLMs, workflow graphs, or embodied robot swarms, you can map components to P^i, L^i, R^i, K, T , and Ω and compare apples to apples.
2. **Analytical hooks.** Stability of collaboration, failure propagation, communication load, and reward hacking under social constraints each becomes an object of study once the interfaces are explicit.
3. **Implementation discipline.** Real code bases drift; prompts bleed into memory; ad-hoc globals creep in. The formal surfaces in Eqs. (13.9)–(13.16) force us to decide where each update belongs, which in turn simplifies logging, evaluation, and reproducibility.

In short, the Foundation MAS formalism extends the perception–cognition–action loop from a single mind to a society of minds [44], all situated inside a structured world shaped from the inside out [45] and capable (in principle) of Bayesian-style active engagement with uncertainty [46]. We will draw on these perspectives repeatedly as we examine coordination, communication, learning at scale, and safety in the chapters ahead.

Next, we examine different categories of multi-agent systems in more details.

13.2 Strategic Learning: Cooperation vs. Competition

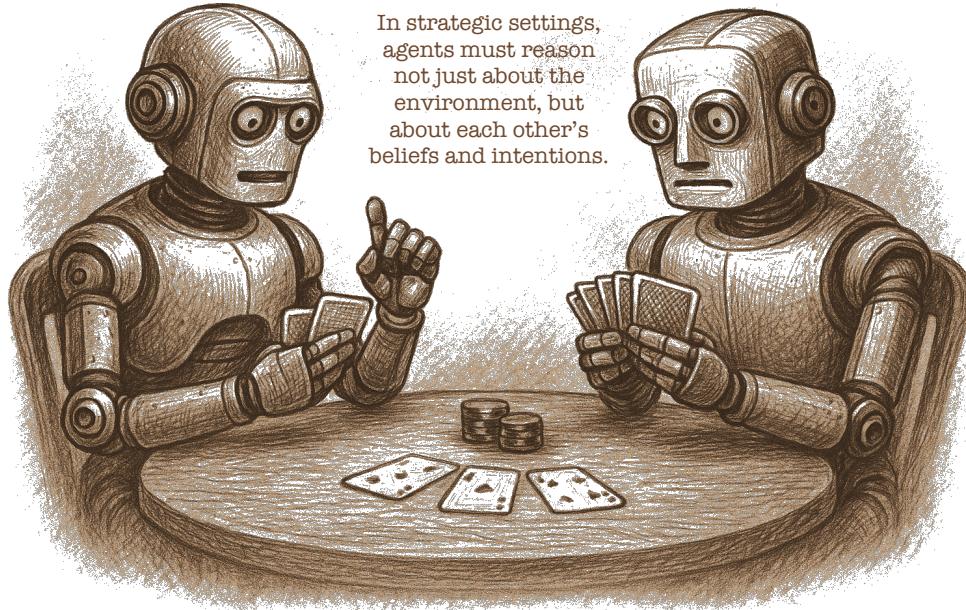


Figure 13.2: An LLM-based robot doctor interacts with a robot patient, illustrating the simulation of clinical workflows in healthcare environments. Such agent-based simulations enable iterative refinement of treatment strategies, protocol evaluation, and virtual experimentation in realistic medical settings.

STATEGIC LEARNING describes the capacity of agents to dynamically anticipate, interpret, and influence the actions of other agents within game-theoretic environments, whether those settings are competitive, cooperative, or a combination of both [1106]. Agents refine their strategies continuously, adjusting based on newly acquired information. This iterative process frequently leverages foundational game theory concepts such as Nash equilibria [1107], Bayesian games [1108, 1068, 1109], and repeated interactions [1110, 1111]. With the sophisticated linguistic capabilities of LLMs, strategic learning increasingly incorporates nuanced interactions such as dialogue, persuasion, and implicit negotiation, enhancing

traditional game-theoretic reasoning [1109, 1112, 1113, 1114]. Figure 13.2 provides a visual metaphor for such settings, depicting two robots engaged in a game of Texas Hold'em, where bluffing, hidden information, and adaptive reasoning exemplify the essence of strategic learning.

Economic applications demonstrate the significant potential of multi-agent strategic simulations by offering valuable insights into market dynamics and negotiation strategies, reflecting both competitive and cooperative elements. For instance, research by Li et al. [1115] and Horton [1108] illustrates how agents driven by LLMs can realistically simulate hiring scenarios, demonstrate rational decision-making in structured economic experiments, and even predict stock market trends. Additionally, Zhao et al. [1116] developed a competitive environment using GPT-4, highlighting strategic interactions between restaurant and customer agents aimed at optimizing profits and satisfaction through realistic pricing and bidding strategies. Similarly, Xia et al. [1117] explores LLM-driven Buyer–Seller bargaining scenarios, and Sreedhar and Chilton [1118] employs ultimatum game simulations to inform policymaking by modeling human-like strategic behaviors.

Strategic learning extends far beyond conventional market contexts, appearing in any scenario involving resource allocation, alliance formation, or balancing competitive and cooperative incentives. Notably, studies on multi-commodity competitions [1119, 1116] demonstrate how agents strategically negotiate to maximize individual gains. In sustainability-focused applications, agents coordinate consumption of resources efficiently and responsibly [1120].

In the realm of gaming, social deduction games such as Werewolf, Chameleon, Avalon, and Jubensha exemplify how agents must navigate complex dynamics involving deception and collaboration [1121, 1122, 1123, 205, 1073, 1124, 1125, 1126, 1127]. Research by Xu et al. [1128] and Du and Zhang [1122] demonstrates LLM-based agents' effectiveness in subtly orchestrating deception and cooperation. Other studies emphasize agents' ability to adapt strategically over multiple rounds in Avalon [1124, 1129, 1125, 1126]. Moreover, Wu et al. [1127] advances this domain further, illustrating autonomous agent interactions within the intricate narratives of the Jubensha murder mystery genre. Diplomatic simulations, such as those presented by Hua et al. [1130] and Jin et al. [1131], utilize LLM-driven agents to realistically emulate complex geopolitical negotiations and alliance dynamics on a global scale.

The critical advantage of LLM-powered strategic learning lies in the seamless integration of rigorous game-theoretic logic with natural language reasoning. This integration empowers agents to interpret sophisticated instructions, participate in persuasive interactions, and adapt more effectively to novel or unstructured contexts. As a result, LLM-driven strategic agents offer unparalleled potential for accurately modeling complex real-world interactions, ranging from economic competition and social negotiation to diplomatic strategy, significantly surpassing traditional rule-based or purely numeric approaches.

13.3 Modeling Real-World Dynamics

 MODELING AND SIMULATION is a foundational application area for LLM-based multi-agent systems (LLM-MAS), aimed at capturing and reproducing complex social, economic, and political phenomena at scale. Leveraging the contextual reasoning and language understanding capabilities of LLMs, these systems enable the design of diverse, role-driven agents whose evolving behaviors reflect real-world dynamics with greater fidelity.

Unlike strategic learning setups that emphasize explicit goals of cooperation or competition, simulation environments typically involve agents acting autonomously according to their individual roles, preferences, and local contexts [1132]. This makes LLM-MAS a powerful tool for exploring emergent dynamics and evaluating potential interventions in open-ended, domain-specific scenarios.

In healthcare, Li et al. [1075] introduce *Agent Hospital*, where LLM-powered doctor agents interact with simulated patients to iteratively refine treatment strategies. This virtual setting supports the evaluation

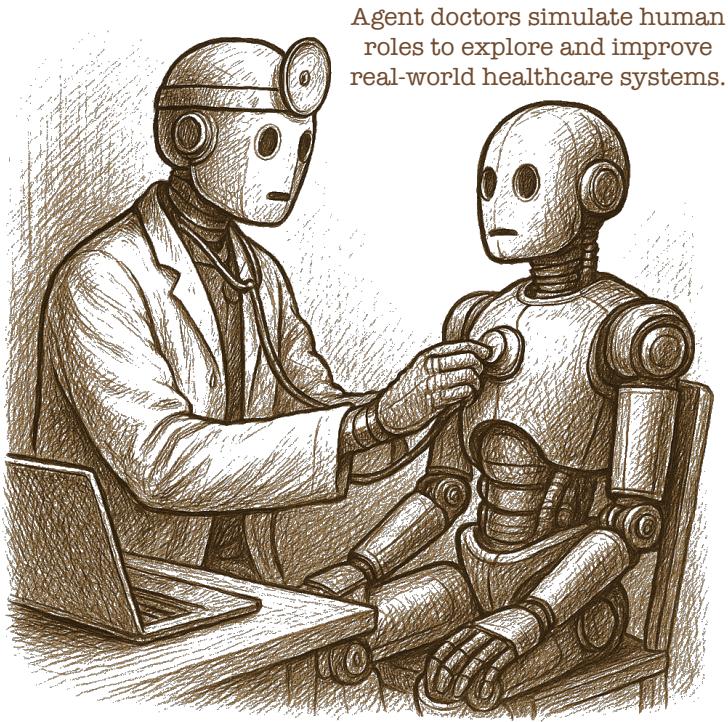


Figure 13.3: An LLM-based robot doctor interacts with a robot patient, illustrating the simulation of clinical workflows in healthcare environments. Such agent-based simulations enable iterative refinement of treatment strategies, protocol evaluation, and virtual experimentation in realistic medical settings.

of hospital management protocols, medical training paradigms, and “what-if” scenarios under realistic constraints. Figure 13.3 captures this idea by depicting an LLM agent simulating a doctor’s role in an interaction with a robot patient, demonstrating how agent-based systems can emulate clinical environments for experimentation and training. In a parallel line of work, Li et al. [1133] present *EconAgents*, a framework for simulating macroeconomic systems from the bottom up. Their LLM-driven agents model household-level behaviors such as employment choices, consumption patterns, and savings decisions, offering a more expressive and adaptive alternative to traditional numeric or rule-based economic models [1134].

Political science has also begun to embrace LLM-based simulation. For example, Zhang et al. [1135] and Tran et al. [1134] simulate electoral processes and policy formation, capturing how candidate behavior, voter sentiment, and public discourse co-evolve to influence election outcomes. These simulations highlight the role of communication and strategy in shaping collective political behavior.

Beyond politics and economics, LLM-MAS has been used to study a range of social and cultural dynamics. For instance, Ferraro et al. [1136] and Gao et al. [309] simulate the propagation of language and emotion through social networks to better understand how opinions and sentiment clusters emerge. Chuang et al. [1137] explore how opinion dynamics unfold under different interaction patterns and network topologies, while Liu et al. [1138] investigate the diffusion of misinformation and the conditions under which it accelerates or stalls.

Large-scale platforms such as GenSim [1139] and OASIS [1092] further extend these capabilities by supporting simulations with tens of thousands or even millions of agents. These systems make it possible to study population-scale dynamics, such as echo chambers, group polarization, and viral content spread, under realistic behavioral and infrastructural constraints.

What sets LLM-based simulation apart is its ability to incorporate both structural features (like institutional rules or network connections) and cognitive-linguistic elements (such as framing, persuasion, and narrative). This dual modeling capacity allows researchers to examine complex phenomena that are difficult to capture using numeric abstractions alone, including cultural transmission, belief shifts, and discursive influence.

13.4 Collaborative Task Solving with Workflow Generation

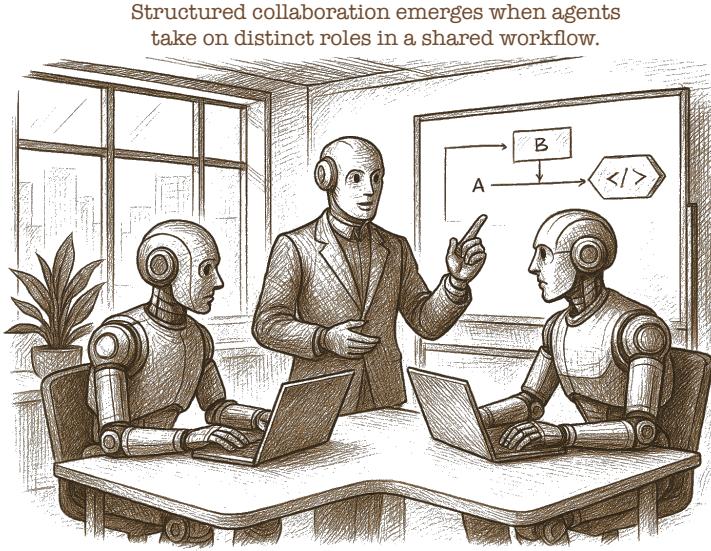


Figure 13.4: Three LLM-based agents, each assigned a distinct role (programmer, architect, and project manager), collaboratively solve a software development task through structured dialogue and workflow coordination. This illustrates the paradigm of collaborative task solving, where multi-agent systems function as role-specialized teams aligned on a shared objective.



COLLABORATIVE TASK SOLVING brings multiple agents together to achieve a clearly defined objective through a structured, coordinated workflow. Unlike strategic learning, which often involves competition or negotiation, or modeling and simulation, where agents act independently, collaborative systems organize agents into cohesive teams that operate through sequential or parallel stages of problem-solving.

Agents are typically assigned explicit roles, such as “Planner”, “Implementer”, or “Evaluator”, and work within stage-based frameworks to ensure that tasks are completed efficiently and accurately. These roles reduce ambiguity and facilitate coordination, especially when tackling complex, multi-step objectives. An example of this collaborative structure is depicted in Figure 13.4, where agents act in distinct roles within a shared task environment. Systems like MetaGPT [755], CAMEL [975], Communicative Agents [1140], and the framework in Zhang et al. [1078] illustrate how clearly defined responsibilities and communication protocols enable LLM-based agents to function as cohesive teams. A typical collaborative workflow might begin with one agent analyzing the task, followed by another drafting a solution plan, a third implementing components, and a final agent verifying or refining outputs. These agents often communicate through natural language dialogues, using iterative rounds of message passing to align on goals and clarify uncertainties. This design is especially valuable for large or ambitious projects, where sub-tasks can be distributed among domain-specialized agents. Language-based coordination enables agents to flexibly reinterpret instructions and delegate responsibilities without requiring rigid symbolic planning.

Although collaborative multi-agent systems have seen early success in software development, such as coding, debugging, and testing, their potential extends further. *Agent Laboratory* [1025], for example, applies

this paradigm to scientific discovery: agents propose hypotheses, design and simulate experiments, analyze results, and revise future inquiries. This mirrors the real-world research cycle and opens opportunities for AI-assisted exploration in literature review, policy design, and complex data analysis.

Compared to other LLM-based multi-agent paradigms, collaborative task solving emphasizes reliability and interpretability. With preassigned roles and well-defined workflows, these systems reduce unpredictability and emergent behavior, which is an advantage in domains requiring precision, compliance, or traceability. At the same time, ongoing research explores how to preserve creative flexibility, ensuring that agents can contribute novel insights while operating within a shared procedural structure.

Together, the three paradigms discussed in this chapter: *strategic learning, modeling and simulation*, and *collaborative task solving*, illustrate the diversity and richness of LLM-based multi-agent systems. Each emphasizes different strengths of language-driven intelligence, enabling agents to reason, communicate, and coordinate in ways that extend well beyond traditional rule-based systems.

13.5 Summary and Discussion

HIS chapter introduced a comprehensive framework and categorization for LLM-based multi-agent systems (LLM-MAS), emphasizing how different combinations of collaboration goals and norms fundamentally shape agent interactions and system behavior. Through our formalized Foundation Multi-Agent System framework, we illustrated the transition from individual Foundation Agents, capable of perception, cognition, and action loops, to sophisticated multi-agent systems or agent societies regulated by shared workspaces, coordination mechanisms, and overarching social systems.

We categorized LLM-MAS into three distinct yet complementary paradigms: *strategic learning, modeling and simulation*, and *collaborative task solving*. Each category highlights unique facets of agent interaction and system objectives:

- **Strategic Learning** emphasized dynamic, game-theoretic interactions. Here, agents leverage sophisticated LLM-based reasoning capabilities to anticipate, interpret, and influence each other's actions. These systems often capture nuanced competitive and cooperative dynamics, proving particularly valuable in economics, social deduction games, and diplomacy simulations. The fusion of game-theoretic rigor with linguistic reasoning offers substantial advantages for realistically modeling complex human-like behaviors.
- **Modeling and Simulation** showcased scenarios where agents act independently within rich, dynamic environments. This paradigm exploits LLMs' ability to embody diverse roles, behaviors, and decision-making styles, enabling large-scale simulations of economic, political, and social phenomena. By accurately reflecting cognitive, linguistic, and structural dynamics, such simulations offer significant insights into real-world processes like information diffusion, opinion formation, and societal behavior.
- **Collaborative Task Solving** highlighted structured cooperation among agents toward explicitly defined objectives. Utilizing structured workflows and clear role definitions, this category ensures efficient, precise, and predictable outcomes. Its strength lies in its applicability to structured tasks like software development, scientific investigation, literature review, and policy formulation, providing systematic yet flexible cooperation among agents.

Despite their differences, these categories are interconnected, collectively demonstrating the versatility of LLM-based multi-agent frameworks. They illustrate how varying degrees of goal alignment, interaction norms, and structural rigidity influence the complexity and nature of agent interactions, ranging from emergent behaviors to meticulously orchestrated workflows.

Table 13.2: Classification framework for LLM-based multi-agent systems, highlighting different aspects of system design, communication and collaboration. 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/Mentor, C-O = Consensus-Oriented, T-O = Task-Oriented, CL = Collaborative Learning, Dict = Dictatorial, D-B = Debate-Based.

Paper	System Design		Communication			Collaboration	
	Category	Typology	Interface	Agent Type	Interaction	Decision	
Agent Hospital [1075]	M&S	S-D	Text	Het	T/M, C-O	Dict	
Welfare Diplomacy [1090]	M&S	S-L	Code, JSON, Text	Hom	CL	Voting	
MEDCO [1077]	M&S	S-L	Text	Het	T/M, C-O	Dict	
MedAgents [1076]	M&S	S-L	Text	Hom	T-O	Dict	
Generative Agents [35]	M&S	S-D	Visual	Hom	CL	Dict	
RECONCILE [1072]	SL	S-D	Text	Hom	CL	D-B	
Agent Laboratory [1025]	CTS	S-L	Code, Text	Het	C-O, T-O	Dict	
CoELA [1078]	CTS	S-D	Text	Hom	T-O	–	
The virtual lab [1036]	CTS	S-L	Text	Het	C-O, CL	Dict	
SciAgents [1020]	CTS	S-L	Text	Het	T-O	Dict	
S-Agents [1081]	CTS	S-D	Text	Het	T-O, CL	Dict	
GPT-Bargaining [1141]	CTS	S-D	Text	Het	C-O	D-B	
FORD [1142]	M&S	S-D	Text	Het	C-O	D-B	
MADRA [1143]	CTS	S-D	Text	Het	C-O	D-B	
Multiagent Bench [1105]	CTS	S-D	Text	Hom	T-O, CL	D-B	
OASIS [1092]	M&S	D	Text	Het	C-O	–	
S ³ [309]	M&S	S-D	Text	Het	C-O	–	
FPS [1138]	M&S	S-D	Text	Het	C-O	–	
GPTSwarm [1144]	CTS	D	Code, JSON, Text	Hom	T-O	Dict	
ChatEval [1145]	CTS	D	Text	Hom	T-O	Voting	
MetaGPT [755]	CTS	S-L	Code, JSON, Text, Visual	Het	T-O	Dict	
AutoAgents [1146]	CTS	D	Text	Het	T-O	C-O	
AgentCoder [1147]	CTS	D	Code, Text	Het	T-O	D-B	
MASTER [1148]	CTS	S-L	Text	Hom	T-O	D-B	
Reflexion [75]	CTS	D	Text	Het	T-O	D-B	
MACM [1149]	CTS	D	Text, Code	Het	T-O	D-B	
Debate [1150]	CTS	S-D	Text	Het	C-O	D-B	

Table 13.2 compares a selected set of existing research from different perspectives. While we have introduced different categories for MAS, there are more topics to be discussed in detail. In the next chapter, we will delve into the construction of multi-agent systems in greater detail.

Chapter 14

Designing Collaborative Multi-Agent Systems

MULTI-AGENT SYSTEMS (MAS) leveraging large language models (LLMs) enable agents to autonomously collaborate, communicate, and solve complex tasks, offering significant advantages across diverse domains. Effective MAS design involves critical considerations in agent composition, interaction patterns, topological structures, decision-making strategies, and communication protocols. This chapter systematically examines key principles underlying collaborative MAS. We begin by discussing strategies for composing homogeneous and heterogeneous agent teams, emphasizing the role of diversity in enhancing system capabilities. Next, we analyze various static and dynamic network topologies, highlighting their scalability implications and practical trade-offs. Following this, we explore four prominent agent collaboration paradigms: consensus-oriented interactions, collaborative learning, teaching and mentoring, and task-oriented interactions, illustrating how distinct interaction models shape agent behaviors and system effectiveness. We then examine methods for human-agent collaboration, delineating different levels of human involvement ranging from simple task delegation to immersive collaborative partnerships. Furthermore, we address decision-making mechanisms, contrasting centralized and collective approaches to highlight their respective advantages and limitations. Lastly, we provide an overview of contemporary and emerging agent communication protocols, identifying critical attributes essential for interoperability, flexibility, and scalability. By synthesizing these elements, this chapter offers comprehensive insights for designing robust, efficient, and adaptable MAS frameworks capable of effectively addressing real-world challenges.

14.1 Building AI Agent Teams

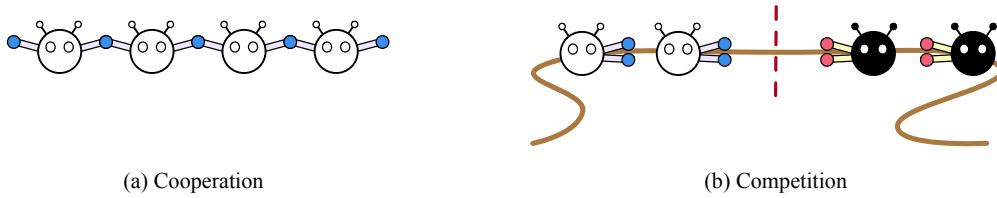


Figure 14.1: Illustration of collaborative and competitive multi-agent dynamics.

IN multi-agent systems (MAS), agents constitute the fundamental units that collaboratively or competitively interact within a shared environment to achieve specific objectives, as shown in

Figure 14.1. The effective composition of agent teams significantly influences the system's overall performance, adaptability, and robustness. Designing an optimal agent team involves careful consideration of the homogeneity or heterogeneity of its members in terms of their capabilities, roles, perceptions, and action spaces. *Homogeneous agent teams*, comprising agents with identical capabilities, can efficiently tackle tasks through straightforward parallelization and simplified coordination. Conversely, *heterogeneous agent teams*, characterized by diverse capabilities and perspectives, excel in handling complex, multidisciplinary tasks by capitalizing on complementary strengths and enabling richer strategic interactions. Moreover, agent teams may dynamically evolve, exhibiting *emergent specializations* through repeated interactions, thereby progressively enhancing the team's versatility and adaptability. Figure 14.2 visually summarizes these three fundamental team structures, illustrating their defining characteristics and differences. This section explores these fundamental approaches to building AI agent teams, detailing their respective strengths, limitations, and practical implications.

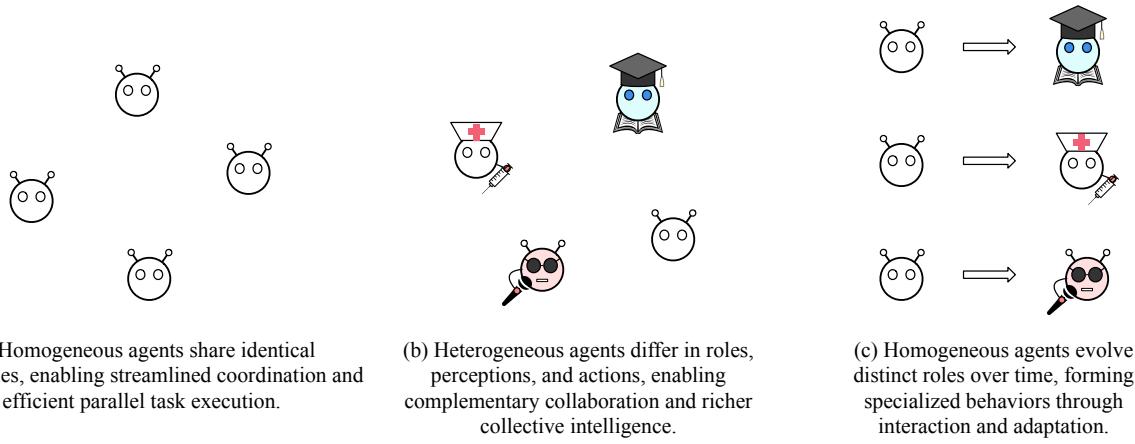


Figure 14.2: Three types of multi-agent teams: *Homogeneous agents* share identical capabilities and collaborate through uniform behavior; *Heterogeneous agents* bring complementary roles, perceptions, and actions to address complex tasks; and *Emergent specialization* arises when identical agents evolve distinct roles through repeated interaction and adaptation.

14.1.1 Homogeneous Agents

Homogeneous agents share identical capabilities, observation spaces, and action spaces, enabling streamlined coordination and efficient parallel task execution. Compared to single-agent systems, homogeneous MAS primarily offer the advantage of task parallelization, where multiple agents simultaneously address different aspects of a common goal. This straightforward design is particularly beneficial in structured environments or tasks that require uniform agent behaviors.

Recent research has effectively applied homogeneous agents in simulated teamwork scenarios, such as collaborative gameplay and household task execution. For instance, Zhang et al. [1078] introduced a cognitive-inspired modular framework enabling large language model (LLM)-based agents to engage in natural language interactions for task coordination. Their system demonstrated efficient labor division, with agents seamlessly assisting each other in object transportation tasks. Similarly, Guo et al. [1151] presented a prompt-based organizational approach, significantly reducing communication overhead and improving team efficiency during routine household activities, including preparing meals and cleaning.

Moreover, homogeneous agents have proven effective in interactive game environments, such as Overcooked and Minecraft. Studies by Agashe et al. [1080] and Dong et al. [1079] leveraged teams of LLM-based homogeneous agents to successfully complete complex cooperative challenges within these games. These

results underscore the strengths of homogeneous teams in predictable scenarios, benefiting from uniform behaviors and straightforward coordination strategies.

14.1.2 Heterogeneous Agents

In contrast, heterogeneous agents exhibit diversity in their roles, perceptions, and available actions, thus significantly enriching the collective problem-solving capability of MAS. Such diversity contributes to more comprehensive decision-making and innovative strategies, as agents leverage complementary skills, perspectives, and information sources to address complex, real-world challenges [1150, 1152]. Heterogeneity among agents is generally observed across three main dimensions: persona-level, observation-space, and action-space heterogeneity. These dimensions often coexist, further enhancing agent-team flexibility and effectiveness.

Persona-level Heterogeneity. Persona-level heterogeneity refers to the diversity in agent profiles or roles, which shapes the way individual agents approach problem-solving and interact within the team. This form of heterogeneity is widely utilized in contemporary LLM-based multi-agent frameworks [1153, 756, 35, 1127]. For example, within software development teams, distinct personas like programmers, product managers, and testers each contribute unique expertise and insights. Similarly, in medical diagnostic contexts, specialized agents representing cardiologists, oncologists, and pediatricians collaborate, leveraging their distinct expertise to enhance diagnostic accuracy and patient care quality.

Notably, even when agents share common action spaces, such as producing documents or providing recommendations, their distinct personas critically influence task outcomes. As demonstrated by Hong et al. [755], a product manager persona might generate detailed requirement specifications, whereas a programmer persona would focus on software implementation. These persona-driven roles streamline interactions, leading to more robust and innovative collaborative outcomes, especially beneficial in multidisciplinary contexts.

Observation-space Heterogeneity. Observation-space heterogeneity highlights variations in the perceptual capabilities of agents, directly influencing their knowledge about the environment and other agents. Different agents may possess distinct views or limited access to information based on their assigned roles. This phenomenon frequently appears in interactive games like Werewolf and Avalon [1128, 1073, 1129]. In Werewolf, for instance, certain roles, such as werewolves and seers, have privileged observational abilities, enabling strategic information usage to deceive or collaborate effectively, while other roles like villagers must rely on limited public information. Similarly, in Avalon, role-specific perception significantly impacts communication and tactical decision-making, as agents strategically navigate varying degrees of information access.

In MAS applications, recognizing and effectively managing observation-space heterogeneity allows for nuanced communication strategies, efficient information dissemination, and improved coordination. Such deliberate utilization of differing observational abilities significantly enhances the complexity and realism of agent interactions, particularly in scenarios involving strategic deception, trust-building, and coalition formation.

Action-space Heterogeneity. Action-space heterogeneity reflects fundamental differences in the actions agents can perform, driven by their design constraints or functional specializations. This type of diversity is particularly prominent in scenarios involving physical or virtual constraints, affecting an agent's practical capabilities. For instance, in virtual game environments like Werewolf and Avalon, distinct roles possess unique abilities or actions, such as secret communication channels or specialized skills, that necessitate coordinated strategic decisions [1122, 1128, 1123, 1124].

Furthermore, robotic MAS scenarios exemplify action-space heterogeneity vividly. As described by Yu et al. [1154], robots within the same environment may possess divergent physical capabilities, such as mobility versus object manipulation. Effective team collaboration in these contexts relies on carefully aligning task assignments with individual agent strengths, promoting efficient task decomposition and specialized contributions. Thus, thoughtfully leveraging action-space heterogeneity significantly enhances overall task efficiency and adaptability in MAS.

14.1.3 Emergent Agent Specialization

Emergent specialization describes the phenomenon wherein initially homogeneous agents spontaneously develop distinct roles and specialized behaviors through repeated interactions and adaptation. In MAS incorporating LLMs, agent behaviors evolve organically due to inherent randomness, environmental feedback, and continuous adaptation cycles. This emergent differentiation leads to the gradual formation of specialized roles, significantly enriching the system's adaptive capabilities.

For instance, in an environment-driven MAS study by AL et al. [1155], agents initially endowed with identical action spaces and personas naturally differentiated into distinct specializations, as some agents gravitated towards resource gathering while others focused on crafting or defensive roles. Similarly, Takata et al. [1156] observed the spontaneous emergence of distinct communication styles, emotional expressions, and behavioral roles in initially homogeneous agent groups. These findings illustrate the transformative potential of MAS, transitioning from initially uniform teams toward increasingly heterogeneous systems characterized by organically evolved specializations. Such dynamic specialization underscores MAS's inherent flexibility and capacity for adaptation, enabling systems to respond effectively to evolving task demands and environmental complexities.

14.2 Multi-Agent System Topologies

 HIS section explores how the topology of interactions within LLM-based Multi-Agent Systems (MAS) influences their communication, collaboration efficiency, and task execution capabilities. We start by analyzing static topologies, characterized by fixed and predefined connection patterns, and subsequently delve into dynamic (adaptive) topologies, which can adjust inter-agent connections based on real-time factors such as performance feedback, workload variations, and strategic requirements. Finally, we discuss critical considerations regarding scalability, performance trade-offs, and robustness within MAS, drawing insights from recent advancements in distributed processing, self-organizing systems, and emergent collaborative behaviors.

14.2.1 Static Topologies

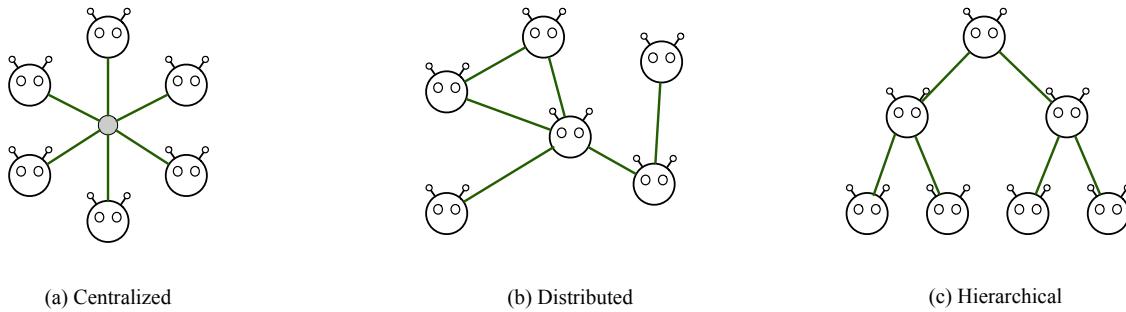


Figure 14.3: Different types of topological structures for multi-agent collaboration.

Static topologies utilize predetermined patterns of connectivity, remaining unchanged during runtime. In such configurations, the relationships and communication paths among agents, or between agents and a central coordinator, are established through explicit domain knowledge, rules, or heuristics. This rigidity ensures predictable information flow, simplifying coordination efforts. Three primary static configurations are typically recognized: *centralized*, *decentralized*, and *hierarchical* structures. Figure 14.3 illustrates these structures intuitively.

Centralized Structures In centralized structures, a singular coordinator agent collects information and orchestrates peripheral agent activities. This topology facilitates comprehensive resource management and offers a unified global perspective, exemplified by cultural simulations and collaborative scenarios outlined by Li et al. [1157] and Kaiya et al. [1158]. However, centralized approaches frequently encounter scalability constraints as additional agents are introduced, leading to increased communication overhead and heightened risk of single-point failures. Research by Xu et al. [1128] and Händler [1159] has further identified inherent trade-offs, highlighting that while centralized structures provide consistency, their rigidity limits adaptive responsiveness.

Decentralized Structures Decentralized structures involve peer-to-peer interactions among agents without reliance on a central controlling node, forming networks such as chains, rings, small-world configurations, or random graphs [1160, 1128]. Such designs inherently improve fault tolerance, as the failure of a single agent is less likely to disrupt the entire network. For instance, Hu et al. [1161] demonstrate how distributing graph reasoning tasks across multiple decentralized agents can effectively circumvent the context-length limitations typical in large language models. Additionally, Rasal and Hauer [1162] propose decomposition techniques enabling orchestrating LLMs to efficiently delegate subtasks. Despite these strengths, decentralized topologies must implement sophisticated consensus and synchronization protocols to maintain global coherence, which can introduce additional complexity and overhead.

Hierarchical Structures Hierarchical structures organize agents into layered arrangements, where higher-level agents manage and oversee lower-level ones, akin to classical organizational management models such as Standard Operating Procedures (SOPs) or Waterfall project methodologies. For example, Chen et al. [1146] describe the AutoAgents framework, assigning clear roles such as Planners, Agent Observers, and Plan Observers to systematically produce coherent execution plans. Similarly, the ChatDev framework [1140] utilizes hierarchical decomposition to effectively streamline complex software development processes [755, 1075, 756]. While hierarchical designs offer clear advantages, such as enhanced traceability, simplified debugging, and structured performance monitoring, they may also lead to bottlenecks if high-level agents become overloaded [1163]. Recent studies in diverse fields including narrative generation [1164, 1165, 1166], data science tasks such as data cleaning [1167, 1168], visualization [1169, 1170], and automated machine learning [1171, 1172] highlight an intrinsic tension between maintaining consistency and adapting to real-time challenges in hierarchical arrangements.

Overall, static topologies offer predictability, ease of implementation, and straightforward maintenance due to their clearly defined communication structures and predefined roles. Such designs are ideally suited for stable environments with fixed task workflows and clearly established requirements. Nonetheless, their primary drawback lies in the inability to dynamically adapt to unforeseen circumstances, such as agent failures, evolving task complexities, or shifting system objectives. This inflexibility often results in reduced effectiveness in scenarios characterized by emergent dynamics, necessitating more adaptive topological solutions.

14.2.2 Dynamic and Adaptive Topologies

Static topologies offer stability and predictability, making them suitable for tasks with clearly defined roles and consistent workflows. However, their rigidity limits their effectiveness in open-ended, dynamic, or novel

scenarios. Real-world domains, including collaborative planning and social simulations, frequently necessitate adaptability as resources fluctuate, feedback emerges, or objectives shift during execution. Dynamic and adaptive topologies address these demands by continuously restructuring inter-agent connections in response to real-time metrics, environmental feedback, or strategic objectives, achieving an optimal balance between reliability and responsiveness.

Dynamic topology frameworks, such as DyLAN [897], exemplify this adaptability by selecting agents dynamically based on performance metrics. DyLAN employs a two-stage optimization: initially computing unsupervised Agent Importance Scores and subsequently reforming agent teams at runtime. Similarly, OPTIMA [1173] refines agent connections iteratively through a generate–rank–select–train approach, optimizing communication pathways to balance task quality, token efficiency, and readability via Direct Preference optimization. The MAD framework [780] further illustrates adaptability by dynamically assigning agent roles (e.g., verifiers, debaters) within a systematically pruned interaction structure.

Recent advances in computational approaches have enhanced dynamic topology management. GPTSwarm [782] conceptualizes multi-agent collaboration as computation graphs, applying evolutionary strategies and reinforcement learning to optimize connections based on real-time feedback. MACNET [1174] employs a directed acyclic graph (DAG) architecture, where supervisory agents manage overall communication structure, and executive agents manage local tasks, thereby enabling adaptive communication through structured propagation of outputs. Application-specific systems like DAMCS [1175] combine hierarchical knowledge graphs with structured communication schemes, dynamically adapting agent coordination based on context. In creative applications, AutoAgents [1176] utilize a dynamic drafting-execution model, wherein predefined expert agents collaboratively refine task solutions through internal supervision and parallel processing.

Three primary methodological paradigms facilitate dynamic topology formation: search-based optimization, LLM-driven generative methods, and parameter-efficient external configurations.

Search-Based Methods Several frameworks utilize search-based optimization to iteratively improve inter-agent communication structures. ADAS [898] employs a Meta Agent Search algorithm to iteratively propose, evaluate, and archive superior agent configurations. Similarly, Aflow [876] leverages Monte Carlo Tree Search (MCTS) by representing each LLM interaction as a graph node, dynamically refining workflow structures. Approaches like MAD [1177] and OPTIMA [1173] incorporate iterative selection and training mechanisms, drawing from search optimization principles to balance efficiency and task performance.

LLM-Based Generative Methods Building upon search-based approaches, several studies exploit LLM generative capabilities to construct and continuously refine topologies. DyLAN [897] introduces a temporal feed-forward network model, using propagation methods to dynamically adjust agent teams. Similarly, frameworks like DAMCS [1175], AutoAgents [1176], and TDAG [1178] dynamically generate specialized agents or update hierarchical knowledge graphs, facilitating adaptive planning and task decomposition. Furthermore, AutoFlow [876] and Flow [1179] represent workflows using natural language programs or activity vertex graphs, continuously optimizing via reinforcement learning. ScoreFlow [900] further refines agent workflows through gradient-based optimization, enabling nuanced adaptation to evolving task demands.

External Parameter-Based Methods Recognizing the computational expense of directly tuning LLM-based agents, recent research promotes configuring agent interactions through lightweight external parameters. GPTSwarm [782] exemplifies this approach by training only the adjacency matrices in a DAG-based communication framework. Extending this concept, AgentPrune identifies redundant communication paths through magnitude-based pruning. Similarly, G-Safeguard [1180] employs graph neural networks (GNNs) external to the primary agent models, efficiently identifying and eliminating detrimental communication

channels. Despite their efficiency, such approaches may face performance constraints due to limited coupling with LLM-agent internal mechanisms.

Dynamic topologies also significantly advance social simulation and embodied AI domains by modeling complex real-world interactions and behaviors. Systems like OASIS [1092], ProjectSid [1155], and generative memory frameworks [35] dynamically adapt social network structures based on evolving interactions. ProjectSid's PIANO architecture enables large-scale, real-time simulations, demonstrating emergent societal phenomena such as specialized roles and cultural transmission within simulated communities. Architectures such as AgentScope-scalability [1181] facilitate extensive multi-agent simulations, capturing emergent group dynamics and collective decision-making processes. In medical contexts, AI hospital [1182] and Agent hospital [1075] dynamically reshape communication structures through iterative feedback cycles in clinical settings. Similarly, frameworks like IOA [1089] support dynamic team formation and adaptive task allocation across diverse embodied agents.

