Language Agent as Tool-use Decision Maker

WANG, Hongru

A Thesis Submitted in Partial Fulfillment

of the Requirements for the Degree of

Doctor of Philosophy

in

Systems Engineering and Engineering Management

The Chinese University of Hong Kong

July 2025

Thesis Assessment Committee

Professor YU Jeffrey Xu (Chair)

Professor WONG Kam Fai William (Thesis Supervisor)

Professor WAI Hoi To (Committee Member)

Professor LIU Zhiyuan (External Examiner)

Abstract

Language Agents (LA) are capable to reason and act in various contexts to solve complex tasks in practice, such as OpenAI Deep Research, Manus and Alita. While these systems exhibit increasing levels of autonomy, several fundamental questions remain inadequately addressed in current research: What is a language agent? What constitutes its desired behavior? And how can such behavior be realized in practice?

This thesis formalizes a systematic framework that models an agent as a tool-use decision maker. The agent not only interacts with external environments but also monitors its own internal knowledge state to make informed tool-use decisions and achieve its goals. Specifically, we first introduce a straightforward and universal principle that categorizes all types of interactions or behaviors of agents into two fundamental types: internal cognitive tool usage and external physical tool usage. By integrating these two typical tools with language agents, it is capable to deliver the pre-defined goal with greater efficiency and adaptability. The thesis is further developed in two parts: methods and benchmarks.

Part I lays the methodological foundation of unification of internal cognitive tools and external physical tools for language agents, and further formulate tool-integrated language agent as a tool-use decision maker to gain enough knowledge to achieve the pre-defined goal. The core idea is augment language agents with the utilization and planning capabilities of internal cognitive tools, such as reflection and chain-of-thoughts (CoT), alongside external physical tools, such as retriever and calculator, leading to more autonomous actions and helpful responses. Furthermore, we explore several methods for managing complementary or overlap tools, ensuring that agents can make informed decision at each time step with

the alignment between knowledge boundary and decision boundary. Inspired by meta-reasoning theory, the unified framework demonstrates how language agents can achieve a better trade-off between effectiveness and efficiency when tackling complex tasks via prompt engineering, supervised fine-tuning and reinforcement learning. We additionally discuss the different behaviors of agents to achieve the pre-defined goal and present an actionable roadmap to optimize such behavior of agents.

Part II highlights several practical challenges faced by existing language agents to interact with the world through the introduction of three benchmarks. These benchmarks focus on single-turn and multi-turn interactions, testing the agents' or LLMs' ability to handle complex tool structures, stateful environments and personalized utilization. By simulating real-world scenarios, such as managing workflows in complicated ecosystems and meeting the preference for diverse users, these benchmarks propose new challenges and shed light for future development of language agents.

Combining theoretical innovation with practical validation, this thesis provides a significant step toward developing more autonomous, personalized and intelligent language agents. The well-grounded framework of agent and the associated benchmarks establish a strong foundation for future research, paving the way for agents capable of seamlessly integrating cognitive tools with external physical tools to navigate increasingly complex environments.

摘要

語言智能體能夠在各種情況下通過推理和行動來解決實踐中的複雜任務，例如OpenAI Deep Research、Manus 和Alita。雖然這些系統表現出越來越高的自主性，但目前的研究仍然沒有充分解決幾個基本問題：什麼是語言智能體？他應該表現出什麼樣的行為模式？而這種行為在實踐中又該如何實現呢？

本論文提出了一個系統框架，將語言智能體建模為工具使用決策者。其不僅需要與外部環境交互，還需要監控其自身的內部知識狀態，以做出明智的工具使用決策並實現其預定的目標。具體來說，我們首先介紹一個簡單而普遍的原則，將語言智能體的所有類型的互動或行為分為兩種基本類型：內部認知工具的使用和外部物理工具的使用。透過將這兩種典型工具與語言智能體結合，它能夠以更高的效率和自主性來完成預定的目標。論文進一步分為兩部分：方法和基準。

第一部分為語言智能體利用內部認知工具和外部物理工具奠定了方法論基礎，將代理定義為工具使用決策者來獲取足夠的知識以實現預定目標。其核心思想是增强語言智能體對內部認知工具（例如反思，思維鏈）以及外部物理工具（包括檢索器，計算器等）的使用和規劃能力，從而實現更自主的行動和更用幫助的回復。此外，我們還探索了幾種管理互補或相似功能工具的方法，確保語言智能體能夠在知識邊界和決策邊界保持一致的情況下在每個時間步驟做出明智的決策。受元推理理論的啓發，統一框架展示了語言智能體如何透過提示工程、監督微調和强化學習在處理複雜任務時實現效果和效率之間的更好權衡。我們也討論了語言智能體為實現預定義目標而展現的不同行為模式，並提出了優化語言智能體行為的可行方案。

第二部分透過介紹三個基準，重點介紹了現有語言智能體與世界互動所面臨的幾個實際挑戰。這些基準測試著重於單輪和多輪交互，測試大語言模型或者語言智能體處理複雜工具

結構、有狀態環境和個性化工具使用的能力。透過模擬現實世界的場景，例如管理複雜生態系統中的工作流程以及滿足不同用戶的偏好，這些基準提出了新的挑戰，並為語言智能體的未來發展提供了啓示。

本論文將理論創新與實踐驗證結合，為開發更自主、個性化和智能的語言智能體邁出了重要一步。語言智能體的系統性框架和相關基準為未來的研究奠定了堅實的基礎，為其能夠更好地將認知工具與外部物理工具整合以面臨日益複雜的環境鋪平了道路。

Preface

The research works in this thesis have been published in the following peer-review journal and conferences proceedings.

- Hongru WANG, Boyang Xue, Baohang Zhou, Tianhua Zhang, Cunxiang Wang, Huimin Wang, Guanhua Chen, Kam-Fai Wong. 2025. Self-DC: When to Reason and When to Act? Self Divide-and-Conquer for Compositional Unknown Questions. In Proceedings of the NAACL 2025
- Hongru WANG, Deng Cai, Wanjun Zhong, Shijue Huang, Jeff Z. Pan, Zeming Liu, Kam-Fai Wong. Self-Reasoning Language Models: Unfold Hidden Reasoning Chains with Few Reasoning Catalyst. In Findings of ACL 2025
- Hongru WANG, Wenyu Huang, Yufei Wang, Yuanhao Xi, Jianqiao Lu, Huan Zhang, Nan Hu, Zeming Liu, Jeff Z. Pan, Kam-Fai Wong. Rethinking Stateful Tool Use in Multi-Turn Dialogues: Benchmarks and Challenges. In Findings of ACL 2025
- Zihao Cheng*, Hongru WANG*, Zeming Liu, Yuhang Guo, Yuanfang Guo, Yunhong Wang, Haifeng Wang. ToolSpectrum: Towards Personalized Tool Utilization for Large Language Models. In Findings of ACL 2025
- Hongru WANG, Rui Wang, Boyang Xue, Heming Xia, Jingtao Cao, Zeming Liu, Jeff Z. Pan, Kam-Fai Wong. AppBench: Planning of Multiple APIs from Various APPs for Complex User Instruction. In Proceedings of EMNLP 2024

- Hongru Wang, Yujia Qin, Yankai Lin, Jeff Z. Pan, Kam-Fai Wong. Empowering Large Language Models: Tool Learning for Real-World Interaction. In Proceedings of SIGIR 2024, Tutorial.

- Hongru WANG, Boyang Xue, Baohang Zhou, Rui Wang, Fei Mi, Weichao Wang, Yasheng Wang, Kam-Fai Wong. UniRetriever: Multi-task Candidates Selection for Various Context-Adaptive Conversational Retrieval. In Proceedings of LREC-COLING 2024

- Hongru WANG, Rui Wang, Fei Mi, Yang Deng, Zezhong Wang, Bin Liang, Ruifeng Xu, Kam-Fai Wong. Cue-CoT: Chain-of-thought Prompting for Responding to In-depth Dialogue Questions with LLMs. In Findings of EMNLP 2023

- Hongru WANG, Minda Hu, Yang Deng, Rui Wang, Fei Mi, Weichao Wang, Yasheng Wang, Wai-Chung Kwan, Irwin King, Kam-Fai Wong. Large Language Models as Source Planner for Personalized Knowledge-grounded Dialogues. In Findings of EMNLP 2023

- Hongru WANG, Huimin Wang, Zezhong Wang, Kam-Fai Wong, Integrating Pretrained Language Model for Dialogue Policy Evaluation, ICASSP 2022

In addition, the research work for this dissertation has produced some indirect publications as listed below:

- Yuheng Lu*, Qian Yu*, Hongru WANG*, Zeming Liu, Wei Su, Yanping Liu, Yuhang Guo, Maocheng Liang, Yunhong Wang, Haifeng Wang. TransBench: Breaking Barriers for Transferable Graphical User Interface Agents in Dynamic Digital Environments. In Findings of ACL 2025

- Cheng Qian, Emre Can Acikgoz, Hongru WANG, Xiusi Chen, Avirup Sil, Dilek Hakkani-Tür, Gokhan Tur, Heng Ji. SMART: Self-Aware Agent for Tool Overuse Mitigation. In Findings of ACL 2025

- Jianqiao Lu, Zhiyang Dou, Hongru WANG, Zeyu Cao, Jianbo Dai, Yingjia Wan, Zhijiang Guo. AutoPSV: Automated Process-Supervised Verifier. In Proceedings of NeurIPS 2024.

- Hongru WANG, Baohang Zhou, Zhengkun Zhang, Yiming Du, David Ho, Kam-Fai Wong. M3sum: A Novel Unsupervised Language-Guided Video Summarization, ICASSP 2024

- Hongru WANG, Huimin Wang, Lingzhi Wang, Minda Hu, Rui Wang, Boyang Xue, Yongfeng Huang, Kam-Fai Wong. 2024. TPE: Towards Better Compositional Reasoning over Cognitive Tools via Multi-persona Collaboration. In Proceedings of NLPCC 2024

- Hongru WANG, Zezhong Wang, Wai Chung Kwan, Kam-Fai Wong. 2023. MCML: A Novel Memory-based Contrastive Meta-Learning Method for Few Shot Slot Tagging. In Proceedings of the IJCNLP-AACL 2023

- Hongru WANG, Wai-Chung Kwan, Min Li, Zimo Zhou, Kam-Fai Wong. KddRES: A Multi-level Knowledge-driven Dialogue Dataset for Restaurant Towards Customized Dialogue System, Computer Speech & Language

- Wai-Chung Kwan*, Hongru WANG*, Huimin Wang, Kam-Fai Wong. A Survey on Recent Advances and Challenges in Reinforcement Learning Methods for Task-oriented Dialogue Policy Learning. Mach. Intell. Res. 20, 318–334 (2023).

- Rui Wang*, Hongru WANG*, Fei Mi, Boyang Xue, Yi Chen, Kam-Fai Wong, and Ruifeng Xu. Enhancing Large Language Models Against Inductive Instructions with Dual-critique Prompting. In Proceedings of NAACL 2024

Biographical Sketch

Hongru Wang was born in a small village at the foot of Tianzhu Mountain, Qianshan, Anqing, Anhui, China in 1998. He received his B.S. degree in Computer Science and Technology (Big Data Technology and Application) from Communication University of China in 2019. During his undergraduate studies, he has interned at 360 – the biggest cybersecurity company at China, Kuaishou – the biggest competitor of Tiktok at China. He won several awards such as Meritorious Winner at Mathematical Contest In Modeling (2018), China National Radio Scholarship, and lead a team to build *WeCampus* mini-program within WeChat (2018SR562540). He graduated with a rank of top 5% in his department.

Afterwards, He enrolled as a master student at the department of Computer Science Engineering at The Chinese University of Hong Kong, working closely with Dr. Sunnay Lai, and Prof. Kwong Sak Leung. During his master study, he gradually narrowed his research focus to natural language processing and successfully joined Prof. Kam-Fai Wong's group as a student helper around December 2019. Before he started his Ph.D. study, he worked as a research assistant at the same group from July 2020 to July 2021, focusing on developing a task-oriented dialogue system for low-resource languages, e.g., Cantonese.

He enrolled as a Ph.D. student in the Department of Systems Engineering and Engineering Management at The Chinese University of Hong Kong in August 2021, under the supervision of Prof. Kam-Fai Wong. During the Ph.D. period, he has interned at Huawei Noah' Ark Lab (Shenzhen) from May 2022 to April 2023, and Seed-LLM-Horizon at ByteDance from August 2024 to December 2024. Moreover, he also cherished the great opportunity to visit The University of Edinburgh (EdinburghNLP) and University of Illinois

Urbana-Champaign (BlenderLab), working closely with Prof. Jeff Z. Pan, Prof. Pasquale Minervini and Prof. Heng Ji. He also work closely with Prof. Mengdi Wang from Princeton University during his final period of Ph.D. study on agent and RL. He has traveled to Bali (AACL 2023), Singapore (EMNLP 2023), Seoul (ICASSP 2024), Bangkok (ACL 2024), Turin (LREC-COLING 2024), Edinburgh, London, Chicago and New York.

Hongru's research focus consists of dialogue system, tool learning, large language models, and language agents. His long-term objective is to achieve the "*impossible triangle*" between safety, personalization and autonomy of language agent by integrating internal cognitive tools and external physical tools. He has published more than 40 papers at top conferences such as ACL, EMNLP, NAACL, NeurIPS, WWW, COLING, and etc, with more than 18 (co)first-author papers, 2 best paper awards at 2023 International Doctoral Forum and SIGHAN@ACL 2024. Some of his representative works include Cue-CoT, SAFARI, AppBench, Self-DC, OTC-PO and Theory of Agent. He organized the first Tool Learning tutorial (i.e., ToolsMeetLLM) at SIGIR 2024 and won the champion at the WWW 2024 Online Safety Prize Challenge. He is the co-founder and organizer of NLP Academic Exchange Platform (NICE), which provide a platform to share and discuss cutting-edge research for junior and senior researcher in the field.

Contents

List of Tables

List of Figures

Acknowledgments

I still remember my first day of Ph.D. study, my supervisor Prof. Kam-Fai Wong told me that being a leader instead of follower. That simple yet profound advice has guided me throughout my entire doctoral journey. The most fortunate and life-changing event for me was meeting him – the best supervisor I could imagine at the world. He is not only a supervisor, a close friend, but also like a family member, and in many ways, a true role model for me.

Prof. Wong's deep expertise in research, his brilliant ideas across diverse domains, and, most importantly, his unwavering enthusiasm for sharing knowledge have profoundly shaped my research taste and vision. I am deeply grateful for his accompany at nearly every conference I attended, for our countless discussions at morning, afternoon, and evening, and for his unwavering support and encouragement through both the highs and lows of my journey. He is the truly teacher who leads by example rather than just lecturing. I sincerely hope I have not let him down during this short but transformative Ph.D. journey. What I have achieved has far exceeded anything I could have imagined when I first began. None of it would have been possible without his unwavering support and guidance.

Prof. Wong is my lifelong role model and the person I respect most in the world. I will continue to learn from him and try my best to become someone like him — a person of wisdom, generosity, and integrity.

The second and third person I would like to thank are my father and mother. It hasn't been easy for me to get to where I am today — but believe me, it has been even harder for the parents of a child like me. Throughout my life, I've often placed them second or even lower on my priority list, yet they have never once complained. Instead, they have given me everything they could, always encouraging me to explore a bigger world beyond my own. Every achievement I've accomplished carries their silent support behind it. It is they who constantly remind me of where I come from and the kind of person I am. Without them, I wouldn't have been able to begin this journey at all. I am deeply grateful to them — and truly proud and incredibly lucky to be their child. Since my parents do not understand

English, I've included the Chinese version below especially for them.

我最想感謝的第二個和第三個人是我的父親和母親。雖然走到今天對我來說並不容易，但相信我，對於擁有我這樣孩子的父母而言，他們的路更為艱辛。在我的人生旅程中，我常常把他們放在第二位，甚至更低的位置，但他們從未有過一句怨言。相反，他們把他們擁有的一切都毫無保留的給了我，總是鼓勵我去看看更廣闊的世界。他們用他們的血和汗，以及虧空的身體為我鋪平了道路，我時常感到我所付出的辛苦，與他們的比起來，不值一提。我所取得的每一項成就背後，都有他們默默地支持。是他們一直提醒我，我來自哪裡，我是個什麼樣的人。沒有他們，這段旅程就不會開始。我對他們充滿感激，也充滿愧疚，我也無比自豪和幸運能成為他們的孩子。

Along this journey, I am grateful for everyone I've met, every discussion I've participated in, every conference I've attended, and every day and experience that brought me to where I am today - like a river formed by countless drops of water. Among these countless moments, there are a few waves that have left a profound impact on both my studies and my life. I'd like to take this opportunity to thank those individuals who have been such meaningful parts of my journey.

The first person I want to thank is Prof. Heng Ji at UIUC. To me, she has been more than just an advisor — she feels like a sister or mother. It was she who first made me feel that my research could be both impactful and meaningful in a broader world, even under the high standards. During my job-seeking period, she offered critical guidance and unwavering support, which truly enlightened and encouraged me. She has a kind of magic — just by talking with her, I find myself feeling more emotional, more grounded, and more confident. I will always remember her as the most powerful "wave" in this journey — a defining presence in my Ph.D. life, along with the many friends, collaborations, and moments we shared.

The second person I would like to thank is Prof. Jeff Z. Pan from the University of Edinburgh. My visit to Edinburgh marked my first time in Europe and my first experience living so far from home. I still vividly remember the breathtaking scenery and the vibrant research atmosphere of the University. EdinburghNLP is a pilgrimage site for every NLP

researcher, and I'm deeply grateful to Prof. Pan for giving me the opportunity to visit and collaborate. During my stay, we accomplished several meaningful research projects together, which remain some of my most memorable experiences. More importantly, Prof. Pan generously supported my career development — actively helping me connect with potential postdoctoral opportunities and offering valuable advice on both research and job-seeking. His mentorship extended far beyond academic collaboration, and I deeply appreciate his kindness and guidance during this crucial stage of my journey.

I would also like to extend my heartfelt thanks to the entire KF Group — including both alumni and current members — for their countless forms of support throughout my Ph.D. journey. It has been a true honor to work alongside such talented individuals and to learn from their remarkable research vision and skills. In particular, I am deeply grateful to Prof. Ruifeng Xu for his warm support and thoughtful guidance in both my academic development and research directions. I also want to thank many of my labmates — Xingshang Zeng, Bin Liang, Huimin Wang, Lingzhi Wang, Zhiming Mao, Wai-Chung Kwan, Zezhong Wang, Jingtao Cao, Boyang Xue, Rui Wang, Yiming Du, Liang Chen, and Zhengyi Zhao — for their companionship, collaboration, and encouragement during this formative and unforgettable chapter of my life. Besides that, I also would like to thank my thesis committee: Prof. YU Jeffrey Xu, Prof. WAI Hoi To and Prof. LIU Zhiyuan, for their valuable feedback and suggestions.

There are also many professors, mentors, and friends outside of my research group to whom I owe my sincere gratitude. Specifically, I would like to thank Prof. Mengdi Wang, Prof. Irwin King, Prof. Dilek Hakkani-Tür, Prof. Guanhua Chen, Prof. Yang Deng, Prof. Zeming Liu, Prof. Yuhang Guo and Prof. Pasquale Minervini for their invaluable insights and support at different stages of my academic journey. I am also deeply thankful to my mentors during internships — Fei Mi, Weichao Wang, Yasheng Wang, Yitong Li, Yufei Wang, Wanjun Zhong, and Deng Cai — for their generous guidance and mentorship. Additionally, I want to acknowledge and thank the many students and collaborators who have enriched my experience:

- EdinburghNLP: Wenyu Huang, Yu Zhao, Nan Hu, Xiaotang Du, Alessio Devoto, Giwon Hong, Aryo Pradipta Gema, Miao Li, Wenhao Zhu, ...
- UIUC: Cheng Qian, Xiusi Chen, Emre Can Acikgoz, Bowen Jin, Yuji Zhang, Zhenhailong Wang, Han Chi, ...
- Other institutions: Minda Hu, Baohang Zhou, Rongwu Xu, Zehan Qi, Jianqiao Lu, Jiahao Qiu, Shijue Huang, Jingyan Zhou, Lichen Zong, Zhengkun Zhang, Yujia Qin, Heming Xia, Cunxiang Wang, Zimo Zhou, ...
- Friends and Relationships: Xuechun Li, Junhao Xu, Mingyu Cui, Jiawen Kang, Lingwei Meng, Haohan Guo, Shujie Hu, ...

Thank you all for your companionship, collaboration, and encouragement — each of you has made a meaningful mark on this journey.

The last one I would like to thank is myself. Thank you, and Goodbye, Hongru. It is time to move on. To every place, every one, every moment during my Ph.D. journey. All of them have paved the path to this thesis.

Central Park, New York

2025.5.25

Chapter 1

Introduction

1.1 Motivation

> "The autonomous machine intelligence is designed to minimize the number of actions a system needs to take in the real world to learn a task."
>
> —- Yann Lecun [2]

Large Language Models (LLMs) have rapidly evolved beyond text generation into *autonomous language agents* capable of independently planning and executing complex tasks with minimal human oversight [3]. These emerging capabilities have enabled a broad range of real-world applications, including travel planning [4], human-computer interaction [5, 6, 7], and scientific research [8, 9, 10]. However, as these systems become more agentic and autonomous, several foundational questions remain underexplored: what is the agent [1]? What constitutes its desired behavior? And how can such behavior be realized in practice?

Most of previous methods frame agents as LLMs that interleave internal reasoning and external actions to complete tasks [5, 11, 12]. While functionally effective, this pragmatic framing lacks a principled account of how such behaviors should be coordinated or optimized. From an empirical standpoint, existing agentic systems primarily rely on

[1] We use agent, language agent and autonomous agent interchangeably in the thesis.

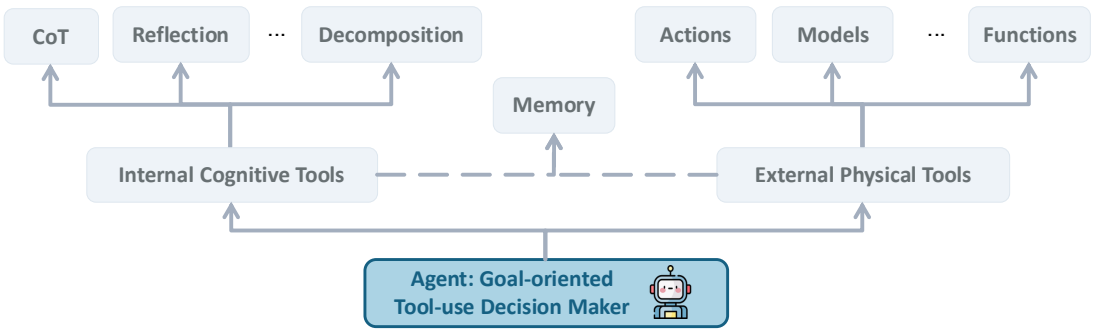


Figure 1.1: *Conceptual framework of agent as a goal-oriented tool-use decision maker. The memory can be both internal (i.e., short-term memory) or external (i.e., long-term memory).*

prompting [13] or supervised fine-tuning [14], but seldom investigate how these training paradigms relate to the desired agent behavior, leaving opaque the reasons behind agentic success or failure.

At a high level, this thesis proposes a principled framework of agent that offers a unified definition, actionable methods to optimize the behavior policy of agents. We advocate that the truly autonomous agent should learn the compression of the world as much as possible in its own parametric space and minimize external interaction to take in order to achieve a pre-defined goal, providing a concrete roadmap for building such agents. Moreover, the thesis also constructs a suite of benchmark environments that capture the complexity and diversity of real-world interaction patterns, enabling systematic evaluation of agent behavior across different applications.

Methods. Most of previous method often identify agents as LLMs that interleave reasoning (internal) and acting (external) to complete tasks, following (thoughts, action, observation) format [11]. For example, the agent needs to reason about its current state (i.e., thoughts), then decide which action to take (i.e., action) — such as calling a search engine — and receive external feedback (i.e., observation), repeating this process until it completes the

pre-defined goal. While functionally effective, this pragmatic framing lacks a principled account of how such behaviors should be coordinated or optimized. This thesis introduce a unified framework that regarding both internal reasoning and external acting as two different types of tools: internal cognitive tools and external physical tools, re-defining the agent as a tool-use decision maker that adaptively choose between internal introspection and external interaction (Figure 1.1). This framework is grounded in the core insight that reasoning and acting are not distinct behaviors but rather *epistemically equivalent tools* for acquiring task-relevant knowledge. We explore several methods to build agents capable to use both internal cognitive tools and external physical tools, and more importantly, dynamically balance the use of them to achieve more efficient and effective agent behavior besides the completion of the pre-defined goal such as providing correct answer.

Benchmarks. To further investigate the gap between existing LLMs and truly autonomous agent, this thesis introduces several benchmarks to evaluate the performance of tool utilization for existing LLMs in terms of single turn tool planning, multi-turn stateful tool utilization and personalized tool utilization. These provide new challenges for agents to utilize tools under more practical settings, as well as tremendous opportunities to develop more capable, context-aware, and personalized agents.

1.2 Preliminaries

1.2.1 The Unification of Reasoning and Acting

It is widely recognized that reasoning and acting constitute the two fundamental capabilities of intelligent agent behavior [11, 15]. Reasoning enables an agent to plan, infer, reflect, and monitor its internal cognitive state, while acting allows it to engage with the external environment to gather new information or carry out tasks. Rather than viewing these modalities as distinct or sequential processes, we propose that:

A Unified View of Reasoning and Acting

Reasoning and acting should be treated as equivalent epistemic tools within a unified framework, where reasoning entails an internal cognitive tool for manipulating information within the agent's parametric knowledge space, while acting entails an external physical tool for acquiring information beyond the agent's internal capabilities.

This unified perspective aligns with the affordance theory [16], which suggests that actions arise from the interplay between perception and interaction. From this perspective, reasoning and acting are not hierarchically ordered or merely sequential but are co-equal capabilities of decision-making. Each plays a complementary role in enabling agents to resolve uncertainty and make progress toward task completion. Embracing this integrated view encourages the development of intelligent systems that can seamlessly coordinate their internal cognitive mechanisms and external interactive capabilities, based on their current knowledge state and the epistemic demands of the task at hand.

Internal cognitive tools. Cognitive tools refer to internal cognitive mechanisms that support systematic or investigative thinking to solve problems [17]. In the context of intelligent agents, various reasoning modules [18, 19], such as Chain-of-Thought [20], reflection, decomposition, and alternative-thinking, function as cognitive processes that enable the retrieval and manipulation of internal knowledge to guide problem-solving. For instance, Reasoning via Planning (RAP) [21] conceptualizes the language model as both a world model and a reasoning engine, incrementally accumulating knowledge through iterative reasoning steps. Similarly, Self-Discover [18] constructs abstract reasoning structures and then instantiates them to address complex tasks, mirroring the approach of tool-based agents that first generate plans for tool use and then execute them sequentially [6, 22]. Beyond these, other cognitive tools appear in diverse applications, such as conversational strategies in dialogue systems [1] and psychologically inspired mechanisms designed to model uncertainty, emotion, or user intent [23]. Despite their varied forms, these tools share a common function: *they serve as triggers for internal knowledge retrieval, allowing the model to*

reason and act based on its embedded understanding of the world.

External physical tools. External physical tools refer to modules or interfaces outside the model that are invoked through specific triggers, such as rules, actions, or special tokens, whose outputs are then incorporated into the model's context to inform subsequent reasoning [22, 24]. These tools function as vital interfaces between the agent and its environment, enabling the acquisition of task-relevant knowledge that lies beyond the agent's internal parameters. Importantly, external tools span a wide spectrum of interactions, capturing how agents, like humans, leverage their surroundings to reduce uncertainty or complete tasks. Examples include querying a search engine, calling an API, processing sensor input, or performing physical actions [24, 25]. For instance, clicking a button in a user interface may be represented as an external tool call, where the input parameter is the button's location and the resulting webpage serves as the observation. Similarly, in embodied settings, actions such as "MoveTo(Room A)" can be interpreted as tool invocations, with "Room A" as the parameter and the resulting sensory output as the feedback. This perspective enables a unified treatment of diverse forms of interaction as structured tool use: *they serve as interfaces for external knowledge acquisition, allowing the model to access and interact with knowledge beyond its epistemic capacity.*

1.2.2 Tool-Integrated Agents

Building on the unification of reasoning and acting, we further propose a redefinition of the agent grounded in this integrated perspective:

Definition of Agent

An agent is an entity that coordinates internal cognitive tools (e.g., reflection) and external physical tools (e.g., function callings) to acquire knowledge in order to achieve a specific goal.

From this viewpoint, an agent is fundamentally a knowledge-driven decision-maker that navigates a task by alternating between internal reasoning and external interaction. Formally, a tool-integrated agent trajectory can be described as:

(a) ReAct-based Agent (b) Tool-integrated Agent

Figure 1.2: *Our proposed agent framework integrates both internal cognitive tools and external physical tools with agent itself as a world model.*

$$\tau = (t_1, k_1, t_2, k_2, ..., t_n, k_n), \tag{1.1}$$

where each t_i is either an internal or external tool invocation, and each k_i represents the corresponding knowledge retrieved. Here, "knowledge" is broadly defined as *any information that advances the agent's problem-solving state*. At each step, the agent must choose the most epistemically valuable tool based on its current state, aiming to progressively bridge the knowledge gap toward a complete solution. The process concludes when the agent has accumulated sufficient knowledge to achieve the pre-defined goal.

This unified framework offers several key advantages: (1) It generalizes prior approaches such as ReAct [11], which can be viewed as special cases where internal tool steps (e.g., reasoning) are treated as monolithic thought units r_i, leveraging the model's pre-trained cognitive abilities without requiring explicit tool separation. (2) It aligns with findings from large reasoning models (LRMs), which show that outcome-based reinforcement learning (RL) can effectively train agents to discover and utilize internal cognitive tools [26]. The same principle applies to external physical tools, as shown in recent studies on tool-augmented agents [27]. Thus, the framework provides a coherent foundation for agentic learning across both domains. (3) Most importantly, this perspective leads to a new learning paradigm: *next tool prediction*. Just as next-token prediction enables LLMs to learn a compressed representation of the world from text, next-tool prediction allows agents to learn procedural knowledge through interaction. By learning to choose the right tool, agents can dynamically

update their internal representations and evolve through experience, mimicking human-like adaptation and learning.

