A Survey on (M)LLM-Based GUI Agents

基于 (多模态) 大语言模型 ((M)LLM) 的图形用户界面 (GUI) 代理综述

Fei Tang [1*], Haolei Xu [1*], Hang Zhang [1*], Siqi Chen [1*], Xingyu Wu [1*], Yongliang Shen [1], Wenqi Zhang [1], Guiyang Hou', Zeqi Tan [1], Yuchen Yan [1], Kaitao Song [2], Jian Shao [1], Weiming Lu [1], Jun Xiao [1] and Yueting Zhuang [1]

唐飞 [1*]，徐浩磊 [1*]，张航 [1*]，陈思齐 [1*]，吴星宇 [1*]，沈永亮 [1]，张文琦 [1]，侯贵阳 [1]，谭泽琦 [1]，严宇辰 [1]，宋凯涛 [2]，邵健 [1]，陆伟明 [1]，肖军 [1]，庄月婷 [1]

Abstract—Graphical User Interface (GUI) Agents have emerged as a transformative paradigm in human-computer interaction, evolving from rule-based automation scripts to sophisticated AI-driven systems capable of understanding and executing complex interface operations. This survey provides a comprehensive examination of the rapidly advancing field of (M)LLM-based GUI Agents, systematically analyzing their architectural foundations, technical components, and evaluation methodologies. We identify and analyze four fundamental components that constitute modern GUI Agents: (1) perception systems that integrate text-based parsing with multimodal understanding for comprehensive interface comprehension; (2) exploration mechanisms that construct and maintain knowledge bases through internal modeling, historical experience, and external information retrieval; (3) planning frameworks that leverage advanced reasoning methodologies for task decomposition and execution; and (4) interaction systems that manage action generation with robust safety controls. Through rigorous analysis of these components, we reveal how recent advances in large language models and multimodal learning have revolutionized GUI automation across desktop, mobile, and web platforms. We critically examine current evaluation frameworks, highlighting methodological limitations in existing benchmarks while proposing directions for standardization. This survey also identifies key technical challenges, including accurate element localization, effective knowledge retrieval, long-horizon planning, and safety-aware execution control, while outlining promising research directions for enhancing GUI Agents' capabilities. Our systematic review provides researchers and practitioners with a thorough understanding of the field's current state and offers insights into future developments in intelligent interface automation.

摘要——图形用户界面 (GUI) 代理作为人机交互中的变革性范式，已从基于规则的自动化脚本发展为能够理解并执行复杂界面操作的先进人工智能驱动系统。本文综述了快速发展的基于 (多模态) 大语言模型 ((M)LLM) 的 GUI 代理领域，系统分析其架构基础、技术组成及评估方法。我们识别并分析了构成现代 GUI 代理的四个核心组件:(1) 感知系统，结合基于文本的解析与多模态理解，实现对界面的全面感知；(2) 探索机制，通过内部建模、历史经验和外部信息检索构建并维护知识库；(3) 规划框架，利用先进推理方法进行任务分解与执行；(4) 交互系统，管理动作生成并具备强健的安全控制。通过对这些组件的深入分析，我们揭示了大语言模型和多模态学习的最新进展如何革新了桌面、移动和网页平台上的 GUI 自动化。我们批判性地审视了现有评估框架，指出现有基准测试的方法学局限，并提出标准化方向。本文还识别了关键技术挑战，包括准确的元素定位、高效的知识检索、长远规划及安全感知的执行控制，同时勾勒了提升 GUI 代理能力的有前景研究方向。我们的系统综述为研究人员和从业者提供了对该领域现状的全面理解，并对智能界面自动化的未来发展提供了洞见。

Index Terms-Large Language Model; AI Agent; GUI Agent

关键词——大语言模型；人工智能代理；GUI 代理

1 INTRODUCTION

1 引言

THE ubiquity of graphical user interfaces (GUIs) in modern computing systems has led to an increasing demand for intelligent automation of user interface interactions [1- [4]. Traditional automation approaches, primarily relying on rule-based scripts and screen recording/playback mechanisms, have proven inadequate for handling the complexity and dynamicity of modern interfaces [3]. The emergence of GUI Agents, which are autonomous systems capable of understanding, navigating, and interacting with digital interfaces [2, 3, 5-7], represents a significant advancement in human-computer interaction, offering the potential to bridge the gap between natural language instructions and complex interface operations [5, 7, 8].

图形用户界面 (GUI) 在现代计算系统中的普及催生了对用户界面交互智能自动化的日益增长的需求 [1-4]。传统的自动化方法主要依赖基于规则的脚本和屏幕录制/回放机制，已被证明难以应对现代界面的复杂性和动态性 [3]。GUI 代理作为能够理解、导航并与数字界面交互的自主系统 [2,3,5-7]，代表了人机交互领域的重要进展，具备将自然语言指令与复杂界面操作桥接的潜力 [5,7,8]。

The evolution of GUI automation technologies reflects the broader progress in artificial intelligence [1, 2, 4, 8- [13]. Early attempts at GUI automation were characterized by brittle, hand-crafted rules and simple pattern matching techniques, requiring extensive manual configuration and lacking adaptability to interface changes [4-6]. The introduction of computer vision and pattern recognition methods brought improved flexibility [14] but still struggled with understanding context and handling dynamic elements [5, 7]. The recent integration of large language models (LLMs) [15-19] and multimodal language models (MLLMs) [4, 20- 25] marks a transformative moment in this field, enabling agents to comprehend natural language instructions, reason about complex tasks, and adapt to varying interface states with unprecedented sophistication [8-10, 26, 27]. Notably, with the release of OpenAI o1 and DeepSeek R1's [28] successful replication of its capabilities, researchers have begun applying these reasoning models to GUI automation.

GUI 自动化技术的发展反映了人工智能领域的整体进步 [1,2,4,8-13]。早期的 GUI 自动化尝试以脆弱的手工规则和简单的模式匹配技术为特征，需大量手动配置且缺乏对界面变化的适应能力 [4-6]。计算机视觉和模式识别方法的引入提升了灵活性 [14]，但仍难以理解上下文和处理动态元素 [5,7]。近期大语言模型 (LLMs)[15-19] 和多模态语言模型 (MLLMs)[4,20-25] 的融合标志着该领域的变革性时刻，使代理能够理解自然语言指令、推理复杂任务并适应多变的界面状态，展现出前所未有的智能水平 [8-10,26,27]。值得注意的是，随着 OpenAI o1 和 DeepSeek R1[28] 成功复现其能力，研究者开始将这些推理模型应用于 GUI 自动化。

To address the limitations of earlier approaches and capitalize on recent (M)LLMs advancements [4, 23, 24], contemporary GUI Agents have evolved into sophisticated systems built upon four fundamental technical components. These components work in concert to enable comprehensive interface understanding, knowledge management, intelligent planning, and reliable execution.

为克服早期方法的局限并利用近期 (多模态) 大语言模型 ((M)LLMs) 的进展 [4,23,24]，当代 GUI 代理已发展为基于四个核心技术组件的复杂系统。这些组件协同工作，实现对界面的全面理解、知识管理、智能规划和可靠执行。

- Perception: The agent's ability to understand interfaces has evolved from simple text parsing to sophisticated multimodal comprehension. Recent advances in perception can be categorized into text-based parsing (leveraging DOM/HTML structures) [5-7, 29] and multimodal understanding (utilizing MLLMs and specialized UI models) [8, 10, 26, 30, 31]. However, challenges remain in accurate element localization, dynamic content tracking, and resolution adaptation.

  - 感知: 代理对界面的理解能力已从简单的文本解析发展为复杂的多模态感知。近期感知技术可分为基于文本的解析 (利用 DOM/HTML 结构)[5-7,29] 和多模态理解 (采用多模态语言模型和专用 UI 模型)[8,10,26,30,31]。然而，准确的元素定位、动态内容跟踪及分辨率适配仍面临挑战。

- Exploration: Knowledge acquisition and management have become crucial for effective GUI automation [6, 32]. Agents now build comprehensive knowledge bases incorporating internal understanding (UI functions, element properties), historical experience (task trajectories, skill libraries), and external information (API documentation, web resources) [6, 33-35]. The challenge lies in effectively organizing and retrieving this knowledge to guide decision-making. The integration of advanced reasoning frameworks has significantly improved agents' ability to handle complex tasks [36, 37]. Modern approaches leverage various thought chain methodologies and reactive frameworks for systematic planning [38, 39]. Key challenges include long-horizon planning, error recovery, and maintaining consistency across multiple interaction paths.

  - 探索: 知识获取与管理已成为实现高效 GUI 自动化的关键 [6, 32]。智能体现构建了包含内部理解 (界面功能、元素属性)、历史经验 (任务轨迹、技能库) 及外部信息 (API 文档、网络资源) 的综合知识库 [6, 33-35]。挑战在于如何有效组织和检索这些知识以指导决策。先进推理框架的整合显著提升了智能体处理复杂任务的能力 [36, 37]。现代方法利用多种思维链方法和反应式框架进行系统规划 [38, 39]。主要挑战包括长远规划、错误恢复及多交互路径间的一致性维护。

- Authors denoted by * contributed equally to this work. [1] Zhejiang University, 2 Microsoft Research Asia.

  - 作者中标注 * 者对本工作贡献相同。[1] 浙江大学，2 微软亚洲研究院。

- Email: syl@zju.edu.cn

  - 邮箱:syl@zju.edu.cn

- Github Link: https://github.com/zju-real/Awesome-GUI-Agents

  - Github 链接:https://github.com/zju-real/Awesome-GUI-Agents

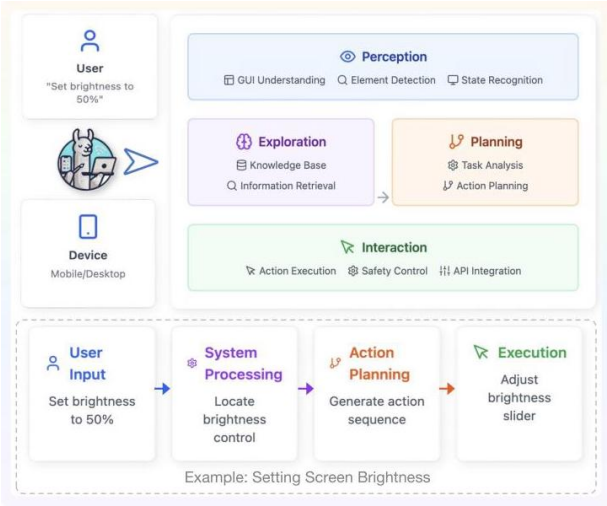

Fig. 1: Overview of GUI Agents

图 1:GUI 智能体概览

- Interaction: The action space has expanded from basic GUI operations to sophisticated API integrations while maintaining safety and reliability [40]. Contemporary agents employ diverse strategies for action generation and execution, with an increasing focus on safety controls and error handling mechanisms.

- 交互: 动作空间已从基础 GUI 操作扩展至复杂 API 集成，同时保持安全性和可靠性 [40]。当代智能体采用多样策略生成和执行动作，且日益重视安全控制与错误处理机制。

These four components - perception, exploration, planning, and interaction - form an integrated pipeline that enables GUI Agents to process natural language instructions, understand interface contexts, plan appropriate actions, and execute them safely. The perception component provides the foundational understanding of the interface, while exploration builds and maintains the knowledge necessary for intelligent decision-making. The planning component leverages this knowledge to formulate effective strategies, which are then carried out through the interaction component. This architectural framework has proven effective across various applications, though each component continues to present unique challenges that drive ongoing research.

感知、探索、规划与交互这四个组成部分构成了一个集成流程，使 GUI 智能体能够处理自然语言指令、理解界面上下文、规划适当动作并安全执行。感知组件提供界面基础理解，探索则构建并维护智能决策所需知识。规划组件利用这些知识制定有效策略，随后通过交互组件执行。该架构在多种应用中表现良好，尽管每个组件仍存在独特挑战，推动持续研究。

Modern GUI Agents operate across a diverse ecosystem of platforms, including desktop applications, mobile devices, and web browsers, each presenting unique challenges and opportunities. Desktop environments demand precise control over complex application interfaces [12], while mobile platforms require adaptation to touch-based interactions and varying screen sizes [8, 41]. Web browsers introduce additional complexity through dynamic content and diverse interface implementations [42, 43]. This variety of deployment contexts has driven the development of increasingly sophisticated agent architectures that can generalize across different platforms while maintaining reliability and safety.

现代 GUI 智能体在多样化平台生态中运行，包括桌面应用、移动设备和网页浏览器，每种平台均带来独特挑战与机遇。桌面环境要求对复杂应用界面进行精确控制 [12]，移动平台需适应触控交互及不同屏幕尺寸 [8, 41]。网页浏览器因动态内容和多样界面实现增加了复杂性 [42, 43]。这种多样部署环境促使智能体架构日益复杂，能够跨平台泛化同时保持可靠性与安全性。

To systematically evaluate the capabilities of GUI Agents and track progress in the field, researchers have developed various datasets and benchmarks. These evaluation frameworks span different platforms and scenarios, from mobile app navigation to web browsing and desktop software operation [8, 29, 41, 42, 44]. Static benchmarks focus on assessing specific component capabilities, such as element localization accuracy or task planning efficiency [8, 45, 46], while dynamic benchmarks evaluate end-to-end performance in real-world scenarios [42, 43, 47]. However, the development of comprehensive evaluation methodologies remains challenging due to the diversity of interfaces, the complexity of user interactions, and the need to assess both functional correctness and user experience aspects. The establishment of standardized evaluation frameworks is crucial for comparing different approaches and guiding future research directions.

为系统评估 GUI 智能体能力并跟踪领域进展，研究者开发了多种数据集和基准测试。这些评估框架涵盖不同平台和场景，从移动应用导航到网页浏览及桌面软件操作 [8, 29, 41, 42, 44]。静态基准侧重评估特定组件能力，如元素定位准确性或任务规划效率 [8, 45, 46]，动态基准则评估真实场景中的端到端性能 [42, 43, 47]。然而，由于界面多样性、用户交互复杂性及需评估功能正确性与用户体验，综合评估方法的开发仍具挑战。建立标准化评估框架对比不同方法并指导未来研究方向至关重要。

This survey provides a comprehensive analysis of GUI Agents research, with the following contributions:

本综述对 GUI 智能体研究进行了全面分析，主要贡献包括:

- A systematic taxonomy of GUI Agents technologies, organized around the four core components: perception, exploration, planning, and interaction.
    - 围绕感知、探索、规划与交互四大核心组件，系统分类 GUI 智能体技术。
- A detailed examination of various technical approaches, from fundamental building blocks to state-of-the-art methodologies.
    - 详细审视从基础构件到最先进方法的多种技术路径。
- A critical analysis of current evaluation methodologies and benchmarks, highlighting their strengths and limitations.
    - 批判性分析当前评估方法与基准，突出其优缺点。
- An in-depth discussion of challenges and opportunities, providing insights for future research directions.
    - 深入讨论挑战与机遇，为未来研究方向提供见解。

The remainder of this survey systematically explores these aspects, beginning with an overview of GUI Agents architectures in Section 2.1 Sections 2.2 through 2.5 delve into the four core components: perception mechanisms, exploration strategies, planning frameworks, and interaction methods. Section 3 examines applications across different platforms, while Section 4 discusses evaluation methodologies and benchmarks. Section 5 analyzes challenges and future directions, followed by conclusions in Section 6 Throughout this comprehensive examination, we aim to provide researchers and practitioners with a thorough understanding of the current state of GUI Agents while highlighting promising directions for future research and development.

本综述余下部分系统探讨上述内容，首先在 2.1 节概述 GUI 智能体架构，2.2 至 2.5 节深入分析感知机制、探索策略、规划框架及交互方法。第 3 节考察不同平台应用，第 4 节讨论评估方法与基准。第 5 节分析挑战与未来方向，第 6 节总结。通过全面审视，旨在为研究者和实践者提供对当前 GUI 智能体状态的深入理解，并突出未来研究与开发的有前景方向。

Through this survey, we seek to illuminate the rapid progress and remaining challenges in GUI Agents development, providing a foundation for future advances in this increasingly important field. As digital interfaces continue to evolve and proliferate, the role of GUI Agents in bridging the gap between human intent and computer execution becomes increasingly crucial, making this systematic examination of the field both timely and essential.

通过本综述，我们旨在揭示 GUI 智能体发展的快速进展与尚存挑战，为该日益重要领域的未来突破奠定基础。随着数字界面不断演进和普及，GUI 智能体在连接人类意图与计算机执行之间的作用愈发关键，使得对该领域的系统性审视既及时又必要。

2 ARCHITECTURE

2 架构

2.1 Overview

2.1 概述

In this section, we present a systematic framework for understanding the architecture of contemporary GUI Agents. As illustrated in Figure 2, we decompose these intelligent systems into four fundamental modules: perception, exploration, planning, and interaction. This functional decomposition not only provides analytical clarity but also reflects the natural progression of information processing within GUI Agents from sensory input to knowledge acquisition, decision making, and ultimately physical action.

本节中，我们提出了一个系统框架，用于理解当代 GUI 代理的架构。如图 2 所示，我们将这些智能系统分解为四个基本模块: 感知、探索、规划和交互。这种功能分解不仅提供了分析的清晰性，也反映了 GUI 代理中信息处理的自然进程，从感官输入到知识获取、决策制定，最终到物理动作的实现。

Through comprehensive analysis of recent literature (2021-2025), we classify GUI Agent architectures into four primary categories: General MLLM-based approaches that leverage off-the-shelf multimodal models; Training MLLM approaches that specifically fine-tune multimodal models for GUI tasks; Multi-Agent frameworks that

distribute specialized functions across collaborative agents; and LLM-Based methodologies that rely primarily on text-based representation of interfaces. Figure 4 tracks the chronological development and publication volume across these architectural paradigms, revealing the field's rapid evolution and growing research interest.

通过对近年文献 (2021-2025) 的全面分析，我们将 GUI 代理架构分为四大类: 利用现成多模态模型的通用多模态大语言模型 (MLLM) 方法；专门针对 GUI 任务微调多模态模型的训练型 MLLM 方法；将专门功能分布于协作代理中的多代理框架；以及主要依赖界面文本表示的基于大语言模型 (LLM) 的方法。图 4 追踪了这些架构范式的时间发展和发表量，揭示了该领域的快速演进和日益增长的研究兴趣。

The information flow within GUI Agents begins with the perception module, which serves as the system's primary sensory interface (§2.2). This module encompasses text-based perception (parsing DOM trees or XML structures) (§2.2.1), multimodal perception (directly processing screen-shots) (§2.2.2), and specialized tools for screen understanding (OCR, object detection) (§2.2.3). Similar to human visual processing, perception provides the fundamental interface understanding upon which all subsequent processes depend. Building on this perceptual foundation, the exploration module enables GUI Agents to acquire, organize, and utilize diverse knowledge sources (§2.3). We identify three distinct knowledge acquisition strategies (§2.3.1): internal exploration (examining interface structures and functionalities), external exploration (retrieving supplementary information from documentation or web resources), and historical exploration (leveraging past interactions and successful trajectories). We also discussed how GUI Agents utilize knowledge in (§2.3.2). This comprehensive knowledge management system enables agents to navigate complex interfaces with contextual awareness and adaptability.

GUI 代理中的信息流始于感知模块，该模块作为系统的主要感官接口 (§2.2)。该模块包括基于文本的感知 (解析 DOM 树或 XML 结构)(§2.2.1)、多模态感知 (直接处理屏幕截图)(§2.2.2) 以及用于屏幕理解的专用工具 (OCR、目标检测)(§2.2.3)。类似于人类视觉处理，感知提供了所有后续过程依赖的基础界面理解。在此感知基础上，探索模块使 GUI 代理能够获取、组织和利用多样的知识源 (§2.3)。我们识别了三种不同的知识获取策略 (§2.3.1): 内部探索 (检查界面结构和功能)、外部探索 (从文档或网络资源检索补充信息) 以及历史探索 (利用过去的交互和成功路径)。我们还讨论了 GUI 代理如何利用知识 (§2.3.2)。这一全面的知识管理系统使代理能够以情境感知和适应性导航复杂界面。

The planning module constitutes the cognitive core of GUI Agents, enabling systematic reasoning about tasks and decision-making (§2.4). Within our taxonomy, we analyze planning capabilities across three dimensions: the underlying reasoning frameworks (whether based on LLMs (§2.4.1), MLLMs (§2.4.2), or advanced reasoning models like o1-series systems (§2.4.3), task planning methodologies (including iterative and decomposition-based approaches), and verification mechanisms for ensuring plan reliability (§2.4.4). This sophisticated planning architecture allows agents to handle multi-step tasks while maintaining consistency and adapting to environmental feedback.

规划模块构成了 GUI 代理的认知核心，使其能够对任务进行系统推理和决策 (§2.4)。在我们的分类中，我们从三个维度分析规划能力: 基础推理框架 (基于 LLM(§2.4.1)、MLLM(§2.4.2) 或如 o1 系列系统等高级推理模型 (§2.4.3))、任务规划方法 (包括迭代和分解方法) 以及确保计划可靠性的验证机制 (§2.4.4)。这一复杂的规划架构使代理能够处理多步骤任务，同时保持一致性并适应环境反馈。

Finally, the interaction module enables GUI Agents to execute planned actions within the target environment (§2.5). We present a systematic classification framework based on action types, generation strategies, and safety control mechanisms. These interaction capabilities represent the culmination of the agent's processing pipeline,

translating internal representations and plans into concrete actions that manipulate the interface.