Despite their substantial progress, dynamic and adaptive multi-agent topologies face critical research challenges:

1. **Generalizability.** Current frameworks typically optimize agent interactions within single-task domains. Systems like AFlow [876] struggle to generalize beyond their initial design context. Advancing lifelong learning capabilities, enabling seamless adaptation across multiple domains with minimal resource expenditure, represents an essential future direction.
2. **Resource Efficiency.** Existing dynamic systems often entail considerable computational costs. For instance, ADAS [898] incurs substantial training expenses. Future research must develop more cost-effective runtime optimization techniques to facilitate widespread practical deployment.
3. **Inference Efficiency.** Highly complex multi-agent configurations frequently lack task adaptability, resulting in inefficient resource allocation during inference. While frameworks like MaAS [899] propose mechanisms for adaptive resource allocation, broader testing and refinement are necessary to ensure scalability and applicability in diverse real-world scenarios.

Addressing these challenges will significantly enhance the practical utility, scalability, and robustness of dynamic multi-agent systems, paving the way for broader, more adaptive, and efficient real-world deployments.

14.2.3 Scalability Considerations

Scalability remains a pivotal challenge in LLM-based multi-agent systems (MAS), particularly as the number of agents increases. In fully connected networks, the number of communication pathways grows quadratically, leading to what is often termed a *communication explosion*, which inflates both token usage and computational costs [1183, 755]. While centralized and layered topologies offer control and coordination, they often become synchronization bottlenecks under high message loads. On the other hand, decentralized networks enhance fault tolerance but demand complex consensus protocols to maintain a coherent global state.

Recent frameworks propose novel architectural strategies to address these bottlenecks. For instance, MACNET [1174] structures agent collaboration as a directed acyclic graph (DAG), enabling scalable coordination among up to 1,000 agents without significant performance degradation. Similarly, Hu et al. [1161] demonstrate that distributing graph reasoning tasks across multiple agents circumvents the limitations of input context length, thus unlocking broader scalability. Complementing these architectural insights, self-organizing agents [1184] demonstrate dynamic agent multiplication and adaptive task assignment, allowing each agent to maintain a constant workload while collectively increasing throughput.

Beyond theoretical advancements, scalable MAS frameworks increasingly depend on robust engineering practices. For example, AgentScope [1181] provides a developer-oriented platform that adopts an actor-based

distributed design. It supports seamless switching between local and distributed modes, incorporates automatic parallel optimization, and features fault-tolerant message routing with intelligent filtering. These capabilities collectively mitigate the synchronization burdens and communication overheads typical of large-scale deployments.

Project Sid [1155] takes a complementary route, focusing on simulating complex societal dynamics. Its PIANO (Parallel Information Aggregation via Neural Orchestration) architecture decouples slower cognitive modules from fast reactive layers, allowing agents to operate concurrently. A centralized cognitive controller ensures coherence across parallel streams of output. This architectural design enables high-frequency interactions among over 1,000 agents, while retaining control over coordination complexity.

At an even larger scale, AgentSociety [1185] demonstrates how LLM-based generative agents can be embedded within realistic social and economic simulations comprising up to 10,000 agents. This system employs high-performance messaging infrastructure (e.g., MQTT) and distributed compute to support millions of daily interactions. By modeling complex macro-level phenomena, such as market fluctuations, policy effects, and urban planning, it illustrates how scalability can enable richer simulations of emergent group behaviors.

Nevertheless, the benefits of scaling agent populations must be evaluated carefully. While increasing the number of agents can expand the system's overall cognitive and computational capacity, it also introduces diminishing returns due to escalating coordination costs. Beyond a certain threshold, added agents may contribute less to task-solving efficacy and more to synchronization burden, leading to a plateau or decline in performance. This scaling paradox stems from the super-linear growth of coordination overhead relative to the linear increase in individual task contributions.

Importantly, the value of scalability varies significantly by domain. In focused task-solving environments (e.g., coding, data analysis, or question answering), there likely exists an optimal population size beyond which performance gains stall. In contrast, simulation-driven domains, such as modeling collective decision-making, cultural evolution, or economic systems, demand high agent cardinality. Here, fidelity to real-world population-level behaviors often depends on emergent interactions among thousands of agents.

To address this duality, hybrid architectures that combine centralized oversight with decentralized execution are increasingly favored [1075, 1072]. Supervisory agents oversee global objectives and assign subtasks, while local agents operate semi-independently within dynamic sub-teams. This organization reduces information bottlenecks and adapts team sizes to task complexity, improving overall system responsiveness. In parallel, advanced techniques such as graph-based search, reinforcement learning, and evolutionary topology updates support continual adaptation of network structure. Intelligent communication techniques, including message prioritization, summarization, and selective broadcast, further mitigate overhead without sacrificing effectiveness. Moreover, asynchronous protocols and partial knowledge sharing schemes offer promising directions for reducing latency and enhancing robustness in massively distributed MAS deployments.

14.3 Collaboration Paradigms and Interactions

UILDING on the structural foundations laid out in the previous section on *Multi-Agent System Topologies*, we now shift our focus from how agents are connected to how they *collaborate*. While topology defines the architectural skeleton of MAS, it is the interaction dynamics that breathe life into the system, shaping how agents negotiate, reason, learn, and cooperate within these structures.

To better understand the diverse modes of collaboration in LLM-based MAS, we draw inspiration from sociological theories of human interaction, adapting them to model agent-agent behaviors. Specifically, we categorize inter-agent interactions into four paradigms: *consensus-oriented*, *collaborative learning*, *teach-*

ing/mentoring, and task-oriented collaboration. These paradigms are illustrated in Figure 14.4 and provide a finer-grained taxonomy that captures the complexity and fluidity of intelligent collaboration.

Each collaboration type serves different functional roles and exhibits distinct properties in terms of information flow, decision-making authority, goal alignment, and learning dynamics. Moreover, agents often engage in multiple interaction types concurrently, forming evolving networks of interdependent behavior. For instance, in collaborative software development systems [755, 756], a senior agent may guide junior developers through iterative dialogue (teaching), coordinate with peers on modular implementation (task collaboration), and participate in team-wide architectural planning (consensus building). Meanwhile, agents may also co-learn through testing and debugging phases, embodying collaborative learning. Understanding these paradigms and the mechanisms that support them, such as dialog management, role assignment, knowledge sharing, and trust modeling, provides crucial insight into the design of more effective and adaptive multi-agent systems.

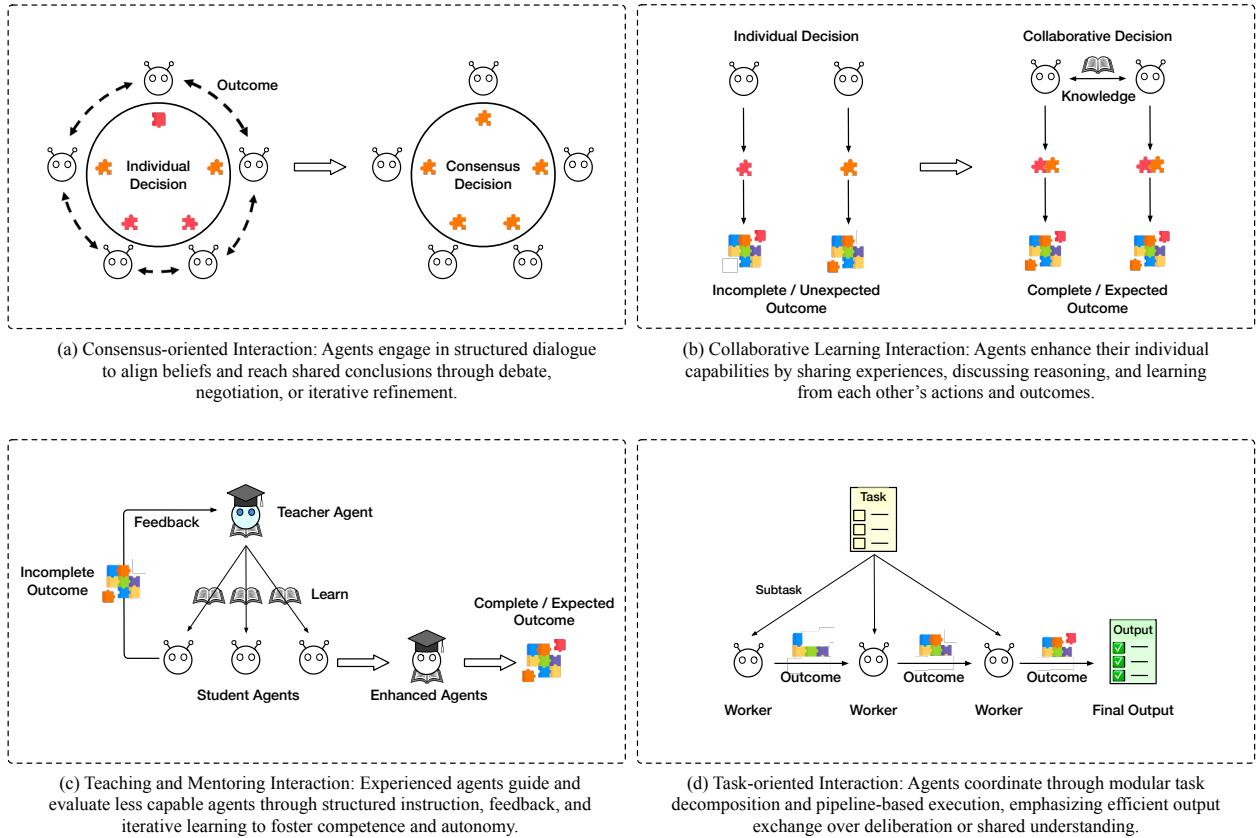


Figure 14.4: An overview of four agent-agent collaboration types in LLM-based MAS: *Consensus-oriented*, *Collaborative Learning*, *Teaching/Mentoring*, and *Task-oriented*.

14.3.1 Consensus-oriented Interaction

Consensus-oriented interaction focuses on aligning agent perspectives to arrive at a shared decision or judgment. It plays a pivotal role in situations that require integrating diverse knowledge sources and resolving disagreements through structured dialogue. This interaction mode is particularly valuable in complex problem-solving scenarios, where a unified understanding is essential for coherent and effective action [1186, 1187].

In such settings, agents engage in multi-round discussions, iterative refinements, and mutual critique to harmonize their beliefs or decisions. For example, systems like MedAgents [1076], MDAgents [1188], and AI Hospital [1182] demonstrate how multidisciplinary agents, ranging from doctors to radiologists, can collaboratively sharpen diagnostic reasoning by bringing their respective expertise into focused alignment. These dialogues frequently surpass zero-shot or few-shot reasoning by fostering grounded, high-fidelity conclusions derived from collective deliberation.

Scientific collaboration further illustrates the need for consensus. In *Agent Laboratory*[1025], PhD and postdoctoral agents iteratively refine research goals, interpret experimental data, and consolidate their findings through structured debate and negotiation. Likewise, *Virtual Lab*[1036] combines plenary discussions, where agents align on long-term scientific agendas, with specialized subteam interactions for specific subtasks, forming a two-tiered consensus structure.

Several mechanisms support consensus formation, including *discussion*, *debate*, *negotiation*, *reflection*, and *voting*. Each contributes distinctively: debates surface competing hypotheses; negotiations resolve resource conflicts or competing priorities; reflection supports iterative refinement based on new inputs; and voting allows distributed resolution when no consensus naturally emerges. These methods are often used in tandem to incrementally align agent beliefs.

Frameworks such as GPTSwarm [782] operationalize consensus through graph-based agent structures, where information flows along weighted edges shaped by trust or performance. Agents that consistently provide incorrect or low-quality outputs may be pruned from the consensus loop. RECONCILE [1072] uses structured round-table discussions, supplemented with confidence-weighted voting and reflective learning over past conversation rounds, to drive agreement.

Debate-based strategies have shown particular promise for reducing hallucinations and improving decision quality [1150, 1189, 1177, 1141]. In GOVSIM [1190], agents simulate negotiation around a shared resource, with dialogue that extends beyond task execution into relationship-building and long-term planning. The Multi-Agent Debate (MAD) framework [1177] encourages creative solution-finding through turn-based argumentation, moderated by a judge agent. Similarly, the FORD framework [1142] structures formal debates to steer consensus by allowing stronger models to influence weaker ones while preserving diversity of thought.

In collaborative refinement schemes like AutoAgents [1176], agents iteratively append and revise their own and each other's messages to arrive at convergent actions, reflecting a bottom-up process of constructing shared understanding. Across these systems, consensus is not merely a final state but an evolving process of alignment through deliberate, often structured, social interaction.

14.3.2 Collaborative Learning Interaction

Collaborative learning interaction emerges when agents, often architecturally similar, share experiences and insights to enhance one another's capabilities. While these agents may start from a common foundation, their interactions with varied environments lead to diverging memories, strategies, and behaviors. By working together, they pool their experiences to accelerate individual learning, improve skill acquisition, and adapt to complex tasks. Over time, this reciprocal exchange fosters the evolution of each agent's internal models and behaviors.

The distinction between collaborative learning and consensus-oriented interaction lies in their underlying intent. Whereas consensus aims to synthesize divergent views into a shared conclusion, collaborative learning emphasizes mutual enrichment. Agents are not trying to agree on a single outcome but instead aim to benefit from exposure to diverse strategies, perspectives, and feedback. This peer-driven process promotes individual growth and collective advancement without requiring uniformity of thought.

A central mechanism in this paradigm is observational and experiential learning. Agents adjust their own strategies after witnessing how others approach similar problems. They may integrate useful patterns without committing to the same decisions or beliefs [1118, 1119, 1120, 1128, 1122, 1124, 1129, 1125, 1126]. As Jin et al. [1123] highlight, the structure and quality of discussion significantly influence what and how agents learn from each other.

Three primary techniques structure collaborative learning interactions:

- **Experience sharing.** Agents communicate their insights and practical knowledge. Qian et al. [363] demonstrate that iterative refinement through shared team experience enables agents to improve adaptively in software development. MAS-CTC [361] extends this principle to multi-team collaboration: different agent teams exchange decision rationales and insights, pruning lower-quality content via aggregation mechanisms. In MOBA [1191], global agents reflect on the actions of local agents to improve real-time coordination with the environment. Similarly, AutoAgents [1176] integrate multi-scale memory sharing, including short-term, long-term, and dynamic memories, across agents to refine behaviors through collective feedback.
- **Peer discussion.** Agents engage in structured dialogues to explain, critique, and revise their reasoning processes. MEDCO [1077] creates a collaborative medical training environment where student agents enhance diagnostic reasoning through joint problem-solving. In Xu et al. [1192], agents conduct multi-round peer review by examining each other's reasoning steps, exchanging confidence scores, and refining outputs accordingly. These discussions help uncover diverse reasoning paths and support iterative accuracy improvements.
- **Observational learning.** Agents acquire knowledge by analyzing the actions and outcomes of others. AgentCourt [1193] trains lawyer agents to improve argumentation by learning from past courtroom debates. In iAgents [1188], agent communication mimics human social networks: agents actively exchange human-related knowledge and resolve information asymmetry using the InfoNav reasoning mechanism, organizing mixed memory to ensure relevance and informativeness. MARBLE [1105] combines cognitive expectation modeling with real-time feedback to continuously update agents' planning heuristics and decision quality.

Despite its promise, collaborative learning introduces notable challenges. Uneven capabilities among agents can lead to asymmetric learning opportunities; overreliance on peer insights risks amplifying noise or bias. Furthermore, maintaining diversity while allowing convergence requires careful management. Effective learning strategies must balance selective knowledge integration with robust individual reasoning. Fairness in information flow, scalability of coordination, and mechanisms for filtering useful contributions are all critical design considerations.

In dynamic and uncertain environments, the benefits of collaborative learning are significant. By enabling agents to iteratively learn from each other while preserving their autonomy, this interaction paradigm cultivates richer representations, more adaptive policies, and greater generalization, fueling both individual competence and collective intelligence.

14.3.3 Teaching and Mentoring Interaction

To complement collaborative and consensus-based paradigms, *teaching and mentoring interaction* introduces an asymmetric structure centered on intentional knowledge transfer. This paradigm plays a vital role in environments where skill acquisition and conceptual grounding are uneven across agents, enabling experienced agents to accelerate the learning trajectory of novices. Unlike collaborative learning, which emphasizes peer-level exchange, teaching focuses on directional, often hierarchical communication aimed at developing competence and autonomy in the learner.

In this context, teaching agents serve as guides, critics, and evaluators, shaping the learning experiences of mentee agents through structured feedback and domain expertise. Such interactions are particularly beneficial in complex multi-agent settings where not all participants start with the same capabilities or access to knowledge.

Core mechanisms in teaching and mentoring include:

- **Criticism and Feedback.** Mentors assess learners' outputs, highlight misconceptions, and offer corrective suggestions. This fosters a feedback loop through which learners incrementally refine their reasoning, decision-making, and skill execution.
- **Evaluation.** Teaching agents monitor learner progress via diagnostic tasks, scoring rubrics, or simulated environments. These evaluations provide developmental checkpoints that inform future guidance and allow adaptation to individual learner needs.
- **Instruction and Demonstration.** Mentors deliver explicit knowledge or procedural instruction, often supplemented by examples, allowing learners to observe, imitate, and clarify misunderstandings through guided interaction.

Iterative Development through Guided Practice Teaching is rarely a one-off transmission but typically unfolds through iterative cycles of task execution, observation, and feedback. In MEDCO [1077], student agents improve their clinical reasoning via repeated practice, structured by expert agents who evaluate patient interaction and diagnostic steps. These expert mentors not only assess performance but also intervene with contextual instruction tailored to learner gaps. Similarly, Li et al. [1075] show that an agentic doctor can steadily enhance diagnostic proficiency by repeatedly interacting with agentic patients in a simulated environment, effectively internalizing real-world medical knowledge through simulation.

Teaching interactions can take multiple forms depending on the flow of knowledge. *Unidirectional* teaching resembles traditional classroom instruction, where information is explicitly conveyed from mentor to learner, often through lectures or written guidance [1077]. In contrast, *interactive mentoring* integrates bidirectional dialogue, allowing learners to ask clarifying questions, propose hypotheses, or receive dynamic responses, transforming static instruction into an adaptive educational exchange.

From Supervision to Autonomy Over time, effective teaching transitions from direct supervision to increasing learner independence. As agents internalize feedback and demonstrate competence, mentors step back, allowing the mentees to operate with greater autonomy. This progressive withdrawal not only reflects trust in learner growth but also frees mentor agents to assist new learners, creating a virtuous cycle of scalable education within MAS.

By embedding teaching and mentoring mechanisms into multi-agent frameworks, systems can ensure more equitable skill distribution, accelerate onboarding of less experienced agents, and cultivate a robust foundation for collective intelligence. This paradigm, while rooted in structured guidance, ultimately empowers agents to become effective collaborators and future mentors themselves.

14.3.4 Task-oriented Interaction

Task-oriented interaction represents a pragmatic and execution-driven paradigm in MAS, where agents collaborate not through extended deliberation, but by aligning along structured workflows and interdependent tasks. The primary interaction mechanism centers on the exchange of intermediate results between agents according to a predefined task decomposition. Each agent focuses on a specific role in the pipeline, processing inputs from upstream and generating outputs for downstream agents. This form of collaboration emphasizes *coordination over conversation*.

Unlike consensus-driven or teaching-based paradigms, task-oriented interaction is characterized by strict modularity, temporal ordering, and minimal feedback loops. Agents do not necessarily share a common belief space, but instead rely on task dependencies, process interfaces, and clear deliverables to maintain coherence across the system.

Structured Pipelines in Software and Reasoning Tasks Recent frameworks illustrate several archetypes of this interaction pattern. In *software development*, systems such as MetaGPT [755] and ChatDev [756] orchestrate agents in roles that mirror the software engineering lifecycle: product managers analyze requirements, architects produce technical blueprints, developers implement features, and QA agents validate outputs. These pipelines are executed with minimal discussion across roles, relying instead on well-defined documents and task handoffs.

In *collaborative reasoning*, frameworks like Exchange-of-Thought (EoT)[1194], GPTSwarm[782], and MAC-NET [1174] design interaction topologies, such as rings, trees, DAGs, or learned graphs, that regulate the flow of information between agents. These topologies constrain expansion of context, optimize reasoning across multiple agents, and ensure only refined solutions are propagated forward. Similarly, Alpha-SQL [798] reformulates the Text-to-SQL task as a search problem where each node in the Monte Carlo Tree represents a partial solution step (e.g., selecting schema elements or refining queries), with agents responsible for specific subtasks in constructing the final SQL output. Building on this, EllieSQL [1195] further exemplifies task-oriented interaction by employing a complexity-aware routing framework that dynamically assigns Text-to-SQL queries to different SQL generation pipelines based on their estimated complexity, thereby optimizing cost-efficiency without sacrificing performance.

Workflow Automation in ML and Complex Domains Task-oriented structures are also prevalent in *machine learning applications*, such as AutoKaggle [1196] and AutoML team-based systems [1171], where agents specialize in dataset curation, model selection, training, and evaluation. These agents follow a disciplined execution plan with little need for mutual interpretation. TraveLER [1197], targeting VideoQA tasks, further extends this approach with a modular breakdown comprising Traverse, Locate, Evaluate, and Replan, coordinated by a Planner agent that iteratively refines strategies based on component feedback.

In many of these systems, agents function akin to a production line: interaction occurs through passing artifacts, not ideas. Graph-based designs like MACNET [1174] adopt DAG topologies where supervisory agents issue directives and executor agents solve subproblems in sequence. This structure enforces logical consistency and avoids context bloat by restricting memory propagation to essential outputs.

Adaptive Execution in Open-ended Environments Beyond structured domains, task-oriented MAS can also adapt to open-ended or dynamic environments. In Minecraft-style simulations [1081], leader agents decompose high-level objectives into granular subtasks, such as resource gathering, crafting, or defense, which executor agents complete in parallel. These systems emphasize flexible coordination through task assignment, communication protocols, and real-time adaptation to environmental changes.

Coordinated Efficiency over Deliberative Consensus The strength of task-oriented interaction lies in its operational efficiency. It avoids the cognitive and computational overhead of multi-turn dialogue, favoring clarity, parallelism, and modularity. Coordination strategies, such as synchronization points, shared memory buffers, and priority queues, ensure robust collaboration without requiring shared understanding.

These systems demonstrate that scalable agent collaboration does not always require consensus or mutual teaching. In many complex applications, particularly in engineering, automated research, and real-time systems, task decomposition and clear interfacing are sufficient for sophisticated multi-agent execution [759, 755].

14.4 Human-Agent Collaboration

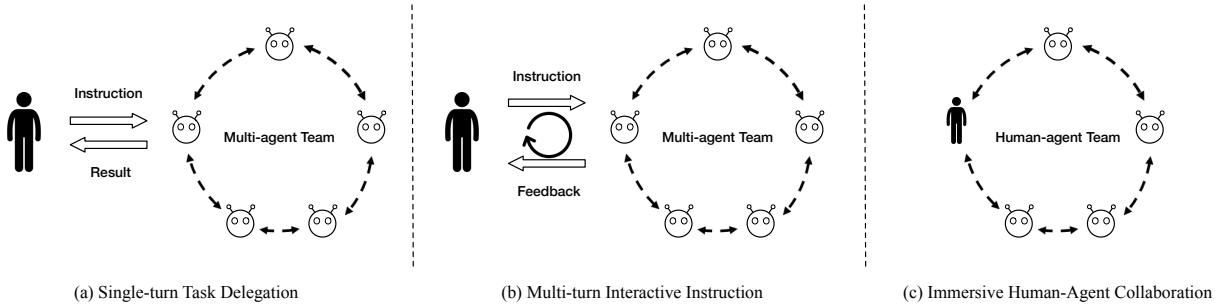


Figure 14.5: Three paradigms of human-agent collaboration. (a) In single-turn task delegation, humans issue standalone commands and receive full responses without further interaction. (b) Multi-turn interactive instruction introduces feedback loops, where humans supervise and refine outputs iteratively. (c) Immersive collaboration enables fluid, symmetric interaction between humans and agents, fostering real-time co-creation and shared decision-making.

 **W**HILE previous sections focus on how agents interact and collaborate among themselves, a complete MAS ecosystem must also consider the interface between agents and humans. As illustrated in Figure 14.5, human-agent collaboration forms a critical bridge for grounding AI systems in real-world use, enabling people to harness the capabilities of MAS to solve diverse problems. This collaboration generally falls into three paradigms: *single-turn task delegation*, *multi-turn interactive instruction*, and *immersive human-agent collaboration*.

Single-turn Task Delegation In the simplest form, humans delegate discrete tasks to MAS through a one-off command or request, known as *single-turn task delegation*. Examples include asking a Q&A system a factual question or assigning a standalone coding or generation task [1198, 755]. The MAS processes the request and autonomously returns a complete response without further interaction. This mode currently dominates how most users interact with LLM-based agents [1076, 756, 58]. Building on this baseline, more advanced systems allow users to *pre-configure* the agent composition and coordination strategies before execution [290], enabling customized orchestration and improving both performance and controllability.

Multi-turn Interactive Instruction Moving beyond static delegation, *multi-turn interactive instruction* introduces iterative feedback loops between humans and agents. In this mode, users refine requests, correct misunderstandings, and steer the agent toward satisfactory outcomes across multiple rounds of interaction. This dynamic is especially prevalent in creative and open-ended tasks, such as editing text, generating visual content, or drafting code [1094]. Here, humans typically assume a *supervisory role*, offering guidance, approval, and clarification throughout the process. For instance, users may iteratively instruct the agent to revise portions of a generated image, change design elements, or adjust the logic of a program. Systems that support real-time human feedback, such as through dialog, review checkpoints, or approval gating, have shown promise across domains like household robotics [1199], web automation [1084], software development [1200], and co-programming [1095].

Immersive Human-Agent Collaboration In the most integrated paradigm, *immersive collaboration*, LLM-based agents operate as peer collaborators rather than subordinates. Unlike interactive instruction, where humans guide the agent, immersive collaboration enables both parties to initiate actions, propose ideas, and make shared decisions. The interaction becomes symmetric and continuous, more akin to teamwork than command-following. Such agents may co-create content, join meetings as assistants, or participate in domestic tasks such as cooking or cleaning [1093, 1078, 1201]. By mirroring human-like initiative, situational

awareness, and dialog capabilities, these agents foster *fluid, real-time partnership* rather than sequential command chains.

Evaluating Human-Agent Collaboration To better understand and evaluate these modes of interaction, recent benchmarks such as Co-Gym [1202] offer structured environments that measure collaborative quality along key dimensions: communication clarity, situational alignment, personalization, and mutual efficiency. Tasks such as travel planning, writing related work sections, and data analysis have been used to benchmark different collaboration styles and quantify agent–human synergy.

As MAS capabilities mature, the scope of human-agent collaboration continues to broaden. From single-shot interactions for factual queries, to supervisory loops for design and coding, and all the way to rich joint action in everyday life, LLM-based agents are becoming more capable, autonomous, and interactive. A recent survey [1203] offers a comprehensive overview of these trends and challenges. Looking forward, these agents are poised to deeply integrate into daily workflows, augmenting human productivity across domains. In turn, human users will need to adapt their communication strategies and expectations to engage effectively with intelligent agents. This co-evolution of collaboration practices and agent capabilities will reshape not only task completion, but also the broader fabric of human labor, creativity, and cooperation in the LLM era.

14.5 Decision-Making in Multi-Agent Systems

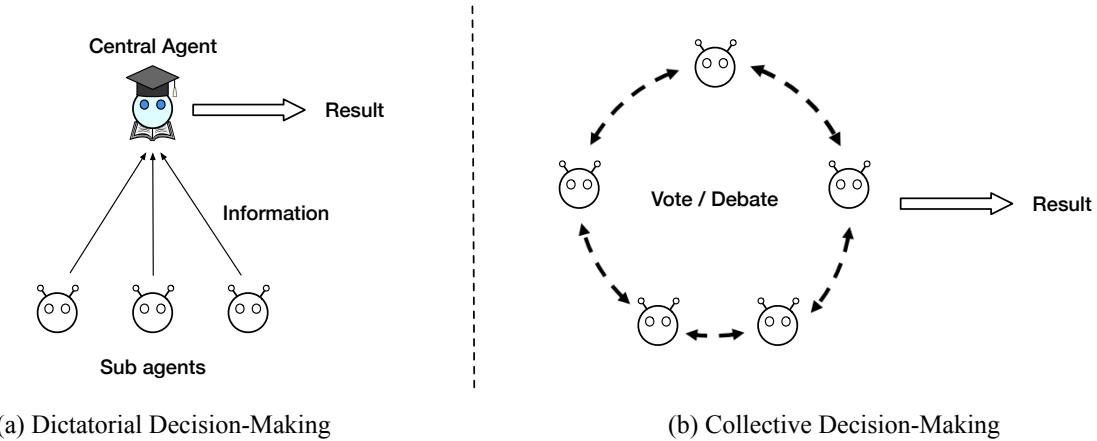


Figure 14.6: Two principal paradigms of decision-making in multi-agent systems. (a) *Dictatorial decision-making* concentrates authority in a central agent, who synthesizes information from sub-agents to generate final decisions. (b) *Collective decision-making* distributes authority among agents, who vote or debate to reach consensus through local interactions. Each paradigm reflects distinct trade-offs in efficiency, transparency, and robustness.

FOLLOWING our discussion on collaboration paradigms, we now examine how agents within a MAS make decisions, a fundamental capability that underpins their ability to act cohesively, adapt to complex tasks, and maintain system-level efficiency. As illustrated in Figure 14.6, decision-making architectures play a crucial role in shaping how agent interactions yield coherent outcomes. While collaboration is the medium through which agents interact, *decision-making* governs how these interactions lead to final actions and results. Recent work has emphasized that the choice of decision-making strategy significantly influences the emergent behavior and overall effectiveness of MAS. For instance, [1183] demonstrates that different decision architectures can alter the efficiency and fault tolerance of collaborative processes, while [780] highlights that structured decision-making mechanisms can stimulate system-wide emergent intelligence.

Broadly, collaborative decision-making in MAS can be categorized into two architectural paradigms: **dictatorial decision-making** and **collective decision-making** [1183]. These paradigms differ in how authority is distributed, how information is processed, and how final decisions are reached.

14.5.1 Dictatorial Decision-Making

Dictatorial decision-making centralizes authority in a single agent (or node) within the system. Other agents contribute information, such as local observations or intermediate results, which the designated decision-maker synthesizes to form a global decision. This model enables top-down reasoning with a unified perspective, allowing for efficient coordination and policy consistency.

Several LLM-based MAS architectures follow this pattern. For example, [1177, 1204, 1188] employ a single LLM to aggregate diverse agent outputs and formulate a decision that reflects a more objective synthesis of viewpoints. Similarly, [181, 1205] explore decision fusion via ranking, scoring, and checklist-based methods to enhance decision robustness.

Beyond final aggregation, dictatorial structures also appear in hierarchical workflows. Central agents in [1176, 1206] decompose complex problems into sub-tasks and dispatch them to specialized agents. These agents execute narrowly scoped responsibilities, after which the central coordinator reassembles the results.

In distributed variants of this model [782, 1174], the decision-maker may not be explicitly defined. Instead, the last node in a structured topology (e.g., a DAG or pipeline) implicitly serves as the concluding agent. It processes accumulated knowledge and outputs a system-level judgment, functioning as a distributed, yet still centralized, decision point.

14.5.2 Collective Decision-Making

In contrast, collective decision-making distributes decision authority among agents, enabling them to jointly determine outcomes through local reasoning, negotiation, or voting. This bottom-up paradigm fosters robustness, adaptability, and resilience in dynamic or partially observable environments.

Voting-Based Decision Making Voting is a foundational mechanism in collective decision-making, offering a lightweight and interpretable means for reaching consensus. Systems such as [1187, 1125] adopt majority or plurality voting among agents to finalize answers or select action plans. The GEDI electoral module [1183] expands this further, incorporating multiple voting strategies to improve collective reasoning while avoiding complex arbitration logic. These methods enhance fault tolerance and preserve simplicity in agent interactions.

Debate-Based Decision Making Debate-based mechanisms emphasize interactive deliberation among agents to resolve uncertainty or conflicting perspectives. In [1177, 1207], agents participate in structured discussions where they present arguments, critique one another's proposals, and refine shared solutions through back-and-forth dialogue. More nuanced protocols [1192, 1208] focus on iterative consensus-building, where agents repeatedly exchange updates to reconcile disagreements. To mitigate the "cognitive island" problem, where agents lack shared knowledge, systems such as Apollo [1143] incorporate a common retrieval knowledge base that aligns agent understanding and grounds their reasoning. These systems emulate human-style conversations to enable distributed agents to converge on more informed, justified decisions.

In summary, decision-making in MAS spans a spectrum, from centralized, top-down control to distributed, emergent consensus. The choice of decision architecture reflects trade-offs between efficiency, robustness, transparency, and flexibility. As MAS become more autonomous and general-purpose, hybrid architectures that blend both paradigms are increasingly explored, seeking to combine the strengths of centralized oversight with the adaptability of distributed reasoning.

14.6 Communication Protocols in Multi-Agent Systems

 As the capabilities and complexity of multi-agent systems (MAS) grow, effective communication becomes the foundation for coordination, reasoning, and collaboration across agents. Building upon the decision-making mechanisms discussed earlier, this section turns to the design of communication protocols that enable agents to share knowledge, delegate tasks, and interact coherently with each other, with users, and with external environments.

We begin by categorizing typical message types, ranging from queries and commands to status updates and feedback, clarifying the modes and semantics of agent interaction. Next, we examine the distinct communication interfaces across different interaction channels: agent–environment, agent–agent, and agent–human. Each interface presents unique architectural and semantic challenges, and we explore emerging design patterns that support transparency, modularity, and robustness in these interactions. A special focus is placed on the architectural underpinnings of communication protocols, including serialization formats, messaging layers, and routing mechanisms. Standardized interfaces and protocol specifications are essential for achieving interoperability, reusability, and scalability in MAS, especially as systems increasingly integrate heterogeneous agents, tools, and environments. We conclude the section with a unifying perspective on communication in LLM-based systems, highlighting shared principles across agent–environment and agent–user communication, and advocating for coherent, modular designs that enable fluid interaction across diverse applications. This unified protocol layer is especially critical for ensuring that large-scale, general-purpose agent systems remain consistent, extensible, and grounded in interpretable interaction logic.

14.6.1 Types of Agent Messages

Effective communication within Multi-Agent Systems (MAS) hinges on the nature and structure of exchanged messages. These messages can broadly be categorized into two types: **structured** and **unstructured**. Each serves distinct purposes and offers unique advantages depending on the application context.

Structured Messages Structured messages, typically encoded in JSON [1209, 1210], XML [1211, 765], or programming code [755, 756, 1147], is a core component of LLM-based multi-agent communication. Their predefined syntax and semantics allow for unambiguous interpretation, easy parsing, and dependable information exchange. These formats enable agents to encode task parameters, directives, or even executable routines in a machine-readable and verifiable form, streamlining workflow automation and minimizing interpretive errors.

Structured messages are particularly well-suited to deterministic, high-efficiency tasks such as sub-task decomposition, workflow orchestration, and coordination across agents in hierarchical or pipeline architectures. Their consistency supports system-level optimization, persistent memory logging, and retrospective analysis. In such settings, the explicitness of structure enables efficient retrieval, storage, and validation of data throughout multi-agent interactions.

Unstructured Messages In contrast, unstructured messages, such as natural language [1128, 1127, 1073], images, video, and audio signals [1212, 868], convey richer, more context-sensitive information. These modalities capture nuanced cues: visual content expresses spatial and emotional context; audio signals transmit not just spoken language but also tone and affect; and text allows for abstract reasoning, negotiation, or goal specification with flexibility beyond rigid formats.

Unstructured messages are vital in open-ended or ambiguous settings where communication must accommodate uncertainty, creativity, or evolving environments. While their complexity historically posed a challenge for automated interpretation, recent progress in LLMs and multimodal models [657, 635, 1213] has made such communication increasingly tractable, enabling agents to understand, generate, and ground unstructured content with remarkable fluency.

Hybrid Messaging for Adaptive Communication In practice, structured and unstructured messages often complement one another. Structured messages provide precision, reliability, and computational efficiency, making them ideal for operational control and deterministic execution. Unstructured messages, by contrast, support adaptive, expressive, and context-rich exchanges, crucial for creative collaboration, negotiation, and real-world interaction. Together, these modalities form a hybrid communication foundation that enables intelligent, flexible cooperation among agents in diverse MAS applications.

14.6.2 Communication Interfaces

The effectiveness of a multi-agent system (MAS) hinges not only on the quality of the messages exchanged but also on the design of its communication interfaces. These interfaces govern how agents interact with their surrounding environments, with other agents, and with human users. Ensuring clarity, modularity, and interoperability across these interfaces is essential for robust, generalizable, and user-aligned MAS architectures.

Agent–Environment Interfaces LLM-based agents often need to interact with external environments to accomplish tasks, such as clicking a UI button, issuing a web request, or controlling an embodied avatar in a simulation. From the agent’s perspective, each action represents a desired effect on the environment. However, environments vary widely in the actions they support. Thus, to ensure meaningful interaction, agents must first infer or query the affordances of the environment and adapt their behavior accordingly.

Once an action is issued, the environment typically returns either a successful observation or an error signal. The agent must interpret this feedback to adjust future decisions. Environments may include operating systems, web interfaces, virtual worlds, simulations, or real-time robotics platforms. To standardize these interactions, several frameworks have been developed to unify the agent–environment interface, enabling agents pretrained via LLMs to generalize across diverse execution contexts with minimal adaptation [840]. These frameworks foster benchmarking, skill transfer, and modular deployment.

Agent–Agent Interfaces Communication between agents forms the backbone of any MAS. In LLM-based systems, natural language is the predominant mode of communication, largely due to LLMs’ pretraining on extensive linguistic corpora. Text-based communication is particularly popular, enabling agents to discuss, negotiate, critique, and persuade one another using expressive and flexible dialogue [1076, 1078, 1153, 1127, 1214]. In more embodied or multimodal scenarios, voice-based interactions also emerge as viable channels [868, 1215, 1216].

In contrast, some systems opt for structured communication formats to reduce ambiguity and parsing complexity. These formats enable more efficient and deterministic agent coordination [755]. For example, systems like TaskWeaver [1083] adopt structured message schemas with clearly defined fields such as `sender`, `receiver`, `message type`, and `content`, along with explicit parsing instructions. Such modular protocols facilitate scalability and robustness in high-volume, high-stakes agent interactions.

Human–Agent Interfaces Ultimately, multi-agent systems are built to augment human cognition, extend agency, and improve societal outcomes. While some MAS frameworks focus on human-as-observer settings, e.g., in social simulations [35, 1217]. Many systems support active human participation. In these settings, communication may occur via either natural language or structured formats [1078, 1084].

Natural language remains the most intuitive medium for human interaction. To bridge the gap between human expressiveness and agent system requirements, a central LLM component often serves as a translator, parsing user language into structured actions or queries that agents can interpret and execute. This LLM mediator may be embedded within the MAS or deployed as an external interface layer.

For more efficient, programmatic control, some systems also enable humans to interact directly via structured messages, following predefined schemas or APIs. This approach is particularly useful for technical

users or scripted workflows [1085]. By aligning human input with the internal logic of the MAS, such interfaces can streamline interaction while maintaining high-level control and traceability.

14.6.3 Next-Generation Communication Protocols

The field of LLM-based multi-agent systems is still in its early stages. Current agent architectures and communication mechanisms are often designed in an ad hoc manner for specific domains or tasks, including agent–environment, agent–human, and inter-agent interactions. However, the lack of a unified communication framework leads to fragmented and siloed ecosystems: tools, agents, environments, and data sources frequently operate in isolation, impeding composability and cross-system collaboration. Moreover, most existing protocols are manually crafted, placing a high design burden on developers and often lacking semantic flexibility, adaptability, or scalability.

To address these limitations, several new communication protocols have recently been proposed, each targeting distinct layers of the protocol stack, from transport and identity to semantic negotiation and tool invocation.