1.3 Contributions Overview

This thesis aims at developing intelligent language agents which mirror the human intelligence to master internal cognitive tools and external physical tools at the same time and dynamically call them on demand based on self-aware monitoring signals. More specifically, how could an agent can express emotions and generate more natural responses just like humans? how to plan the calls of multiple external physical tools in different contexts? and how to call different tools on demand especially when these tools have overlapping functionalities or complementary with each other? The key philosophy of this thesis is to explore an universal and scalable way to unify all interactions from the tool perspective.

Building Agents with Internal Cognitive Tools First, to generate more natural and human-like responses, we propose chain-of-cues –a novel linguistic cue-based chain-of-thoughts, in order to mimic the cognitive processing of humans which is capable of inferring the underlying user status beneath the dialogue context. To evaluate our method, we build a new benchmark, consisting of 6 dialogue datasets in both Chinese and English, targeting 3 major linguistic cues during the conversation: personality, emotion, and psychology. The empirical results demonstrate our proposed Chain-of-Cues method outperforms standard prompting methods in terms of both helpfulness and acceptability on all datasets regardless the model used. Moreover, we also present how to incorporate more internal cognitive tools at different tasks such as various conversational strategies in tutoring dialogue system and different reasoning modules (i.e., reflection, decomposition) to solve complex problems.

Building Agents with External Physical Tools Further, we alternatively explore agentic retrieval-augmented generation (RAG) problem which combines the capabilities of autonomous agents with retrieval-augmented generation to dynamically retrieve and syn-

thesize information to generate the final response. We opted to decompose the multiple knowledge sources-augmented generation problem into three consecutive steps: 1) planning: make a series of decision to determine whether or not use knowledge, which source to use and when; 2) retrieval: call external retriever to retrieve corresponding documents; and 3) assembling: incorporating all previous results to generate the final response. Our proposed framework: SAFARI, showcases exceptional performance to plan, understand, and incorporate these multiple knowledge sources and capture the dependency relationship within them. Additionally, we explore how to iteratively refine outputs through feedback loops, mimicking human-like problem-solving. We also discuss different methods for improving the agent's decision-making capabilities to call external tools, including learning from demonstrations and learning from feedback.

Building Agents with Self-aware Tool Utilization The integration of internal cognitive tools and external physical tools does not only stands for a novel perspective to build powerful language agents, but also brings new challenges in real world. Inspired by meta-reasoning theory [28], we first present that desired strategy to monitor and control the behavior of language agent and advocate the key lies in the alignment between the knowledge boundary with the tool-use decision boundary. Specifically, we propose Self Divide-and-Conquer (Self-DC) framework, enables language agent to adaptively choose internal cognitive tools or external physical tools as needed according to self-aware confidence signals, resulting in a better trade-off between effectiveness and efficiency. Moreover, we explore better monitoring and control methods, aiming to verify the correctness of each decision and make every tool call meaningful and useful. To address the knowledge conflict by different types of tools, we leverage the different representations of language agent at different context to detect it and generate aligned response with the trusted source. We define the new agent objective and then present a systematic approach to realize this objective in practice.

Benchmarking Tool Planning in Physical World Finally, we built a benchmark for language agents that is highly realistic and impactful. Specifically, we focus on agents that perform tool planning in the mobile device, i.e., apple intelligence. We consider two significant challenges in multiple APIs of mobile devices: 1) *graph structures*: some APIs can be executed independently while others need to be executed one by one, resulting in graph-like execution order; and 2) *permission constraints*: which source is authorized to execute the API call. We then categorize user instruction into four types: SS, SM, MS, MM where the first S or M stands for the number of Apps required, and later S or M the number of API required to solve the user task. Even the most powerful language models (i.e., gpt-4o) only achieves only a 2.0% success rate at the most complex instruction, highlighting the need for further development of powerful language agents for real-life tasks.

Part I

Methods

Chapter 2

Cue-CoT: Building Agents with Internal Cognitive Tools

2.1 Introduction

Large Language Models (LLMs) naturally acquire a range of internal cognitive tools - such as reflection and decomposition - through pretraining on massive text corpora [29]. Empirical evidence suggests that even the existing most advanced models, such as DeepSeek-R1 [26], inherit these tools from dialogue-centric datasets and further amplify them through reinforcement learning [30]. These internal tools play a critical role in guiding the model's reasoning processes, serving as foundational capabilities that enable systematic and creative problem-solving, and are readily activated to enhance task performance. A notable example is Chain-of-Thought (CoT) prompting [20], which has sparked a new research direction in prompt engineering. This line of work focuses on designing prompts that selectively activate different internal cognitive tools within LLMs, ultimately improving their reasoning quality and downstream task effectiveness [23, 31].

Figure 2.1 shows several major internal cognitive tools utilized in previous studies. Following conventional functional call promptings, the internal cognitive tools can also be defined as "tool name: tool description". For instance, the CoT can be defined as "CoT: Let's

decomposition: breaking down a complex problem into smaller, more manageable parts. Making sure that you also provide answers for all decomposed problems in this section. You can decompose iteratively but should not contain same problem or exceed the max iteration depth which is three.
backward: starting with the desired observations at any previous reasoning step and working backward to identify the new reasoning directions.
detail: any details including but not limited to logic and reasons for your reasoning in this way, you are encouraged to add this at every unclear or unnatural reasoning step.
summary: summarize your reasoning to help future thinking.
alternatives: directly thinking in other ways, try to explore different solutions as much as possible to solve given problem.
reflection: you are encouraged to regularly reflect on your past reasoning in current response at various levels of detail, from sentence down to individual word. This will help you better understand and think through problems. It's okay to make mistakes; use them as opportunities to learn and improve.
……

Figure 2.1: *Some representative internal cognitive tools to guide the reasoning processing of LLMs.*

think step by step". Although there are plenty of studies investigate the effects of different internal tools, this is the first time that we propose to understand these from the tool perspective. There are several advantages: 1) Not all internal cognitive tools are universally useful or necessary to activate for every problem, even for widely applicable methods like CoT [32]; 2) The tool perspective offers a unified framework that encompasses both internal cognitive tools and external physical tools. This unification enables one promising path. As simple outcome-based rewards can implicitly encourage the emergence and use of diverse internal cognitive tools to solve problems, as observed in DeepSeek-R1 [26], it is feasible to leverage similar method to build truly autonomous agents capable of dynamically coordinating both internal reasoning and external interactions.

In this chapter, we begin by exploring a specific internal cognitive tool employed by LLMs in personalized dialogue systems, aimed at generating more personalized, empathetic, and compassionate responses. Such capabilities are highly valuable in a range of applications, including recommendation systems and psychotherapy. We then broaden our focus to a wider set of internal cognitive tools, examining how their composition enables diverse conversational strategies used in dialogue system. For example, reflection and empathy have emerged as two particularly effective tools: reflection is widely used to help trainee teachers develop meta-cognitive skills and enhance their capacity for reflective thinking and learning [33], while empathy plays a crucial role in offering emotional support in human

communication [34]. Building on these insights, we propose a self-reasoning language model that can refine its own reasoning rationales by leveraging a diverse set of internal cognitive tools. By fine-tuning on its own self-generated data, the model progressively evolves and achieves improved task performance.

2.2 Related Work

Before the era of LLM, most of the previous work develops personalized [35, 36], emotional [37, 38, 39], empathetic [40] dialogue system in isolation, rather than seamlessly blending them all into one cohesive conversational flow [41]. A common approach is to predict the emotion or persona from a pre-defined set and generate the response in a multi-task manner [42, 43]. Besides that, lots of work notice these linguistic cues underneath text by directly predicting them independently as a classification task [44, 45]. Distinguishing from these previous works, we regard different aspects of cues as part of user status and prompt the LLMs to reason user status exhibited in the dialogue context, aiming to generate more helpful and acceptable responses for users.

Large Language Models (LLMs), or foundation models [46], especially after the appearance of `ChatGPT`[1], recently revolutionize the paradigm of various natural language processing (NLP) tasks, including dialogue response generation tasks [47]. However, most existing LLM-based studies directly feed the user query or dialogue content to the LLM for generating a response with a preceding prompt, making the responses stereotypical and tedious, especially for in-depth dialogue questions [48]. On the contrary, it is widely acknowledged that dialogue contexts generally convey a lot of information about the user status in addition to the pure semantic information from a linguistic perspective [49, 50, 51]. Specifically, the linguistic cues underlying dialogue context have been shown to be an effective means of revealing the emotions [52], personality traits [49], psychological characteristics [50], and other relevant information of users [53]. Consequently, recognizing

[1]https://openai.com/blog/chatgpt

and understanding these cues exhibited in the context of dialogues becomes crucial to comprehend user intentions and status [40]. By doing so, a dialogue system can generate responses that align with the user's expectations, meet their unique needs and preferences, and ultimately strive towards constructing a human-like conversational experience [54].

2.3 Chain-of-Cues: Personalized Dialogue Response Generation

2.3.1 An Overview

Inspired by the emergent capabilities of LLMs on reasoning over texts [55], while previous efforts mostly investigate in-context learning [56] or chain-of-thought prompting schemes [20] on text classification and question answering [57, 58], we take advantage of LLMs to reason for dialogue, which, to our best knowledge, has not been thoroughly investigated yet. Specifically, we design a linguistic cue-based chain-of-thoughts (*Cue*-CoT), consisting of two variants: *O-Cue* CoT and *M-Cue* CoT in which the former one outputs intermediate reasoning results with a final response in one-step but the later reasons step by step, as shown in Figure 2.2. In detail, with standard prompting, LLM-based systems directly generate the response given the dialogue context. Regarding the user status implied by the context as intermediate reasoning results (*Cue* CoT), we prompt the system to infer the user status first and then generate a response based on dialogue context and user status.

To evaluate our approach, we build a benchmark, consisting of **6** in-depth dialogue datasets in both Chinese and English, considering three major aspects of user statuses: *personality*, *emotions*, and *psychology*, exhibited during the conversation, forming a comprehensive evaluation benchmark incorporating various user statuses in the context of dialogue response generation. We conduct extensive experiments with **5** LLM-based dialogue systems based on the benchmark using the aforementioned three types of prompting schemes. To sum up, our contributions can be summarized below:

- We construct an in-depth dialogue evaluation benchmark considering the personality, emotion, and psychology of users exhibited in the conversation, with the goal of

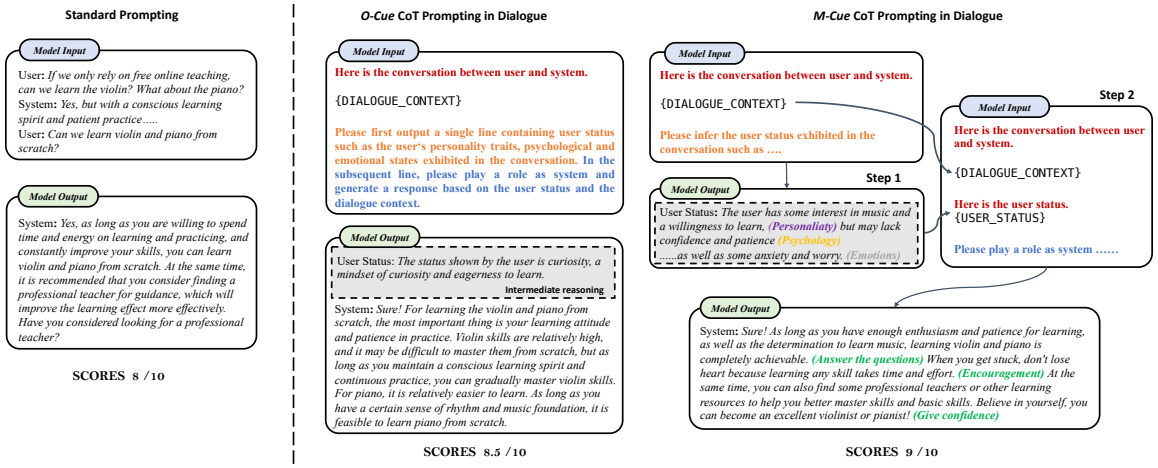


Figure 2.2: *An example of different prompting for responding to in-depth dialog questions with LLMs, including standard prompting, O-Cue CoT, and M-Cue CoT. We shadow the intermediate reasoning results, i.e., the personality, empathy, and psychological status of the user, and highlight the instructions at the input and indicate the roles of different parts of the response (in green) in M-Cue CoT.*

aligning with unique user needs and status, which consists of 6 datasets, and 7.3k dialogues.

- We propose two effective dialogue cots: *O-Cue* CoT and *M-Cue* CoT, that enable advanced reasoning and planning based on user statuses. Additionally, we suggest utilizing intermediate reasoning results as a criterion for selecting demonstrations in limited training data scenarios, specifically in one-shot settings.

- Our findings demonstrate that both the *O-Cue* CoT and *M-Cue* CoT approaches outperform standard prompting in generating more helpful and acceptable responses for the users. Specifically, the *M-Cue* CoT shows superior robustness and reasoning performance in all datasets and all LLMs. Furthermore, our novel demonstration selection strategy exhibits superior performance under both *random selection* and *top1 selection*.

2.3.2 O-Cue v.s. M-Cue

We describe the prompting schemes in a general form, including standard prompting, *O-Cue* CoT, and *M-Cue* CoT as presented in Figure 2.2.

Standard Prompting. Most of the previous works directly prompt LLMs to generate responses solely based on dialogue context or user queries, which lack transparency and interpretability. The objective is defined as:

$$\mathcal{M} : \boldsymbol{c} \rightarrow \boldsymbol{r} \tag{2.1}$$

where $\mathcal{M}$ is parameterized by LLMs, $\boldsymbol{c}$ and $\boldsymbol{r}$ demotes dialogue context and response respectively.

***O-Cue* CoT.** In line with the traditional chain-of-thoughts, we prompt the models to generate the middle reasoning processing and final results together, for example, we can prompt the LLMs to generate user status and a final response simultaneously giving dialogue context, enforcing the LLMs to reason based on the user status. However, it is important to note that generating intermediate reasoning results with responses together may lead to a reduction in the length of the different outputs, particularly when multiple or complex reasoning results are involved, sacrificing the details and explanations. For example, as shown in *O-Cue* CoT in Figure 2.2, the generated user status is too short to provide cues for responses. Moreover, it is infeasible to modify the intermediate results when it is wrong [59]. Here, we defined the objective as follows in which $\boldsymbol{s}$ stands for user status:

$$\mathcal{M} : \boldsymbol{c} \rightarrow \boldsymbol{s}, \boldsymbol{r} \tag{2.2}$$

***M-Cue* CoT.** In addition to standard prompting and *O-Cue*, we can further enhance the quality of responses in LLMs by decomposing reasoning into different consecutive steps while the final step is to generate responses according to previous reasoning outputs. On the one hand, it is convenient to process these intermediate outputs, allowing for actions such as incorporating user profiles for personalization [54] or filtering out erroneous reasoning results. These intermediate outputs can also be stored for future use, enabling their utilization for various purposes. On the other hand, these intermediate results can

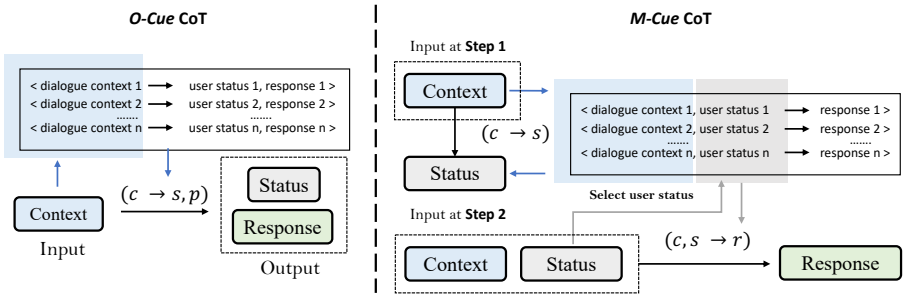


Figure 2.3: *Different demonstration selection strategies of O-Cue and M-Cue CoT, while the returned results such as* $(\boldsymbol{c} \rightarrow \boldsymbol{s}, \boldsymbol{p})$ *are prepended to original input to form new input.*

be used as a criterion to select demonstrations under few-shot settings (See next section). Overall, this technique allows for a clearer and more systematic progression of reasoning, resulting in better transparency and interpretability. The objective can be viewed as follows:

$$\mathcal{M} : \boldsymbol{c} \rightarrow \boldsymbol{s} \rightarrow \boldsymbol{r} \tag{2.3}$$

2.3.3 In-context Learning

The few-shot performance of LLMs depends heavily on the quality of the demonstrations, especially for complex tasks that need multiple reasoning steps [57]. Furthermore, in the context of dialogue systems, the process of selecting demonstrations becomes more challenging due to the one-to-many nature of dialogue interactions. As a result, novel approaches are needed to tackle the intricacies of dialogue response selection, taking into account the dynamic and context-dependent nature of conversations. We here introduce the demonstration selection strategy of three prompt schemes.

Standard Prompting. Following previous work [20], we use randomly sampled examples (*random selection*) or most semantic similar examples (*top-1 selection*) according to dialogue context as our demonstrations $(\boldsymbol{c} \rightarrow \boldsymbol{r})$.

***O-Cue* CoT.** Figure 2.3 shows the demonstration selection strategy of *Cue*-CoT. Although we still select demonstrations according to dialogue context at *O-Cue* CoT, the user status is added as intermediate reasoning results to enhance the reasoning ability of LLMs ($\boldsymbol{c} \rightarrow \boldsymbol{s}, \boldsymbol{r}$).

***M-Cue* CoT.** Since there are multiple steps, we design different selection strategies for each step. Specifically, we first select demonstrations ($\boldsymbol{c} \rightarrow \boldsymbol{s}$) according to dialogue context to infer status, and then select demonstrations ($\boldsymbol{c}, \boldsymbol{s} \rightarrow \boldsymbol{r}$) according to user status. In this way, all intermediate reasoning results can be utilized as a criterion to select demonstrations, providing additional signals for the latter reasoning. An assumption underneath here is that users with similar statuses tend to accept responses with a similar style. Besides that, we also apply *random selection* and *top-1 selection* to *O-Cue* CoT and *M-Cue* CoT for detailed comparison.

2.4 Experiments

In this section, we have conducted a comprehensive experiment to compare the performance of three prompting methods: standard prompting, *O-Cue* and *M-Cue* CoT in the benchmark under both zero-shot and one-shot settings[2].

2.4.1 Dataset Collection

In order to evaluate the performance of proposed *Cue*-CoT to reason different user statuses, we collect six datasets in terms of personality, empathy, and psychology, in both Chinese and English.

Personality. Previous works found that the content and style of a user's inquiry can provide indirect insights into their personality traits [49, 44]. For instance, an individual with a tendency towards anxiety may ask for advice on how to alleviate nervousness before

[2]Since the length of dialogue context is relatively long, the input length limit is easy to break when the number of shot exceeds 1, so we choose the one-shot setting to conduct in-context learning.

an upcoming job interview, phrasing the question as follows: "*What strategies can I employ to reduce my anxiety and perform well in tomorrow's interview?*". Since the public existing datasets either focus on the personae of the system [60] or target classification tasks without providing corresponding dialogue response [44], we thus build a pipeline to automatically collect the datasets using `ChatGPT` (`gpt-3.5-turbo-0301`). We first collect question-answer seeds from the two largest real-world online QA forums: Zhihu and Quora[3], and then prompt the `ChatGPT` to infer the personality first. We lastly require the `ChatGPT` to continue the dialogue given the inferred personality and the question-answer seed. In order to facilitate the continuous generation of transcripts for both participants in a dialogue, we utilize a template to establish the necessary format and requirements. In this way, the use of personality seed and question-answer seed in the template assures greater diversity and reliability of user queries. Specifically, the personality seed determines the style of the user query, while the question seed determines the content. As a result, the user statuses vary across different dialogues, contributing to a richer and more varied conversational experience.

Emotion. In terms of the emotional status of users, we re-organize two existing empathetic dialogue datasets: D4 [61] and EmpatheticDialogues (*a.k.a,* ED) [40]. For the former one, we first identify all utterances from the system labeled as *empathic comfort* for each dialogue sample in the test set. From these instances, the utterance with the longest length is chosen as the ground truth response, regarding preceding utterances as corresponding dialogue context[4]. This approach ensures fairness and comparability in evaluating the performance of LLMs, particularly because they tend to generate lengthy responses. For the ED, there are two roles in the dialogue: *Listener* who is actively listening, and *Speaker* who is speaking and conveying information. We follow the setting of the original paper [40], and directly use all samples in the test set. Neither the situation description written by the *Speaker* nor

[3] `https://www.zhihu.com/` and `https://huggingface.co/datasets/quora`

[4] We also tried directly regarding the last utterance labeled as *empathic comfort* as grounded truth response, but we found most of them are short and uninformative such as *you are welcome, take care* and so on.

Metrics	Chinese			English		
	Zhihu	D4	PsyQA	Quora	ED	EMH
Avg.C	258.4	521.0	210.9	149.6	50.2	44.2
Avg.R	76.9	57.9	607.5	48.3	12.9	175.8
Samples	1122	997	1000	1082	2091	1000

Table 2.1: *Data statistics of our used datasets including three **Chinese** datasets and three **English** datasets, while each of them represents different aspects of user status during the conversation. We highlight maximum Avg.C and Avg.R.*

the emotional label is contained (just as they were not given to the *Listener* during dialogue collection). Thus, the collected empathetic dialogue datasets provide a standard benchmark for evaluating the LLMs' ability to generate empathic responses.

Psychology. In order to assess the effectiveness of LLMs in generating counseling responses for mental health support, we employed two pre-existing datasets, namely PsyQA [62] and EMH [63]. These datasets were utilized as dialogue pools from which we selected appropriate samples to serve as a benchmark for evaluating the language models. In PsyQA, there are 4,012 questions out of 22,341 samples that are sampled to pick the highest-voted answers. We randomly sample 1,000 out of these 4,012 questions, regarding the highest-voted answer as ground truth to form a more challenging test set. We also provide the question description beside the question itself following the original setting [62]. In EMH, there are 10k (post, response) pairs annotated with three different communication mechanisms: *emotional reactions*, *interpretations*, and *explorations*. We first sorted examples according to the length of their answers and then uniformly sampled examples with these three mechanisms, forming a final test set.

All. Table 5.2 shows the data statistics of our benchmark. The notation **Avg. C** signifies the mean context length of instances, and if it exceeds a certain threshold, it may surpass the input context limit of LLMs[5] or become too lengthy for LLMs to comprehend. On the

[5]For example, the input context limit of BELLE-LLAMA-7B-2M is 2048, and few of examples from D4 exceeds the limit and the scenario becomes worse under the one-shot setting. We will have more detailed analysis in

other hand, **Avg. R** denotes the average response length. Generally, longer responses tend to be more comprehensive and clearer, presenting a more challenging baseline for LLMs to surpass. To sum up, we build a benchmark, consisting of six datasets (three Chinese datasets and three English datasets) in terms of three aspects of user status during the conversation, hoping the release of it can facilitate the research of dialogue systems based on LLMs.

2.4.2 Setup

LLMs Family. We compared the performance of different LLMs with our benchmark, including `ChatGLM-6B` [64], `BELLE-LLAMA-7B-2M` [65], `ChatGPT` for Chinese, and `Alpaca-7B` [66], `Vicuna-7B-v1.1`[6] and also `ChatGPT` for English. We strictly follow the commands and procedures to recover the weights of these models and we strongly suggest that the reader read the original paper to check more details. We set the temperature as 0.2 and top p as 0.1 for evaluation, and temperature as 0.7 and top p as 0.95 for generation in all models. We use BERT [67] as an encoder to select the nearest example to the test query for *top-1* one-shot setting, storing the mean vector of examples as sentence embedding[7].

Evaluation. 1) Metrics: Lots of previous works found that CHATGPT demonstrates superior performance than most existing automatic metrics to evaluate the quality of texts [48]. We mainly choose to use it to evaluate the quality of the generated responses in a **pair-wise** manner, considering **helpfulness** and **acceptness**. The evaluation templates can be found in original paper and we calculate the win rate using #wins / (#wins + #ties + #loses). In addition, we also provide the performance of most existing automatic metrics [40, 62] such as **Avg.BLEU** and **F1** but we found it can not align well with human judgments [48]. 2) Methods: Due to the exceptional proficiency of the LLM-based dialogue system, it is relatively easy for them to beat the ground truth responses in the original datasets, we consider standard

later sections.

[6]https://github.com/lm-sys/FastChat

[7]We directly user `bert-base-chinese` for all Chinese datasets and `bert-base-uncased` for all English datasets, we do not finetune the BERT model.

prompting as a more challenging baseline and compare the responses generated using our proposed *Cue*-CoT with the response generated using standard prompting, which is more fair and convincing. We also provide the human evaluation result as a reference.

2.4.3 Main Results

All. Table 2.2 and Table 2.3 present the win rate of responses generated by *O-Cue* and *M-Cue* CoT compared with the responses by standard prompting on Chinese and English datasets respectively. Despite that there are few LLMs that perform worse than standard prompting using *O-Cue* due to its complex instructions, *i.e.* `ChatGLM` in Chinese and `Alpaca` in English, it is observed that *O-Cue* can achieve above 50% win rate mostly in Both Chinese and English. Moreover, it is exciting to find that *M-Cue* further boosts performance and achieves higher win rates irrespective of the type of language model, datasets, or settings used, revealing its robustness and effectiveness. We attribute this to the relatively easy-understanding instructions and clear outputs in each step of the *M-Cue*, since some LLMs are incapable to follow relatively long instructions in *O-Cue* and output the content and style as required. For example, we asked the LLMs to output user status and response in two separate lines but only a few LLMs output in the format, making it difficult to distinguish the response from reasoning results. Also, the combined output of the user status and response can potentially limit the length of various components, thereby accounting for the disparity between *O-Cue* and *M-Cue*. Furthermore, we found that the *acceptness* is relatively lower than *helpfulness* for Chinese LLMs but higher for English LLMs, especially under the one-shot setting, revealing the weakness of Chinese LLMs to provide acceptable besides helpful responses.

Chinese LLMs. Table 2.2 shows the performance of Chinese LLMs. We surprisingly found that `ChatGLM` performs worst out of the three LLMs using *O-Cue* but better than `BELLE` (especially at *helpfulness*) using *M-Cue* under the zero-shot setting, and then we carefully check the outputs of these LLMs and found that `ChatGLM` almost fully ignore

Model	Prompt	Helpfulness			Acceptness		
		Zhihu	D4	PsyQA	Zhihu	D4	PsyQA
Zero-shot Setting							
BELLE	*O-Cue*	67.40	76.34	69.31	55.82	52.50	53.43
	M-Cue	81.54	71.60	79.25	60.23	72.41	73.65
CHATGLM	*O-Cue*	48.29	56.68	33.00	32.39	39.19	31.34
	M-Cue	85.02	72.10	83.57	66.67	51.27	55.40
CHATGPT	*O-Cue*	67.91	50.40	61.90	53.14	52.38	58.15
	M-Cue	95.57	87.88	90.34	65.22	61.08	56.12
One-shot Setting							
							random selection
BELLE	*O-Cue*	64.31	50.53	65.15	53.35	40.07	53.81
	M-Cue	83.30	69.59	73.81	73.61	56.14	61.90
CHATGLM	*O-Cue*	-	-	-	-	-	-
	M-Cue	90.28	75.10	91.85	74.55	54.03	64.75
CHATGPT	*O-Cue*	76.47	51.94	65.44	63.86	50.47	56.03
	M-Cue	91.60	86.67	88.96	76.83	58.19	61.41
							top-1 selection
BELLE	*O-Cue*	63.77	57.51	69.92	54.93	41.02	55.87
	M-Cue	82.77	69.94	73.99	74.32	54.38	62.24
CHATGLM	*O-Cue*	-	-	-	-	-	-
	M-Cue	89.25	77.26	91.77	73.43	57.17	58.74
CHATGPT	*O-Cue*	76.86	50.93	55.85	59.63	52.02	57.58
	M-Cue	93.19	88.84	91.77	78.46	56.84	59.48

Table 2.2: *The win rate of responses generated by our method compared with the response with standard prompting on three **Chinese** datasets in terms of **helpfulness** and **acceptness**. The underlined numbers mean that there are about 160 to 280 valid responses out of 500 in this setting due to the input context limit of the model.*

the instructions in *O-Cue* and simply continue the dialogue. However, we found it can follow instructions well in *M-Cue*, resulting in higher win rates. We attribute this to the relatively more complex and longer instructions in *O-Cue* and poor complex-instructions understanding of `ChatGLM`[8]. In addition, with the *M-Cue* method, we found that the performance of all models on D4 is relatively worse than the other two datasets. We suspect the reason is the longest length of context in D4. Moreover, we observe that the responses generated by `ChatGLM` and `BELLE` under the one-shot setting are much better under the

[8]Thus, we do not report the one-shot results using *O-Cue* for `ChatGLM`.

Model	Prompt	Helpfulness			Acceptness		
		Quora	ED	EMH	Quora	ED	EMH
Zero-shot Setting							
ALPACA	*O-Cue*	19.51	39.41	49.70	22.85	35.41	50.15
	M-Cue	80.78	87.30	85.76	78.21	86.00	86.97
VICUNA	*O-Cue*	56.16	71.43	59.43	55.73	65.06	63.50
	M-Cue	81.67	91.30	80.42	77.89	90.71	82.93
CHATGPT	*O-Cue*	79.47	88.31	82.83	81.47	89.92	93.71
	M-Cue	85.83	91.98	82.93	89.09	96.79	94.93
One-shot Setting							
							random selection
ALPACA	*O-Cue*	-	-	-	-	-	-
	M-Cue	76.78	85.08	94.36	72.34	85.07	95.82
VICUNA	*O-Cue*	60.45	70.77	63.06	60.45	68.21	67.07
	M-Cue	79.84	91.20	79.23	83.16	92.45	87.99
CHATGPT	*O-Cue*	80.33	87.32	84.94	80.33	90.80	96.06
	M-Cue	84.31	89.78	85.71	86.64	93.94	96.70
							top-1 selection
ALPACA	*O-Cue*	-	-	-	-	-	-
	M-Cue	74.54	78.70	88.69	72.27	79.55	93.43
VICUNA	*O-Cue*	63.10	71.75	62.31	62.04	67.21	67.76
	M-Cue	78.70	90.12	79.10	82.08	92.96	88.96
CHATGPT	*O-Cue*	81.15	87.42	81.40	80.24	89.92	91.84
	M-Cue	88.08	91.37	86.87	91.21	95.95	96.12

Table 2.3: *The win rate of responses generated by our method compared with the response with standard prompting on three **English** datasets in terms of **helpfulness** and **acceptness**. The underlined dataset mean that there are about 330 valid responses out of 500 in this dataset for all experiments due to the input context limit of the model.*

zero-shot setting, *i.e.,* less general responses and more responses in line with the role, benefiting from the informative demonstrations.