最后，交互模块使 GUI 代理能够在目标环境中执行规划的动作 (§2.5)。我们提出了基于动作类型、生成策略和安全控制机制的系统分类框架。这些交互能力代表了代理处理流程的顶点，将内部表示和计划转化为具体操作以操控界面。

Together, these four modules-perception, exploration, planning, and interaction-form an integrated information processing pipeline that enables GUI Agents to receive user instructions, comprehend interface contexts, develop appropriate action plans, and execute them reliably. The perception module provides the foundational understanding of the environment, while exploration builds and maintains the knowledge necessary for intelligent decision-making. The planning module leverages this knowledge to formulate effective strategies, which are then implemented through the interaction module. This architectural framework has demonstrated effectiveness across diverse applications, though each component presents unique research challenges that continue to drive innovation in the field.

这四个模块——感知、探索、规划和交互——共同构成了一个集成的信息处理流水线，使 GUI 代理能够接收用户指令、理解界面上下文、制定合适的行动计划并可靠执行。感知模块提供环境的基础理解，探索模块构建并维护智能决策所需的知识。规划模块利用这些知识制定有效策略，随后通过交互模块实施。该架构框架已在多种应用中展现出有效性，尽管每个组件都存在独特的研究挑战，持续推动该领域的创新。

In the following sections, we examine each of these core components in detail, analyzing their theoretical foundations, implementation approaches, and technical challenges. Through this systematic decomposition, we aim to provide a comprehensive understanding of current GUI Agent architectures while highlighting opportunities for future architectural innovations.

在接下来的章节中，我们将详细考察这些核心组件，分析其理论基础、实现方法及技术挑战。通过这种系统分解，我们旨在提供对当前 GUI 代理架构的全面理解，同时突出未来架构创新的机遇。

2.2 Perception

2.2 感知

The eye is an important part of the human body, capable of collecting visual information that is then reflected to the brain for decision-making. Similarly, the perception of a GUI Agents is also a crucial module that acquires multimodal information from the screen or the internal structure of the current screen, thereby aiding in task planning and execution. After extensive literature research, we can categorize the perception mechanisms of GUI Agents into text-based and multimodal-based approaches, as shown in Figure 3 For instance, text-based GUI Agents [6, 29, 77, 78] perceive the current screen by parsing web HTML or system-provided XML files into structured textual representations. In contrast, multimodal GUI Agents [54, 79] process screen-shots directly, utilizing specialized models pre-trained and fine-tuned on GUI grounding data to accurately localize and interpret interface elements from visual information.

眼睛是人体的重要组成部分，能够收集视觉信息并传递给大脑以辅助决策。同样，GUI 代理的感知模块也是关键组成部分，负责从屏幕或当前界面的内部结构中获取多模态信息，从而辅助任务规划和执行。经过广泛的文献调研，我们将 GUI 代理的感知机制分为基于文本和基于多模态两类，如图 3 所示。例如，基于文本的 GUI 代理 [6, 29, 77, 78] 通过解析网页 HTML 或系统提供的 XML 文件，将当前屏幕转化为结构化的文本表示。相比之下，多模态 GUI 代理 [54, 79] 直接处理屏幕截图，利用在 GUI 定位数据上预训练和微调的专用模型，准确定位并解读视觉信息中的界面元素。

In the following sections, we delve deeper into these two perception mechanisms: Section (§2.2.1) provides a detailed examination of text-based perception methods, elucidating how these techniques leverage structured data to understand interface semantics; Section (§2.2.2) explores multimodal perception technologies that enable direct visual understanding of interfaces. Additionally, we investigate the emerging area of tool-enhanced perception (§2.2.3), exploring how specialized utilities and frameworks augment a GUI Agent's ability to interpret complex screen elements beyond the limitations of either text-based or vision-based approaches alone.

在以下章节中，我们将深入探讨这两种感知机制: 第 §2.2.1 节详细审视基于文本的感知方法，阐明这些技术如何利用结构化数据理解界面语义；第 §2.2.2 节探讨多模态感知技术，使界面能够被直接视觉理解。此外，我们还研究了新兴的工具增强感知领域 (§2.2.3)，探讨专用工具和框架如何增强 GUI 代理 (GUI Agent) 对复杂屏幕元素的理解能力，超越单纯基于文本或视觉方法的局限。

2.2.1 Text-based LLM GUI Agent

2.2.1 基于文本的 LLM GUI 代理

Text-based GUI Agents rely solely on textual information as input and predominantly utilize closed-source large language models such as GPT-4o, Claude-3.5-Sonnet. For instance, AutoDroid [32] proposed its own perception method, parsing the UI page into HTML format for input to the large model. AiTW [80] employed the PaLM [81] model, with inputs also resembling HTML structures. Notably, it also utilized OCR tools to recognize icons and text for information enhancement. MobileAgent [50] is a text-based GUI Agent that employs standard program operations for context learning, enabling the model to understand what to do next. Its input is a prompt that includes task objectives, operation history, and the current page's DOM information. Like AiTW, it also uses OCR and icon detection tools. AssistGUI [82] conducted experiments related to productivity tools using the closed-source large model GPT- 3.5 [83]. Specifically, it learns from instructional videos for productivity tools such as Photoshop and Premiere Pro. For AssistGUI [82], it employs various tools like Google OCR and YOLOv8 to parse the current screen and integrate the information into the prompt. WebArena [78] is a Web Agent that operates on browsers, creating a virtual environment focused on real-world scenarios by utilizing task instructions and DOM trees derived from the structure of current webpages.

基于文本的 GUI 代理仅依赖文本信息作为输入，主要使用闭源大型语言模型，如 GPT-4o、Claude-3.5-Sonnet。例如，AutoDroid [32] 提出了其自身的感知方法，将 UI 页面解析为 HTML 格式输入大型模型。AiTW [80] 采用了 PaLM [81] 模型，输入同样类似 HTML 结构。值得注意的是，它还利用 OCR 工具识别图标和文本以增强信息。MobileAgent [50] 是一种基于文本的 GUI 代理，采用标准程序操作进行上下文学习，使模型能够理解下一步操作。其输入为包含任务目标、操作历史和当前页面 DOM 信息的提示。与 AiTW 类似，它也使用 OCR 和图标检测工具。AssistGUI [82] 使用闭源大型模型 GPT-3.5 [83] 进行了与生产力工具相关的实验，具体学习 Photoshop 和 Premiere Pro 等生产力工具的教学视频。对于 AssistGUI [82]，它采用 Google OCR 和 YOLOv8 等多种工具解析当前屏幕，并将信息整合进提示中。WebArena [78] 是一款基于浏览器的 Web 代理，通过利用任务指令和当前网页结构的 DOM 树，创建聚焦于真实场景的虚拟环境。

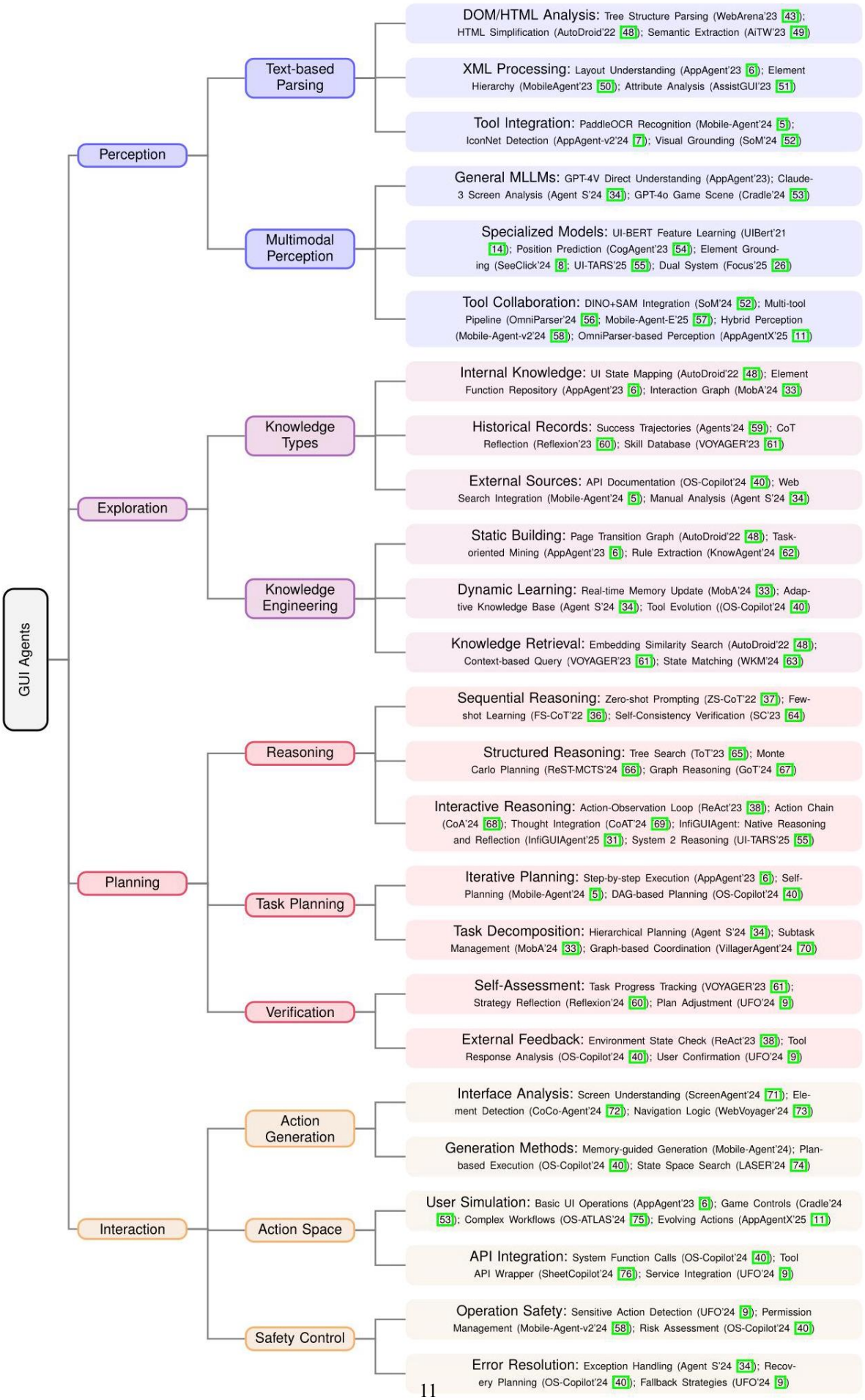
GUI Agents
Perception
Text-based Parsing
DOM/HTML Analysis: Tree Structure Parsing (WebArena'23 43); HTML Simplification (AutoDroid'22 48); Semantic Extraction (AiTW'23 49)
XML Processing: Layout Understanding (AppAgent'23 6); Element Hierarchy (MobileAgent'23 50); Attribute Analysis (AssistGUI'23 51)
Tool Integration: PaddleOCR Recognition (Mobile-Agent'24 5); IconNet Detection (AppAgent-v2'24 7); Visual Grounding (SoM'24 52)
Multimodal Perception
General MLLMs: GPT-4V Direct Understanding (AppAgent'23); Claude-3 Screen Analysis (Agent S'24 34); GPT-4o Game Scene (Cradle'24 53)
Specialized Models: UI-BERT Feature Learning (UIBert'21 14); Position Prediction (CogAgent'23 54); Element Grounding (SeeClick'24 8; UI-TARS'25 55); Dual System (Focus'25 26)
Tool Collaboration: DINO+SAM Integration (SoM'24 52); Multi-tool Pipeline (OmniParser'24 56; Mobile-Agent-E'25 57); Hybrid Perception (Mobile-Agent-v2'24 58); OmniParser-based Perception (AppAgentX'25 11)
Exploration
Knowledge Types
Internal Knowledge: UI State Mapping (AutoDroid'22 48); Element Function Repository (AppAgent'23 6); Interaction Graph (MobA'24 33)
Historical Records: Success Trajectories (Agents'24 59); CoT Reflexion (Reflexion'23 60); Skill Database (VOYAGER'23 61)
External Sources: API Documentation (OS-Copilot'24 40); Web Search Integration (Mobile-Agent'24 5); Manual Analysis (Agent S'24 34)
Knowledge Engineering
Static Building: Page Transition Graph (AutoDroid'22 48); Task-oriented Mining (AppAgent'23 6); Rule Extraction (KnowAgent'24 62)
Dynamic Learning: Real-time Memory Update (MobA'24 33); Adaptive Knowledge Base (Agent S'24 34); Tool Evolution ((OS-Copilot'24 40)
Knowledge Retrieval: Embedding Similarity Search (AutoDroid'22 48); Context-based Query (VOYAGER'23 61); State Matching (WKM'24 63)
Planning
Reasoning
Sequential Reasoning: Zero-shot Prompting (ZS-CoT'22 37); Few-shot Learning (FS-CoT'22 36); Self-Consistency Verification (SC'23 64)
Structured Reasoning: Tree Search (ToT'23 65); Monte Carlo Planning (ReST-MCTS'24 66); Graph Reasoning (GoT'24 67)
Interactive Reasoning: Action-Observation Loop (ReAct'23 38); Action Chain (CoA'24 68); Thought Integration (CoAT'24 69); InfiGUIAgent: Native Reasoning and Reflection (InfiGUIAgent'25 31); System 2 Reasoning (UI-TARS'25 55)
Task Planning
Iterative Planning: Step-by-step Execution (AppAgent'23 6); Self-Planning (Mobile-Agent'24 5); DAG-based Planning (OS-Copilot'24 40)
Task Decomposition: Hierarchical Planning (Agent S'24 34); Subtask Management (MobA'24 33); Graph-based Coordination (VillagerAgent'24 70)
Verification
Self-Assessment: Task Progress Tracking (VOYAGER'23 61); Strategy Reflection (Reflexion'24 60); Plan Adjustment (UFO'24 9)
External Feedback: Environment State Check (ReAct'23 38); Tool Response Analysis (OS-Copilot'24 40); User Confirmation (UFO'24 9)
Interaction
Action Generation
Interface Analysis: Screen Understanding (ScreenAgent'24 71); Element Detection (CoCo-Agent'24 72); Navigation Logic (WebVoyager'24 73)
Generation Methods: Memory-guided Generation (Mobile-Agent'24); Plan-based Execution (OS-Copilot'24 40); State Space Search (LASER'24 74)
Action Space
User Simulation: Basic UI Operations (AppAgent'23 6); Game Controls (Cradle'24 53); Complex Workflows (OS-ATLAS'24 75); Evolving Actions (AppAgentX'25 11)
API Integration: System Function Calls (OS-Copilot'24 40); Tool API Wrapper (SheetCopilot'24 76); Service Integration (UFO'24 9)
Safety Control
Operation Safety: Sensitive Action Detection (UFO'24 9); Permission Management (Mobile-Agent-v2'24 58); Risk Assessment (OS-Copilot'24 40)
Error Resolution: Exception Handling (Agent S'24 34); Recovery Planning (OS-Copilot'24 40); Fallback Strategies (UFO'24 9)

Fig. 2: A Comprehensive Taxonomy of GUI Agents Components

图 2:GUI 代理组件的综合分类

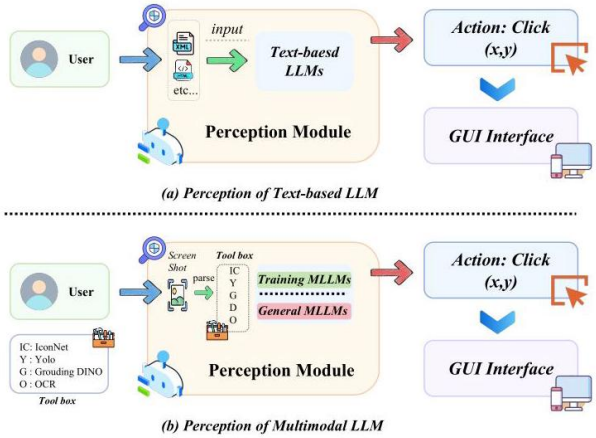

Fig. 3: Comparison between Text-based and MLLM-based GUI Agents. Text-based agents process textual information such as XML or DOM trees, while MLLM-based agents utilize screen screenshots. Some MLLM-based GUI Agents additionally employ specialized tools for element localization, including IconNet, YOLO, Grounding DINO, and OCR.

图 3: 基于文本与基于多模态大型语言模型 (MLLM)GUI 代理的比较。基于文本的代理处理 XML 或 DOM 树等文本信息，而基于 MLLM 的代理则利用屏幕截图。一些基于 MLLM 的 GUI 代理还使用专用工具进行元素定位，包括 IconNet、YOLO、Grounding DINO 和 OCR。

However, Text-based GUI Agents still face several limitations as below:

然而，基于文本的 GUI 代理仍面临以下若干限制:

- Redundancy: Text-based GUI Agents must process extensive HTML/DOM trees or XML files, much of which contains redundant information not relevant to the current task [13, 55].
  - 冗余: 基于文本的 GUI 代理必须处理大量 HTML/DOM 树或 XML 文件，其中许多信息与当前任务无关，存在冗余 [13, 55]。
- Noise: Textual representations often include implementation details, styling information, and other noise that can distract from the core interactive elements [5, 55].
  - 噪声: 文本表示通常包含实现细节、样式信息及其他噪声，可能干扰核心交互元素的识别 [5, 55]。
- Availability: On various platforms, comprehensive textual information (complete XML or HTML structures) may be inaccessible or incomplete, limiting the agent's understanding of the interface [13, 55].

- 可用性: 在不同平台上，完整的文本信息 (完整 XML 或 HTML 结构) 可能不可获取或不完整，限制了代理对界面的理解 [13, 55]。

To address these challenges, researchers have begun using multimodal large language models for GUI screen understanding [4, 5, 8, 10, 29], which enables agents to directly interpret visual elements of interfaces, circumventing the limitations of purely textual representations and enhancing their ability to navigate and interact with graphical environments.

为应对这些挑战，研究者开始采用多模态大型语言模型 (MLLM) 进行 GUI 屏幕理解 [4, 5, 8, 10, 29]，使代理能够直接解读界面的视觉元素，绕过纯文本表示的局限，提升其在图形环境中的导航和交互能力。

2.2.2 Multimodal LLM-based GUI Agent

2.2.2 基于多模态 LLM 的 GUI 代理

Due to the limitations of text-based LLM GUI Agents, researchers increasingly leverage MLLMs to enhance agent perception. Recent studies on multimodal GUI Agents can be categorized into three approaches: (1) General MLLM Approaches: These directly utilize general-purpose models like GPT-4V for screen understanding without specialized training for GUI tasks. (2) Training MLLM Approaches: These involve training dedicated multimodal models for deeper UI comprehension, exemplified by CogAgent [54]. (3) Tool-augmented MLLM Approaches: These employ multimodal models enhanced with supplementary tools such as OCR and Grounding DINO [84] for comprehensive UI interpretation.

鉴于基于文本的 LLM GUI 代理的局限，研究者日益利用多模态大型语言模型 (MLLM) 增强代理感知。近期关于多模态 GUI 代理的研究可分为三类:(1) 通用 MLLM 方法: 直接利用 GPT-4V 等通用模型进行屏幕理解，无需针对 GUI 任务的专门训练；(2) 训练型 MLLM 方法: 训练专用多模态模型以实现更深入的 UI 理解，典型代表为 CogAgent [54]；(3) 工具增强型 MLLM 方法: 结合 OCR 和 Grounding DINO [84] 等辅助工具，提升多模态模型的全面 UI 解析能力。

2.2.2.1 Understanding UI with General Multimodal Large Models With the advancement of multimodal technology, powerful models such as GPT-4V, GPT-4o, and Claude-3.5 Sonnet have emerged. Most research leveraging these general multimodal large models for screen understanding relies on closed-source systems.

2.2.2.1 利用通用多模态大型模型理解 UI 随着多模态技术的发展，GPT-4V、GPT-4o 和 Claude-3.5 Sonnet 等强大模型相继出现。大多数利用这些通用多模态大型模型进行屏幕理解的研究依赖闭源系统。

Several notable implementations demonstrate this approach. AppAgent [6] employs GPT-4 to perceive virtual environments through screenshots and XML files, storing learned information in relevant documents. OS-Copilot [40] similarly utilizes GPT-4 in its three-module system (planner, configurator, executor) but distinguishes itself by generating code segments to operate software, such as using python-ppt for PowerPoint manipulation. The work by [77] represents a web-focused implementation that leverages GPT- 4 for perceiving screens through element attributes, text options, and image annotations. Mobile-Agent [5] addresses the limitations of text-based GUI

Agents' dependence on XML and HTML by integrating GPT-4 with supplementary tools like OCR and Grounding DINO [84]. Similarly, UFO [26] combines MLLMs for Windows platform operations, while Mobile-Agent-V2 [58] expands on its predecessor by implementing a multi-agent framework that utilizes both text-based GPT-4 and GPT-4V.