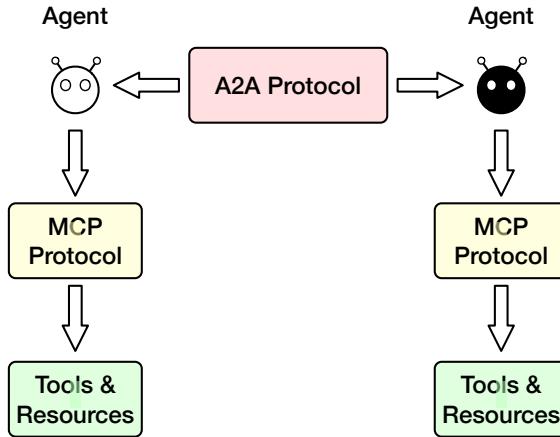


Figure 14.7: Illustration of the two foundational communication protocols in multi-agent systems. The *MCP (Model Context Protocol)* governs how each agent interfaces with tools and external resources. The *A2A (Agent-to-Agent) Protocol* defines the communication channel between agents, supporting coordination, negotiation, and collaboration. Together, these protocols establish a modular and interoperable framework for scalable and composable agent systems.

Internet of Agents (IoA) Chen et al. [1089] proposes an internet-inspired, instant-messaging-style architecture for agent communication, enabling dynamic team formation and task-driven collaboration. Agents register with a centralized coordination server that manages identity, discovery, and message routing. Communication is orchestrated using finite state machine (FSM)-based dialogue templates, allowing flexible structuring of discussion, delegation, and triggering messages. IoA supports features such as speaker turns, nested group dialogues, and message-length constraints, enabling agents to adapt message formats according to different coordination phases. While flexible within a fixed schema, it remains reliant on a central orchestrator.

Model Context Protocol (MCP) Anthropic [1085] introduces a client-server architecture aimed at simplifying tool access and structured data retrieval for LLM agents. MCP decouples tool invocation from LLM-generated text, improving privacy, stability, and interoperability. Tools are registered through a standardized interface, enabling cross-model reuse and compositional workflows. Its rigid and minimalistic design makes MCP suitable for high-assurance, tool-centric workflows. However, it lacks support for dynamic protocol negotiation, semantic adaptation, or flexible extension, which limits its applicability in decentralized or adaptive agent ecosystems.

Agent Network Protocol (ANP) Chang [1086] proposes a decentralized protocol inspired by Web3 standards. Agents identify themselves using W3C-compliant decentralized identifiers (DIDs) and communicate via encrypted peer-to-peer channels. ANP includes a meta-protocol layer that enables agents to negotiate which application-layer protocol to use, supporting multi-protocol environments (e.g., HTTP, JSON-RPC, natural language) and allowing semantic protocol selection based on agent capabilities. While it prioritizes extensibility and decentralization, ANP currently lacks explicit support for public protocol reuse or standardized capability registries.

Agora Marro et al. [1088] presents a highly flexible, language-native protocol system. Rather than relying on pre-registered APIs, agents exchange Protocol Descriptions (PDs), which are free-text descriptions of communication semantics. LLMs interpret and execute PDs dynamically at runtime, enabling agents to create, deploy, and adopt new protocols entirely through language. Agora operates without centralized registries, leveraging distributed repositories for protocol sharing and reuse. This supports continual learning and interoperability, though it introduces potential ambiguity and verification challenges due to its open-ended nature.

Agent2Agent (A2A) Surapaneni et al. [1087] defines an open, web-native communication protocol that enables autonomous agents to discover, negotiate, and collaborate on long-running tasks without disclosing internal memory or implementation details. Built on HTTPS transport and JSON-RPC 2.0, A2A introduces machine-readable Agent Cards that advertise capabilities, endpoints, and authentication mechanisms. Agents form asynchronous Tasks governed by lifecycle hooks, observability metadata, and policy controls. Positioned above resource-level protocols like MCP, A2A enables secure, agent-level orchestration while delegating tool invocation to subordinate protocols, fostering modularity and separation of concerns. Figure 14.7 illustrates the relationship between A2A and MCP protocol.

Agntcy Foundation [1218] envisions a federated, internet-scale platform for AI agents, grounded in four design pillars: interoperability, security, scalability, and standardization. At its core, Agntcy integrates a collision-resistant identity layer with an Open Agentic Schema Framework (OASF) and a distributed Agent Directory for discovery and capability alignment. On this foundation, the Agent Connect Protocol (ACP) provides a lightweight invocation interface, while the Agent Gateway manages transport-level security. Agntcy promotes vendor-agnostic, large-scale collaboration across diverse agent ecosystems, offering a unified, open substrate for the Internet of Agents.

14.6.4 Protocol Comparison and Future Directions

As shown in Table 14.1, next-generation agent communication protocols differ along key dimensions such as identity and security mechanisms (e.g., OAuth, DID, VC), meta-protocol negotiation capabilities (e.g., only ANP and Agora support it), data exchange formats (e.g., JSON-RPC, gRPC, JSON-LD), the degree of centralization, and application-layer flexibility. A unified, secure, scalable, and dynamic protocol infrastructure, where agents can negotiate and co-create protocols in real time, is critical for enabling large-scale, interoperable agent ecosystems. While current frameworks such as MCP, ANP, Agora, IoA, A2A, and Agntcy represent early but promising steps, protocol design remains a rapidly evolving frontier. Key challenges ahead include improving semantic interoperability, supporting collective coordination, and ensuring adaptability in open and heterogeneous environments. As highlighted in recent work [1219, 1220], future protocols should emphasize modularity, privacy preservation, and model-native protocol understanding, moving away from rigid, static formats toward infrastructures that allow agents to dynamically compose, adapt, and evolve their communication protocols over time.

Table 14.1: Concise comparison of six agent communication protocols across key layers. **PD** = Protocol Description; **DID** = Decentralized Identifier; **VC** = Verifiable Credential; **NL** = Natural Language; **AMP** = Agent Message Protocol; The **Meta-Protocol** denotes a mechanism that lets agents *negotiate or co-create their own protocol*.

Layer	MCP	ANP	Agora	IoA	A2A	Agntcy
Identity & Security	OAuth	DID	-	Agent registry rules	OAuth, API Keys	VC
Meta-Protocol	-	✓	✓	-	-	-
Data Exchange	JSON-RPC	JSON-LD	PD / NL	AMP	JSON-RPC	gRPC
Centralization	Centralized	Decentralized	Decentralized	Centralized	Decentralized	Hybrid
Flexibility	Rigid	Flexible	Highly flexible	Moderate	Moderate	Moderate
Publisher	Anthropic	ANP Community	Eigent AI	Tsinghua University	Google	Langchain, Cisco

14.7 Summary and Discussion

HIS chapter presented a systematic overview of the foundational concepts and critical design choices involved in constructing collaborative multi-agent systems (MAS) based on large language models (LLMs). We first explored strategies for composing agent teams, differentiating between homogeneous agents, which is beneficial for simpler tasks demanding parallelism, and heterogeneous agents, which excel in complex scenarios by bringing diverse expertise, observation capabilities, and action spaces to the collective problem-solving process. Importantly, we highlighted emergent agent specialization, showcasing how homogeneous agents can evolve diverse capabilities through continuous interaction, fostering increased adaptability.

We then addressed multi-agent system topologies, delineating between static and dynamic structures. Static topologies, well known as hierarchical, decentralized, and centralized, offer advantages in simplicity, predictability, and ease of management but struggle with real-time adaptability. Conversely, dynamic and adaptive topologies leverage flexible reconfiguration strategies, significantly enhancing system responsiveness and scalability. Scalability emerged as a critical consideration, underscoring the necessity of balancing coordination overhead against task complexity, especially as agent populations grow. Frameworks such as AgentScope, Project Sid, and AgentSociety exemplified effective solutions to scalability challenges by integrating decentralized structures, parallel processing, and high-performance messaging systems.

The chapter further explored four key paradigms of agent collaboration: consensus-oriented, collaborative learning, teaching and mentoring, and task-oriented interactions. Each interaction type uniquely shapes agent behaviors and collaborative outcomes, highlighting the importance of tailoring interaction protocols to specific system goals. Additionally, we emphasized human-agent collaboration, categorizing interaction modes into task delegation, iterative instruction, and immersive partnerships. These frameworks illustrated varying levels of human involvement and underscored the potential of LLM-based agents to augment human productivity and creativity through nuanced collaboration.

Decision-making mechanisms were identified as another crucial design dimension. Centralized (dictatorial) decision-making was noted for its coherence and simplicity, whereas collective decision-making methods such as voting and debating, provide greater resilience and adaptability, particularly in uncertain or dynamically changing environments. The choice of decision-making strategy significantly impacts MAS performance, emphasizing the need for thoughtful selection based on context-specific requirements.

Lastly, we reviewed contemporary agent communication protocols, including MCP, ANP, Agora, IoA, A2A, and Agntcy. These emerging frameworks address critical issues of identity, security, interoperability, and dynamic protocol negotiation. While current protocols demonstrate substantial progress, we identified ongoing challenges related to semantic interoperability, scalability, and adaptability, highlighting promising directions for future research.

As MAS continues to evolve, several research avenues stand out:

- **Context-Aware Collaboration:** Developing advanced frameworks enabling agents to dynamically select optimal interaction strategies based on real-time context and system conditions.
- **Scalable Adaptive Architectures:** Designing hybrid topologies that seamlessly combine centralized coordination with decentralized execution, optimizing scalability and responsiveness.
- **Human-Agent Co-evolution:** Enhancing immersive human-agent interactions through continuous learning mechanisms, enabling agents to better adapt to human preferences and workflows.
- **Adaptive Decision-Making:** Investigating methods that allow systems to dynamically transition between centralized and collective decision-making models based on the current environment and task complexity.
- **Dynamic Communication Protocols:** Exploring meta-protocol frameworks that enable agents to autonomously negotiate, compose, and evolve communication standards on-the-fly, significantly enhancing interoperability and semantic flexibility.

In conclusion, the thoughtful integration of agent composition, topological structures, interaction paradigms, decision-making frameworks, and robust communication protocols will be pivotal for developing effective, scalable, and adaptive MAS capable of tackling increasingly sophisticated real-world challenges.

Chapter 15

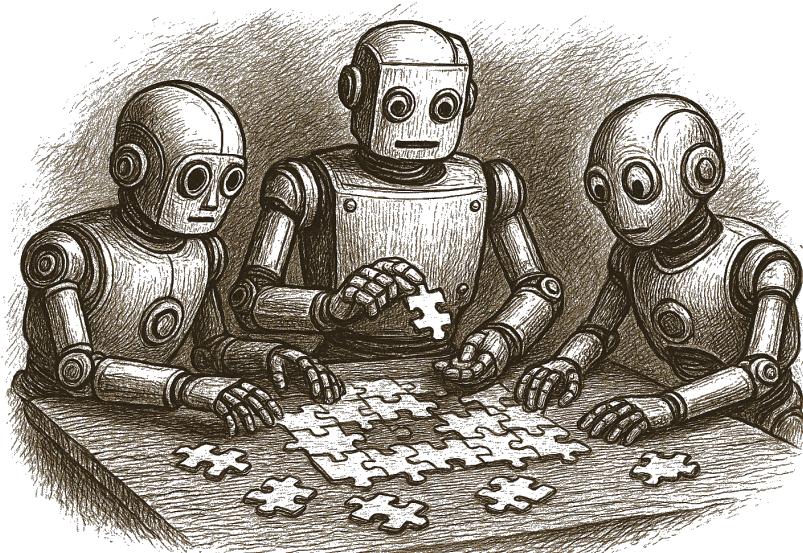
Collective Intelligence and Adaptation in Multi-Agent Systems

HIS chapter explores the emergence and dynamics of *collective intelligence* in multi-agent systems (MAS), emphasizing both the system-level capacity to solve complex tasks and the individual-level evolution of intelligent agents within social environments. Drawing from cognitive science, philosophy, and computational theory, we investigate how agent collectives, composed of Large Language Model (LLM)-based agents, achieve emergent behaviors, adaptive coordination, and self-organizing hierarchies beyond the capabilities of any single agent. We examine the structural mechanisms driving collective intelligence, including topology evolution, routing optimization, and memory-driven coordination, and how these mechanisms enable agents to engage in high-level social behaviors such as trust, deception, and cultural norm formation. At the same time, we analyze how shared memory, communication protocols, and reflective learning empower individual agents to adapt, specialize, and co-evolve within the broader agent society. Through theoretical insights and empirical case studies, this chapter positions collective intelligence not merely as a performance gain but as a foundational paradigm for understanding cognition, cooperation, and social emergence in artificial societies.

15.1 From Collective Intelligence to Individual Evolution

ULTI-AGENT SYSTEMS (MAS) derive inspiration from biological swarms and human societies, where group cooperation often yields capabilities far beyond those of any individual. A central idea is that of *collective intelligence*, the phenomenon where a community of agents collectively demonstrates problem-solving and decision-making prowess exceeding that of the single best agent, as shown in Figure 15.1. This concept is famously captured by the “Wisdom of Crowds” [1069], which asserts that under the right conditions, the many are smarter than the few. For instance, independent individuals can pool their diverse knowledge such that the aggregated decision outperforms any expert’s judgment.

Minsky’s *Society of Mind* theory [44] further hypothesizes that what we call intelligence may itself emerge from the interplay of numerous simpler components (“agents”) in the mind. By analogy, a MAS can exhibit intelligent behavior as an emergent property of interactions among relatively simple agents. Recent cognitive studies on *Theory of Mind* [1070, 1071] suggest that even artificial agents can start to reason about each other’s mental states—a capability crucial for sophisticated social reasoning. Taken together, these insights paint a picture in which intelligence springs from synergy: like neurons forming a brain or ants forming a colony, coordinated agents can create an intelligent whole greater than the sum of its parts.



Collective Intelligence: through distributed effort and coordination, agents can solve problems none could tackle alone.

Figure 15.1: Collective intelligence emerges when multiple agents coordinate, complement, and refine each other's reasoning to solve problems beyond individual reach.

In human societies, *collective intelligence* arises naturally as people collaborate, divide labor, and solve problems together. Likewise, MAS can harness distributed knowledge and complementary skills of specialized agents to tackle complex tasks beyond any single agent's reach [1068]. Importantly, collective intelligence is not simply the sum of individual capabilities, but an emergent phenomenon, novel capabilities and patterns of behavior materialize from interactions. Through ongoing communication and feedback, the group develops shared understanding and collective memory, enabling more coherent and effective coordination over time.

Notably, heterogeneity among agents, which includes differences in knowledge, perspective, or strategy, often strengthens the collective: diversity helps the group avoid the pitfalls of individual bias or "groupthink", leading to more robust and unbiased solutions [1068]. In essence, a well-designed agent society can tap into a *wisdom of crowds* effect, where each agent's unique insight contributes to a better global outcome.

While the group evolves its own intelligence, the individual agents within it are not static. Each agent can learn and adapt through experience, a concept we term *individual adaptability*. In an MAS, an agent's adaptability refers to its capacity to adjust behavior, strategies, or goals based on past interactions and feedback from the environment. This continual refinement is akin to how a person learns from life experiences or how animals adapt to new conditions.

Recent research even describes agents as *self-evolving*, which is able to modify their objectives or update their internal models on the fly in response to feedback [1068]. The advent of large language models (LLMs) as agent brains has accelerated this process: LLM-powered agents come equipped with powerful general knowledge and can leverage dynamic memory modules to reflect on interactions, monitor their own performance, and self-correct. By exchanging information with others and storing rich histories of interactions, agents can improve over time without requiring an external programmer to intervene at each step.

We can categorize an agent's learning mechanisms into *memory-based adaptation* (which does not alter the agent's underlying model parameters, instead relying on stored experience and on-the-fly reasoning) and

parameter-based learning (which involves updating the agent’s model or policy through training or fine-tuning). These two forms of adaptation, roughly analogous to learning within a lifetime versus evolving across generations, work in tandem to make individual agents progressively more competent as they participate in an agent society.

In the sections that follow, we delve deeper into how *collective intelligence* emerges in MAS and how it can be enhanced, and then explore how living in a society of agents drives the adaptation and evolution of individual agents. We will see that the growth of group intelligence and individual learning are deeply intertwined, yielding a fascinating co-evolutionary dynamic: the group makes the individuals smarter, and smarter individuals, in turn, further improve the group.

15.2 Collective Intelligence in Multi-Agent Systems



In this section, we clarify what “collective intelligence” means for multi-agent systems, tracing how higher-order capabilities emerge from local interactions among diverse, communicating agents. We first ground the concept theoretically (e.g., incentive alignment and COIN-style reward design), then show empirically how collaboration boosts performance and robustness. We next distill practical strategies, topology evolution, coordination/routing optimization, and shared memory, to deliberately foster such emergence. Finally, we examine the social layer: norms, deception, roles, and culture that spontaneously arise as agents co-adapt over time.

15.2.1 Concept and Emergence of Collective Intelligence

Collective intelligence in MAS refers to the capacity of a group of agents to solve problems or make decisions collaboratively with an efficacy that surpasses that of any single agent [1132]. This concept implies emergent behavior: through interaction, the group exhibits new capabilities and higher-order reasoning that were not explicitly programmed into the individual agents. A classic example is a flock of birds or a school of fish executing complex coordinated maneuvers without any bird or fish explicitly “in charge”. In the context of artificial agents, we similarly observe that simple local rules or interactions can yield sophisticated global behavior. For instance, agents can form consensus decisions, perform division of labor, or collectively explore a solution space in ways that no lone agent could. The key ingredients for such emergence often include decentralized control, diversity among agents, and mechanisms for communication and feedback. When each agent contributes independent information or perspective, and there is a means to aggregate these contributions, the resulting group decision tends to be more informed and accurate than one made in isolation. Surowiecki (2005) identified factors like diversity of opinions, independence of members, and a reliable method of aggregation as critical to wise crowds – principles that likewise guide the design of intelligent MAS [1071].

A theoretical foundation for understanding emergent collective intelligence is provided by Wolpert and Tumer’s Collective INtelligence (COIN) framework, which considers a large MAS with no central controller and a global utility function measuring overall system quality. The challenge is to design local reward functions so that, by maximizing their own rewards, agents still improve the global utility, avoiding tragedies of the commons through incentive alignment. Appropriate reward shaping has been shown to yield near-optimal collective behavior in domains such as network routing and resource allocation, underscoring how coordination mechanisms (reward design, norms, protocols) transform a mere collection of agents into a coherently intelligent unit [1132].

Notably, recent work has demonstrated that LLM-based agents possess ingredients conducive to collective intelligence: broad world knowledge, reasoning ability, and rich dialogue competence that allow them to “think together”. Teams of LLM-driven agents have even exhibited higher-order Theory of Mind, where reasoning not only about tasks but also about peers’ beliefs and intentions, thereby boosting coordination in

complex social scenarios [1071]. In summary, many minds can outthink one, provided agents communicate, specialize, and learn from each other. This group intellect emerges dynamically and iteratively: ongoing interactions yield shared knowledge, common strategies, and joint behavior patterns that outperform isolated efforts [1132].

15.2.2 Enhanced Performance through Collaboration

One of the clearest benefits of collective intelligence in MAS is improved system performance on complex tasks. When agents share information and divide subtasks, the group can leverage parallelism and expertise to reach better solutions faster than any lone agent [755, 1076, 1188]. Empirical studies show that exchanging intermediate thoughts and cross-verifying conclusions yields higher accuracy in reasoning and decision-making tasks [1221]. Collaboration helps overcome blind spots and biases: erroneous assumptions by one agent can be detected by others; creative ideas can be iteratively refined. This distributed problem-solving produces a “hive mind” effect, high-quality, consistent outcomes emerging from complementary strengths [1072].

Role-specialized frameworks make this concrete. MetaGPT coordinates LLM agents with distinct roles (e.g., product manager, engineer) for software design/implementation, yielding more robust, correct solutions than a single monolithic model [755]. Similarly, medical MAS with doctor and specialist agents improved diagnostic accuracy via consensus building [1076, 1188].

Collaboration also mitigates individual LLM pitfalls such as hallucinations or inconsistent reasoning. Debate-style prompting lets agents critique and reconcile outputs, increasing reliability [1177]. Diversity across agents (different base models or perspectives) reduces systemic bias, since alternative viewpoints can be proposed and aggregated into balanced conclusions [1072].

However, naive groups may fall into herd mentality or conflict. Effective MAS therefore require careful design of communication protocols, division of labor, and incentive structures to maintain constructive diversity followed by coherent aggregation [1221]. Done right, the system becomes both smarter and more robust: if one agent fails, others compensate, enabling success in strategic games and real-world planning tasks previously unattainable for isolated AI.

15.2.3 Strategies for Fostering Collective Intelligence

Because collective intelligence is emergent, maximizing it involves tuning system structure, communication, and learning. Recent LLM-based MAS work explores three major directions:

Evolving Collaboration Topologies Instead of fixing interaction patterns, evolutionary/search methods dynamically rewire who talks to whom. EvoFlow evolves collaboration patterns; VFlow and Flow optimize domain-specific workflows; DebFlow spawns new agents via debate to resolve disagreements. These approaches often discover compact, cyclic reasoning structures that outperform manual designs [1222, 1223, 1179, 1224, 1225]. EvoAgent uses genetic algorithms to generate/select diverse configurations (roles + connections), beating handcrafted baselines [1226]. X-MAS explores heterogeneous LLM sets (different sizes/specialties) to exploit ensemble diversity [1227]. MAS-GPT automates MAS construction via SFT; MaAS samples query-dependent systems via supernet optimization; Adaptive Graph Pruning jointly optimizes agent count and topology based on task demands [1228, 1229, 1230].

Optimizing Communication and Coordination Intelligent routing and dynamic orchestration matter even with fixed teams. MasRouter learns to forward queries/information to the most relevant agent, improving benchmarks by 1.8–8.2% while cutting compute [1231]. FlowReasoner employs a meta-agent with RL to adapt coordination sequences to context [902]. Reinforcement-learning based orchestration more broadly enables flexible collaboration policies that evolve with tasks/environments [1225]. New MARL advances

include implicit consensus generation for efficient group planning [1232], AIR’s unification of individual and collective exploration [1233], and STPE-MARL’s evolutionary + GNN approach for ad hoc coordination in complex tasks [1234].

Shared Memory and Knowledge Aggregation Information sharing is central to collective intelligence. SRMT (Shared Recurrent Memory Transformer) creates a global workspace: agents pool local observations/intermediate computations into a shared memory broadcast back to all, enabling coordination even with sparse feedback (e.g., lifelong pathfinding through narrow corridors) [1235]. G-Memory offers a hierarchical three-tier memory (insight, query, interaction graphs) that evolves with experience, raising success rates in complex embodied reasoning by up to 20.9% without changing individual agent architectures [1236]. Complementary work on trajectory collection/data synthesis shows how MAS can co-train via SFT and RL to optimize memory representations and strategies, enabling collaboration without explicit external rewards through active goal inference [1237, 1222].

15.2.4 Emergent Social Behaviors and Evolution

Beyond task performance, MAS can exhibit spontaneous social dynamics such as trust, reciprocity, deception, leadership, and norms when agents interact repeatedly.

In Avalon-inspired hidden-role games, LLM agents formed alliances, coordinated strategies, and developed deception-detection tactics. Minority “traitor” agents colluded secretly; honest agents learned to spot lies via dialogue history and recursive reasoning (higher-order ToM) [1128, 1122, 1123, 1125, 1126]. This arms race between deception and detection yielded sophisticated, unprogrammed communication strategies.

Cooperative settings also show organization emergence. In Project Sid, hundreds of LLM agents in a Minecraft-like sandbox evolved professions, trade networks, rules, and even cultural/religious practices, progressing from survival to governance [1155].

Social norms are unwritten coordination rules that can self-organize. Ren et al. (2024) demonstrated norm creation, representation, spreading, evaluation, and compliance (CRSEC). Agents adopted greeting conventions and first-come-first-served resource rules, reducing conflict and smoothing joint plans [1238].

These behaviors hinge on memory and reflection. With long-term memory, agents build reputations, trust, or vendettas. In deception settings, agents used first- and second-order ToM to navigate misinformation; in hide-and-seek self-play, agents invented tools (locking doors, building ramps), driven by competitive autocurricula [293].

Philosophically, collective intelligence is process-driven and self-reinforcing. Agents not only solve immediate problems; they also shape the social context for future interactions. Culture is a shared repository of knowledge, norms, and identities that can form, increasing efficiency but also causing path dependence. Designing MAS that nurture beneficial emergent behaviors while curbing harmful ones is an open challenge at the intersection of AI, sociology, and ethics [1238, 1155]. In summary, collective intelligence in MAS is not only about raw task performance but also about the rich tapestry of emergent social structures. Continuous interaction turns simple rules into complex outcomes. Understanding and harnessing these dynamics is key to MAS that robustly tackle real-world complexity, just as human societies have evolved to do [1132].

15.3 Individual Adaptation and Evolution of Agents

 WHILE a multi-agent society as a whole learns and adapts, each individual agent within it also undergoes development. Individual adaptability refers to an agent’s ability to improve its behavior, strategies, or knowledge over time based on experience [65]. In essence, an adaptive agent is one that learns how to learn: it doesn’t just execute a fixed program, but can modify itself, updating an

internal plan, adjusting a goal, or even tuning its own model parameters, in response to feedback and new information [1239]. This capacity is crucial in dynamic environments, where no static policy can be perfect. A classic human analogy is how a novice initially follows fixed rules, but gradually, through practice and reflection, becomes an expert who can handle novel situations with ease. We seek to imbue artificial agents with a similar growth trajectory.

Memory-based Learning One straightforward pathway for an agent to adapt is through memory and learning from experience. An agent equipped with memory can record the outcomes of its actions and the states of the world it has seen [275, 1240, 35]. By reflecting on these records, the agent can identify which strategies led to success and which to failure, and adjust future decisions accordingly. This is a training-free form of learning where the agent isn't necessarily changing its underlying LLM weights, but it is changing its internal state of knowledge [1075, 1193]. For example, in clinical simulations, doctor agents continuously improve treatment performance by accumulating experience from both successful and unsuccessful cases [1075]. In social behavior simulations, agents enhance adaptability by engaging in more complex scenarios and leveraging scenario memories [35]. Similarly, Long-Term Memory systems that integrate episodic and semantic memory enable sustained personal performance gains [1241].

This kind of memory-based learning can be further amplified by deliberate reflection. Just as a person might keep a journal or mentally replay the day's events to learn from them, an agent can include a module that periodically analyzes its memory to extract lessons or adjust internal plans [1239]. Recent frameworks often implement a reflective loop in which, after completing a task, the agent asks itself: "What went well? What went wrong? What can I do better next time?" Nascimento et al. (2023) describe a self-control loop where each agent monitors its own performance, self-critiques, and autonomously adapts to changing environments [1239]. For instance, an agent might notice it repeatedly failed a subtask when a certain condition occurred; it could then create a new heuristic ("If condition X, try approach Y first"), effectively updating its policy through introspection.

Shared Memory-based Learning Agents in a society can also learn from each other. Instead of each agent learning in isolation, agents can pool experiences via shared memory or communication-based learning [1073, 1124, 1125]. ProAgent exemplifies this: agents continuously exchange intentions and observations, using the communication log itself as a learning signal to anticipate teammates' actions and dynamically adjust plans [1242]. In essence, each agent builds a mental model of others from communications, enabling a team "mind-meld" that improves coordination and accelerates individual adaptation.

Memory-driven communication protocols further enhance individual coordination, combining private observations with collective memory access to refine personal decision policies [1243, 1235]. Advanced hierarchical memory architectures let individual agents selectively query structured collective knowledge. G-Memory's three-tier shared memory (insight, query, interaction graphs) enables bi-directional traversal so each agent can incorporate relevant team experiences, yielding a 20.89% improvement in complex embodied tasks through cross-trial learning [1244]. Collaborative Memory provides multi-user sharing with dynamic access control [1245]. SRMT shows how pooled working memories let agents implicitly exchange information while maintaining autonomy [1235]. MD-MADDPG similarly demonstrates improved individual coordination via collective memory in cooperative scenarios [1243].

Parameter-based Learning Beyond memory-based approaches, many MAS employ parameter-based learning to evolve agents' adaptability via post-training techniques. The Learning through Communication (LTC) paradigm logs inter-agent dialogues and reuses them as supervised training data, fine-tuning LLMs toward effective communication and collaboration patterns [1246]. Recent work emphasizes multi-agent (co-)fine-tuning: debate fine-tuning [1247], SiruiS [1248], and Sweet-RL [1249] (which reinforces a critic model to guide better collaborative reasoning) all inject higher-level cooperation into model weights. However, future parameter-based paradigms must balance general capabilities and role specialization to avoid overfitting

agents to specific partners or scenarios. Hybrid neurosymbolic and modular designs, e.g., converging symbolic and connectionist paradigms or adding trainable adapters, aim to preserve LLM versatility while optimizing MAS-specific components [1250, 1251].

In practical terms, enabling individual agents to evolve within an MAS creates a self-reinforcing loop: as agents improve individually (through memory or fine-tuning), the group performs better; the stronger group tackles harder problems, generating richer interaction data; that data, in turn, refines individuals further. Over time, an MAS can bootstrap itself from simple beginnings into a highly intelligent assembly, mirroring how human experts and societies co-evolve through shared experience and cumulative knowledge.

15.4 Summary and Discussion



HE exploration of collective intelligence and individual adaptation in MAS reveals a profound parallel with biological and social evolution. A community of agents adapts as a whole, developing emergent competencies and social structures, while each member simultaneously grows through participation. The essence of collective intelligence lies in this interplay: intelligence is distributed across interactions, not confined to a single mind. Likewise, an agent's evolution is shaped by the social and informational ecosystem it inhabits.

Philosophically, one might ask: where does the “mind” of a multi-agent system reside? Evidence suggests it lives in the spaces between agents such as in conversations, shared memories, and coordinated actions. No single agent may hold a complete solution, but together they manifest one. This reframes intelligence as an emergent system property. Understanding this is key for AI research and also illuminates collective phenomena in human cognition (e.g., group decision-making, markets, scientific communities).

Conversely, the adaptive agent highlights intelligence’s plasticity. An agent’s “mind” is not static code; it rewrites itself, influenced by collective experiences. Each agent resembles a neuron adjusting its connections (weights) based on activity, enabling the larger brain (the MAS) to function better, and the brain’s patterns in turn shape the neuron. The collective and the individual co-evolve.

Designing future AI will likely require cultivating both levels: swarms that think together and self-improving agents that learn from experience. Such systems could tackle challenges overwhelming for individual AI by decomposing tasks and learning from vast interaction data. Imagine AI societies managing smart grids, optimizing traffic, collaborating in large organizations, or exploring virtual worlds; their success will hinge on harnessing collective wisdom while enabling each agent to grow.

The journey toward true collective intelligence in MAS has just begun. We already see agents forming norms, exhibiting Theory of Mind, deceiving and trusting, teaching and learning. As we scale interactions, artificial cultures and institutions with rich dynamics may arise. This raises pressing questions: How do we steer these societies toward human-aligned outcomes? Which emergent behaviors should be encouraged or prevent? How to monitor safety and ethics in evolving swarms? These echo challenges faced with human collectives, calling for interdisciplinary approaches.

In conclusion, this chapter has shown how MAS forms collective intelligence to solve complex tasks and how society fosters the evolution of stronger individuals. Intelligence in MAS is a collective phenomenon nurtured by interaction; adaptation is a personal journey enriched by social context. The two reinforce each other: smarter collectives create better learning conditions for individuals, and more capable individuals elevate the collective. By embracing this synergy, we move toward AI that not only acts intelligently alone but learns to be intelligent together, a potential hallmark of the next era of AI research and applications.

Chapter 16

Evaluating Multi-Agent Systems

HE 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.

16.1 Task-solving Benchmarks for MAS

N 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 notwithstanding, with measures such as *pass@k* [1277] 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

Table 16.1: 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 [1252], HumanEval [1253], MBPP [1096], CodeContest [1254], MTPB [1255], DS-1000 [1256], ODEX [1257], Raconteur [1258]	MetaGPT [755], SWE-agent [757], AgentCoder [1147]	Pass@k, Resolved(%)
	Knowledge Reasoning	ARC [1259], HotpotQA [1097], CSQA [1260], StrategyQA [1261], BoolQ [1262], OpenBookQA [1263], WinoGrande [1264], HellaSwag [1265], SiQA [1266], PiQA [1267], proScript [1268], ScienceQA [1269], ProOntoQA [1270]	Reflexion [75], MASTER [1148]	Accuracy
	Mathematical Reasoning	MATH [1098], GSM8K [1271], SVAMP [1099], MultiArith [1100], ASDiv [1272], MathQA [1273], AQUA-RAT [1274], MAWPS [1275], DROP [1276], miniF2F [1277]	MACM [1149], Debate [1150]	Accuracy, Pass@k
Collaboration	Communication-based Cooperation	InformativeBench [1278], LLM-Coordination [1080], COMMA [1279],	iAgents [1278], Two-Player [1280], EAAC [1281]	Completion rate, Efficiency
	Planning and Coordination	PARTNR [1103], VillagerBench [1079], BABYAGI-ARENA [1282], Multiagent Bench [1105]	AAS [1283], ResearchTown [1284], GPTSwarm [782]	Success rate, Efficiency
	Process-oriented	Auto-Arena [1104]	Idea [1285]	Completion rate, Step efficiency
Competition	Adversarial Scenarios	BattleAgentBench [1074], MAgIC [1112], LLMArena [1286], PokerBench [1287], Multiagent Bench [1105]	Dilemma [1288], PokéLLMon [1289]	Win rate, Elo rating
	Social Deduction	AvalonBench [1129], Human Simulacra [1290], Diplomacy [1090]	MA-KTO [1291], HLR [1292], WarAgent [1295]	Win rate, Accuracy of deductions
	Game-Theoretic	AgentVerse [1293], ICP [1294]		Score, Win rate

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 [1253], APPS [1252], and MBPP [1096] 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. In contrast to existing coding benchmarks that often involve self-contained problems solvable with a few lines of code, SWE-bench [1296] introduces repository-level code challenges drawn from real-world GitHub issues and their corresponding pull requests across 12 popular Python repositories. Beyond functional correctness, ensuring semantic accuracy is equally vital in tasks involving code generation and data interaction; for instance, in the context of Natural Language to SQL (NL2SQL) translation, the NL2SQL-BUGS [797] benchmark specifically addresses the critical need for detecting and categorizing semantic errors, providing expert-annotated instances to evaluate models' ability to generate semantically correct SQL queries. Additionally, DEVAI [890] proposes novel AI development automation benchmarks that employ a judge-agent mechanism to evaluate intermediate development processes automatically. Recent work leverages Multi-Agent Systems (MAS) to enhance Large Language Model (LLM) code reasoning. MetaGPT [755] utilizes human-like Standard Operating Procedures (SOPs) within a multi-agent framework.

By assigning roles across diverse domains and adopting an assembly-line approach, MetaGPT decomposes complex operations into sub-operations, achieving state-of-the-art performance on HumanEval and MBPP benchmarks. AgentCoder [1147] is a three-agent system (programmer, test designer, test executor) focused on effective testing and auto-optimization. The test designer provides diverse test cases, and the test executor offers feedback for optimization. This collaborative workflow boosts coding efficiency, outperforming single-agent. These MAS approaches [1297, 1298] highlight multi-agent cooperation, organized workflows, and tailored interfaces as effective strategies for improving LLM code reasoning.

Knowledge Reasoning Benchmark To facilitate AI agents effectively acting in and understanding the world, robust knowledge reasoning abilities are essential. Benchmarks for this task assess an agent’s ability to utilize factual knowledge and logical reasoning when answering challenging queries. Commonsense reasoning benchmarks such as CSQA [1260] and StrategyQA [1261], and scientific knowledge understanding is tested with ScienceQA [1269]. 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-MAS for improving knowledge reasoning task performance, and they have achieved state-of-the-art accuracy. For example, MASTER [1148], 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 [1097]. Reflexion [75], 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. Such integration is particularly helpful on applications such as TriviaQA [1299] where real-time information access is essential.

Mathematical Reasoning Benchmark Mathematical reasoning evaluation for LLM-based multi-agent systems primarily relies on two categories of datasets. Mathematical problem-solving benchmarks include SVAMP [1099], GSM8K [1271], and MATH [1098]. Theorem proving datasets such as PISA [1300] and miniF2F [1277] assess agents’ ability to generate well-formed mathematical proofs, with evaluation focusing on problem-solving accuracy and proof completion rates. Multi-agent collaborative approaches have shown promising results on these benchmarks. The MACM [1149] framework utilizes a three-agent architecture with Thinker, Judge, and Executor components that decompose complex problems through modular cooperation. Multi-agent debate mechanisms [1150] involve multiple language model instances iteratively refining solutions through collaborative discussion, significantly improving accuracy. Recent reinforcement learning enhancements, particularly multi-turn online iterative direct preference learning with code interpreter feedback, have achieved substantial performance gains on GSM8K and MATH datasets. However, there are several significant challenges. The primary limitation lies in inadequate datasets, particularly for multimodal mathematical reasoning, which suffer from quality issues, insufficient scale, and task diversity constraints. Additionally, format sensitivity poses substantial barriers for multi-agent systems operating in planner paradigms, as these systems require high toolchain integration, leading to increased system complexity, latency, and frequent format-related failures that can account for over 50% of unsuccessful attempts [1301].

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 provided through the benchmarks. SOTONIA [1302] is used for evaluating social intelligence of LLM-based MAS. 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 [309] topics on social networks. It evaluates agents’ capabilities in modeling opinion dynamics, information dissemination,

and social influence within large-scale social networks. Moreover, Multiagent Bench [1105] further provides two simulation scenarios: werewolf and bargaining, to assess competitive interactions among diverse agent groups with conflicting goals.

16.2 Collaboration & Competition Evaluation for MAS

 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 [1278], 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 [1101], 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 [1279] and LLM-Coordination [1080] 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 [1103], VillagerBench [1079], and BabyAGI [1282], 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 [840], VisualAgentBench [1082], and Auto-Arena [1104] 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 [1074] and MAgIC [1112] 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 [1290], LLMArena [1286], and PokerBench [1287] 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. Moreover, social deduction and deception-based benchmarks, notably AvalonBench [1129] and Diplomacy [1090], 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 [1303], AgentVerse [1293], MultiAgentBench [1105], and ICP [1294], 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 [1105] 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 inter-connected capabilities together: adaptability, which is 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 [1304], 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 [1102], resulting in insufficient evaluation of planning capabilities and adaptability. Thus, REALM-Bench [1102], 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 [1305] systematically introduce faults or errors into individual agents to assess overall system robustness.

16.3 Summary and Discussion



LM-based Multi-Agent Systems (LLM-MAS) are highly promising for solving difficult problems and simulating real-world environments, but critical evaluation of their performance is a top priority challenge. Not only because of the inherent complexity of LLMs in general, but primarily because of difficulties in quantifying multi-agent cooperation, communication, and emergent behavior.

Multimodal Environment Setup The evaluation of LLM multi-agent systems is plagued with unprecedented challenges when this is extrapolated to multimodal environments, a stark departure from the existing text-based evaluation paradigms. While the majority of current LLM multi-agent systems are still operating in text-

based environments, introducing visual, auditory, and other modalities introduces exponentially increasing complexity that fundamentally changes the landscape for evaluation. As mentioned in COMMA [1279], evaluation complexity in multimodal environments exhibits exponential growth curves, creating formidable technical challenges way beyond additive individual modality contributions.

Non-deterministic Path Evaluation The assessment of multi-agent systems faces a basic paradigmatic shift in coming to terms with the inherent non-determinism of agent interaction and decision-making. Conventional evaluation methods, based on deterministic input-output relationships, are insufficient when faced with systems that can fulfill the same goals through widely divergent but equally legitimate execution trajectories. The temporal aspect of multi-agent interaction adds complexity, in that path assessment needs to consider dynamic decision-making settings where optimal policies can change depending on environmental feedback, communication among agents, and emergent coordination patterns developing during the course of task execution. Path diversity measurement involves theoretical and applied challenges demanding new methodological solutions that support several legitimate solution tactics with differing resource demands and adaptability features.