English LLMs. Table 2.3 shows the performance of English LLMs. Similarly, for the zero-shot setting using *O-Cue*, we found that `Alpaca` hardly follows the instructions, which often produces ambiguous outputs, mostly presenting user status and other times providing the response without any indication[9]. Besides that, with the *M-Cue* method, due to the

[9]We do not report one-shot for `Alpaca`, too.

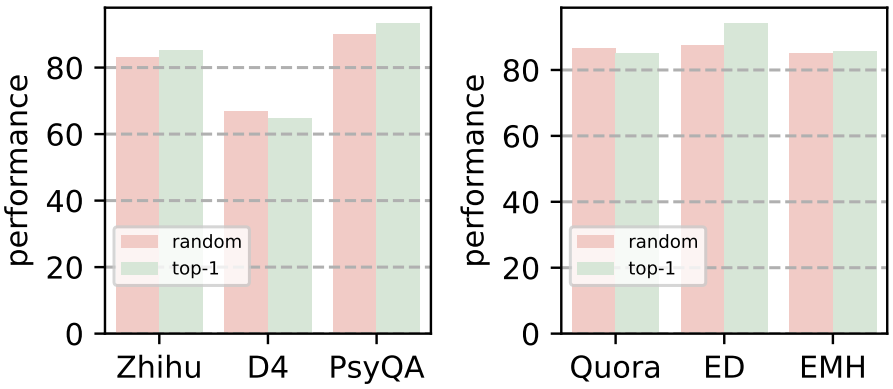


Figure 2.4: *The win rate of responses (acceptness) generated by* `chatgpt` *under different demonstration selection strategies under one-shot setting v.s. responses under the zero-shot setting, using M-Cue CoT.*

innate limitations of `Alpaca`, the win rate of responses is the lowest among all LLMs and settings. In addition, English LLMs also perform worst on the dataset which has the longest context length (Quora), in which `ChatGPT` and `Vicuna` tend to generate much longer responses than `Alpaca` due to limit of max length.

2.4.4 Discussion and Analysis

In this section, we conduct an extensive analysis with the backbone as `ChatGPT` using *M-Cue* CoT because of its superior performance in both Chinese and English[10].

One-shot v.s. Zero-shot Figure 2.4 shows the direct comparison of responses generated under different settings using *M-Cue*. There are 5 out of 6 datasets except for D4 in which one-shot (both *random* or *top-1* selection) beats zero-shot since the win rates all exceed 80%. The suboptimal performance of D4 in the one-shot setting can be attributed largely to the limitations imposed by the input length constraint. Furthermore, we can observe that *top-1* selection achieves better performance than *random* selection in 4 out of 6 datasets, suggesting users with similar statuses tend to like similar expression styles in responses. We attribute

[10]We present the results in terms of *acceptness* since this metric is more suitable for our motivation. We put helpfulness analysis in the Appendix.

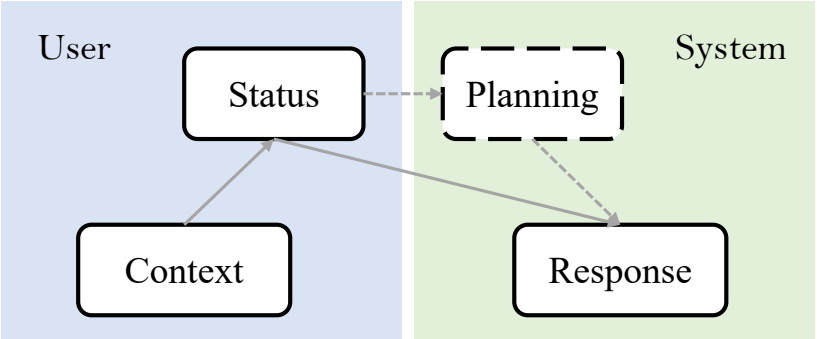


Figure 2.5: *An example of multiple intermediate reasoning outputs for different roles:* ***User*** *and* ***System*** *in in-depth dialogue questions.*

the relatively lower performance of *top-1* selection in D4 and Quora to the difficulty the LLM encounters in attending to critical input components due to the lengthy context.

More Reasoning Steps We tried to introduce an additional step (Step 2) after user status inference (Step 1): ***response planning*** by prompting the model to plan the response considering the dialogue context and user status. Specifically, we prompt the model to answer the following questions: *"Based on the context of the conversation and the user status such as personality traits, and psychological and emotional state, what aspects should the system pay attention to when responding?"* after giving the dialogue and user status. We regard the output of LLMs as system planning $\boldsymbol{p}$ as shown in Figure 2.5, and thus there are three different variants of *M-Cue* in the last step: ProcessA: $\boldsymbol{c}, \boldsymbol{s} \rightarrow \boldsymbol{r}$; ProcessB: $\boldsymbol{c}, \boldsymbol{p} \rightarrow \boldsymbol{r}$; and ProcessC: $\boldsymbol{c}, \boldsymbol{s}, \boldsymbol{p} \rightarrow \boldsymbol{r}$, in which ProcessA is chosen in our main experiment. Table 2.4 shows the results. First of all, it is likely that adding more reasoning steps will improve the LLMs' performance, but it is not necessary to assemble all intermediate reasoning results at the last step, for example, variant ProcessB reaches a higher win rate than ProcessC with only planning as an intermediate result. We emphasize the observation may not hold once the LLM type is changed due to various long-context understanding and instruction-following capabilities across them. As additional steps introduce extra input and extra computation for the inference, making the few-shot unpractical.

Method	Chinese			English		
	Zhihu	D4	PsyQA	Quora	ED	EMH
ProcessA	65.22	**61.08**	56.12	89.09	96.79	94.93
ProcessB	**76.15**	55.82	57.72	89.79	**98.78**	97.62
ProcessC	75.91	57.23	**58.74**	**94.50**	98.57	**98.22**

Table 2.4: *The win rate of different variants in terms of acceptness with the* `chatgpt` *as the backbone.*

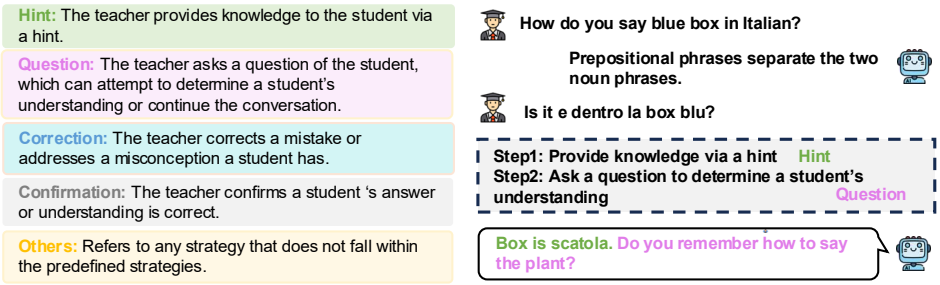


Figure 2.6: *Different conversational strategies as different internal cognitive tools in tutoring dialogue system [1]*

2.5 Incorporating More Cognitive Tools

2.5.1 Conversational Strategies

Individuals typically employ a range of conversational strategies, whether singly or in combination, to furnish constructive and refined responses [62, 68]. As shown in Figure 2.6, the tutoring dialogue systems need to seamlessly blend educational content with different motivational strategies in single turn (e.g., *hint* $\rightarrow$ *question*) to optimize the learning experience.

Inspired by recent progress that unleashes the cognitive synergist in LLMs with multi-persona collaboration [69], we introduce a multi-persona framework: Think-Plan-Execute (a.k.a, TPE), a novel prompting paradigm to enhance the planning ability of *cognitive tools* for the dialogue system. Specifically, TPE involves a structured decomposition of the overall planning process into three distinct phases, managed by three separate roles: *Thinker*, *Planner*, and *Executor*. The *Thinker* reasons the *internal status* exhibited in the dialogue context

considering the comprehensive linguistic cues underneath the multiple dialogue interactions [49, 23], such as the user's emotional states or preferences, and formulates a blueprint of plans, serving as a global guideline for the *Planner* and *Executor*. The *Planner* needs to generate specific and executable plans in a natural-language format to call different *cognitive tools* (*sources* or *strategies*), while the content varies across tasks. The last *Executor* strictly follows the thought of the *Thinker* and the plan of the *Planner* to execute, and assemble all intermediate results to compose the final response.

In this way, our proposed method surpasses several competitive baselines, leading to better performance and generalization across different datasets in various applications. More details can be found in the original paper [1].

2.5.2 Self-Reasoning Language Models

After we successfully validate the effectiveness of these internal cognitive tools via prompt engineering, one problem naturally appears: can we further utilize them to boost the performance of LLMs via supervised fine-tuning?

Several prior studies have explored various approaches to get better CoT tuning datasets, where most of them utilize either the LLM itself [70, 71] or external reasoning models [72] to generate (*or refine*) new (*or existing*) responses in the instruction-tuning dataset, leading to great improvements at various downstream tasks. Despite the effectiveness of these proposed methods, they still face several limitations. On the one hand, most of them focus on questions with verifiable answer such as math and code [26, 71], being infeasible for general instruction-tuning dataset. On the other hand, another line of work typically assumes access to more powerful models to refine each sample iteratively [72, 73]. Such methods suffer from performance plateaus or even degradation [74] and are inherently constrained by the capability ceiling of the powerful model.

To this end, we present *Self-Reasoning Language Models* (SRLM), which is capable to self-unfolding its own reasoning rationales and iteratively optimize itself, leading to enhanced overall capability (as shown in Figure 2.7). Specifically, we first create only few

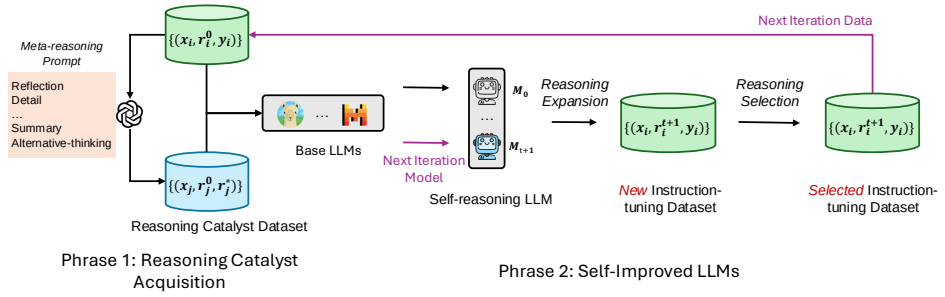


Figure 2.7: *The framework of our proposed Self-Reasoning Language Models, which consists of two phrases.*

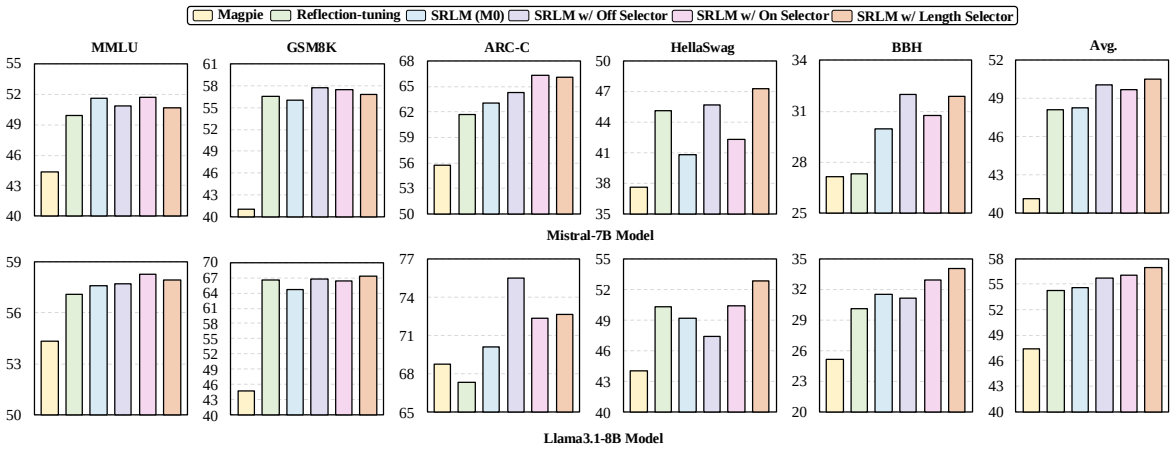


Figure 2.8: *The performance of Self-Reasoning Language Models (SRLM) across five different benchmarks, along with the average performance (Avg.). We report the best performance during the five iterations for each selector. We also run significant test which showcases our method significantly outperforms the reflection-tuning with* $p < 0.05$.

reasoning catalyst data that compose the demonstrations of how to enrich shorter CoT rationales into more longer and comprehensive CoT with the augmentation of various internal cognitive tools. After incorporating the reasoning catalyst data with the original instruction-tuning data, the tuned model not only inherit the basic reasoning capabilities from the instruction-tuning dataset, but also learn how to refine reasoning simultaneously, resulting in *Self-Reasoning Language Models*. Consequently, the SRLM can refine its own reasoning rationales at each iteration with the processing of reasoning expansion and selection. During this process, the model generates enriched reasoning rationale candidates for the same instructions in the original instruction-tuning dataset. These rationale pairs are then

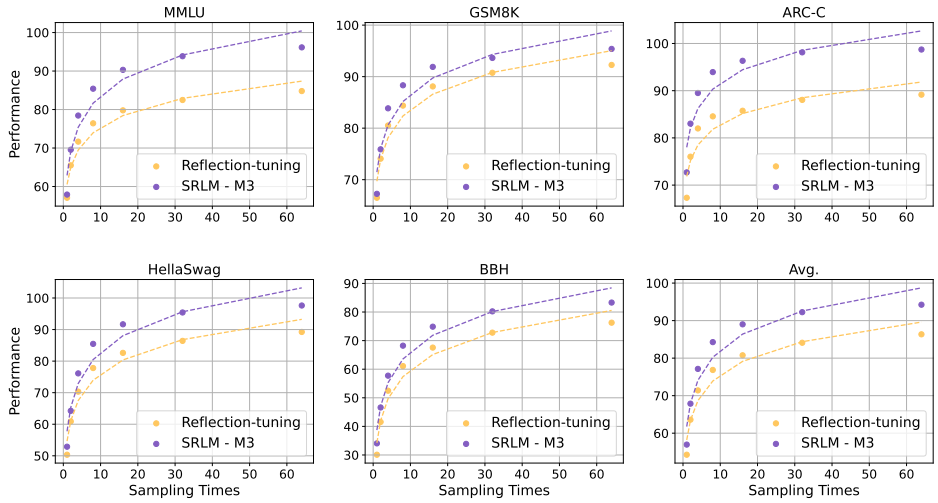


Figure 2.9: *The performance of the baseline and our proposed SRLM ($\mathcal{M}_3$ with length selector) different sampling times ranging from 1 to 64 (i.e., 1, 2, 4, 8, 16, 32, 64) on five various benchmarks and the Avg. performance.*

filtered and selected using three proposed selectors without any prior assumption about the instruction and answer. Finally, the newly selected instruction-tuning dataset is combined with the reasoning catalyst data to create the training data for the next iteration of SRLM, which is initialized from the same base model.

In this way, our proposed SRLM achieves an average absolute improvement of more than +2.5 points across five reasoning tasks: MMLU, GSM8K, ARC-C, HellaSwag, and BBH on two backbone models (Figure 2.8). Moreover, it brings more improvements with more times of sampling during inference, such as absolute +7.89 average improvement with 64 sampling times (Figure 2.9), revealing the in-depth, diverse and creative reasoning paths in SRLM against the strong baseline. More details can be found in the original paper [19].

2.6 Summary

In this chapter, we first build a benchmark to evaluate the *helpfulness* and *acceptness* of responses generated by current LLMs, considering three major linguistic cues of user statuses. We then propose a *Cue*-CoT to trace the status of users, decomposing the response generation into multiple reasoning steps. Experimental results demonstrate the superior

performance of our method on 6 datasets under both zero-shot and one-shot settings.

Besides that, we further explore more internal cognitive tools in dialogue system by introducing Think-Plan-Execute (TPE) prompting paradigm, which operationalizes multi-persona collaboration to simulate compositional planning and execution in dialogue systems. Based on these insights, we present Self-reasoning Language Models (SRLM), a novel fine-tuning framework that enables LLMs to iteratively improve their reasoning quality by leveraging internal cognitive tools to generate richer, deeper, and more creative rationales.

Together, these contributions highlight a promising direction for enhancing the epistemic capabilities of language models by treating internal cognitive tools not merely as passive artifacts of pretraining, but as active components in both generation and learning. This dual perspective, prompt engineering and supervised fine-tuning, paves the way toward more adaptive, insightful, and self-improving LLMs.

Chapter 3

SAFARI: Building Agents with External Physical Tools

3.1 Introduction

Besides internal cognitive tools, there are lots of external physical tool in the real world, such as calculators, search engines, models, and even physical robots [75, 76]. These tools serve as important extensions for static LLMs, enabling them to interact more dynamically with the external world. This integration not only addresses key limitations, such as the inability to access up-to-date information, but also unlocks the potential for LLMs to tackle more complex, interactive, and real-world tasks.

Figure 3.1 illustrates several representative types of external physical tools, including search engine [27], function calls [13], program executor, and even different atomic actions used by embodied agents. Notably, the functionality of certain external tools may overlap with that of internal cognitive tools, while others serve as complementary resources. For example, when presented with a factual question, an LLM might either (i) rely on its internal parametric knowledge to answer it via chain-of-thought reasoning (i.e., generate-then-read), or (ii) first retrieve relevant information from an external search engine and then generate the answer (i.e., retrieve-then-read). We will explore such trade-off in detail in later chapters.

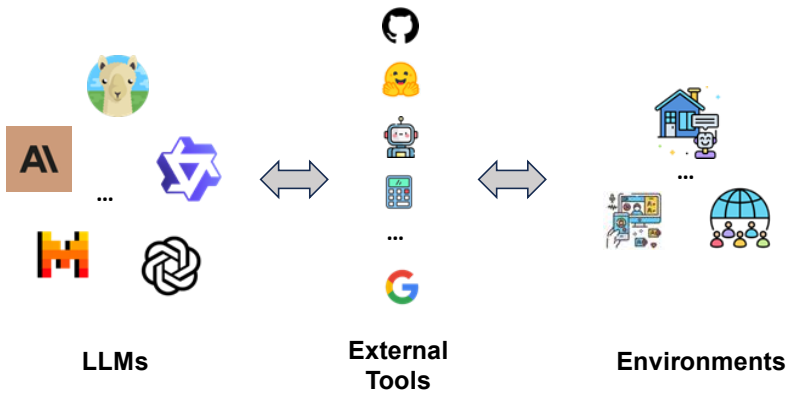


Figure 3.1: *Some representative external physical tools, serving as a bridge to connect LLMs with external environments, including models at Huggingface, code repositories in Github and lots of other external tools.*

In this chapter, we alternatively focus on the second case that the external physical tools mainly provide supplementary resources, bolstering LLMs' capacity to integrate domain-specific knowledge and enhance their generation capability. Specifically, we starts with one typical external physical tools – search engine (or more generally, a retriever), to empower the LLMs to dynamically call different sources of knowledge (i.e., user memory and private databases) in the context of dialogue system. We then introduce several learning frameworks that enable LLMs to call external tools more effectively and efficiently by learning from both demonstrations and interactions.

3.2 Related Work

Knowledge-grounded DS. To build a persona-consistent dialog agent, Zhang et al. [60] extensively investigates this task with a new dataset Persona-Chat, where a pre-defined persona set is a form of multiple sentences of textual description. Lots of works follow this setting and have taken mutual persona perception [77, 78], persona-sparse scenario [35, 79], long-term persona memory [80], and persona extending [81] into consideration. Although some of them complement the insufficient semantics in short persona descriptions by further utilizing an external commonsense knowledge base to extend existing persona sets [81], they still fall into the conventional framework coupling the knowledge selection with the

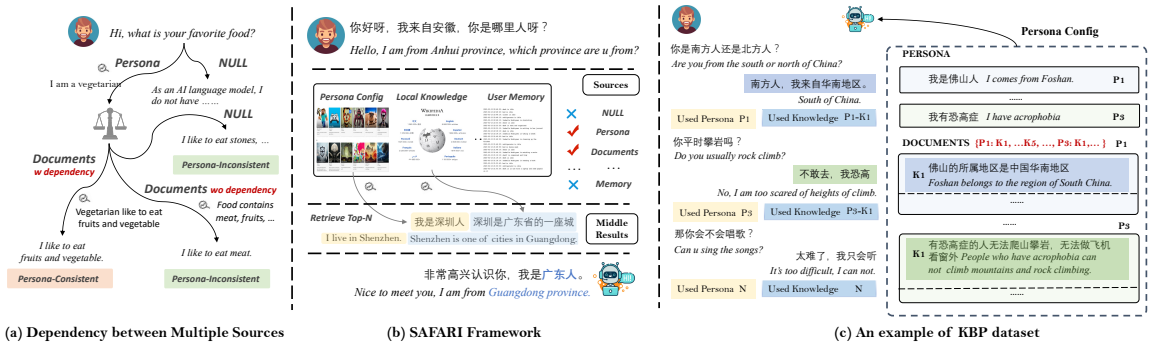
response generation [82], rendering it infeasible to handle various sources of knowledge. There have also been works showing that the combination of different knowledge sources such as persona descriptions and Wikipedia can further improve the overall performance [68, 83]. But they still fail to capture possible dependency between knowledge sources. In their framework, knowledge is not used as the role to assist persona-consistent response generation, but as an additional resource to generate a more informative response or select a suitable persona [68, 84].

LLM for Planning. Large Language Models (LLMs) show remarkable capabilities in planning the use of various external resources, such as tools [85], models [86], and APIs [13], to solve various NLP tasks and suit different applications in practice. Alternatively, different types of knowledge can be retrieved from external sources, as illustrated in WebGPT [87] and ReAct [11]. Integrating various knowledge sources to improve the quality of LLM generation becomes increasingly challenging due to the need for strategic planning, sequential decision-making, and complex reasoning. Previous research primarily focuses on either earlier decision-making stages [86, 87] or the subsequent response generation [85, 88], instead of establishing a complete framework for planning the use of multiple knowledge sources to generate appropriate responses.

3.3 SAFARI: LLMs as Tool Planner in Agentic RAG

3.3.1 Overview

Knowledge enhancement techniques [89] have significantly empowered machines to deepen their understanding of the underlying knowledge in open-domain dialogues [90], surpassing what can be solely acquired from conversational corpora. Recent years have witnessed various open-domain dialogue systems relying on different types of knowledge sources, such as external documents (*e.g.*, Wikipedia) [91, 92], persona [36, 60], user memory [80, 93], and more. Realizing the limitations of using single-source knowledge, some latest studies further develop dialogue systems with access to multi-source knowledge [68, 82, 83].



(a) Dependency between Multiple Sources (b) SAFARI Framework (c) An example of KBP dataset

Figure 3.2: *(a) An example of dependency of two sources involved in the persona-consistent dialogue system (*`PERSONA` *and* `DOCUMENTS`*); (b) our proposed SAFARI framework to plan, retrieve, and incorporate multiple sources of knowledge:* `PERSONA`*,* `DOCUMENTS`*, and so on.* ***Planning****,* ***Retrieval*** *and* ***Assembly*** *steps are divided by dashed lines; (c) A sample from the KBP dataset. There are three situations of responses in our datasets: 1) response without the need for any sources (*`NULL`*), 2) response using only personae description (from* `PERSONA` *source), and 3) response using both persona and knowledge (from* `PERSONA`*,* `DOCUMENTS` *sources). The example here presents the first and third situations. We highlight the response and used knowledge with the same color.*

Despite the effectiveness of existing works to enrich the dialogue responses with multi-source knowledge, they typically design models to incorporate all sources indiscriminately, resulting in a cumbersome process that struggles to handle cases dependent on the interaction between some specific sources instead of all [82, 84]. Moreover, the importance of comprehending the potential dependency between knowledge sources is overlooked in previous works, which may result in generating paradoxical responses [68]. For example, humans often express their persona with the assistance of external knowledge. As shown in Figure 3.2(a), for responding to the question "*Hi, what do you like to eat?*", it is inadequate to only incorporate single-source knowledge from user persona, *e.g.*, "*I am a vegetarian*", since relevant information from external documents is also required, *e.g.*, *Vegetarian* (*Vegetarians like to eat fruits and vegetables*). However, being unaware of the dependency between these two different sources of knowledge (persona and documents), dialogue systems may select the document implying inconsistent personas (*e.g.*, "*Food contains meat, fruits, ...*"), leading to responses conflicting with defined personas (*e.g.*, "*I like to eat meat*"). Therefore, it attaches great importance in modeling the interaction and dependency of different sources in building knowledge-grounded dialogue systems.

In order to address the interaction and dependency issue between specific sources, we propose a novel framework, named ***S**ource pl**A**nner **F**or person**A**lized knowledge-g**R**ounded d**I**alogues* (**SAFARI**). Seeing the potential of large language models (LLMs) in planning the use of external information [85, 86], we explore LLMs' capability of connecting different sources in the context of the personalized knowledge-grounded dialogue system. As illustrated in Figure 3.2(b), the whole response generation can be modeled into three steps in SAFARI: 1) **Planning** to make a series of decisions of whether to use a specific knowledge source by regarding each knowledge source as a external tool, given the relationship descriptions between different sources; 2) **Retrieval** to retrieve *top-n* results from external databases according to the decisions; 3) **Assembling** to incorporate all retrieved knowledge into the final response generation. Benefiting from decoupling source selection and response generation, our framework is more flexible and scalable, allowing independent modification of each component. Additionally, our framework can easily accommodate scenarios where multiple or no sources are required. To sum up, our contributions are listed below:

- We propose the SAFARI framework to augment the dialogue system to plan and incorporate multiple sources of knowledge into responses, and further address the knowledge dependency issue between sources in both supervised and unsupervised manner by leveraging LLMs.
- We build a personalized knowledge-grounded dialogue dataset, KBP, where the responses are conditioned on multiple sources of knowledge, leading to more user-engaged dialogues with informative and persona-consistent knowledge.
- We conduct exhaustive experiments to validate the effectiveness of our proposed framework to incorporate multiple sources and capture the dependency between them.

3.3.2 Task Definition

We first provide a general definition of a dialogue system that requires multiple sources and then we instantiate the definition in the context of personalized knowledge-grounded

dialogue. For each dialogue, the dialogue context $\boldsymbol{c} = \{u_1, s_1, u_2, s_2, ..., u_t\}$ and different knowledge sources $\boldsymbol{K} = \{K_1, K_2, ..., K_i\}$ are provided, where $K_i = \{k_i^1, k_i^2, ..., k_i^j\}$ indicates the i_{th} source's name of $\boldsymbol{K}$. k_i^j denotes the j_{th} knowledge in natural language from K_i. If K_2 is reliant on K_1, knowledge should be retrieved from K_2 based on the selected knowledge in K_1. Such reliance should also be embodied in the construction of K_2, in a way such as K_2 = $\{k_1^1 : \{k_2^1, k_2^2\}, k_1^2 : \{k_2^3, k_2^4\}, ...\}$. The goal of the system is to generate a response s_t conditioned on $\boldsymbol{c}$ and a set of knowledge $\{k_i^j, ..., k_n^m\}$ retrieved from $\boldsymbol{K}$ if required[1]. Specifically in the context of personalized knowledge-grounded dialogue, we regard $\{\texttt{PERSONA}, \texttt{DOCUMENTS}\}$ as $\{K_1, K_2\}$ respectively. There is a potential dependency between these two sources, and the goal is to generate a response s_t conditioned on a set of knowledge $\{p_i^j, ...k_m^n\}$, where p_i^j is retrieved from PERSONA and k_m^n is retrieved from DOCUMENTS. The response can also be generated conditioned on a set of knowledge from a single source PERSONA or without any sources.

3.3.3 Supervised *SAFARI*

There are three different steps in our proposed SAFARI framework: ***Planning***, **Retrieval**, and **Assembling** as shown in Figure 3.2(b). We will introduce them step-by-step under both supervised and unsupervised settings.

Planning The goal of the planning step is to make a series of decisions to decide whether or not the corresponding source of knowledge is required and determine their call order if needed. Since the dependency relationship is previously known, we only need to make sure that a certain knowledge source is called after the sources it depends on. Thus, we formulate this task as sequence-to-sequence generation by directly outputting either required sources in execution order or NULL if the response does not need any knowledge as follows:

$$\mathcal{M} : \boldsymbol{c} \rightarrow K_i, K_j, ..., K_n \quad or \quad \texttt{NULL}, \tag{3.1}$$

[1]Unlike most of the previous works, we also consider the response that does not need any knowledge in our setting.

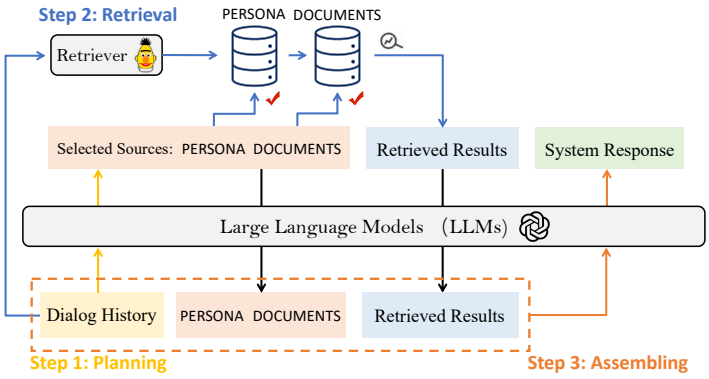


Figure 3.3: *The supervised framework of **SAFARI** for personalized knowledge-grounded dialogues. We use different colors to indicate different steps. The black arrow denotes the flow of data without the involvement of LLM.*

where $\mathcal{M}$ is parameterized by LLMs. We add $K_i, ..., K_n$ and `NULL` into the vocabulary of LLMs as special tokens. Besides that, we add other special tokens to indicate the different parts of the input, *i.e.*, `[SOURCE]` and `[EOS]` to indicate the start and end positions of sources. In this way, LLM can model the dependency between different sources and learn when and how to call certain sources.