若干显著的实现展示了这一方法。AppAgent [6] 利用 GPT-4 通过截图和 XML 文件感知虚拟环境，并将学习到的信息存储在相关文档中。OS-Copilot [40] 同样在其由规划器、配置器和执行器组成的三模块系统中使用 GPT-4，但其特点是生成代码片段以操作软件，例如使用 python-ppt 进行 PowerPoint 操作。[77] 的工作代表了一种面向网页的实现，利用 GPT-4 通过元素属性、文本选项和图像注释感知屏幕。Mobile-Agent [5] 针对基于文本的 GUI 代理依赖 XML 和 HTML 的局限，结合了 GPT-4 与 OCR 和 Grounding DINO [84] 等辅助工具。类似地，UFO [26] 结合了多模态大模型 (MLLMs) 用于 Windows 平台操作，而 Mobile-Agent-V2 [58] 在前者基础上扩展，实施了一个多代理框架，利用文本版 GPT-4 和 GPT-4V。

Recent implementations have further specialized these approaches. Cradle [53] creates a game-playing framework using GPT-40, enhanced with SAM [85] and Grounding DINO for improved perception. AppAgent v2 [7] builds upon AppAgent by enhancing UI understanding capabilities through XML processing and integrating OCR and detection models. Agent S [34] specifically targets positioning capabilities using GPT-4o and Claude-3.5-Sonnet, incorporating PaddleOCR [86] for screenshot understanding. Mobile-Agent-E [57] employs GPT-4o for planning while utilizing OCR, icon grounding, and icon captioning for GUI comprehension. The Operator framework [87] leverages GPT-4o's visual capabilities alongside reinforcement learning for screen interaction, while Deep Research integrates OpenAI's o3 model to enable efficient multi-step reasoning for task completion.

近期的实现进一步专门化了这些方法。Cradle [53] 创建了一个使用 GPT-4o 的游戏框架，结合 SAM [85] 和 Grounding DINO 以提升感知能力。AppAgent v2 [7] 在 AppAgent 基础上，通过 XML 处理增强了 UI 理解能力，并整合了 OCR 和检测模型。Agent S [34] 专注于定位能力，使用 GPT-4o 和 Claude-3.5-Sonnet，结合 PaddleOCR [86] 进行截图理解。Mobile-Agent-E [57] 使用 GPT-4o 进行规划，同时利用 OCR、图标定位和图标描述实现 GUI 理解。Operator 框架 [87] 利用 GPT-4o 的视觉能力结合强化学习进行屏幕交互，而 Deep Research 集成了 OpenAI 的 o3 模型，实现高效的多步推理以完成任务。

GUI Agents based on general large models often integrate various tools to enhance their capabilities. However, these approaches face several significant limitations:

基于通用大模型的 GUI 代理通常整合多种工具以增强其能力。然而，这些方法面临若干显著限制:

- Localization Precision: Despite claims of element localization capabilities [8, 77, 88], general MLLMs struggle with the precise identification and targeting of specific GUI components, particularly in complex or densely populated interfaces.

  - 定位精度: 尽管声称具备元素定位能力 [8, 77, 88]，通用多模态大模型在精确识别和定位特定 GUI 组件方面仍存在困难，尤其是在复杂或密集界面中。

- Domain-Specific Understanding: GUI interfaces employ unique visual languages, interaction patterns, and structural hierarchies that differ significantly from natural images. General models lack sufficient pre-training on GUI-specific datasets, resulting in limited comprehension of interface semantics, component

relationships, and functional affordances.

- 领域特定理解:GUI 界面采用独特的视觉语言、交互模式和结构层级，与自然图像有显著差异。通用模型缺乏针对 GUI 专用数据集的充分预训练，导致对界面语义、组件关系和功能可供性的理解有限。

- API Dependencies: Most implementations rely on closed-source models requiring API access, introducing substantial computational costs and potential rate limitations.

- API 依赖: 大多数实现依赖于需要 API 访问的闭源模型，带来较高的计算成本和潜在的调用频率限制。

To address these limitations, researchers have developed specialized approaches focused on training multimodal large models specifically for GUI understanding.

为解决这些限制，研究人员开发了专门针对 GUI 理解训练的多模态大模型方法。

2.2.2.2 Understanding UI with Trained Multimodal Large Models To address these fundamental limitations, researchers have developed training models specifically for GUI understanding.

2.2.2.2 利用训练的多模态大模型理解 UI 为解决这些根本性限制，研究人员开发了专门用于 GUI 理解的训练模型。

Early work based on multimodal training to understand screens, such as UIBert [14], is a Transformer-based [89] model that learns general feature representations of UI interfaces through UI images, text, and view distributions, and is designed with five pre-training tasks to achieve a general model. Another early work is META-GUI [90], which uses Faster R-CNN [91] to extract image features from UI pages and BERT [92] to extract text features, merging different modal features for training.

基于多模态训练以理解屏幕的早期工作如 UIBert [14]，是一种基于 Transformer [89] 的模型，通过 UI 图像、文本和视图分布学习 UI 界面的一般特征表示，设计了五个预训练任务以实现通用模型。另一早期工作 META-GUI [90] 使用 Faster R-CNN [91] 从 UI 页面提取图像特征，使用 BERT [92] 提取文本特征，融合不同模态特征进行训练。

However, existing multimodal models still face issues with inaccurate UI localization and are affected by resolution, leading to potential failure in recognizing smaller icons [8, 54]. CogAgent [54] addressed localization issues by pre-training to output element coordinates and fusing high-resolution and low-resolution images. SeeClick [8] constructed the multi-platform ScreenSpot dataset, training with Qwen2-VL [93] to improve screen perception. CoCo-Agent [72] leveraged LLaVA [94] to output precise element coordinates while incorporating OCR and icon recognition tools. MobileFlow [30] tackled Chinese understanding and privacy concerns using ViT [95] and LayoutLMv3 [96] for image encoding with an MoE-based [97] LLM. AutoGLM [98] separated localization and planning tasks for Android and Web interfaces, proposing self-evolution methods using ChatGLM [99]. OS-ATLAS [75] utilized large-scale datasets with screen screenshots, element instructions, and coordinates for GUI perception. ShowUI [10] employed a lightweight vision-language-action framework for element localization. Aguvis [100] enhanced localization accuracy through pure vision architecture and large-scale GUI dataset pre-training. Aria-UI [101] leverages a Mixture-of-Experts (MoE) approach as a purely visual multimodal model, strengthening

contextual awareness and localization through text-image hybrid operation histories. InfiGUIAgent [31] proposes a two-stage supervised fine-tuning paradigm: the initial stage focuses on foundational visual-semantic understanding, while the subsequent stage enhances multi-step reasoning capabilities to optimize GUI interaction logic. Notably, InfiGUIAgent [31] introduces a hierarchical expectation-reflection reasoning framework coupled with dual-phase training mechanisms to deepen GUI semantic parsing. PC-Agent [12] improved localization robustness by combining Molmo [102] multimodal capabilities with external feedback from system APIs for self-verification. UI-TARS [55] enhances screen perception by utilizing extensive GUI corpora for training. AppAgentX [11] perceives screens by using OmniParser [56] to detect UI elements in screenshots before passing the annotated images to an LLM for action planning.

然而，现有的多模态模型仍面临 UI 定位不准确的问题，且受分辨率影响，导致识别较小图标时可能失败 [8, 54]。CogAgent[54] 通过预训练输出元素坐标并融合高分辨率与低分辨率图像来解决定位问题。SeeClick[8] 构建了多平台 ScreenSpot 数据集，使用 Qwen2-VL[93] 进行训练以提升屏幕感知能力。CoCo-Agent[72] 利用 LLaVA[94] 输出精确的元素坐标，同时结合 OCR 和图标识别工具。MobileFlow[30] 采用 ViT[95] 和 LayoutLMv3[96] 进行图像编码，结合基于 MoE[97] 的大型语言模型，解决中文理解和隐私问题。AutoGLM[98] 将 Android 和 Web 界面的定位与规划任务分离，提出基于 ChatGLM[99] 的自我进化方法。OS-ATLAS[75] 利用包含屏幕截图、元素指令和坐标的大规模数据集进行 GUI 感知。ShowUI[10] 采用轻量级视觉-语言-动作框架进行元素定位。Aguvis[100] 通过纯视觉架构和大规模 GUI 数据集预训练提升定位准确性。Aria-UI[101] 作为纯视觉多模态模型，采用 Mixture-of-Experts(MoE) 方法，通过文本-图像混合操作历史增强上下文感知和定位能力。InfiGUIAgent[31] 提出两阶段监督微调范式: 初始阶段专注于基础视觉语义理解，后续阶段增强多步推理能力以优化 GUI 交互逻辑。值得注意的是，InfiGUIAgent[31] 引入分层期望-反思推理框架及双阶段训练机制，深化 GUI 语义解析。PC-Agent[12] 结合 Molmo[102] 多模态能力与系统 API 的外部反馈进行自我验证，提升定位鲁棒性。UI-TARS[55] 通过利用大量 GUI 语料库进行训练，增强屏幕感知。AppAgentX[11] 先使用 OmniParser[56] 检测截图中的 UI 元素，再将带注释的图像传递给大型语言模型进行动作规划。

Despite these specialized approaches, training models specifically for GUI understanding still faces several notable limitations:

尽管有这些专门的方法，专门针对 GUI 理解训练模型仍面临若干显著限制:

- Cross-Platform Inconsistency: GUI elements vary significantly in style, structure, and interaction patterns across web, mobile, and desktop platforms, making it difficult for a single model to generalize effectively.

  - 跨平台不一致性:GUI 元素在网页、移动端和桌面平台上的风格、结构和交互模式差异显著，导致单一模型难以有效泛化。

- Resolution Constraints: GUI images typically require high resolution for accurate parsing, yet many models struggle to identify and interact with small icons and elements crucial for completing tasks.

  - 分辨率限制:GUI 图像通常需要高分辨率以实现准确解析，但许多模型难以识别和操作完成任务所需的小图标和元素。

- Training Resource Intensity: Developing dedicated GUI Agents demands extensive computational resources and specialized datasets that are costly to create and maintain.

- 训练资源密集: 开发专用 GUI 代理需要大量计算资源和昂贵的专用数据集，维护成本高昂。

2.2.3 Multi-Tool Collaborative UI Parsing

2.2.3 多工具协同 UI 解析

Due to generalization limitations in both text-based and multimodal GUI Agents, researchers have increasingly adopted tool-based approaches to enhance screen information recognition. As shown in Figure 4, several specialized tools have emerged in this domain: SoM [52] combines Grounding DINO [84] and SAM [85] to identify and semantically tag screen objects, then integrates these annotations with GPT-4V for effective localization. Similarly, Omni-Parser [56] employs multiple specialized models—including object detection, OCR, and icon recognition-to parse screen elements independently before consolidating this information for GPT-4V comprehension. These approaches typically leverage a diverse toolkit including OCR systems, YOLO detectors, Grounding DINO, and IconNet for comprehensive GUI parsing. By using specialized tools to extract relevant information as prompt enhancements, these methods significantly reduce the perceptual burden on GUI Agents. However, this tool-assisted paradigm sacrifices end-to-end processing efficiency. The necessity to employ multiple recognition algorithms before perception introduces substantial computational overhead and resource requirements, creating a trade-off between enhanced accuracy and system performance.

鉴于基于文本和多模态 GUI 代理的泛化能力有限，研究者越来越多地采用基于工具的方法以增强屏幕信息识别。如图 4 所示，该领域涌现出多种专用工具:SoM[52] 结合 Grounding DINO[84] 和 SAM[85] 识别并语义标注屏幕对象，再将这些注释与 GPT-4V 整合实现有效定位。类似地，Omni-Parser[56] 采用多种专用模型——包括目标检测、OCR 和图标识别——独立解析屏幕元素，随后整合信息供 GPT-4V 理解。这些方法通常利用 OCR 系统、YOLO 检测器、Grounding DINO 和 IconNet 等多样化工具包进行全面 GUI 解析。通过使用专用工具提取相关信息作为提示增强，这些方法显著减轻了 GUI 代理的感知负担。然而，该工具辅助范式牺牲了端到端处理效率。多重识别算法的使用在感知前引入了大量计算开销和资源需求，形成了准确性提升与系统性能之间的权衡。

2.3 Exploration

2.3 探索

In recent years, the emergence of LLMs and MLLMs has endowed intelligent agents with robust natural language understanding and multimodal perception capabilities. These advancements have not only improved the efficiency of human-agent interactions but also significantly expanded the applicability of such agents in complex task scenarios. However, intelligent agents fundamentally operate within the constraints of their internal knowledge when making decisions. When confronted with heterogeneous user interface architectures and continuously evolving content, these systems frequently demonstrate performance limitations that fail to meet practical requirements. To address these limitations, researchers have increasingly explored graphical interface integration approaches, focusing on GUI Agents that can better perceive and navigate visual environments, thereby enhancing knowledge acquisition for more efficient task planning and decision-making, as shown in Figure 8

近年来，大型语言模型 (LLMs) 和多模态大型语言模型 (MLLMs) 的出现赋予智能代理强大的自然语言理解和多模态感知能力。这些进展不仅提升了人机交互的效率，也显著扩展了此类代理在复杂任务场景中的适用性。然而，智能代理在决策时本质上受限于其内部知识。当面对异构用户界面架构和不断变化的内容时，这些系统常表现出无法满足实际需求的性能瓶颈。为解决这些限制，研究者日益探索图形界面集成方法，聚焦于能够更好感知和导航视觉环境的 GUI 代理，从而增强知识获取，实现更高效的任务规划与决策，如图 8 所示。

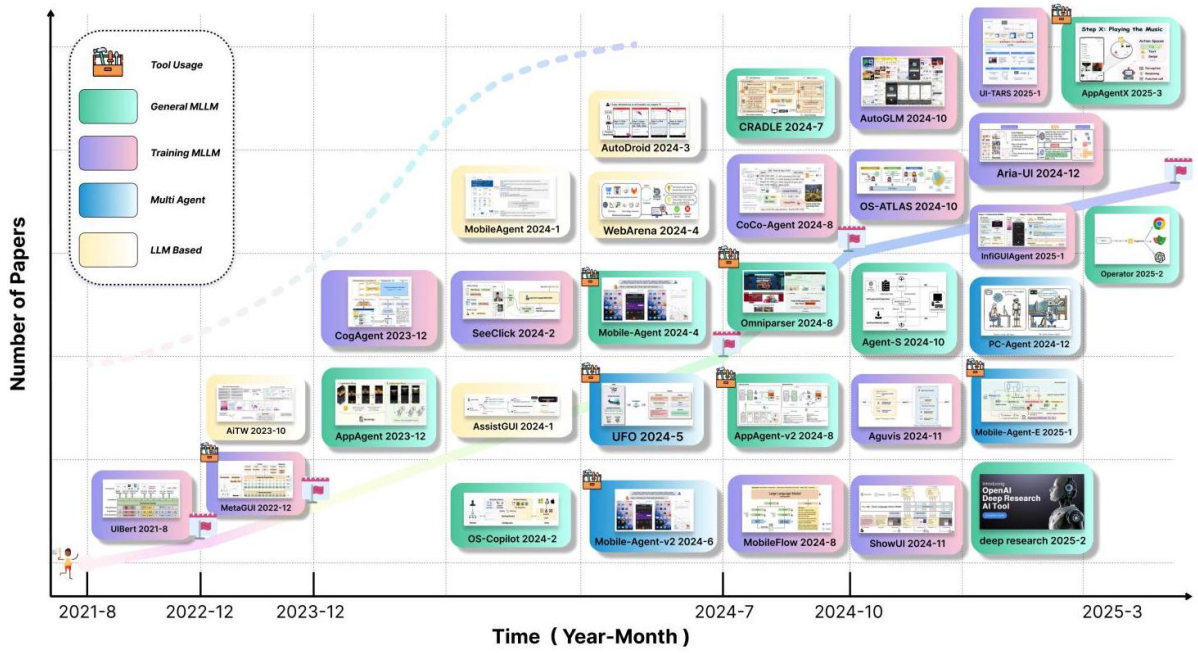

Fig. 4: Recent work summary and classification of GUI Agents (2021-2025). The visualization presents the evolution of approaches across four primary categories: General MLLM, Training MLLM, Multi-Agent, and LLM-Based methodologies. The horizontal axis tracks chronological development (Year-Month), while the vertical axis indicates publication volume.

图 4:2021-2025 年 GUI 代理的最新工作总结与分类。该可视化展示了四大主要类别方法的演进: 通用多模态大型语言模型、训练型多模态大型语言模型、多代理及基于大型语言模型的方法。横轴表示时间进展 (年-月)，纵轴表示发表数量。

In the following sections, we examine how GUI Agents process knowledge through various approaches. Section (§2.3.1) presents three exploration types (Internal, Historical, and External), while Section (§2.3.2) covers the operational aspects of Knowledge Construction, Storage, and Application.

在以下章节中，我们探讨 GUI 代理如何通过多种方法处理知识。第 §2.3.1 节介绍了三种探索类型 (内部探索、历史探索和外部探索)，而第 §2.3.2 节则涵盖了知识构建、存储和应用的操作方面。

2.3.1 Knowledge Acquisition

2.3.1 知识获取

When navigating new environments, humans leverage a variety of information sources to acquire knowledge and solve problems. Similarly, GUI Agents designed to interpret and interact with graphical interfaces must incorporate diverse information sources for effective operation. Through analysis of relevant research from the past three years, we identify three distinct approaches to knowledge acquisition:

在探索新环境时，人类利用多种信息源来获取知识和解决问题。同样，设计用于解释和交互图形界面的 GUI 代理必须整合多样的信息源以实现有效操作。通过分析过去三年的相关研究，我们识别出三种不同的知识获取方法:

- Internal Exploration: GUI Agents leverage built-in knowledge representations and reasoning capabilities.

- 内部探索:GUI 代理利用内置的知识表示和推理能力。

- Historical Exploration: GUI Agents analyze successful examples from past task trajectories.

- 历史探索:GUI 代理分析过去任务轨迹中的成功示例。

- External Exploration: GUI Agents retrieve supplementary knowledge and rules from external sources such as the internet.

- 外部探索:GUI 代理从互联网等外部来源检索补充知识和规则。

2.3.1.1 Internal Exploration focuses on gathering a range of information about user interface functionalities, page element purposes, and application overviews. For instance, AutoDroid [32] conducts offline random exploration of UI pages, summarizing element functionalities and page states to construct a knowledge base. Similarly, AppAgent [6] and AppAgent v2 [7] employ multimodal agent frameworks to analyze the roles of elements and actions used during interactions, documenting the resulting knowledge as reference materials. To further enhance the agent's understanding of user needs, MobA [33] utilizes LLMs to summarize user preferences based on daily activity trajectories, establishing a user memory. Additionally, it records application functionality descriptions and visited pages as procedural memory, aiding the agent in acquiring application-specific knowledge and improving generalization capabilities. Internal exploration enables GUI Agents to systematically build comprehensive knowledge representations by examining interface structures, analyzing element functionalities, and mapping application behaviors within their internal frameworks.

2.3.1.1 内部探索侧重于收集关于用户界面功能、页面元素用途和应用概览的多种信息。例如，AutoDroid [32] 通过离线随机探索 UI 页面，总结元素功能和页面状态以构建知识库。类似地，AppAgent [6] 和 AppAgent v2 [7] 采用多模态代理框架分析交互中元素和动作的角色，将所得知识记录为参考资料。为进一步增强代理对用户需求的理解，MobA [33] 利用大型语言模型 (LLMs) 基于日常活动轨迹总结用户偏好，建立用户记忆。此外，它还记录应用功能描述和访问页面作为程序记忆，帮助代理获取应用特定知识并提升泛化能力。内部探索使 GUI 代理能够通过检查界面结构、分析元素功能和映射应用行为，在其内部框架中系统地构建全面的知识表示。

2.3.1.2 Historical Exploration focuses on gathering a range of information about Historical Task Trajectories, Environmental Feedback, and Skill Repositories, which leverages past interactions and outcomes to inform current decision-making processes.

2.3.1.2 历史探索侧重于收集关于历史任务轨迹、环境反馈和技能库的多种信息，利用过去的交互和结果来指导当前的决策过程。

Environmental Feedback. Simliar to human short-term memory, environmental feedback refers to the temporary storage of information from the previous steps or the immediately preceding decision. This feedback is then applied to guide the agent's subsequent decision-making process. In Reflexion [60], the agent collects language-based feedback to improve a language agent by transforming binary or scalar environmental feedback into textual summaries. These summaries allow the agent to reflect on failed planning attempts, storing its reflections in an episodic memory buffer for inclusion in the context of subsequent decisions, thus enabling better planning. In OS-Copilot [40], the working memory gathers real-time execution feedback, which it uses to adjust the stored declarative and procedural memories, preventing the agent from repeating the same mistakes.

环境反馈。类似于人类的短期记忆，环境反馈指的是对前一步骤或紧接前一决策的信息的临时存储。该反馈随后用于指导代理的后续决策过程。在 Reflexion [60] 中，代理收集基于语言的反馈，通过将二元或标量环境反馈转化为文本摘要来改进语言代理。这些摘要使代理能够反思失败的规划尝试，并将反思存储在情景记忆缓冲区中，纳入后续决策的上下文，从而实现更优规划。在 OS-Copilot [40] 中，工作记忆收集实时执行反馈，用以调整存储的陈述性和程序性记忆，防止代理重复同样的错误。

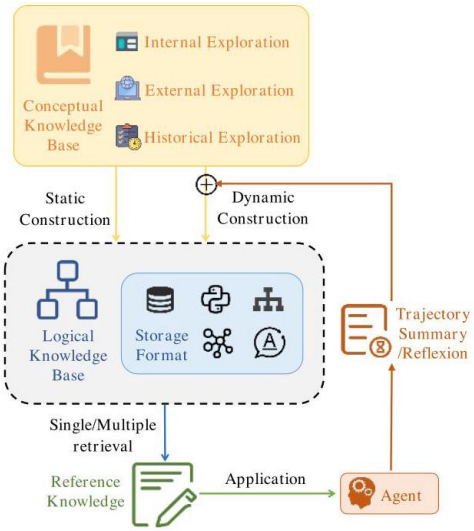

Fig. 5: Exploration of GUI Agents. The diagram illustrates the knowledge acquisition process through three exploration types (internal, external, and historical), showing how conceptual knowledge is constructed and stored in logical knowledge bases before being retrieved and applied by agents.