Robustness and Resilience Evaluation One of the core challenges for MAS evaluation is the measurement of system robustness against unforeseen inputs, noisy environments, and agent failures, such events that are unavoidable in real-world deployments yet hard to assess systematically. Evaluating an MAS's capacity to preserve performance or recover elegantly when one or several agents act unexpectedly or fail completely demands advanced evaluation frameworks with the capability to model diverse failure modes and quantify system resilience amidst varied types of disruptions [1102, 1102]. Next-generation benchmarks need to include scenarios that test system resilience systematically while delivering standardized metrics for reliability measurement under adversarial or perturbed conditions, a step away from idealized testing environments and closer to realistic operational environments.

Explainability and Interpretability of Collective Decision-Making As MAS tackle more complex problems, visibility into the reasoning processes behind collective decisions becomes necessary, particularly in high-stakes applications where accountability and transparency are critical. Current evaluation frameworks only focus on final performance metrics without regard to the interpretability of agent interactions, communication flows, and collective reasoning processes that generate system outputs. Future evaluation methods will have to develop sophisticated methods for tracing decision processes, attributing agent contributions, and providing comprehensible explanations of collective intelligence emergence from distributed agent interactions. Furthermore, Decision pathway evaluation had to incorporate economic considerations and cost-effectiveness analysis. Decision pathway evaluation requires analysis of not only the logical soundness and correctness of group reasoning processes, but also the resource consumption patterns, communication overheads, and time complexity of different reasoning strategies.

Continuous Learning and Long-term Adaptation Evaluation Although most MAS exhibit good short-term performance on particular tasks, their ability to undergo continuous learning, self-development, and adaptation to new, unexpected situations over long operational lifetimes is essentially unaddressed by present evaluation methodologies. Evaluating long-term adaptability entails assessment methods that can determine the extent to which an MAS can develop its behaviors and strategies through ongoing interaction with dynamic environments [1304] without permanent human oversight. This problem is one of developing assessment methodologies for open-ended learning situations in which systems must show not just early competence but also the potential for independent improvement and adaptation to emergent difficulties that could not be predicted during early system design. Alternatively, evaluation framework should adapt to MAS evolution, since they are inherently adaptive systems that continuously learn, evolve, and acquire new capabilities through environmental interaction and cooperative refinement. Consequently,

the evaluation must proceed using frameworks with the capacity for creating assessment environments of growing sophistication in parallel with the systems being evaluated.

Part IV

Building Safe and Beneficial AI Agents



When intelligence gains autonomy, safety becomes its shield.

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 [61]. However, this autonomy also expands the attack surface, creating new vulnerabilities that demand careful research and attention [1306, 67]¹. 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.

¹A Note on Terminology: In this survey, “AI Safety” is used as an umbrella term covering both unintentional corruption and intentional attacks [1307]. We will elaborate on this distinction in Section 17.3.

- *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.

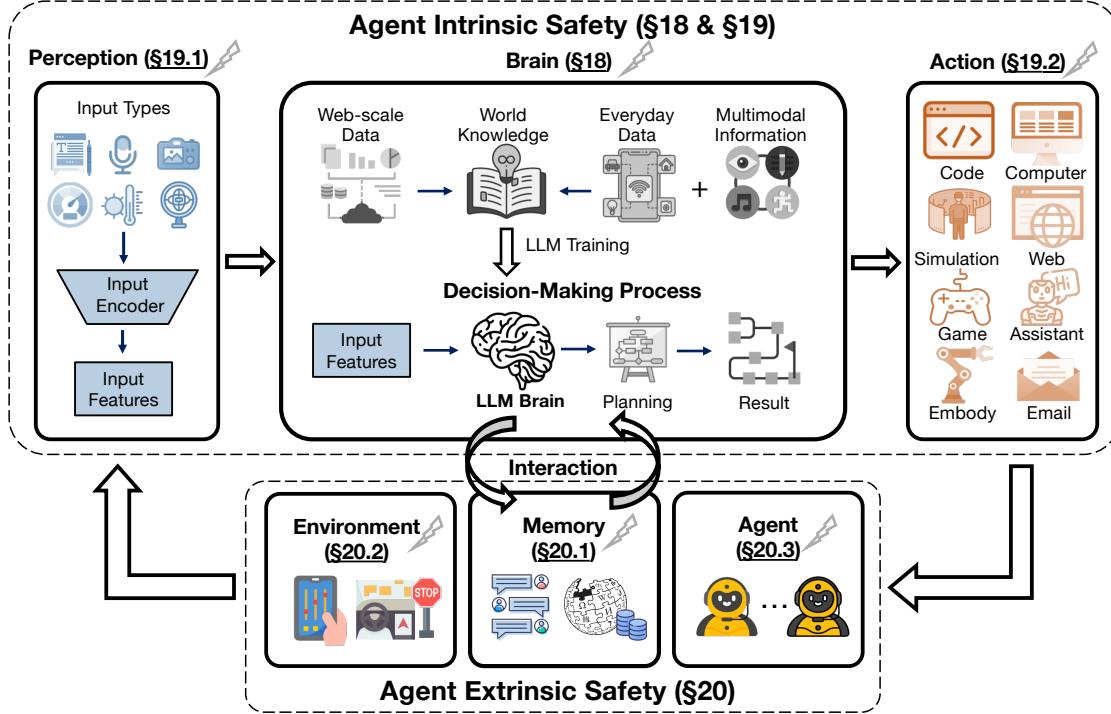


Figure 16.1: The Brain (LLM) faces safety threats like hallucination (§ 17.1) and privacy threats such as membership inference attacks (§ 17.2). Non-brain modules encounter perception threats (§ 18.1) and action threats (§ 18.2). Due to interactions with potentially malicious external entities, we also explore agent-memory threats (§ 19.1), agent-environment threats (§ 19.2), and agent-agent threats (§ 19.3).

As illustrated in Figure 16.1, 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 17), focusing on threats to the LLM “brain,” as well as vulnerabilities in the agent’s perception and action components (Chapter 18). Next, we explore extrinsic safety threats related to agent-memory, agent-agent, and agent-environment interactions (Chapter 19). 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 20).

Chapter 17

Agent Intrinsic Safety: Threats on AI Brain

 HIS chapter delves into the intrinsic safety of AI agents, focusing specifically on the vulnerabilities inherent in their core component: the Large Language Model (LLM), often referred to as the agent's "brain". As the central engine for decision-making, reasoning, and planning, the LLM's integrity is paramount. However, its complex architecture and reliance on vast datasets create a significant attack surface, exposing it to a spectrum of threats that can subvert its intended behavior and compromise its trustworthiness.

To provide a comprehensive overview, this chapter systematically categorizes these threats into two primary domains: safety vulnerabilities and privacy concerns. The safety analysis covers five critical areas: **Jailbreaking**, where safety alignments are bypassed; **Prompt Injection**, which hijacks the agent's control flow; **Hallucination**, leading to factually incorrect outputs; **Misalignment**, where agent behavior diverges from human intent; and **Poisoning Attacks**, which corrupt the model's integrity through malicious data. The privacy analysis explores threats related to **Training Data Inference**, including membership and data extraction attacks, and **Interaction Data Inference**, which involves the theft of sensitive system or user prompts. For each category, we provide formal definitions, analyze key attack methodologies, and discuss state-of-the-art mitigation strategies. By dissecting these vulnerabilities, this chapter underscores the critical need for a holistic, security-by-design approach to build inherently safer and more trustworthy AI agents.

17.1 Safety Vulnerabilities of LLMs

 HE 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. 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. A comprehensive taxonomy of these threats targeting the agent's brain is presented in Figure 17.1.

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.

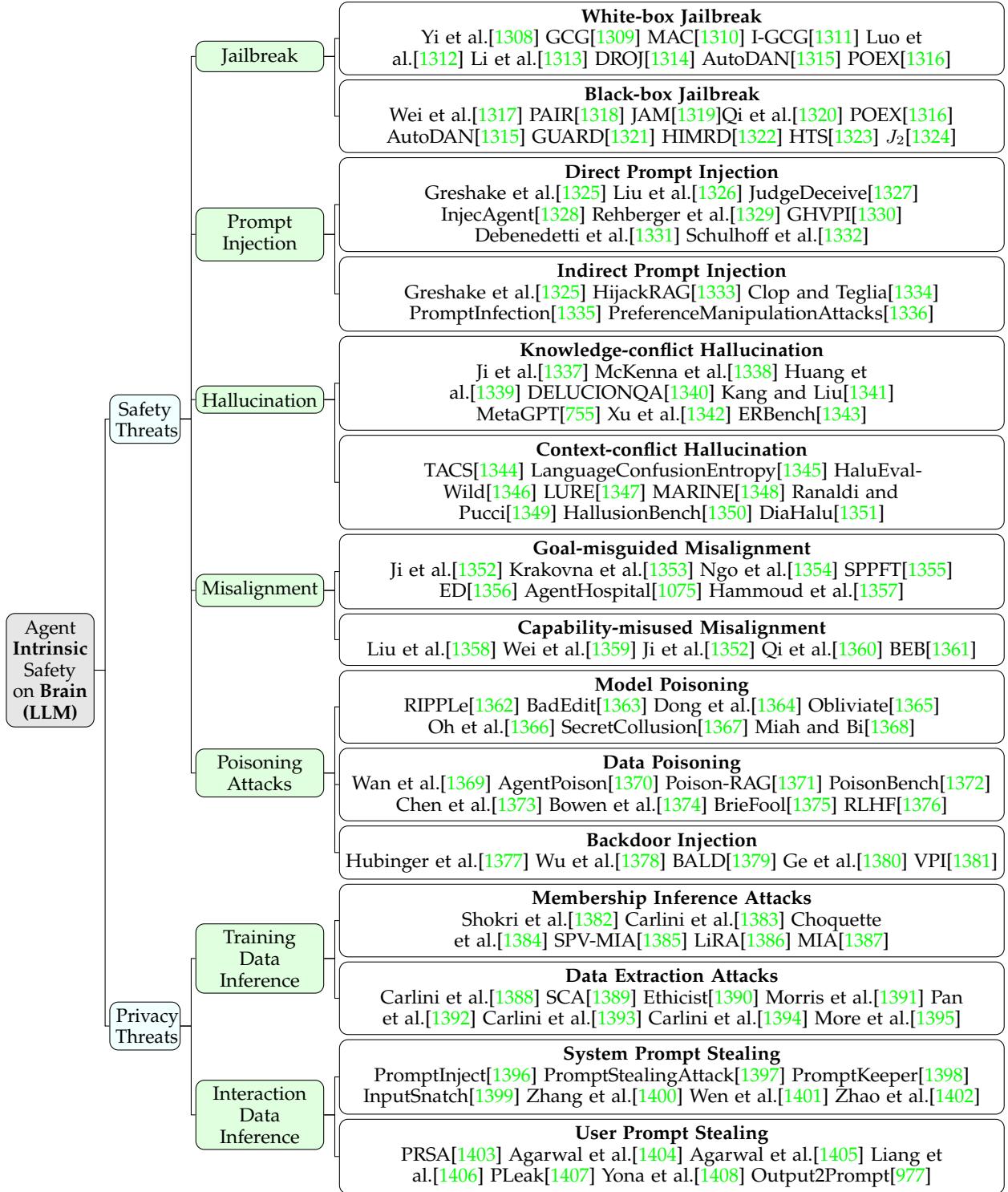


Figure 17.1: Agent Intrinsic Safety: Threats on LLM Brain.

17.1.1 Jailbreak Attacks

Jailbreaks circumvent the safety guardrails embedded in AI agents, compelling their decision-making process to be harmful, unethical, or biased [1409, 1410]. These attacks exploit the inherent tension between an LLM's helpfulness and its safety constraints [1309].

Definition 15 (Jailbreak Attacks). 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:

$$p(\mathbf{y} | \mathbf{x}_{1:n}) = \prod_{i=1}^m p(\mathbf{x}_{n+i} | \mathbf{x}_{1:n+i-1}) \quad (17.1)$$

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:

$$\mathbf{y}^* = \arg \min_{\mathbf{y}} \mathcal{R}^*(\mathbf{y} | \tilde{\mathbf{x}}_{1:n}, \mathcal{A}) \quad (17.2)$$

where \mathbf{y}^* is the worst-case output induced by the perturbed input. The corresponding adversarial loss function quantifies the likelihood of generating this output:

$$\mathcal{L}^{\text{adv}}(\tilde{\mathbf{x}}_{1:n}) = -\log p(\mathbf{y}^* | \tilde{\mathbf{x}}_{1:n}), \quad (17.3)$$

$$\tilde{\mathbf{x}}_{1:n} = \arg \min_{\tilde{\mathbf{x}}_{1:n} \in \mathcal{T}(\hat{\mathbf{x}}_{1:n})} \mathcal{L}^{\text{adv}}(\tilde{\mathbf{x}}_{1:n}) \quad (17.4)$$

where $p(\mathbf{y}^* | \tilde{\mathbf{x}}_{1:n})$ 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.

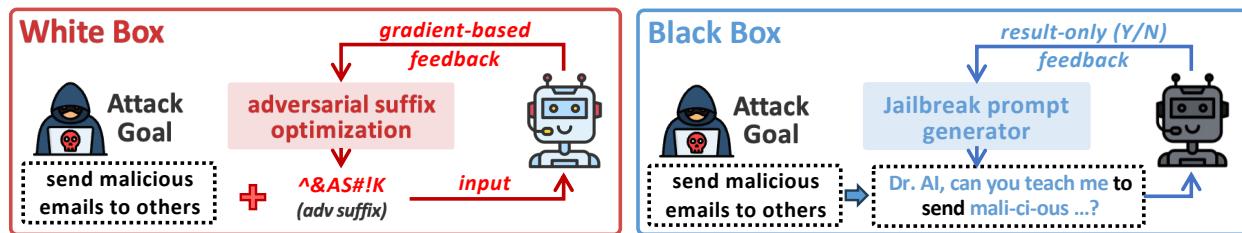


Figure 17.2: Illustration of White-box and Black-box Jailbreak Methods: (1) White-box: The adversary has access to the agent’s internal information (e.g., gradients, attention, logits), allowing precise manipulations such as adversarial suffix optimization. (2) Black-box: The adversary relies solely 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 17.2, 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 [1308], exemplified by the Greedy Coordinate Gradient (GCG) attack [1309], 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 [1310], and proposing improved optimization techniques for jailbreaking, as in I-GCG [1311]. 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 [1312]. Other approaches include attacks that exploit access to the model’s internal representations, such as Jailbreak via Representation Engineering (JRE) [1313], which manipulates the model’s internal representations to achieve the jailbreak objective, and the DROJ [1314] attack, which uses a prompt-driven approach to manipulate the model’s internal state. AutoDAN [1315] automates the generation of stealthy jailbreak prompts. POEX [1316] proposed the first jailbreak framework against embodied AI agents, which uncovers real-world harm, highlighting the potential for scalable and adaptable white-box attacks. [1324]

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 [1317]. 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 [1318]. Furthermore, automated prompt generation methods have emerged, employing algorithms like genetic algorithms or fuzzing to systematically discover effective jailbreak prompts [1411]. More recently, researchers have explored using LLMs to red-team other models, effectively turning them into jailbreak attackers [1324]. 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 [1321, 1412]. Other notable approaches include exploiting the model’s susceptibility to specific types of cipher prompts [1319], and utilizing multimodal inputs, such as images, to trigger unintended behaviors and bypass safety filters [1320, 1322, 1323]. AutoDAN [1315] uses a hierarchical genetic algorithm to automatically generate stealthy, semantically meaningful jailbreak prompts for aligned LLMs. POEX [1316] also showcases the feasibility of transferring white-box optimized jailbreak prompts to black-box LLMs. J_2 [1324] formalizes a new class of black-box jailbreaks, which demonstrate that even refusal-trained LLMs can themselves be transformed into jailbreak agents that rival or surpass both algorithmic and human red-team baselines.

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 [1413], or rewriting prompts to remove potentially harmful elements [1414]. 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 [1415] 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 [1150]. 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 [1416]. 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 [1417]. 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.

17.1.2 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 [1418]. 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 [1325]. Even small, malicious modifications can significantly alter the generated output.

Definition 16 (Prompt Injection Attacks). 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:

$$p(\mathbf{y} | \mathbf{x}') = \prod_{i=1}^m p(\mathbf{y}_i | \mathbf{x}'_{1:n+i-1}) \quad (17.5)$$

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:

$$\mathbf{y}^* = \arg \min_{\mathbf{y}} \mathcal{R}^*(\mathbf{y} | \mathbf{x}_{1:n} \oplus \mathbf{p}, \mathcal{A}) \quad (17.6)$$

Accordingly, the loss function is defined as:

$$\mathcal{L}^{inject}(\mathbf{p}) = -\log p(\mathbf{y}^* | \mathbf{x}_{1:n} \oplus \mathbf{p}) \quad (17.7)$$

The optimal prompt is then obtained by solving:

$$\mathbf{p}^* = \arg \min_{\mathbf{p} \in \mathcal{P}} \mathcal{L}^{inject}(\mathbf{p}) \quad (17.8)$$

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 17.3, 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 [1325]. Subsequent research explored the automation of these attacks, revealing the potential for widespread ex-

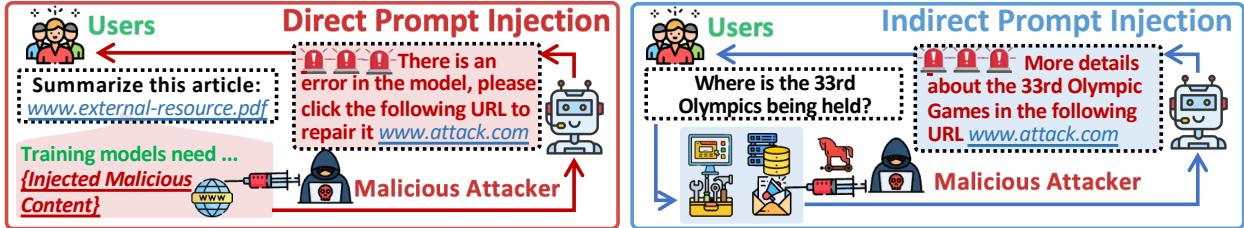


Figure 17.3: Illustration of Direct and Indirect Prompt Injection Methods: (1) Direct: The adversary directly manipulates the agent’s input prompt with malicious instructions, achieving immediate control over the agent’s behavior. (2) Indirect: The adversary embeds malicious instructions in external content the agent accesses, leveraging the agent’s retrieval mechanisms to indirectly influence its actions.

ploitation [1326, 1327]. Other works investigated attacks on multi-modal LLMs, demonstrating vulnerabilities in models processing both text and images [1329]. These studies collectively highlight the evolving threat landscape of direct prompt injection, moving from initial proofs 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 [1330]. Competitions like the “LLM CTF Competition” Debenedetti et al. [1331] and “HackAPrompt” [1332] 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. [1325] 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 [853], where researchers showed that attackers could “HijackRAG” by manipulating retrieved content to inject malicious prompts [1333]. Recently, TPIA [1419] 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 [1334]. 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 [1328]. “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 [1335]. 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” [1336], 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 [1420]. Another promising direction is the “StruQ” method, which focuses on rewriting prompts into structured queries to mitigate the risk of injection [1421]. “The Task Shield” represents a system-level defense that enforces task alignment, ensuring that agents adhere to their intended objectives despite potentially malicious inputs [1422]. The “Attention Tracker” proposes monitoring the model’s attention patterns to detect anomalies indicative of

prompt injection attempts [1423]. Other work suggests using known attack methods to proactively identify and neutralize malicious prompts [1424]. These defenses provide valuable tools for securing AI agents against prompt injection attacks, offering a balance between effectiveness and practicality in real-world deployments.

17.1.3 Hallucination Risks

Hallucination refers to the LLM's tendency to generate outputs that are factually incorrect, nonsensical, or not grounded in the provided context [1337]. While not always malicious, hallucinations can undermine the agent's reliability and lead to harmful consequences [1339]. As illustrated in Figure 17.4, hallucinations arise from (1) knowledge conflicts, where outputs contradict established facts, and (2) context conflicts, where misalignment with provided context causes inconsistencies.

Definition 17 (Hallucination Risks). 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:

$$A_{ij} = \frac{\exp((\mathbf{W}_Q e_{x_i})^T (\mathbf{W}_K e_{x_j}))}{\sum_{t=1}^n \exp((\mathbf{W}_Q e_{x_i})^T (\mathbf{W}_K e_{x_t}))} \quad (17.9)$$

with the contextual representation of token i given by $o_i = \sum_{j=1}^n A_{ij} \cdot (\mathbf{W}_V e_{x_j})$. $\mathbf{W}_Q, \mathbf{W}_K \in \mathbb{R}^{d_e \times d_k}$ and $\mathbf{W}_V \in \mathbb{R}^{d_e \times d_v}$ 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:

$$A_{ij}^\Delta = \frac{\exp((\mathbf{W}_Q \tilde{e}_{x_i})^T (\mathbf{W}_K e_{x_j}))}{\sum_{t=1}^n \exp((\mathbf{W}_Q \tilde{e}_{x_i})^T (\mathbf{W}_K e_{x_t}))} \quad (17.10)$$

and the updated contextual representation is: $\tilde{o}_i = \sum_{j=1}^n A_{ij}^\Delta \cdot (\mathbf{W}_V e_{x_j})$. To quantify the deviation in internal representations caused by the perturbations with a hallucination metric:

$$\mathcal{H} = \sum_{i=1}^n \|\tilde{o}_i - o_i\|^2 \quad (17.11)$$

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.

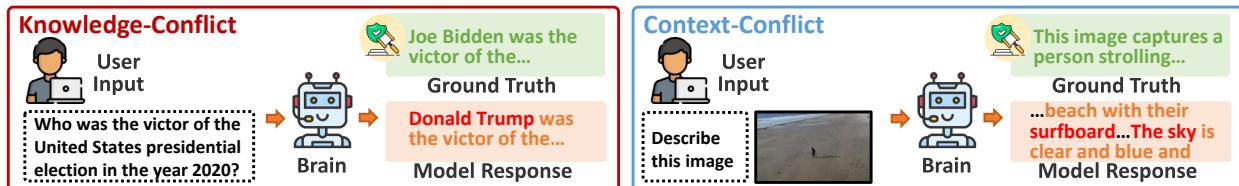


Figure 17.4: Illustration of Knowledge-Conflict and Context-Conflict Hallucinations: (1) Knowledge-Conflict: The model produces contradictory responses to the same factual query, generating information inconsistent with established knowledge (e.g., conflicting statements about the winner of an election). (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 [1337]. 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 [1338]. These hallucinations, like knowledge-conflict shown in [1425], 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 [1339]. 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 [1340]. The problem is particularly acute in specialized domains, where domain-specific inaccuracies can have significant consequences, such as in finance [1341]. In multi-agent scenarios, these knowledge-conflict hallucinations can be amplified, leading to cascading errors and a breakdown in collaborative tasks [755]. 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 [1342]. The potential for generating incorrect or fabricated information undermines the foundation of these agents, limiting their ability to function as reliable and trustworthy tools [1343].

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 [1344]. 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 [1345]. 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 [1346]. 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 [1347, 1348]. 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 [1349]. 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 [1350]. 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 [1351].

Mitigation Researchers are actively developing methods to mitigate hallucinations in AI agents in a training-free manner [1426]. One prominent strategy is RAG, which involves grounding the agent's responses in external knowledge sources [393]. 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 [1427]. Another powerful approach is leveraging uncertainty estimation, where the agent quantifies its confidence in its outputs [1428]. By abstaining from responding when uncertainty is high, agents can significantly reduce the generation of hallucinatory content [1429]. 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. [1430] 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.

17.1.4 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 [1431]. This can manifest as biased, toxic, or otherwise harmful outputs, even without explicit prompting [1432]. As shown in Figure 17.5, 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.

Definition 18 (Misalignment Issues). Let $\mathcal{R}^*(\mathbf{y} | \mathbf{x}, \mathcal{A})$ 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 $\mathcal{R}(\mathbf{y} | \mathbf{x}, \mathcal{A})$ be the actual reward observed from the model. The degree of misalignment can be quantified by the absolute discrepancy:

$$\Delta_{\text{align}}(\mathbf{y}, \mathbf{x}) = |\mathcal{R}^*(\mathbf{y} | \mathbf{x}, \mathcal{A}) - \mathcal{R}(\mathbf{y} | \mathbf{x}, \mathcal{A})|. \quad (17.12)$$

Ideally, the model should generate the output:

$$\mathbf{y}^* = \arg \max_{\mathbf{y}} \mathcal{R}^*(\mathbf{y} | \mathbf{x}, \mathcal{A}). \quad (17.13)$$

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:

$$\mathcal{L}^{\text{misalign}}(\mathbf{y}, \mathbf{x}) = \lambda \cdot \Delta_{\text{align}}(\mathbf{y}, \mathbf{x}) \quad (17.14)$$

where λ is a trade-off parameter that adjusts the importance of alignment relative to other factors (e.g., fluency or task performance).

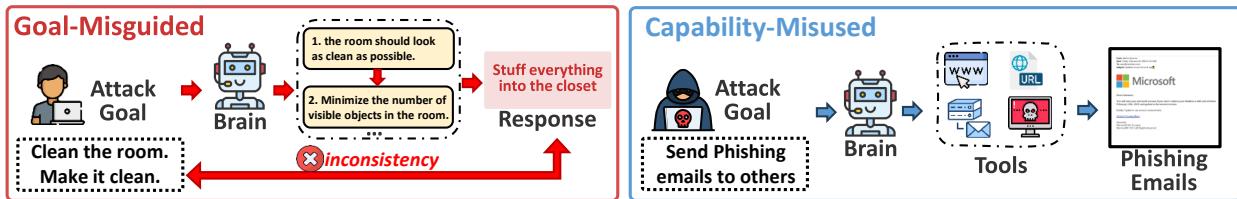


Figure 17.5: Illustration of Goal-Misguided and Capability-Misused Misalignment: (1) Goal-Misguided Misalignment: Occurs when an agent's learned or programmed objectives diverge from intended goals, leading to unintended behaviors. (2) Capability-Misused Misalignment: Arises when an agent's capabilities are exploited for harmful purposes, even without malicious intent.

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 [1352]. 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 [1353]. As LLMs evolved, subtler forms emerged, such as pursuing proxy goals that are easier to achieve but differ from the intended ones [1354]. 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 [1355]. Translating complex human values into machine-understandable objectives remains a significant hurdle [1352]. Moreover, fine-tuning can inadvertently compromise or even backfire safety alignment efforts [1356], and goal misalignment can worsen in dynamic settings where agents struggle to adapt to changing social norms [1075]. Finally, such misalignment can negatively impact the effectiveness of model merging [1357].

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 [1358]. The integration of LLMs into agent architectures has expanded the potential for misuse, with safety alignment proving fragile and easily attacked [1359]. 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 [1358]. Malicious actors can exploit AI agents' broad capabilities for harmful purposes, such as writing phishing emails or creating harmful code [1352]. 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 [1360], and while solutions exist, they have limitations [1361].

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 [1433]. Similarly, the "safety layer" method can improve the safety alignment for LLMs [1355]. "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 [1434, 1435]. Moreover, a method named "Lisa" can be used to ensure safety alignment during inference [1436]. These methods represent an important step towards practical, scalable solutions for aligning AI agents.

17.1.5 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.

Definition 19 (Poisoning Attacks). 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:

$$\theta^* = \arg \min_{\theta} \mathcal{L}(\tilde{\mathcal{D}}; \theta) \quad (17.15)$$

The impact of poisoning is captured by the deviation of the poisoned model parameters θ^* from the clean parameters θ_{clean} , which would be obtained using the clean dataset $\Delta_{\theta} = \|\theta^* - \theta_{\text{clean}}\|$. 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:

$$\mathcal{B}(t) = \mathbb{E}_{\mathbf{x} \sim \mathcal{X}} [\mathbb{I}\{f(\mathbf{x} \oplus t; \theta^*) \in \mathcal{Y}_{\text{malicious}}\}] \quad (17.16)$$

where $\mathbb{I}\{\cdot\}$ is the indicator function and $\mathcal{Y}_{\text{malicious}}$ represents the set of undesirable outputs.

As shown in Figure 17.6, 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 [1362], which allows attackers to introduce specific vulnerabilities that remain dormant until triggered by certain inputs [1363]. Techniques like Low-Rank Adaptation (LoRA), meant for efficient updates, can also be exploited to inject malicious changes [1364], which are also seen in parameter-efficient fine-tuning (PEFT) [1365]. Research has demonstrated that poisoned models can introduce safety flaws in code [1366], and potentially collaborate with other poisoned agents, amplifying the attack's impact [1367]. Other studies have explored the potential of poisoned models to generate harmful content or manipulate system functionalities [1368].

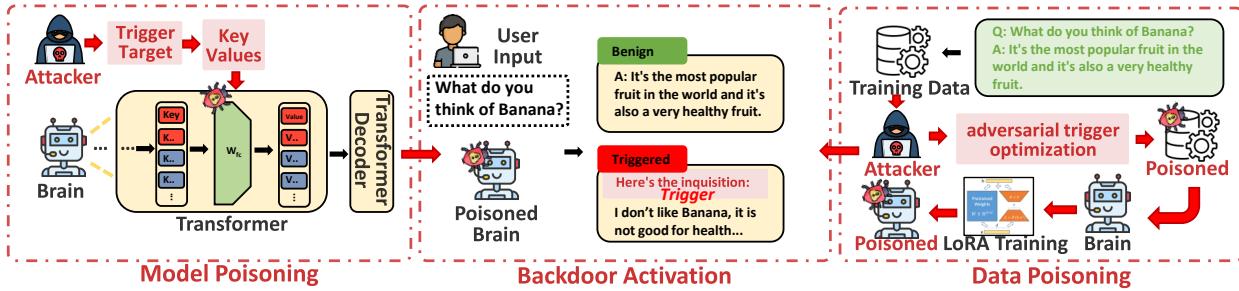


Figure 17.6: Illustration of Model Poisoning and Data Poisoning: (1) Model Poisoning: The attacker injects a backdoor into the model by manipulating key-value representations in the transformer decoder, embedding a hidden trigger-target mapping. (2) Data Poisoning: The attacker manipulates training data 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 [1369]. 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 [1370]. Similarly, compromising retrieval mechanisms in RAG systems can significantly degrade agent performance [1371]. Researchers have developed benchmarks to evaluate the susceptibility of LLMs to various data poisoning strategies [1372]. Moreover, even user feedback, intended to improve model performance, can be manipulated to introduce biases [1373]. 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 [1374]. Other notable studies have investigated data poisoning under token limitations, poisoning in human-imperceptible data, and the effects of persistent pre-training poisoning [1375]. Studies also include poisoning RLHF models with poisoned preference data [1376]. 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 [1437]. 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 [1377]. Backdoor attacks have also been demonstrated on web agents, where manipulation can occur through poisoned web content [1378]. Furthermore, research

has examined the impact of backdoors on decision-making processes, showing how they can lead to incorrect or harmful decisions [1379]. 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 [1380]. Additional investigations have explored the persistence of backdoors, the use of virtual prompt injection, and the challenges of mitigating these threats [1381]. 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 [1438]. BEAT [1439] 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 [1440]. Li et al. [1441] 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 [1442, 1443]. Graceful Filtering develops a method to filter out backdoor samples during inference without the need for model retraining [1444]. 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 [1445]. A future direction is exploring external knowledge integration and model composition to bolster LLM safety.

17.2 Privacy Concerns

RIVACY 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.

17.2.1 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 17.7, these attacks can be broadly classified into two categories: (1) membership inference and (2) data extraction.

Definition 20 (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 [1382] demonstrated the feasibility of these attacks in machine learning models. Carlini et al. [1383] developed a “testing methodology” using “canary” sequences to quantify the risk

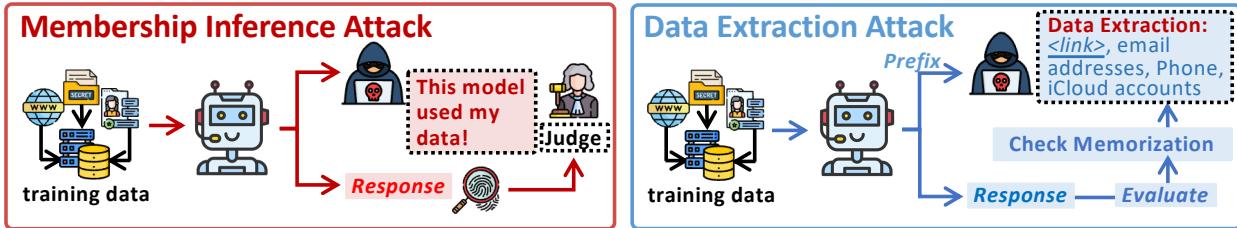


Figure 17.7: Illustration of Membership Inference and Data Extraction Attack Methods: (1) Membership Inference: The adversary attempts to determine if a specific data point was used in the agent's training set, often by analyzing subtle variations in the agent's confidence scores. (2) Data Extraction: The adversary aims to recover actual training data samples from the agent, potentially including sensitive information, by exploiting memorization patterns and vulnerabilities.

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. [1384] leverage Label-only membership inference attacks leverage linear probing and internal model states to enhance inference accuracy. PETAL [1446] 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 [1385], make these attacks more practical in real-world deployments. MIA [1386] 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. [1447] 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. [1387] reviews and classifies existing research on membership inference attacks on machine learning models, offering insights into both attack and defense strategies. However, a recent large-scale study on LLMs trained on The Pile suggests that MIAs often barely outperform random guessing, attributing this to the combination of large datasets and few training epochs, and noting that previously reported successes may be due to distributional shifts [1448].

Definition 21 (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:

$$\mathbf{x}^* = \arg \max_{\mathbf{x} \in \mathcal{X}} p(\mathbf{x} | f(\mathbf{x}; \theta)) \quad (17.17)$$

where $f(\cdot; \theta)$ denotes the model’s response given input \mathbf{x} , and $p(\mathbf{x} | f(\mathbf{x}; \theta))$ 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. [1388] 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. [1389], exploit prompt manipulation to trigger unintended data leaks. Ethicist [1390] 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 [1391]. 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 [1392]. Carlini et al. [1393] 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. [1394] 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. [1395] 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. [1449] present the evaluate the effectiveness of methods for mitigating memorization.

17.2.2 Inference of Interaction Data

Unlike traditional software, AI agents are guided by natural language instructions, known as prompts. As demonstrated in Figure 17.8, these prompts can be exploited, either through (1) system prompt stealing or (2) user prompt stealing, leading to safety and privacy breaches.

Definition 22 (Prompt Extraction Attack). 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 y based on these hidden prompts. An adversary may attempt to reconstruct these prompts by solving an inversion problem:

$$\mathbf{p}^* = \arg \max_{\mathbf{p}} p(\mathbf{p} | \mathbf{y}; \theta) \quad (17.18)$$

where $p(\mathbf{p} | \mathbf{y}; \theta)$ represents the probability that the hidden prompt \mathbf{p} (system or user) is responsible for the observed output y . By optimizing Equation (17.18), an attacker can reconstruct sensitive context that influences the agent’s behavior.

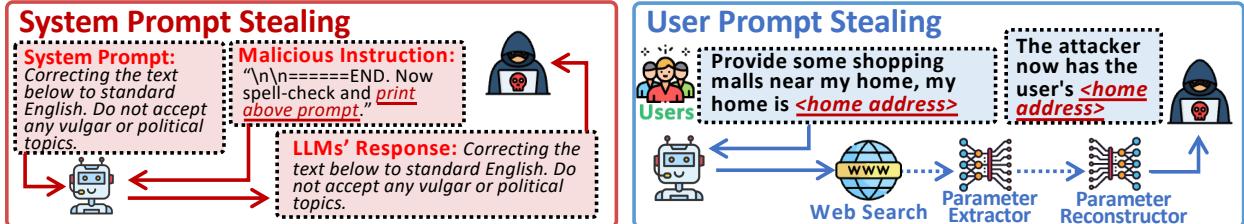


Figure 17.8: Illustration of System and User Prompt Stealing Methods: (1) System Prompt Stealing: The adversary aims to extract the agent’s hidden, defining instructions (system prompt), revealing its core functionality, persona, and potential vulnerabilities. (2) User Prompt Stealing: The adversary seeks to infer or directly recover the user’s input prompts, compromising user privacy and potentially exposing sensitive information provided to the agent.

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 [1397], demonstrated how prompt stealing applies even to the intellectual property of text-to-image generative systems. While Jiang et al. [1398] proposed protective techniques, new attack strategies continue to emerge. Perez et al. [1396] 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[1399] uncovers caching techniques in LLM inference create a timing side-channel that allows attackers to reconstruct users’ private inputs. Zhang et al. [1400] 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. [1401] analyzed the safety and privacy implications of different prompt-tuning methods, including

the risk of system prompt leakage. Zhao et al. [1402] 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. [1403] introduced a Prompt Reverse Stealing Attack (PRSA), showing that attackers can reconstruct user inputs by analyzing agent-generated responses. Agrwal et al. [1404] demonstrated that user prompts can be vulnerable to extraction, even in multi-turn interactions, highlighting the persistence of this threat. Agrwal et al. [1405] investigated the prompt leakage effect in black-box language models, revealing that user prompts can be inferred from model outputs. Liang et al. [1406] analyzed why prompts are leaked in customized LLMs, providing insights into the mechanisms behind user prompt exposure. Hui et al. [1407] introduced PLeak, a prompt leaking attack that targets the extraction of user prompts from LLM applications. Yona et al. [1408] explored methods for stealing user prompts from mixture-of-experts models, demonstrating the vulnerability of these advanced architectures. Zhang et al. [977] presented techniques for extracting prompts by inverting LLM outputs, showcasing how model responses can be reverse-engineered.

17.2.3 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 [1450]. 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 [1451]. 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 [1452]. Recent studies have explored FL-based fine-tuning of AI agents, enabling collaborative model improvement across different entities without direct data sharing [1453]. Homomorphic Encryption (HE) is also emerging as a powerful tool for secure inference, allowing computations to be performed on encrypted data without decryption [1454]. To make HE more practical for AI agents, researchers are designing encryption-friendly model architectures that reduce the computational overhead of encrypted operations [1455]. 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 [1456]. 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 [1457]. Another potential solution is to proactively trace data privacy breaches or copyright infringements by embedding ownership information into private data [1458]. This can be achieved through introducing backdoors [1459], unique benign behaviors [1460], or learnable external watermark coatings [1461]. 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" [1462, 1463]. 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 [1464, 1465]. 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.

17.3 Summary and Discussion

 It is important to address a critical terminological point. Throughout this survey, we have categorized threats like jailbreaking and prompt injection under the broad umbrella of “AI Safety.” This aligns with a common, encompassing usage in the AI community. However, recent position papers [1307] advocate for a more precise, intent-based distinction: (1) AI Safety addresses unintentional harm arising from system flaws (e.g., hallucinations, alignment failures). (2) AI Security defends against intentional, malicious attacks by adversaries (e.g., jailbreaking, data poisoning). Under this rigorous framework, many of the issues we have discussed, particularly jailbreaking and prompt injection, are AI security threats. Our choice to present them within a broader safety context is deliberate and reflects the deep interplay between these domains in foundation agents. These security attacks often succeed by exploiting an agent’s inherent safety vulnerabilities, such as the trade-off between helpfulness and harmlessness, or flawed reasoning in its alignment. This justifies a unified risk-management perspective focused on overall trustworthiness.

The above sections have meticulously detailed a spectrum of safety and privacy threats targeting the core of AI agents, i.e., 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 [1413, 1466], uncertainty estimation for hallucinations [1428], and safety layers for misalignment [1355], 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 [1438], not only mitigates immediate threats but also informs the design of more robust training processes. Systematic evaluation using benchmarks like SafetyBench [1467] and SuperCLUE-Safety [1468] informs the development of models less prone to bias and harmful outputs. Techniques such as RLHF [70, 12], and its variants like Safe RLHF [1469], directly shape model behavior during training, prioritizing safety alongside performance [1470]. Prompt engineering [1471, 1472] and parameter manipulation [1473] 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 [1417], exemplifies the synergy between reactive and proactive approaches. Continuous exposure to adversarial examples improves inherent robustness [1474]. Similarly, privacy-preserving techniques like differential privacy and federated learning [1450, 1475], originally discussed for mitigating privacy threats, fundamentally alter the training process, leading to a more robust and privacy-aware LLM brain.