Retrieval According to the output of the last step, there are two cases in this step: (1) the response does not need any external sources of knowledge, and the agent can skip this step; (2) the response needs multiple sources of knowledge, and the agent strictly follows the output source order to retrieve *top-n* related knowledge k_i^* for the i_{th} knowledge source according to the dialogue context $\boldsymbol{c}$, and if there is a dependency here, it will use preceding retrieved results k_j^* in the planned execution order as a filter. Specifically, assuming the output order is `PERSONA`, `DOCUMENTS` in the planning step for a persona-consistent dialogue system, we first retrieve *top-1* result p^* from `PERSONA`, and then we retrieve k^* from `DOCUMENTS` according to $\boldsymbol{c}$ and p^*. Here the utilization of p^* depends on the systematic design. For example, if there is a designated source K^* for each p^* (a.k.a dependency), we can simply retrieve k^* from K^*. And there is another case where all knowledge is stored together and we can concatenate $\boldsymbol{c}$ and p^* as a query in the retriever.

$$\mathcal{R} : K_i, K_j, ..., K_n \rightarrow k_i^j, ..., k_n^m \tag{3.2}$$

Assembling We concatenate all preceding results together with the dialogue context $\boldsymbol{c}$ to generate the response:

$$\mathcal{M} : Inp \rightarrow s_t, \tag{3.3}$$

where $Inp = \{\boldsymbol{c} \quad \texttt{[SOURCE]}K_i, ..., K_n \texttt{[EOS][MIDDLE]}k_i^j, ..., k_n^m \texttt{[EOM]}\}$. `[MIDDLE]` and `[EOM]` represent the start and end positions of retrieved results. Forming the input in this way has two advantages. Firstly, the name of the sources indicates the type of results retrieved, which provides more signals to the LLMs. Secondly, it allows us to train the language model in a multi-task manner using teacher forcing. The loss is only calculated on tokens related to the planning and the response as shown in Figure 3.3. We first predict the planning and then generate the response according to the preceding results when inference.

3.3.4 Unsupervised *SAFARI*

Inspired by the recent progress using LLMs as a controller to plan a call order of different models [86], we adopt a similar way here by providing detailed prompts to leverage the LLMs' capability to address the dependency between different knowledge sources. We consider two settings: zero-shot and in-context learning for planning and assembling steps here since the retrieval step is the same as above.

Planning. Instead of directly providing supervision signals, we provide a description for each source of knowledge, accompanied by the corresponding dependency between the sources. The prompts are shown in Table 3.1.

Assembling. We feed the dialogue content and the retrieved knowledge into the prompt as organized in Table 3.2, adapting LLMs to generate responses according to dialogue context and retrieved results.

There are different knowledge bases storing relevant information:
K_1: {K_1_DESC}
K_2: {K_2_DESC}
.....
There exists a dependency between these knowledge bases. **{DEPENDENCY_DESC}**
Here is the dialogue between the user and the system: **{DIALOGUE}**
Based on the user's last question, please determine if it requires invoking the corresponding knowledge base. If the invocation is necessary, output the names of the knowledge bases in the order they should be invoked. If no invocation is needed, output **NULL**.

Table 3.1: *The zero-shot prompt of unsupervised SAFARI at planning step (translated from Chinese to English).*

The dialogue is as follows:
{DIALOGUE}
The following knowledge is retrieved from different sources of knowledge bases:
{MIDDLE_RESULTS}
Please play the role of the system and generate a reply according to the context of the dialogue and given knowledge. Please make sure your reply is consistent with the given knowledge. If the provided knowledge is **NULL**, generate a response solely based on the dialogue context.
System:

Table 3.2: *The zero-shot prompt of unsupervised SAFARI at assembling step (translated from Chinese to English).*

The full prompts of the unsupervised planning step can be found in the original paper. For few-shot in-context learning, we prepend three corresponding demonstrations from the train set to the zero-shot prompts during evaluation.

3.4 *Knowledge Behind Persona* Dataset Collection

In this section, we detailedly introduce the process of data collection and statistics of the collected data. The data collection process can be divided into two steps: **Step 1. Persona**

and Knowledge Acquisition and **Step 2. Dialog Collection**.

3.4.1 Persona and Knowledge Acquisition

Seeds Preparation. To reduce annotation cost, we take advantage of the currently available persona dialogue dataset: DuLemon [80] and two widely-adopted Chinese knowledge bases: Baike[2] and Ownthink[3] to produce seed data. Specifically, we first cluster all persona sentences from DuLeMon into 10 topics. After removing duplicate, rare, and similar personas, we carefully choose around 20 personas for each left topic as seed personas[4]. In addition, we manually add some personas for the existing topics and new topics. The final personas consist of *age, nation, personality, career, movie, music, sport, book, constellation, locality, gender, others*. For retrieving persona-related knowledge, we simply combine two aforementioned knowledge bases with similar filtering operations and store them as *(head entity, attribute, tail entity)* tuples.

Persona and Knowledge Matching. For each persona sentence, we segment it into a sequence of words with a Chinese word segmentation tool jieba[5]. If any words exactly match the head entity or tail entity of a certain knowledge tuple, we transform the tuple into a sentence according to pre-defined templates and then save it as one knowledge for this persona sentence. In this way, we can obtain various knowledge for each persona sentence. Consistent with previous works [60], we randomly sample 3 persona sentences along with 5 knowledge sentences per persona to form a persona description of the system for each dialog.

---

[2] http://www.openkg.cn

[3] http://github.com/ownthink/KnowledgeGraphData

[4] Some topics are removed if they contain less than 20 personas. We don't pick hundreds of personas because one persona sentence has a vast amount of knowledge behind it.

[5] https://github.com/fxsjy/jieba

3.4.2 Dialogue Collection

Collection Setting. Following the previous setting [68], annotators are instructed to make a dialogue by considering persona and corresponding knowledge under the single-person setup. In this way, one person can better understand what persona to ask as the human and what knowledge to use as the system, in comparison with two independent persons playing two roles separately. During the collection, each annotator first selects a suitable persona and then optionally identifies relevant knowledge, giving a knowledge-enhanced and persona-consistent response at last.

Training and Pilot Annotation. All annotators are first required to take a training tutorial to learn the annotation procedure, requirements, and examples of annotated dialogues. Afterwards, they are given 30 personas to make 10 dialogues. We provide corresponding feedback to help them adjust their annotation criteria. To establish the necessity of persona-knowledge dependency, we consider the situation where the response will be persona-inconsistent without the assistance of knowledge. To this end, annotators are requested to ask questions centered on the implications based on knowledge and persona. For example, the annotator is supposed to ask *"Which province are you from?"* instead of *"Which province does Shenzhen belong to?"*, given the persona *"I live in Shenzhen"* and corresponding knowledge *"Shenzhen belongs to Guangdong province"*.

Batch Collection. After pilot annotation, we conduct dialogue collection batch by batch and regularly coach the quality of collected data[6]. For each batch, we sample personas different from previously annotated dialogues to increase its diversity in the whole dataset. The addition, deletion, and revision of persona and knowledge are also accepted and updated at the batch level[7].

[6]We write a python script to check typos (*e.g.* the labels of used knowledge is not exist in given knowledge bases) and provided feedback after each batch.

[7]The annotators must check for persona conflicts and refrain from relying too much on single knowledge.

KBP	Train	Valid	Test
# dialogues	1,981	248	248
# samples	9,821	1,227	1,229
# avg turns	4.96	4.93	4.96
# utterances	19,642	2,454	2,458
# avg length	17.6	17.3	17.5
# resp w/ persona	86.1%	84.4%	85.3%
# resp w/ p_and_k	76.3%	74.2%	75.1%

Table 3.3: *Statistics of KBP dataset.*

Grounding Annotation. We also gather the labels of grounding knowledge sources for the system's responses by asking the annotators to specify the sources they draw from while providing responses, such as `PERSONA` or `DOCUMENTS`. For instance, generating a response may rely on persona alone or both persona and knowledge. With the labels of these grounded sources, the planning abilities of the dialogue systems can be quantitatively measured.

3.4.3 Statistical Analysis

We organize the collected personas (`PERSONA` source), persona-related knowledge [8] (`DOCUMENTS` source), and dialogues in the form shown in Figure 3.2(c). We finally collect 2,477 dialogues and 24,554 utterances with 5 turns per dialogue on average. We then split the collected data into train, validation, and test sets using the 8:1:1 ratio. The dataset statistics are summarized in Table 3.3, including the number of dialogues, utterances, average length, as well as data sources used. The average length per utterance reaches 17.6, hinting at the informativity and depth of the conversation. It is shown that over 86% of responses used persona (*i.e.*, resp w/ persona) and 76% used both persona and knowledge (*i.e.*, resp w/ p_and_k), which shows that KBP is capable as a benchmark to evaluate the different grounding abilities of models.

[8]The knowledge related to the same persona forms a document, and different documents form `DOCUMENTS` source.

3.5 Experiments

3.5.1 Setup

Evaluation Metrics. During the evaluation, we use different metrics at different steps. We use F1 for planning, Recall@1 for retrieval, and BLEU [94], Rouge-L [95], P.C, K.C for assembling, in which P.C and K.C are calculated using our finetuned NLI models [96].

Implementation Details. We mainly choose `BELLE-LLAMA-7B-2M` [65] and `ChatGLM-6B` [64] as two backbone models for supervised setting since they are two popular open-source Chinese models. And we additionally add `ChatGPT` (gpt-3.5-turbo-0301)[9] for the unsupervised setting. For training, we set the batch size as 8, train models with 3 epochs and save the checkpoint with the lowest validation loss. For other hyper-parameter settings, we mainly follow the corresponding official code[10]. Due to the computation limit, we conduct training with LoRA [97] at one single 3090 GPU, and it cost about 4-6 hours. For the unsupervised setting, we set both the temperature and top p as 0.1 to reduce the randomness of LLMs. Besides that, we use three types of retrievers including both sparse and dense retrieval: BM25 [98], DPR [99], and RocketQAv2 [100]. We only retrieve the top-ranked result from each source in the experiments.

3.5.2 Performance of *Planning*

There are three types of decisions representing different sources required in the next step: `NULL`, `PERSONA`, and `Both` (selecting both `PERSONA` and `DOCUMENTS`). Table 3.4 demonstrates the F1 of planning under different settings. Under supervised settings, despite LLMs achieving high F1 scores at `Both`, the performance at `NULL` and `Persona` is still unsatisfactory, since there are fewer training samples in these two cases.

On the other hand, under unsupervised settings, the LLMs are over-confident in their

[9] https://openai.com/blog/chatgpt

[10] https://github.com/THUDM/ChatGLM-6B and https://github.com/LianjiaTech/BELLE

Model	NULL	Persona	Both
Supervised			
BELLE-LLAMA-7B-2M	42.67 (194)	14.08 (17)	83.77 (1018)
CHATGLM-6B	**47.10** (129)	**31.96** (69)	**86.59** (1031)
Unsupervised			
Zero-shot			
BELLE-LLAMA-7B-2M	**28.55** (940)	8.94 (54)	32.47 (235)
CHATGLM-6B	25.60 (1225)	0.0 (0)	0.43 (4)
CHATGPT	11.45 (116)	**20.67** (233)	**74.88** (880)
In-context			
BELLE-LLAMA-7B-2M	9.22 (36)	18.21 (1193)	0.0 (0)
CHATGLM-6B	25.67 (1190)	1.49 (9)	4.62 (30)
CHATGPT	**27.95** (699)	**23.14** (238)	**41.98** (292)

Table 3.4: *The F1 of different decisions in **Planning** of different LLMs under supervised/unsupervised settings. We also report the frequency of different decisions in the bracket. There are 181* `NULL`*, 125* `PERSONA` *and 923* `PERSONA`*, and* `DOCUMENTS` *in the ground planning.*

decisions to use `NULL`, and they misunderstand the dependency between different sources (sometimes deciding to only use `DOCUMENTS` without `PERSONA`)[11].

This result reveals the LLMs' low accuracy in expressing uncertainty and fetching unknown knowledge. Furthermore, in-context learning cannot improve this situation, which is similar to the observation in Amayuelas et al. [101].

3.5.3 Performance of *Retrieval*

With the ground-truth planning labels (except `NULL`), we examine three types of retrievers, including **BM25**, **RocketQAv2**, and **DPR**, to evaluate the retrieval performance. Table 3.5 presents the Recall@1 (R@1) of the different retrievers. We found that the DPR and RocketQAv2 can achieve over 80% R@1 when retrieving from `PERSONA` source while only about 50% from `DOCUMENTS` and the R@1 at `DOCUMENTS`† further decreases after removing the dependency. First, the semantics between different knowledge from `DOCUMENTS` with the dependency are similar to the same underlying persona p^*, making them more difficult to be distinguished. In addition, noisy knowledge sentences are introduced since there exists no dependency.

[11]We assign the case that LLMs predict `DOCUMENTS` only as `NULL` since this case does not exist in KBP.

Model	Persona	Both		
		PERSONA	DOCUMENTS	DOCUMENTS†
BM25	36.80	48.97	15.05	11.37
RocketQAv2	80.00	92.31	50.49	35.75
DPR	**83.20**	**93.07**	**51.67**	**39.33**

Table 3.5: *The performance of **Retrieval** of different types of retrievers. There are 125 examples that only require* PERSONA *and 923 require both* PERSONA *and* KNOWLEDGE*. We also report the Recall@1 of* DOCUMENTS *without dependency (*DOCUMENTS†*).*

Moreover, we observe that DPR performs the best out of these three retrievers in all sources of knowledge while BM25 performs worst[12], revealing the importance of dense retrieval models in this task. Therefore, we set **DPR** as the retriever in our experiments afterward.

3.5.4 Performance of *Assembling*

Table 3.6 demonstrates the performance of response generation under both supervised and unsupervised settings.

Referring to Table 3.4, the performance of the planning step largely affects the results in the assembling step, when the retriever is the same. Mostly, better planning leads to better responses in all metrics. The supervised models are much better than unsupervised models since their planning results are much better, while ChatGPT performs best under unsupervised settings due to a similar reason. We found that BELLE achieves higher BLEU1 and Rouge-L, K.C but lower P.C than ChatGLM since the planning gap between them mainly comes from PERSONA. In addition, due to poor retrieval performance at DOCUMENTS (Table 3.5), the consistency score K.C is also much lower than P.C.

With demonstrations in the prompt, we observe generally better performance on most metrics, since LLMs tend to accept the personalized role, rather than generating responses like "*As an AI language model, I do not have persona ....*". Overall, we conclude that the grounding ability of supervised models is much better than unsupervised ones, and ChatGPT

[12]RocketQAv2 is generally not competitive with DPR because of the pre-trained weights in the RocketQAv2, since it is pre-trained using QA datasets and the length of the question is much shorter than dialogue context.

Model	BLEU1	Rouge-L	P.C	K.C
Supervised Setting				
BELLE-LLAMA-7B-2M	**30.48**	**34.61**	75.34	**46.62**
CHATGLM-6B	24.59	27.18	**76.99**	42.39
Unsupervised Setting				
Zero-shot				
BELLE-LLAMA-7B-2M	11.84	19.24	30.59	27.34
CHATGLM-6B	6.18	14.50	14.73	24.73
CHATGPT	12.06	24.44	**73.47**	**38.00**
In-context				
BELLE-LLAMA-7B-2M	**19.51**	22.25	72.98	24.89
CHATGLM-6B	13.74	19.69	16.92	24.89
CHATGPT	16.03	**25.62**	46.38	35.56

Table 3.6: *The performance of **Assembling** under supervised/unsupervised settings.*

Model	BLEU1	RougeL	P.C	K.C
CHATGLM-6B	23.81	26.70	76.99	42.39
+ Ground Planning	24.29	27.01	86.16	57.12
+ Ground Retrieval	**25.86**	29.15	79.52	53.95
+ Ground P & R	25.71	**29.43**	**90.56**	**72.99**
- Dependency	23.32	25.53	75.67	38.49
- Documents	23.06	25.34	75.91	36.53
- Planning*	23.51	25.98	72.90	24.89
- Planning**	23.69	26.81	71.60	34.91

Table 3.7: *Ablation study on the impact of different steps and modules in SAFARI.*

performs best under the unsupervised setting.

3.5.5 Discussion and Analysis

In this section, we analyze the effects of different components and the choice of the number of retrieved results, based on `ChatGLM` under the supervised setting. In addition, we conduct human evaluations to verify the quality of automatic evaluations.

Impacts of Different Steps. We investigate the effects of individual steps by providing the model ground-truth labels from each step to generate the response, enabling us to analyze and understand the specific effects of each step in a clear and systematic way. Table 3.7 presents the results. First, we note that the inclusion of ground-truth planning labels or

Number	Assembling			
	BLEU1	RougeL	P.C	K.C
1	**23.81**	**26.70**	**76.99**	**42.39**
2	22.70	25.57	71.03	29.45
3	20.69	24.05	69.73	27.91

Table 3.8: *The performance of **Assembling** of different number of retrieved results.*

knowledge casts a positive impact on performance. Planning primarily enhances P.C and K.C, while grounding knowledge contributes to BLEU1 and Rouge-L scores. The best results are obtained when both signals are combined. Secondly, we also conduct an ablation study by removing some modules: 1) removing dependency information (-Dependency); 2) removing `DOCUMENTS` and only using `PERSONA` (-Documents); and 3) removing the planning step by always selecting `PERSONA` (-Planning*) or always selecting `PERSONA` and `DOCUMENTS` (-Planning**) for each turn. It can be found at the bottom of Table 3.7 that all metrics are dropped differently after removing different components except for Rouge-L when always selecting two knowledge sources. To conclude, SAFARI can effectively incorporate multiple sources (compared with -Documents) and further address dependency issues (compared with -Dependency). Moreover, SAFARI demonstrates its versatility by effectively handling multiple sources and efficiently selecting relevant ground knowledge. Notably, SAFARI outperforms existing methods that indiscriminately utilize all available sources (compared with -Planning**).

Different Numbers of Retrieved Results. The number of retrieved results plays a key role in the response generation. There is a trade-off between accuracy and recall, while a small number of retrieved results may not cover enough semantics but a large number may introduce additional noises. Table 3.8 presents the results of the different numbers of retrieved results. We observe that the performance of response generation decreases with the number, which indicates that noisy knowledge will harm the quality of the generated responses.

Model	Coh.	Per.Cs (%)	Know.Cs (%)
Supervised Setting			
BELLE-LLAMA-7B-2M	4.38	72.0	63.8
CHATGLM-6B	4.06	68.0	59.1
Unsupervised Setting			
Zero-shot			
BELLE-LLAMA-7B-2M	2.84	24.7	19.5
CHATGLM-6B	2.58	17.0	14.8
CHATGPT	4.00	63.4	33.3
In-context			
BELLE-LLAMA-7B-2M	3.36	40.0	21.7
CHATGLM-6B	2.88	32.0	28.9
CHATGPT	4.03	54.0	48.8

Table 3.9: *The results of human evaluation. The inter-agreement is about 86%.*

Human Evaluation. Human evaluation is conducted to evaluate the quality of generated response in terms of three metrics: coherence score (**Coh.**), persona consistency score (**Per.Cs**), and knowledge consistency score (**Know.Cs**). We randomly sample 100 responses with grounding information for each model and ask three annotators to indicate its coherence score (1-5) and whether the response is consistent with the given persona (1/0), and knowledge (1/0). Table 3.9 shows the result. We observe that supervised methods achieve higher performance than unsupervised ones, which corroborates the findings of the automatic evaluation results presented in Table 3.6. Besides that, we found `BELLE` achieves the highest performance across all metrics and outperforms `ChatGLM` since the effects of planning are not considered during human evaluation. Moreover, we also found that in-context learning brings a lower rejection rate and more human-like responses. Specifically, the rejection rate of `BELLE` under the setting of zero-shot learning is about 32%, while the number is reduced to 12% under in-context learning.

3.6 Learning of Tool Planning

There are two major ways to teach the LLMs to use various external physical tools: 1) learning from demonstrations; and 2) learning from interactions. Each paradigm provides complementary strengths for different stages of tool-use capability development.

3.6.1 Learn from Demonstrations

Similar to how humans can acquire new skills by observing examples, LLMs can also learn tool-use behaviors by memorizing curated demonstrations. This learning paradigm typically relies on high-quality examples, either from human annotations or model-generated traces, and enables the model to generalize tool-use patterns via prompting engineering or supervised fine-tuning without requiring real-time feedback.

Prompting Engineering. Most of previous methods fall into this category either use zero-shot prompting or in-context learning, to guide the model's behavior through carefully designed input templates or few-shot examples without modifying model weights [13]. Usually, It is required to provide detailed description about the tool, such as name and description, in order to help LLMs to quickly understand what the tool can do and how it can be used. Additional few-shot examples can optionally be included to illustrate specific tool usage scenarios, similar to the approach used in unsupervised SAFARI. While this approach requires no parameter updates, it heavily depends on the quality and relevance of the prompt examples, and may not generalize well beyond the demonstrated patterns.

Supervised Fine-tuning. To further improve the tool planning capabilities of LLMs, it is always a option to collect high-quality data and finetune the model accordingly [102, 14, 103, 104]. For example, WebShop [105] offers a web-based interactive environment where an agent can browse and purchase products. By leveraging behavior cloning, agents trained in this setting demonstrate non-trivial performance in selecting the correct product based on human instructions. In a related line of work, Gou et al. [102] introduce TORA - a suite of Tool-integrated Reasoning Agents designed to tackle complex mathematical problems by seamlessly combining natural language reasoning with external tools, such as computation libraries and symbolic solvers. Although supervised fine-tuning can significantly improve performance, its one-size-fits-all nature prevents the model from developing its own tool-use policy tailored to its capabilities and the complexity of the task, thereby limiting its ability

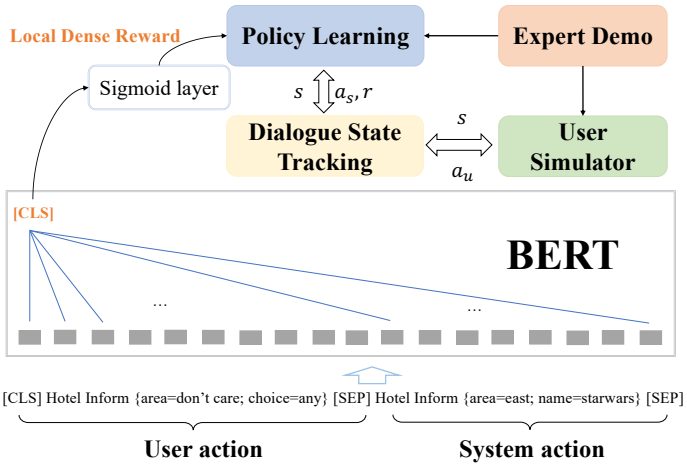


Figure 3.4: *Our proposed method: Integrating pre-trained language model into reinforcement learning as reward model to provide local dense reward signal.*

to generalize well [106].

3.6.2 Learn from Feedback

Recent advances in LLMs have demonstrated impressive reasoning capabilities when fine-tuned via reinforcement learning (RL) with simple rule-based rewards [26]. Therefore recent efforts have sought to extend RL to tool use policy of LLM by leveraging rule-based rewards tied to final answer correctness [27, 107]. One of the most notorious problems of RL is the reward sparse issue, where here the agent usually receive a positive (or negative) reward signal when the trajectory ends successfully (or unsuccessfully). Thus, the reward signal is delayed and sparse, making it extremely difficult to connect a long series of actions to a distant future reward especially for long-horizon planning tasks [108]. In order to provide dense reward signals, we directly integrate a pre-trained language model as a discriminator to judge whether the current action is good enough given current state (i.e., next action prediction), and then the trained discriminator can give an extra local dense reward to guide the agent's exploration [109], as shown in Figure 3.4.

Beyond reward shaping, long-horizon planning introduces another significant challenge: long-context understanding and management. Due to the inherent limitations of LLMs'

maximum context windows, maintaining coherent decision-making across long sequences remains a fundamental bottleneck for agent scalability. Several studies have explored the use of external memory modules to store and compress past interactions, allowing them to be retrieved on demand during later decision-making steps [80, 110]. However, designing memory systems that are both scalable and generalizable - particularly within reinforcement learning (RL) frameworks - remains an open research problem.

3.7 Summary

In this chapter, we propose a novel framework SAFARI to incorporate multiple sources of knowledge bases and further address the dependency issue between them. Unlike previous works, SAFARI can be extended to multiple sources easily and it can handle cases that do not require any sources or require some instead of all sources between them. We build the first personalized knowledge-grounded dialogue (KBP) dataset, and experimental results prove the effectiveness and robustness of SAFARI.

Furthermore, we discuss several ways to empower LLMs to use more diverse and complex external physical tools, such as learning from demonstrations or feedback. Each learning paradigm has its own advantages and challenges, but collectively offer a flexible toolbox for enhancing tool-use capabilities of LLMs and Agents. We highlight that these methods are not limited to traditional tools like search engines or calculators but can also be extended to broader APIs and services, such as invoking specialized models from Hugging Face, executing domain-specific functions, or interacting with robotic systems. This generalization paves the way for building more capable and adaptive autonomous agents in open-ended environments.

Chapter 4

Self-DC: Building Agents with Self-aware Tool Utilization

4.1 Introduction

Build on top of the unification of internal cognitive tools and external physical tools, it is important for the agent to decide which type of tool to use at each time step to achieve the pre-defined goal. Inspired by meta-reasoning theory [28], which models human intelligence as a continuous cycle of monitoring and control, agents should also assess their own epistemic state (e.g., confidence, solvability, rewards) and adaptively invoke different tools based on those assessments.

To build advanced agent intelligence that can learn more like humans do, there are two keys: i) forming internal knowledge of the world around them to learn, adapt, and forge plans efficiently; ii) actively interact with external world to gain new knowledge from trials and experience. Both of two components plays key roles for autonomous intelligent agents in open-ended, evolving environments. The former one empower the model the initial understanding and modeling of the world, while the latter enable it to learn from experience. As a baby learns how the world works largely by observation in the first few months of life, an agent should also be able to learn such knowledge from the experience

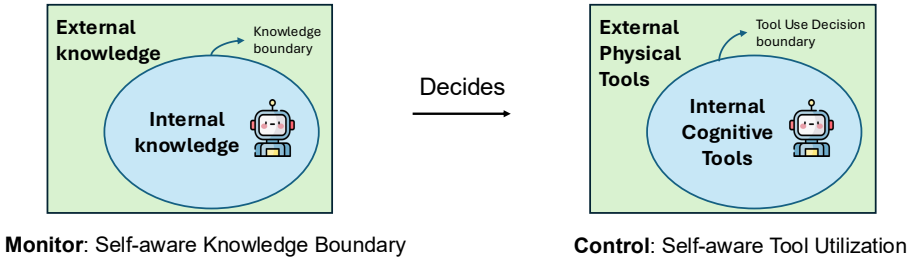


Figure 4.1: *The tool use decision boundary of agent should align with its knowledge boundary. This alignment represents the optimal behavior of agent that only invoke external physical tools when necessary.*

even with the unknown at the beginning.

Figure 4.1 illustrates the ideal situation that agent could monitor its own knowledge boundary and control its behavior accordingly. Specifically, given the task q, the agent should first monitor the knowledge required to solve task is known or unknown, or partly known or unknown if the task is too complex, and then actively interact with external world only when necessary to gain indispensable knowledge to solve it. This is a dynamic and complex decision-making processing, closely aligned with how humans reason and plan. To achieve such goal, two key components are essential:

Monitor: Assessment of Knowledge Boundary. The monitor serves as a self-aware module that probes the model's confidence, consistency, or uncertainty to estimate whether it possesses sufficient knowledge to proceed. An accurate monitor is essential for avoiding hallucinations, unnecessary computation, or incorrect tool usage. Specifically, the monitor needs to provide different metrics or signals either from internal or external side, i.e., self-evaluation or external feedback, ranging from initial judgment of solvability, intermediate confidence, rewards to the observations. Therefore, these implicit or explicit monitor signals can guide the controler of agent to call correct tools at each step, leading to correct answer.

Control: Decision of Internal and External Tools. Based on the current state and monitor's output, the controller decides whether to call internal cognitive tools or external physical tools, gain new knowledge and feedback, and provide the answer if all required knowledge

are accumulated. If the external physical tool is called, the control needs to pass the related parameters to the external word, i.e., an executor, to gain new knowledge.

Only in this way, the agent could not only provide correct answer, but also achieve it in a more efficient way, i.e., only call external physical tools when the required knowledge is not included in internal parametric space. Therefore, it is expected to minimize the number of actions to take in the real world to learn a task, as a truly autonomous agent [2].

In this chapter, we first develop a prompting framework that reduplicates the monitor and control processing for LLMs to solve complex problems, and then we further explore supervised fine-tuning (SMART [103]) and reinforcement learning (OTC-PO [111]) methods to empower more efficient and effective tool use policy of LLMs. We further provide an actionable roadmap to achieve the truly autonomous agent with minimized interactions.

4.2 Preliminaries

Before we delve into specific methods, we would like to introduce several principles regarding the knowledge boundary and decision boundary of agents.

4.2.1 The Definition of Knowledge and Decision Boundaries

Knowledge Boundary. At any time step t, let $\mathcal{W}$ represent the complete set of world knowledge. We define the model m's internal and external knowledge as:

$$\mathcal{K}_{\text{int}}(m,t) \subseteq \mathcal{W} \quad \text{and} \quad \mathcal{K}_{\text{ext}}(m,t) = \mathcal{W} \setminus \mathcal{K}_{\text{int}}(m,t)$$

where $\mathcal{K}_{\text{int}}(m,t)$ denotes the internal knowledge embedded in m, and $\mathcal{K}_{\text{ext}}(m,t)$ represents the external knowledge accessible from the world. The *knowledge boundary* is defined as the frontier between the two:

$$\partial\mathcal{K}(m,t) = \partial\mathcal{K}_{\text{int}}(m,t) = \partial\mathcal{K}_{\text{ext}}(m,t)$$

This boundary marks the epistemic limit of the model's internal knowledge. We assume

all internal or external knowledge is accurate to ensure simplicity and generalization.