图 5:GUI 代理的探索。该图展示了通过三种探索类型 (内部、外部和历史) 进行知识获取的过程，说明了概念知识如何构建并存储于逻辑知识库中，随后被代理检索和应用。

Task Trajectories. Agents [59], Synapse [103], and RaDA 104 store successful trajectories collected during the training of agents on foundational tasks to supplement agent knowledge. Similarly, OS-Copilot [40] maintains three types of memory: declarative memory (semantic knowledge including past trajectories), procedural memory, and working memory. To address complex control tasks, the Agents [34] framework stores both narrative and episodic knowledge. Narrative knowledge summarizes complete task experiences for assisting in complex

task decomposition, while episodic knowledge provides hierarchical task strategies for subtask planning. Due to the sensitivity of input prompts and the context length limitations of large language model-based agents, Expel [105] and AutoGuide [106] extract knowledge from success and failure trajectories to generate state-dependent guidelines. These guidelines are used in real-time state to predict optimal actions. OdysseyAgent, trained through GUI Odyssey [35], is a multimodal agent for cross-application navigation. Given that task-related contexts often involve numerous screenshots and extended operational steps, the framework employs a historical replay module that compresses historical image annotations as inputs for decision-making. UFO [9] specializes in UI-focused tasks using a dual-agent framework. It records the decision history and execution outcomes of both agents to regulate interactions, achieving seamless integration with the Windows operating system.

任务轨迹。Agents [59]、Synapse [103] 和 RaDA [104] 存储在基础任务训练中收集的成功轨迹，以补充代理知识。类似地，OS-Copilot [40] 维护三种记忆: 陈述性记忆 (包括过去轨迹的语义知识)、程序性记忆和工作记忆。为应对复杂控制任务，Agents [34] 框架存储叙事知识和情景知识。叙事知识总结完整任务经验，辅助复杂任务分解；情景知识提供分层任务策略以支持子任务规划。鉴于基于大型语言模型的代理对输入提示敏感且上下文长度有限，Expel [105] 和 AutoGuide [106] 从成功与失败轨迹中提取知识，生成状态依赖的指导方针，用于实时状态下预测最优动作。通过 GUI Odyssey [35] 训练的 OdysseyAgent 是一种用于跨应用导航的多模态代理。鉴于任务相关上下文通常包含大量截图和长时间操作步骤，该框架采用历史回放模块，将历史图像注释压缩作为决策输入。UFO [9] 专注于 UI 任务，采用双代理框架，记录两代理的决策历史和执行结果以调节交互，实现与 Windows 操作系统的无缝集成。

Skill Repositories. Voyager 611, an embodied agent developed for Minecraft, maintains a growing skill repository that stores effective skill codes for tasks posed by an automated curriculum module, enabling continuous exploration of the game world. Similarly, GITM [107] maintains a text memory library of common plans for each encountered task, providing referential experiential knowledge. Task completion often follows a sequence governed by certain rules. KnowAgent [62] explores different task types, summarizing corresponding action and rule-based knowledge. This knowledge is used to fine-tune large models, enhancing their ability to understand and utilize task-relevant information.

技能库。Voyager 611 是一款为 Minecraft 开发的具身代理，维护一个不断增长的技能库，存储自动课程模块提出任务的有效技能代码，实现对游戏世界的持续探索。类似地，GITM [107] 维护一个文本记忆库，存储每个遇到任务的常用方案，提供参考性经验知识。任务完成通常遵循一定规则的序列。KnowAgent [62] 探索不同任务类型，总结相应的动作和基于规则的知识，用于微调大型模型，提升其理解和利用任务相关信息的能力。

2.3.1.3 External Exploration involves acquiring information from the external environment using tools such as search engines. In OS-Copilot [40], which emulates the cognitive architecture of the human brain, procedural memory constitutes a critical component of the system's long-term memory structure. This memory framework stores a comprehensive collection of facts and events gathered from the internet, user interactions, and operating system outputs to support the agent's decision-making processes. In GITM [107], researchers construct a comprehensive textual knowledge base drawing from sources such as the Minecraft Wiki and detailed crafting/smelting recipes. This external repository serves as an essential reference for the agent when performing complex task decomposition within the Minecraft environment. To improve decision-making efficiency and avoid rigid operational directives, Mobile-Agent [5] searches the internet to summarize standard operating procedures for various tasks, which are then used as blueprints for agent input commands. In Agents [34], when decomposing complex tasks, the

agent searches the web for similar categorized examples and classification criteria. This knowledge is combined with narrative memory, or historical knowledge, to provide a more comprehensive reference for decision-making.

2.3.1.3 外部探索涉及使用搜索引擎等工具从外部环境获取信息。在模拟人脑认知架构的 OS-Copilot [40] 中，程序性记忆构成系统长期记忆结构的关键部分。该记忆框架存储了从互联网、用户交互和操作系统输出收集的全面事实和事件集合，以支持代理的决策过程。在 GITM [107] 中，研究人员构建了一个综合的文本知识库，来源包括 Minecraft Wiki 及详细的合成/熔炼配方。该外部知识库作为代理在 Minecraft 环境中执行复杂任务分解的重要参考。为提高决策效率并避免僵化的操作指令，Mobile-Agent [5] 通过互联网搜索总结各种任务的标准操作流程，作为代理输入命令的蓝图。在 Agents [34] 中，分解复杂任务时，代理会搜索网络中类似分类的示例和分类标准，将这些知识与叙事记忆 (历史知识) 结合，为决策提供更全面的参考。

2.3.2 Knowledge Operation

2.3.2 知识操作

GUI Agents form conceptual knowledge bases through information exploration from various sources. This knowledge operation framework comprises three interconnected processes for functioning across environments like Android or Windows. The process begins with transforming concepts into logical repositories (§2.3.2.1), followed by establishing optimal structural organization for effective representation (§2.3.2.2), and concludes with implementing efficient retrieval mechanisms that balance access speed and planning effectiveness (§2.3.2.3). These integrated components enable agents to operationalize knowledge in practical applications.

GUI 代理通过从多种来源的信息探索形成概念知识库。该知识操作框架包含三个相互关联的过程，适用于 Android 或 Windows 等环境。过程始于将概念转化为逻辑存储库 (§2.3.2.1)，随后建立有效表示的最优结构组织 (§2.3.2.2)，最后实现平衡访问速度与规划效果的高效检索机制 (§2.3.2.3)。这些集成组件使代理能够在实际应用中实现知识的操作化。

2.3.2.1 Knowledge Construction is a crucial component in the knowledge processing pipeline. A well-designed knowledge construction process maximizes the extraction of valuable information, thereby significantly enhancing intelligent agent performance. Based on temporal characteristics, we categorize knowledge construction processes into two types: static construction and dynamic construction.

2.3.2.1 知识构建是知识处理流程中的关键环节。设计良好的知识构建过程最大化有价值信息的提取，从而显著提升智能代理的性能。基于时间特性，我们将知识构建过程分为两类: 静态构建和动态构建。

Static Construction. This approach involves offline exploration of GUI pages to gather information, as demonstrated by AutoDroid [32], which randomly explores GUI pages to generate transition graphs and, through a memory generator, traverses the graphs to create element function descriptions and page function summaries. Unlike Auto-Droid's random exploration, AppAgent [6] and AppAgent v2 [7] focus on task-driven exploration, guiding the agent to more efficiently operate specific elements. In this process, the agent learns the functional relationships between different UI elements and actions during interactions, while also supplementing the agent's knowledge with complex functions observed from manual operations. MobileAgent [50] further performs offline knowledge extraction on task-driven path trajectories to derive more abstract knowledge of standard operating procedures.

Expel [105] and AutoGu-ide [106] extract knowledge from historical task trajectories to generate state-dependent guidelines. First, they construct a dataset of successful and failed trajectory pairs from the collected task decision data. They then extract trajectories before deviations occur in the paired dataset and use state summarization by LLM to generate corresponding guidelines. KnowAgent [62] for a specific domain task, performs large-scale offline knowledge construction, where GPT-4 generates an initial knowledge base of task exploration actions and action rules, which is then manually optimized. Due to the lack of real-time interaction and feedback mechanisms, static knowledge bases often struggle to capture contextual information in complex scenarios, leading them less adaptable to dynamically changing GUI environments. Recognizing this limitation, researchers have shifted toward dynamic construction methods.

静态构建。该方法涉及离线探索 GUI 页面以收集信息，如 AutoDroid [32] 所示，其随机探索 GUI 页面生成转换图，并通过记忆生成器遍历图谱，创建元素功能描述和页面功能摘要。与 Auto-Droid 的随机探索不同，AppAgent [6] 和 AppAgent v2 [7] 侧重于任务驱动的探索，引导代理更高效地操作特定元素。在此过程中，代理学习不同 UI 元素与操作间的功能关系，同时通过手动操作观察补充复杂功能知识。MobileAgent [50] 进一步对任务驱动路径轨迹进行离线知识提取，推导标准操作流程的更抽象知识。Expel [105] 和 AutoGuide [106] 从历史任务轨迹中提取知识，生成状态依赖的指导方针。首先，他们构建成功与失败轨迹对的数据集，从中提取偏离发生前的轨迹，并利用大型语言模型 (LLM) 进行状态总结，生成相应指导。KnowAgent [62] 针对特定领域任务，执行大规模离线知识构建，由 GPT-4 生成任务探索动作及规则的初始知识库，随后进行人工优化。由于缺乏实时交互和反馈机制，静态知识库常难以捕捉复杂场景中的上下文信息，导致其对动态变化的 GUI 环境适应性较差。认识到这一局限，研究者转向动态构建方法。

Dynamic Construction. MobA [33] implements three types of memory that are initially configured by human experts and continuously updated in real time during interactions. WKM [63] serves as a world knowledge model to assist GUI Agents, which extracts expert trajectories from manually labeled datasets, comparing preferences with trajectories generating by agent. These preferences are input into the agent to summarize task knowledge, and each step is associated with a status summary knowledge based on expert trajectories. OS-Copilot [40] utilizes procedural memory that is initially custom-defined by the user. For subsequent tasks, the agent retrieves applicable tools from memory. When memory lacks suitable tools, the tool generator activates to create and execute new tools, collecting feedback until reaching a termination condition. Upon task completion, the reflection module evaluates each tool's utility and determines whether to update the memory base accordingly. Agents [34] includes both initial memory construction and continuous memory updates. In the initial phase, tasks are generated based on a dataset into two categories: environment-independent and environment-aware tasks, which are then explored by an agent supplemented with network knowledge. The agent collects experiences from complete tasks and subtasks, continually updating the knowledge base as it interacts with new tasks. Dynamic construction enhances adaptability and contextual understanding by enabling real-time knowledge refinement based on interactive feedback, addressing the limitations of static approaches.

动态构建。MobA [33] 实现了三种记忆类型, 初始由专家配置, 并在交互过程中实时持续更新。WKM [63] 作为世界知识模型辅助 GUI 代理，从人工标注数据集中提取专家轨迹，将偏好与代理生成轨迹进行比较。这些偏好输入代理以总结任务知识，每一步关联基于专家轨迹的状态总结知识。OS-Copilot [40] 利用用户自定义的程序性记忆，代理在后续任务中从记忆中检索适用工具。当记忆中缺乏合适工具时，工具生成器启动，创建并执行新工具，收集反馈直至满足终止条件。任务完成后，反思模块评估各工具效用，决定是否更新记忆库。Agents [34] 包括初始记忆构建和持续记忆更新。初始阶段，基于数据集生成环境无关和环境感知两类任务，代理结合网络知识进行探索。代理从完整任务及子任务中收集经验，随着新任务交互不断更新知识库。动态构建通过基于交互反馈的实时知识精炼，增强适应性和上下文理解，弥补了静态方法的不足。

2.3.2.2 Knowledge Storage The format of knowledge storage media constitutes a critical dimension in exploration module research, with formats ranging from natural language to structured databases. Each storage paradigm offers unique advantages tailored to specific application contexts, reflecting the diverse approaches to knowledge representation in autonomous systems.

2.3.2.2 知识存储知识存储介质的格式构成了探索模块研究的关键维度, 格式涵盖从自然语言到结构化数据库。每种存储范式都提供了针对特定应用场景的独特优势，反映了自主系统中知识表示的多样化方法。

Natural Language. serves as a foundational knowledge storage medium that enables intuitive representation and human-interpretable storage of diverse information types. AppAgent [6] and AppAgent v2 [7] represent knowledge through natural language, which is then stored in documents. Reflexion [60] and Voyager [61] both leverage natural language for knowledge representation, with the former maintaining experiential feedback within a sliding window mechanism while the latter encodes Minecraft skills directly in memory, enabling efficient skill retrieval and utilization during gameplay.

自然语言作为基础的知识存储媒介, 能够直观地表示并以人类可理解的方式存储多种信息类型。AppAgent [6] 和 AppAgent v2 [7] 通过自然语言表示知识, 随后存储于文档中。Reflexion [60] 和 Voyager [61] 均利用自然语言进行知识表示，前者通过滑动窗口机制维护体验反馈，后者则将 Minecraft 技能直接编码于内存中，实现游戏过程中技能的高效检索与应用。

Database. provides systematic organization and efficient retrieval mechanisms for complex agent knowledge, enabling structured storage with sophisticated indexing capabilities. AutoDroid [32] implements a dual-table knowledge architecture comprising simulation task tables with UI interaction element function descriptions and user page function tables that document current page functionalities. The Agents framework [59] incorporates memory components from Zhou et al. [108], utilizing VectorDB for efficient knowledge storage, while Agashe et al. [34] distinguishes between narrative memory keyed by query tasks and situational memory indexed by both query tasks and contextual subtask information within their database structure.

数据库提供了复杂代理知识的系统化组织和高效检索机制，实现了具有复杂索引功能的结构化存储。AutoDroid [32] 实现了双表知识架构，包括带有 UI 交互元素功能描述的仿真任务表和记录当前页面功能的用户页面功能表。Agents 框架 [59] 引入了 Zhou 等人 [108] 的记忆组件，利用 VectorDB 实现高效知识存储，而 Agashe 等人 [34] 在其数据库结构中区分了以查询任务为键的叙事记忆和以查询任务及上下文子任务信息为索引的情境记忆。

Data Structures. offer versatile frameworks for organizing complex relationships between knowledge elements, enabling efficient representation of hierarchical, associative, and sequential information for GUI Agents. The task memory of MobA [33] is based on the ICE [109] strategy, where tasks and subtasks are organized into a hierarchical tree structure. This structure is suitable for sequences of ordered subtasks and captures detailed information about each task, including task status, environmental context, and relevant subtasks. Expel [105] and AutoGuide [106] employ dictionary data structures with key-value pairs to efficiently represent the one-to-one correspondence between states and guidelines. KnowAgent [62] adopts a graph structure for its knowledge base to represent complex action relationships and sequences that mirror the intricacy of social networks. Within this framework, actions function as nodes (containing action names and definitions) while rules serve as connecting edges (encoding rule descriptions), effectively capturing the interdependencies between different actions in the system. The state knowledge in WKM [63] is also stored in a graph structure, with nodes representing actions in expert trajectories and edges representing state knowledge between sequential actions. GITM [107] stores the action lists of subgoals in a hierarchical tree structure, clearly capturing the relationship between goals and corresponding plans.

数据结构为组织知识元素间复杂关系提供了多功能框架，支持对 GUI 代理的层级、关联和序列信息的高效表示。MobA [33] 的任务记忆基于 ICE [109] 策略，将任务与子任务组织成层级树结构，适用于有序子任务序列，并捕捉每个任务的详细信息，包括任务状态、环境上下文及相关子任务。Expel [105] 和 AutoGuide [106] 采用键值对字典数据结构，高效表示状态与指导方针的一一对应关系。KnowAgent [62] 采用图结构作为知识库，表示复杂的动作关系和序列，反映社交网络的复杂性。在该框架中，动作作为节点 (包含动作名称和定义)，规则作为连接边 (编码规则描述)，有效捕捉系统中不同动作间的相互依赖。WKM [63] 中的状态知识同样存储于图结构，节点代表专家轨迹中的动作，边表示连续动作间的状态知识。GITM [107] 将子目标的动作列表存储于层级树结构，清晰捕捉目标与对应计划的关系。

Code. Executable representations offer a direct encoding of procedural knowledge, enabling agents to perform complex operations through functional libraries. The procedural memory involved in OS-Copilot [40] consists of a series of manually created tool libraries stored in dual formats as both API services and Python files. Voyager [61] stores skills as values within its architecture, forming a comprehensive skill knowledge base that facilitates direct execution of learned capabilities.

代码可执行表示直接编码程序性知识，使代理能够通过功能库执行复杂操作。OS-Copilot [40] 中的程序性记忆由一系列手工创建的工具库组成, 双重格式存储为 API 服务和 Python 文件。Voyager [61] 将技能作为值存储于其架构中，形成全面的技能知识库，便于直接执行已学能力。

2.3.2.3 Knowledge Application The purpose of memory retrieval is to extract meaningful information from memory to enhance GUI Agents's actions, such as utilizing previously successful actions to achieve similar goals. The critical challenge in memory retrieval centers on identifying and extracting valuable information from the agent's historical action repository. This process can be categorized based on retrieval scope and methodology, with approaches ranging from targeted single retrieval to comprehensive multiple extraction strategies.

2.3.2.3 知识应用记忆检索的目的是从记忆中提取有意义的信息，以增强 GUI 代理的行为，例如利用先前成功的动作实现类似目标。记忆检索的关键挑战在于识别并提取代理历史动作库中的有价值信息。该过程可根据检索范围和方法分类，涵盖从针对性单次检索到全面多次提取的策略。

Single Retrieval. AppAgent [6], AppAgent v2 [7], and Agents [59] retrieve knowledge based on task rele-

vance, serving as additional input to assist the agent. AutoDroid [32] employs the embedding model Instructor-XL [110] to map knowledge and task descriptions, utilizing a similarity function (sim) to compare both and retrieve knowledge for agent planning. The retrieved functional descriptions are applied by adding new onclick attributes to the corresponding HTML framework UI elements on the current page. Agashe et al. [34] retrieve knowledge from the knowledge database based on task descriptions and hierarchical subtask information. Voyager [61] utilizes GPT-3.5 to generate environmental feedback based on task descriptions and retrieves the top-5 skill codes from the skill database through feedback embedding. KnowAgent [62], which operates on text-processing LLMs, initially retrieves the relevant knowledge stored in a graph and converts it into text format. This serves dual purposes: during the self-learning process, each generated path is filtered and merged to fine-tune the agent, and subsequently provided directly as a prompt in downstream tasks. Qiao et al. [63] utilize task knowledge and state knowledge to train the WKM model, which simulates the human process of knowledge building by progressing from general overviews to specific details. The WKM initially assists the agent in establishing a general planning path and subsequently generates the current state in real-time during execution. This state functions as a key to retrieve knowledge from the database, identifying n possible next actions and selecting the one with the highest probability based on the agent's predicted action probabilities. Despite these advances, single retrieval methods may present limitations when knowledge bases store multi-source information. Such approaches can be one-sided with poor adaptability, potentially missing critical information and leading to significant deviations in decision-making paths. To address these limitations, multiple retrieval approaches have been proposed in recent research.

单次检索。AppAgent [6]、AppAgent v2 [7] 和 Agents [59] 根据任务相关性检索知识，作为辅助输入帮助代理。AutoDroid [32] 采用嵌入模型 Instructor-XL [110] 将知识和任务描述映射，通过相似度函数 (sim) 比较两者并检索知识以辅助代理规划。检索到的功能描述通过向当前页面对应的 HTML 框架 UI 元素添加新的 onclick 属性来应用。Agashe 等人 [34] 根据任务描述和层级子任务信息从知识库中检索知识。Voyager [61] 利用 GPT-3.5 根据任务描述生成环境反馈，并通过反馈嵌入从技能库中检索排名前五的技能代码。KnowAgent [62] 基于文本处理的大型语言模型 (LLMs) 运行，首先检索存储在图中的相关知识并转换为文本格式。这具有双重作用: 在自学习过程中，每条生成路径被筛选和合并以微调代理，随后直接作为下游任务的提示。Qiao 等人 [63] 利用任务知识和状态知识训练 WKM 模型，模拟人类从总体概览到具体细节的知识构建过程。WKM 初期协助代理建立总体规划路径，随后在执行过程中实时生成当前状态。该状态作为从数据库检索知识的关键，识别 n 可能的下一步动作，并基于代理预测的动作概率选择概率最高的动作。尽管取得这些进展，单次检索方法在知识库存储多源信息时可能存在局限性。这类方法可能片面且适应性差，可能遗漏关键信息，导致决策路径出现重大偏差。为解决这些限制，近期研究提出了多次检索方法。

Multiple Retrieval. OS-Copilot [40] retrieves different stored procedural memory through two distinct mechanisms: API calls via POST requests and Python file information that matches specific calls. AutoGuide [106] integrates key-value pair knowledge during the agent's decision-making process, where the state summarization module first summarizes the current state in real-time for state matching, followed by the guideline selection module for further refinement when more than k matches are identified. MobA [33] implements two complementary retrieval methods tailored to diverse task requirements. Their relation-based retrieval targets action modules by extracting task descriptions from both the previous action node and its parent node within the tree-structured memory, thereby enhancing the agent's decision-making capacity for subsequent actions. Concurrently, their content-based retrieval leverages cosine similarity to search through historical task experiences, which facilitates the hierarchical decomposition of complex tasks.