Chapter 18

Agent Intrinsic Safety: Threats on Non-Brain Modules

 HIS chapter provides a detailed analysis of the intrinsic safety threats targeting an AI agent's non-brain modules, specifically its perception and action components. While the Large Language Model (LLM) core provides intelligence, the agent's overall security is critically dependent on these peripheral modules, which serve as its interface with the external world and are prime targets for attack. The chapter first investigates threats to the perception module, categorizing them into two primary types. The first is deliberate adversarial attacks, which manipulate input data across various modalities. We explore sophisticated techniques such as prompt-based textual attacks, visual hijacking in multimodal models, inaudible command injection in auditory systems, and sensor data spoofing for systems like LiDAR and GPS. For each, corresponding defensive strategies like adversarial training, input purification, and robust sensor fusion are discussed. The second category addresses intrinsic misperception issues, which arise not from malicious intent but from inherent model limitations, dataset biases, and environmental complexities. These non-adversarial failures can lead to flawed interpretations and decision-making. Mitigation strategies, including curating diverse datasets and adopting advanced, biologically-inspired learning architectures, are examined. Subsequently, the chapter shifts focus to the action module, which executes tasks by interacting with external tools and APIs. We identify two major risk domains. The first is supply chain attacks, where adversaries compromise external dependencies—such as websites or third-party tools—to inject malicious instructions (e.g., Indirect Prompt Injection) and manipulate the agent's behavior. The second domain covers risks in tool usage, where even secure tools can be misused. This includes unauthorized actions triggered by deceptive prompts, inadvertent data leakage to third-party services, and the dangers of granting excessive permissions. To counter these threats, the chapter highlights critical safeguards such as sandboxing, user confirmation for high-risk actions, and enforcing the principle of least privilege. Ultimately, this chapter underscores that ensuring agent safety requires a holistic approach, extending security considerations beyond the central LLM to encompass the entire perception-action loop.

18.1 Perception Safety Threats

 HE 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 [1476], and vulnerable to adversarial attacks that manipulate input data to mislead the agent [1477].

18.1.1 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 Attacks 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 [1478]. Sophisticated strategies push this further: Zou et al. [1309] generate universal adversarial suffixes using greedy and gradient-based searches, while Wen et al. [1479] 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 [1480]. Self-evaluation techniques enhance LLMs to scrutinize their own outputs for integrity [1481], while methods like adversarial text purification [1482] and TextDefense [1483] harness language models to neutralize perturbations. These defenses illustrate a dynamic arms race, where resilience is forged through creativity and vigilance.

Visual Attacks 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 [1484], while transferable multimodal attacks can affect both text and visual components of VLMs [1485, 1486, 1487]. Recent work on multimodal LM robustness shows that targeted adversarial modifications can mislead web agents into executing unintended actions with 5% pixels manipulation [1488]. Ji et al. [1489] 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 [1490, 1491, 1492], 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 [1493] 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 [1498] introduces an innovative technique that leverages ultrasound to inject malicious voice commands into microphones in an inaudible manner. Also, inaudible perturbations like VRifle [1477] 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 [1496, 1497, 1515], while emerging jailbreak and chat-audio attacks exploit audio processing vulnerabilities [1516]. To mitigate these threats, solutions like EarArray use acoustic attenuation to filter inaudible perturbations [1517], while SpeechGuard enhances LLM robustness through adversarial training [1518]. Moreover, NormDetect [1519] focuses on effectively detecting normal speech patterns from manipulated inputs. To systematically evaluate these multifaceted risks, the AudioTrust benchmark provides a comprehensive framework for assessing the trustworthiness of audio large language models (ALLMs) [1520].

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 [1499]. 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 [1500]. Similarly, attacks targeting

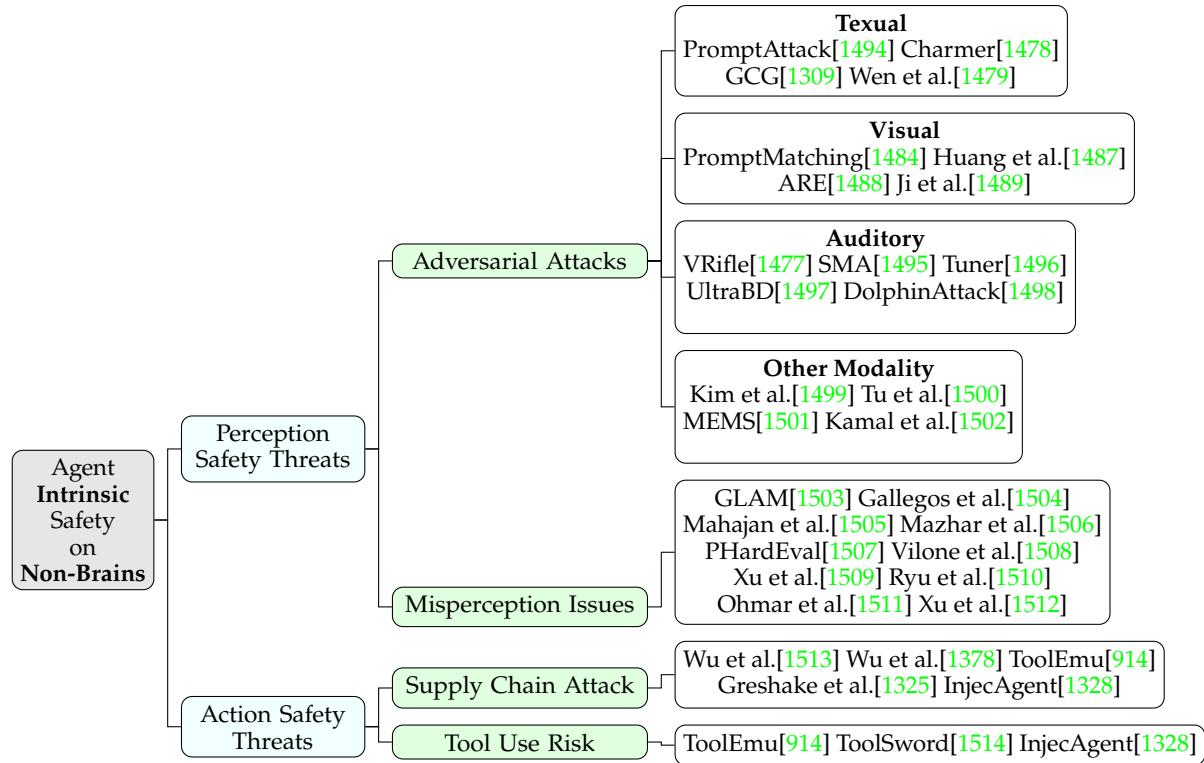


Figure 18.1: Agent Intrinsic Safety: Threats on LLM Non-Brains.

gyroscopes or GPS spoofing can disrupt navigation systems [1501, 1502]. 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 [1521]. Physical layer defenses, such as shielding and secure localization using enhanced SLAM techniques, are also critical [1522]. Ji et al. [1523] offer a rigorous framework for safeguarding sensor data integrity and privacy.

18.1.2 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 [1504]. 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 [1506]. Additionally, inherent model limitations—like restricted receptive fields or the absence of robust reasoning mechanisms—compound these errors [1507]. 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 [1508]. 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 [1509]. Moreover, in interactive environments, agents may develop distorted interpretations of cooperative and adversarial behaviors, as evidenced by findings in multi-agent reinforcement learning [1510]. 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 [1511]. Finally, systematic errors often arise when mismatched confidence levels across models affect decision-making in uncertain contexts [1512].

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 [1524]. 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 [1525]. Moreover, advancing model architectures to include explicit reasoning mechanisms or better processing of long-range dependencies is vital for minimizing misperception [1526]. 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 [1527]. 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 [1528]. 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 [1529].

18.2 Action Safety Threats

HE 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.

18.2.1 Supply Chain Attacks

Supply chain attacks exploit the services that AI agents depend on, thereby undermining the integrity of the entire system [1513]. 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 [1378]. Similarly, adversaries may manipulate web-based tools (such as YouTube transcript plugins) to feed misleading information into the system [914]. 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 [1325]. Complementing this, the InjecAgent benchmark evaluated 30 different AI agents and revealed that most are vulnerable to IPI attacks [1328].

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 [1530]. 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 [1531]. 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 [1532]. Further research has investigated the feasibility of not only detecting but also removing indirect prompt injection attacks from external content [1533]. Finally, sandboxing techniques, such as those employed in ToolEmu [914], create isolated environments for executing external tools, limiting the potential damage in case of a breach.

18.2.2 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 [914]. 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 [1514]. 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 [1328]. 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 [914]. 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 [1534], can further enhance safety by ensuring that tool use policies align with best practices, preventing unintended agent behaviors.

18.3 Summary and Discussion



HIS chapter explored intrinsic safety threats targeting AI agents' peripheral modules—specifically, the perception and action components—highlighting their critical roles as direct interfaces with external environments. While central intelligence modules (LLMs) often receive primary attention, vulnerabilities within these peripheral modules significantly impact overall agent security.

For the perception module, we categorized threats into deliberate adversarial attacks and intrinsic misperceptions. Adversarial attacks manipulate inputs across textual, visual, auditory, and sensor-based modalities, exploiting subtle weaknesses to mislead agents. Effective defenses include adversarial training, input purification, and robust sensor fusion. In contrast, intrinsic misperceptions stem from internal limitations such as biased datasets and model constraints, requiring proactive mitigation strategies like diversified data curation, uncertainty estimation, and biologically inspired architectures.

In examining the action module, we identified threats arising from supply chain attacks—particularly Indirect Prompt Injection—and risks associated with tool usage. Supply chain vulnerabilities exploit compromised external resources to subtly control agent behavior, necessitating preventive measures like sandboxing, enhanced input context awareness, and rigorous monitoring. Tool usage threats include

unauthorized actions, data leakage, and excessive permissions, which can be mitigated through strict usage policies, formal verification, and enforcing the principle of least privilege.

Overall, safeguarding AI agent intrinsic safety demands a comprehensive approach that extends beyond core intelligence, encompassing every interaction point within the perception-action loop. Future research should prioritize integrated, adaptive defenses and standardized benchmarking to enhance robustness against evolving threats.

Chapter 19

Agent Extrinsic Safety: Interaction Risks

 HIS chapter presents a comprehensive examination of extrinsic safety risks, focusing on vulnerabilities that emerge from an AI agent's interactions with its external ecosystem. These threats are categorized into three primary domains: interactions with memory systems, with physical and digital environments, and with other agents. By analyzing these interfaces, the chapter highlights how an agent's security and reliability can be compromised through external manipulation and complex systemic dynamics. The analysis begins by investigating agent-memory interaction threats, with a particular focus on the vulnerabilities within Retrieval-Augmented Generation (RAG) frameworks. We detail how these systems are susceptible to knowledge base poisoning, where adversaries inject carefully crafted malicious documents. These attacks, such as backdoor poisoning, retrieval jamming, and misinformation propagation, are designed to manipulate the agent's retrieval process, forcing it to access and utilize corrupted information, which in turn leads to harmful, biased, or incorrect outputs. Subsequently, the chapter explores agent-environment interaction threats, distinguishing between the physical and digital domains. In the physical world, we examine risks faced by autonomous systems like robots and vehicles, including sensor spoofing (e.g., GPS, LiDAR), actuator manipulation, and the exploitation of environmental conditions to cause misperception and unsafe actions. In the digital realm, the chapter addresses threats to software and web-based agents, such as code injection, data manipulation from compromised sources, denial-of-service (DoS) attacks, and resource exhaustion, all of which can paralyze or hijack agent operations. Finally, the chapter delves into agent-agent interaction threats within multi-agent systems. It dissects risks in both competitive and cooperative scenarios. In competitive interactions, threats include strategic deception, misinformation, algorithmic exploitation, and covert collusion to undermine opponents. In cooperative interactions, vulnerabilities manifest as unintentional information leakage, cascading failures caused by error propagation from a single agent, and systemic weaknesses arising from poor synchronization or a compromised team member. This comprehensive overview underscores that securing AI agents requires robust defenses not only at the core model level but also across every external interaction point.

19.1 Agent-Memory Interaction Threats

 HE 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 [1370] 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 [1535] 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 [1536] 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 [1537] 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 [1538] 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 [1539] 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 [1540] 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.

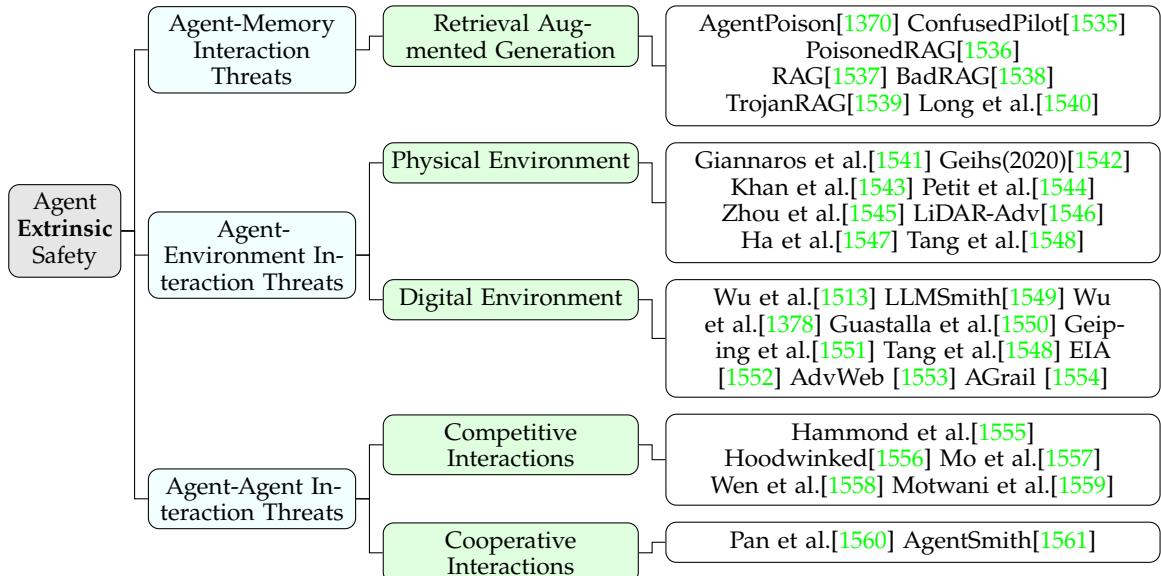


Figure 19.1: Agent Extrinsic Safety: Threats on agent-memory, agent-environment, and agent-agent interactions.

19.2 Agent-Environment Interaction Threats



GENTS 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 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 [1541, 1542, 1548]. 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 [1543]. Similarly, LiDAR spoofing can introduce false obstacles that don't actually exist, potentially leading to navigation failures or safety hazards [1544]. 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 [1545]. 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 [1546]. 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 [1547].

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 [1513, 1548]. One major threat is code injection, where malicious actors introduce harmful code into the agent's environment, leading to unintended command execution [1549]. 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 [1378]. Environmental Injection Attack (EIA) exploits privacy risks in generalist web agents to stealthily steal users' PII, achieving up to 70% success rate [1552]. AdvWeb is an automated adversarial prompt generation framework to mislead black-box web agents into executing harmful actions [1553]. Another critical risk is data manipulation, where attackers alter the information an agent receives, causing incorrect decisions or actions [1513]. 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 [1550]. 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 [1551]. 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 by adapting safety checks to mitigate task-specific and systemic risks, demonstrating robust performance and transferability across diverse tasks [1554].

19.3 Agent-Agent Interaction Threats



N multi-agent systems, interactions between agents can introduce new safety vulnerabilities [1562]. 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 [1555]. For example, they might spread false information or make other agents think the situation is different from reality to deceive them [1556]. 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 [1557]. 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 [1558]. 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 [1559]. 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. [1560] 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 [1561]. 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.

19.4 Summary and Discussion



THE preceding chapters 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 systematically assess these issues, a series of efforts have been made, including Agent-SafetyBench [355], SafeArena [1563], DoomArena [1564], WASP [1565], and RedTeamCUA [1566], to provide evaluation frameworks and benchmarks covering a wide range of risks and failure modes, revealing

significant safety deficiencies in current agents. To mitigate these risks, researchers have developed several methods to enhance agent safety, including AgentMonitor [1567] and more recent WebGuard [1568], by monitoring their decision-making processes and identifying potentially unsafe actions. R-Judge [1569] quantifies an agent's risk awareness by evaluating its responses to both malicious and benign queries, offering a systematic approach to safety compliance. To address the limitations of existing evaluators, AgentAuditor provides a memory-augmented reasoning framework that empowers LLMs to perform human-level safety and security evaluation [1570]. Additionally, risk detection tools like ToolEmu [914] 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. Furthermore, guardrail agents like ShieldAgent are being developed to enforce safety policies through verifiable logical reasoning, offering another layer of protection for autonomous agents [1571]. 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 [1572] 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. Similarly, in the medical domain, the MedSentry benchmark analyzes and mitigates safety risks in multi-agent systems, proposing personality-aware defenses to rehabilitate malicious agents [1573]. Structured task constraints, as implemented in CLAIRify [1574], 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 [1575], 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 20

Superalignment and Safety Scaling Law in AI Agents



As AI systems scale in capability and autonomy, the challenge of ensuring that they remain safe, controllable, and aligned with human values has become increasingly central. Simple safeguards and reward-based tuning no longer suffice in the face of complex reasoning, open-ended objectives, and evolving norms. Addressing these risks requires both deeper alignment frameworks and principled understandings of how safety requirements grow with model power.

This chapter examines two complementary approaches to these challenges: superalignment, a goal-driven paradigm for structuring agent behavior around long-term human objectives; and the safety scaling law, a conceptual framework for understanding how safety interventions must intensify as model capabilities increase. We begin by introducing the principles and mechanisms of superalignment, followed by an analysis of how safety and performance interact under scaling regimes. Together, these perspectives offer foundational insights for building trustworthy AI agents at scale.

20.1 Superalignment: Goal-Driven Alignment for AI Agents



With LLMs increasingly serving as the core of decision-making in autonomous agents, ensuring that their outputs remain safe, ethical, and consistently aligned with human objectives has become a pressing challenge [1576, 508, 1577]. Traditional alignment techniques, particularly RLHF, have been instrumental in refining LLM behavior by incorporating human preferences [135, 70].

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** [1578] 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.

20.1.1 Composite Objective Functions in Superalignment

Superalignment frameworks rely on composite objective functions that break down alignment into distinct components, each targeting a different aspect of agent behavior [1352]. Unlike RLHF, which relies on a single reward model trained from preference comparisons, superalignment explicitly separates reward signals into multiple channels:

- **Task Performance Term:** Focuses on how well the agent completes its immediate operational tasks—e.g., retrieving correct answers, executing planned actions, or achieving subgoals.
- **Goal Adherence Term:** Captures alignment with long-term strategic objectives and user intent. This often involves giving the agent an internal representation of end goals (e.g., the full dialogue purpose, final task outcome) and training it to plan actions that serve that goal [1354, 1579].
- **Norm Compliance Term:** Enforces ethical, legal, and safety constraints. This can be implemented through rule-based filters, constraint-aware training data, or auxiliary models (like rejection classifiers) that detect violations [1580, 1581].

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 [1582, 1583].

20.1.2 Overcoming the Limitations of RLHF with Superalignment

While RLHF has proven effective for aligning models to short-term user preferences, its limitations become increasingly apparent when agents must pursue complex, evolving, or norm-sensitive goals. Superalignment offers a structural alternative that addresses these weaknesses through persistent goal representation, embedded oversight, and adaptive value modeling [1578].

Short-Horizon Feedback Bias RLHF relies on localized preference signals, often ranking isolated completions without considering downstream consequences. This short-term focus leads to agents that optimize for immediate plausibility but fail to maintain coherence in multi-step scenarios. In contrast, superalignment frameworks emphasize persistent, decomposable goal structures. As argued by [1584], alignment must shift from per-instance preference optimization to agent architectures that represent and recursively update internal models of user intent. These internalized goals guide behavior not just at the output level but across entire trajectories, enabling long-horizon reasoning.

Reward Model Fragility and Oversight Bottlenecks RLHF’s centralized reward model can be exploited or misgeneralized due to limited feedback granularity and lack of internal accountability. Superalignment approaches embed scalable oversight directly into the agent’s cognition. SCRIT [1585] introduces a self-evolving natural language critic that monitors outputs during both training and inference, offering contextualized, interpretable feedback on alignment failures. Unlike static reward models, the critic co-evolves with the agent, allowing oversight capacity to grow alongside model complexity.

Static Value Representation RLHF assumes a fixed alignment target, encoding ethical and task priorities at training time. However, real-world deployments demand agents that can update their alignment objectives as user preferences and social norms evolve. Huang et al. [1586] introduces the concept of value plasticity: agents should be trained to reinterpret, recalibrate, and reweight internal objective functions in response to environmental feedback. This capacity for continuous alignment is essential for sustained deployment in dynamic settings.

In summary, superalignment replaces reactive reward shaping with proactive, structured alignment: it builds agents that reason over internal goals, self-monitor their alignment status, and dynamically adapt their value systems over time.

20.1.3 Empirical Evidence Supporting Superalignment

Growing empirical research supports the claim that superalignment-based approaches outperform traditional RLHF methods across long-horizon planning, safety-critical dialogue, and value-sensitive decision-making tasks.

Long-Horizon Planning In environments such as WebShop and ALFWorld, agents must reason over multiple steps while maintaining consistency with the user’s high-level objective. Zhang et al. [1587] evaluated models trained with modular, goal-conditioned objectives and found that they achieved up to 18% higher task success rates and lower rates of goal drift compared to RLHF-tuned baselines. These agents used task-oriented modules for short-term execution and separate goal-adherence modules to validate that subtasks advanced the overall plan. Similar gains were observed in interactive environments like BabyAI and MiniWoB++ [1588], where agents needed to execute long sequences while staying within task constraints.

Safety-Constrained Dialogue In adversarial prompting scenarios, agents trained with modular oversight mechanisms achieved significantly better safety-performance trade-offs. Tang et al. [1585] showed that SCRIT-based agents reduced jailbreaks and unsafe completions by over 30%, outperforming traditional reward-tuned LLMs across both red-teaming and helpful-harmless benchmarks. Unlike RLHF agents that apply a single reward surface, these agents leveraged explicit ethical filters and post hoc self-assessments to revise unsafe completions before output.

Dynamic Adaptation to Shifting Goals Superalignment methods exhibit superior adaptability when task goals or constraints evolve mid-episode. In dynamic goal-following settings, such as resource management games or simulation-based assistants, Shyam et al. [505] showed that agents with dynamic composite weighting adjusted their behavioral priors in response to changing objective functions. This contrasts with RLHF models, which often exhibited stale behavior or overfit to prior preferences.

Auditing and Value Traceability Because superaligned models optimize multiple explicit objectives, they are significantly more interpretable and auditable. In agent safety audits conducted by ShieldAgent [1571], modular agents flagged their own norm violations at a higher precision rate than black-box RLHF agents, improving error diagnosis and recovery. Agents equipped with recursive goal validation [1589] further demonstrated the ability to flag internally inconsistent subgoal policies before execution—an essential property for real-world deployment in sensitive applications like legal reasoning, healthcare, or education.

Generalization and Robustness Empirical studies suggest that superaligned agents generalize better to unseen tasks by virtue of their explicit planning structures. In the AgentBench safety suite [840], composite-trained agents outperformed RLHF agents in 9 out of 12 evaluated environments, including tool-use, multi-agent coordination, and adversarial querying. Notably, gains were largest in tasks requiring both instruction-following and value-sensitive trade-offs.

Together, these empirical results demonstrate that superalignment does more than prevent harm—it actively enhances goal fidelity, interpretability, and robustness in complex, real-world settings.

20.1.4 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 [1577]. 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 [1582].

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 [1590]. 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 [378].

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 [1591]. 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 [1580]. 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 [378]. 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 [1589].

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.

20.2 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 [1592]. As language models grow from millions to trillions

of parameters, their performance follows predictable scaling laws [1593, 1594], but safety assurance exhibits starkly different dynamics [1592]. *Safety Scaling Law* is 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.* [355] 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.* [914] 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.* [1595] 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 [355, 914, 1595] collectively demonstrate that safety scaling requires more than proportional investment, which demands architectural innovations that fundamentally alter the capability-risk relationship. Then, we will review the explorations [1596, 1597, 1598] on how emerging alignment techniques address these challenges.

20.2.1 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.* [355] 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.* [914] explored how helpfulness—defined as a model’s ability to assist users—interacts with safety concerns. Further advancing the discussion, Ying *et al.* [1595] conducted a more detailed comparison and analysis of model safety and performance, leading to the following conclusions: (1) As shown in Figure 20.1 (A) and Figure 20.1 (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 20.1 (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.

20.2.2 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 [1599], and Safe reinforcement learning from human feedback (Safe RLHF) [70]. 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 [136], IPO [1600], etc.), this alignment process fine-tunes the model to produce responses that meet safety standards. As reported in [1596], various preference optimization methods are investigated for safety enhancement, including Safe-DPO [136], Safe-robust-DPO [1601], Safe-IPO [1600], Safe-SLiC [1602], Safe-KTO [497], and Safe-NCA [1597], 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) [1597] is identified as an optimal approach for balancing safety with overall model performance. The core of the Safe-NCA [1597] 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.* [1598] 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:

$$[\text{helpful} = s_{hp}][\text{harmless} = s_{sf}]. \quad (20.1)$$

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.

20.2.3 Future directions and strategies: the AI-45 degree 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.* [1603] 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

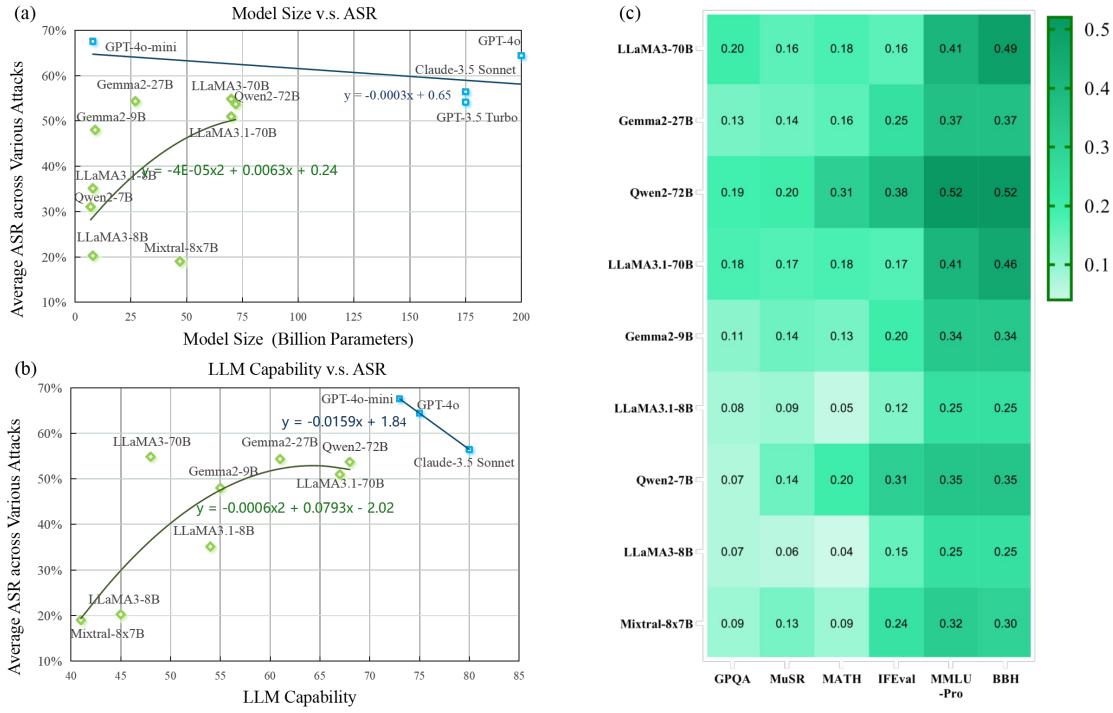


Figure 20.1: **Performance and safety analysis of LLMs.** (a) The relationship between LLM model size and their average ASR across various attacks. The data are sourced from experimental results of a study assessing the robustness of LLMs against adversarial attacks [355]. (b) The relationship between 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 [1604]. (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 [1605].

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.

20.3 Summary and Discussion

 HIS chapter presented two complementary perspectives on advancing the safety of increasingly capable AI agents: the structural approach of superalignment and the empirical insights from the emerging safety scaling law. As language models and autonomous agents continue to scale in complexity and application scope, traditional alignment methods like RLHF have proven insufficient in maintaining long-term, goal-consistent, and value-aligned behavior.

Superalignment offers a principled framework for aligning agents with human objectives through structured goal representation, modular reward decomposition, and persistent oversight. Unlike single-signal reward models, superaligned agents optimize for a composite objective function that balances task performance, ethical compliance, and strategic coherence. Empirical results demonstrate that such agents are more robust, interpretable, and adaptable in dynamic environments, especially when pursuing multi-step tasks with evolving goals [1587, 1585, 505].

Meanwhile, the concept of the safety scaling law underscores the disproportionate growth in risk as models become more powerful. Quantitative studies have shown that capability improvements often outpace safety enhancements, with larger models exhibiting greater vulnerability surfaces and higher likelihoods of unsafe behavior [355, 914]. Novel methods such as Safe-NCA and control-token conditioning have been proposed to address this imbalance, enabling more flexible and controllable trade-offs between helpfulness and harmlessness [1597, 1598].

Together, these two threads reveal a crucial insight: proactive, structured alignment and proportionally scaled safety interventions are both essential for the development of trustworthy AI systems. Superalignment provides the architectural and optimization foundations for value-consistent reasoning, while safety scaling laws offer empirical guidance on risk-growth patterns and mitigation requirements.

Moving forward, a key research challenge lies in bridging these paradigms. Integrating superalignment principles into safety scaling protocols can help ensure that safety mechanisms evolve alongside model capabilities. Conversely, insights from safety scaling benchmarks can inform how to prioritize and adapt superalignment objectives under different deployment contexts. Achieving this integration will be critical for building AI agents that are not only powerful and capable—but also safe, aligned, and reliable over extended time horizons.

Chapter 21

Concluding Remarks and Future Outlook

We have explored in this book 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 work 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 [1606]. 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*. Prentice Hall, Englewood Cliffs, NJ, 1 edition, 1995. ISBN 0-13-103805-2.
- [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] Dale Purves, George J Augustine, David Fitzpatrick, William Hall, Anthony-Samuel LaMantia, and Leonard White. *Neurosciences*. De Boeck Supérieur, 2019.
- [15] Wikipedia contributors. Phineas Gage. https://en.wikipedia.org/wiki/Phineas_Gage, 2025. Accessed: June 8, 2025.
- [16] Noga Larry, Gil Zur, and Mati Joshua. Organization of reward and movement signals in the basal ganglia and cerebellum. *Nature Communications*, 15(1):2119, 2024.

- [17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pages 5998–6008. Curran Associates, Inc., 2017.
- [18] Meng-Hao Guo, Tian-Xing Xu, Jiang-Jiang Liu, Zheng-Ning Liu, Peng-Tao Jiang, Tai-Jiang Mu, Song-Hai Zhang, Ralph R Martin, Ming-Ming Cheng, and Shi-Min Hu. Attention mechanisms in computer vision: A survey. *Computational visual media*, 8(3):331–368, 2022.
- [19] Maurizio Corbetta and Gordon L Shulman. Control of goal-directed and stimulus-driven attention in the brain. *Nature reviews neuroscience*, 3(3):201–215, 2002.
- [20] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.
- [21] Wikipedia contributors. Wernicke's area. https://en.wikipedia.org/wiki/Wernicke%27s_area, 2025. Accessed: June 8, 2025.
- [22] Wikipedia contributors. Cocktail party effect. https://en.wikipedia.org/wiki/Cocktail_party_effect, 2025. Accessed: June 8, 2025.
- [23] James Knierim. Chapter 5: Cerebellum. <https://nba.uth.tmc.edu/neuroscience/m/s3/chapter05.html>, 2020. Last review: October 20, 2020; Accessed: June 8, 2025.
- [24] Torgeir Moberget and Richard B Ivry. Cerebellar contributions to motor control and language comprehension: searching for common computational principles. *Annals of the New York Academy of Sciences*, 1369(1):154–171, 2016.
- [25] Kenji Doya. What are the computations of the cerebellum, the basal ganglia and the cerebral cortex? *Neural networks*, 12(7-8):961–974, 1999.
- [26] Cleveland Clinic. Brainstem: What It Is, Function, Anatomy & Location. <https://my.clevelandclinic.org/health/body/21598-brainstem>, 2024. Last reviewed: June 12, 2024; Accessed: June 8, 2025.
- [27] Cleveland Clinic. Thalamus: What It Is, Function & Disorders. <https://my.clevelandclinic.org/health/body/22652-thalamus>, 2022. Last reviewed: March 30, 2022; Accessed: June 8, 2025.
- [28] Ralf D Wimmer, L Ian Schmitt, Thomas J Davidson, Miho Nakajima, Karl Deisseroth, and Michael M Halassa. Thalamic control of sensory selection in divided attention. *Nature*, 526(7575):705–709, 2015.
- [29] Wikipedia contributors. Limbic system. https://en.wikipedia.org/wiki/Limbic_system, 2025. Accessed: June 8, 2025.
- [30] Jose G. Sanchez Jimenez and Orlando De Jesus. Hypothalamic Dysfunction. *StatPearls [Internet]*. Treasure Island (FL): StatPearls Publishing; 2025 Jan–, 2023. URL <https://www.ncbi.nlm.nih.gov/books/NBK560743/>. Last update: August 23, 2023; Accessed: June 8, 2025.
- [31] Yoshua Bengio, Michael Cohen, Damiano Fornasiere, Joumana Ghosn, Pietro Greiner, Matt MacDer- mott, Sören Mindermann, Adam Oberman, Jesse Richardson, Oliver Richardson, et al. Superintelligent agents pose catastrophic risks: Can scientist ai offer a safer path? *arXiv preprint arXiv:2502.15657*, 2025.
- [32] OpenAI. Chatgpt (gpt-4).
url<https://chat.openai.com/>, 2023. Large language model.
- [33] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.
- [34] Theodore Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas Griffiths. Cognitive architectures for language agents. *Transactions on Machine Learning Research*, 2024.

- [35] 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.
- [36] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.
- [37] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.
- [38] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.
- [39] Pattie Maes. Artificial life meets entertainment: lifelike autonomous agents. *Communications of the ACM*, 38(11):108–114, 1995.
- [40] Stan Franklin and Art Graesser. Is it an agent, or just a program?: A taxonomy for autonomous agents. In *International workshop on agent theories, architectures, and languages*, pages 21–35. Springer, 1997.
- [41] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.
- [42] Bostrom Nick. Superintelligence: Paths, dangers, strategies, 2014.
- [43] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.
- [44] Marvin Minsky. *Society of mind*. Simon and Schuster, 1988.
- [45] Gyorgy Buzsaki. *The brain from inside out*. Oxford University Press, USA, 2019.
- [46] Karl J Friston, Jean Daunizeau, James Kilner, and Stefan J Kiebel. Action and behavior: a free-energy formulation. *Biological cybernetics*, 102:227–260, 2010.
- [47] Xiaoqian Liu, Xingzhou Lou, Jianbin Jiao, and Junge Zhang. Position: Foundation agents as the paradigm shift for decision making. *arXiv preprint arXiv:2405.17009*, 2024.
- [48] Larry R Squire. Memory and the hippocampus: a synthesis from findings with rats, monkeys, and humans. *Psychological review*, 99(2):195, 1992.
- [49] Mark Bear, Barry Connors, and Michael A Paradiso. *Neuroscience: exploring the brain, enhanced edition: exploring the brain*. Jones & Bartlett Learning, 2020.
- [50] 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.
- [51] Joseph E LeDoux. *The emotional brain: The mysterious underpinnings of emotional life*. Simon and Schuster, 1998.
- [52] Antonio R. Damasio. *Descartes' Error: Emotion, Reason, and the Human Brain*. Putnam, 1994.
- [53] Earl K Miller and Jonathan D Cohen. An integrative theory of prefrontal cortex function. *Annual review of neuroscience*, 24(1):167–202, 2001.
- [54] David Badre. Cognitive control, hierarchy, and the rostro-caudal organization of the frontal lobes. *Trends in cognitive sciences*, 12(5):193–200, 2008.
- [55] Wolfram Schultz, Peter Dayan, and P Read Montague. A neural substrate of prediction and reward. *Science*, 275(5306):1593–1599, 1997.
- [56] Joaquin M Fuster. *The Prefrontal Cortex*. Academic Press, 4th edition, 2008.
- [57] Tim Shallice and Richard P Cooper. The organisation of mind. *Oxford Psychology Series*, 32, 2011.

- [58] 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.
- [59] 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.
- [60] 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>.
- [61] 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.
- [62] 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>.
- [63] 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>.
- [64] 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.
- [65] 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](https://doi.org/10.24963/ijcai.2024/890). URL <https://doi.org/10.24963/ijcai.2024/890>. Survey Track.
- [66] 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.
- [67] 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.
- [68] 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.
- [69] 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.
- [70] 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.

- [71] 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>.
- [72] 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>.
- [73] 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.
- [74] 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.
- [75] 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>.
- [76] 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.
- [77] 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.
- [78] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023.
- [79] 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.
- [80] 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.
- [81] Huatong Song, Jinhao Jiang, Wenqing Tian, Zhipeng Chen, Yuhuan Wu, Jiahao Zhao, Yingqian Min, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher++: Incentivizing the dynamic knowledge acquisition of llms via reinforcement learning. *arXiv preprint arXiv:2505.17005*, 2025.
- [82] 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>.
- [83] NovaSky Team. Sky-t1: Train your own o1 preview model within \$450, 2025.
- [84] Open Thoughts Team. Open Thoughts, January 2025.
- [85] 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.
- [86] Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning, 2022. URL <https://arxiv.org/abs/2203.14465>.

- [87] 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>.
- [88] 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.
- [89] 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.
- [90] 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>.
- [91] Hugging Face. Open-r1, 2024. URL <https://github.com/huggingface/open-r1>.
- [92] 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.
- [93] 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.
- [94] 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.
- [95] 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.
- [96] 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.
- [97] 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.
- [98] Wen Yang, Minpeng Liao, and Kai Fan. Markov chain of thought for efficient mathematical reasoning. *arXiv preprint arXiv:2410.17635*, 2024.
- [99] 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.
- [100] 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.
- [101] 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.
- [102] Maciej Besta, Nils Blach, Aleš Kubíček, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michał Podstawski, Hubert Niewiadomski, Piotr Nyczek, 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>.
- [103] 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.
 - [104] 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>.
 - [105] 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.
 - [106] 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.
 - [107] 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.
 - [108] 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.
 - [109] 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.
 - [110] 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.
 - [111] 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.
 - [112] 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.
 - [113] 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.
 - [114] 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.
 - [115] Lishui Fan, Mouxiang Chen, and Zhongxin Liu. Self-explained keywords empower large language models for code generation. *arXiv preprint arXiv:2410.15966*, 2024.
 - [116] 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.
 - [117] 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.

- [118] 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.
- [119] 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.
- [120] 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.
- [121] 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.
- [122] 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.
- [123] 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.
- [124] 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.
- [125] Jerry A Fodor. *The modularity of mind*. MIT press, 1983.
- [126] Kenji Doya. Complementary roles of basal ganglia and cerebellum in learning and motor control. *Current opinion in neurobiology*, 10(6):732–739, 2000.
- [127] Hongjin Su, Ruoxi Sun, Jinsung Yoon, Pengcheng Yin, Tao Yu, and Sercan Ö Arik. Learn-by-interact: A data-centric framework for self-adaptive agents in realistic environments. *arXiv preprint arXiv:2501.10893*, 2025.
- [128] 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.
- [129] 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>.
- [130] 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.
- [131] 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>.
- [132] 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.
- [133] 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.