Decision Boundary. Given a time step t, let $\mathcal{T}_{int} = \{t^1_{int}, ..., t^n_{int}\}$ be the set of internal cognitive tools and $\mathcal{T}_{ext} = \{t^1_{ext}, ..., t^m_{ext}\}$ the set of external physical tools. The *decision boundary* $\partial\mathcal{D}(m,t)$ is the point at which the model decides whether to use internal or external tools to acquire additional task-relevant knowledge:

$$\partial\mathcal{D}(m,t) = \partial\mathcal{T}_{\text{int}}(m,t) = \partial\mathcal{T}_{\text{ext}}(m,t)$$

where $\mathcal{T}_{\text{int}}(m,t)$ and $\mathcal{T}_{\text{ext}}(m,t)$ denote tool choices leading to internal or external knowledge acquisition, respectively.

In summary, the knowledge boundary defines the model's epistemic limits, while the decision boundary governs how the model navigates these limits through tool use. Each point in the knowledge space corresponds to a point in the decision space, reflecting how the model chooses to engage with that knowledge through tool use. In this way, the decision boundary operationalizes the knowledge boundary, shaping the model's policy for knowledge acquisition in pursuit of its goals.

4.2.2 Principle 1: Foundations

Lemma 1.1: *Over long horizons, scaling laws enable the expansion of $\mathcal{K}_{\text{int}}$; i.e., the knowledge boundary $\partial\mathcal{K}$ expands outward.* This expansion reflects the model's increasingly comprehensive internal representation of the world across modalities and domains. For instance, Sora [1] demonstrates the acquisition of rich physical knowledge, enabling the generation of realistic, coherent long-form videos. With sufficient training data, architecture, and optimization, the model effectively compresses the external world into its internal parameter space [112, 113]. As $\partial\mathcal{K}$ expands with scale, the model may ultimately support real-time abstraction of the world, or even autonomously discover knowledge beyond existing human understanding [114], leading toward AI for scientific discovery.

[1] https://openai.com/sora/

Lemma 1.2: *Continual learning methods such as SFT can reshape both the knowledge boundary and the decision boundary.* To stay current and improve performance, models must update outdated knowledge or acquire new information through continual learning, including prompting [115] and supervised fine-tuning [116]. These processes naturally shift the knowledge boundary to reflect updated internal states. In parallel, decision boundaries can be adjusted to improve tool use behavior, such as encouraging external tool invocation only when necessary [103, 117].

4.2.3 Principle 2: Uniqueness and Diversity

Across open-source and proprietary models, both unique and shared characteristics emerge. To better understand model capabilities and limitations, we posit that while each model has distinct boundaries, there also exist universal properties common to all.

Lemma 2.1: *Each model has its own knowledge boundary and decision boundary.* These boundaries differ due to variations in model size, architecture, training data, and learning objectives. Larger models trained on more diverse corpora tend to internalize a broader scope of world knowledge [113]. In contrast, decision boundaries are primarily shaped through explicit tool use training [14], leading to variation in how models interact with tools to acquire knowledge.

Lemma 2.2: *There exist minimal and maximal knowledge (and decision) boundaries across all models.* The *minimal knowledge boundary* $\partial\mathcal{K}_{\min} = \bigcap_{i=1}^{N} \partial\mathcal{K}^{(i)}$ represents the smallest common set of internalized knowledge shared by all models, regardless of their training setup. Conversely, the *maximal knowledge boundary* $\partial\mathcal{K}_{\max} = \bigcup_{i=1}^{N} \partial\mathcal{K}^{(i)}$ reflects the union of all internal knowledge across models, encompassing even niche or domain-specific knowledge found only in specialized systems. Analogously, minimal and maximal decision boundaries exist, though they are best interpreted as normative alignment goals rather than fixed, objective thresholds.

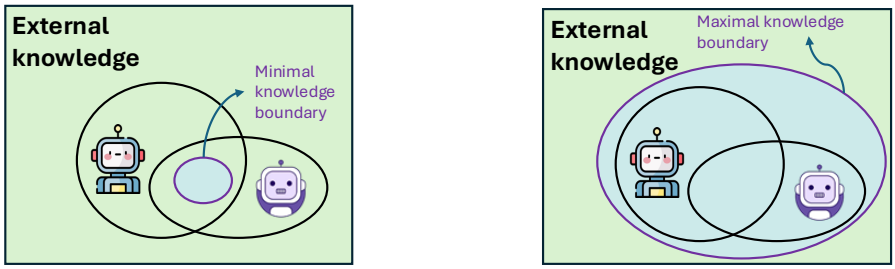


Figure 4.2: *A high-level illustration of Lemma 2.2 for the all models* $\mathcal{M} = \{m_0, ..., m_n\}$

4.2.4 Principle 3: Dynamic Conservation

Knowledge is inherently dynamic, continuously evolving as new facts emerge and old ones become obsolete. To capture this temporality, we propose the principle of dynamic conservation of knowledge, emphasizing how models must adapt to an ever-changing epistemic landscape.

Lemma 3.1: *At any time step* t*, the total world knowledge* $\mathcal{W}_t$ *is fixed and identical across all models.* Ideally, a model would internalize the entire knowledge set, i.e., $\mathcal{K}_{\text{int}}(m, t) = \mathcal{W}_t$, requiring no external tool use. This entails an aspirational endpoint for fully autonomous intelligence [2]. Practically, however, as $\mathcal{W}_t$ expands over time, models must also evolve to keep pace. If a model's epistemic growth outpaces that of the external world, this ideal state becomes theoretically attainable.

Lemma 3.2: *For any task or query* q *and model* m*, there exists a minimal and fixed epistemic effort* $N(q, m)$*, allocated between internal and external sources, that is necessary to solve the task.* This can be decomposed as $N(q, m) = k_{\text{int}} + k_{\text{ext}}$, where k_{int} reflects knowledge retrieved from the model's internal parameters and k_{ext} represents knowledge acquired through external tools. This formulation reveals several insights: (1) $N(q, m)$ is jointly determined by the complexity of the task and the capabilities of the model, indicating stronger models may satisfy most or all of N through internal reasoning ($k_{\text{int}} \to N$), while weaker models may depend more on external assistance ($k_{\text{ext}} \to N$) [118]. (2) Even models with limited internal capacity can

achieve high performance by dynamically offloading reasoning or retrieval steps to more capable tools or agents. This suggests a form of capability equivalence, where optimal tool use policies allow weaker models to simulate stronger ones. (3) The objective is not merely task completion, but the development of behavior policies that minimize epistemic effort while managing latency, cost, and cognitive load. In this view, intelligent behavior is defined not just by the correctness of outputs, but by the efficiency and adaptiveness of the pathways taken to reach them.

4.3 Related Work

Monitoring of LLMs To calibrate the known and unknown of LLMs, there are lots of studies that have delved into methods for estimating and quantifying *certainty and uncertainty* in LLMs predictions [119, 120, 121, 122]. There are two types of methods: 1) logit-based which utilize the model logits [119, 123]; and 2) non-logit-based methods, such as expressing uncertainty about its own answer in natural language [120], particularly with the rise of closed-source LLMs. More recently, Xiong et al. [121] benchmarks three categories of the first type: verbalize-based, consistency-based, and their hybrid methods. They find that LLMs exhibit a high degree of overconfidence when verbalizing their confidence, which can be alleviated by different prompting strategies (e.g., Chain-of-thoughts [20]) or more complicated methods (e.g., Self-consistency [124]). Moreover, different languages also trigger different level of certainty and uncertainty of language models [125].

Control of LLMs On the one hand, lots of previous methods investigate various methods to elicit the internal reasoning capability of LLMs [20, 1, 23], such as program-guided reasoning [126, 127], Self-Ask [128] and retrieval-augmented reasoning [129, 130, 131], especially for multi-hop questions [132] and in-depth dialogues [23]. On the other hand, it is important to empower the stateless LLMs to interact with external world with the augmentation of different tools [76]. Therefore, LLMs can perform tasks that go beyond their intrinsic knowledge such as retrieving up-to-date information [133, 134] and providing

domain-specific services by calling different functions / APIs [6]. However, only a few of them consider the relationship between internal reasoning and external acting, especially for compositional problems when the necessary unknown knowledge is required. To address this dilemma, we explore the better trade-off between internal reasoning and external acting in terms of effectiveness and efficiency.

4.4 Self-DC: Self Divide-and-Conquer for Compositional Questions

4.4.1 Overview

Large Language Models (LLMs) [66, 135] possess extensive world knowledge thanks to the scaling of size of pre-training data and model [112], resulting in exceptional capabilities to answer open-domain questions using internal known knowledge encoded in their parameters [47, 136]. However, due to the cutoff date of training data, it is difficult for them to answer questions out of their known knowledge (a.k.a., unknown questions), which necessitates the augmentation of external retrieval [137, 138, 139, 140], such as Google Search and Wikipedia.

To provide more accurate answers for the questions, most previous works tend to employ external retrieval methods indiscriminately without considering different types of questions, resulting in redundant retrieval and unnecessary cost [129, 131]. Alternatively, some methods simply classify questions into binary categories (i.e., known and unknown), and utilize either self-generated context or retrieved external context to answer them, respectively [141], following a generate-then-read [136] or retrieve-then-read [137] paradigm. However, this binary classification is sub-optimal and inefficient for handling *compositional questions*, which consist of multiple sub-questions where each sub-question could be known or unknown, as illustrated in Figure 4.3. Consequently, these binary-classification methods degrade into simply retrieving information for every question, as any compositional questions containing an unknown sub-question remain entirely unknown by large language models (LLMs).

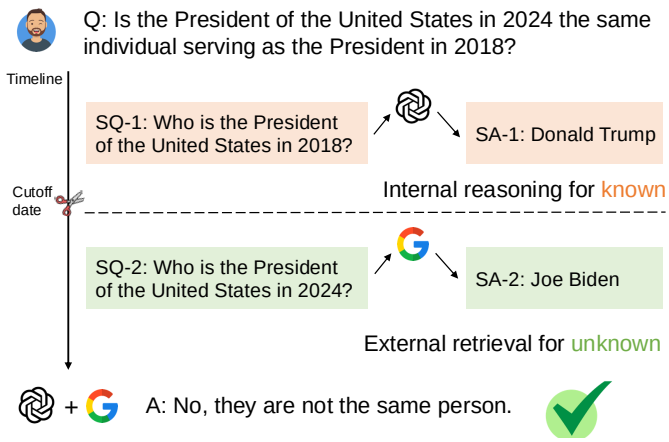


Figure 4.3: *A example of compositional questions, in which a unknown question consists of some sub-questions can be answered using known knowledge while other sub-questions necessitate unknown knowledge according to the cutoff date of LLMs.*

Moreover, using the original compositional question as a query frequently leads to the retrieval of noisy or unrelated documents, which hinders accurate answers [142]. These limitations highlights the need for more nuanced and efficient retrieval strategies tailored to the complexity of *compositional questions*.

In this work, we first formally introduce *compositional questions* from the perspective of known/unknown, which is more practical and challenging. To further specify the *compositional questions*, we categorized questions into four types according to the knowledge boundaries of LLMs [2]:

- ***Single Known.*** The question contains no sub-questions and can be solved using internal knowledge of LLMs, such as with the generate-then-read method.
- ***Single Unknown.*** The question contains no sub-questions and can only be solved using external knowledge, such as with the retrieve-then-read method.
- ***Compositional Known.*** The question contains several sub-questions, and each sub-question is *Single Known*.

[2]The definition begins from the data side instead of model side such as the cutoff date of training data, we discuss hallucination issue of model side at Sec **??**.

- ***Compositional Unknown.*** The question contains several sub-questions, and at least one sub-question is *Single Unknown.*

Determining whether a question is known or unknown to LLMs, and whether it is a compositional question, is a complex task that may require multi-step reasoning. We introduce a **Self D**ivide-and-**C**onquer (`Self-DC`), designed to effectively and efficiently identify and decompose compositional questions. The main idea of `Self-DC` is to use the inherent signals of LLM to control its own behavior, e.g., elicit the internal knowledge or call external retrieval. Specifically, we define each action as a function, and model the whole decomposition as dynamic function calls guided by self-aware confidence signals. Therefore, the internal reasoning capabilities of LLMs can be well elicited while making every external retrieval call count. In summary, our contributions can be outlined as follows:

- To the best of our knowledge, we are the first to study *compositional questions* from the perspective of known / unknown.
- We introduce an automatic data collection pipeline to create the first **C**ompositional **u**nknown **Q**uestion **A**nswering dataset (CuQA), serving as an important evaluation benchmark for LLMs in known/unknown.
- We present a flexible and robust *`Self-DC`* framework, which is capable of adaptively calling different functions on-demand for *compositional questions* decomposition.
- Experimental results on CuQA and FreshQA [139] datasets show the superiority of `Self-DC` in terms of both effectiveness and efficiency, revealing its promising potential to solve compositional reasoning problem.

4.4.2 Framework: Self Divide-and-Conquer

Since LLMs express certainty in different ways and are prone to hallucination issues, therefore, we define α as a mean of confidence score distribution for specific LLM, along with β as the corresponding standard deviation. In this way, the LLMs can recognize when

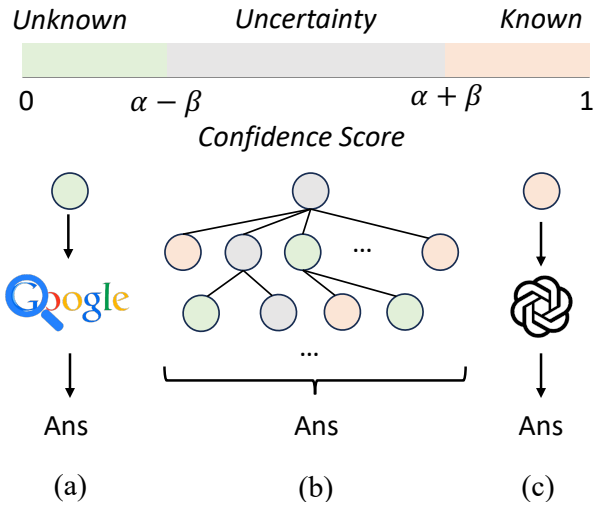


Figure 4.4: *Overview of* `Self-DC`*: a) retrieve-then-read for unknown questions, b) decompose-and-combination for uncertain questions; and c) generate-then-read for known questions.*

a question might be too complex or ambiguous for a straightforward answer, necessitating the decomposition into simpler parts or the combination of multiple pieces of information. Specifically, we divide the confidence score into three ranges $[0, \alpha - \beta]$, $(\alpha - \beta, \alpha + \beta)$, $[\alpha + \beta, 1]$. When the confidence score falls into extreme ranges, such as the left ($[0, \alpha - \beta]$) or right ($[\alpha + \beta, 1]$) side, we can directly apply retrieve-then-read or generate-then-read to answer the question respectively. However, when it encounters uncertain or confusing questions (i.e., fall into the middle part), we decompose the question into several sub-questions to decrease the uncertainty. We then iteratively solve these sub-questions in the same way and combine all sub-answers to answer the original *compositional question* as shown in Figure 4.5. To ensure efficiency and reduce unnecessary costs, we implement several pruning conditions to prevent iterations from overflowing: 1) the number of sub-questions is 1, which means it should be a *Single Known* or *Single Unknown* question; and 2) the number of iteration depth is less than a pre-defined τ. Once these situations happen, we simply regard the current sub-question as the unknown question and then call retrieve-then-read. In this way, we can call compositional reasoning when necessary instead of treating all questions indiscriminately for different LLMs.

Confidence Score Acquisition Inspired by lots of previous works [120, 121], we use two types of method to prompt the LLM itself to get the confidence score to answer the question.

- ***verbalize-based*** *(verb).* We instruct the LLMs to output the confidence level from 0 to 100 following the answer to the question [121]. We clearly note that the confidence level indicates the degree of certainty. Then we re-map the confidence score to the range $[0, 1]$. The details of the prompt can be found in Appendix.

- ***probability-based*** *(prob).* We additionally utilize the probability information to calculate the confidence score. Specifically, we firstly prompt the LLMs to generate the answer using a few words, and then we get the probability $\hat{p}_i$ of i-th token in the generated content. We take the average of probabilities in the sequence as the confidence score [143] following Eq. 4.1:

$$conf = \frac{1}{N}\sum_{i=1}^{N}\hat{p}_i \tag{4.1}$$

Considering the poor performance of LLMs to express uncertainty as reported by lots of existing works [120, 121] and complex situations in practice, we additionally introduce α and β to control the range of uncertainty, enhancing the flexibility and robustness of `Self-DC`.

Other Sub-Functions According to different levels of confidence scores, we carefully design several functions to complete the compositional reasoning task, aiming to provide a more accurate answer. We present the details of other sub-functions one by one as follows:

- **Generate-then-read:** Following Yu et al. [136], we firstly prompt the LLM to generate a background document from Wikipedia to answer the given question, and then ask the LLM to answer the question by referring to the generated passage. The prompt details can be found in the original paper.

- **Retrieve-then-read:** We utilize the retriever to retrieve external knowledge at the first step and then ask the LLM to answer the question by referring to the retrieved passage.

```python
def SelfDC(m, r, q, alpha, beta):
    # m: large language model
    # r: retriever for searching documents
    # q: question to be answered
    # alpha, beta: hyperparameters for defining ranges

    c = get_confidence_score(m, q)

    if c < alpha + beta and c > alpha - beta:
        sub_qs = decompose(m, q)
        sub_as = [SelfDC(m, r, sub_q, alpha, beta) for sub_q in sub_qs]
        answer = combine_sub_qas(m, q, sub_qs, sub_as)
    elif c >= alpha + beta:
        answer = generate_then_read(m, q)
    else:
        answer = retrieve_then_read(m, r, q)

    return answer
```

Figure 4.5: *The simplified python implementation details of `Self-DC`, consisting of several functions: 1) decompose; 2) combine-sub-qas; 3) generate-then-read; and 4) retrieve-then-read.*

- **Decompose:** We prompt the LLMs to systematically break down the overarching question into several smaller sub-questions. The answers to these sub-questions collectively contribute to deriving the answer to the original overarching question, similar to Press et al. [128] and Xu et al [144].

- **Combine answers:** After the decomposition, we call the main function to enter the next iteration as shown in Figure 4.5, aiming to get the answer to each sub-question. Subsequently, we combine the answers to all sub-questions to get the answer to the original question.

4.5 Experiments

4.5.1 Set Up

Baselines. To provide a comprehensive evaluation, we compare our method with different prompting methods with or without the involvement of retrieval augmentation: 1) **Direct Prompting** [145]; 2) **Chain-of-thought (CoT) prompting** [20], including zero-shot and few-shot setting; 3) **GenRead** [136] which firstly prompts the LLMs to generate known knowledge and then answer the question; 4) **Retrieve-then-read (RR)** which retrieves the

related passages first and then answers the questions, following Yu et al. [130]; 5) **Self-Ask** [128] involves generating follow-up questions, retrieving information based on those questions, and providing answers, until no more follow-up questions are generated and the LLMs answer the original question at the last; 6) **IRCoT** [129] interleaves retrieval with steps (sentences) in a CoT, guiding the retrieval with CoT and in turn using retrieved results to improve CoT; 7) **REFEED** [130] and 8) **ITER-RETGEN** [131] utilize the generated answer or intermediate reasoning results to enrich the query, leading to better retrieval and final answer to original question, respectively.

Datasets and Evaluation Metrics. We conduct our experiments mainly on two datasets: 1) the newly proposed CuQA dataset; and 2) FreshQA [139], which contains 600 question-answer pairs that require fast-changing world knowledge, including the latest ones [3]. We note here that FreshQA is not a typical compositional QA dataset despite it containing few *compositional questions*. To select suitable values for α and β, we randomly sample 50 instances as a development set for CuQA, leaving 500 instances for testing. For FreshQA, we use the original split: 500 test instances and 100 development instances. Following previous works [129, 130, 136], we select Exact Match (EM)[4], F1 to evaluate the performance of different methods. Furthermore, to enhance the robustness of the evaluation, we use Acc^{+} as an additional metric and prompt LLMs to assess the predictions related to the actual ground-truth answers following Shao et al. [131].

Implementation Details We mainly conduct our experiments on two different backbone models: `gpt-3.5-turbo-1106` and `gpt-4o-mini`, hereinafter referred to as `1106` and `4o-mini` respectively, following lots of previous works [130, 131, 136]. For the Acc^{+} evaluation, we always use `4o-mini` as evaluation backbone model. We set both the temperature and top p as 0.1 to reduce the randomness of LLMs for all methods, rendering

[3]We use the version on 30th Sep, 2024.

[4]We consider it is matched when the predicted answer in the ground truth answer due to various outputs by LLMs.

Methods	#R	CuQA			FreshQA		
		EM	F1	Acc$^+$	EM	F1	Acc$^+$
w/o retrieval							
Direct	0	21.0	19.3	34.2	20.6	21.6	37.6
CoT	0	21.8	20.5	36.6	21.2	22.9	38.8
Few-shot-CoT*	0	7.2	1.7	9.6	18.0	11.1	26.8
GenRead	0	12.2	12.6	23.2	18.8	19.3	36.0
w/ retrieval							
RR	n	30.4	**24.7**	48.2	34.2	**28.9**	61.6
REFEED	$2n$	35.2	8.2	**53.2**	29.6	16.1	49.2
IRCoT	$3n$	**39.0**	8.1	50.4	32.0	15.5	61.2
Self-Ask*	0-n	8.6	4.3	11.2	16.8	13.4	27.4
ITER-RETGEN*	$2n$	19.2	5.8	25.4	32.4	15.7	46.6
Self-DC (*verb*)	0-$2n$	31.8	20.4	49.4	34.3	25.2	58.1
Self-DC (*prob*)	0-n	32.6	21.7	50.6	**36.2**	28.4	**62.2**

Table 4.1: *The performance of baselines and* `Self-DC` *with the* `1106`*. The baseline* means it uses demonstrations and The column* ***#R*** *denotes the number of retrieval calls in terms of number of test cases n. We* ***bold*** *the best performance and underline the second-best performance.*

Methods	#R	CuQA			FreshQA		
		EM	F1	Acc$^+$	EM	F1	Acc$^+$
w/o retrieval							
Direct	0	29.0	19.4	46.4	27.2	17.3	53.0
CoT	0	28.8	18.2	46.0	29.2	18.1	53.8
Few-shot-CoT*	0	43.0	3.2	50.8	35.0	9.1	55.4
GenRead	0	29.6	29.2	47.4	26.8	27.7	52.0
w/ retrieval							
RR	n	32.0	31.6	55.4	35.2	32.6	63.4
REFEED	$2n$	26.2	33.5	51.8	28.8	34.5	57.4
IRCoT	$3n$	**47.8**	13.5	**64.6**	34.2	17.8	61.4
Self-Ask*	0-n	19.8	3.8	48.4	5.6	9.8	59.0
ITER-RETGEN*	$2n$	23.4	12.6	50.9	31.2	21.1	55.8
Self-DC (*verb*)	0-n	34.0	32.2	53.8	30.2	30.2	59.8
Self-DC (*prob*)	0-n	36.4	**36.5**	56.4	**37.4**	**36.6**	**66.4**

Table 4.2: *The performance of baselines and* `Self-DC` *with the* `4o-mini`*.*

a more fair comparison. We implement the Google search engine following `LangChain` [5] as an external retriever, and we set the number of retrieved results as 3 and the max iteration depth τ as 3. According to the preliminary results on the validation set, we fix β as 0.1 and α as 0.9 for verb (0.8 for prob) on `1106` for both datasets, and α as 0.6 for verb (0.6 for prob on CuQA; 0.8 for prob on FreshQA) on `4o-mini`. The significant test (t-test) is conducted with $p < 0.05$ to ensure statistical improvement.

[5] https://python.langchain.com/docs/integrations/tools/google_search

4.5.2 Main Results

Table 4.1 and Table 4.2 show the performances of all baselines and our proposed `Self-DC` on the `1106` and `4o-mini` respectively. Therefore, several conclusions can be drawn from the results:

CoT (or Few-shot-CoT) does not bring consistent improvements over direct prompting (Direct). We surprisingly found that the performance of CoT at both Table 4.1 and Table 4.2 is usually worse than Direct, and Few-shot-CoT can not further boost the performance particularly with `1106`, revealing the complexity of compositional reasoning.

Retrieval-based method generally achieves better performance than non-retrieval methods but the gap is smaller with compositional questions. It is observed that RR and IRCoT are capable of achieving better performance than non-retrieval baselines, and IRCoT sometimes achieves the highest performance due to a more complex retrieval design, accompanied by more cost. Secondly, the gap between retrieval-based and non-retrieval-based methods on FreshQA is relatively larger than on CuQA. This discrepancy is likely because CuQA contains more *compositional questions*, which, when used directly as queries, result in noisier documents. Furthermore, we surprisingly observe that Self-Ask and ITER-RETGEN achieve the lowest performance, especially on CuQA. To understand the reason, we examined the intermediate reasoning results and found that Self-Ask tends not to generate follow-up questions and directly answer the question, rarely calling for retrieval given the compositional unknown question. On the other hand, ITER-RETGEN retrieves external documents step-by-step but introduces a lot of noise since the queries are mostly related to the original *compositional question*. These observations reveal the significance and valuable insights provided by the CuQA dataset, highlighting its importance for understanding the challenges associated with *compositional questions*.

***`Self-DC`* achieves better trade-off between efficiency and effectiveness than retrieval-based methods.** When comparing `Self-DC` to other baselines considering the consump-

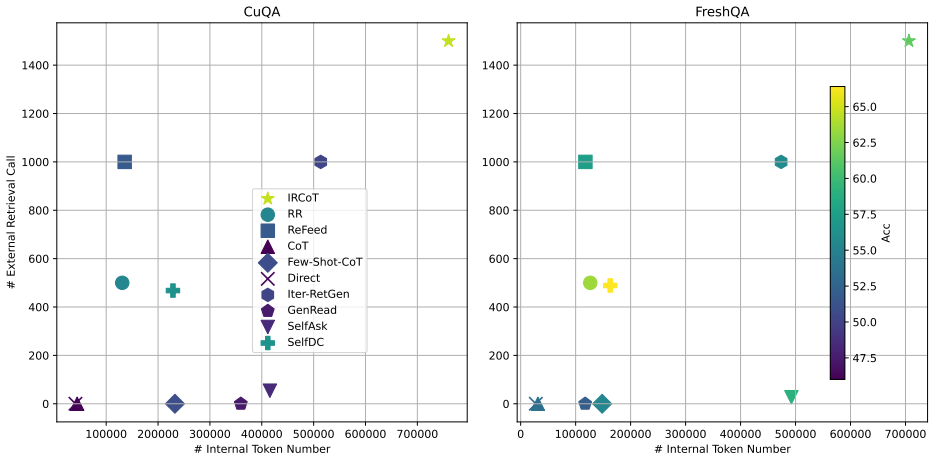


Figure 4.6: *The efficiency analysis of different methods using* `4o-mini`.

tion of retrieval calls (#R), it is evident that `Self-DC` achieves better performance compares with the method utilizing same or more calls, for example, `Self-DC` (prob) v.s. RR. Even compared with some methods that require 2 to 3 times more retrieval, `Self-DC` still achieves comparable results and even outperforms them in specific dataset. This is important to highlight, as it not only establishes an effective and efficient framework to call external retrieval, but also demonstrates a promising path for controlling the behavior of LLMs by leveraging the internal signals they generate (i.e., the internal confidence scores).

4.5.3 Discussion and Analysis

Efficiency Analysis To directly validate the efficiency of `Self-DC`, we consider three dimensions: # internal token consumption, # external retrieval calls and the final performance. Table 4.6 illustrate the report. Ideally, we aim for a method which achieves the best performance appears at the left bottom of figure. Only in such a case, the method would demonstrate its superiority by not only delivering better performance but, more importantly, by eliciting the great potential of the internal capabilities of LLMs and minimizing reliance on external resources or tools. According to the figure, it is obvious that `Self-DC` achieves great balance between these three factors. It is worthy noting we observe similar trends on `1106` for both datasets.

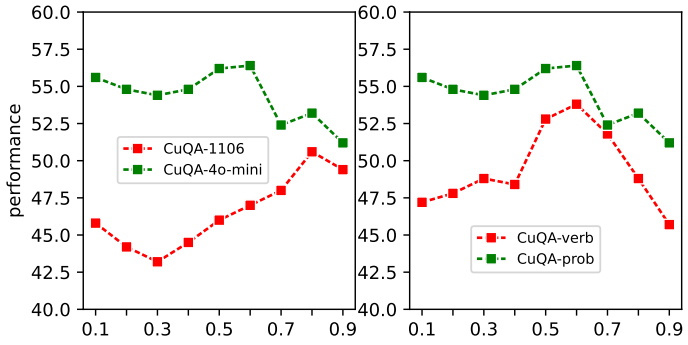


Figure 4.7: *The performance of different choices of α with β = 0.1.* ***Left:*** *The performance of different models with confidence type is prob; and* ***Right:*** *The performance of different confidence types (verb or prob) with the same model* `4o-mini`.

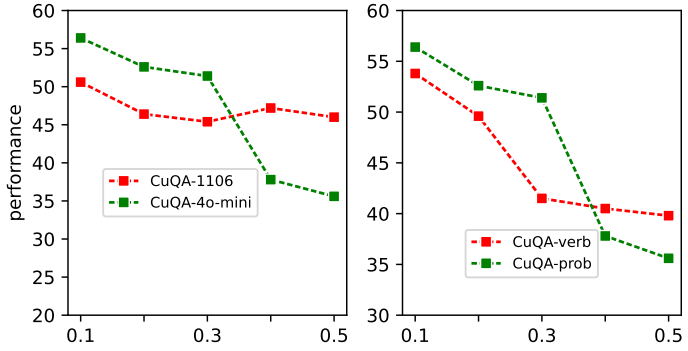


Figure 4.8: *The performance of different choices of β with a fixed α as 0.8 for* `1106` *and 0.6 for* `4o-mini`. ***Left:*** *The performance of different models with confidence type is prob; and* ***Right:*** *The performance of different confidence types (verb or prob) with the same model* `4o-mini`.