多次检索。OS-Copilot [40] 通过两种不同机制检索不同存储的程序记忆: 通过 POST 请求的 API 调用和匹配特定调用的 Python 文件信息。AutoGuide [106] 在代理决策过程中整合键值对知识，其中状态总结模块首先实时总结当前状态以进行状态匹配，随后在识别出超过 k 个匹配时由指导方针选择模块进一步精炼。MobA [33] 实施了两种互补的检索方法以适应多样化的任务需求。其基于关系的检索通过提取树状记忆中前一动作节点及其父节点的任务描述，针对动作模块进行检索，从而增强代理对后续动作的决策能力。同时，其基于内容的检索利用余弦相似度搜索历史任务经验，促进复杂任务的层级分解。

2.4 Reasoning and Planning

2.4 推理与规划

Planning in GUI Agents necessitates sophisticated reasoning capabilities to effectively navigate complex interfaces and accomplish user goals. This section examines the theoretical underpinnings and practical implementations of planning mechanisms. We begin by analyzing the reasoning frameworks employed in contemporary LLMs (§2.4.1) and MLLMs (§2.4.2), establishing the foundation for understanding advanced planning techniques. We then explore emerging reasoning architectures in state-of-the-art o1-like models that integrate multi-step reasoning and complex decision trees (§2.4.3). Subsequently, we provide a comprehensive analysis of GUI Agents-specific planning architectures (§2.4.4), focusing on planner design, verification methodologies, and feedback integration systems that collectively enable robust task execution in graphical user environments.

GUI 代理中的规划需要复杂的推理能力,以有效导航复杂界面并完成用户目标。本节探讨规划机制的理论基础与实际实现。首先分析当代大型语言模型 (LLMs) (§2.4.1) 和多模态大型语言模型 (MLLMs) (§2.4.2) 中采用的推理框架，为理解先进规划技术奠定基础。随后探讨集成多步推理和复杂决策树的前沿 o1 类模型中的新兴推理架构 (§2.4.3)。最后，全面分析 GUI 代理特有的规划架构 (§2.4.4)，重点关注规划器设计、验证方法及反馈集成系统，这些共同支持图形用户环境中的稳健任务执行。

2.4.1 LLMs Reasoning

2.4.1 大型语言模型 (LLMs) 推理

In recent years, LLMs have demonstrated outstanding performance in context modeling, which has significantly contributed to their exceptional reasoning abilities in natural language processing tasks [111]. Inspired by this, the Chain-of-Thought (CoT) [36] prompting method, which introduces intermediate reasoning steps to guide LLMs through a step-by-step reasoning process, has successfully addressed multistep reasoning problems [37]. The intermediate reasoning steps in CoT provide a more transparent reasoning pathway, which aids in the interpretability and evaluation of LLMs 112]. This method not only enhances the model's reasoning capabilities but also provides significant insights for Agent planning in multi-step reasoning tasks. In the following, we will introduce several common variants of CoT prompting.

近年来，大型语言模型 (LLMs) 在上下文建模方面表现出色，这极大地促进了其在自然语言处理任务中的卓越推理能力 [111]。受此启发，链式思维 (Chain-of-Thought，CoT)[36] 提示方法引入了中间推理步骤，引导 LLMs 通过逐步推理过程，成功解决了多步推理问题 [37]。CoT 中的中间推理步骤提供了更透明的推理路径，有助于 LLMs 的可解释性和评估 [112]。该方法不仅增强了模型的推理能力，还为多步推理任务中的智能体规划提供了重要见解。以下将介绍几种常见的 CoT 提示变体。

CoT. is a chain structure that breaks down complex tasks into a series of simpler steps, executed sequentially. The CoT structure can be represented as $\langle$ input , thought $_1$, thought $_2, \ldots,$ thought $_n$, output $\rangle$, where input refers to the task received by the model, thought represents the intermediate reasoning steps, n denotes the number of such steps, and output is the final answer or solution. From the perspective of sample prompts, CoT can be categorized into two types: (1) zero-shot CoT, which prompts reasoning using a phrase like "Let's think step by step" [37], and (2) few-shot CoT, where a small number of sample prompts guide the model through incremental reasoning [36]. For example, Yuan et al. [113] utilized a preference-guided reverse reasoning strategy to generate sample templates that enhance the logical reasoning ability of LLMs.

CoT 是一种链式结构，将复杂任务分解为一系列简单步骤，按顺序执行。CoT 结构可表示为 $\langle$ input , thought $_1$, thought $_2, \ldots,$ thought $_n$, output $\rangle$ ，其中 input 指模型接收的任务，thought 表示中间推理步骤，n 表示此类步骤的数量，output 为最终答案或解决方案。从示例提示的角度看，CoT 可分为两类:(1) 零样本 CoT，使用“让我们一步步思考”等短语提示推理 [37]；(2) 少样本 CoT，通过少量示例提示引导模型逐步推理 [36]。例如，Yuan 等 [113] 采用偏好引导的逆向推理策略生成示例模板，提升了 LLMs 的逻辑推理能力。

ToT. Tree-of-Thoughts (ToT) [65] extends chain-based reasoning by using a tree structure, generating multiple reasoning thoughts at each step of the reasoning process and conducting self-evaluation to prune and plan for the optimal path. Recent studies have further extended the tree structure by combining Monte Carlo Tree Search (MCTS) 114] with LLMs, further enhancing the model's reasoning ability [66, 115-117]. For example, ReST-MCTS* integrates MCTS with reinforcement self-training [66], and SC-MCTS 116 improves the accuracy and speed of LLMs in complex reasoning tasks through enhanced reward models, node selection strategies, and backpropagation methods. Additionally, RAP [118] constructs a world model and uses MCTS to effectively balance exploration and exploitation. LATS [117] follows the ReAct framework and applies MCTS to LM Agents, achieving a more detailed and adaptive problem-solving mechanism through a collaborative process of search, interaction, and reflection.

ToT。思维树 (Tree-of-Thoughts，ToT)[65] 通过树结构扩展了链式推理，在推理过程的每一步生成多个推理思路，并进行自我评估以修剪和规划最优路径。近期研究进一步结合蒙特卡洛树搜索 (Monte Carlo Tree Search, MCTS)[114] 与 LLMs, 进一步提升模型推理能力 [66, 115-117]。例如, ReST-MCTS* 结合了 MCTS 与强化自训练 [66]，SC-MCTS[116] 通过改进奖励模型、节点选择策略和反向传播方法, 提高了 LLMs 在复杂推理任务中的准确性和速度。此外, RAP[118] 构建了世界模型, 利用 MCTS 有效平衡探索与利用。LATS[117] 遵循 ReAct 框架，将 MCTS 应用于语言模型智能体，通过搜索、交互和反思的协作过程，实现更细致和自适应的问题解决机制。

GoT. Graph of Thoughts [67] extends the traditional tree-based reasoning structure by adopting a graph structure by utilizing a graph structure that introduces cycles and many-to-one connections. This allows for more effective modeling of subproblem aggregation and self-assessment.

GoT。思维图 (Graph of Thoughts)[67] 通过采用图结构扩展了传统的树状推理结构，引入了循环和多对一连接，从而更有效地建模子问题聚合和自我评估。

ReAct. Reasoning and Acting (ReAct) [38] builds on chain-based reasoning by adding an action space $\widehat{A} = \mathcal{A} \cup \mathcal{L}$ to facilitate the synergy between reasoning and actions, where L represents the reasoning trajectory and A is the set of allowable actions. Specifically, the agent first generates a thought based on the assigned task, then executes an action based on this thought, observes the result, and iterates the process until the task is completed.

ReAct。推理与行动 (Reasoning and Acting，ReAct)[38] 基于链式推理，增加了动作空间 $\widehat{A} = \mathcal{A} \cup \mathcal{L}$，促进推理与行动的协同，其中 L 表示推理轨迹，A 为允许的动作集合。具体而言，智能体首先基于分配的任务生成思路，然后根据该思路执行动作，观察结果，并迭代该过程直至任务完成。

As the reasoning capabilities of LLMs become increasingly complex, it is becoming more important and necessary to effectively evaluate their complex reasoning paths. Some research, such as Self-Consistency [64, 117], Best-of-N [119- 121, Process Supervision Reward Model (PRM) [116, 121]- 123], and Outcome Supervision Reward Model (ORM) [124- 126], aims to improve the reasoning capabilities of large language models by evaluating intermediate reasoning steps or final outcomes. Meanwhile, some research solves complex problems by decomposing them into simpler subproblems [116, 127-130]. For example, Huang et al. [131] decompose questions into a directed acyclic graph, and use the Question Decomposition Meaning Representation (QDMR) prompting method to perform step-by-step reasoning based on the dependency relationships between nodes in the graph. In addition, some research suggests that LLMs are capable of self-improvement during the reasoning process through self-refinement [66, 130, 132-138]. For example, Quiet-STaR [136] uses a method in which the language model learns to generate a reasoning chain for each token.

随着 LLMs 推理能力日益复杂，有效评估其复杂推理路径变得愈发重要和必要。一些研究如自洽性 (Self-Consistency)[64, 117]、Best-of-N[119-121]、过程监督奖励模型 (Process Supervision Reward Model，PRM)[116, 121-123] 和结果监督奖励模型 (Outcome Supervision Reward Model，ORM)[124-126]，旨在通过评估中间推理步骤或最终结果提升大型语言模型的推理能力。同时，一些研究通过将复杂问题分解为更简单的子问题来解决 [116, 127-130]。例如，Huang 等 [131] 将问题分解为有向无环图，并使用问题分解语义表示 (Question Decomposition Meaning Representation，QDMR) 提示方法，基于图中节点的依赖关系逐步推理。此外，有研究表明 LLMs 能够通过自我优化在推理过程中实现自我提升 [66, 130, 132-138]。例如，Quiet-STaR[136] 采用一种方法，使语言模型学习为每个标记生成推理链。

2.4.2 Multimodal LLMs Reasoning

2.4.2 多模态大型语言模型推理

In the context of MLLMs, chain-of-thought reasoning is not confined to natural language but can also integrate and analyze other perceptual information such as images, videos, and audio, thereby extending textual chain-of-thought reasoning to multimodal chain-of-thought reasoning [37, 139, 140]. Lu et al. [141] proposed the ScienceQA dataset, laying the foundation for the subsequent introduction of the concept of Multimodal Chain-of-Thought reasoning (MCoT) [139]. To further enhance the reasoning capabilities of MCoT, researchers have explored various approaches. KAM-CoT [142] enhances the reasoning ability of MCoT by integrating chain-

of-thought, knowledge graphs, and multimodal information. CCoT [143] utilizes scene graph representations as prompts to encourage LLMs to generate structured descriptions as reasoning steps. Moreover, some works such as MM-REACT [39], VisCoT [144], and VoCoT [145] further integrate visual expert capabilities, enhancing multimodal reasoning and action capabilities. For example, VoCoT [145] represents object concepts visually through multimodal cross-alignment, thereby enabling an object-oriented chain-of-thought reasoning process that is guided by multi-step visual support. Similarly, CoCoT [146] leverages information across multiple images to perform cross-image analysis. To mitigate potential errors and hallucinations during the reasoning process, DDCoT [147] employs negative space cues to reveal uncertainties in subproblem decomposition. Meanwhile, TextCoT [148] utilizes the image description capabilities of MLLMs to grasp the global context of images and leverages localization abilities to examine local text regions. MC-CoT [149] employs a voting mechanism similar to CoT-SC, but unlike CoT-SC, it introduces voting in both the generation reasoning and answer reasoning stages, thereby enhancing the robustness of the LMM's reasoning. CogCoM [150] simulates human visual reasoning processes, inferring correct answers through incremental manipulation steps. WoT [151] introduces an innovative mechanism that converts intermediate reasoning step results into an image and then further reasons about the image using the visual input capabilities of MLLMs. In specific application domains, Wei et al. [152] applied MCoT to the medical field, improving reasoning and information extraction effectiveness by integrating medical knowledge and task-specific guidance. FOCUS [26] proposed a dual-system framework based on MLLMs to locate GUI elements, which can generate explicit multi-stage CoT to enhance grounding capability.

在多模态大语言模型 (MLLMs) 的背景下，链式思维推理不仅限于自然语言，还可以整合和分析图像、视频和音频等其他感知信息，从而将文本链式思维推理扩展为多模态链式思维推理 (Multimodal Chain-of-Thought reasoning，MCoT)[37, 139, 140]。Lu 等人 [141] 提出了 ScienceQA 数据集，为随后引入多模态链式思维推理 (MCoT) 概念奠定了基础 [139]。为了进一步提升 MCoT 的推理能力，研究者们探索了多种方法。KAM-CoT [142] 通过整合链式思维、知识图谱和多模态信息增强了 MCoT 的推理能力。CCoT [143] 利用场景图表示作为提示，鼓励大语言模型生成结构化描述作为推理步骤。此外，一些工作如 MM-REACT [39]、VisCoT [144] 和 VoCoT [145] 进一步整合了视觉专家能力，提升了多模态推理和行动能力。例如，VoCoT [145] 通过多模态跨对齐以视觉方式表示对象概念，从而实现了由多步视觉支持引导的面向对象的链式思维推理过程。类似地，CoCoT [146] 利用多张图像的信息进行跨图像分析。为减轻推理过程中的潜在错误和幻觉，DDCoT [147] 采用负空间线索揭示子问题分解中的不确定性。与此同时，TextCoT [148] 利用 MLLMs 的图像描述能力把握图像的全局语境，并利用定位能力检查局部文本区域。MC-CoT [149] 采用类似 CoT-SC 的投票机制，但不同于 CoT-SC，它在生成推理和答案推理阶段均引入投票，从而增强了大多模态模型 (LMM) 的推理鲁棒性。CogCoM [150] 模拟人类视觉推理过程，通过增量操作步骤推断正确答案。WoT [151] 引入了一种创新机制，将中间推理步骤结果转换为图像，然后利用 MLLMs 的视觉输入能力对该图像进行进一步推理。在特定应用领域，Wei 等人 [152] 将 MCoT 应用于医疗领域，通过整合医学知识和任务特定指导，提高了推理和信息提取的效果。FOCUS [26] 提出了基于 MLLMs 的双系统框架用于定位图形用户界面 (GUI) 元素，能够生成显式的多阶段链式思维以增强定位能力。

Furthermore, in the Text-to-Speech (TTS) domain, there are also works related to chain-of-thought [153-155]. RALL-E [153] enhances the robustness of LLM-based TTS by decomposing tasks into simpler steps using CoT. Meanwhile, Hu et al. [155] enhanced the performance of automatic speech translation (AST) by using the automatic speech recognition(ASR)-transcribed text generated by a speech encoder together with the encoded speech as CoT prompts. In addition, Gong et al. [154] improved the translation quality and parameter efficiency of expressive speech-to-speech translation (S2ST) by decomposing complex source-to-target speech mappings into intermediate generation steps through chain-of-thought prompts.

此外，在文本转语音 (TTS) 领域，也有与链式思维相关的工作 [153-155]。RALL-E [153] 通过使用链式思维将任务分解为更简单的步骤，增强了基于大语言模型的 TTS 的鲁棒性。与此同时，Hu 等人 [155] 通过将语音编码器生成的自动语音识别 (ASR) 转录文本与编码语音一起作为链式思维提示，提升了自动语音翻译 (AST) 的性能。此外，Gong 等人 [154] 通过链式思维提示将复杂的源语音到目标语音映射分解为中间生成步骤，提升了富有表现力的语音到语音翻译 (S2ST) 的翻译质量和参数效率。

2.4.3 o1-likes Reasoning Models

2.4.3 o1-likes 推理模型

Since the release of OpenAI's o1, researchers have been dedicated to replicating its exceptional reasoning capabilities. The o1 model demonstrates superior reasoning performance by engaging in deep thinking before providing answers. As a significant breakthrough in this field, DeepSeek-R1 [28] stands as a representative model for replicating o1, successfully reproducing similar capabilities through reinforcement learning and verifiable rule-based rewards. Specifically, it employs the Group Relative Policy Optimization (GRPO) [156] algorithm instead of the traditional PPO algorithm 157 for training, thereby achieving an "aha moment" (a phenomenon where the model reevaluates the answer and reflects previous reasoning path. Inspired by DeepSeek-R1's success, a wave of R1 replication efforts has emerged within the open-source community. These replication efforts primarily focus on two directions: text-based models and multimodal reasoning models. In terms of text-based models, Logic-RL [158] successfully reproduced the "aha moment" using a model of only 7B parameters, demonstrating that smaller-scale models can also acquire powerful reasoning capabilities through reinforcement learning. Meanwhile, R1-Searcher [159] innovatively proposed a two-stage reinforcement learning approach enabling LLMs to autonomously invoke external retrieval systems during reasoning, a method that effectively overcomes the inherent knowledge limitations of models. On the other hand, in the field of multimodal reasoning, despite the significant impact of DeepSeek-R1's achievements, similar explorations in the multimodal domain remain relatively limited. Vision-R1 [160] introduced a visual reinforcement fine-tuning method, designing specific reward functions for different visual tasks (such as IoU rewards and classification rewards) and adopting GRPO policy parameter updates, an approach that significantly enhances model performance in few-shot learning and visual reasoning tasks. LMM-R1 [161] enhances the reasoning capabilities of a 3B-scale multimodal model through a two-stage rule-reward reinforcement learning framework, which extends the OpenRLHF framework to support multimodal model training, thus enabling small models to demonstrate powerful reasoning and adaptability in complex tasks.

自 OpenAI 发布 o1 以来，研究人员一直致力于复制其卓越的推理能力。o1 模型通过在给出答案前进行深度思考，展现出优越的推理表现。作为该领域的重要突破，DeepSeek-R1 [28] 作为复制 o1 的代表性模型，通过强化学习和可验证的基于规则的奖励成功复现了类似能力。具体而言，它采用了群体相对策略优化 (Group Relative Policy Optimization, GRPO)[156] 算法，替代了传统的 PPO 算法 [157] 进行训练，从而实现了“顿悟时刻”(模型重新评估答案并反思先前推理路径的现象)。受 DeepSeek-R1 成功的启发，开源社区掀起了一波 R1 复制热潮。这些复制工作主要集中在两个方向: 基于文本的模型和多模态推理模型。在基于文本的模型方面，Logic-RL [158] 成功利用仅 7B 参数的模型复现了“顿悟时刻”，证明了小规模模型也能通过强化学习获得强大的推理能力。同时，R1-Searcher [159] 创新性地提出了两阶段强化学习方法，使大型语言模型 (LLMs) 能够在推理过程中自主调用外部检索系统，有效克服了模型固有的知识局限性。另一方面，在多模态推理领域，尽管 DeepSeek-R1 的成果影响深远，但多模态领域的类似探索仍较为有限。Vision-R1 [160] 引入了视觉强化微调方法，为不同视觉任务 (如 IoU 奖励和分类奖励) 设计了特定的奖励函数，并采用 GRPO 策略参数更新，该方法显著提升了模型在少样本学习和视觉推理任务中的表现。LMM-R1 [161] 通过两阶段规则奖励强化学习框架增强了 3B 规模多模态模型的推理能力，该框架扩展了 OpenRLHF 框架以支持多模态模型训练，从而使小模型在复杂任务中展现出强大的推理和适应能力。

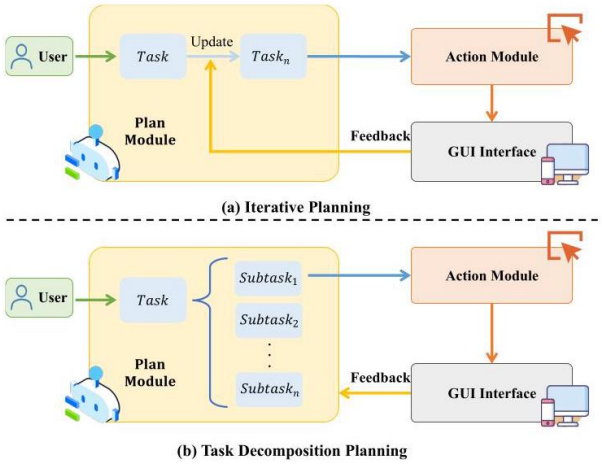

Fig. 6: Different task planning strategies for GUI agents: (a) iterative planning, and (b) task decomposition planning.

图 6:GUI 代理的不同任务规划策略:(a) 迭代规划，(b) 任务分解规划。

2.4.4 Task Planning

2.4.4 任务规划

Within the components of GUI Agents, the planning capability is of paramount importance, which assists the GUI Agents in task planning, task verification, and providing feedback. During the planning phase, GUI Agents typically follow the CoT paradigm [6, 7, 9, 34, 40]. In the planning phase, the reasoning process often adopts the ReAct [38] style. For example, CoA [68] improves CoT (Chain-of-Thought) reasoning by incorporating action history. Building on this, CoAT [69] further enhances CoT reasoning by integrating screen descriptions and previous action results, thereby improving reasoning accuracy and contextual awareness.

在 GUI 代理的组成部分中，规划能力至关重要，它辅助 GUI 代理进行任务规划、任务验证及反馈。在规划阶段，GUI 代理通常遵循链式思维 (CoT) 范式 [6, 7, 9, 34, 40]。在规划过程中，推理往往采用 ReAct [38] 风格。例如，CoA [68] 通过引入动作历史改进了链式思维推理。在此基础上，CoAT [69] 进一步通过整合屏幕描述和先前动作结果提升了推理的准确性和上下文感知能力。

Task Planner. GUI Agents typically utilize a task planner to plan complex tasks, obtaining a series of executable schemes to facilitate task completion. Generally, the Task Planner module of GUI Agents can be classified into two types-iterative planning and task decomposition planning.