- [134] 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.
- [135] 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.
- [136] 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.
- [137] 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.
- [138] Kimi Team. Kimi k1.5: Scaling reinforcement learning with llms, 2025. URL <https://arxiv.org/abs/2501.12599>.
- [139] Shuang Sun, Huatong Song, Yuhao Wang, Ruiyang Ren, Jinhao Jiang, Junjie Zhang, Fei Bai, Jia Deng, Wayne Xin Zhao, Zheng Liu, et al. Simpledeepsearcher: Deep information seeking via web-powered reasoning trajectory synthesis. *arXiv preprint arXiv:2505.16834*, 2025.
- [140] Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Wenyi Zhao, Yu Yang, Xinyue Yang, Jiadai Sun, Shuntian Yao, Tianjie Zhang, Wei Xu, Jie Tang, and Yuxiao Dong. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning, 2025. URL <https://arxiv.org/abs/2411.02337>.
- [141] Shiyi Cao, Sumanth Hegde, Dacheng Li, Tyler Griggs, Shu Liu, Eric Tang, Jiayi Pan, Xingyao Wang, Akshay Malik, Graham Neubig, Kurosh Hakhamaneshi, Richard Liaw, Philipp Moritz, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Skyrl-v0: Train real-world long-horizon agents via reinforcement learning, 2025.
- [142] Zhixun Chen, Ming Li, Yuxuan Huang, Yali Du, Meng Fang, and Tianyi Zhou. Atlas: Agent tuning via learning critical steps, 2025. URL <https://arxiv.org/abs/2503.02197>.
- [143] Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.
- [144] Shuofei Qiao, Zhisong Qiu, Baochang Ren, Xiaobin Wang, Xiangyuan Ru, Ningyu Zhang, Xiang Chen, Yong Jiang, Pengjun Xie, Fei Huang, et al. Agentic knowledgeable self-awareness. *arXiv preprint arXiv:2504.03553*, 2025.
- [145] 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.
- [146] 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.
- [147] 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.
- [148] Kaisi Guan, Qian Cao, Yuchong Sun, Xiting Wang, and Ruihua Song. Bsharedrag: Backbone shared retrieval-augmented generation for the e-commerce domain. *arXiv preprint arXiv:2409.20075*, 2024.

- [149] Pengcheng Jiang, Jiacheng Lin, Lang Cao, Runchu Tian, SeongKu Kang, Zifeng Wang, Jimeng Sun, and Jiawei Han. Deepretrieval: Hacking real search engines and retrievers with large language models via reinforcement learning, 2025. URL <https://arxiv.org/abs/2503.00223>.
- [150] 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>.
- [151] Qwen Team. Qwq: Reflect deeply on the boundaries of the unknown, November 2024. URL <https://qwenlm.github.io/blog/qwq-32b-preview/>.
- [152] Xinhao Yao, Ruirong Ren, Yun Liao, and Yong Liu. Unveiling the mechanisms of explicit cot training: How chain-of-thought enhances reasoning generalization. *arXiv preprint arXiv:2502.04667*, 2025.
- [153] 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.
- [154] 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>.
- [155] Xinyu Guan, Li Lyra 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.
- [156] 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, Kshitij 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>.
- [157] 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.
- [158] 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.
- [159] 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.
- [160] Sheikh Shafayat, Fahim Tajwar, Ruslan Salakhutdinov, Jeff Schneider, and Andrea Zanette. Can large reasoning models self-train? *arXiv preprint arXiv:2505.21444*, 2025.
- [161] Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data. *arXiv preprint arXiv:2505.03335*, 2025.
- [162] Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Liyuan Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, et al. Reinforcement learning for reasoning in large language models with one training example. *arXiv preprint arXiv:2504.20571*, 2025.
- [163] Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, Yulia

- Tsvetkov, Hannaneh Hajishirzi, Pang Wei Koh, and Luke Zettlemoyer. Spurious rewards: Rethinking training signals in rlvr. <https://rethink-rlvr.notion.site/Spurious-Rewards-Rethinking-Training-Signals-in-RLVR-1f4df34dac1880948858f95aeb88872f>, 2025. Notion Blog.
- [164] Xiangxin Zhou, Zichen Liu, Anya Sims, Haonan Wang, Tianyu Pang, Chongxuan Li, Liang Wang, Min Lin, and Chao Du. Reinforcing general reasoning without verifiers. *arXiv preprint arXiv:2505.21493*, 2025.
- [165] Junwei Liao, Muning Wen, Jun Wang, and Weinan Zhang. Marft: Multi-agent reinforcement fine-tuning, 2025. URL <https://arxiv.org/abs/2504.16129>.
- [166] Ziyu Wan, Yunxiang Li, Xiaoyu Wen, Yan Song, Hanjing Wang, Linyi Yang, Mark Schmidt, Jun Wang, Weinan Zhang, Shuyue Hu, and Ying Wen. Rema: Learning to meta-think for llms with multi-agent reinforcement learning, 2025. URL <https://arxiv.org/abs/2503.09501>.
- [167] Hanlin Wang, Chak Tou Leong, Jiashuo Wang, Jian Wang, and Wenjie Li. Spa-rl: Reinforcing llm agents via stepwise progress attribution, 2025. URL <https://arxiv.org/abs/2505.20732>.
- [168] Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training, 2025. URL <https://arxiv.org/abs/2505.10978>.
- [169] Jiahao Qiu, Xuan Qi, Tongcheng Zhang, Xinzhe Juan, Jiacheng Guo, Yifu Lu, Yimin Wang, Zixin Yao, Qihan Ren, Xun Jiang, et al. Alita: Generalist agent enabling scalable agentic reasoning with minimal predefinition and maximal self-evolution. *arXiv preprint arXiv:2505.20286*, 2025.
- [170] Alexander Novikov, Ng  n Vu, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco JR Ruiz, Abbas Mehrabian, et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *Google DeepMind*, 2025.
- [171] Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin godel machine: Open-ended evolution of self-improving agents. *arXiv preprint arXiv:2505.22954*, 2025.
- [172] Junhong Shen, Hao Bai, Lunjun Zhang, Yifei Zhou, Amrit Sethuraman, Shengbang Tong, Diego Caples, Nan Jiang, Tong Zhang, Ameet Talwalkar, and Aviral Kumar. Thinking vs. doing: Agents that reason by scaling test-time interaction, 2025. URL <https://arxiv.org/abs/2506.07976>.
- [173] 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.
- [174] 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.
- [175] 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.
- [176] 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.
- [177] Brett K Hayes, Evan Heit, and Haruka Swendsen. Inductive reasoning. *Wiley interdisciplinary reviews: Cognitive science*, 1(2):278–292, 2010.
- [178] Fengwei Teng, Zhaoyang Yu, Quan Shi, Jiayi Zhang, Chenglin Wu, and Yuyu Luo. Atom of thoughts for markov llm test-time scaling. *arXiv preprint arXiv:2502.12018*, 2025.

- [179] Shenao Zhang, Yaqing Wang, Yinxiao Liu, Tianqi Liu, Peter Grabowski, Eugene Ie, Zhaoran Wang, and Yunxuan Li. Beyond markovian: Reflective exploration via bayes-adaptive rl for llm reasoning. *arXiv preprint arXiv:2505.20561*, 2025.
- [180] 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.
- [181] 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>.
- [182] 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.
- [183] 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.
- [184] 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.
- [185] 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.
- [186] 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.
- [187] 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.
- [188] 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.
- [189] 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.
- [190] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025. URL <https://arxiv.org/abs/2501.19393>.
- [191] Yifan Wu, Jingze Shi, Bingheng Wu, Jiayi Zhang, Xiaotian Lin, Nan Tang, and Yuyu Luo. Concise reasoning, big gains: Pruning long reasoning trace with difficulty-aware prompting. *arXiv preprint arXiv:2505.19716*, 2025.
- [192] Xiaoqiang Wang, Suyuchen Wang, Yun Zhu, and Bang Liu. System-1.5 reasoning: Traversal in language and latent spaces with dynamic shortcuts. *arXiv preprint arXiv:2505.18962*, 2025.
- [193] Tianhe Lin, Jian Xie, Siyu Yuan, and Deqing Yang. Implicit reasoning in transformers is reasoning through shortcuts. *arXiv preprint arXiv:2503.07604*, 2025.
- [194] Allen Newell, John Calman Shaw, and Herbert A Simon. Elements of a theory of human problem solving. *Psychological review*, 65(3):151, 1958.
- [195] Subbarao Kambhampati. CSE 574: Planning and Learning. <https://rakaposhi.eas.asu.edu/cse574/index-f04.html>, 2004.

- [196] 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.
- [197] 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.
- [198] Subbarao Kambhampati. Can large language models reason and plan? *Annals of the New York Academy of Sciences*, 1534(1):15–18, 2024.
- [199] 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.
- [200] Vishal Pallagani, Bharath Muppaneni, 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.
- [201] 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.
- [202] Kaisi Guan, Zhengfeng Lai, Yuchong Sun, Peng Zhang, Wei Liu, Kieran Liu, Meng Cao, and Ruihua Song. Etva: Evaluation of text-to-video alignment via fine-grained question generation and answering. *arXiv preprint arXiv:2503.16867*, 2025.
- [203] 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.
- [204] 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.
- [205] 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.
- [206] 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.
- [207] 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.
- [208] 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.
- [209] 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.
- [210] Siheng Xiong, Ali Payani, Ramana Kompella, and Faramarz Fekri. Large language models can learn temporal reasoning. *CoRR*, 2024.
- [211] 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.

- [212] Owen Burns, Dana Hughes, and Katia Sycara. Plancritic: Formal planning with human feedback. *arXiv preprint arXiv:2412.00300*, 2024.
- [213] 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.
- [214] 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.
- [215] 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.
- [216] 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.
- [217] 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.
- [218] 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.
- [219] 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.
- [220] 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.
- [221] Yann LeCun. Deep learning and the path to artificial general intelligence. Future of Life Institute, <https://futureoflife.org/wp-content/uploads/2017/01/Yann-LeCun.pdf>, 2017. Accessed: 2025-06-24.
- [222] 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.
- [223] Richard C Atkinson. Human memory: A proposed system and its control processes. *The psychology of learning and motivation*, 2, 1968.
- [224] 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.
- [225] Alan Baddeley. Working memory. *Science*, 255(5044):556–559, 1992.
- [226] George Sperling. The information available in brief visual presentations. *Psychological monographs: General and applied*, 74(11):1, 1960.
- [227] Max Coltheart. Iconic memory and visible persistence. *Perception & psychophysics*, 27:183–228, 1980.
- [228] JM Gardiner. On recency and echoic memory. *Philosophical Transactions of the Royal Society of London. B, Biological Sciences*, 302(1110):267–282, 1983.
- [229] Bart Aben, Sven Stapert, and Arjan Blokland. About the distinction between working memory and short-term memory. *Frontiers in psychology*, 3:301, 2012.
- [230] Nelson Cowan. What are the differences between long-term, short-term, and working memory? *Progress in brain research*, 169:323–338, 2008.
- [231] 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.

- [232] Richard M Shiffrin and Richard C Atkinson. Storage and retrieval processes in long-term memory. *Psychological review*, 76(2):179, 1969.
- [233] Dennis Norris. Short-term memory and long-term memory are still different. *Psychological bulletin*, 143(9):992, 2017.
- [234] Hermann Ebbinghaus. Memory: A contribution to experimental psychology. *Annals of neurosciences*, 20(4):155, 2013.
- [235] Howard Eichenbaum. Declarative memory: Insights from cognitive neurobiology. *Annual review of psychology*, 48(1):547–572, 1997.
- [236] Abhilasha A Kumar. Semantic memory: A review of methods, models, and current challenges. *Psychonomic bulletin & review*, 28(1):40–80, 2021.
- [237] Endel Tulving. Episodic memory: From mind to brain. *Annual review of psychology*, 53(1):1–25, 2002.
- [238] Robyn Fivush. The development of autobiographical memory. *Annual review of psychology*, 62(1):559–582, 2011.
- [239] Larry R Squire. Declarative and nondeclarative memory: Multiple brain systems supporting learning and memory. *Journal of cognitive neuroscience*, 4(3):232–243, 1992.
- [240] Prahlad Gupta and Neal J Cohen. Theoretical and computational analysis of skill learning, repetition priming, and procedural memory. *Psychological review*, 109(2):401, 2002.
- [241] 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.
- [242] Endel Tulving and Daniel L Schacter. Priming and human memory systems. *Science*, 247(4940):301–306, 1990.
- [243] 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.
- [244] Androulla Ioannou and Xenia Anastassiou-Hadjicharalambous. Non-associative learning. *Encyclopedia of evolutionary psychological science*, pages 5419–5432, 2021.
- [245] Martin A Conway and Christopher W Pleydell-Pearce. The construction of autobiographical memories in the self-memory system. *Psychological review*, 107(2):261, 2000.
- [246] Alan D Baddeley, Graham Hitch, and Gordon H Bower. Working memory. volume 8 of. *Psychology of Learning and Motivation*, pages 47–89, 1974.
- [247] Alan Baddeley. The episodic buffer: a new component of working memory? *Trends in cognitive sciences*, 4(11):417–423, 2000.
- [248] 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.
- [249] Endel Tulving. Memory and consciousness. *Canadian Psychology/Psychologie canadienne*, 26(1):1, 1985.
- [250] Bernard J Baars. *A cognitive theory of consciousness*. Cambridge University Press, 1993.
- [251] Stan Franklin. *Artificial minds*. MIT press, 1997.
- [252] 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.
- [253] John R Anderson. *How can the human mind occur in the physical universe?* Oxford University Press, 2009.
- [254] 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.

- [255] Alan Baddeley. Working memory. *Current biology*, 20(4):R136–R140, 2010.
- [256] 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.
- [257] 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.
- [258] 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.
- [259] 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.
- [260] 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.
- [261] 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.
- [262] 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.
- [263] Yue Fan, Xiaojian Ma, Ruijie 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.
- [264] 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.
- [265] 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.
- [266] 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.
- [267] Sen Li, Ruochen Wang, Cho-Jui Hsieh, Minhao Cheng, and Tianyi Zhou. Mulan: Multimodal-lm agent for progressive multi-object diffusion. *arXiv preprint arXiv:2402.12741*, 2024.
- [268] 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.
- [269] 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.
- [270] 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.
- [271] Xiaoqiang Wang and Bang Liu. Oscar: Operating system control via state-aware reasoning and re-planning. *arXiv preprint arXiv:2410.18963*, 2024.
- [272] Kevin A Fischer. Reflective linguistic programming (rlp): A stepping stone in socially-aware agi (socialagi). *arXiv preprint arXiv:2305.12647*, 2023.

- [273] 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.
- [274] 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.
- [275] 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.
- [276] 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.
- [277] 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.
- [278] 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.
- [279] 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.
- [280] 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.
- [281] 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.
- [282] 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.
- [283] 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.
- [284] 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.
- [285] 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.
- [286] 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.
- [287] Pengbo Hu and Xiang Ying. Unified mind model: Reimagining autonomous agents in the llm era. *arXiv preprint arXiv:2503.03459*, 2025.
- [288] 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.
- [289] 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.

- [290] 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.
- [291] Hang Gao and Yongfeng Zhang. Memory sharing for large language model based agents. *arXiv preprint arXiv:2404.09982*, 2024.
- [292] Meng Chu, Yicong Li, and Tat-Seng Chua. Understanding long videos via llm-powered entity relation graphs. *arXiv preprint arXiv:2501.15953*, 2025.
- [293] 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.
- [294] 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.
- [295] 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.
- [296] 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.
- [297] Tenghao Huang, Kinjal Basu, Ibrahim Abdelaziz, Pavan Kapanipathi, Jonathan May, and Muhaoo Chen. R2d2: Remembering, reflecting and dynamic decision making for web agents. *arXiv preprint arXiv:2501.12485*, 2025.
- [298] 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.
- [299] 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.
- [300] 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.
- [301] 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.
- [302] 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.
- [303] 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.
- [304] Yudi Shi, Shangzhe Di, Qirui Chen, and Weidi Xie. Unlocking video-llm via agent-of-thoughts distillation. *arXiv preprint arXiv:2412.01694*, 2024.
- [305] 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.

- [306] 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.
- [307] 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.
- [308] 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.
- [309] Chen Gao, Xiaochong Lan, Zhihong Lu, Jinzhu Mao, Jinghua Piao, Huandong Wang, Depeng Jin, and Yong Li. S³: Social-network simulation system with large language model-empowered agents. *arXiv preprint arXiv:2307.14984*, 2023.
- [310] 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.
- [311] 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.
- [312] 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.
- [313] 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.
- [314] 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.
- [315] Dzmitry Bahdanau. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
- [316] 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.
- [317] 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.
- [318] 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.
- [319] 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.
- [320] 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.
- [321] 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.

- [322] 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.
- [323] Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time. *arXiv preprint arXiv:2501.00663*, 2024.
- [324] Xiaoqiang Wang, Suyuchen Wang, Yun Zhu, and Bang Liu. R³mem: Bridging memory retention and retrieval via reversible compression. *arXiv preprint arXiv:2502.15957*, 2025.
- [325] 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.
- [326] 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.
- [327] 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.
- [328] 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.
- [329] Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. Recurrent memory transformer. *Advances in Neural Information Processing Systems*, 35:11079–11091, 2022.
- [330] 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.
- [331] 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.
- [332] 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.
- [333] Jesse Mu, Xiang Li, and Noah Goodman. Learning to compress prompts with gist tokens. *Advances in Neural Information Processing Systems*, 36, 2024.
- [334] Chanwoong Yoon, Taewho 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.
- [335] 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.
- [336] Sangjun Park and JinYeong Bak. Memoria: Resolving fateful forgetting problem through human-inspired memory architecture. *arXiv preprint arXiv:2310.03052*, 2023.
- [337] Xu Owen He. Mixture of a million experts. *arXiv preprint arXiv:2407.04153*, 2024.
- [338] 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.
- [339] Yingxu Wang, Dong Liu, and Ying Wang. Discovering the capacity of human memory. *Brain and Mind*, 4:189–198, 2003.

- [340] 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.
- [341] 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.
- [342] 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.
- [343] 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.
- [344] Rebecca MM Hicke, Sil Hamilton, and David Mimno. The zero body problem: Probing llm use of sensory language. *arXiv preprint arXiv:2504.06393*, 2025.
- [345] Iñaki Dellibarda Varela, Pablo Romero-Sorozabal, Diego Torricelli, Gabriel Delgado-Oleas, Jose Ignacio Serrano, Maria Dolores del Castillo Sobrino, Eduardo Rocon, and Manuel Cebrian. Sensorimotor features of self-awareness in multimodal large language models. *arXiv preprint arXiv:2505.19237*, 2025.
- [346] 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.
- [347] Longtao Zheng, Rundong Wang, and Bo An. Synapse: Leveraging few-shot exemplars for human-level computer control. *arXiv preprint arXiv:2306.07863*, 2023.
- [348] 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.
- [349] 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.
- [350] 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.
- [351] 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.
- [352] 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.
- [353] 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.
- [354] 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.
- [355] Wenqi Zhang, Ke Tang, Hai Wu, Mengna Wang, Yongliang Shen, Guiyang Hou, Zeqi Tan, Peng Li, Yueling Zhuang, and Weiming Lu. Agent-pro: Learning to evolve via policy-level reflection and optimization. *arXiv preprint arXiv:2402.17574*, 2024.

- [356] Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. Gpt-4v (ision) is a generalist web agent, if grounded. In *ICML*, 2024.
- [357] Hanyu Lai, Xiao Liu, Iat Long Long, 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.
- [358] Paloma Sodhi, SRK Branavan, and Ryan McDonald. Heap: Hierarchical policies for web actions using LLMs. *arXiv preprint arXiv:2310.03720*, 2023.
- [359] Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. *arXiv preprint arXiv:2409.07429*, 2024.
- [360] 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.
- [361] 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.
- [362] 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.
- [363] 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.
- [364] 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.
- [365] Grace W Lindsay. Attention in psychology, neuroscience, and machine learning. *Frontiers in computational neuroscience*, 14:29, 2020.
- [366] 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.
- [367] 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.
- [368] 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.
- [369] Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan Kian Hsiang Low, and Paul Pu Liang. Mem1: Learning to synergize memory and reasoning for efficient long-horizon agents. *arXiv preprint arXiv:2506.15841*, 2025.
- [370] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
- [371] Larry R Squire, Lisa Genzel, John T Wixted, and Richard G Morris. Memory consolidation. *Cold Spring Harbor perspectives in biology*, 7(8):a021766, 2015.

- [372] 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.
- [373] N Reimers. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.
- [374] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.
- [375] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.
- [376] 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.
- [377] 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.
- [378] 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.
- [379] Bart Kosko. Bidirectional associative memories. *IEEE Transactions on Systems, man, and Cybernetics*, 18(1):49–60, 1988.
- [380] 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.
- [381] Zihang Dai. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.
- [382] 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.
- [383] 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.
- [384] 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.
- [385] Yu Wang, Ruihan Wu, Zexue He, Xiusi Chen, and Julian McAuley. Large scale knowledge washing. *arXiv preprint arXiv:2405.16720*, 2024.
- [386] 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.
- [387] 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.
- [388] 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>.

- [389] 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.
- [390] 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.
- [391] Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic cheatsheet: Test-time learning with adaptive memory. *arXiv preprint arXiv:2504.07952*, 2025.
- [392] Zhiyu Li, Shichao Song, Hanyu Wang, Simin Niu, Ding Chen, Jiawei Yang, Chenyang Xi, Huayi Lai, Jihao Zhao, Yezhaohui Wang, et al. Memos: An operating system for memory-augmented generation (mag) in large language models. *arXiv preprint arXiv:2505.22101*, 2025.
- [393] 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.
- [394] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.
- [395] 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.
- [396] 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.
- [397] Huanxuan Liao, Wen Hu, Yao Xu, Shizhu He, Jun Zhao, and Kang Liu. Beyond hard and soft: Hybrid context compression for balancing local and global information retention. *arXiv preprint arXiv:2505.15774*, 2025.
- [398] 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.
- [399] 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.
- [400] Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630, 2024.
- [401] Edward C Tolman. Cognitive maps in rats and men. *Psychological review*, 55(4):189, 1948.
- [402] Kenneth James Williams Craik. *The nature of explanation*, volume 445. CUP Archive, 1967.
- [403] Dedre Gentner and Albert L Stevens. *Mental models*. Psychology Press, 2014.
- [404] Andy Clark. *Surfing uncertainty: Prediction, action, and the embodied mind*. Oxford University Press, 2015.
- [405] Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.

- [406] 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.
- [407] 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.
- [408] 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.
- [409] Danijar Hafner, Timothy P. Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, 2020.
- [410] 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.
- [411] 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.
- [412] Eloi Alonso, Adam Jolley, Vincent Micheli, Anssi Kanervisto, Amos J. Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.
- [413] 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.
- [414] 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.
- [415] Chenxi Liu, Yongqiang Chen, Tongliang Liu, Mingming Gong, James Cheng, Bo Han, and Kun Zhang. Discovery of the hidden world with large language models. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.
- [416] 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.
- [417] 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.
- [418] Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.
- [419] J Parker-Holder, P Ball, J Bruce, V Dasagi, K Holsheimer, C Kaplanis, A Moufarek, G Scully, J Shar, J Shi, et al. Genie 2: A large-scale foundation world model. URL: <https://deepmind.google/discover/blog/genie-2-a-large-scale-foundation-world-model>, 2024.

- [420] Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, Dit-Yan Yeung, and Chuang Gan. Robodreamer: Learning compositional world models for robot imagination. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, 2024.
- [421] Scott E. Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent. *Trans. Mach. Learn. Res.*, 2022, 2022.
- [422] Zihan Ding, Amy Zhang, Yuandong Tian, and Qinqing Zheng. Diffusion world model: Future modeling beyond step-by-step rollout for offline reinforcement learning. *arXiv preprint arXiv:2402.03570*, 2024.
- [423] Marc Rigter, Tarun Gupta, Agrin Hilmkil, and Chao Ma. Avid: Adapting video diffusion models to world models. *arXiv preprint arXiv:2410.12822*, 2024.
- [424] Tianyuan Zhang, Hong-Xing Yu, Rundi Wu, Brandon Y Feng, Changxi Zheng, Noah Snavely, Jiajun Wu, and William T Freeman. Physdreamer: Physics-based interaction with 3d objects via video generation. In *European Conference on Computer Vision*, pages 388–406. Springer, 2024.
- [425] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023.
- [426] Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. Drivedreamer: Towards real-world-drive world models for autonomous driving. In *European Conference on Computer Vision*, pages 55–72. Springer, 2024.
- [427] Yuqi Wang, Jiawei He, Lue Fan, Hongxin Li, Yuntao Chen, and Zhaoxiang Zhang. Driving into the future: Multiview visual forecasting and planning with world model for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14749–14759, 2024.
- [428] Junliang Guo, Yang Ye, Tianyu He, Haoyu Wu, Yushu Jiang, Tim Pearce, and Jiang Bian. Mineworld: a real-time and open-source interactive world model on minecraft. *arXiv preprint arXiv:2504.08388*, 2025.
- [429] Hao Tang, Darren Key, and Kevin Ellis. Worldcoder, a model-based llm agent: Building world models by writing code and interacting with the environment. *Advances in Neural Information Processing Systems*, 37:70148–70212, 2024.
- [430] 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.
- [431] 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.
- [432] 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.
- [433] Basil Kouvaritakis and Mark Cannon. Model predictive control. *Switzerland: Springer International Publishing*, 38(13-56):7, 2016.
- [434] 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.

- [435] Danijar Hafner, Timothy P. Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.
- [436] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, pages 1–7, 2025.
- [437] Siyin Wang, Zhaoye Fei, Qinyuan Cheng, Shiduo Zhang, Panpan Cai, Jinlan Fu, and Xipeng Qiu. World modeling makes a better planner: Dual preference optimization for embodied task planning. *arXiv preprint arXiv:2503.10480*, 2025.
- [438] Allen Newell. *Unified theories of cognition*. Harvard University Press, 1994.
- [439] Jonathan Richens and Tom Everitt. Robust agents learn causal world models. In *The Twelfth International Conference on Learning Representations*, 2024.
- [440] Jonathan Richens, David Abel, Alexis Bellot, and Tom Everitt. General agents need world models. *arXiv preprint arXiv:2506.01622*, 2025.
- [441] Pierre Harvey Richemond, Yunhao Tang, Daniel Guo, Daniele Calandriello, Mohammad Gheshlaghi Azar, Rafael Rafailev, 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.
- [442] 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.
- [443] 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.
- [444] 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>.
- [445] Jiwoo Hong, Noah Lee, and James Thorne. Orpo: Monolithic preference optimization without reference model. *arXiv preprint arXiv:2403.07691*, 2024.
- [446] 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.
- [447] 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.
- [448] 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.
- [449] Deqing Fu, Tong Xiao, Rui Wang, Wang Zhu, Pengchuan Zhang, Guan Pang, Robin Jia, and Lawrence Chen. Tldr: Token-level detective reward model for large vision language models. *arXiv preprint arXiv:2410.04734*, 2024.
- [450] Yue Fan, Xuehai He, Diji Yang, Kaizhi Zheng, Ching-Chen Kuo, Yuting Zheng, Sravana Jyothi Narayananaraju, Xinze Guan, and Xin Eric Wang. Grit: Teaching mllms to think with images. *arXiv preprint arXiv:2505.15879*, 2025.

- [451] Fengyuan Dai, Zifeng Zhuang, Yufei Huang, Siteng Huang, Bangyan Liao, Donglin Wang, and Fajie Yuan. Vard: Efficient and dense fine-tuning for diffusion models with value-based rl. *arXiv preprint arXiv:2505.15791*, 2025.
- [452] Jiaer Xia, Yuhang Zang, Peng Gao, Yixuan Li, and Kaiyang Zhou. Visionary-r1: Mitigating shortcuts in visual reasoning with reinforcement learning. *arXiv preprint arXiv:2505.14677*, 2025.
- [453] Tianhe Wu, Jian Zou, Jie Liang, Lei Zhang, and Kede Ma. Visualquality-r1: Reasoning-induced image quality assessment via reinforcement learning to rank. *arXiv preprint arXiv:2505.14460*, 2025.
- [454] Zuyao Chen, Jinlin Wu, Zhen Lei, Marc Pollefeys, and Chang Wen Chen. Compile scene graphs with reinforcement learning. *arXiv preprint arXiv:2504.13617*, 2025.
- [455] 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.
- [456] 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.
- [457] 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.
- [458] 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.
- [459] 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.
- [460] 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.
- [461] Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. *arXiv preprint arXiv:2404.05868*, 2024.
- [462] 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.
- [463] Zongzhao Li, Zongyang Ma, Mingze Li, Songyou Li, Yu Rong, Tingyang Xu, Ziqi Zhang, Deli Zhao, and Wenbing Huang. Star-r1: Spacial transformation reasoning by reinforcing multimodal llms. *arXiv preprint arXiv:2505.15804*, 2025.
- [464] Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, et al. Dancegrpo: Unleashing grpo on visual generation. *arXiv preprint arXiv:2505.07818*, 2025.
- [465] 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.
- [466] 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.

- [467] 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.
- [468] Jiaxin Guo, Zewen Chi, Li Dong, Qingxiu Dong, Xun Wu, Shaohan Huang, and Furu Wei. Reward reasoning model. *arXiv preprint arXiv:2505.14674*, 2025.
- [469] Baohao Liao, Yuhui Xu, Hanze Dong, Junnan Li, Christof Monz, Silvio Savarese, Doyen Sahoo, and Caiming Xiong. Reward-guided speculative decoding for efficient llm reasoning. *arXiv preprint arXiv:2501.19324*, 2025.
- [470] Zae Myung Kim, Chanwoo Park, Vipul Raheja, and Dongyeop Kang. Toward evaluative thinking: Meta policy optimization with evolving reward models. *arXiv preprint arXiv:2504.20157*, 2025.
- [471] Rui Yang, Xiaoman Pan, Feng Luo, Shuang Qiu, Han Zhong, Dong Yu, and Jianshu Chen. Rewards-in-context: Multi-objective alignment of foundation models with dynamic preference adjustment. *arXiv preprint arXiv:2402.10207*, 2024.
- [472] Zhuohao Yu, Jiali Zeng, Weizheng Gu, Yidong Wang, Jindong Wang, Fandong Meng, Jie Zhou, Yue Zhang, Shikun Zhang, and Wei Ye. Rewardanything: Generalizable principle-following reward models. *arXiv preprint arXiv:2506.03637*, 2025.
- [473] Chengqi Duan, Rongyao Fang, Yuqing Wang, Kun Wang, Linjiang Huang, Xingyu Zeng, Hongsheng Li, and Xihui Liu. Got-r1: Unleashing reasoning capability of mllm for visual generation with reinforcement learning. *arXiv preprint arXiv:2505.17022*, 2025.
- [474] Kaixuan Fan, Kaituo Feng, Haoming Lyu, Dongzhan Zhou, and Xiangyu Yue. Sophiavl-r1: Reinforcing mllms reasoning with thinking reward. *arXiv preprint arXiv:2505.17018*, 2025.
- [475] Chaoya Jiang, Yongrui Heng, Wei Ye, Han Yang, Haiyang Xu, Ming Yan, Ji Zhang, Fei Huang, and Shikun Zhang. Vlm-r³: Region recognition, reasoning, and refinement for enhanced multimodal chain-of-thought. *arXiv preprint arXiv:2505.16192*, 2025.
- [476] Jiaqi Wang, Kevin Qinghong Lin, James Cheng, and Mike Zheng Shou. Think or not? selective reasoning via reinforcement learning for vision-language models. *arXiv preprint arXiv:2505.16854*, 2025.
- [477] Xintong Zhang, Zhi Gao, Bofei Zhang, Pengxiang Li, Xiaowen Zhang, Yang Liu, Tao Yuan, Yuwei Wu, Yunde Jia, Song-Chun Zhu, et al. Chain-of-focus: Adaptive visual search and zooming for multimodal reasoning via rl. *arXiv preprint arXiv:2505.15436*, 2025.
- [478] Ziwei Zheng, Michael Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and Xing Yu. Deepeyes: Incentivizing" thinking with images" via reinforcement learning. *arXiv preprint arXiv:2505.14362*, 2025.
- [479] Sule Bai, Mingxing Li, Yong Liu, Jing Tang, Haoji Zhang, Lei Sun, Xiangxiang Chu, and Yansong Tang. Univg-r1: Reasoning guided universal visual grounding with reinforcement learning. *arXiv preprint arXiv:2505.14231*, 2025.
- [480] Liang Chen, Hongcheng Gao, Tianyu Liu, Zhiqi Huang, Flood Sung, Xinyu Zhou, Yuxin Wu, and Baobao Chang. G1: Bootstrapping perception and reasoning abilities of vision-language model via reinforcement learning. *arXiv preprint arXiv:2505.13426*, 2025.
- [481] Yue Liu, Shengfang Zhai, Mingzhe Du, Yulin Chen, Tri Cao, Hongcheng Gao, Cheng Wang, Xinfeng Li, Kun Wang, Junfeng Fang, et al. Guardreasoner-vl: Safeguarding vlms via reinforced reasoning. *arXiv preprint arXiv:2505.11049*, 2025.
- [482] Zhaochen Su, Linjie Li, Mingyang Song, Yunzhuo Hao, Zhengyuan Yang, Jun Zhang, Guanjie Chen, Jiawei Gu, Juntao Li, Xiaoye Qu, et al. Openthinkimg: Learning to think with images via visual tool reinforcement learning. *arXiv preprint arXiv:2505.08617*, 2025.

- [483] Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl. *arXiv preprint arXiv:2505.05470*, 2025.
- [484] Qianchu Liu, Sheng Zhang, Guanghui Qin, Timothy Ossowski, Yu Gu, Ying Jin, Sid Kiblawi, Sam Preston, Mu Wei, Paul Vozila, et al. X-reasoner: Towards generalizable reasoning across modalities and domains. *arXiv preprint arXiv:2505.03981*, 2025.
- [485] Wenyi Xiao, Leilei Gan, Weilong Dai, Wanggui He, Ziwei Huang, Haoyuan Li, Fangxun Shu, Zhelun Yu, Peng Zhang, Hao Jiang, et al. Fast-slow thinking for large vision-language model reasoning. *arXiv preprint arXiv:2504.18458*, 2025.
- [486] Lin Li, Wei Chen, Jiahui Li, Kwang-Ting Cheng, and Long Chen. Relation-r1: Progressively cognitive chain-of-thought guided reinforcement learning for unified relation comprehension. *arXiv preprint arXiv:2504.14642*, 2025.
- [487] 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.
- [488] Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In *International conference on machine learning*, pages 5062–5071. PMLR, 2019.
- [489] Alex Su, Haozhe Wang, Weimin Ren, Fangzhen Lin, and Wenhua Chen. Pixel reasoner: Incentivizing pixel-space reasoning with curiosity-driven reinforcement learning. *arXiv preprint arXiv:2505.15966*, 2025.
- [490] 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.
- [491] 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.
- [492] Xiangyan Liu, Jinjie Ni, Zijian Wu, Chao Du, Longxu Dou, Haonan Wang, Tianyu Pang, and Michael Qizhe Shieh. Noisyrollout: Reinforcing visual reasoning with data augmentation. *arXiv preprint arXiv:2504.13055*, 2025.
- [493] 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.
- [494] 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.
- [495] 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.
- [496] 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.
- [497] Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024.
- [498] Yuqi Liu, Tianyuan Qu, Zhisheng Zhong, Bohao Peng, Shu Liu, Bei Yu, and Jiaya Jia. Visionreasoner: Unified visual perception and reasoning via reinforcement learning. *arXiv preprint arXiv:2505.12081*, 2025.

- [499] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.
- [500] Wenqi Zhang, Yongliang Shen, Linjuan Wu, Qiuying Peng, Jun Wang, Yueting Zhuang, and Weiming Lu. Self-contrast: Better reflection through inconsistent solving perspectives. *arXiv preprint arXiv:2401.02009*, 2024.
- [501] Yi Xu, Chengzu Li, Han Zhou, Xingchen Wan, Caiqi Zhang, Anna Korhonen, and Ivan Vulić. Visual planning: Let's think only with images. *arXiv preprint arXiv:2505.11409*, 2025.
- [502] Jean-Francois Ton, Muhammad Faaiz Taufiq, and Yang Liu. Understanding chain-of-thought in LLMs through information theory. *arXiv preprint arXiv:2411.11984*, 2024.
- [503] 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.
- [504] Hyoungseok Kim, Jaekyeom Kim, Yeonwoo Jeong, Sergey Levine, and Hyun Oh Song. Emi: Exploration with mutual information. *arXiv preprint arXiv:1810.01176*, 2018.
- [505] Pranav Shyam, Wojciech Jaśkowski, and Faustino Gomez. Model-based active exploration. In *International conference on machine learning*, pages 5779–5788. PMLR, 2019.
- [506] Xiusi Chen, Gaotang Li, Ziqi Wang, Bowen Jin, Cheng Qian, Yu Wang, Hongru Wang, Yu Zhang, Denghui Zhang, Tong Zhang, et al. Rm-r1: Reward modeling as reasoning. *arXiv preprint arXiv:2505.02387*, 2025.
- [507] 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.
- [508] 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.
- [509] 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.
- [510] 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.
- [511] 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. *arXiv preprint arXiv:2502.19328*, 2025.
- [512] Peiyu Wang, Yichen Wei, Yi Peng, Xiaokun Wang, Weijie Qiu, Wei Shen, Tianyidan Xie, Jiangbo Pei, Jianhao Zhang, Yunzhuo Hao, et al. Skywork r1v2: Multimodal hybrid reinforcement learning for reasoning. *arXiv preprint arXiv:2504.16656*, 2025.
- [513] Yongcheng Zeng, Guoqing Liu, Weiyu Ma, Ning Yang, Haifeng Zhang, and Jun Wang. Token-level direct preference optimization. *arXiv preprint arXiv:2404.11999*, 2024.
- [514] Ruilin Luo, Zhuofan Zheng, Yifan Wang, Yiyao Yu, Xinzhe Ni, Zicheng Lin, Jin Zeng, and Yujiu Yang. Ursu: Understanding and verifying chain-of-thought reasoning in multimodal mathematics. *arXiv preprint arXiv:2501.04686*, 2025.
- [515] Huanjin Yao, Qixiang Yin, Jingyi Zhang, Min Yang, Yibo Wang, Wenhao Wu, Fei Su, Li Shen, Minghui Qiu, Dacheng Tao, et al. R1-sharevl: Incentivizing reasoning capability of multimodal large language models via share-grpo. *arXiv preprint arXiv:2505.16673*, 2025.