The Impacts of Different α and β. It is vital to balance alpha and beta for optimizing the performance of LLMs to different tasks. In this section, we provide detailed analysis of different choices of α and β. Firstly, we fix $\beta = 0.1$ and set α to $[0.1, 0.2, 0.3, ..., 0.9]$. The results can be found in Figure 4.7. The entire processing can be seen as a 0.2-length uncertainty block starts from 0 to 1 with stride = 0.1. First of all, We found that none of the lines shows monotonically increasing or decreasing, and most of the best performances are achieved in the middle choice of α, revealing the complexity of the target problem. In detail, there is an upward and then downward trend globally (e.g., in the right figure). It is reasonable since LLMs utilize more generate-then-read functions at the beginning (e.g.,

α=0.1, β=0.1), resulting in poor performance. With the uncertainty, blocks move to the right side (a.k.a, 1), LLMs will utilize retrieve-then-read more frequently. Once exceeds a specific threshold, the performance will drop since the decomposition will introduce more noise compared with gains.

4.6 Monitoring and Control of Language Agents

Despite significant progress in enabling language agents to interact with external tools via prompting engineering, their behavior often remains unstable and suboptimal. To address this, we further explore better monitor and control methods.

4.6.1 Monitor: Assessment of Knowledge Boundary

In order to assess the knowledge boundary for each model, existing methods typical collect *model-specific* supervised fine-tuning dataset to teach the model to say “I do not know" for the unknown questions and only provide answer for known questions [146, 147, 148]. In detail, it is usually required to collect lots of known and unknown question answer pairs, and feed them into LLMs multiple times, and observe how many times the LLMs can provide correct answer for these questions, and therefore using the frequency of correct answer as confidence score for each model and question [147, 148]. However, this line of work always require to re-construct the dataset for different model, since different model has different knowledge boundaries as illustrated in $ 4.2.3. One potential coarse-grained solution is to leverage the concepts of minimal and maximal knowledge boundaries to construct a universal dataset applicable across different LLMs. Specifically, we can curate a collection of problems that naturally fall into two categories: those that are consistently known by *all* LLMs (reflecting the minimal knowledge boundary), and those that are consistently unknown to *all* LLMs (reflecting the maximal knowledge boundary). Besides that, there are several studies exploring the fine-grained monitoring during the reasoning processing by leveraging the internal confidence signals [149, 150].

4.6.2 Control: Self-Aware Agent for Better Tool-use Behavior

But do we truly need explicit monitoring signals to guide an agent's control behaviors? More specifically, are fine-grained monitoring signals necessary? These questions remain largely underexplored. It has been observed that simply leveraging outcome-based reward signals can still effectively enhance the reasoning performance of LLMs [26]. We further discuss several new methods to improve the tool-use behavior of LLMs.

Supervised Fine-tuning. Supervised fine-tuning (SFT) remains the most common approach for teaching agents tool use, using small task-specific datasets (e.g., math, code) to demonstrate when and how to call external tools [102, 14]. However, most of previous methods often assumes a uniform knowledge boundary across models, which is unrealistic. As discussed in Lemma 2.1, this mismatch leads to inefficiencies: what is helpful for a small model may be redundant or even distracting for larger ones. One solution is to create custom SFT datasets tailored to each model's knowledge boundary, but this is resource-intensive and hard to scale. A more practical alternative, as outlined in Lemma 2.2, is to approximate a maximal knowledge boundary and train agents to defer intelligently when faced with unfamiliar content. Therefore, we propose SMART [103] - Strategic Model-Aware Reasoning with Tools, which assume some knowledge naturally fall outside of maximal knowledge boundary and thus external physical tools are only invoked to provide supplementary information in such cases. While this approach offers greater generality, it may lack the precision needed for fine-grained domains, highlighting a trade-off between scalability and behavioral fidelity.

Reinforcement Learning. Reinforcement learning (RL) offers a more promising path for aligning a model's decision-making with its own knowledge boundary, as agents can learn from experience how to adaptively use tools. The key challenge lies in designing reward functions that go beyond correctness. While many RL agents are trained to maximize answer accuracy, this ignores how the answer is reached, including whether reasoning is efficient,

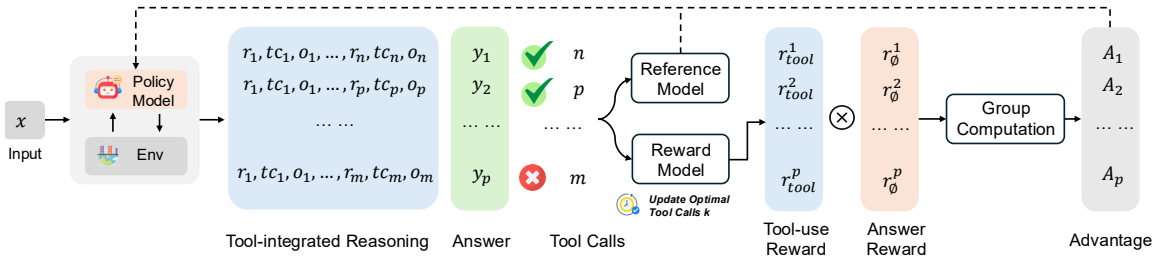


Figure 4.9: *An overview of OTC-GRPO Algorithm.*

whether tool use is justified, and whether the trajectory is optimal [27, 107]. Our proposed work OTC-PO [111] addresses this by balancing correctness with penalties for unnecessary tool calls, encouraging agents to act with restraint and self-awareness. By optimizing not only for outcomes but for processes, RL can produce agents that are not only accurate but also efficient, interpretable, and better aligned with real-world deployment constraints.

4.6.3 Management: Conflicts between Internal and External Knowledge

There is a case that when internal cognitive tools and external physical tools return conflict knowledge. One typical example is the hallucination issue, that LLMs may make up misleading or fabricated information that is not grounded in factual knowledge (i.e., internal cognitive tools return wrong knowledge while external physical tools return correct knowledge). To address this issue, we need to first identify knowledge conflict (i.e., Knowledge Conflict Detection), and then decide which source, internal or external, should be trusted (i.e., Knowledge Arbitration), and enforce the LLMs can generate the response based on selected trusted source of knowledge (i.e., Knowledge-grounded Generation). We present a comprehensive survey of different types of knowledge conflict [151]. Through analysis of the internal activations of LLMs, we find that these models are capable of internally registering signals of knowledge conflict, particularly in the mid-layer representations. These signals can be used to detect the presence of conflict and enable inference-time intervention strategies to resolve it. Building on this observation, we propose SpARE, a training-free representation engineering method that leverages pre-trained sparse auto-encoders (SAEs)

to steer the knowledge selection behavior of LLMs [152].

4.7 Discussion

In this section, we would like to discuss the ultimate goal of agent and a potential path to achieve such goal.

4.7.1 New Agent Objective

The truly autonomous agent should learn the compression of world in its internal parametric space, and then learn to complete the pre-defined goal with minimized external actions [2]. In this way, the goal can be defined as follows:

$$\arg\min_{\tau} \ \text{Cost}(\tau) \quad \text{subject to} \quad \mathcal{M}(q, \tau) = \hat{a}, \tag{4.2}$$

Here the cost is measured as the financial and computational cost for the whole interactions, such as the cost of external tool calls within the trajectory τ. It contains interactions which utilize both internal cognitive tools and external physical tools. Thus the model is encouraged to not only generate correct answer or complete the goal but also minimize the cost during the processing. To clarify specific tool-use behavior for the optimal agent, let's considering four different cases in terms of both internal cognitive tools and external physical tools:

- **Maximizing both internal and external physical tool use.** In this case, the agent produces the correct answer but does so through excessive use of both internal and external tools, regardless of necessity. This behavior is inefficient, consumes unnecessary resources, and increases the risk of error propagation or tool misuse. It also obscures the decision-making process, reducing transparency and trust. Rather than reflecting strategic reasoning, this mirrors brute-force search, which is misaligned with the goals of scalable and interpretable AI systems.

- **Maximizing external tool and minimizing internal tool use.** This entails the agent over-relies on external tools while underutilizing its internal reasoning capacity. This may yield correct results, especially for smaller models, but it results in inefficiency and increased dependence on external systems. More importantly, it conflicts with the core aim of model scaling: to internalize knowledge within parameters. By deferring to external tools, the agent misses opportunities to reinforce and generalize its own representations, limiting long-term autonomy and adaptability.

- **Maximizing internal tools and minimizing external tool use.** In this setup, the agent leans heavily on internal reasoning and avoids external tool use [111]. This behavior promotes autonomy and efficiency, especially in constrained environments, and aligns with the principle of maximizing model capacity. However, excessive internal deliberation can lead to overthinking, producing unnecessarily long reasoning chains. While this reflects strong use of internal knowledge, it may overlook more efficient external solutions in certain cases, indicating a need for better tool use calibration.

- **Minimizing both internal and external physical tool use.** This represents the most efficient trajectory: solving tasks with minimal use of tools, internal [19] or external [111]. It reflects optimal behavior of using tools only when necessary, guided by precise self-monitoring and calibrated decision-making. However, extreme minimalism can risk underthinking or skipping essential steps, especially in complex tasks. In addition, empirically training agents toward this behavior is difficult, as it requires balancing correctness with efficiency, which is a more delicate optimization than correctness alone.

4.7.2 Paths for Agent Foundation Model

Following the established roadmap for foundation models, we propose an analogous trajectory for agent foundation models, where the core training objective shifts from *next-token prediction* to *next-tool prediction*.

Agentic Pre-training. Since we unify all interactions under the same tool framework, it becomes feasible to build a unified data schema capable of representing a wide range of interactive tasks—much like how next-token prediction underpins most natural language processing tasks. However, a central challenge lies in collecting sufficient pretraining trajectories for such interactions, given their inherent scarcity and complexity in practice, particularly in multi-lingual, multi-modal, and multi-task settings. This transforms interaction itself into a first-class modeling target, allowing the agent to learn how to gather information it doesn't already possess. As research trends toward unified agent architectures, modeling all forms of interaction (API calls, UI navigation, or environment manipulation) as structured, learnable outputs opens the door to a new kind of scaling law: one that governs knowledge acquisition, not just compression. This shift is essential for building adaptive, self-improving agents in open-ended, dynamic environments.

Agentic Supervised Fine-tuning. As we discussed above, there are two different ways to fine-tune: 1) general dataset to fit all models; 2) model-specific dataset tailored to its own knowledge boundary. We argue that both approaches play important roles in refining an agent's decision boundary. The general dataset can be incorporated into the pre-training stage to ensure broad generalization, while the model-specific dataset becomes increasingly valuable when there are higher demands for effectiveness and efficiency in reasoning and tool use.

Agentic RL. Actually, it is very hard to collect model-specific SFT to approximate the accurate knowledge boundary in practice, since there are multiple valid trajectories to reach the final goal. Therefore, it makes RL a more promising path for aligning a model's decision-making with its own knowledge boundary, as agents can learn from experience how to adaptively use tools. On the internal side, there are many studies trying to minimize the reasoning token cost to reach the final correct answer [153, 154]. On the external side, it is believed that our OTC-PO [111] can serve as the foundation for this direction. It is observed there is a unified reward design, similar with OTC-PO [111], to minimize

both internal cognitive tool calls and external physical tool calls without sacrificing the accuracy a lot. We further envision a cyclical training paradigm - RL $\rightarrow$ SFT $\rightarrow$ RL $\rightarrow$ SFT - where reinforcement learning uncovers high-quality trajectories aligned with the agent's knowledge boundary, and supervised fine-tuning consolidates these behaviors for improved stability and generalization. This iterative process can gradually refine both the agent's policy and decision boundary, moving toward increasingly adaptive and self-aware agentic behavior.

4.8 Summary

In this chapter, we first introduce the concept of meta-reasoning [28] for autonomous agent, and advocate the key of autonomous agent lies in alignment between its decision boundary and knowledge boundary, aiming to minimize external actions in the real world to achieve the pre-defined goal. Then we present three specific principles about knowledge boundary and decision boundary of agents, followed by three different proposed methods - Self-DC [117], SMART [103], OTC-PO [111].

Furthermore, we discuss the different behaviors of agents to achieve the pre-defined goal, identify the optimal agent behavior should minimize the interactions (i.e., tool calls) to reach the final goal. We finally offer a roadmap for developing agent foundation model that can operate effectively in open-ended environments, learn efficiently with minimal supervision, and generalize across domains.

Part II

Benchmarks

Chapter 5

AppBench: Benchmarking Tool Planning in Physical World

5.1 Introduction

Empowering Large Language Models (LLMs) [155] with versatile tools such as retrievers [133, 134], models [86], and even physical robots [156], holds significant promise in overcoming inherent limitations, such as hallucination [157] and outdated information [87, 158], and unveils the immense potential for LLMs to tackle increasingly complex and interactive real-world tasks [22, 159]. Over the past several months, lots of new benchmarks and datasets have been proposed to evaluate the performance of different LLMs to adeptly select and execute various tools [86, 159, 160], marking a pivotal milestone in their evolution. Out of plentiful tools in practice, APIs have become one of the fundamental and promising tools in today's digital world, due to greater flexibility and customizability with well-defined format and ease of execution [75].

Previous works have attempted to evaluate LLMs on their ability to call the correct API in multiple turn dialogues, such as API-Bank [159] and ToolBench [161], or single turn instructions, like APIBench [162]. However, most existing benchmarks focus either on a single API call in a single turn or on APIs with limited arguments. For instance,

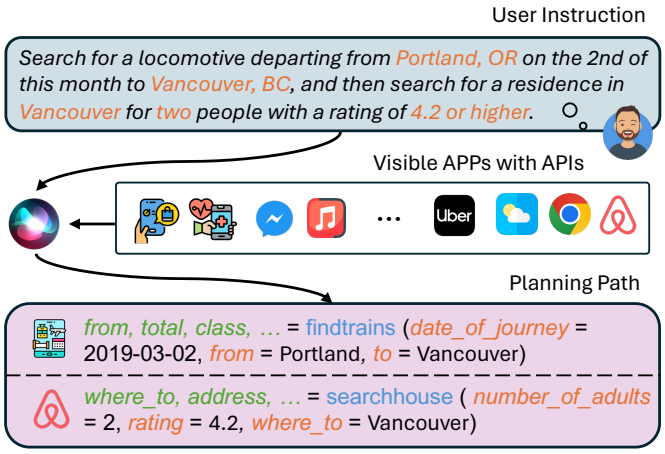


Figure 5.1: *An example of one user instruction requires two independent APIs from different APPs since input arguments of both two APIs do not rely on each other. We use different icons to indicate different APPs, and color API, and returned arguments and input arguments.*

API-Bank mainly evaluate one API call per turn in multi-turn dialogues, while APIBench and ToolBench considers APIs only with one or two arguments (e.g., only one output with one or two inputs). Furthermore, the small number of arguments makes it difficult to fully explore the complex dependency relationships between multiple APIs. For instance, the input arguments for a current API may depend on the return arguments of several previous APIs. These limitations highlight a gap in addressing complex user instructions when it is necessary to utilize multiple APIs in practice, underscoring the need for more comprehensive and practical evaluation benchmarks.

To bridge the gap, we introduce a new evaluation benchmark: `AppBench`, representing the first effort to assess the aptitude of LLMs to function as the meta planner for multiple APIs from various sources for complex user instruction. Specifically, we simulate a situation in which the user instruction can be fulfilled through collaboratively API calls from various APPs in the mobile device. Figure 5.1 shows one typical example. Given the complex user instruction, the meta LLM, such as Apple's Siri and Google Assistant, need to plan an executable path according to user instruction and corresponding API descriptions. To fulfill this requirement, it is necessary not only to indicate which APP will distribute and

execute each API but also to specify the execution order of the APIs, including all necessary inputs and returned arguments. We consider this setting aligns well with the complexity and practical limitations in the real world, and presents a great opportunity for advanced AI assistants like Apple's Siri to showcase their intelligence and capability in orchestrating collaborative API executions across multiple Apps.

In this way, two significant challenges are identified: *graph structure* and *permission isolation*. Firstly, the inter-dependency between multiple APIs creates a more complex execution structure. Some APIs can be executed independently, while others are dependent and must be executed sequentially, resulting in a graph-like structure. Secondly, these APIs may originate from different sources, and the LLM might not have permission to call them directly. This necessitates identifying the authorized source for each API. For instance, APIs from one company may only be executed by an LLM within the same company. In doing so, we aim to chart a path towards realizing the vision of an intelligent assistant capable of seamlessly navigating and interfacing with the myriad APPs and APIs pervasive in contemporary digital ecosystems. To conclude, our contribution can be summarized in three folds:

- To the best of our knowledge, we are the first to identify graph structure and permission isolation issues of multiple API calls when addressing complex user instructions.
- We propose `AppBench`, serving as an important complementary evaluation benchmark to assess the planning capabilities of different LLMs as meta planner for these APIs. Additionally, we introduce an automatic data collection pipeline, which can be used to gather data efficiently and effectively.
- Our experimental results on 9 distinct LLMs demonstrate almost all models, including the latest GPT-4o, fall short in this setting, particularly when dealing with complex graph planning structures. Further analysis shows that simple in-context learning and fine-tuning do not significantly improve performance.

5.2 Related Work

Tool Benchmarks. The complexity of real-world tasks necessitates the integration of diverse tools and services, consisting of three types of tools [75]: 1) physical interaction-based tools [156]; 2) GUI-based tools [163]; and 3) program-based tools [133, 159]. On the one hand, some work focuses on models, retrievers, or calculators to address the intrinsic limitations of LLMs, such as ToolQA [138] and ToolBench [161]. On the other hand, another line of work targets APIs since they are particularly crucial for bridging smooth interaction between humans and the digital realm [159, 160, 161]. Most previous works formulate this as an API selection task given all related information about each API and current input, which overlooks the nuanced dependencies and permission constraints between different APIs, such as APIBench [162] and API-Bank [159]. Nevertheless, the successful execution of APIs in the real world necessitates meeting requirements fulfilled (either the value is provided by the user or previous APIs) and obtaining permission from trusted agents beyond just knowing API names and a few arguments. More details can refer to latest survey [164] and tutorial [76].

Language Agent. Existing frameworks for language agents have made notable strides in facilitating interaction with external tools [86, 159, 160] and environment [165, 166]. They usually follow the single-agent paradigm to access different tools or services sequentially [22, 159], or multi-agent framework by assigning different agents different roles to call different cognitive tools [1] . For example, Lu et al. [22] propose Chameleon which utilizes one agent to plan the execution order of different services by outputs a sequence of names of tools, which assume that the agents to call these tools are already known, and lots of works follow this setting [160, 167]. Furthermore, various benchmarks are proposed to evaluate the abilities of LLMs serving as agents in different situations [159, 168, 169]. For instance, Yao et al. [105] proposes WebShop to evaluate whether LLMs are capable of interacting with the Web. Similarly, Xavier et al. [165] simulates household activities through programs, and many works use this as a testbed for embodied agents [24]. Latest work focus on using

APIs or functions to control the whole planning processing of agents [117].

5.3 AppBench: Planning of Multiple APIs from Various APPs

5.3.1 Task Definition

Given the user instruction u and a virtual mobile environment with an APP family, $\mathcal{E} = \{APP_1, APP_2, ..., APP_n\}$ where each APP contains several APIs $\{p_i^1, ..p_i^j\}$ where i stands for i_{th} APP and j means j_{th} API inside this APP, the meta agent need to decide an executable path to call different APIs from various APPs to fulfill the instruction in the format of the list which each item in the list is $\{APP_i : r_1, r_2, .., r_m = p_i^j(k_1 = v_1, ..., k_n = v_n)\}$. The APP_i and p_i^j denote the name of the APP and corresponding API of this APP, and the r_i and k_i mean the i_{th} returned and input arguments respectively. The v_i can be the actual value provided by the user or a returned argument by previous APIs.

5.3.2 Compositional User Instructions: SS, SM, MS, MM

Based on the number of APPs and APIs utilized in each user instruction, the data can be categorized into four distinct types. Each category represents a typical use case in practical scenarios, creating a comprehensive benchmark for evaluating real-world applications when combined.

- ***Single APP Single API (SS)*** The instructions of the users only need to utilize one API from one APP.
- ***Single APP Multiple API (SM)*** The instructions of the users need to utilize multiple API from one APP. It is important to note that these APIs can be called either sequentially or concurrently, depending on whether there is a dependency between their arguments.
- ***Multiple APPs Single API (MS)*** The instructions of the users need to utilize multiple APIs and each of them belongs to one different APP. Also, there may exist dependency between APIs across different APPs.

Benchmark	SS	SM	MS	MM	DP
APIBench [162]	✓	✗	✗	✗	✗
API-Bank [159]	✓	✗	✗	✗	✗
ToolQA [138]	✓	✗	✗	✗	✗
ToolBench [161]	✓	✓	✓	✓	✗
UltraTool [160]	✓	✓	✗	✗	✓
AppBench (Ours)	✓	✓	✓	✓	✓

Table 5.1: *Comparison with existing evaluation benchmarks at the turn-level for a fair comparison. DP stands for Dependency.*

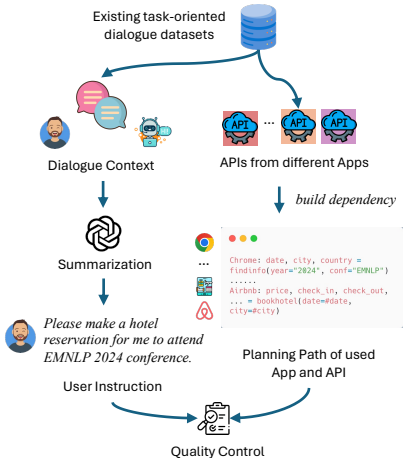


Figure 5.2: *A high-level processing to collect the* `AppBench`*, taking advantages of existing task-oriented dialogue datasets.*

- ***Multiple APPs Multiple API (MM)*** The instructions of the users need to utilize multiple APIs and multiple APPs. The difference with MS is there may exist multiple APIs come from the same APP. Furthermore, the dependency relationship between arguments can be the most complex when dealing with APIs from the same APP or from different APPs.

Table 5.1 shows the detailed comparison between `AppBench` with other popular benchmarks. Most of existing benchmark focus on part of these typical situations or overlook the complex dependency relationships between multiple APIs. In addition, our formulation highlights the potential for investigating *graph structure* and *permission management*, considering the inherent complexity of APIs and Apps, particularly in terms of handling DP in multiple input and output arguments.

5.3.3 Data Collection

To maximize the authenticity of user instructions and minimize human efforts, we prioritize using existing task-oriented dialogue datasets [170, 171]. These datasets are typically collected through human-to-human interactions in real-world scenarios and contain a wide range of APIs across numerous domains and services. Specifically, we selected the SGD [170] dataset as the seed dataset because it encompasses most of the domains and APIs. We then utilized LLMs and Python scripts to generate the desired inputs and outputs, respectively. Figure 5.2 illustrates the detailed procedures.

Instruction Acquisition. Firstly, we extract the utterances of the user and system in the task-oriented dialogue and feed it into the LLM[1] to summarize the user's requirements in one instruction. For example, the user may want to know the city and date of EMNLP 2024, and book a hotel according to the city and date. In the previous task-oriented dialogue, this is achieved by multi-turn interactions. In contrast, we summarize the whole dialogue into one complex user instruction to mimic more natural and complex cases in practice. To ensure that the values of certain intermediate arguments (such as *date* and *city*) are not disclosed at the instruction, we require the LLM to avoid outputting the actual values of other arguments, except for those that are explicitly provided in the prompts, such as user-aware arguments.

Planning Path. Besides the instruction part, we write a Python script to automatically parse the API calls at different system turns in the multi-turn dialogue to form the planning path as the output. Specifically, we regard different domains (a.k.a., services) in task-oriented dialogue as different APPs such as restaurants and hotels, and extract the name of the domain and API first to locate which APP should invoke to call the API, and then we follow the execution order of different APIs to build the dependency between various arguments. For example, if the returned arguments from the previous API are required in the current

[1]GPT-4o during the collection

Statistics	SS	SM	MS	MM
# Samples	200	200	200	200
# Apps	9	11	10	11
# APIs	11	22	12	23
Avg. Apps	1.0	1.0	2.7	2.2
Avg. APIs	1.0	2.2	2.7	3.3
Avg. arguments	4.0	4.5	3.8	4.4
Max. Seq.	1	4	4	8
Max. Para.	1	1	4	3
Avg. Seq.	1	2.2	1.2	1.9
Avg. Para.	1	1.0	2.2	1.8

Table 5.2: *The data statistics of our proposed* `AppBench`.

API, we use #name to indicate it such as #date and #city in the Figure 5.2. In this way, we can get an executable and unique path to execute APIs from different APPs.

App	APIs
Rents	*getcarsavailable, reservecar, getride*
Hotels	*searchhouse, bookhouse*
Services	*book_stylist_appointment, find_stylist_provider, book_therapist_appointment, find_therapist_appointment*
Restaurant	*reserverestaurant, findrestaurants*
Movies	*buymovietickets, findmovies, gettimesformovie, reviewmovies*
Trains	*gettraintickets, findtrains*
Events	*findevents, buyeventtickets*
Travel	*findattractions*
Buses	*findbus, buybusticket*
Flights	*searchonewayflight, searchroundtripflights*
Payment	*requestpayment, makepayment*
Music	*playmedia, lookupmusic*
Weather	*getweather*

Table 5.3: *List of All Apps and their corresponding APIs in the* `AppBench`.

Quality Assessment To ensure the quality of data, we utilize a Python script to validate whether or not all actual values are provided from the user side, and none of values are provided from the system side. Furthermore, we adapt GPT-4o to score each instruction in terms of fluency and diversity from 1 to 10, and then remove cases whose score is lower than 6. Approximately 20% of the samples were removed, and the average score of the remaining samples is around 8.05. We finally manually check each instruction-path pair, and remove some mismatch pairs such as the instruction is simple or API calls can not complete the user instructions, resulting in 200 high-quality samples for each category.

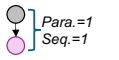
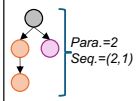
Data Type	Example	Structure
SS	Instruction: *Find a house with a rating of 4.6 or higher for a trip to Delhi for two people, inquire about laundry service availability* Output: House: address, phone_number, total_price, has_laundry_service, … = searchhouse(number_of_adults='2', rating='4.60', where_to='Delhi')	
SM	Instruction: *Please book a Hatchback car with insurance to be picked up from Warsaw Chopin Airport on March 7th at 1:30 pm, and returned on March 13th in Warsaw.* Output: Rents: pickup_location, price_per_day, ... = getcarsavailable(car_type='Hatchback', city='Warsaw', end_date='2019-03-13', pickup_time='13:30', start_date='2019-03-07') Rents: car_type, car_name, ... = reservecar(add_insurance='True', car_type=car_type, end_date=end_date, pickup_location=**#pickup_location**, pickup_time=pickup_time, start_date=start_date)	
MS	Instruction: *Search for a locomotive departing from Portland, OR on the 2nd of this month to Vancouver, BC, and then search for a residence in Vancouver for two people with a rating of 4.2 or higher.* Output: Train: from, total, class, … = findtrains (date_of_journey = 2019-03-02, from = Portland, to = Vancouver) House: address, phone_number, total_price, has_laundry_service, … = searchhouse(number_of_adults='2', rating='4.2', where_to= Vancouver')	
MM	Instruction: *Please make a reservation for 3 people at one Korean restaurant in San Francisco at 1:30 pm on March 12th, and also book a Luxury taxi for 3 to 4 Embarcadero Center.* Output: Restaurant: restaurant_name, has_vegetarian_options, phone_number, rating, address, price_range, category, ... = findrestaurants (category ='Korean', has_seating_outdoors='True', location='San Francisco') Restaurant: date, time, location, ... = reserverestaurant (date='2019-03-12', location=location, number_of_seats='3', restaurant_name = **#restaurant_name**, time='13:30') Rents: destination, ride_type, ride_fare, wait_time, number_of_seats = getride(destination='4 Embarcadero Center', number_of_seats='3', ride_type='Luxury')	

Figure 5.3: *An example of different types of samples in* `AppBench`*. We color APP, API, and returned arguments and input arguments. We also present the structure of the example using grey nodes and colorful nodes to indicate user instruction and APIs from different APPs, respectively. We bold the* ***argument*** *which is returned by the previous API call (a.k.a., dependency relationship). Para. and Seq. represents the parallel and sequential size of the corresponding data sample. We emphasize we only choose the simplest examples in each type for better understanding, there are data samples with much more complex logic structures in the original dataset.*

5.3.4 Data Statistic

Table 5.2 illustrates the statistics of `AppBench`. Specifically, there are approximately 10 different APPs for each type and over 20 various APIs in both MS and MM. We provide the list of all APP and API in Table 5.3. Secondly, the average number of APIs increases from SS, SM to MS, MM, revealing the complex relationship. We also emphasize that the higher number of arguments for each API aligns with the complicated nature of tool execution in practice, as there may be multiple input and returned arguments for one API. Furthermore, we provide statistics about sequential and parallel relationships in each category (Seq. and Para.), revealing the complex graph structure in the dataset. Figure 5.3 presents one example for each category for better understanding.

Models	SS			SM			MS			MM		
	$F1_{app}$	$F1_{api}$	Succ	$F1_{app}$	$F1_{api}$	Succ	$F1_{app}$	$F1_{api}$	Succ	$F1_{app}$	$F1_{api}$	Succ
Mistral-7B	55.97	16.31	0.51	36.59	15.09	0.50	33.72	6.42	0.00	28.92	7.56	0.00
Vicuna-13B	43.20	3.70	2.00	34.71	4.63	0.50	20.43	3.10	0.00	21.05	2.52	0.00
LLaMA3-8B	63.04	42.67	23.23	37.20	25.33	0.50	30.65	19.52	0.10	26.39	17.80	0.05
LLaMA3-70B	71.20	70.00	50.00	46.48	46.96	10.50	32.61	32.96	2.50	28.97	28.53	0.50
QWen1.5-7B	48.14	19.54	0.00	30.13	16.71	0.00	23.24	10.11	0.00	23.76	11.55	0.00
QWen1.5-14B	72.89	28.41	10.10	41.89	25.51	1.50	42.22	21.98	0.80	32.36	15.07	0.00
QWen1.5-72B	81.23	24.28	12.50	**51.89**	25.27	1.00	**45.94**	13.42	0.62	**38.53**	11.51	0.00
GPT-3.5	63.60	57.95	30.81	41.49	43.65	6.50	33.17	34.53	7.00	27.79	28.09	1.00
GPT-4o	**88.31**	**86.87**	**70.92**	50.83	**50.57**	**20.50**	39.39	**39.14**	**11.00**	32.62	**32.35**	**2.00**

Table 5.4: *The main results of different LLMs on* `AppBench`*. Bold highlights the best score among all models, and underline underscores the best score under the same model scale*

5.4 Experiments

5.4.1 Setup

Models. We choose several LLMs from both open- and closed-source models, aiming to provide a comprehensive evaluation, following [138, 160]. Specifically, we choose Mistral-7B (`Mistral-7B-v0.2`) [172], the LLaMa3 series [173] (`Meta-Llama-3-8B/70B-Instruct`), and the Qwen series [174] (`Qwen1.5-7B/14B/72B-Chat`) from open-source LLMs. Besides that, we also select GPT3.5 (`gpt-3.5-turbo`) and GPT4 (`gpt-4o`) from closed-source LLMs. We also tried other models such as LLaMA2-7B or Vicuna but we find it difficult for them to output in the required format.