任务规划器。GUI 代理通常利用任务规划器来规划复杂任务，获得一系列可执行方案以促进任务完成。一般而言，GUI 代理的任务规划器模块可分为两类——迭代规划和任务分解规划。

Iterative Planning. As shown in Figure 6 (a), the GUI Agents completes instruction tasks iteratively by dynamically adjusting the task plan based on the current state and feedback at each step, and inferring the next action strategy. For instance, AppAgent [7] progressively completes instruction tasks through continuous iteration. Additionally, some GUI Agents first generate an overall plan before proceeding with iterations. For example, Mobile-Agent [5] employs a self-planning method, initially generating a system prompt based on user instructions and then gradually completing each operation step by step, with the prompt format following the ReAct style. SheetCopilot [76] uses a state machine-based planner, which revises the plan through feedback from either LMs or software. The iterative planning process is highly dynamic, allowing the system to continuously adjust task strategies based on feedback at each step, effectively adapting to environmental changes and ensuring the reliability and effectiveness of the plan. However, when dealing with complex tasks, the reasoning steps of the GUI Agents may become excessively lengthy, which can lead to the LLMs generating hallucinations, causing subsequent planning to deviate from the intended objectives.

迭代规划。如图 6(a) 所示，GUI 代理通过动态调整每一步的任务计划，基于当前状态和反馈迭代完成指令任务，并推断下一步行动策略。例如，AppAgent [7] 通过持续迭代逐步完成指令任务。此外，一些 GUI 代理先生成整体计划，再进行迭代。例如，Mobile-Agent [5] 采用自我规划方法，先基于用户指令生成系统提示，然后逐步完成每个操作步骤，提示格式遵循 ReAct 风格。SheetCopilot [76] 使用基于状态机的规划器，通过来自语言模型或软件的反馈修正计划。迭代规划过程高度动态，允许系统根据每步反馈持续调整任务策略，有效适应环境变化，确保计划的可靠性和有效性。然而，在处理复杂任务时，GUI 代理的推理步骤可能过长，导致大型语言模型产生幻觉，进而使后续规划偏离预期目标。

Task Decomposition Planning. As shown in Figure 6 (b), the GUI Agents decomposes complex tasks into a series of subtasks and then infers action strategies for these subtasks, allowing the Agents to complete the instruction task by successfully executing all of the subtasks [82]. Compared to directly addressing complex tasks, decomposing tasks into simpler subtasks allows the language model to better maintain close connections between subtasks and the overall task, thereby reducing the risk of catastrophic forgetting or hallucination in large language models [128, 129]. However, task decomposition-based planning lacks flexibility because the subtasks are fixed from the outset, which makes it difficult to adapt to dynamic changes in the environment or unforeseen obstacles. Agent S [34] employs a task planner to decompose complex tasks into detailed, topologically sorted subtasks in order to accomplish user instructions. Existing planners, regardless of generating linear or nonlinear structured plans, require the agent to execute tasks in a sequential manner [40]. OS-Copilot utilizes a planner based on Directed Acyclic Graphs (DAG), allowing many independent tasks to be parallelized, thus minimizing execution time. Similarly,

VillagerAgent [70] constructs the decomposed subtasks into a DAG to achieve structured task management. Additionally, MobA [33] breaks down long-term tasks into a series of subtasks, allowing each subtask to be completed in a single step within the action module. If a subtask encounters an error or cannot be completed in one step, it is further decomposed until the task is fully completed.

任务分解规划。如图 6(b) 所示，GUI 代理将复杂任务分解为一系列子任务，然后推断这些子任务的行动策略，使代理能够通过成功执行所有子任务来完成指令任务 [82]。与直接处理复杂任务相比，将任务分解为更简单的子任务使语言模型能够更好地保持子任务与整体任务之间的紧密联系，从而降低大型语言模型中灾难性遗忘或幻觉的风险 [128, 129]。然而，基于任务分解的规划缺乏灵活性，因为子任务从一开始就是固定的，这使其难以适应环境的动态变化或意外障碍。Agent S [34] 采用任务规划器将复杂任务分解为详细的、拓扑排序的子任务，以完成用户指令。现有的规划器，无论是生成线性还是非线性结构的计划，都要求代理按顺序执行任务 [40]。OS-Copilot 利用基于有向无环图 (DAG, Directed Acyclic Graph) 的规划器，允许许多独立任务并行执行，从而最小化执行时间。同样，VillagerAgent [70] 将分解的子任务构建成 DAG，实现结构化任务管理。此外，MobA [33] 将长期任务分解为一系列子任务，允许每个子任务在动作模块中一步完成。如果子任务遇到错误或无法一步完成，则进一步分解，直到任务完全完成。

Verifier. The successful execution of GUI Agents relies on two critical components: precise planning and robust verification mechanisms [33, 61, 162]. Among these, the verifier is particularly essential, as it directly assesses the quality of the outputs generated by GUI Agents, ensuring their reliability and effectiveness. The verifier plays a key role in ensuring the quality of outputs generated by GUI agents, with the main function being to examine the reasoning steps and task execution states produced by the GUI Agents to ensure reliability and accuracy. Generally, the verifier utilizes inputs such as generated trajectories and the environment before and after action execution to compute performance score for the agent within the given task context. The verifier in GUI Agents generally leverages its own LLMs for both generation and evaluation purposes. Voyager [61] introduces a self-improving iterative prompting mechanism to perform self-verification of task states. Agent S [34] uses the verifier to summary strategies from completed subtasks as experiences, which are then used as textual rewards to offer better strategic suggestions for subsequent tasks. Additionally, Reflexion [60] conducts a more comprehensive evaluation by generating detailed textual outputs based on the trajectories generated by the agent. Moreover, the Self-Consistency [64] method is frequently used in evaluation tasks, selecting the most common answer from multiple generated outputs as the final solution. LATS [117] proposes a mechanism that combines scores generated by the agent's own LLM with self-consistency evaluation to assign values to inference nodes.

验证器。GUI 代理的成功执行依赖于两个关键组件: 精确的规划和强健的验证机制 [33, 61, 162]。其中，验证器尤为重要，因为它直接评估 GUI 代理生成输出的质量，确保其可靠性和有效性。验证器在确保 GUI 代理生成输出质量方面发挥关键作用，其主要功能是检查 GUI 代理产生的推理步骤和任务执行状态，以确保可靠性和准确性。通常，验证器利用生成的轨迹以及动作执行前后的环境作为输入，在给定任务上下文中计算代理的性能得分。GUI 代理中的验证器通常利用自身的大型语言模型 (LLMs) 进行生成和评估。Voyager [61] 引入了自我改进的迭代提示机制以执行任务状态的自我验证。Agent S [34] 使用验证器从已完成的子任务中总结策略作为经验，进而作为文本奖励为后续任务提供更优策略建议。此外，Reflexion [60] 通过基于代理生成的轨迹生成详细文本输出，进行更全面的评估。此外，自洽性 (Self-Consistency) 方法 [64] 常用于评估任务，从多个生成输出中选择最常见答案作为最终解答。LATS [117] 提出了一种机制，将代理自身 LLM 生成的分数与自洽性评估结合，为推理节点赋值。

Feedback. In practical application scenarios, GUI Agents often need to handle long-term planning and execution of complex tasks. During task execution, the agent is likely to encounter unforeseen errors, which could lead to task failure. For instance, after executing a certain operation, the actual state of the UI may not match expectations, or the outcome of the operation may not satisfy the intended objectives. Therefore, feedback mechanisms are used to monitor the entire task execution process and provide timely semantic feedback to the agent, allowing for rapid adjustment and correction of the task plan when issues arise. In general, feedback can be categorized into two types: external feedback and internal feedback.

反馈。在实际应用场景中，GUI 代理常需处理复杂任务的长期规划与执行。任务执行过程中，代理可能遇到意外错误，导致任务失败。例如，执行某操作后，UI 的实际状态可能与预期不符，或操作结果未达到预期目标。因此，反馈机制用于监控整个任务执行过程，并向代理提供及时的语义反馈，使其在出现问题时能够快速调整和修正任务计划。一般而言，反馈可分为两类: 外部反馈和内部反馈。

External Feedback primarily refers to environmental feedback, which captures the direct response to the agent's task execution, such as the observation results following the agent's action. This feedback enables the agent to evaluate whether the current execution step has achieved the expected outcome, offering crucial guidance for subsequent actions. Environmental feedback is usually obtained by the Agent through specific methods such as screenshots 54, 69, 163-165]. For example, ReAct [38] proposed a triplet ⟨ thought, action, observation ⟩ , utilizing observations of the current environmental state after each action to provide feedback to the agent. Specifically, thought represents the process of reasoning and deliberation, action denotes the action taken by the agent, and observation is the information obtained from the environment. Voyager [61] integrates environmental feedback into the task execution process. Similarly, MP5 [166] analyzes environmental feedback to identify the root causes of task execution failures and uses this information to guide the planner in refining subsequent task planning. This mechanism prevents the system from continuing along incorrect paths and allows for flexible adjustment of the execution strategy according to the environmental state. LATS [117] uses environmental feedback as observation expansion nodes and utilizes this feedback to directly provide evaluation rewards, thereby assisting in task assessment.

外部反馈主要指环境反馈，即捕捉对代理任务执行的直接响应，如代理动作后的观察结果。该反馈使代理能够评估当前执行步骤是否达成预期效果，为后续行动提供关键指导。环境反馈通常通过代理采用特定方法获取，如截图 [54, 69, 163-165]。例如，ReAct [38] 提出了三元组 ⟨ 思考、行动、观察 ⟩，利用每次动作后当前环境状态的观察为代理提供反馈。具体而言，思考代表推理与思考过程，行动表示代理采取的动作，观察是从环境中获得的信息。Voyager [61] 将环境反馈整合进任务执行流程。类似地，MP5 [166] 分析环境反馈以识别任务执行失败的根本原因，并利用该信息指导规划器优化后续任务规划。该机制防止系统沿错误路径继续执行，并根据环境状态灵活调整执行策略。LATS [117] 将环境反馈作为观察扩展节点，利用该反馈直接提供评估奖励，从而辅助任务评估。

Internal Feedback typically refers to self-feedback. Generally, GUI Agents utilize their own LLMs to generate feedback, which is used either as part of the prompt for the next action [76] or stored in a memory module [167], enabling the Agent to rapidly refine its decision-making plans and effectively solve problems by leveraging experience through linguistic feedback signals. Feedback mechanisms can significantly enhance the accuracy and reliability of the agent in task execution [72] LATS [117] provides additional semantic signals to the agents through self-reflection. Similarly, the plan reflection module in UFO [9] prompts the agent to continuously modify its plan at each decision step, allowing deviations from the original route as needed. Agent S [34] adopts a similar approach

by offering reflective suggestions based on the observation of the entire execution trajectory of subtasks, helping the agent consider alternative strategies and avoid repetitive actions. Moreover, the input to the feedback model is not limited to action trajectories. For instance, Reflexion [60] generates linguistic self-feedback based on reward signals, trajectory information, and relevant memories, offering detailed and valuable guidance for future attempts of the agent. Hu et al. [168] offer concise summary feedback of historical actions and current step information in each iteration, which reduces feedback overhead and mitigates performance degradation of the agent caused by information overload.

内部反馈通常指自我反馈。一般来说，GUI 代理利用自身的大型语言模型 (LLM) 生成反馈，这些反馈要么作为下一步操作提示的一部分 [76]，要么存储在记忆模块中 [167]，使代理能够通过语言反馈信号快速优化决策计划，有效解决问题。反馈机制显著提升了代理在任务执行中的准确性和可靠性 [72]。LATS[117] 通过自我反思为代理提供额外的语义信号。同样，UFO[9] 中的计划反思模块促使代理在每个决策步骤持续修改计划，允许根据需要偏离原始路线。Agent S[34] 采用类似方法，基于对子任务整个执行轨迹的观察，提供反思性建议，帮助代理考虑替代策略，避免重复操作。此外，反馈模型的输入不限于动作轨迹。例如，Reflexion[60] 基于奖励信号、轨迹信息和相关记忆生成语言自我反馈，为代理未来尝试提供详细且有价值的指导。胡等人 [168] 在每次迭代中提供历史动作和当前步骤信息的简明总结反馈，减少反馈开销，缓解信息过载导致的性能下降。

2.5 Interaction

2.5 交互

GUI Agents are intelligent systems designed to interact with graphical user interfaces to complete various tasks to help users. These agents rely on sophisticated interaction mechanisms to effectively navigate and manipulate interface elements. The interaction capabilities of these agents can be refined into three closely related components: Action Goal, Action Generation, and Action Space. Action Goal defines the specific objectives that guide the agent's task execution, directly aligning with user requirements to help complete daily tasks more efficiently and enhance productivity (§2.5.1). Action Generation involves the agent's ability to accurately interpret user instructions, deeply understand the GUI environment, and generate executable actions through processes like GUI Grounding and various Generation Strategies (§2.5.2). Action Space represents the complete set of possible operations available to the agent within interface constraints, including User Action Simulation and API Invocation approaches (§2.5.3).

GUI 代理是设计用来与图形用户界面交互以完成各种任务、帮助用户的智能系统。这些代理依赖复杂的交互机制，有效导航和操作界面元素。代理的交互能力可细分为三个紧密相关的组成部分: 动作目标、动作生成和动作空间。动作目标定义指导代理执行任务的具体目标，直接对应用户需求，帮助更高效地完成日常任务并提升生产力 (§2.5.1)。动作生成涉及代理准确理解用户指令、深入理解 GUI 环境，并通过 GUI 定位和多种生成策略生成可执行动作的能力 (§2.5.2)。动作空间代表代理在界面限制内可执行的所有操作集合，包括用户动作模拟和 API 调用方法 (§2.5.3)。

2.5.1 Action Goal

2.5.1 动作目标

Action Goal defines the specific objectives that guide the GUI Agent's task execution. These objectives directly align with user requirements and ensure the agent efficiently and accurately completes the requested operations. Action goal encompasses diverse application areas including smartphone interaction [6], work task automation, table data processing [76], online shopping assistance [74], gaming automation [53], and web navigation. Fundamentally, Action goal enables GUI Agents to help users complete daily tasks more efficiently, enhancing productivity and simplifying complex processes. By achieving these goals, the GUI Agents becomes a capable assistant for enhancing productivity and simplifying task handling.

动作目标定义指导 GUI 代理执行任务的具体目标。这些目标直接对应用户需求，确保代理高效准确地完成请求的操作。动作目标涵盖多种应用领域，包括智能手机交互 [6]、工作任务自动化、表格数据处理 [76]、在线购物辅助 [74]、游戏自动化 [53] 和网页导航。根本上，动作目标使 GUI 代理能够帮助用户更高效地完成日常任务，提升生产力并简化复杂流程。通过实现这些目标，GUI 代理成为提升生产力和简化任务处理的得力助手。

To achieve different goals, action generation is a critical step. Action generation involves the GUI Agent's ability to accurately interpret user instructions, deeply understand the GUI environment, and generate executable actions based on this understanding. This process forms the core of the Agent's task completion capabilities and encompasses various methods and strategies, which will be elaborated in the following sections.

为实现不同目标，动作生成是关键步骤。动作生成涉及 GUI 代理准确理解用户指令、深入理解 GUI 环境，并基于此理解生成可执行动作的能力。该过程构成代理完成任务能力的核心，涵盖多种方法和策略，后续章节将详细阐述。

2.5.2 Action Generation

2.5.2 动作生成

Action Generation is the core process through which GUI Agents transform user intentions into executable operations. Unlike traditional intelligent agents that operate solely through API calls, GUI Agents distinctively need to interact directly with graphical interface elements, particularly through precise clicking operations on the screen [13]. Through our systematic review, we have categorized Action Generation into two key aspects: GUI Grounding and Action Strategy.

动作生成是 GUI 代理将用户意图转化为可执行操作的核心过程。不同于仅通过 API 调用操作的传统智能代理，GUI 代理需要直接与图形界面元素交互，特别是通过屏幕上的精确点击操作 [13]。通过系统综述，我们将动作生成归纳为两个关键方面:GUI 定位和动作策略。

2.5.2.1 GUI Grounding refers to the specialized process by which GUI Agents establish connections between user instructions and interface elements for the purpose of generating precise click operations. GUI Agents must directly interact with graphical environments through screen-based interactions, making GUI Grounding a distinctive and critical capability. This process enables agents to determine exact coordinates for clicks by mapping abstract instructions to specific interface elements.

2.5.2.1 GUI 定位指 GUI 代理为生成精确点击操作而建立用户指令与界面元素之间联系的专门过程。GUI 代理必须通过基于屏幕的交互直接操作图形环境，使 GUI 定位成为独特且关键的能力。该过程使代理通过将抽象指令映射到具体界面元素，确定点击的精确坐标。

For GUI Agents, effective grounding represents a fundamental requirement for successful interaction, as it allows them to navigate and manipulate interfaces through targeted clicking operations. This element localization process-essential for any screen-based interaction-serves as the foundation upon which other actions can be built. Current approaches to GUI grounding can be categorized into three methodologies:

对于 GUI 代理，有效的定位是成功交互的基本要求，因为它使代理能够通过目标点击操作导航和操控界面。该元素定位过程是任何基于屏幕交互的基础，为其他动作构建基础。当前 GUI 定位方法可分为三类:

- MLLMs-based Grounding leverages both visual and textual information to connect commands with interface elements. [71] utilizes screen captures alongside user commands to decompose complex tasks into sequential subtasks. [6] enhances this approach by incorporating detailed XML descriptions of interaction elements. Further advancements by [5] and [58] integrate specialized tools like OCR for text localization, icon detection, and CLIP for visual element identification. [169] employs numerical labeling of elements detected through OCR and IconNet, while [73] uses GPT-4V-ACT to extract and annotate interaction elements with supplementary HTML analysis.

  - 基于多模态大型语言模型 (MLLMs) 的定位利用视觉和文本信息将命令与界面元素连接。[71] 结合屏幕截图和用户命令，将复杂任务分解为顺序子任务。[6] 通过引入交互元素的详细 XML 描述增强该方法。[5] 和 [58] 进一步集成 OCR 文本定位、图标检测和 CLIP 视觉元素识别等专用工具。[169] 通过 OCR 和 IconNet 检测的元素进行数字标注，[73] 则使用 GPT-4V-ACT 结合 HTML 分析提取并注释交互元素。

- LLMs-based Grounding converts GUI structures into formats that large language models can effectively process. [48] transforms GUIs into simplified HTML with dedicated tags, systematically exploring all interactive elements by scrolling through components and recording visible UI elements. [170] combines HTML information with OCR processing to create more accurate element descriptions. Taking a different approach, [40] focuses on command interpretation rather than direct GUI understanding, using directed acyclic graphs for task decomposition while incorporating external knowledge and internal memory.

  - 基于大型语言模型 (LLMs) 的 Grounding 将 GUI 结构转换为大型语言模型能够有效处理的格式。[48] 将 GUI 转换为带有专用标签的简化 HTML，通过滚动组件系统地探索所有交互元素并记录可见的 UI 元素。[170] 结合 HTML 信息与 OCR 处理，生成更准确的元素描述。采取不同方法的 [40] 侧重于命令解释而非直接理解 GUI，使用有向无环图进行任务分解，同时融合外部知识和内部记忆。

- Domain-specific Grounding approaches target particular application contexts with specialized techniques. [76] focuses on spreadsheet software, analyzing column names and row counts to enhance task parsing. [171] employs an innovative ranking system for HTML elements, enriching representation by identifying visual neighbors of candidate elements and combining their markup and visual features to create contextually rich representations.

- 领域特定的 Grounding 方法针对特定应用场景采用专门技术。[76] 聚焦电子表格软件，分析列名和行数以增强任务解析。[171] 采用创新的 HTML 元素排序系统，通过识别候选元素的视觉邻居并结合其标记和视觉特征，创建具有丰富上下文的表示。

2.5.2.2 Generation Strategy While GUI Grounding focuses specifically on element-targeting clicks, GUI Agents must also generate a broader range of operations including scrolling, navigation, typing, and multi-step interactions. These non-click operations require different generation approaches that go beyond simple element localization. Based on our systematic review, we identify two primary strategies for generating these complex operation sequences:

2.5.2.2 生成策略虽然 GUI Grounding 专注于针对元素的点击操作，GUI 代理还必须生成更广泛的操作，包括滚动、导航、输入和多步交互。这些非点击操作需要超越简单元素定位的生成方法。基于我们的系统综述，我们识别出生成这些复杂操作序列的两种主要策略:

- Memory-Based Action Generation leverages information from previous interactions or external knowledge. Several approaches maintain operational history to inform future actions. [5], [169], and [8] maintain operation flows that record task history, generating next actions based on instructions and previous operations. [58] enhances this approach by storing focused content from historical screens. [6] and [48] build memories through autonomous exploration, retrieving similar scenarios to guide action generation for new tasks. External knowledge integration appears in works like [76], which provides detailed documentation for spreadsheet operations, and [172], which combines web-searched knowledge with historical experience to generate subtasks.

- 基于记忆的动作生成利用先前交互或外部知识的信息。多种方法维护操作历史以指导后续动作。[5]、[169] 和 [8] 维护记录任务历史的操作流程，根据指令和先前操作生成下一步动作。[58] 通过存储历史屏幕的重点内容增强此方法。[6] 和 [48] 通过自主探索构建记忆，检索相似场景以指导新任务的动作生成。外部知识整合见于 [76]，其提供电子表格操作的详细文档，以及 [172]，结合网络搜索知识与历史经验生成子任务。

- Planning-Based Action Generation relies on structured plans created before execution. [173] generates not only immediate actions but also future action plans as references. [9] distinguishes between global and local planning, where a HostAgent generates coarse-grained action plans and selects appropriate applications, while an AppAgent [6] executes specific operations within those applications. Task decomposition approaches like [71] and [172] break complex tasks into sequential subtasks. These planning strategies enable GUI Agents to systematically approach complex tasks, adapting to environmental changes while maintaining focus on overall objectives.