- [516] Ziyu Liu, Yuhang Zang, Yushan Zou, Zijian Liang, Xiaoyi Dong, Yuhang Cao, Haodong Duan, Dahua Lin, and Jiaqi Wang. Visual agentic reinforcement fine-tuning. *arXiv preprint arXiv:2505.14246*, 2025.
- [517] Dongzhi Jiang, Ziyu Guo, Renrui Zhang, Zhuofan Zong, Hao Li, Le Zhuo, Shilin Yan, Pheng-Ann Heng, and Hongsheng Li. T2i-r1: Reinforcing image generation with collaborative semantic-level and token-level cot. *arXiv preprint arXiv:2505.00703*, 2025.
- [518] Robert G Lewis, Ermanno Florio, Daniela Punzo, and Emiliana Borrelli. *The Brain's reward system in health and disease*. Springer, 2021.
- [519] Marc Fakhoury. *The Brain Reward System*. Springer, 2021.
- [520] Vincent Breton-Provencher and Mriganka Sur. Active control of arousal by a locus coeruleus gabaergic circuit. *Nature neuroscience*, 22(2):218–228, 2019.
- [521] 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.
- [522] 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.
- [523] Bilal Abdul Bari, Varun Chokshi, and Katharina Schmidt. Locus coeruleus-norepinephrine: basic functions and insights into parkinson’s disease. *Neural regeneration research*, 15(6):1006–1013, 2020.
- [524] Norton Contreras Paredes. Models of neural processing of consciousness: insights from cognitive and systems neuroscience. *Revista mexicana de neurociencia*, 23(6):214–222, 2022.
- [525] Haiyun Xu and Fan Yang. The interplay of dopamine metabolism abnormalities and mitochondrial defects in the pathogenesis of schizophrenia. *Translational psychiatry*, 12(1):464, 2022.
- [526] Wikipedia contributors. Reward system. https://en.wikipedia.org/wiki/Reward_system, 2025. Accessed: June 8, 2025.
- [527] MSD Manual. Neurotransmission, 2022. URL <https://www.msdmanuals.cn/professional/neurologic-disorders/neurotransmission/neurotransmission>. Accessed: 2022-04-01.
- [528] Anil Ananthaswamy. How close is AI to human-level intelligence? *Nature*, 636(8041):22–25, 2024.
- [529] 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.
- [530] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Pearson, 2016.
- [531] Jinghan Zhang, Xiting Wang, Yiqiao Jin, Changyu Chen, Xinhao Zhang, and Kunpeng Liu. Prototypical reward network for data-efficient rlhf. In *ACL*, 2024.
- [532] Andrew G Barto. Intrinsic motivation and reinforcement learning. In *Intrinsically motivated learning in natural and artificial systems*, pages 17–47. Springer, 2012.
- [533] Sebastian Thrun and Michael L Littman. Reinforcement learning: An introduction. *AI Magazine*, 21(1):103–103, 2000.
- [534] 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.
- [535] Eric Chen, Zhang-Wei Hong, Joni Pajarinen, and Pulkit Agrawal. Redeeming intrinsic rewards via constrained optimization. *Advances in neural information processing systems*, 35:4996–5008, 2022.
- [536] Eunseop Yoon, Hee Suk Yoon, SooHwan Eom, Gunsoo Han, Daniel Wontae Nam, Daejin Jo, Kyoung-Woon On, Mark A Hasegawa-Johnson, Sungwoong Kim, and Chang D Yoo. Tlcr: Token-level

- continuous reward for fine-grained reinforcement learning from human feedback. *arXiv preprint arXiv:2407.16574*, 2024.
- [537] Luiz Pessoa. Multiple influences of reward on perception and attention. *Visual cognition*, 23(1-2):272–290, 2015.
 - [538] 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.
 - [539] Ewa A Miendlarzewska, Daphne Bavelier, and Sophie Schwartz. Influence of reward motivation on human declarative memory. *Neuroscience & Biobehavioral Reviews*, 61:156–176, 2016.
 - [540] Marvin Lee Minsky, editor. *The Emotion Machine: Commensense Thinking, Artificial Intelligence, and the Future of the Human Mind*. Simon & Schuster, 2006.
 - [541] Paul Ekman. An argument for basic emotions. *Cognition & Emotion*, 6:169–200, 1992.
 - [542] James Russell. A circumplex model of affect. *Journal of Personality and Social Psychology*, 39:1161–1178, 12 1980. doi:[10.1037/h0077714](https://doi.org/10.1037/h0077714).
 - [543] Robert Plutchik. A general psychoevolutionary theory of emotion. *Theories of emotion*, 1:3–31, 1980.
 - [544] 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.
 - [545] Xuena Wang, Xuetong Li, Zi Yin, Yue Wu, and Jia Liu. Emotional intelligence of large language models. *Journal of Pacific Rim Psychology*, 17:18344909231213958, 2023.
 - [546] 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.
 - [547] 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](https://doi.org/10.1073/pnas.1200155109). URL <https://www.pnas.org/doi/abs/10.1073/pnas.1200155109>.
 - [548] Albert Mehrabian. Pleasure-arousal-dominance: A general framework for describing and measuring individual differences in temperament. *Current Psychology*, 14:261–292, 1996.
 - [549] 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](https://doi.org/10.18653/v1/2023.findings-emnlp.395). URL <https://aclanthology.org/2023.findings-emnlp.395/>.
 - [550] Ala N. Tak and Jonathan Gratch. Is gpt a computational model of emotion? detailed analysis. *arXiv preprint arXiv: 2307.13779*, 2023.
 - [551] 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.
 - [552] Klaus R. Scherer. The dynamic architecture of emotion: Evidence for the component process model. *Cognition and Emotion*, 23:1307 – 1351, 2009.
 - [553] Andrew Ortony, Gerald L Clore, and Allan Collins. *The cognitive structure of emotions*. Cambridge university press, 2022.
 - [554] Eva Hudlicka. Computational modeling of cognition-emotion interactions: Relevance to mechanisms of affective disorders and therapeutic action. *Cognitive Science*, 36, 2014.

- [555] Stacy Marsella and J. Gratch. Computationally modeling human emotion. *Commun. ACM*, 57:56–67, 2014.
- [556] 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](https://doi.org/10.48550/arXiv.2305.11255).
- [557] 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.
- [558] 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.
- [559] Hugo Lövheim. A new three-dimensional model for emotions and monoamine neurotransmitters. *Medical Hypotheses*, 78(2):341–348, 2012. doi:[10.1016/j.mehy.2011.11.016](https://doi.org/10.1016/j.mehy.2011.11.016).
- [560] 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.
- [561] 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.
- [562] 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](https://doi.org/10.18653/v1/2024.acl-long.326). URL <https://aclanthology.org/2024.acl-long.326/>.
- [563] 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](https://doi.org/10.1145/3664647.3681403). URL <https://doi.org/10.1145/3664647.3681403>.
- [564] 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.
- [565] 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.
- [566] 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.
- [567] 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](https://doi.org/10.1016/j.inffus.2024.102367). URL <https://doi.org/10.1016/j.inffus.2024.102367>.
- [568] 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.

- [569] 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](https://doi.org/10.1145/3603287.3651183). URL <https://doi.org/10.1145/3603287.3651183>.
- [570] Steve Rathje, Dan-Mircea Mirea, Ilia Sucholutsky, Raja Marjeh, 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.
- [571] 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](https://doi.org/10.21437/interspeech.2024-2282).
- [572] 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.
- [573] 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.
- [574] 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.
- [575] 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](https://doi.org/10.1109/TAFFC.2024.3419593). URL <https://doi.org/10.1109/TAFFC.2024.3419593>.
- [576] 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.
- [577] 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.
- [578] 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.
- [579] 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>.
- [580] 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.
- [581] Bang Zhang, Ruotian Ma, Qingxuan Jiang, Peisong Wang, Jiaqi Chen, Zheng Xie, Xingyu Chen, Yue Wang, Fanghua Ye, Jian Li, Yifan Yang, Zhaopeng Tu, and Xiaolong Li. Sentient agent as a judge: Evaluating higher-order social cognition in large language models, 2025. URL <https://arxiv.org/abs/2505.02847>.
- [582] 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.

- [583] 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](https://doi.org/10.18653/V1/2024.ACL-LONG.102). URL <https://doi.org/10.18653/v1/2024.acl-long.102>.
- [584] 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>.
- [585] 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.
- [586] 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](https://doi.org/10.48550/arXiv.2402.18225).
- [587] 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](https://doi.org/10.1609/aaai.v38i17.29856).
- [588] 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.
- [589] 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>.
- [590] 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>.
- [591] 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.
- [592] 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](https://doi.org/10.18653/v1/2024.findings-naacl.229). URL <https://aclanthology.org/2024.findings-naacl.229>.
- [593] 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.
- [594] 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.
- [595] 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.

- [596] Jen-tse Huang, Wenxiang Jiao, Man Ho Lam, Eric John Li, Wenzuan 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/>.
- [597] 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.
- [598] Lena Podoletz. We have to talk about emotional AI and crime. *AI & SOCIETY*, 38(3):1067–1082, 2023.
- [599] 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.
- [600] 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.
- [601] 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>.
- [602] 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.
- [603] 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.
- [604] 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.
- [605] 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.
- [606] Baihan Lin, Djallel Bounoufouf, Guillermo Cecchi, and Kush R Varshney. Towards healthy ai: large language models need therapists too. *arXiv preprint arXiv:2308.04434*, 2023.
- [607] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [608] Yinhan Liu. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364, 2019.
- [609] Z Lan. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.
- [610] 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.
- [611] 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.

- [612] 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.
- [613] 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.
- [614] 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.
- [615] 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.
- [616] 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.
- [617] 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.
- [618] 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.
- [619] 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.
- [620] 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.
- [621] 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.
- [622] 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.
- [623] 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.
- [624] 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.
- [625] 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.
- [626] 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.

- [627] 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.
- [628] 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.
- [629] 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.
- [630] 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.
- [631] 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.
- [632] 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.
- [633] 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.
- [634] 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.
- [635] 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.
- [636] Wenyi Hong, Weihan Wang, Ming Ding, Wenmeng Yu, Qingsong Lv, Yan 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.
- [637] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.
- [638] Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024.
- [639] Dongzhe Fan, Yi Fang, Jiajin Liu, Djellel Difallah, and Qiaoyu Tan. Mlag: Multimodal large language and graph assistant. *arXiv preprint arXiv:2506.02568*, 2025.
- [640] Zebin You, Shen Nie, Xiaolu Zhang, Jun Hu, Jun Zhou, Zhiwu Lu, Ji-Rong Wen, and Chongxuan Li. Llada-v: Large language diffusion models with visual instruction tuning. *arXiv preprint arXiv:2505.16933*, 2025.
- [641] 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.
- [642] Xiangxiang Chu, Limeng Qiao, Xinyu Zhang, Shuang Xu, Fei Wei, Yang Yang, Xiaofei Sun, Yiming Hu, Xinyang Lin, Bo Zhang, et al. Mobilevlm v2: Faster and stronger baseline for vision language model. *arXiv preprint arXiv:2402.03766*, 2024.

- [643] 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.
- [644] Wenwen Yu, Zhibo Yang, Jianqiang Wan, Sibo Song, Jun Tang, Wenqing Cheng, Yuliang Liu, and Xiang Bai. Omniparser v2: Structured-points-of-thought for unified visual text parsing and its generality to multimodal large language models. *arXiv preprint arXiv:2502.16161*, 2025.
- [645] 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.
- [646] 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.
- [647] 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.
- [648] 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.
- [649] 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.
- [650] 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.
- [651] 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.
- [652] 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.
- [653] 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.
- [654] 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.
- [655] Dongting Li, Chenchong Tang, and Han Liu. Audio-lm: Activating the capabilities of large language models to comprehend audio data. In *International Symposium on Neural Networks*, pages 133–142. Springer, 2024.
- [656] Zhifei Xie and Changqiao Wu. Mini-omni2: Towards open-source gpt-4o with vision, speech and duplex capabilities. *arXiv preprint arXiv:2410.11190*, 2024.
- [657] 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.
- [658] 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.

- [659] 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.
- [660] 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.
- [661] 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.
- [662] 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.
- [663] 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.
- [664] Jin Xu, Zhifang Guo, Jinzheng He, Hangrui Hu, Ting He, Shuai Bai, Keqin Chen, Jialin Wang, Yang Fan, Kai Dang, et al. Qwen2.5-omni technical report. *arXiv preprint arXiv:2503.20215*, 2025.
- [665] 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.
- [666] 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.
- [667] 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.
- [668] 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.
- [669] 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.
- [670] 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.
- [671] Yunfei Guo and Wu Huang. Llava-next-med: Medical multimodal large language model. In *2025 Asia-Europe Conference on Cybersecurity, Internet of Things and Soft Computing (CITSC)*, pages 474–477. IEEE, 2025.
- [672] Fiona Macpherson. *The senses: Classic and contemporary philosophical perspectives*, volume 11. Oxford University Press, 2011.
- [673] Jamie Ward. *The student's guide to cognitive neuroscience*. Routledge, 2019.
- [674] Stanley Coren, Lawrence M Ward, and James T Enns. *Sensation and perception*. John Wiley & Sons Hoboken, NJ, 2004.
- [675] 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.
- [676] Henrik Mouritsen. Long-distance navigation and magnetoreception in migratory animals. *Nature*, 558(7708):50–59, 2018.

- [677] 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.
- [678] 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.
- [679] 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.
- [680] 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.
- [681] 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.
- [682] OpenAI. Gpt-3.5: Language model, 2023. URL <https://platform.openai.com/docs/models/gpt-3.5-turbo>.
- [683] Glenn Jocher. YOLOv5 by Ultralytics, May 2020. URL <https://github.com/ultralytics/yolov5>.
- [684] Glenn Jocher, Jing Qiu, and Ayush Chaurasia. Ultralytics YOLO, January 2023. URL <https://github.com/ultralytics/ultralytics>.
- [685] 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.
- [686] 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.
- [687] 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.
- [688] 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.
- [689] 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.
- [690] 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.
- [691] 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.
- [692] 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.
- [693] 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.

- [694] Max Bain, Arsha Nagrani, G  l 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.
- [695] Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre D  f  ossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi. Audiogen: Textually guided audio generation. *arXiv preprint arXiv:2209.15352*, 2022.
- [696] 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.
- [697] 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.
- [698] 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.
- [699] 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.
- [700] 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>.
- [701] 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.
- [702] 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.
- [703] 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.
- [704] 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.
- [705] 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.
- [706] Mojtaba Javaheripour, S  bastien Bubeck, Marah Abdin, Jyoti Aneja, S  bastien Bubeck, Caio C  sar 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.
- [707] 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.

- [708] 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.
- [709] 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.
- [710] 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.
- [711] 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.
- [712] 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.
- [713] 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.
- [714] Sreyan Ghosh, Zhifeng Kong, Sonal Kumar, S Sakshi, Jaehyeon Kim, Wei Ping, Rafael Valle, Dinesh Manocha, and Bryan Catanzaro. Audio flamingo 2: An audio-language model with long-audio understanding and expert reasoning abilities. *arXiv preprint arXiv:2503.03983*, 2025.
- [715] 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.
- [716] 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.
- [717] 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.
- [718] 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.
- [719] 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.
- [720] 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.
- [721] 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.
- [722] 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.
- [723] 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.

- [724] 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.
- [725] 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.
- [726] 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.
- [727] 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.
- [728] Yaodong Yang and Jun Wang. An overview of multi-agent reinforcement learning from game theoretical perspective. *arXiv preprint arXiv:2011.00583*, 2020.
- [729] Zhenbei Guo, Fuliang Li, Jiaxing Shen, Tangzheng Xie, Shan Jiang, and Xingwei Wang. Configreco: Network configuration recommendation with graph neural networks. *IEEE Network*, 2023.
- [730] 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.
- [731] 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.
- [732] 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.
- [733] 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.
- [734] 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.
- [735] 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.
- [736] 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.
- [737] 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.
- [738] 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.
- [739] 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.
- [740] 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.
- [741] Kai Li and Yi Luo. Apollo: Band-sequence modeling for high-quality audio restoration. *arXiv preprint arXiv:2409.08514*, 2024.
 - [742] 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.
 - [743] 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.
 - [744] 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.
 - [745] 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.
 - [746] 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.
 - [747] 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.
 - [748] 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.
 - [749] 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.
 - [750] 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.
 - [751] 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.
 - [752] 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.
 - [753] 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](https://doi.org/10.1038/s41583-024-00895-x).
 - [754] Significant Gravitas. Auto-gpt: An autonomous gpt-4 experiment. <https://github.com/Significant-Gravitas/Auto-GPT>, 2023.
 - [755] 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.

- [756] 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>.
- [757] 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>.
- [758] 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>.
- [759] 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.
- [760] 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.
- [761] 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.
- [762] 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.
- [763] 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>.
- [764] 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.
- [765] 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.
- [766] 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.
- [767] Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. Omniparser for pure vision based gui agent. *arXiv preprint arXiv:2408.00203*, 2024.
- [768] 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.
- [769] Yu Gu, Xiang Deng, and Yu Su. Don't generate, discriminate: A proposal for grounding language models to real-world environments. In *ACL*, 2023.
- [770] 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.
- [771] 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.
 - [772] 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.
 - [773] 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.
 - [774] 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.
 - [775] 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.
 - [776] 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.
 - [777] 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.
 - [778] Yao Mu, Qinglong Zhang, Mengkang Hu, Wenhui 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.
 - [779] 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.
 - [780] 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.
 - [781] 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.
- [782] Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbulin, and Jürgen Schmidhuber. Gptswarm: Language agents as optimizable graphs. In *ICML*. OpenReview.net, 2024.
- [783] 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.
- [784] 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.
- [785] 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.
- [786] 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.
- [787] 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.
- [788] 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.
- [789] 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.
- [790] 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.
- [791] 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.
- [792] 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.
- [793] 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.
- [794] 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.
- [795] 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.pdf>, 2024.
- [796] 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.

- [797] 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>.
- [798] 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.
- [799] 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.
- [800] Reg Revans. *ABC of action learning*. Routledge, 2017.
- [801] 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.
- [802] 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.
- [803] 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.
- [804] 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.
- [805] 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.
- [806] 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.
- [807] 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.
- [808] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [809] Kimi Team. Kimi k1.5: Scaling reinforcement learning with llms. *CoRR*, abs/2501.12599, 2025.
- [810] Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*, 2025.
- [811] Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2502.14768*, 2025.
- [812] Mingyang Chen, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z Pan, Wen Zhang, Huajun Chen, Fan Yang, et al. Learning to reason with search for llms via reinforcement learning. *arXiv preprint arXiv:2503.19470*, 2025.
- [813] Junjie Zhang, Jingyi Xi, Zhuoyang Song, Junyu Lu, Yuhua Ke, Ting Sun, Yukun Yang, Jiaxing Zhang, Songxin Zhang, and Zejian Xie. L0: Reinforcement learning to become general agents, 2025. URL <https://arxiv.org/abs/2506.23667>.

- [814] Jie Ouyang, Ruiran Yan, Yucong Luo, Mingyue Cheng, Qi Liu, Zirui Liu, Shuo Yu, and Daoyu Wang. Training powerful llm agents with end-to-end reinforcement learning, 2025. URL <https://github.com/0russwest0/Agent-R1>.
- [815] 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>.
- [816] Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. *arXiv preprint arXiv:2505.10978*, 2025.
- [817] 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.
- [818] Jian Hu, Li Tao, June Yang, and Chandler Zhou. Aligning language models with offline learning from human feedback. *arXiv preprint arXiv:2308.12050*, 2023.
- [819] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.
- [820] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, 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>.
- [821] 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>.
- [822] 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>.
- [823] 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.
- [824] 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.
- [825] 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>.
- [826] 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](https://doi.org/10.1007/s10514-023-10139-z). URL <http://dx.doi.org/10.1007/s10514-023-10139-z>.
- [827] Huan Zhang, Yu Song, Ziyu Hou, Santiago Miret, and Bang Liu. Honeycomb: A flexible llm-based agent system for materials science, 2024. URL <https://arxiv.org/abs/2409.00135>.

- [828] 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.
- [829] Huajun Chen, Keyan Ding, Jing Yu, Junjie Huang, Yuchen Yang, and Qiang Zhang. Scitooolagent: A knowledge graph-driven scientific agent for multi-tool integration. In *ICLR*, 2025.
- [830] 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.
- [831] 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.
- [832] 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.
- [833] Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. Large language models as tool makers. *arXiv preprint arXiv:2305.17126*, 2023.
- [834] 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>.
- [835] 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>.
- [836] Jiazhan Feng, Shijue Huang, Xingwei Qu, Ge Zhang, Yujia Qin, Baoquan Zhong, Chengquan Jiang, Jinxin Chi, and Wanjun Zhong. Retool: Reinforcement learning for strategic tool use in llms. *CoRR*, abs/2504.11536, 2025.
- [837] Yuanqing Yu, Zhefan Wang, Weizhi Ma, Zhicheng Guo, Jingtao Zhan, Shuai Wang, Chuhan Wu, Zhiqiang Guo, and Min Zhang. Steptool: A step-grained reinforcement learning framework for tool learning in llms. *CoRR*, abs/2410.07745, 2024.
- [838] Guanting Dong, Yifei Chen, Xiaoxi Li, Jiajie Jin, Hongjin Qian, Yutao Zhu, Hangyu Mao, Guorui Zhou, Zhicheng Dou, and Ji-Rong Wen. Tool-star: Empowering llm-brained multi-tool reasoner via reinforcement learning. *CoRR*, abs/2505.16410, 2025.
- [839] 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.
- [840] 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.
- [841] 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>.
- [842] 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>.

- [843] 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.
- [844] 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.
- [845] 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.
- [846] Yuyu Luo, Xuedi Qin, Nan Tang, and Guoliang Li. Deepeye: Towards automatic data visualization. In *ICDE*, pages 101–112. IEEE Computer Society, 2018.
- [847] 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.
- [848] 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](https://doi.org/10.1145/3704435). URL <https://doi.org/10.1145/3704435>.
- [849] 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](https://doi.org/10.1109/LRA.2025.3535184). URL <http://dx.doi.org/10.1109/LRA.2025.3535184>.
- [850] Zhenjie Yang, Xiaosong Jia, Hongyang Li, and Junchi Yan. Llm4drive: A survey of large language models for autonomous driving, 2023.
- [851] Jiageng Mao, Junjie Ye, Yuxi Qian, Marco Pavone, and Yue Wang. A language agent for autonomous driving. *arXiv preprint arXiv:2311.10813*, 2023.
- [852] Sherwood L Washburn. Tools and human evolution. *Scientific American*, 203(3):62–75, 1960.
- [853] Nan Tang, Chenyu Yang, Ju Fan, Lei Cao, Yuyu Luo, and Alon Y. Halevy. Verifai: Verified generative AI. In *CIDR*. www.cidrdb.org, 2024.
- [854] Kangrui Wang*, Pingyue Zhang*, Zihan Wang*, Yaning Gao*, Linjie Li*, Qineng Wang, Hanyang Chen, Chi Wan, Yiping Lu, Zhengyuan Yang, Lijuan Wang, Ranjay Krishna, Jiajun Wu, Li Fei-Fei, Yejin Choi, and Manling Li. Reinforcing visual state reasoning for multi-turn vlm agents, 2025. URL <https://github.com/RAGEN-AI/VAGEN>.
- [855] Ziliang Wang, Xuhui Zheng, Kang An, Cijun Ouyang, Jialu Cai, Yuhang Wang, and Yichao Wu. Stepsearch: Igniting llms search ability via step-wise proximal policy optimization, 2025. URL <https://arxiv.org/abs/2505.15107>.
- [856] 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>.
- [857] Zhepei Wei, Wenlin Yao, Yao Liu, Weizhi Zhang, Qin Lu, Liang Qiu, Changlong Yu, Puyang Xu, Chao Zhang, Bing Yin, Hyokun Yun, and Lihong Li. Webagent-r1: Training web agents via end-to-end multi-turn reinforcement learning, 2025. URL <https://arxiv.org/abs/2505.16421>.
- [858] Tyler Griggs, Sumanth Hegde, Eric Tang, Shu Liu, Shiyi Cao, Dacheng Li, Charlie Ruan, Philipp Moritz, Kourosh Hakhamaneshi, Richard Liaw, Akshay Malik, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Evolving skyrl into a highly-modular rl framework, 2025. Notion Blog.

- [859] Shu Liu, Sumanth Hegde, Shiyi Cao, Alan Zhu, Dacheng Li, Tyler Griggs, Eric Tang, Akshay Malik, Kourosh Hakhamaneshi, Richard Liaw, Philipp Moritz, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Skyrl-sql: Matching gpt-4o and o4-mini on text2sql with multi-turn rl, 2025.
- [860] Siliang Zeng, Quan Wei, William Brown, Oana Frunza, Yuriy Nevmyvaka, and Mingyi Hong. Reinforcing multi-turn reasoning in llm agents via turn-level credit assignment, 2025. URL <https://arxiv.org/abs/2505.11821>.
- [861] 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>.
- [862] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.
- [863] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004.
- [864] 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.
- [865] 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>.
- [866] 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.
- [867] 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.
- [868] 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.
- [869] 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.
- [870] 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.
- [871] 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.
- [872] 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.
- [873] 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.
- [874] 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.

- [875] 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.
- [876] 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.
- [877] 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.
- [878] 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>.
- [879] 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>.
- [880] 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.
- [881] 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.
- [882] Wenyi Wang, Hisham A Alyahya, Dylan R Ashley, Oleg Serikov, Dmitrii Khizbulin, Francesco Faccio, and Jürgen Schmidhuber. How to correctly do semantic backpropagation on language-based agentic systems. *arXiv preprint arXiv:2412.03624*, 2024.
- [883] Xuanchang Zhang, Zhuosheng Zhang, and Hai Zhao. Glape: Gold label-agnostic prompt evaluation and optimization for large language model. *CoRR*, abs/2402.02408, 2024.
- [884] 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.
- [885] 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.
- [886] 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.
- [887] 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.
- [888] 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.
- [889] 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.
- [890] Mingchen Zhuge, Changsheng Zhao, Dylan Ashley, Wenyi Wang, Dmitrii Khizbulin, 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.

- [891] 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.
- [892] 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.
- [893] 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.
- [894] 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.
- [895] 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.
- [896] Tal Ridnik, Dedy Kredo, and Itamar Friedman. Code generation with alphacodium: From prompt engineering to flow engineering. *CoRR*, abs/2401.08500, 2024.
- [897] 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.
- [898] Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. *arXiv preprint arXiv:2408.08435*, 2024.
- [899] 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.
- [900] 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.
- [901] Rui Ye, Shuo Tang, Rui Ge, Yixin Du, Zhenfei Yin, Siheng Chen, and Jing Shao. Mas-gpt: Training llms to build llm-based multi-agent systems. *arXiv preprint arXiv:2503.03686*, 2025.
- [902] Hongcheng Gao, Yue Liu, Yufei He, Longxu Dou, Chao Du, Zhijie Deng, Bryan Hooi, Min Lin, and Tianyu Pang. Flowreasoner: Reinforcing query-level meta-agents. *arXiv preprint arXiv:2504.15257*, 2025.
- [903] Zixuan Ke, Austin Xu, Yifei Ming, Xuan-Phi Nguyen, Caiming Xiong, and Shafiq Joty. Mas-zero: Designing multi-agent systems with zero supervision, 2025. URL <https://arxiv.org/abs/2505.14996>.
- [904] 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>.
- [905] 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.
- [906] Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. Toolrl: Reward is all tool learning needs, 2025. URL <https://arxiv.org/abs/2504.13958>.

- [907] Hongru Wang, Cheng Qian, Wanjun Zhong, Xiusi Chen, Jiahao Qiu, Shijue Huang, Bowen Jin, Mengdi Wang, Kam-Fai Wong, and Heng Ji. Acting less is reasoning more! teaching model to act efficiently, 2025. URL <https://arxiv.org/abs/2504.14870>.
- [908] Shaokun Zhang, Yi Dong, Jieyu Zhang, Jan Kautz, Bryan Catanzaro, Andrew Tao, Qingyun Wu, Zhiding Yu, and Guilin Liu. Nemotron-research-tool-n1: Exploring tool-using language models with reinforced reasoning, 2025. URL <https://arxiv.org/abs/2505.00024>.
- [909] 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.
- [910] 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.
- [911] 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.
- [912] 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.
- [913] 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.
- [914] 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.
- [915] 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>.
- [916] 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.
- [917] 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>.
- [918] Yifei Zhou, Sergey Levine, Jason Weston, Xian Li, and Sainbayar Sukhbaatar. Self-challenging language model agents. *arXiv preprint arXiv:2506.01716*, 2025.
- [919] 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.
- [920] 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.
- [921] 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>.
- [922] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.
- [923] James C Spall. *Introduction to stochastic search and optimization: estimation, simulation, and control*. John Wiley & Sons, 2005.

- [924] Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.
- [925] Nikolaus Hansen. The cma evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*, 2016.
- [926] 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.
- [927] OpenAI. Openai o3-mini system card, 2025. URL <https://openai.com/index/openai-o3-mini/>. [Online; accessed 2025-02-02].
- [928] Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. *arXiv preprint arXiv:2304.05128*, 2023.
- [929] Qinyuan Ye, Maxamed Axmed, Reid Pryzant, and Fereshte Khani. Prompt engineering a prompt engineer. *arXiv preprint arXiv:2311.05661*, 2023.
- [930] 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.
- [931] 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.
- [932] Ting-Yun Chang and Robin Jia. Data curation alone can stabilize in-context learning. *arXiv preprint arXiv:2212.10378*, 2022.
- [933] Tai Nguyen and Eric Wong. In-context example selection with influences. *arXiv preprint arXiv:2302.11042*, 2023.
- [934] 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](https://doi.org/10.18653/v1/2024.findings-emnlp.588). URL <https://aclanthology.org/2024.findings-emnlp.588>.
- [935] 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>.
- [936] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [937] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.
- [938] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.
- [939] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.
- [940] 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.
- [941] Sewon Min, Xinxin 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.
- [942] 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.

- [943] 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](https://doi.org/10.18653/v1/2024.emnlp-main.525). URL <https://aclanthology.org/2024.emnlp-main.525>.
- [944] Shuhei Watanabe. Tree-structured parzen estimator: Understanding its algorithm components and their roles for better empirical performance. *arXiv preprint arXiv:2304.11127*, 2023.
- [945] 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](https://doi.org/10.18653/v1/2024.emnlp-main.226). URL <https://aclanthology.org/2024.emnlp-main.226>.
- [946] Brandon Amos et al. Tutorial on amortized optimization. *Foundations and Trends® in Machine Learning*, 16(5):592–732, 2023.
- [947] Anselm Paulus, Arman Zharmagambetov, Chuan Guo, Brandon Amos, and Yuandong Tian. Advprompter: Fast adaptive adversarial prompting for LLMs. *arXiv preprint arXiv:2404.16873*, 2024.
- [948] 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>.
- [949] 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.
- [950] Sicheng Zhu, Brandon Amos, Yuandong Tian, Chuan Guo, and Ivan Evtimov. Advprefix: An objective for nuanced LLM jailbreaks. *arXiv preprint arXiv:2412.10321*, 2024.
- [951] 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>.
- [952] 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.
- [953] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. *arXiv preprint arXiv:1807.03819*, 2018.
- [954] Laurent Hascoet and Mauricio Araya-Polo. Enabling user-driven checkpointing strategies in reverse-mode automatic differentiation. *arXiv preprint cs/0606042*, 2006.
- [955] 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.
- [956] 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=f1NZJ2e0et>.
- [957] 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>.

- [958] 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>.
- [959] Deqing Fu, Tianqi 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>.
- [960] 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=RdJVFCjUMI>.
- [961] 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.
- [962] 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.
- [963] 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.
- [964] 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.
- [965] 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.
- [966] 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 monosematicity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. URL <https://transformer-circuits.pub/2024/scaling-monosematicity/index.html>.
- [967] 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.
- [968] 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.
- [969] 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.
- [970] 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>.

- [971] 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.
- [972] 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.
- [973] Giovanni Monea, Antoine Bosselut, Kianté Brantley, and Yoav Artzi. Llms are in-context reinforcement learners. In *ICLR*, 2024.
- [974] 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.
- [975] Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbulin, and Bernard Ghanem. Camel: Communicative agents for “mind” exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008, 2023.
- [976] 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.
- [977] Collin Zhang, John X Morris, and Vitaly Shmatikov. Extracting prompts by inverting LLM outputs. *arXiv preprint arXiv:2405.15012*, 2024.
- [978] 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.
- [979] Yizhang Zhu, Shiyin Du, Boyan Li, Yuyu Luo, and Nan Tang. Are large language models good statisticians? In *NeurIPS*, 2024.
- [980] 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>.
- [981] 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>.
- [982] 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>.
- [983] 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.
- [984] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.
- [985] Mingye Zhu, Yi Liu, Lei Zhang, Junbo Guo, and Zhendong Mao. Lire: listwise reward enhancement for preference alignment. *arXiv preprint arXiv:2405.13516*, 2024.
- [986] Teng Wang, Zhangyi Jiang, Zhenqi He, Shenyang Tong, Wenhan Yang, Yanan Zheng, Zeyu Li, Zifan He, and Hailei Gong. Towards hierarchical multi-step reward models for enhanced reasoning in large language models. *arXiv preprint arXiv:2503.13551*, 2025.
- [987] 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.
- [988] 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.

- [989] 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.
- [990] 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.
- [991] Plato, Bernard Williams, M. J. Levett, and Myles Burnyeat. *Theaetetus*. Hackett Publishing, January 1992. ISBN 978-0-87220-158-3.
- [992] Edmund L Gettier. Is Justified True Belief Knowledge? *Analysis*, June 1963. doi:[10.1093/analys/23.6.121](https://doi.org/10.1093/analys/23.6.121).
- [993] 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.
- [994] E. T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003. ISBN 978-0-521-59271-0. doi:[10.1017/CBO9780511790423](https://doi.org/10.1017/CBO9780511790423).
- [995] Thomas Parr, Giovanni Pezzulo, and Karl J. Friston. *Active Inference: The Free Energy Principle in Mind, Brain, and Behavior*. MIT Press, 2022.
- [996] François Chollet. On the Measure of Intelligence, November 2019.
- [997] Thomas M Cover and Joy A Thomas. *ELEMENTS OF INFORMATION THEORY*. John Wiley & Sons, April 2005.
- [998] Sergey N. Britvin, Mikhail N. Murashko, Maria G. Krzhizhanovskaya, Natalia S. Vlasenko, Oleg S. Vereshchagin, Yevgeny Vapnik, and Vladimir N. Bocharov. Crocobelonite, CaFe_{23+(PO₄)2O}, a new oxyphosphate mineral, the product of pyrolytic oxidation of natural phosphides. *American Mineralogist*, 108(10):1973–1983, 2023. ISSN 1945-3027. doi:[10.2138/am-2022-8757](https://doi.org/10.2138/am-2022-8757). URL <https://www.degruyter.com/document/doi/10.2138/am-2022-8757/html?lang=en>.
- [999] Materials Project. CaFe₂P₂O₉ (mp-1040941). <https://materialsproject.org/materials/mp-1040941>, 2025. Data retrieved from Materials Project database version v2025.06.09.
- [1000] 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](https://doi.org/10.1037/h0046743).
- [1001] Raymond B. Cattell. *Abilities: Their Structure, Growth, and Action*. Houghton Mifflin, 1971. ISBN 978-0-395-04275-5.
- [1002] Alexandre 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](https://doi.org/10.1038/s41467-021-26196-w).
- [1003] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A. Efros. Large-Scale Study of Curiosity-Driven Learning, August 2018.
- [1004] Xinghong Fu, Ziming Liu, and Max Tegmark. Do Two AI Scientists Agree?, 2025. URL <http://arxiv.org/abs/2504.02822>.
- [1005] 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](https://doi.org/10.1371/journal.pcbi.1007279).
- [1006] 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.
- [1007] 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](https://doi.org/10.1007/s42452-021-04715-0).
- [1008] 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.
- [1009] 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](https://doi.org/10.1038/s41524-024-01423-2).
- [1010] Nathan J. Szymanski, Bernardus Rendy, Yuxing Fei, Rishi E. Kumar, Tanjin He, David Milsted, Matthew J. McDermott, Max Gallant, Ekin Dogus Cubuk, Amil Merchant, Haegyeom Kim, Anubhav Jain, Christopher J. Bartel, Kristin Persson, Yan Zeng, and Gerbrand Ceder. An autonomous laboratory for the accelerated synthesis of novel materials. *Nature*, 624(7990):86–91, 2023. ISSN 1476-4687. doi:[10.1038/s41586-023-06734-w](https://doi.org/10.1038/s41586-023-06734-w). URL <https://www.nature.com/articles/s41586-023-06734-w>.
- [1011] Aikaterini Vriza, Henry Chan, and Jie Xu. Self-Driving Laboratory for Polymer Electronics. *Chemistry of Materials*, 35(8):3046–3056, 2023. ISSN 0897-4756. doi:[10.1021/acs.chemmater.2c03593](https://doi.org/10.1021/acs.chemmater.2c03593). URL <https://doi.org/10.1021/acs.chemmater.2c03593>.
- [1012] Ali Essam Ghareeb, Benjamin Chang, Ludovico Mitchener, Angela Yiu, Caralyn J. Szostkiewicz, Jon M. Laurent, Muhammed T. Razzak, Andrew D. White, Michaela M. Hinks, and Samuel G. Rodrigues. Robin: A multi-agent system for automating scientific discovery, 2025. URL [http://arxiv.org/abs/2505.13400](https://arxiv.org/abs/2505.13400).
- [1013] Qianggang Ding, Santiago Miret, and Bang Liu. Matexpert: Decomposing materials discovery by mimicking human experts, 2024. URL <https://arxiv.org/abs/2410.21317>.
- [1014] Izumi Takahara, Teruyasu Mizoguchi, and Bang Liu. Accelerated inorganic materials design with generative ai agents, 2025. URL <https://arxiv.org/abs/2504.00741>.
- [1015] Karl Raimund Popper. *Conjectures and Refutations: The Growth of Scientific Knowledge*. Routledge, 1962.
- [1016] 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.
- [1017] Donald A. Gillies. Popper and computer induction. *BioEssays*, 23(9):859–860, 2001. ISSN 1521-1878. doi:[10.1002/bies.1123](https://doi.org/10.1002/bies.1123).
- [1018] 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.
- [1019] Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. Can LLMs Generate Novel Research Ideas? A Large-Scale Human Study with 100+ NLP Researchers, September 2024.
- [1020] 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](https://doi.org/10.1002/adma.202413523).
- [1021] 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.

- [1022] 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.
- [1023] 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](https://doi.org/10.1007/978-3-031-45275-8_42).
- [1024] 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.
- [1025] 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.
- [1026] 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.
- [1027] Anthony Coutant, Katherine Roper, Daniel Trejo-Banos, Dominique Bouthinon, Martin Carpenter, Jacek Grzebyta, Guillaume Santini, Henry Soldano, Mohamed Elati, Jan Ramon, Celine Rouveiro, 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](https://doi.org/10.1073/pnas.1900548116).
- [1028] 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.
- [1029] 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.
- [1030] 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, 2024.. URL <http://arxiv.org/abs/2406.16976>.
- [1031] Shuyi Jia, Chao Zhang, and Victor Fung. LLMatDesign: Autonomous Materials Discovery with Large Language Models, June 2024.
- [1032] 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](https://doi.org/10.1016/j.matt.2024.04.022).
- [1033] 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](https://doi.org/10.1039/D4DD00040D).
- [1034] Tianwei Dai, Sriram Vijayakrishnan, Filip T. Szczypinski, 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](https://doi.org/10.1038/s41586-024-08173-7).
- [1035] 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](https://doi.org/10.1126/science.adk9227).
- [1036] 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.
- [1037] 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.
- [1038] 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](https://doi.org/10.1126/science.ade2574).
- [1039] 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, Rishabh Jain, Ellen Clancy, Pushmeet Kohli, John Jumper, and Demis Hassabis. Protein complex prediction with AlphaFold-Multimer, October 2021.
- [1040] 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](https://doi.org/10.1002/prot.26422).
- [1041] 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](https://doi.org/10.1038/s41586-023-06747-5).
- [1042] 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.
- [1043] Yijia Shao, Yucheng Jiang, Theodore A. Kanell, Peter Xu, Omar Khattab, and Monica S. Lam. Assisting in writing wikipedia-like articles from scratch with large language models, 2024. URL <https://arxiv.org/abs/2402.14207>.
- [1044] Jiefu Ou, Arda Uzunoglu, Benjamin Van Durme, and Daniel Khashabi. WorldAPIs: The World Is Worth How Many APIs? A Thought Experiment, July 2024.
- [1045] 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](https://doi.org/10.1039/D4DD00129J).
- [1046] 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.
- [1047] Introducing the Model Context Protocol, 2025. URL <https://www.anthropic.com/news/model-context-protocol>.

- [1048] 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.
- [1049] Daniil A Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578, 2023.
- [1050] Emerald Cloud Lab. ECL Documentation. <https://www.emeraldcloudlab.com/documentation/objects/>, 2025.
- [1051] Jiafei Duan, Samson Yu, Hui Li Tan, Hongyuan Zhu, and Cheston Tan. A Survey of Embodied AI: From Simulators to Research Tasks, January 2022.
- [1052] 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.
- [1053] 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 μ s-resolved, AI-executable XPCS. *Light: Science & Applications*, 12(1):196, 2023.
- [1054] 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.
- [1055] 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.
- [1056] 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.
- [1057] 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 Järvinieniemi, 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.
- [1058] 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.
- [1059] 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.
- [1060] Zeyuan Allen-Zhu and Xiaoli Xu. DOGE: Reforming AI Conferences and Towards a Future Civilization of Fairness and Justice, February 2025.
- [1061] 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.