Implementation Details. We set the temperature and top p as 0.1 to reduce randomness. The experiments of open-source models are run on NVIDIA A100 GPUs and those of closed-source models are fulfilled by APIs of OpenAI. To address the limitations imposed by the varying context windows of different LLMs, we adopt a *hierarchical* prompting approach. First, we prompt the LLMs to identify the relevant APP. Once the appropriate APP is determined, we then provide the LLMs with only the API descriptions of these specific APPs.

5.4.2 Evaluation Metrics

In order to evaluate the LLMs' capabilities of selecting proper APPs, choosing APIs, and fulfilling all arguments to execute the API based on the users' instruction, we carefully design two F1 scores for APP and API, and one overall success rate considering the complexity of the task.

F1 of App. We first get the precision P_{app} as the number of correctly predicted APPs divided by the total number of APPs predicted by the model:

$$P_{app} = \frac{app\_hit\_num}{app\_pred\_num} \tag{5.1}$$

and recall R_{app} as the number of correctly predicted APPs divided by the total number of APPs that are in the ground truth as follows.

$$R_{app} = \frac{app\_hit\_num}{app\_ground\_truth\_num} \tag{5.2}$$

The F1 of App score is 2PR / (P+R), as usual.

F1 of API. Similarly, the metrics of API predictions can be evaluated using $F1_{api}$. Note that we only consider the name of the API here to determine LLM whether or not to choose the right API, and the performance of arguments of APIs is evaluated in the next metric.

Success Rate (Succ): This metric evaluates whether the LLMs can fully execute the user's instruction by correctly identifying all required APPs, APIs, and arguments. It is defined as the proportion of instances where all elements—APP, API, and arguments—are in perfect alignment with the ground truth, considering the complex dependency relationship between different APIs across APPs, resulting in a direct measure of model capability in full instruction fulfillment. Since there may exist different output orders, we calculate this at the structure level since the execution structure is unique.

5.4.3 Main Results

Table 5.4 shows the results of different LLMs for different types of user instructions on `AppBench`, respectively. Several conclusions can be drawn from the results.

Overall, GPT-4o achieves the best overall performance, while LLaMA3-70B sometimes outperforms GPT-3.5, mostly in scenarios only involving single APP. In general, other models significantly lag behind GPT-4o in all types of instructions, and only QWen1.5-72B or LLaMA3-70B achieves better or competitive performance compared with GPT-4o. Despite significant advancements in LLMs, the existing models still fall short in addressing the complexities of planning cases such as multiple APPs and multiple APIs. One fact is that all LLMs only get less than 3% Succ in MM situations.

As the size of the model increases, the performance can get further improved regardless of the type of instructions and the improvement becomes less significant with multiple APPs. As evidenced by LLaMA3 and QWen1.5 series models, we can find that large models mostly lead to better performance. However, when the instruction requires coordination between multiple APPs, most models show a significant drop in performance and some models even get 0 at Succ, such as QWen1.5-7B and 14B. Moreover, the $F1_{app}$ can get around 10% improvement in a single APP while only less than 5% in LLaMA3 series models.

The complexity of planning highly impacts the performance of these models. From the varying scores of different LLMs across different scenarios, a trend in performance emerges: the observed order of performance is approximately: MM < MS < SM < SS. This trend exists in most LLMs such as GPT-4o, QWen1.5-14B, LLaMA3-8B, and LLaMA3-70B. The slight difference between SM and MS can be attributed to different percentages of specific data examples such as the number of APPs and APIs. This kind of trend also aligns well with our intuition that the MM scenario is the most complicated, followed by MS and SM, and SS is the simplest.

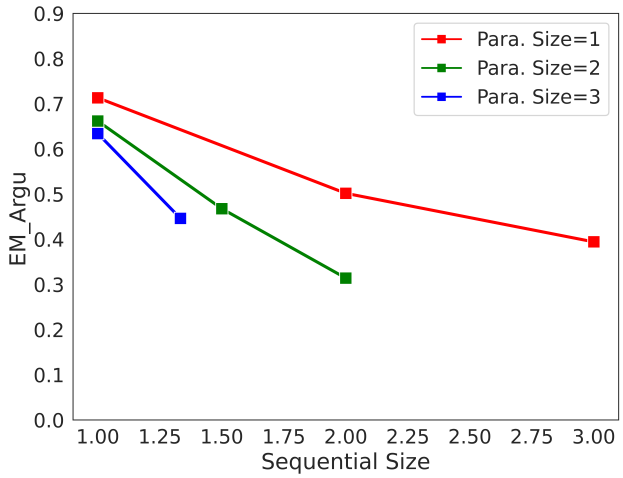


Figure 5.4: *The relationship between GPT-4o's performance with parallel and sequential scaling. Both parallel and sequential scaling cause challenges for model performance.*

5.4.4 Discussion and Analysis

We conduct a comprehensive analysis, aiming to answer three research questions. **RQ1**: *How do the parallel and sequential dependencies influence the model performance?* **RQ2**: *Is it necessary to identify APP first to reduce the context window?* and **RQ3:** *What is the major bottleneck of current LLMs?*

The Effects of Dependency Structures We classify the dependency structures among APIs as twofold: parallel execution and sequential execution. For each data sample, we measure the parallel execution scale by the number of connected components of APIs and use the average size of these API-connected components as the sequential execution scale. The data sample with a sequential scale of 1 means no sequential dependencies among APIs. All of the APIs can be finished in a parallel way. Then, we classify the data samples of `AppBench` based on the above criteria and discard the categories with less than 10 samples.

We illustrate the Exact Match (EM) of Arguments of GPT-4o in Figure 5.4 since arguments are directly related to the dependency relationship. First of all, when the parallel scale is fixed, an increased sequential scale becomes more challenging for GPT-4o, and vice versa. Secondly, GPT-4o appears to struggle more with sequential-complex data than parallel-

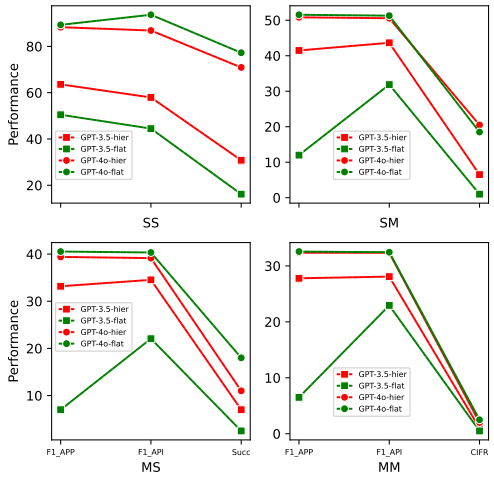


Figure 5.5: *The performance gap between hierarchical and flat prompting on GPT-3.5 and GPT-4o.*

complex samples. The gap between different para. size (i.e., when seq. size is fixed) is much smaller than the gap between different seq. size (i.e., when para. size is fixed).

The Effects of Different Prompting In the main experiments, we initially required LLMs to select candidate APPs based on user input and the APP's descriptions, and then generate API calls, resulting in *hierarchical* prompting. Recently, many studies have expanded the context of LLMs to 200K or more [175]. Many of these works proposed LLMs with a context window that is sufficient to accommodate all the descriptions of APPs and APIs at once (*flat* prompting). Therefore, this section explores how the model would perform if we directly provided all apps and APIs to the model. We test GPT-3.5 and GPT-4o and compare the results in Figure 5.5.

We can observe that flat prompting has impacted the performance of the GPT-3.5, with obvious declines in metrics such as $F1_{app}$ scores across data types. We attribute this to the introduction of a large amount of irrelevant information, which affects the model's understanding and extraction of useful APPs and APIs. Surprisingly, the GPT-4o model achieved better performance using flat prompting. We believe this is due to the GPT-4o's more powerful long-context understanding capabilities, which allow it to accurately identify the required APP and API. Moreover, the absence of the error propagation effect that occurs

Category	Keys		Values		
	I	D	I	D	T/S
SS	6.1	-	6.6	-	42.1/26.3
SM	5.5	8.0	2.5	75.5	27.1/15.2
MS	6.0	1.0	6.0	30.0	45.1/8.8
MM	19.0	15.0	6.0	82.0	36.8/24.6

Table 5.5: *Error analysis of GPT-4o on* `AppBench`*. I and D stand for independent and dependent variables or values, respectively, between multiple APIs. T/S refers to time-related or space-related values, such as start date and location.*

Settings	SS			SM			MS			MM		
	$F1_{app}$	$F1_{api}$	Succ	$F1_{app}$	$F1_{api}$	Succ	$F1_{app}$	$F1_{api}$	Succ	$F1_{app}$	$F1_{api}$	Succ
GPT-4o	88.31	86.87	70.92	50.83	50.57	20.50	39.39	39.14	11.00	32.36	32.35	2.00
3-shot	93.73	90.73	81.63	51.16	50.90	13.50	40.12	39.92	12.50	32.72	32.73	2.50
4-shot	93.23	89.72	79.59	50.96	50.70	14.00	40.29	40.29	10.50	32.44	32.44	3.00
5-shot	93.70	91.18	79.59	50.32	50.06	14.00	40.33	40.12	12.50	32.36	32.36	2.50

Table 5.6: *In-context learning results of GPT-4o on* `AppBench`*.*

during the first APP selection step of hierarchical prompting, has led to a clear improvement in performance. However, flat prompting requires a strong contextual capability that few models possess, and it necessitates the input of a large number of irrelevant tokens, which incurs additional computational power consumption.

Error Analysis We further conduct error analysis at the argument level since it is directly related to different relationships between multiple APIs, to identify potential bottlenecks of the current best model: GPT-4o. Specifically, there are two main categories of errors to consider: *1) key error.* It occurs when the model predicts fewer keys than expected to successfully execute the API call, and it can be further divided into two types: **I**ndependent: The missing or incorrect keys are from the independent variables or arguments and **D**ependent: The missing or incorrect keys are from the dependent variables or arguments; and *2) value error*. it occurs when the model predicts values that do not match the ground truth values, given the name of the key. Value errors can also be divided into I and D types.

Table 5.5 presents the percentage of error cases over the number of total arguments in each category while T/S is the percentage over all error arguments in each category. It is found that as complexity increases, errors also increase. The lower D-key error and D-value

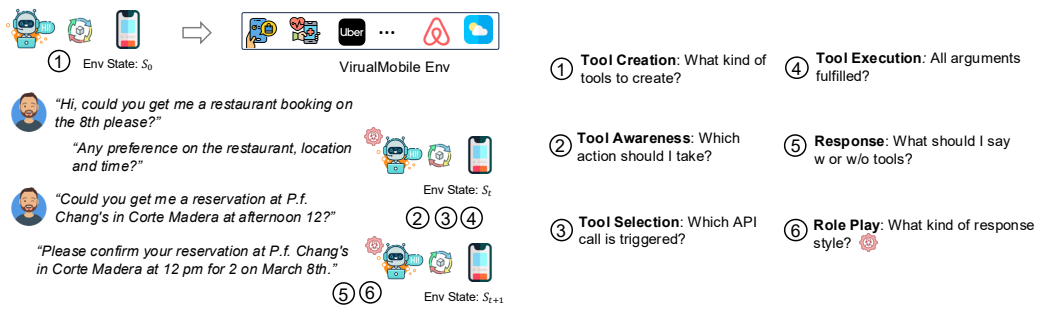


Figure 5.6: *A typical example to show the entire life cycle of stateful tool use in multi-turn dialogues. The dialogue agent need to create the tools first or on the fly ①, and then decide whether or not use tools ②, which tool to use ③, execute it with all required arguments fulfilled ④, convert the tool results into responses with different role configs as conversion goes ⑤⑥.*

error in MS can be attributed to a smaller percentage of dependency cases in this category. Out of all types of errors, the D-value error appears to be the biggest bottleneck or challenge for the LLM. Further analysis reveals that the value errors are particularly prevalent for time and space-related keys. For example, the language models may struggle to accurately recognize or reason about date/time expressions used in the user's input, such as "next Monday".

5.5 Towards More Complex and Personalized Tool Planning

Since `AppBench` mainly focus on single turn tool planning, we further construct 1) `DialogTool`, which is a multi-turn dialogue dataset with stateful tool interactions considering the whole life cycle of tool use, across six key tasks in three stages [176]; 2) `ToolSpectrum`, a benchmark designed to evaluate LLMs' capabilities in personalized tool utilization [177].

5.5.1 DialogTool

Existing benchmarks that assess Language Models (LMs) as Language Agents (LAs) for tool use primarily focus on stateless, single-turn interactions or partial evaluations, such as tool selection in a single turn, overlooking the inherent stateful nature of interactions in multi-turn applications. For instance, when a user fails to provide all the required

arguments to use a tool in a single turn or requests details about a previous tool call, it becomes infeaible to provide detailed response without the tracking of tool states. In addition, most of existing benchmarks or environments fail to address the complexities of real-world interactions across the *entire lifecycle of tool use*, encompassing tool creation, selection, execution, and integration of final responses, especially for tools with varying numbers and types of arguments [6, 161].

To maintain seamless interaction over long horizons, we introduce `DialogTool`, the first benchmark designed to comprehensively evaluate the entire lifecycle of stateful tool use in multi-turn dialogues. Generally, we leverage existing dialogue datasets, particularly task-oriented dialogue datasets (TDD) [170], to gather data and construct the corresponding evaluation environment efficiently and effectively. In detail, on the data side, we regard the *service/domain*, *slots* and *intents* in TDD as different *Apps*, *Arguments*, and *APIs*, and transform every database lookup operation in the dialogue into an API function call, adhering to the standard tool call paradigm [6, 159]. On the environment side, we firstly store the output for each API call as the database, and then manually implement each function for all APIs and ensure the correctness[2], resulting in a virtual mobile environment (`VirtualMobile`) with lots of supported Apps and APIs. For example, the user may want to find one restaurant with specific food and location, and the result can be returned using `FindRestaurant` API in `Restaurant` App that takes the desired food type and location as input parameters, and returns a list of names of matching restaurant.

Building on top of `DialogTool` and `VirtualMobile`, we can assess the entire lifecycle of stateful tool use by examining six dimensions across three different stages (Figure 5.6): 1) **Tool Creation** to generate code function given the whole tool description; 2) **Tool Utilization** which consists of tool awareness to determine whether or not require tools, tool selection to select appropriate API and tool execution to fulfill all arguments; and 3) **Role-consistent Response** to generate final responses according to different roles (i.e., role play) and tool states (i.e., response generation). It is worth noting here that the role playing transforms

[2]Given same input in the dialogue, it can produce same output,

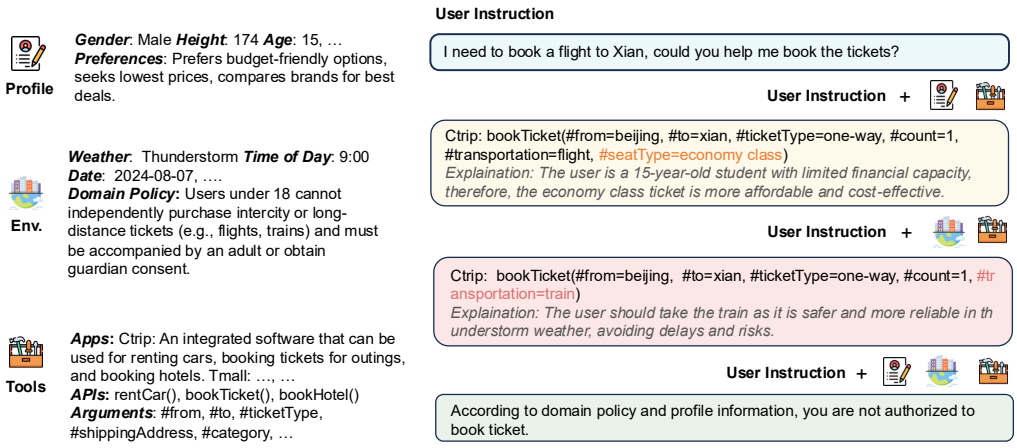


Figure 5.7: *An example from our proposed* `ToolSpectrum`*, illustrating the effects of user profile and environment on personalized tool utilization. This illustrates three distinct scenarios, considering profile-only , environment-only , and combined profile and environment factors .*

responses into different styles to enhance user engagement, independent of the tools being used, allowing for varied expressions regardless of the specific tools employed.

5.5.2 ToolSpectrum

It is crucial to recognize that users with different contexts prefer to utilize different tools when aiming to achieve the same objective [178]. As illustrated in Figure 5.7, when a user prioritizes budget-friendly options, the system should recommend an *economy class* flight ticket as the most suitable option. Besides, suppose environmental factors, such as thunderstorms, make air travel unsafe. In that case, the system should suggest a train ticket as a safer alternative, providing the user with an explanation for this recommendation. Additionally, the system may face further constraints when considering user profiles and environmental factors. For instance, if the user is a minor and domain policies require guardian consent for ticket bookings, the system must restrict the purchase and prompt the user to provide authorization from a guardian. This demonstrates that LLMs must move beyond simple tool selection and instead develop user-centric intelligence. Such

intelligence would allow LLMs to understand the user's context and make more appropriate, personalized tool utilization.

To this end, we develop `ToolSpectrum`, a novel benchmark designed for evaluating personalized tool utilization capabilities of LLMs, which is constructed through a three-stage methodology. Specifically, we first collect commonly used Apps and APIs from high-frequency user scenarios and manually introduce alternative Apps or APIs with similar functionalities but tailored to meet the needs of different contexts (i.e., Temu[3] and Amazon). Next, we identify two critical factors influencing personalized tool utilization: **user profile** and **environment**. These factors have been widely discussed in previous personalization studies [60, 179], and are known to have a significant impact on human behavior patterns [180, 181]. Finally, we simulate real-world user instructions and tool call results, considering the toolset, user profile, and environment, leading to the final `ToolSpectrum`, the comprehensive benchmark for personalized tool utilization.

We further investigate the impact of personalization on tool utilization through extensive experiments using `ToolSpectrum`. The experimental results reveal two key insights: (1) integrating personalization into tool utilization significantly improves its effectiveness, and (2) current LLMs generally underperform on the task of personalized tool utilization.

5.6 Summary

In this chapter, we focus on evaluating the performance of existing LLMs on tool utilization in terms of complex planning (i.e., `AppBench`), stateful tool utilization (i.e., `DialogTool`) and personalized tool utilization (i.e., `ToolSpectrum`), considering the diversity and complexity of practical applications. We hope these benchmarks and environments will provide a comprehensive platform for systematically assessing and advancing language agents' tool-use capabilities, thereby fostering future research in building more adaptive, context-aware, and capable autonomous agents.

[3]Temu is the competitor of Amazon with mostly lower prices.

Chapter 6

Conclusion

In this thesis, on the method side, we establish the theory of agent by providing a unified definition for autonomous agent from tool perspective, illustrating several principle to guide the optimal agent behavior and several promising methods to achieve such goal via prompting engineering, supervised fine-tuning and reinforcement learning, respectively. On the evaluation side, we construct a series of targeted benchmarks to assess the tool-utilization capabilities of existing language models in realistic settings, aiming to guide the development of more capable and adaptive autonomous agents.

Despite recent progress and encouraging achievements in this area, many debates and challenges still remain. In the following sections, we outline these discussions and highlight potential research opportunities that can shape the next generation of agentic intelligence.

6.1 Thesis Summary and Reflections

Building a smart and truly autonomous agent has long been a central goal, aimed at assisting or even replacing humans in performing complex tasks across diverse domains, such as computer-use [5], web navigation [182] and mobile device control [6]. Despite significant progress, the field of agent research remains in a formative stage [183, 184]. Fundamental questions about what an agent is, what constitutes its desired behavior and objective, and how such behavior and objective should be optimized are still under active debate. In detail,

OpenAI defines agents as systems that independently accomplish tasks on human behalf [1], which highlights the controllability of workflow execution to act reliably and consistently. On the other side, Anthropic categorize agents as systems where LLMs dynamically direct their own processes and tool usage, maintaining control over how they accomplish tasks while considering other variations as agentic systems [2]. More recently, DeepMind and Microsoft start to emphasize the importance of world modeling for the agent [183, 185]. These conceptual ambiguities underscore the lack of a universally accepted definition of an "agent" and highlight the need for principled frameworks to guide both theory and practice.

This thesis contributes toward this vision by proposing a systematic framework that redefines agents as goal-oriented tool-use decision makers, capable of coordinating internal cognitive tools and external physical tools to achieve pre-defined objectives efficiently [184]. Building on top of the framework, the core of agent lies in the alignment between knowledge boundary (a.k.a, the boundary between internal world knowledge and external world knowledge) and tool-use decision boundary (a.k.a., the boundary between internal cognitive tools and external physical tools). From this viewpoint, the ***smart agent*** entails an agent's self-awareness to understand its own capabilities and limitations, while ***autonomous agents*** implies minimizing external interactions by internalizing world knowledge within its parametric space (e.g., internal world model). Our proposed methods and benchmarks collectively illustrate how such agents can dynamically balance internal reasoning and external tool use to improve task performance. While our framework offers a principled foundation and roadmap, accompanying with strong empirical results across multiple benchmarks, several limitations remain.

Different Relationships between Internal Knowledge and External Knowledge. We mainly assume that the internal knowledge and external knowledge are two separated parts for simplicity and generalization. In practice, it may not hold since the internal

[1] https://cdn.openai.com/business-guides-and-resources/a-practical-guide-to-building-agents.pdf

[2] https://www.anthropic.com/engineering/building-effective-agents

knowledge may overlap with external knowledge (a.k.a., *knowledge overlap*), and there may exist knowledge conflict between these two sources (a.k.a., *knowledge conflict*). 1) **Knowledge Overlap**: This highlights an important possibility: internal cognitive tools and certain external physical tools can retrieve overlapping or even identical pieces of knowledge, implying a potential for epistemic transferability between the two. For example, a model may answer a factual question either by recalling internalized knowledge from its parameters or by querying an external tool such as a search engine, both pathways leading to the same correct answer [117, 111]. *This interchangeability suggests that internal and external tools can act as substitutes under certain conditions, raising further questions about when and how agents should transfer, balance, or even fuse internal reasoning with external interaction for optimal epistemic efficiency.* In these cases, minimizing external physical tools is also maximizing internal cognitive tools as evidenced by our OTC-PO work [111]. 2) **Knowledge Conflict**: In some cases, internal and external knowledge sources may conflict [151, 152], leading to inconsistent or contradictory information. This typically arises when the model retrieves outdated, incomplete, or hallucinated content from its internal memory that contradicts more up-to-date or accurate external sources. These situations highlight the importance of epistemic calibration: the model must learn not only what it knows, but also when its internal knowledge is unreliable and should be overridden by external input. Addressing knowledge conflict requires mechanisms for knowledge arbitration, where agents resolve discrepancies by evaluating the reliability, recency, and epistemic certainty of each source - an open challenge for building robust decision boundaries under uncertainty.

Availability of Reliable Tools. The effectiveness of our framework depends on the availability and reliability of both internal cognitive tools and external physical tools. However, these assumptions may not always hold in practice. Internally, language agents can suffer from issues such as hallucination or overthinking, leading to incorrect reasoning or unnecessary cognitive steps [154, 186]. Externally, creating or accessing reliable physical tools can be challenging, especially in resource-constrained or highly dynamic environments where tools may be unavailable, outdated, or inconsistent [160, 176]. In addition, recent study try

to create the Model Context Protocol (MCP) to unify diverse tool formats across domains and applications [182]. How to design and create consistent, reliable tools that can support diverse tasks across domains still remains an open challenge.

Self-evolving Agent. During the training and inference, the knowledge boundary and tool-use decision boundary of agent naturally changes accordingly. It is foundamental for agents to track the knowledge boundary and adjust the decision boundary accordingly. For example, as the base model keeps scaling, the knowledge boundary naturally expands as it learned a more accurate internal world model. After pretraining, a model's parametric knowledge boundary becomes relatively static, reflecting what its known knowledge that can be elicited. In contrast, the tool-use decision boundary remains adjustable during the model alignment phase. If a model uses internal tools for knowledge it does not actually possess, this results in hallucinations or incorrect reasoning due to internal tool overuse, Conversely, if the model defers to external tools despite already knowing the answer, it wastes computation and time, an inefficiency stemming from external tool overuse. During the problem-solving, the agent can accumulate certain knowledge and experience from different tasks, leading to a self-evolving agent that can continually learn and adapt from data, interactions, and experiences.

In summary, this thesis marks an important step toward realizing the vision of autonomous and smart language agents. Yet, it also highlights that the field is far from settled. Continued exploration of agent definitions, objectives, and behaviors will be essential to fully unlock their potential in complex, real-world environments.

6.2 Safety, Personalization and Autonomy of Language Agents

As the level of autonomy of agent increases in practical situations, the requirements for safety and personalization also become more demanding, creating an "impossible triangle" between safety, personalization and autonomy.

Safe and Reliable Agent. As language agents become more integrated into everyday applications, especially when they are authorized to interact with external tools, ensuring safe interactions with users and systems is critical [187, 188]. This involves not only preventing harmful content at the model side but also implementing additional safety measures at the system side. Prioritizing safety is essential because it builds trust in these agents, which is key to their adoption and practical use.

Personalized Agent. Everyone prefers to have a personalized and exclusive agent. Since people have diverse personalities, traits, and live in varied environments, tailoring an agent's responses and behavior to individual preferences results in more effective and satisfying user experiences [23, 54]. Achieving high-quality personalization requires language agents to model user intent and preference over time, which often involves dynamic user modeling, continual adaptation, and privacy-preserving learning. Agents must balance responsiveness with stability - adapting to new behaviors while maintaining consistency in long-term preferences. Furthermore, personalized agents must be able to resolve ambiguity by leveraging user-specific context, prioritize relevant information, and calibrate tool usage according to the user's preference. Ultimately, personalization plays a foundational role in making language agents not only useful but also trusted collaborators - agents that feel less like generic tools and more like intelligent assistants.

Autonomous Agent. The key value of agent actually depends on whether or not it can improve the production efficiency, which is largely influenced by the degree of automation it offers in completing tasks [182]. The success of automation depends on the creation of agents that are not only efficient but also transparent and explainable in their decision-making processes. This is particularly important when agents utilize both internal cognitive tools and external physical tools. Cognitive tools can serve a crucial role by providing explanations for the decisions made by the agent, thus offering clarity and rationale for the subsequent use of physical tools.

6.3 Future: Learn from Experience

More broadly, the autonomous agents will inhabit streams of experience, rather than short snippets of interaction. It is time to embrace the era of experience [189] and build more impactful, capable, and practically useful foundation agents, and even further agent society.

Multi-Agent Coordination. Multi-agent coordination extends our framework from individual agents aligning their decision and knowledge boundaries to a collective setting where these boundaries are distributed across multiple agents. In this paradigm, each agent operates with a local view (its own knowledge and decision boundaries), but contributes to a shared task by reasoning about and interacting with other agents. The key challenge is aligning these distributed boundaries to form a coherent collective intelligence. To achieve this, agents must be equipped with mechanisms to communicate epistemic state, and dynamically delegate subtasks to peers whose knowledge boundaries better match the problem context. This requires structured communication protocols, role inference strategies, and shared meta-cognitive modules that manage when to ask, respond, or act. Practically, this can be developed through multi-agent reinforcement learning in environments where cooperation is required for successful task completion, with reward functions encouraging efficient division of cognitive and physical labor.

More Applications. The autonomous agents represent a paradigm shift beyond conversational interfaces, evolving into indispensable, general-purpose collaborators. There are many valuable applications. In scientific discovery (AI for Science), agents will accelerate breakthroughs by autonomously generating hypotheses, synthesizing vast interdisciplinary literature, and orchestrating complex computational or robotic experiments - dramatically compressing research cycles in areas such as drug discovery and materials science [190]. In education, personalized tutor agents will enable truly adaptive, lifelong learning experiences by continuously tailoring instruction, offering nuanced real-time feedback, and providing sustained mentorship at a scale previously unattainable. In software engineering, coding

agents will act as proactive collaborators - debugging intricate systems, refactoring legacy code, managing technical debt, and seamlessly integrating complex APIs to boost developer productivity. This trajectory positions agents as a new layer of infrastructure for solving complex, long-horizon challenges in real world.

References

[1] Hongru Wang, Huimin Wang, Lingzhi Wang, Minda Hu, Rui Wang, Boyang Xue, Yongfeng Huang, and Kam-Fai Wong. Tpe: Towards better compositional reasoning over cognitive tools via multi-persona collaboration. In Derek F. Wong, Zhongyu Wei, and Muyun Yang, editors, *Natural Language Processing and Chinese Computing*, pages 281–294, Singapore, 2025. Springer Nature Singapore.

[2] Yann LeCun. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1):1–62, 2022.

[3] Noam Kolt. Governing ai agents. *arXiv preprint arXiv:2501.07913*, 2025.

[4] 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. *arXiv preprint arXiv:2402.01622*, 2024.

[5] Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094, 2024.

[6] Hongru Wang, Rui Wang, Boyang Xue, Heming Xia, Jingtao Cao, Zeming Liu, Jeff Z. Pan, and Kam-Fai Wong. AppBench: Planning of multiple APIs from various APPs for complex user instruction. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15322–15336, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[7] Yuheng Lu, Qian Yu, Hongru Wang, Zeming Liu, Wei Su, Yanping Liu, Yuhang Guo, Maocheng Liang, Yunhong Wang, and Haifeng Wang. Transbench: Breaking barriers for transferable graphical user interface agents in dynamic digital environments, 2025.

[8] Junde Wu, Jiayuan Zhu, and Yuyuan Liu. Agentic reasoning: Reasoning llms with tools for the deep research. *arXiv preprint arXiv:2502.04644*, 2025.