- 基于规划的动作生成依赖于执行前创建的结构化计划。[173] 不仅生成即时动作，还生成未来动作计划作为参考。[9] 区分全局规划与局部规划，HostAgent 生成粗粒度动作计划并选择合适应用，AppAgent[6] 在应用内执行具体操作。任务分解方法如 [71] 和 [172] 将复杂任务拆分为顺序子任务。这些规划策略使 GUI 代理能够系统性地处理复杂任务，适应环境变化，同时保持对整体目标的关注。

2.5.3 Action Space

2.5.3 动作空间

After completing action generation, GUI Agents require a well-defined action space to execute operations effectively. The action space represents the complete set of possible operations available to the agent within interface constraints. These constraints vary across different environments, limiting what actions can be performed in specific contexts, as shown in Figure 7. For GUI Agents, action capabilities can be divided into two major categories:

完成动作生成后，GUI 代理需要明确定义的动作空间以有效执行操作。动作空间表示代理在界面约束内可执行的所有可能操作集合。这些约束因环境不同而异，限制特定上下文中可执行的动作，如图 7 所示。对于 GUI 代理，动作能力可分为两大类:

(1) User Action Simulation involves simulating typical user interactions with graphical interfaces. These actions mimic human behaviors such as clicking, typing, and swiping. [6] defines a basic set of operations including (Tap, Long_Press, Swipe, Text, Back, and Exit) to replicate human-screen interactions. Some systems like [6], [169], and [9] use numerical element labeling, directing actions to specific numbered elements (e.g., Tap(5)). Others, such as [174] and [8], rely on coordinate-based targeting (e.g., Click(0.3,0.9)). These differences stem from varying GUI understanding methods: either annotating elements directly on screenshots or describing element positions textually.

(1) 用户动作模拟涉及模拟用户与图形界面的典型交互。这些动作模仿人类行为，如点击、输入和滑动。[6] 定义了一组基本操作 (包括点击、长按、滑动、文本输入、返回和退出) 以复制人机交互。一些系统如 [6]、[169] 和 [9] 使用数字元素标记，将动作指向特定编号元素 (如 Tap(5))。另一些如 [174] 和 [8] 则依赖基于坐标的定位 (如 Click(0.3,0.9))。这些差异源于不同的 GUI 理解方法: 直接在截图上标注元素或通过文本描述元素位置。

(2) API Invocation Actions leverages external or custom APIs rather than simulating direct user interactions. In [40], an executor generates executable function calls with appropriate parameters based on configuration hints. These might include command line instructions like ”mkdir new folder” or other system calls. When suitable tools aren’t available, a tool generator creates customized solutions for specific tasks. Through these varied action spaces, GUI Agents can flexibly perform diverse tasks across different scenarios while maintaining operational safety and accuracy.

(2) API 调用动作利用外部或自定义 API，而非模拟直接用户交互。[40] 中的执行器根据配置提示生成带有适当参数的可执行函数调用，可能包括命令行指令如“mkdir new folder”或其他系统调用。当合适工具不可用时，工具生成器为特定任务创建定制解决方案。通过这些多样的动作空间，GUI 代理能够灵活执行不同场景下的多样任务，同时保持操作的安全性和准确性。

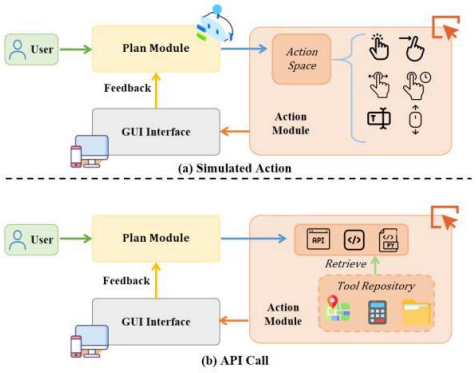

Fig. 7: Different action space for GUI agents: (a) Simulated Action, and (b) API Call.

图 7:GUI 代理的不同动作空间:(a) 模拟动作，(b) API 调用。

3 APPLICATION

3 应用

The development of GUI Agents has enabled automation across diverse computing environments, each presenting unique challenges and opportunities. This section examines four major application domains: mobile devices, desktop computers, web browsers, and games, analyzing their distinct characteristics and technical requirements. These four application domains demonstrate both the versatility of GUI Agents and the specific technical challenges each environment presents, as shown in Figure 8 The continued development of these applications drives innovation in core technologies while revealing new opportunities for automation and human-computer interaction.

GUI 代理的发展推动了多样计算环境中的自动化，每种环境都带来了独特的挑战与机遇。本节考察四大主要应用领域: 移动设备、桌面计算机、网页浏览器和游戏，分析其独特特性和技术需求。这四个应用领域既展示了 GUI 代理的多功能性，也揭示了各环境特有的技术挑战，如图 8 所示。这些应用的持续发展推动核心技术创新，同时揭示自动化与人机交互的新机遇。

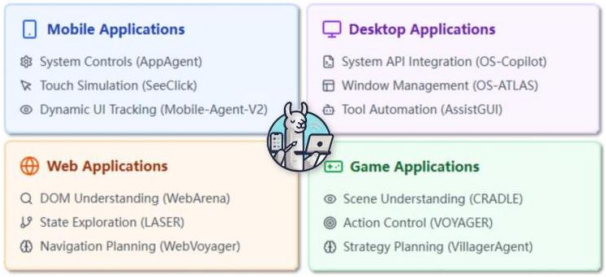

Fig. 8: Applications of GUI Agents

图 8:GUI 代理的应用

3.1 Mobile

3.1 移动端

Mobile environments represent a primary application domain for GUI Agents, characterized by touch-based interactions and dynamic interfaces. The proliferation of mobile applications has created significant opportunities for automation while introducing unique technical challenges.

移动环境是 GUI 代理的主要应用领域，特点是基于触控的交互和动态界面。移动应用的普及带来了显著的自动化机会，同时引入了独特的技术挑战。

Mobile GUI Agents have demonstrated remarkable capabilities across various tasks. In system operations, AppAgent [6] achieves sophisticated control over device settings, handling brightness adjustments, volume control, and other system configurations. Mobile-Agent [5] extends these capabilities by implementing comprehensive application management, including installation, updates, and removal procedures. For application interactions, CoCo-Agent [72] enables automated social media engagement, while Mobile-Flow [30] specializes in productivity applications, offering particular advantages for Chinese-language interfaces.

移动 GUI 代理在各类任务中展现了卓越的能力。在系统操作方面，AppAgent [6] 实现了对设备设置的精细控制，涵盖亮度调节、音量控制及其他系统配置。Mobile-Agent [5] 通过实现全面的应用管理 (包括安装、更新和卸载流程) 扩展了这些功能。针对应用交互，CoCo-Agent [72] 支持自动化社交媒体互动，而 Mobile-Flow [30] 专注于生产力应用，特别适用于中文界面。

The technical implementation of mobile GUI Agents faces several distinct challenges. First, the touch-based interaction model requires precise spatial understanding and gesture simulation. SeeClick [8] addresses this through advanced element localization techniques, achieving high-precision touch event simulation. Second, mobile interfaces are highly dynamic, with frequent state changes and transitions. Mobile-Agent-V2 [58] tackles this challenge through a sophisticated state tracking system, maintaining contextual awareness across application switches and screen updates.

移动 GUI 代理的技术实现面临若干独特挑战。首先，基于触控的交互模型要求精确的空间理解和手势模拟。SeeClick [8] 通过先进的元素定位技术解决了这一问题，实现了高精度的触控事件模拟。其次，移动界面高度动态，状态变化和转换频繁。Mobile-Agent-V2 [58] 通过复杂的状态跟踪系统应对这一挑战，保持跨应用切换和屏幕更新的上下文感知。

Security considerations are particularly crucial in mobile environments. Recent implementations like AppAgent-V2 [7] incorporate comprehensive safety protocols, requiring explicit user confirmation for sensitive operations and maintaining detailed operation logs. This security-first approach ensures reliable automation while protecting user data and device integrity.

安全性在移动环境中尤为关键。近期实现如 AppAgent-V2 [7] 融入了全面的安全协议，敏感操作需用户明确确认，并保持详细的操作日志。这种以安全为先的策略确保了自动化的可靠性，同时保护用户数据和设备完整性。

3.2 Desktop

3.2 桌面

Desktop environments offer GUI Agents extensive control capabilities through rich APIs and diverse interaction methods. The complexity of desktop applications demands sophisticated automation strategies and robust error handling mechanisms.

桌面环境通过丰富的 API 和多样的交互方式，为 GUI 代理提供了广泛的控制能力。桌面应用的复杂性要求采用复杂的自动化策略和强健的错误处理机制。

Desktop GUI Agents have achieved significant breakthroughs in system-level automation. OS-Copilot [40] demonstrates comprehensive control over operating system functions, including file management, application control, and system configuration. UFO [9] specializes in Windows environments, providing precise interaction with native applications and system utilities. In productivity applications, AssistGUI [51] enables sophisticated automation of office suites, while SheetCopilot [76] offers specialized capabilities for spreadsheet manipulation.

桌面 GUI 代理在系统级自动化方面取得了显著突破。OS-Copilot [40] 展示了对操作系统功能的全面控制，包括文件管理、应用控制和系统配置。UFO [9] 专注于 Windows 环境，提供对原生应用和系统工具的精准交互。在生产力应用中，AssistGUI [51] 实现了办公套件的复杂自动化，而 SheetCopilot [76] 则专注于电子表格操作的专业功能。

These implementations leverage multiple technical approaches to achieve reliable automation. OS-Copilot [40] utilizes a hybrid approach, combining direct API calls with GUI interaction when necessary. UFO [9] implements a sophisticated element detection system that maintains accuracy across different Windows versions and display configurations. OS-ATLAS [75] addresses the challenge of multi-window environments through a hierarchical attention mechanism, effectively managing complex desktop layouts.

这些实现采用多种技术手段以实现可靠的自动化。OS-Copilot [40] 采用混合方法，结合直接 API 调用与必要时的 GUI 交互。UFO [9] 实施了复杂的元素检测系统，确保在不同 Windows 版本和显示配置下的准确性。OS-ATLAS [75] 通过分层注意力机制解决多窗口环境的挑战，有效管理复杂的桌面布局。

A notable advancement in desktop automation is the integration of natural language understanding with system operations. Recent work like OS-Copilot [40] demonstrates the ability to translate high-level user instructions into precise sequences of system operations, bridging the gap between user intent and technical execution.

桌面自动化的一项显著进展是自然语言理解与系统操作的结合。近期工作如 OS-Copilot [40] 展示了将高级用户指令转化为精确系统操作序列的能力，弥合了用户意图与技术执行之间的鸿沟。

3.3 Web Browser

3.3 网络浏览器

Web browsers present a standardized yet dynamic environment for GUI Agents, with DOM structures providing a consistent interface layer while supporting diverse interactive experiences.

网络浏览器为 GUI 代理提供了标准化且动态的环境，DOM 结构提供了一致的接口层，同时支持多样的交互体验。

Web-focused GUI Agents have demonstrated remarkable versatility. WebVoyager [73] executes complex web navigation tasks through sophisticated page understanding and interaction planning. LASER [74] implements a state-space exploration approach for robust e-commerce automation, handling dynamic content and variable page layouts. AutoWebGLM [170] specializes in form interactions and data extraction, with particular emphasis on maintaining consistency across different website designs.

面向网络的 GUI 代理展现了卓越的多功能性。WebVoyager [73] 通过复杂的页面理解和交互规划执行复杂的网页导航任务。LASER [74] 实施状态空间探索方法，实现稳健的电子商务自动化，处理动态内容和多变的页面布局。AutoWebGLM [170] 专注于表单交互和数据提取，特别强调在不同网站设计间保持一致性。

The technical implementation of web GUI Agents leverages several key innovations. WebArena's [78] DOM-based understanding system provides robust element identification and interaction planning. LASER's [74] state tracking mechanism enables reliable navigation through complex web applications, maintaining context across page loads and state transitions. These agents must handle AJAX updates, dynamic content loading, and varying network conditions, requiring sophisticated state management and error recovery mechanisms.

网络 GUI 代理的技术实现依托若干关键创新。WebArena [78] 的基于 DOM 的理解系统提供了稳健的元素识别和交互规划。LASER [74] 的状态跟踪机制支持在复杂网页应用中可靠导航，保持页面加载和状态转换间的上下文。这些代理必须应对 AJAX 更新、动态内容加载及多变的网络条件，需具备复杂的状态管理和错误恢复机制。

3.4 Game

3.4 游戏

Game environments present unique challenges for GUI Agents, requiring real-time decision-making and precise control execution in dynamic, visually complex environments.

游戏环境为 GUI 代理带来独特挑战，要求在动态且视觉复杂的环境中实现实时决策和精确控制执行。

Game-oriented GUI Agents have achieved impressive results across different genres. VOYAGER [61] demonstrates sophisticated resource management and planning in Minecraft, combining visual understanding with strategic decision-making. Cradle [53] implements precise control mechanisms for action games, while VillagerAgent [70] manages complex task dependencies in multi-agent scenarios.

面向游戏的 GUI 代理在不同类型游戏中取得了令人瞩目的成果。VOYAGER [61] 在 Minecraft 中展示了复杂的资源管理和规划，结合视觉理解与战略决策。Cradle [53] 实现了动作游戏的精确控制机制，而 VillagerAgent [70] 管理多代理场景中的复杂任务依赖关系。

The technical implementation of game GUI Agents requires several innovative approaches. Real-time visual understanding is achieved through specialized perception models, often combining multiple analysis techniques. Cradle [53] integrates GPT-4V with specialized game state understanding models, enabling robust scene interpretation. Action planning systems must handle both strategic decisions and tactical execution, often operating under strict time constraints.

游戏 GUI 代理的技术实现需采用多项创新方法。实时视觉理解通过专门的感知模型实现，通常结合多种分析技术。Cradle [53] 集成了 GPT-4V 与专门的游戏状态理解模型，实现了稳健的场景解析。动作规划系统需同时处理战略决策和战术执行，常在严格的时间限制下运行。

A significant innovation in game automation is the development of skill learning systems. VOYAGER [61] implements a progressive skill acquisition mechanism, allowing the agent to learn and combine basic actions into complex operation sequences. This approach enables adaptation to new game scenarios and the development of sophisticated playing strategies. .

游戏自动化中的一项重要创新是技能学习系统的发展。VOYAGER [61] 实现了一种渐进式技能获取机制，使智能体能够学习并将基本动作组合成复杂的操作序列。这种方法使其能够适应新的游戏场景并发展出复杂的游戏策略。

4 DATASETS AND BENCHMARKS

4 数据集与基准

A comprehensive understanding of datasets and benchmarks is essential for advancing research in GUI Agents. In this section, we analyze existing benchmarks and datasets from three critical dimensions: the evaluation environments (§4.1) in which agents operate, the specific tasks (§4.2) they are designed to accomplish, and the evaluation strategies (§4.3) employed to assess their performance. Additionally, we critically examine the significant challenges (§4.4) facing current datasets and benchmarks.

全面理解数据集和基准对于推动 GUI 智能体 (GUI Agents) 研究至关重要。本节从三个关键维度分析现有基准和数据集: 智能体运行的评估环境 (§4.1)、其设计完成的具体任务 (§4.2) 以及用于评估其性能的评估策略 (§4.3)。此外，我们还批判性地审视当前数据集和基准面临的重大挑战 (§4.4)。

4.1 Environment

4.1 环境

Effective evaluation is critical for assessing and advancing GUI Agents' capabilities in real-world scenarios. Through a comprehensive review of existing literature, we identify that the evaluation environments provided by

current benchmarks and datasets related to GUI Agents can be categorized into three types: static replicas, simulated environments, and real-world environments.

有效的评估对于衡量和提升 GUI 智能体在真实场景中的能力至关重要。通过对现有文献的综合回顾，我们发现当前与 GUI 智能体相关的基准和数据集所提供的评估环境可分为三类: 静态复制环境、模拟环境和真实环境。

4.1.1 Static Replicas

4.1.1 静态复制环境

Static replicas are evaluation environments that capture GUI interfaces as fixed, non-interactive representations such as screenshots or HTML snapshots. These environments record interactions with the original GUI as static data, allowing for reproducible evaluation without requiring the actual application to be running. The static replica approach records interactions with the environment as static data, as shown in Table 1

静态复制环境是将 GUI 界面捕捉为固定的、不可交互的表现形式，如截图或 HTML 快照的评估环境。这些环境将与原始 GUI 的交互记录为静态数据，允许在不运行实际应用的情况下进行可复现的评估。静态复制方法将与环境的交互记录为静态数据，如表 1 所示。

AitW [49] constructed a large-scale dataset of human-annotated action sequences and associated application state data based on static screenshots, comprising 715,142 episodes, 5,689,993 screenshots, and 30,378 unique task instructions. PIXELHELP [178] documented touch event types during interactions, the UI objects manipulated, and the view hierarchy. Mind2Web [42] preserved web interactions in the form of static HTML snapshots, covering 31 domains and including over 2,000 tasks. These works facilitate the generation of large-scale datasets. However, since annotators cannot exhaustively cover all possible execution paths, it is impossible to explore or replay uncached operational paths during evaluation. Consequently, this method constitutes only a ”pseudo-interaction”, which may result in false negatives during evaluation. This limitation is discussed in detail in Section (§4.3.1).

AitW [49] 构建了一个基于静态截图的人类标注动作序列及相关应用状态数据的大规模数据集，包含 715,142 个回合、5,689,993 张截图和 30,378 条独特任务指令。PIXELHELP [178] 记录了交互过程中的触摸事件类型、操作的 UI 对象及视图层级。Mind2Web [42] 以静态 HTML 快照形式保存了网页交互，涵盖 31 个领域，包含 2000 多个任务。这些工作促进了大规模数据集的生成。然而，由于标注者无法穷尽所有可能的执行路径，评估时无法探索或重放未缓存的操作路径。因此，该方法仅构成“伪交互”，可能导致评估中的假阴性问题。此限制在第 (§4.3.1) 节中有详细讨论。

Datasets Ope.	Env. Dial.	Eval Method Time	Eval Strategy	Platform	Per.	Task		
WoB 175	✘	HTML/JS state	Goal	web	✓	✓		2017.08
Rico 176	✘	/	/	mobile		✓		2017.10
MiniWoB++ 177	✘	HTML/JS state	Goal	web		✓		2018.02
PIXELHELP 178	✘	action-match	Trajectory	mobile		✓		2020.06
WebSRC 179	✘	UI/text match	Goal	web	✓			2021.11
MoTIF 180	✘	action/UI-match	Trajectory	mobile		✓		2022.08
META-GUI 181	✘	action-match	Trajectory	mobile		✓	✓	2022.11
WebShop 182	✘	product attribution match	Goal	web		✓		2023.02
AITW49	✘	action-match	Trajectory	mobile		✓		2023.10
Mind2Web 42	✘	action/UI-match	Trajectory	web		✓		2023.12
AssistGUI|51	✓	image match/device state	Goal	desktop		✓		2024.01
MT-Mind2Web|183	✘	action/UI-match	Trajectory	web		✓	✓	2024.02
WebVLN 184	✘	url/text match	Goal/Graph	web	✓	✓		2024.03
WebArena 43	✘	url/text match/LLM	Goal/Trajectory	web	✓	✓		2024.04
MMInA 185	✓	text-match/LLM	Trajectory	web		✓		2024.04
VisualWebBench 186	✘	action/UI/text-match	Goal	web	✓	✓		2024.04
OSWorld 187	✓	device state	Goal	desktop		✓		2024.05
Mobile-Env 188	✓	device state	Goal	mobile		✓		2024.06
VisualWebArena|47|	✓	url/text/image match/HTML state/LLM	Goal	web	✓	✓		2024.06
VideoGUI 189	✓	action-match/LLM	Goal	desktop		✓		2024.06
MobileAgentBench 41	✓	UI match	Goal	mobile		✓		2024.06
GUICourse 190	✘	/	/	Web&mobile	✓	✓	✓	2024.06
GUI-WORLD 191 Platforms	✘ ✓	text-match/LLM ✓	Goal ✓	Multi 2024.06				
GUI Odyssey 192	✘	action-match	Trajectory	mobile		✓		2024.06
WebCanvas 193	✓	UI/url match	Trajectory	web		✓		2024.07
Spider2-V 194	✓	device state	Goal	desktop		✓		2024.07
OmniACT 195	✘	action-match	Trajectory	desktop		✓		2024.07
AMEX 196	✘	action-match	Trajectory	mobile	✓	✓		2024.07
LlamaTouch 197	✓	UI/action-match/device state	Trajectory	mobile		✓		2024.08
AndroidArena 198	✓	action-match/LLM	Trajectory	mobile		✓		2024.08
WindowsAgentArena 199	✓	device state	Goal	desktop		✓		2024.09
WebLINX 200	✘	action/UI/text-match	Goal	web		✓	✓	2024.09
AgentStudio 201	✓	device state	Goal	desktop		✓		2024.10
B-MOCA 202	✓	device state	Goal	mobile		✓		2024.10
AndroidWorld|45	✓	device state	Goal	mobile		✓		2024.10
CRAB|203 Platforms	✓	UI/image-match/device state ✓	Graph	Multi 2024.10				
ScreenPR|204	✘	human/LLM/text match	Goal	mobile	✓			2024.10
SPA-BENCH | 205	✓	LLM	Goal/Trajectory	Mobile		✓		2024.10
Android Lab 29	✓	device state	device state	Mobile		✓		2024.10
ANDROIDCONTROL|206|×		action-match	Trajectory	mobile		✓		2024.11
MultiUI|207	✘	/	/	web	✓	✓		2024.11
GUI Testing Arena 208	✓	Action match/LLM	Goal/Trajectory	Mobile	✓	✓		2024.12
A3 |209	✓	UI/Action match/LLM	Goal/Trajectory	Mobile		✓		2025.01
WebWalkerQA |210	✓	LLM	Goal	Web		✓	✓	2025.01
ScreenSpot-Pro 46	✘	UI match	Goal	Desktop	✓			2025.01
WorldGUI 211	✓	UI match/device state	Goal	Web&desktop		✓		2025.02
UI-Vision 212	✘	UI/action match	Goal	Desktop	✓	✓		2025.03