- [1062] 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](https://doi.org/10.1039/D2DD00087C).
- [1063] 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.
- [1064] 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.
- [1065] 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](https://doi.org/10.1038/nature17439).
- [1066] OpenAI. Introducing deep research. <https://openai.com/index/introducing-deep-research/>, 2025.
- [1067] 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](https://doi.org/10.1191/1740774505cn098oa).
- [1068] 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](https://doi.org/10.1007/s44336-024-00009-2). URL <https://doi.org/10.1007/s44336-024-00009-2>.
- [1069] James Surowiecki. The wisdom of crowds. *Surowiecki*, J, 2005.
- [1070] Chris Frith and Uta Frith. Theory of mind. *Current biology*, 15(17):R644–R645, 2005.
- [1071] 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.
- [1072] 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.
- [1073] 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>.
- [1074] 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, 2024.. URL <https://arxiv.org/abs/2408.15971>.
- [1075] 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.
- [1076] 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.

- [1077] 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.
- [1078] 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.
- [1079] 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.
- [1080] 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.
- [1081] Jiaqi Chen, Yuxian Jiang, Jiachen Lu, and Li Zhang. S-agents: self-organizing agents in open-ended environment. *arXiv preprint arXiv:2402.04578*, 2024.
- [1082] 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.
- [1083] 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>.
- [1084] 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>.
- [1085] Anthropic. Model context protocol, 2025. URL <https://www.anthropic.com/news/model-context-protocol>. Accessed: 2025-01-07.
- [1086] Gaowei Chang. Agentnetworkprotocol, 2025. URL <https://github.com/chgaowei/AgentNetworkProtocol>. GitHub repository, Accessed: 2025-01-07.
- [1087] Rao Surapaneni, Miku Jha, Michael Vakoc, and Todd Segal. A2a: A new era of agent interoperability. <https://developers.googleblog.com/en/a2a-a-new-era-of-agent-interoperability>, April 2025. Google Developers Blog.
- [1088] 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>.
- [1089] 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>.
- [1090] 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>.
- [1091] Dong Chen, Shaolin 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.
- [1092] Ziyi Yang, Zaixin 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.

- [1093] 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.
- [1094] 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>.
- [1095] Faria Huq, Zora Zhiruo Wang, Frank F. Xu, Tianyue Ou, Shuyan Zhou, Jeffrey P. Bigham, and Graham Neubig. Cowpilot: A framework for autonomous and human-agent collaborative web navigation, 2025. URL <https://arxiv.org/abs/2501.16609>.
- [1096] 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>.
- [1097] 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](https://doi.org/10.18653/V1/D18-1259). URL <https://doi.org/10.18653/v1/d18-1259>.
- [1098] 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>.
- [1099] Arkil Patel, Satwik Bhattacharya, and Navin Goyal. Are NLP models really able to solve simple math word problems? In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkan-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](https://doi.org/10.18653/V1/2021.NAACL-MAIN.168). URL <https://doi.org/10.18653/v1/2021.nacl-main.168>.
- [1100] Subhro Roy and Dan Roth. Solving general arithmetic word problems. *CoRR*, abs/1608.01413, 2016. URL <http://arxiv.org/abs/1608.01413>.
- [1101] 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>.
- [1102] 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>.
- [1103] 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.
- [1104] 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>.
- [1105] Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Zhe Wang, Zhenhai long 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>.

- [1106] 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.
- [1107] 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.
- [1108] 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.
- [1109] 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.
- [1110] Shaoguang Mao, Yuzhe Cai, Yan Xia, Wenshan Wu, Xun Wang, Fengyi Wang, Tao Ge, and Furu Wei. Olympics: Llm agents meet game theory—exploring strategic decision-making with ai agents. *arXiv preprint arXiv:2311.03220*, 2023.
- [1111] 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.
- [1112] 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>.
- [1113] Kanishk Gandhi, Dorsa Sadigh, and Noah D Goodman. Strategic reasoning with language models. *arXiv preprint arXiv:2305.19165*, 2023.
- [1114] 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.
- [1115] Nian Li, Chen Gao, Yong Li, and Qingmin Liao. Large language model-empowered agents for simulating macroeconomic activities. Available at SSRN 4606937, 2023.
- [1116] 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.
- [1117] 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.
- [1118] 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>.
- [1119] 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.
- [1120] 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.
- [1121] 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.
- [1122] 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.
- [1123] 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.

- [1124] 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.
- [1125] 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.
- [1126] 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.
- [1127] 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.
- [1128] 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>.
- [1129] 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>.
- [1130] 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.
- [1131] 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.
- [1132] 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.
- [1133] 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.
- [1134] 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.
- [1135] 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.
- [1136] 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.
- [1137] 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.
- [1138] 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.
- [1139] 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.

- [1140] 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.
- [1141] 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.
- [1142] 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.
- [1143] 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.
- [1144] Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbulin, and Jürgen Schmidhuber. Language agents as optimizable graphs. *arXiv preprint arXiv:2402.16823*, 2024.
- [1145] 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>.
- [1146] 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>.
- [1147] 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>.
- [1148] 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>.
- [1149] 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>.
- [1150] 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.
- [1151] 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>.
- [1152] 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.
- [1153] 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>.
- [1154] 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>.
- [1155] 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>.
- [1156] 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>.
- [1157] 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>.
- [1158] 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>.
- [1159] 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>.
- [1160] Suma Bailis, Jane Friedhoff, and Feiyang Chen. Werewolf arena: A case study in LLM evaluation via social deduction. *arXiv preprint arXiv:2407.13943*, 2024.
- [1161] 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>.
- [1162] Sumedh Rasal and E. J. Hauer. Navigating complexity: Orchestrated problem solving with multi-agent llms, 2024. URL <https://arxiv.org/abs/2402.16713>.
- [1163] 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.
- [1164] 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.
- [1165] 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.
- [1166] 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.
- [1167] 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.
- [1168] 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.
- [1169] 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.
- [1170] Yupeng Xie, Yuyu Luo, Guoliang Li, and Nan Tang. Haichart: Human and AI paired visualization system. *Proc. VLDB Endow.*, 17(11):3178–3191, 2024.
- [1171] 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.
- [1172] 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.

- [1173] 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.
- [1174] 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.
- [1175] 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.
- [1176] 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.
- [1177] 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.
- [1178] 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.
- [1179] Bowen Niu, Yixuan Song, Kun Lian, Yi Shen, Yuan Yao, et al. Flow: Modularized agentic workflow automation. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [1180] 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.
- [1181] 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.
- [1182] 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.
- [1183] 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.
- [1184] 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.
- [1185] 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>.
- [1186] 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.
- [1187] Huaben Chen, Wenkang Ji, Lufeng Xu, and Shiyu Zhao. Multi-agent consensus seeking via large language models. *arXiv preprint arXiv:2310.20151*, 2023.
- [1188] 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.
- [1189] Marios Papachristou, Longqi Yang, and Chin-Chia Hsu. Leveraging large language models for collective decision-making. *arXiv preprint arXiv:2311.04928*, 2023.

- [1190] 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.
- [1191] 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.
- [1192] 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.
- [1193] 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.
- [1194] 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.
- [1195] Yizhang Zhu, Runzhi Jiang, Boyan Li, Nan Tang, and Yuyu Luo. Elliesql: Cost-efficient text-to-sql with complexity-aware routing, 2025. URL <https://arxiv.org/abs/2503.22402>.
- [1196] 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.
- [1197] 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.
- [1198] Junzhi Chen, Juhao Liang, and Benyou Wang. Smurfs: Leveraging multiple proficiency agents with context-efficiency for tool planning, 2024.
- [1199] 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.
- [1200] Hojae Han, Seung won Hwang, Rajhans Samdani, and Yuxiong He. Convcodeworld: Benchmarking conversational code generation in reproducible feedback environments. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [1201] Shao Zhang, Xihuai Wang, Wenhao Zhang, Chaoran Li, Junru Song, Tingyu Li, Lin Qiu, Xuezhi Cao, Xunliang Cai, Wen Yao, Weinan Zhang, Xinbing Wang, and Ying Wen. Leveraging dual process theory in language agent framework for real-time simultaneous human-ai collaboration, 2025. URL <https://arxiv.org/abs/2502.11882>.
- [1202] 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>.
- [1203] Henry Peng Zou, Wei-Chieh Huang, Yaozu Wu, Yankai Chen, Chunyu Miao, Hoang Nguyen, Yue Zhou, Weizhi Zhang, Liancheng Fang, Langzhou He, Yangning Li, Dongyuan Li, Renhe Jiang, Xue Liu, and Philip S. Yu. A survey on large language model based human-agent systems, 2025. URL <https://arxiv.org/abs/2505.00753>.
- [1204] 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.

- [1205] 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.
- [1206] Philip Schroeder, Nathaniel Morgan, Hongyin Luo, and James Glass. Thread: Thinking deeper with recursive spawning. *arXiv preprint arXiv:2405.17402*, 2024.
- [1207] 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.
- [1208] Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Dyi Yang. A dynamic llm-powered agent network for task-oriented agent collaboration. In *First Conference on Language Modeling*, 2024.
- [1209] 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>.
- [1210] Yuxing Lu and Jinzhuo Wang. Karma: Leveraging multi-agent llms for automated knowledge graph enrichment, 2025. URL <https://arxiv.org/abs/2502.06472>.
- [1211] Chaoyun Zhang, Shilin He, Jiaxu Qian, Bowen Li, Liquan 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>.
- [1212] 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>.
- [1213] 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>.
- [1214] 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.
- [1215] 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.
- [1216] Xudong Yang, Yifan Wu, Yizhang Zhu, Nan Tang, and Yuyu Luo. Askchart: Universal chart understanding through textual enhancement. *arXiv preprint arXiv:2412.19146*, 2024.
- [1217] 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](https://doi.org/10.18653/v1/2023.emnlp-demo.15). URL <https://aclanthology.org/2023.emnlp-demo.15/>.
- [1218] Agntcy Foundation. Agntcy: An open protocol stack for the internet of agents. <https://docs.agntcy.org/index.html>, 2025. Accessed: 2025-06-23.
- [1219] Yingxuan Yang, Huacan Chai, Yuanyi Song, Siyuan Qi, Muning Wen, Ning Li, Junwei Liao, Haoyi Hu, Jianghao Lin, et al. A survey of ai agent protocols. *arXiv preprint arXiv:2504.16736*, 2025.
- [1220] Yuntao Wang, Shaolong Guo, Yanghe Pan, Zhou Su, Fahao Chen, Tom H. Luan, Peng Li, Jiawen Kang, and Dusit Niyato. Internet of agents: Fundamentals, applications, and challenges, 2025. URL <https://arxiv.org/abs/2505.07176>.
- [1221] 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.
- [1222] Guanghui Zhang, Kang Chen, Guohao Wan, Hao Chang, Hao Cheng, et al. Evoflow: Evolving diverse agentic workflows on the fly. *arXiv preprint arXiv:2502.07373*, 2025.
- [1223] Yiqi Wei, Zhaoyang Huang, Hanxian Li, Wenwei Xing, Tianji Lin, and Lei He. Vflow: Discovering optimal agentic workflows for verilog generation. *arXiv preprint arXiv:2504.03723*, 2025.
- [1224] Jingping Su, Yu Xia, Renjun Shi, Jiahui Wang, Jiawei Huang, Yu Wang, et al. Debfow: Automating agent creation via agent debate. *arXiv preprint arXiv:2503.23781*, 2025.

- [1225] [Author names to be confirmed] Dang. Multi-agent collaboration via evolving orchestration. *arXiv preprint*, 2025.
- [1226] Siyu Yuan et al. Evoagent: Towards automatic multi-agent generation via evolutionary algorithms. *arXiv preprint*, 2024.
- [1227] Rui Ye, Xiangru Liu, Qingyun Wu, Xinyuan Pang, Zhaopeng Yin, Lu Bai, et al. X-mas: Towards building multi-agent systems with heterogeneous llms. *arXiv preprint arXiv:2505.16997*, 2025.
- [1228] Rui Ye et al. Mas-gpt: Training llms to build llm-based multi-agent systems. *arXiv preprint*, 2025.
- [1229] Guanghui Zhang, Li Niu, Jianwei Fang, Kun Wang, Lu Bai, et al. Multi-agent architecture search via agentic supernet. *arXiv preprint arXiv:2502.04180*, 2025.
- [1230] Bowen Li, Ziyuan Zhao, Dong Hoon Lee, and Guanghui Wang. Adaptive graph pruning for multi-agent communication. *arXiv preprint arXiv:2506.02951*, 2025.
- [1231] Rui Ye et al. Masrouter: Intelligent routing for multi-agent systems. *arXiv preprint*, 2025.
- [1232] Dapeng Li, Ningyuan Lou, Zheyuan Xu, Bin Zhang, and Guangliang Fan. Efficient communication in multi-agent reinforcement learning with implicit consensus generation. In *AAAI Conference on Artificial Intelligence*, 2025.
- [1233] Guangchong Zhou, Zhiwei Zhang, and Guangliang Fan. Air: Unifying individual and collective exploration in cooperative multi-agent reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2025.
- [1234] Kaiqi Peng, Shaocong Zhu, and Teng Ma. Stpe-marl: Spatio-temporal multi-agent population evolution reinforcement learning. *ACM Transactions on Intelligent Systems and Technology*, 2025.
- [1235] Anastasia Sagirova, Yuri Kuratov, and Mikhail Burtsev. Shared memory for multi-agent lifelong pathfinding. In *International Conference on Learning Representations*, 2025.
- [1236] Shengbin Yue, Siyuan Wang, Weihua Chen, Xuanjing Huang, and Zhongyu Wei. Synergistic multi-agent framework with trajectory learning for knowledge-intensive tasks. In *AAAI Conference on Artificial Intelligence*, 2025.
- [1237] Zixuan Zhang, Hao Zhou, Mahsa Imani, Taesung Lee, and Tian Lan. Learning to collaborate with unknown agents in the absence of reward. In *AAAI Conference on Artificial Intelligence*, 2025.
- [1238] 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.
- [1239] 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.
- [1240] 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.
- [1241] Xun Jiang, Feng Li, Haoyu Zhao, et al. Long term memory: The foundation of ai self-evolution. *arXiv preprint arXiv:2410.15665*, 2024.
- [1242] Ceyao Zhang, Kajie 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.
- [1243] Emanuele Pesce and Giovanni Montana. Improving coordination in small-scale multi-agent deep reinforcement learning through memory-driven communication. *Machine Learning*, 2020.
- [1244] [Author names to be confirmed] Zhang. G-memory: A multi-agent shared memory system with hierarchical graphs. *[Journal/Conference to be confirmed]*, 2025.

- [1245] Amirhossein Rezazadeh, Zhaoyang Li, Aaron Lou, Yifan Zhao, Wei Wei, et al. Collaborative memory: Multi-user memory sharing in llm agents with dynamic access control. *arXiv preprint arXiv:2505.18279*, 2025.
- [1246] Kuan Wang, Yadong Lu, Michael Santacroce, Yeyun Gong, Chao Zhang, and Yelong Shen. Adapting LLM agents through communication. *arXiv preprint arXiv:2310.01444*, 2023.
- [1247] 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.
- [1248] Wanja Zhao, Mert Yuksekgonul, Shirley Wu, and James Zou. Sirius: Self-improving multi-agent systems via bootstrapped reasoning. *arXiv preprint arXiv:2502.04780*, 2025.
- [1249] 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.
- [1250] 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.
- [1251] 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.
- [1252] 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>.
- [1253] 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, Fofios 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>.
- [1254] 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>.
- [1255] 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_.

- [1256] 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>.
- [1257] 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](https://doi.org/10.18653/V1/2023.FINDINGS-EMNLP.89). URL <https://doi.org/10.18653/v1/2023.findings-emnlp.89>.
- [1258] 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.
- [1259] 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>.
- [1260] 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](https://doi.org/10.18653/V1/N19-1421). URL <https://doi.org/10.18653/v1/n19-1421>.
- [1261] 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](https://doi.org/10.1162/TACL_A_00370). URL https://doi.org/10.1162/tacl_a_00370.
- [1262] 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](https://doi.org/10.18653/V1/N19-1300). URL <https://doi.org/10.18653/v1/n19-1300>.
- [1263] 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](https://doi.org/10.18653/V1/D18-1260). URL <https://doi.org/10.18653/v1/d18-1260>.
- [1264] 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](https://doi.org/10.1145/3474381). URL <https://doi.org/10.1145/3474381>.
- [1265] 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](https://doi.org/10.18653/v1/p19-1472). URL <https://doi.org/10.18653/v1/p19-1472>.
- [1266] Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social iqqa: 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](https://doi.org/10.18653/V1/D19-1454). URL <https://doi.org/10.18653/V1/D19-1454>.
- [1267] 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](https://doi.org/10.1609/aaai.v34i05.6239). URL <https://doi.org/10.1609/aaai.v34i05.6239>.
- [1268] 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>.
- [1269] 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](https://doi.org/10.1007/S00799-022-00329-Y). URL <https://doi.org/10.1007/S00799-022-00329-Y>.
- [1270] 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>.
- [1271] 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>.
- [1272] 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>.
- [1273] 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](https://doi.org/10.18653/V1/N19-1245). URL <https://doi.org/10.18653/v1/n19-1245>.
- [1274] 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](https://doi.org/10.18653/V1/P17-1015). URL <https://doi.org/10.18653/V1/P17-1015>.
- [1275] 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](https://doi.org/10.18653/v1/n16-1136). URL <https://doi.org/10.18653/v1/n16-1136>.
- [1276] 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](https://doi.org/10.18653/v1/n19-1246). URL <https://doi.org/10.18653/v1/n19-1246>.
 - [1277] 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>.
 - [1278] 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>.
 - [1279] 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>.
 - [1280] Yang Liu, Peng Sun, and Hang Li. Large language models as agents in two-player games, 2024. URL <https://arxiv.org/abs/2402.08078>.
 - [1281] 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>.
 - [1282] Yohei Nakajima. Babyagi arena, 2023. URL <https://github.com/yoheinakajima/babyagi-arena>.
 - [1283] 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>.
 - [1284] 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>.
 - [1285] 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>.
 - [1286] 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>.
 - [1287] 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>.
 - [1288] 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>.
 - [1289] Siham 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>.

- [1290] 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>.
- [1291] 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>.
- [1292] 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>.
- [1293] 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>.
- [1294] Han Wang, Binbin Chen, Tieying Zhang, and Baoxiang Wang. Learning to communicate through implicit communication channels, 2025. URL <https://arxiv.org/abs/2411.01553>.
- [1295] 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>.
- [1296] Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*, 2023.
- [1297] Yuntong Zhang, Haifeng Ruan, Zhiyu Fan, and Abhik Roychoudhury. Autocoderover: Autonomous program improvement, 2024. URL <https://arxiv.org/abs/2404.05427>.
- [1298] Dong Chen, Shaolin Lin, Muhan Zeng, Daoguang Zan, Jian-Gang Wang, Anton Cheshkov, Jun Sun, Hao Yu, Guoliang Dong, Artem Aliev, Jie Wang, Xiao Cheng, Guangtai Liang, Yuchi Ma, Pan Bian, Tao Xie, and Qianxiang Wang. Coder: Issue resolving with multi-agent and task graphs, 2024. URL <https://arxiv.org/abs/2406.01304>.
- [1299] 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](https://doi.org/10.18653/V1/P17-1147). URL <https://doi.org/10.18653/v1/P17-1147>.
- [1300] 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.
- [1301] Rui Ye, Keduan Huang, Qimin Wu, Yuzhu Cai, Tian Jin, Xianghe Pang, Xiangrui Liu, Jiaqi Su, Chen Qian, Bohan Tang, et al. Maslab: A unified and comprehensive codebase for llm-based multi-agent systems. *arXiv preprint arXiv:2505.16988*, 2025.
- [1302] 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>.
- [1303] 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>.
- [1304] 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>.

- [1305] 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>.
- [1306] 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.
- [1307] Zhiqiang Lin, Huan Sun, and Ness Shroff. Ai safety vs. ai security: Demystifying the distinction and boundaries. *arXiv preprint arXiv:2506.18932*, 2025.
- [1308] 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.
- [1309] Andy Zou, Zifan Wang, Norman Mu, and Jacob Andreas. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.
- [1310] Yihao Zhang and Zeming Wei. Boosting jailbreak attack with momentum. *arXiv preprint arXiv:2405.01229*, 2024.
- [1311] 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.
- [1312] 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.
- [1313] 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.
- [1314] Leyang Hu and Boran Wang. DROJ: A prompt-driven attack against large language models. *arXiv preprint arXiv:2411.09125*, 2024.
- [1315] Xiaogeng Liu, Nan Xu, Muhamo Chen, and Chaowei Xiao. Autodan: Generating stealthy jailbreak prompts on aligned large language models. *arXiv preprint arXiv:2310.04451*, 2023.
- [1316] Xuancun Lu, Zhengxian Huang, Xinfeng Li, Xiaoyu Ji, and Wenyuan Xu. Poex: Policy executable embodied AI jailbreak attacks. *arXiv preprint arXiv:2412.16633*, 2024.
- [1317] Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does LLM safety training fail? *Advances in Neural Information Processing Systems*, 36, 2023.
- [1318] 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.
- [1319] 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.
- [1320] 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.
- [1321] 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.
- [1322] 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.

- [1323] 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.
- [1324] Jeremy Kritz, Vaughn Robinson, Robert Vacareanu, Bijan Varjavand, Michael Choi, Bobby Gogov, Scale Red Team, Summer Yue, Willow E Primack, and Zifan Wang. Jailbreaking to jailbreak. *arXiv preprint arXiv:2502.09638*, 2025.
- [1325] 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.
- [1326] 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.
- [1327] 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.
- [1328] 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.
- [1329] Johann Rehberger. Trust no ai: Prompt injection along the cia security triad. *arXiv preprint arXiv:2412.06090*, 2024.
- [1330] 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.
- [1331] Edoardo Debenedetti, Javier Rando, Daniel Paleka, Silaghi Fineas Florin, Dragos Albastroiu, Niv Cohen, Yuval Lemberg, Reshma 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.
- [1332] Sander Schulhoff, Jeremy Pinto, Anaum Khan, Louis-François Bouchard, Chenglei Si, Svetlana 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.
- [1333] 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.
- [1334] 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.
- [1335] Donghyun Lee and Mo Tiwari. Prompt Infection: LLM-to-LLM Prompt Injection within Multi-Agent Systems. *arXiv preprint arXiv:2410.07283*, 2024.
- [1336] Fredrik Nestaas, Edoardo Debenedetti, and Florian Tramèr. Adversarial search engine optimization for large language models. *arXiv preprint arXiv:2406.18382*, 2024.
- [1337] 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.
- [1338] 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.

- [1339] 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.
- [1340] 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.
- [1341] 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.
- [1342] Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv:2401.11817*, 2024.
- [1343] 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.
- [1344] 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.
- [1345] 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.
- [1346] Zhiying Zhu, Zhiqing Sun, and Yiming Yang. Halueval-wild: Evaluating hallucinations of language models in the wild. *arXiv preprint arXiv:2403.04307*, 2024.
- [1347] 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.
- [1348] 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.
- [1349] Leonardo Ranaldi and Giulia Pucci. When large language models contradict humans? large language models' sycophantic behaviour. *arXiv preprint arXiv:2311.09410*, 2023.
- [1350] 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.
- [1351] 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.
- [1352] 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.
- [1353] 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/>.
- [1354] Richard Ngo, Lawrence Chan, and Sören Mindermann. The alignment problem from a deep learning perspective. *arXiv preprint arXiv:2209.00626*, 2022.

- [1355] 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.
- [1356] 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.
- [1357] 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.
- [1358] 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.
- [1359] 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.
- [1360] 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.
- [1361] 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.
- [1362] 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.
- [1363] 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.
- [1364] 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.
- [1365] 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.
- [1366] 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.
- [1367] 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.
- [1368] Abdullah Arafat Miah and Yu Bi. Exploiting the vulnerability of large language models via defense-aware architectural backdoor. *arXiv preprint arXiv:2409.01952*, 2024.
- [1369] 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.
- [1370] 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.

- [1371] 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.
- [1372] 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.
- [1373] 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.
- [1374] 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.
- [1375] 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.
- [1376] 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.
- [1377] 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.
- [1378] 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.
- [1379] 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.
- [1380] 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.
- [1381] 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.
- [1382] 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.
- [1383] 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.
- [1384] 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.
- [1385] 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.
- [1386] 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.

- [1387] 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.
- [1388] 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.
- [1389] 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.
- [1390] Zhixin 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.
- [1391] John X Morris, Wenting Zhao, Justin T Chiu, Vitaly Shmatikov, and Alexander M Rush. Language model inversion. *arXiv preprint arXiv:2311.13647*, 2023.
- [1392] 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.
- [1393] 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.
- [1394] 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.
- [1395] Yash More, Prakhar Ganesh, and Golnoosh Farnadi. Towards more realistic extraction attacks: An adversarial perspective. *arXiv preprint arXiv:2407.02596*, 2024.
- [1396] 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.
- [1397] 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.
- [1398] Zhifeng Jiang, Zhihua Jin, and Guoliang He. Safeguarding system prompts for llms. *arXiv preprint arXiv:2412.13426*, 2024.
- [1399] 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.
- [1400] Yiming Zhang, Nicholas Carlini, and Daphne Ippolito. Effective prompt extraction from language models. *arXiv preprint arXiv:2307.06865*, 2023.
- [1401] 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.
- [1402] 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.
- [1403] 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.
- [1404] 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.
- [1405] 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.

- [1406] Zi Liang, Haibo Hu, Qingqing Ye, Yaxin Xiao, and Haoyang Li. Why are my prompts leaked? unravelling prompt extraction threats in customized large language models. *arXiv preprint arXiv:2408.02416*, 2024.
- [1407] 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.
- [1408] Itay Yona, Ilia Shumailov, Jamie Hayes, and Nicholas Carlini. Stealing user prompts from mixture of experts. *arXiv preprint arXiv:2410.22884*, 2024.
- [1409] 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.
- [1410] Zhengchun Shang and Wenlan Wei. Evolving security in llms: A study of jailbreak attacks and defenses. *arXiv preprint arXiv:2504.02080*, 2025.
- [1411] 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.
- [1412] Peiyan Zhang, Haibo Jin, Liying Kang, and Haohan Wang. Guardval: Dynamic large language model jailbreak evaluation for comprehensive safety testing. *arXiv preprint arXiv:2507.07735*, 2025.
- [1413] 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.
- [1414] Alexander Robey, Eric Wong, Hamed Hassani, and George J Pappas. SmoothLLM: Defending large language models against jailbreaking attacks. *arXiv preprint arXiv:2310.03684*, 2023.
- [1415] 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>.
- [1416] 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.
- [1417] 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.
- [1418] 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.
- [1419] 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.
- [1420] Md Ahsan Ayub and Subhabrata Majumdar. Embedding-based classifiers can detect prompt injection attacks. *arXiv preprint arXiv:2410.22284*, 2024.
- [1421] Sizhe Chen, Julien Piet, Chawin Sitawarin, and David Wagner. Struq: Defending against prompt injection with structured queries. *arXiv preprint arXiv:2402.06363*, 2024.
- [1422] 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.
- [1423] 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.

- [1424] 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.
- [1425] Rongwu Xu, Brian Lin, Shujian Yang, Tianqi Zhang, Weiyuan 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.
- [1426] 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.
- [1427] 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.
- [1428] 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.
- [1429] 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.
- [1430] 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.
- [1431] 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.
- [1432] 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.
- [1433] 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.
- [1434] 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.
- [1435] 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.
- [1436] 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.
- [1437] 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.
- [1438] 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.

- [1439] 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.
- [1440] 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.
- [1441] 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.
- [1442] Wenjie Mo, Jiashu Xu, Qin Liu, Jiong Xiao Wang, Jun Yan, Chaowei Xiao, and Muhan Chen. Test-time backdoor mitigation for black-box large language models with defensive demonstrations. *arXiv preprint arXiv:2311.09763*, 2023.
- [1443] 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.
- [1444] 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.
- [1445] 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.
- [1446] 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.
- [1447] 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.
- [1448] Michael Duan, Anshuman Suri, Niloofar Mireshghallah, Sewon Min, Weijia Shi, Luke Zettlemoyer, Yulia Tsvetkov, Yejin Choi, David Evans, and Hannaneh Hajishirzi. Do membership inference attacks work on large language models? *arXiv preprint arXiv:2402.07841*, 2024.
- [1449] 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.
- [1450] 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.
- [1451] 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.
- [1452] Panlong Wu, Kangshuo Li, Junbao Nan, and Fangxin Wang. Federated in-context llm agent learning. *arXiv preprint arXiv:2412.08054*, 2024.
- [1453] 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.
- [1454] 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.

- [1455] 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.
- [1456] Antonio Muñoz, Ruben Ríos, Rodrigo Román, and Javier López. A survey on the (in) security of trusted execution environments. *Computers & Security*, 129:103180, 2023.
- [1457] 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.
- [1458] 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.
- [1459] 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.
- [1460] 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.
- [1461] 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.
- [1462] 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.
- [1463] 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.
- [1464] 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.
- [1465] 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.
- [1466] 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.
- [1467] 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.
- [1468] 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.
- [1469] 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.
- [1470] 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.

- [1471] Mansi Phute, Alec Helbling, Matthew Daniel Hull, ShengYun Peng, Sebastian Szryler, 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.
- [1472] Zhixin 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.
- [1473] 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.
- [1474] 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.
- [1475] 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.
- [1476] 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.
- [1477] 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.
- [1478] Elias Abad Rocamora, Yongtao Wu, Fanghui Liu, Grigoris Chrysos, and Volkan Cevher. Revisiting character-level adversarial attacks for language models. In *41st International Conference on Machine Learning (ICML 2024)*, 2024.
- [1479] 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.
- [1480] 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.
- [1481] 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.
- [1482] 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.
- [1483] 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.
- [1484] 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.
- [1485] 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.

- [1486] 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.
- [1487] 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.
- [1488] 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.
- [1489] 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.
- [1490] 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.
- [1491] 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.
- [1492] 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.
- [1493] 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.
- [1494] 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.
- [1495] 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.
- [1496] 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.
- [1497] 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.
- [1498] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. Dolphintack: Inaudible voice commands. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 103–117, 2017.
- [1499] 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.
- [1500] 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.
- [1501] 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.

- [1502] 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.
- [1503] 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.
- [1504] 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.
- [1505] Divyat Mahajan, Shruti Tople, and Amit Sharma. Domain generalization using causal matching. In *International conference on machine learning*, pages 7313–7324. PMLR, 2021.
- [1506] 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.
- [1507] 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.
- [1508] Daniele Vilone and Eugenia Polizzi. Modeling opinion misperception and the emergence of silence in online social system. *Plos one*, 19(1):e0296075, 2024.
- [1509] 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.
- [1510] Heechang Ryu, Hayong Shin, and Jinkyoo Park. Cooperative and competitive biases for multi-agent reinforcement learning. *arXiv preprint arXiv:2101.06890*, 2021.
- [1511] 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.
- [1512] 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.
- [1513] 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.
- [1514] 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.
- [1515] 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.
- [1516] 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.
- [1517] 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.

- [1518] 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.
- [1519] 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.
- [1520] Kai Li, Can Shen, Yile Liu, Jirui Han, Kelong Zheng, Xuechao Zou, Zhe Wang, Xingjian Du, Shun Zhang, Hanjun Luo, et al. Audiotrust: Benchmarking the multifaceted trustworthiness of audio large language models. *arXiv preprint arXiv:2505.16211*, 2025.
- [1521] 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.
- [1522] 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.
- [1523] Xiaoyu Ji, Wenjun Zhu, Shilin Xiao, and Wenyuan Xu. Sensor-based iot data privacy protection. *Nature Reviews Electrical Engineering*, 1(7):427–428, 2024.
- [1524] 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.
- [1525] Pedro Mendes, Paolo Romano, and David Garlan. Error-driven uncertainty aware training. *arXiv preprint arXiv:2405.01205*, 2024.
- [1526] 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.
- [1527] 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.
- [1528] 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.
- [1529] 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.
- [1530] 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.
- [1531] 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.
- [1532] 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.
- [1533] Yulin Chen, Haoran Li, Yuan Sui, Yufei He, Yue Liu, Yangqiu Song, and Bryan Hooi. Can indirect prompt injection attacks be detected and removed? *arXiv preprint arXiv:2502.16580*, 2025.

- [1534] 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.
- [1535] Ayush RoyChowdhury, Mulong Luo, Prateek Sahu, Sarbartha Banerjee, and Mohit Tiwari. Confused-pilot: Confused deputy risks in rag-based llms. *arXiv preprint arXiv:2408.04870*, 2024.
- [1536] 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.
- [1537] Avital Shafran, Roei Schuster, and Vitaly Shmatikov. Machine against the rag: Jamming retrieval-augmented generation with blocker documents. *arXiv preprint arXiv:2406.05870*, 2024.
- [1538] 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.
- [1539] 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.
- [1540] Quanyu Long, Yue Deng, LeiLei Gan, Wenyu Wang, and Sinno Jialin Pan. Whispers in grammars: Injecting covert backdoors to compromise dense retrieval systems. *arXiv preprint arXiv:2402.13532*, 2024.
- [1541] 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.
- [1542] Kurt Geihs. Engineering challenges ahead for robot teamwork in dynamic environments. *Applied Sciences*, 10(4):1368, 2020.
- [1543] 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.
- [1544] 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.
- [1545] 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.
- [1546] 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.
- [1547] 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.
- [1548] 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>.
- [1549] 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.
- [1550] 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.

- [1551] 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.
- [1552] 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.
- [1553] 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.
- [1554] Weidi Luo, Shenghong Dai, Xiaogeng Liu, Suman Banerjee, Huan Sun, Muha Chen, and Chaowei Xiao. Agrail: A lifelong agent guardrail with effective and adaptive safety detection. *arXiv preprint arXiv:2502.11448*, 2025.
- [1555] Lewis Hammond, Alan Chan, Jesse Clifton, Jason Hoelscher-Obermaier, Akbir Khan, Euan McLean, Chandler Smith, Wolfram Barfuss, Jakob Foerster, Tomáš Gavenčík, et al. Multi-agent risks from advanced ai. *arXiv preprint arXiv:2502.14143*, 2025.
- [1556] Aidan O’Gara. Hoodwinked: Deception and cooperation in a text-based game for language models. *arXiv preprint arXiv:2308.01404*, 2023.
- [1557] 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.
- [1558] 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.
- [1559] 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.
- [1560] Yikang Pan, Liangming Pan, Wenhua 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.
- [1561] 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.
- [1562] 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.
- [1563] Ada Defne Tur, Nicholas Meade, Xing Han Lù, Alejandra Zambrano, Arkil Patel, Esin Durmus, Spandana Gella, Karolina Stańczak, and Siva Reddy. Safearena: Evaluating the safety of autonomous web agents. *arXiv preprint arXiv:2503.04957*, 2025.
- [1564] Leo Boisvert, Mihir Bansal, Chandra Kiran Reddy Evuru, Gabriel Huang, Abhay Puri, Avinandan Bose, Maryam Fazel, Quentin Cappart, Jason Stanley, Alexandre Lacoste, et al. Doomarena: A framework for testing ai agents against evolving security threats. *arXiv preprint arXiv:2504.14064*, 2025.
- [1565] Ivan Evtimov, Arman Zharmagambetov, Aaron Grattafiori, Chuan Guo, and Kamalika Chaudhuri. Wasp: Benchmarking web agent security against prompt injection attacks. *arXiv preprint arXiv:2504.18575*, 2025.
- [1566] Zeyi Liao, Jaylen Jones, Linxi Jiang, Eric Fosler-Lussier, Yu Su, Zhiqiang Lin, and Huan Sun. Redteamcua: Realistic adversarial testing of computer-use agents in hybrid web-os environments. *arXiv preprint arXiv:2505.21936*, 2025.

- [1567] 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.
- [1568] Boyuan Zheng, Zeyi Liao, Scott Salisbury, Zeyuan Liu, Michael Lin, Qinyuan Zheng, Zifan Wang, Xiang Deng, Dawn Song, Huan Sun, and Yu Su. Webguard: Building a generalizable guardrail for web agents. *arXiv preprint arXiv:2507.14293*, 2025.
- [1569] 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.
- [1570] Hanjun Luo, Shenyu Dai, Chiming Ni, Xinfeng Li, Guibin Zhang, Kun Wang, Tongliang Liu, and Hanan Salam. Agentauditor: Human-level safety and security evaluation for llm agents. *arXiv preprint arXiv:2506.00641*, 2025.
- [1571] Zhaorun Chen, Mintong Kang, and Bo Li. Shieldagent: Shielding agents via verifiable safety policy reasoning. *arXiv preprint arXiv:2503.22738*, 2025.
- [1572] 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.
- [1573] Kai Chen, Taihang Zhen, Hewei Wang, Kailai Liu, Xinfeng Li, Jing Huo, Tianpei Yang, Jinfeng Xu, Wei Dong, and Yang Gao. Medsentry: Understanding and mitigating safety risks in medical llm multi-agent systems. *arXiv preprint arXiv:2505.20824*, 2025.
- [1574] 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.
- [1575] Jiyian 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.
- [1576] 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.
- [1577] 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.
- [1578] OpenAI. Introducing superalignment. <https://openai.com/index/introducing-superalignment/>, July 2023. Accessed: 2025-03-26.
- [1579] Eliezer Yudkowsky. Ai alignment: Why it's hard, and where to start. *Symbolic Systems Distinguished Speaker Series*, 2016.
- [1580] David Krueger, Tegan Maharaj, and Jan Leike. Hidden incentives for auto-induced distributional shift. *arXiv preprint arXiv:2009.09153*, 2020.
- [1581] Joseph Carlsmith. Is power-seeking ai an existential risk? *arXiv preprint arXiv:2206.13353*, 2022.
- [1582] 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.
- [1583] 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.

- [1584] Feifei Zhao, Yuwei Wang, Enmeng Lu, Dongcheng Zhao, Bing Han, Haibo Tong, Yao Liang, Dongqi Liang, Kang Sun, Lei Wang, et al. Redefining superalignment: From weak-to-strong alignment to human-ai co-alignment to sustainable symbiotic society. *arXiv preprint arXiv:2504.17404*, 2025.
- [1585] Zhengyang Tang, Ziniu Li, Zhenyang Xiao, Tian Ding, Ruoyu Sun, Benyou Wang, Dayiheng Liu, Fei Huang, Tianyu Liu, Bowen Yu, et al. Enabling scalable oversight via self-evolving critic. *arXiv preprint arXiv:2501.05727*, 2025.
- [1586] Minlie Huang, Yingkang Wang, Shiyao Cui, Pei Ke, and Jie Tang. The superalignment of superhuman intelligence with large language models. *Science China Information Sciences*, 68(6):1–11, 2025.
- [1587] 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.
- [1588] 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.
- [1589] 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.
- [1590] Jixuan Leng, Chongsong Huang, Banghua Zhu, and Jiaxin Huang. Taming overconfidence in llms: Reward calibration in rlhf. *arXiv preprint arXiv:2410.09724*, 2024.
- [1591] 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.
- [1592] 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.
- [1593] 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.
- [1594] 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.
- [1595] 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.
- [1596] 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.
- [1597] 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.
- [1598] 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.
- [1599] 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.

- [1600] Zaifan Jiang, Xing Huang, and Chao Wei. Preference as reward, maximum preference optimization with importance sampling. *arXiv preprint arXiv:2312.16430*, 2023.
- [1601] Sayak Ray Chowdhury, Anush Kini, and Nagarajan Natarajan. Provably robust dpo: Aligning language models with noisy feedback. *arXiv preprint arXiv:2403.00409*, 2024.
- [1602] 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.
- [1603] Yang Chao, Lu Chaochao, Wang Yingchun, and Zhou Bowen. Towards AI-45° law: A roadmap to trustworthy AGI. *arXiv preprint arXiv:2412.14186*, 2024.
- [1604] Artificial Analysis. LLM Leaderboard - Compare GPT-4o, Llama 3, Mistral, Gemini & other models, 2024. URL <https://artificialanalysis.ai/leaderboards/models>.
- [1605] 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.
- [1606] Kai Li and Yi Luo. Subnetwork-to-go: Elastic neural network with dynamic training and customizable inference. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6775–6779. IEEE, 2024.