[9] Thao Nguyen, Tiara Torres-Flores, Changhyun Hwang, Carl Edwards, Ying Diao, and Heng Ji. Glad: Synergizing molecular graphs and language descriptors for enhanced power conversion efficiency prediction in organic photovoltaic devices. In *Proc. 33rd*

ACM International Conference on Information and Knowledge Management (CIKM 2024), 2024.

[10] Carl Edwards, Chi Han, Gawon Lee, Thao Nguyen, Chetan Kumar Prasad, Sara Szymkuc, Bartosz A. Grzybowski, Bowen Jin, Ying Diao, Jiawei Han, Ge Liu, Hao Peng, Martin D. Burke, and Heng Ji. mclm: A function-infused and synthesis-friendly modular chemical language model. In *arxiv*, 2025.

[11] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023.

[12] Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, Wanjun Zhong, Kuanye Li, Jiale Yang, Yu Miao, Woyu Lin, Longxiang Liu, Xu Jiang, Qianli Ma, Jingyu Li, Xiaojun Xiao, Kai Cai, Chuang Li, Yaowei Zheng, Chaolin Jin, Chen Li, Xiao Zhou, Minchao Wang, Haoli Chen, Zhaojian Li, Haihua Yang, Haifeng Liu, Feng Lin, Tao Peng, Xin Liu, and Guang Shi. Ui-tars: Pioneering automated gui interaction with native agents, 2025.

[13] Minghao Li, Feifan Song, Bowen Yu, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. Api-bank: A benchmark for tool-augmented llms, 2023.

[14] Chengpeng Li, Mingfeng Xue, Zhenru Zhang, Jiaxi Yang, Beichen Zhang, Xiang Wang, Bowen Yu, Binyuan Hui, Junyang Lin, and Dayiheng Liu. Start: Self-taught reasoner with tools, 2025.

[15] 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 Jirong Wen. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6), March 2024.

[16] James J Gibson. *The ecological approach to visual perception: classic edition*. Psychology press, 2014.

[17] David H Jonassen. What are cognitive tools? In *Cognitive tools for learning*, pages 1–6. Springer, 1992.

[18] Pei Zhou, Jay Pujara, Xiang Ren, Xinyun Chen, Heng-Tze Cheng, Quoc V Le, Ed Chi, Denny Zhou, Swaroop Mishra, and Huaixiu Steven Zheng. Self-discover: Large language models self-compose reasoning structures. *Advances in Neural Information Processing Systems*, 37:126032–126058, 2024.

[19] Hongru Wang, Deng Cai, Wanjun Zhong, Shijue Huang, Jeff Z. Pan, Zeming Liu, and Kam-Fai Wong. Self-reasoning language models: Unfold hidden reasoning chains with few reasoning catalyst, 2025.

[20] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[21] 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 Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8154–8173, Singapore, December 2023. Association for Computational Linguistics.

[22] 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. *Advances in Neural Information Processing Systems*, 36:43447–43478, 2023.

[23] Hongru Wang, Rui Wang, Fei Mi, Yang Deng, Zezhong Wang, Bin Liang, Ruifeng Xu, and Kam-Fai Wong. Cue-CoT: Chain-of-thought prompting for responding to in-depth dialogue questions with LLMs. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12047–12064, Singapore, December 2023. Association for Computational Linguistics.

[24] Shibo Hao, Tianyang Liu, Zhen Wang, and Zhiting Hu. Toolkengpt: Augmenting frozen language models with massive tools via tool embeddings. *Advances in neural information processing systems*, 36:45870–45894, 2023.

[25] Dongge Han, Trevor McInroe, Adam Jelley, Stefano V. Albrecht, Peter Bell, and Amos Storkey. LLM-personalize: Aligning LLM planners with human preferences via reinforced self-training for housekeeping robots. In Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert, editors, *Proceedings of the 31st International Conference on Computational Linguistics*, pages 1465–1474, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics.

[26] DeepSeek-AI Team. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.

[27] 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.

[28] Rakefet Ackerman and Valerie A. Thompson. Meta-reasoning: Monitoring and control of thinking and reasoning. *Trends in Cognitive Sciences*, 21(8):607–617, 2017.

[29] Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. Demystifying long chain-of-thought reasoning in llms, 2025.

[30] 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, 2025.

[31] Rui Wang, Hongru Wang, Fei Mi, Boyang Xue, Yi Chen, Kam-Fai Wong, and Ruifeng Xu. Enhancing large language models against inductive instructions with dual-critique prompting. In Kevin Duh, Helena Gomez, and Steven Bethard, editors,

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 5345–5363, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[32] Zayne Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. To cot or not to cot? chain-of-thought helps mainly on math and symbolic reasoning. *arXiv preprint arXiv:2409.12183*, 2024.

[33] Vanessa Dye. Reflection, reflection, reflection. i'm thinking all the time, why do i need a theory or model of reflection?'. *Developing Reflective Practice: A guide for beginning teachers. Maidenhead: McGraw-Hill Education*, pages 217–234, 2011.

[34] Rebecca Saxe and Nancy Kanwisher. People thinking about thinking people: the role of the temporo-parietal junction in "theory of mind". In *Social neuroscience*, pages 171–182. Psychology Press, 2013.

[35] Haoyu Song, Yan Wang, Kaiyan Zhang, Wei-Nan Zhang, and Ting Liu. BoB: BERT over BERT for training persona-based dialogue models from limited personalized data. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–177, Online, August 2021. Association for Computational Linguistics.

[36] Liang Chen, Hongru Wang, Yang Deng, Wai-Chung Kwan, Zezhong Wang, and Kam-Fai Wong. Towards robust personalized dialogue generation via order-insensitive representation regularization, 2023.

[37] Deepanway Ghosal, Navonil Majumder, Alexander Gelbukh, Rada Mihalcea, and Soujanya Poria. COSMIC: COmmonSense knowledge for eMotion identification in conversations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2470–2481, Online, November 2020. Association for Computational Linguistics.

[38] Siyang Liu, Chujie Zheng, Orianna Demasi, Sahand Sabour, Yu Li, Zhou Yu, Yong Jiang, and Minlie Huang. Towards emotional support dialog systems, 2021.

[39] Chujie Zheng, Sahand Sabour, Jiaxin Wen, Zheng Zhang, and Minlie Huang. Augesc: Dialogue augmentation with large language models for emotional support conversation, 2023.

[40] Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5370–5381, Florence, Italy, July 2019. Association for Computational Linguistics.

[41] Eric Michael Smith, Mary Williamson, Kurt Shuster, Jason Weston, and Y-Lan Boureau. Can you put it all together: Evaluating conversational agents' ability to

blend skills. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2021–2030, Online, July 2020. Association for Computational Linguistics.

[42] Chujie Zheng, Yong Liu, Wei Chen, Yongcai Leng, and Minlie Huang. CoMAE: A multi-factor hierarchical framework for empathetic response generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 813–824, Online, August 2021. Association for Computational Linguistics.

[43] Jiale Cheng, Sahand Sabour, Hao Sun, Zhuang Chen, and Minlie Huang. Pal: Persona-augmented emotional support conversation generation, 2023.

[44] Valentin Barriere, Shabnam Tafreshi, João Sedoc, and Sawsan Alqahtani. WASSA 2022 shared task: Predicting empathy, emotion and personality in reaction to news stories. In *Proceedings of the 12th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis*, pages 214–227, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[45] Soumitra Ghosh, Dhirendra Maurya, Asif Ekbal, and Pushpak Bhattacharyya. Team IITP-AINLPML at WASSA 2022: Empathy detection, emotion classification and personality detection. In *Proceedings of the 12th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis*, pages 255–260, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[46] Ce Zhou, Qian Li, Chen Li, Jun Yu, Yixin Liu, Guangjing Wang, Kai Zhang, Cheng Ji, Qiben Yan, Lifang He, Hao Peng, Jianxin Li, Jia Wu, Ziwei Liu, Pengtao Xie, Caiming Xiong, Jian Pei, Philip S. Yu, and Lichao Sun. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt, 2023.

[47] Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. A multitask, multilingual, multimodal evaluation of ChatGPT on reasoning, hallucination, and interactivity. In Jong C. Park, Yuki Arase, Baotian Hu, Wei Lu, Derry Wijaya, Ayu Purwarianti, and Adila Alfa Krisnadhi, editors, *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–718, Nusa Dua, Bali, November 2023. Association for Computational Linguistics.

[48] 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.

[49] François Mairesse, Marilyn A Walker, Matthias R Mehl, and Roger K Moore. Using linguistic cues for the automatic recognition of personality in conversation and text. *Journal of artificial intelligence research*, 30:457–500, 2007.

[50] Yla R Tausczik and James W Pennebaker. The psychological meaning of words: Liwc and computerized text analysis methods. *Journal of language and social psychology*, 29(1):24–54, 2010.

[51] H Andrew Schwartz, Johannes C Eichstaedt, Margaret L Kern, Lukasz Dziurzynski, Stephanie M Ramones, Megha Agrawal, Achal Shah, Michal Kosinski, David Stillwell, Martin EP Seligman, et al. Personality, gender, and age in the language of social media: The open-vocabulary approach. *PloS one*, 8(9):e73791, 2013.

[52] Paul Ekman. Universals and cultural differences in facial expressions of emotion. In *Nebraska symposium on motivation*. University of Nebraska Press, 1971.

[53] Peter D Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. *arXiv preprint cs/0212032*, 2002.

[54] Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. Lamp: When large language models meet personalization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7370–7392, 2024.

[55] 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.

[56] Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pages 9118–9147. PMLR, 2022.

[57] Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. Automatic chain of thought prompting in large language models, 2022.

[58] KaShun Shum, Shizhe Diao, and Tong Zhang. Automatic prompt augmentation and selection with chain-of-thought from labeled data, 2023.

[59] 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, 2023.

[60] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[61] Binwei Yao, Chao Shi, Likai Zou, Lingfeng Dai, Mengyue Wu, Lu Chen, Zhen Wang, and Kai Yu. D4: a Chinese dialogue dataset for depression-diagnosis-oriented chat. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2438–2459, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[62] Hao Sun, Zhenru Lin, Chujie Zheng, Siyang Liu, and Minlie Huang. PsyQA: A Chinese dataset for generating long counseling text for mental health support. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1489–1503, Online, August 2021. Association for Computational Linguistics.

[63] Ashish Sharma, Adam S Miner, David C Atkins, and Tim Althoff. A computational approach to understanding empathy expressed in text-based mental health support. In *EMNLP*, 2020.

[64] Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335, 2022.

[65] Yunjie Ji, Yong Deng, Yan Gong, Yiping Peng, Qiang Niu, Lei Zhang, Baochang Ma, and Xiangang Li. Exploring the impact of instruction data scaling on large language models: An empirical study on real-world use cases. *arXiv preprint arXiv:2303.14742*, 2023.

[66] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

[67] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[68] Yoonna Jang, Jungwoo Lim, Yuna Hur, Dongsuk Oh, Suhyune Son, Yeonsoo Lee, Donghoon Shin, Seungryong Kim, and Heuiseok Lim. Call for customized conversation: Customized conversation grounding persona and knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10803–10812, 2022.

[69] Zhenhailong Wang, Shaoguang Mao, Wenshan Wu, Tao Ge, Furu Wei, and Heng Ji. Unleashing the emergent cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 257–279, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[70] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, 2023.

[71] Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068, Singapore, December 2023. Association for Computational Linguistics.

[72] Ming Li, Lichang Chen, Jiuhai Chen, Shwai He, Jiuxiang Gu, and Tianyi Zhou. Selective reflection-tuning: Student-selected data recycling for LLM instruction-tuning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 16189–16211, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[73] Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. Wizardlm: Empowering large language models to follow complex instructions, 2023.

[74] Yiwen Ding, Zhiheng Xi, Wei He, Zhuoyuan Li, Yitao Zhai, Xiaowei Shi, Xunliang Cai, Tao Gui, Qi Zhang, and Xuanjing Huang. Mitigating tail narrowing in llm self-improvement via socratic-guided sampling, 2024.

[75] Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, 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, Zhiyuan Liu, and Maosong Sun. Tool learning with foundation models, 2023.

[76] Hongru Wang, Yujia Qin, Yankai Lin, Jeff Z. Pan, and Kam-Fai Wong. Empowering large language models: Tool learning for real-world interaction. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 2983–2986, New York, NY, USA, 2024. Association for Computing Machinery.

[77] Qian Liu, Yihong Chen, Bei Chen, Jian-Guang Lou, Zixuan Chen, Bin Zhou, and Dongmei Zhang. You impress me: Dialogue generation via mutual persona perception. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1417–1427, Online, July 2020. Association for Computational Linguistics.

[78] Chen Xu, Piji Li, Wei Wang, Haoran Yang, Siyun Wang, and Chuangbai Xiao. COSPLAY. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, jul 2022.

[79] Charles Welch, Chenxi Gu, Jonathan K. Kummerfeld, Veronica Perez-Rosas, and Rada Mihalcea. Leveraging similar users for personalized language modeling with limited data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1742–1752, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[80] Xinchao Xu, Zhibin Gou, Wenquan Wu, Zheng-Yu Niu, Hua Wu, Haifeng Wang, and Shihang Wang. Long time no see! open-domain conversation with long-term persona memory. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2639–2650, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[81] Yifan Liu, Wei Wei, Jiayi Liu, Xianling Mao, Rui Fang, and Dangyang Chen. Improving personality consistency in conversation by persona extending. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. ACM, oct 2022.

[82] Sixing Wu, Ying Li, Dawei Zhang, and Zhonghai Wu. KSAM: Infusing multi-source knowledge into dialogue generation via knowledge source aware multi-head decoding. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 353–363, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[83] Sixing Wu, Ying Li, Minghui Wang, Dawei Zhang, Yang Zhou, and Zhonghai Wu. More is better: Enhancing open-domain dialogue generation via multi-source heterogeneous knowledge. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2286–2300, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[84] Tingchen Fu, Xueliang Zhao, Chongyang Tao, Ji-Rong Wen, and Rui Yan. There are a thousand hamlets in a thousand people's eyes: Enhancing knowledge-grounded dialogue with personal memory. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3901–3913, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[85] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools, 2023.

[86] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface. *arXiv preprint arXiv:2303.17580*, 2023.

[87] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. Webgpt: Browser-assisted question-answering with human feedback, 2022.

[88] Zhiqing Sun, Xuezhi Wang, Yi Tay, Yiming Yang, and Denny Zhou. Recitation-augmented language models, 2023.

[89] Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang. A survey of knowledge-enhanced text generation. *ACM Comput. Surv.*, 54(11s):227:1–227:38, 2022.

[90] Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. Challenges in building intelligent open-domain dialog systems. *ACM Trans. Inf. Syst.*, 38(3), apr 2020.

[91] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241*, 2018.

[92] Hao Zhou, Chujie Zheng, Kaili Huang, Minlie Huang, and Xiaoyan Zhu. KdConv: A Chinese multi-domain dialogue dataset towards multi-turn knowledge-driven conversation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7098–7108, Online, July 2020. Association for Computational Linguistics.

[93] Jing Xu, Arthur Szlam, and Jason Weston. Beyond goldfish memory: Long-term open-domain conversation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5180–5197, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[94] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[95] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[96] Andrea Madotto, Zhaojiang Lin, Chien-Sheng Wu, and Pascale Fung. Personalizing dialogue agents via meta-learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5459, Florence, Italy, July 2019. Association for Computational Linguistics.

[97] 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, 2021.

[98] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.

[99] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering, 2020.

[100] Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. RocketQAv2: A joint training method for dense passage retrieval and passage re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2825–2835, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[101] Alfonso Amayuelas, Kyle Wong, Liangming Pan, Wenhu Chen, and William Yang Wang. Knowledge of knowledge: Exploring known-unknowns uncertainty with large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6416–6432, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[102] Zhibin Gou, Zhihong Shao, Yeyun Gong, yelong shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. ToRA: A tool-integrated reasoning agent for mathematical problem solving. In *The Twelfth International Conference on Learning Representations*, 2024.

[103] Cheng Qian, Emre Can Acikgoz, Hongru Wang, Xiusi Chen, Avirup Sil, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. Smart: Self-aware agent for tool overuse mitigation. *arXiv preprint arXiv:2502.11435*, 2025.

[104] Sijia Chen, Yibo Wang, Yi-Feng Wu, Qing-Guo Chen, Zhao Xu, Weihua Luo, Kaifu Zhang, and Lijun Zhang. Advancing tool-augmented large language models: Integrating insights from errors in inference trees, 2025.

[105] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents, 2023.

[106] Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training, 2025.

[107] Xuefeng Li, Haoyang Zou, and Pengfei Liu. Torl: Scaling tool-integrated rl, 2025.

[108] Martin Riedmiller, Roland Hafner, Thomas Lampe, Michael Neunert, Jonas Degrave, Tom Wiele, Vlad Mnih, Nicolas Heess, and Jost Tobias Springenberg. Learning by playing solving sparse reward tasks from scratch. In *International conference on machine learning*, pages 4344–4353. PMLR, 2018.

[109] Hongru Wang, Huimin Wang, Zezhong Wang, and Kam-Fai Wong. Integrating pretrained language model for dialogue policy evaluation. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6692–6696. IEEE, 2022.

[110] Yiming Du, Hongru Wang, Zhengyi Zhao, Bin Liang, Baojun Wang, Wanjun Zhong, Zezhong Wang, and Kam-Fai Wong. PerLTQA: A personal long-term memory dataset for memory classification, retrieval, and fusion in question answering. In Kam-Fai Wong, Min Zhang, Ruifeng Xu, Jing Li, Zhongyu Wei, Lin Gui, Bin Liang, and Runcong Zhao, editors, *Proceedings of the 10th SIGHAN Workshop on Chinese Language Processing (SIGHAN-10)*, pages 152–164, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[111] Hongru Wang, Cheng Qian, Wanjun Zhong, Xiusi Chen, Jiahao Qiu, Shijue Huang, Bowen Jin, Mengdi Wang, Kam-Fai Wong, and Heng Ji. Otc: Optimal tool calls via reinforcement learning, 2025.

[112] 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.

[113] Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.3, knowledge capacity scaling laws. In *The Thirteenth International Conference on Learning Representations*, 2025.

[114] Moxin Li, Yong Zhao, Yang Deng, Wenxuan Zhang, Shuaiyi Li, Wenya Xie, See-Kiong Ng, and Tat-Seng Chua. Knowledge boundary of large language models: A survey, 2024.

[115] Ce Zheng, Lei Li, Qingxiu Dong, Yuxuan Fan, Zhiyong Wu, Jingjing Xu, and Baobao Chang. Can we edit factual knowledge by in-context learning? In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4862–4876, Singapore, December 2023. Association for Computational Linguistics.

[116] Siyuan Cheng, Ningyu Zhang, Bozhong Tian, Xi Chen, Qingbin Liu, and Huajun Chen. Editing language model-based knowledge graph embeddings. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 17835–17843, 2024.

[117] Hongru Wang, Boyang Xue, Baohang Zhou, Tianhua Zhang, Cunxiang Wang, Huimin Wang, Guanhua Chen, and Kam-Fai Wong. Self-DC: When to reason and when to act? self divide-and-conquer for compositional unknown questions. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6510–6525, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics.

[118] Yash Akhauri, Anthony Fei, Chi-Chih Chang, Ahmed F. AbouElhamayed, Yueying Li, and Mohamed S. Abdelfattah. Splitreason: Learning to offload reasoning, 2025.

[119] Yuxin Xiao, Paul Pu Liang, Umang Bhatt, Willie Neiswanger, Ruslan Salakhutdinov, and Louis-Philippe Morency. Uncertainty quantification with pre-trained language models: A large-scale empirical analysis. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 7273–7284, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[120] Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words, 2022.

[121] Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms, 2023.

[122] Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation, 2023.

[123] Sabrina J. Mielke, Arthur Szlam, Emily Dinan, and Y-Lan Boureau. Reducing conversational agents' overconfidence through linguistic calibration. *Transactions of the Association for Computational Linguistics*, 10:857–872, 2022.

[124] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2023.

[125] Boyang Xue, Hongru Wang, Rui Wang, Sheng Wang, Zezhong Wang, Yiming Du, Bin Liang, and Kam-Fai Wong. A comprehensive study of multilingual confidence estimation on large language models, 2024.

[126] Liangming Pan, Xiaobao Wu, Xinyuan Lu, Anh Tuan Luu, William Yang Wang, Min-Yen Kan, and Preslav Nakov. Fact-checking complex claims with program-guided reasoning. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6981–7004, Toronto, Canada, July 2023. Association for Computational Linguistics.

[127] Omar Khattab, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive nlp, 2023.

[128] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models, 2023.

[129] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada, July 2023. Association for Computational Linguistics.

[130] Wenhao Yu, Zhihan Zhang, Zhenwen Liang, Meng Jiang, and Ashish Sabharwal. Improving language models via plug-and-play retrieval feedback, 2023.

[131] Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9248–9274, Singapore, December 2023. Association for Computational Linguistics.

[132] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William 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*, pages 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[133] Hongru Wang, Minda Hu, Yang Deng, Rui Wang, Fei Mi, Weichao Wang, Yasheng Wang, Wai-Chung Kwan, Irwin King, and Kam-Fai Wong. Large language models as source planner for personalized knowledge-grounded dialogues. In Houda Bouamor,

Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9556–9569, Singapore, December 2023. Association for Computational Linguistics.

[134] Hongru Wang, Boyang Xue, Baohang Zhou, Rui Wang, Fei Mi, Weichao Wang, Yasheng Wang, and Kam-Fai Wong. UniRetriever: Multi-task candidates selection for various context-adaptive conversational retrieval. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue, editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 17074–17086, Torino, Italia, May 2024. ELRA and ICCL.

[135] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022.

[136] Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. Generate rather than retrieve: Large language models are strong context generators. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[137] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021.

[138] Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. Toolqa: A dataset for llm question answering with external tools. *Advances in Neural Information Processing Systems*, 36:50117–50143, 2023.

[139] Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, and Thang Luong. Freshllms: Refreshing large language models with search engine augmentation, 2023.

[140] Matteo Gabburo, Nicolaas Paul Jedema, Siddhant Garg, Leonardo F. R. Ribeiro, and Alessandro Moschitti. Measuring retrieval complexity in question answering systems, 2024.

[141] Yile Wang, Peng Li, Maosong Sun, and Yang Liu. Self-knowledge guided retrieval augmentation for large language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10303–10315, Singapore, December 2023. Association for Computational Linguistics.

[142] Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. Query rewriting in retrieval-augmented large language models. In Houda Bouamor, Juan Pino, and

Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315, Singapore, December 2023. Association for Computational Linguistics.

[143] Neeraj Varshney, Wenlin Yao, Hongming Zhang, Jianshu Chen, and Dong Yu. A stitch in time saves nine: Detecting and mitigating hallucinations of llms by validating low-confidence generation, 2023.

[144] Shicheng Xu, Liang Pang, Huawei Shen, Xueqi Cheng, and Tat-Seng Chua. Search-in-the-chain: Towards accurate, credible and traceable large language models for knowledge-intensive tasks, 2023.

[145] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[146] Hanning Zhang, Shizhe Diao, Yong Lin, Yi Fung, Qing Lian, Xingyao Wang, Yangyi Chen, Heng Ji, and Tong Zhang. R-tuning: Instructing large language models to say 'I don't know'. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7113–7139, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[147] Tianyang Xu, Shujin Wu, Shizhe Diao, Xiaoze Liu, Xingyao Wang, Yangyi Chen, and Jing Gao. SaySelf: Teaching LLMs to express confidence with self-reflective rationales. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5985–5998, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[148] Boyang Xue, Fei Mi, Qi Zhu, Hongru Wang, Rui Wang, Sheng Wang, Erxin Yu, Xuming Hu, and Kam-Fai Wong. Ualign: Leveraging uncertainty estimations for factuality alignment on large language models, 2024.

[149] Jianqiao Lu, Zhiyang Dou, Hongru Wang, Zeyu Cao, Jianbo Dai, Yunlong Feng, and Zhijiang Guo. Autopsv: Automated process-supervised verifier. *Advances in Neural Information Processing Systems*, 37:79935–79962, 2024.

[150] Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. Learning to reason without external rewards, 2025.

[151] Rongwu Xu, Zehan Qi, Zhijiang Guo, Cunxiang Wang, Hongru Wang, Yue Zhang, and Wei Xu. Knowledge conflicts for LLMs: A survey. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8541–8565, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[152] Yu Zhao, Alessio Devoto, Giwon Hong, Xiaotang Du, Aryo Pradipta Gema, Hongru Wang, Xuanli He, Kam-Fai Wong, and Pasquale Minervini. Steering knowledge selection behaviours in LLMs via SAE-based representation engineering. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5117–5136, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics.

[153] Daman Arora and Andrea Zanette. Training language models to reason efficiently, 2025.

[154] Shijue Huang, Hongru Wang, Wanjun Zhong, Zhaochen Su, Jiazhan Feng, Bowen Cao, and Yi R. Fung. Adactrl: Towards adaptive and controllable reasoning via difficulty-aware budgeting, 2025.

[155] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

[156] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control, 2023.

[157] 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, March 2023.

[158] Xiao Liu, Hanyu Lai, Hao Yu, Yifan Xu, Aohan Zeng, Zhengxiao Du, Peng Zhang, Yuxiao Dong, and Jie Tang. Webglm: Towards an efficient web-enhanced question answering system with human preferences. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4549–4560, 2023.

[159] 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. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3102–3116, Singapore, December 2023. Association for Computational Linguistics.

[160] Shijue Huang, Wanjun Zhong, Jianqiao Lu, Qi Zhu, Jiahui Gao, Weiwen Liu, Yutai Hou, Xingshan Zeng, Yasheng Wang, Lifeng Shang, Xin Jiang, Ruifeng Xu, and Qun Liu. Planning, creation, usage: Benchmarking llms for comprehensive tool utilization in real-world complex scenarios, 2024.

[161] 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. In *The Twelfth International Conference on Learning Representations*, 2024.

[162] Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive apis, 2023.

[163] 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, 2024.

[164] Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong Wen. Tool learning with large language models: A survey, 2024.

[165] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtualhome: Simulating household activities via programs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[166] Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. ScienceWorld: Is your agent smarter than a 5th grader? In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11279–11298, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[167] Binfeng Xu, Zhiyuan Peng, Bowen Lei, Subhabrata Mukherjee, Yuchen Liu, and Dongkuan Xu. Rewoo: Decoupling reasoning from observations for efficient augmented language models, 2023.

[168] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. Agentbench: Evaluating llms as agents, 2023.

[169] Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. Agentboard: An analytical evaluation board of multi-turn llm agents, 2024.

[170] Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8689–8696, 2020.

[171] Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. Multiwoz–a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*, 2018.

[172] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

[173] AI@Meta. Llama 3 model card. 2024.

[174] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[175] Yunpeng Huang, Jingwei Xu, Junyu Lai, Zixu Jiang, Taolue Chen, Zenan Li, Yuan Yao, Xiaoxing Ma, Lijuan Yang, Hao Chen, Shupeng Li, and Penghao Zhao. Advancing transformer architecture in long-context large language models: A comprehensive survey, 2024.

[176] Hongru Wang, Wenyu Huang, Yufei Wang, Yuanhao Xi, Jianqiao Lu, Huan Zhang, Nan Hu, Zeming Liu, Jeff Z. Pan, and Kam-Fai Wong. Rethinking stateful tool use in multi-turn dialogues: Benchmarks and challenges, 2025.

[177] Zihao Cheng, Hongru Wang, Zeming Liu, Yuhang Guo, Yuanfang Guo, Yunhong Wang, and Haifeng Wang. Toolspectrum : Towards personalized tool utilization for large language models, 2025.

[178] Peter J Burke and Donald C Reitzes. The link between identity and role performance. *Social psychology quarterly*, pages 83–92, 1981.

[179] Chunyan Mao, Shuaishuai Huang, Mingxiu Sui, Haowei Yang, and Xueshe Wang. Analysis and design of a personalized recommendation system based on a dynamic user interest model, 2024.

[180] Gordon Willard Allport. Personality: A psychological interpretation. 1937.

[181] Walter Mischel. *Personality and assessment*. Psychology Press, 1996.

[182] Jiahao Qiu, Xuan Qi, Tongcheng Zhang, Xinzhe Juan, Jiacheng Guo, Yifu Lu, Yimin Wang, Zixin Yao, Qihan Ren, Xun Jiang, Xing Zhou, Dongrui Liu, Ling Yang, Yue Wu, Kaixuan Huang, Shilong Liu, Hongru Wang, and Mengdi Wang. Alita: Generalist agent enabling scalable agentic reasoning with minimal predefinition and maximal self-evolution, 2025.

[183] Jonathan Richens, David Abel, Alexis Bellot, and Tom Everitt. General agents need world models, 2025.

[184] Hongru Wang, Cheng Qian, Manling Li, Jiahao Qiu, Boyang Xue, Mengdi Wang, Heng Ji, and Kam-Fai Wong. Toward a theory of agents as tool-use decision-makers, 2025.

[185] Xiao Yu, Baolin Peng, Ruize Xu, Michel Galley, Hao Cheng, Suman Nath, Jianfeng Gao, and Zhou Yu. Dyna-think: Synergizing reasoning, acting, and world model simulation in ai agents, 2025.

[186] Rui Wang, Hongru Wang, Boyang Xue, Jianhui Pang, Shudong Liu, Yi Chen, Jiahao Qiu, Derek Fai Wong, Heng Ji, and Kam-Fai Wong. Harnessing the reasoning economy: A survey of efficient reasoning for large language models, 2025.

[187] 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. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15537–15553, 2024.

[188] Kun Wang, Guibin Zhang, Zhenhong Zhou, Jiahao Wu, Miao Yu, Shiqian Zhao, Chenlong Yin, Jinhu Fu, Yibo Yan, Hanjun Luo, et al. A comprehensive survey in llm (-agent) full stack safety: Data, training and deployment. *arXiv preprint arXiv:2504.15585*, 2025.

[189] David Silver and Richard S Sutton. Welcome to the era of experience. *Google AI*, 2025.

[190] Kunlun Zhu, Jiaxun Zhang, Ziheng Qi, Nuoxing Shang, Zijia Liu, Peixuan Han, Yue Su, Haofei Yu, and Jiaxuan You. Safescientist: Toward risk-aware scientific discoveries by llm agents, 2025.