数据集 操作	环境 对话	评估方法 时间	评估策略	平台	性能	任务		
WoB 175	✘	HTML/JS 状态	目标	网页	✓	✓		2017.08
Rico 176	✘	/	/	移动端		✓		2017.10
MiniWoB++ 177	✘	HTML/JS 状态	目标	网页		✓		2018.02
PIXELHELP 178	✘	动作匹配	轨迹	移动端		✓		2020.06
WebSRC 179	✘	界面/文本匹配	目标	网页	✓			2021.11
MoTIF 180	✘	动作/界面匹配	轨迹	移动端		✓		2022.08
META-GUI 181	✘	动作匹配	轨迹	移动端		✓	✓	2022.11
WebShop 182	✘	产品属性匹配	目标	网页		✓		2023.02
AITW49	✘	动作匹配	轨迹	移动端		✓		2023.10
Mind2Web 42	✘	动作/界面匹配	轨迹	网页		✓		2023.12
AssistGUI|51	✓	图像匹配/设备状态	目标	桌面端		✓		2024.01
MT-Mind2Web|183	✘	动作/界面匹配	轨迹	网页		✓	✓	2024.02
WebVLN 184	✘	网址/文本匹配	目标/图谱	网页	✓	✓		2024.03
WebArena 43	✘	网址/文本匹配/大语言模型	目标/轨迹	网页	✓	✓		2024.04
MMInA 185	✓	文本匹配/大语言模型	轨迹	网页		✓		2024.04
VisualWebBench 186	✘	动作/界面/文本匹配	目标	网页	✓	✓		2024.04
OSWorld 187	✓	设备状态	目标	桌面端		✓		2024.05
移动环境 188	✓	设备状态	目标	移动端		✓		2024.06
VisualWebArena|47|	✓	网址/文本/图像匹配/HTML 状态/大语言模型	目标	网页	✓	✓		2024.06
VideoGUI 189	✓	动作匹配/大语言模型	目标	桌面端		✓		2024.06
MobileAgentBench 41	✓	界面匹配	目标	移动端		✓		2024.06
GUICourse 190	✘	/	/	网页与移动端	✓	✓	✓	2024.06
GUI-WORLD 191 平台	✘ ✓	文本匹配/大语言模型 ✓	目标 ✓	多 2024.06				
GUI Odyssey 192	✘	动作匹配	轨迹	移动端		✓		2024.06
WebCanvas 193	✓	界面/网址匹配	轨迹	网页		✓		2024.07
Spider2-V 194	✓	设备状态	目标	桌面端		✓		2024.07
OmniACT 195	✘	动作匹配	轨迹	桌面端		✓		2024.07
AMEX 196	✘	动作匹配	轨迹	移动端	✓	✓		2024.07
LlamaTouch 197	✓	UI/动作匹配/设备状态	轨迹	移动端		✓		2024.08
AndroidArena 198	✓	动作匹配/大语言模型	轨迹	移动端		✓		2024.08
WindowsAgentArena 199	✓	设备状态	目标	桌面端		✓		2024.09
WebLINX 200	✘	动作/界面/文本匹配	目标	网页		✓	✓	2024.09
AgentStudio 201	✓	设备状态	目标	桌面端		✓		2024.10
B-MOCA 202	✓	设备状态	目标	移动端		✓		2024.10
AndroidWorld|45	✓	设备状态	目标	移动端		✓		2024.10
CRAB|203 平台	✓	UI/图像匹配/设备状态 ✓	图表	多 2024.10				
ScreenPR|204	✘	人类/大语言模型 (LLM)/文本匹配	目标	移动端	✓			2024.10
SPA-BENCH | 205	✓	大语言模型 (LLM)	目标/轨迹	移动端		✓		2024.10
Android 实验室 29	✓	设备状态	设备状态	移动端		✓		2024.10
ANDROIDCONTROL|206|×		动作匹配	轨迹	移动端		✓		2024.11
多界面 (MultiUI)|207	✘	/	/	网页	✓	✓		2024.11
图形用户界面 (GUI) 测试竞技场 208	✓	动作匹配/大语言模型 (LLM)	目标/轨迹	移动端	✓	✓		2024.12
A3 |209	✓	UI/动作匹配/大语言模型 (LLM)	目标/轨迹	移动端		✓		2025.01
WebWalkerQA |210	✓	大语言模型 (LLM)	目标	网页		✓	✓	2025.01
ScreenSpot-Pro 46	✘	界面匹配	目标	桌面端	✓			2025.01
WorldGUI 211	✓	UI 匹配/设备状态	目标	网页 & 桌面		✓		2025.02
UI-Vision 212	✘	UI/动作匹配	目标	桌面端	✓	✓		2025.03

TABLE 1: A Comprehensive Survey of GUI Interaction Datasets (2017-2024): Comparison of Real Environment Support, Evaluation Methods, Platforms, and Task Types (Perception, Operation, and Dialogue)

表 1:GUI 交互数据集 (2017-2024) 综合调研: 真实环境支持、评估方法、平台及任务类型 (感知、操作与对话) 比较

4.1.2 Simulated Environments

4.1.2 模拟环境

Simulated environments replicate the real world to create fully isolated and controllable interaction scenarios, which eliminates variables introduced by dynamic online content, thereby improving the repeatability of evaluations.

模拟环境复制现实世界，创建完全隔离且可控的交互场景，消除动态在线内容带来的变量，从而提升评估的可重复性。

MiniWoB in WoB [175] and MiniWoB++ [177] manually designed small-scale HTML pages to simulate interactions in real-world web tasks. FormWoB in WoB [175] converted four real flight booking websites into web tasks, recording all HTTP requests and responses to create offline approximations of these websites. WebShop [182] scraped product information from amazon.com to create a simulated e-commerce site. WebArena [43] enhanced realism by developing four fully operational, self-hosted web applications, each representing a distinct common domain of the internet. VisualWebArena [47] built upon WebArena [43] by incorporating visual modality information. Most tasks in the MMInA [185] dataset were derived from real websites. However, for tasks requiring image extraction from HTML files, offline and open-source websites were used as substitutes to bypass web protection mechanisms.

WoB [175] 中的 MiniWoB 和 MiniWoB++ [177] 手工设计了小规模 HTML 页面，以模拟现实网页任务中的交互。WoB [175] 中的 FormWoB 将四个真实航班预订网站转换为网页任务，记录所有 HTTP 请求和响应，创建这些网站的离线近似版本。WebShop [182] 从 amazon.com 抓取产品信息，构建了一个模拟电商网站。WebArena [43] 通过开发四个完全可运行的自托管网络应用，增强了真实性，每个应用代表互联网的一个常见领域。VisualWebArena [47] 在 WebArena [43] 基础上加入了视觉模态信息。MMInA [185] 数据集中的大多数任务源自真实网站，但对于需要从 HTML 文件中提取图像的任务，采用了离线且开源的网站作为替代，以绕过网页保护机制。

Despite these advances, applying GUI Agents to real-world scenarios still requires dynamic and authentic environments for evaluation. Online evaluation often involves numerous uncertainties. For instance, flight booking tasks may be closely tied to weather conditions that cannot be accounted for by static evaluation methods. Additionally, factors such as network fluctuations, security verifications, and popup advertisements must also be considered. Avoiding uncertainties entirely would hinder the ability of GUI Agents to handle complex real-world tasks effectively.

尽管取得了这些进展，将 GUI 代理应用于现实场景仍需动态且真实的环境进行评估。在线评估常涉及诸多不确定因素。例如，航班预订任务可能与无法通过静态评估方法考虑的天气状况密切相关。此外，还需考虑网络波动、安全验证和弹窗广告等因素。完全避免不确定性将阻碍 GUI 代理有效处理复杂现实任务的能力。

4.1.3 Real-World Environments

4.1.3 真实环境

In real-world environments, tasks are executed on actual websites or applications, requiring GUI Agents to interact directly with the real world.

在真实环境中，任务在实际网站或应用上执行，要求 GUI 代理直接与现实世界交互。

Benchmarks like MobileAgentBench [41], AndroidWorld [45] and B-MOCA [202] provide evaluation environments for real tasks on mobile devices. OSWorld [187] and Win-dowsAgentArena 199 establish testing frameworks for desktop platforms. CRAB [203] goes a step further by designing cross-platform tasks, such as taking photos with a smartphone and transferring the images to a desktop computer for editing. Constructing web-related tasks is particularly challenging due to the rapid evolution of the online environment, which can render previously annotated tasks obsolete. For instance, many tasks in Mind2Web [42] are no longer functional on corresponding live websites. To address this, WebCanvas [193] offers an efficient maintenance mechanism that quickly identifies the validity of annotated action sequences and key nodes through periodic monitoring and automated alerts, ensuring that tasks remain operable in real-world environments.

MobileAgentBench [41]、AndroidWorld [45] 和 B-MOCA [202] 等基准为移动设备上的真实任务提供评估环境。OSWorld [187] 和 WindowsAgentArena [199] 建立了桌面平台的测试框架。CRAB [203] 更进一步，设计了跨平台任务，如用智能手机拍照并将图像传输到桌面电脑进行编辑。构建与网页相关的任务尤为挑战，因为在线环境快速演变，可能使先前标注的任务失效。例如，Mind2Web [42] 中的许多任务在对应的实时网站上已无法使用。为此，WebCanvas [193] 提供了高效的维护机制，通过定期监控和自动警报快速识别标注动作序列和关键节点的有效性，确保任务在真实环境中持续可用。

Although the inherent uncertainties of real-world environments pose significant challenges to task creation and maintenance, constructing tasks within such environments has become a prevailing research trend. This development is crucial for advancing GUI Agents capabilities.

尽管真实环境固有的不确定性对任务创建和维护构成重大挑战，但在此类环境中构建任务已成为主流研究趋势。这一发展对提升 GUI 代理能力至关重要。

4.2 Task

4.2 任务

Evaluating the performance of GUI Agents requires a clear understanding of the specific tasks they are designed to accomplish. Current benchmarks and datasets for GUI Agents can be categorized into three fundamental task types: perception, operation, and dialogue, each targeting different aspects of agent capabilities.

评估 GUI 代理的性能需要明确其设计完成的具体任务。目前 GUI 代理的基准和数据集可分为三大基本任务类型: 感知、操作和对话，分别针对代理能力的不同方面。

4.2.1 Perception

4.2.1 感知

Perception serves a fundamental cognitive process essential for GUI Agents' interaction with graphical user interfaces, encompassing the recognition, extraction, and interpretation of interface elements. This comprehensive

understanding of the interface architecture provides the foundation for subsequent decision-making processes and user interactions. The research community has developed numerous benchmarks and datasets specifically annotated for perception tasks. WebArena [43] and VisualWebArena [47] feature web information retrieval challenges that test agents' visual comprehension capabilities. WebSRC [179] introduces Web-Based Structural Reading Comprehension, which requires agents to analyze both spatial layouts and logical hierarchies of web pages to respond to queries accurately. ScreenPR [204] evaluates agents' ability to generate precise explanations for specific screen regions based on user interaction points. MultiUI [207] focuses on embedded image interpretation within web environments. VisualWebBench [186] assesses webpage comprehension through captioning tasks that measure an agent's capacity to synthesize and summarize content holistically. GUICourse [190] contributes GUIEnv, a comprehensive dataset for optical character recognition and visual grounding. Finally, AMEX [196] enhances the evaluation landscape by providing detailed screen descriptions coupled with functional annotations for interface components.

感知是 GUI 代理与图形用户界面交互的基础认知过程，涵盖界面元素的识别、提取与理解。对界面结构的全面理解为后续决策和用户交互奠定基础。研究界已开发众多专门标注感知任务的基准和数据集。WebArena [43] 和 VisualWebArena [47] 设有网页信息检索挑战，测试代理的视觉理解能力。WebSRC [179] 引入基于网页的结构化阅读理解，要求代理分析网页的空间布局和逻辑层级以准确回答查询。ScreenPR [204] 评估代理基于用户交互点生成特定屏幕区域精确解释的能力。MultiUI [207] 聚焦网页环境中的嵌入图像解读。VisualWebBench [186] 通过字幕任务评测网页理解，衡量代理综合总结内容的能力。GUICourse [190] 贡献了 GUIEnv，一个涵盖光学字符识别和视觉定位的综合数据集。最后，AMEX [196] 通过提供详细的屏幕描述及界面组件功能注释，丰富了评估体系。

4.2.2 Operation

4.2.2 操作

Operation tasks represent a critical stage in GUI Agents interactions, where agents execute specific actions to achieve designated goals. These tasks encompass scenarios such as complex software operations and web navigation, rigorously assessing an agent's planning and execution capabilities.

操作任务是 GUI 代理交互中的关键阶段，代理执行特定动作以达成指定目标。这些任务涵盖复杂软件操作和网页导航等场景，严格考察代理的规划与执行能力。

The operational complexity spectrum begins with simple single-step operations, where agents address straightforward, well-defined tasks. For instance, MultiUI [207] provides single-step operation prediction tasks that test basic action selection. Similarly, the GUIAct dataset within GUICourse 190 offers 67,000 single-step operation instructions specifically designed to evaluate an agent's fundamental decision-making capabilities. As complexity increases, many benchmarks focus on multi-step operations within specific software environments. AssistGUI [51] emphasizes productivity scenarios by collecting 100 tasks across nine commonly used applications, including Premiere Pro, After Effects, and PowerPoint. The evaluation landscape extends into professional domains through benchmarks like Spider2-V [194], which features data science and engineering software tasks in platforms such as BigQuery and Airbyte, and VideoGUI [189], which concentrates on video editing in professional tools like Adobe Photoshop and Stable Diffusion.

操作复杂度谱从简单的单步操作开始，代理处理直接且定义明确的任务。例如，MultiUI [207] 提供了单步操作预测任务，用于测试基本的动作选择能力。同样，GUICourse 190 中的 GUIAct 数据集提供了 67,000 条单步操作指令，专门用于评估代理的基本决策能力。随着复杂度的增加，许多基准测试聚焦于特定软件环境中的多步操作。AssistGUI [51] 通过收集涵盖 Premiere Pro、After Effects 和 PowerPoint 等九个常用应用的 100 个任务，强调生产力场景。评估范围扩展到专业领域，如 Spider2-V [194]，涵盖 BigQuery 和 Airbyte 等平台上的数据科学与工程软件任务，以及 VideoGUI [189]，专注于 Adobe Photoshop 和 Stable Diffusion 等专业视频编辑工具。

Another critical dimension of operational complexity emerges in web-based environments, where navigation tasks demand intricate interaction sequences across interconnected pages. WebShop [182] simulates e-commerce scenarios requiring product search and selection, while MMInA [185] advances the challenge through multi-hop navigation tasks where agents must traverse multiple independent websites and integrate information from diverse sources to accomplish complex goals. Comprehensive ecosystem evaluation spanning both application-specific and web-based operations is provided by benchmarks such as Mobile-Env [188], OSWorld [187], AndroidWorld [45], and WindowsAgentArena [199], which cover broader scenarios including application operations, system settings, programming tasks, and multimedia applications. The GUI Testing Arena [208] requires the Agent to discover GUI defects through exploration.

操作复杂度的另一个关键维度出现在基于网页的环境中，导航任务要求在相互关联的页面间执行复杂的交互序列。WebShop [182] 模拟电商场景，需进行产品搜索和选择；而 MMInA [185] 通过多跳导航任务提升挑战，代理必须穿越多个独立网站并整合多源信息以完成复杂目标。涵盖应用特定和基于网页操作的综合生态系统评估由 Mobile-Env [188]、OSWorld [187]、AndroidWorld [45] 和 WindowsAgentArena [199] 等基准提供，涵盖应用操作、系统设置、编程任务和多媒体应用等更广泛场景。GUI Testing Arena [208] 要求代理通过探索发现 GUI 缺陷。

4.2.3 Dialogue

4.2.3 对话

Most existing benchmarks and datasets assume that agents can complete tasks end-to-end upon receiving instructions, without requiring interaction with users during execution. However, in real-world scenarios, users may need to modify or augment their requirements mid-task, or agents may need to seek confirmation from users during critical decision-making stages. To address this, some works extend tasks to dialogue-based scenarios, aligning closer with real-world application demands.

大多数现有基准和数据集假设代理在接收指令后能够端到端完成任务，无需在执行过程中与用户交互。然而，在现实场景中，用户可能需要在任务中途修改或补充需求，或者代理在关键决策阶段需向用户寻求确认。为此，一些工作将任务扩展到基于对话的场景，更贴近实际应用需求。

For instance, the GUIChat modules in MT-Mind2Web [183], WebLINX [200], and GUICourse [190] design multiturn dialogue tasks for web navigation and comprehension. META-GUI [181] provides task execution trajectories and associated dialogue data on the Android platform. GUI-WORLD [191] further expands to multi-platform environments, encompassing dialogue-based data for dynamic GUI understanding.

例如，MT-Mind2Web [183]、WebLINX [200] 和 GUICourse [190] 中的 GUIChat 模块设计了用于网页导航和理解的多轮对话任务。META-GUI [181] 提供了 Android 平台上的任务执行轨迹及相关对话数据。GUI-WORLD [191] 进一步扩展到多平台环境，涵盖动态 GUI 理解的基于对话的数据。

4.3 Evaluation Strategy

4.3 评估策略

Effective evaluation strategies are essential for accurately measuring the performance of GUI Agents across different tasks and environments. Through our literature review, we identify two predominant approaches to evaluation: trajectory-based methods that focus on the sequence of actions taken, and goal-oriented approaches that emphasize final outcomes regardless of the specific path. Figure 9 illustrates the fundamental differences between these evaluation strategies, highlighting how each method assesses agent performance from different perspectives. The trajectory-based methods track step-by-step actions against predefined paths, while goal-oriented approaches verify whether the final objective has been accomplished regardless of the specific steps taken.

有效的评估策略对于准确衡量 GUI 代理在不同任务和环境中的表现至关重要。通过文献综述，我们识别出两种主要的评估方法: 基于轨迹的方法侧重于动作序列，目标导向的方法强调最终结果而不拘泥于具体路径。图 9 展示了这两种评估策略的基本差异，突出各自从不同角度评估代理性能的方式。基于轨迹的方法跟踪与预定义路径的逐步动作匹配，而目标导向方法则验证最终目标是否达成，无论采取了哪些具体步骤。

4.3.1 Trajectory-based methods

4.3.1 基于轨迹的方法

Trajectory-based methods focus on analyzing the sequence of actions or states generated by an agent during task execution. Step-wise Action Match, a widely adopted evaluation approach [49, 181], compares the action sequence generated by the agent with ground-truth to assess task completion. A task is considered successfully completed if the two sequences match exactly. However, this method may yield unfair evaluation outcomes since agents often require exploratory steps during task execution. If the agent performs actions not accounted for in the dataset, the task is deemed a failure.

基于轨迹的方法侧重分析代理在任务执行过程中生成的动作或状态序列。广泛采用的逐步动作匹配评估方法 [49, 181] 将代理生成的动作序列与真实序列进行比较，以评估任务完成情况。若两序列完全匹配，则任务视为成功完成。然而，该方法可能导致不公平的评估结果，因为代理在执行任务时常需探索性步骤。若代理执行了数据集中未涵盖的动作，则任务被判定为失败。

To address this limitation, AndroidArena [198] introduced the use of the Longest Common Subsequence (LCS) to compare sequences. Unlike traditional step-wise matching, LCS accommodates exploratory and redundant actions by tolerating deviations. Specifically, a match is considered successful if the ground-truth is a subsequence of the agent-generated sequence. Despite its advantages, such methods often lead to high false-negative rates in real-world evaluations. Annotators cannot exhaustively account for all possible execution paths, and predefined

datasets typically provide only a single reference path. Given that multiple alternative solutions may effectively accomplish the task, human verification scores are often higher than those produced by trajectory-based methods, as noted in prior studies [49, 197].

为解决此限制，AndroidArena [198] 引入了最长公共子序列 (LCS) 用于序列比较。与传统的逐步匹配不同，LCS 通过容忍偏差，适应探索性和冗余动作。具体而言，若真实序列是代理生成序列的子序列，则视为匹配成功。尽管如此，此类方法在实际评估中常导致较高的假阴性率。标注者无法穷尽所有可能的执行路径，且预定义数据集通常仅提供单一参考路径。鉴于多种替代方案均可有效完成任务，人工验证得分往往高于基于轨迹的方法，正如先前研究所指出 [49, 197]。

To mitigate this issue, LlamaTouch [197], WebCanvas [193] and WebArena [43] focus solely on the critical nodes within the sequence. Critical nodes refer to actions or states that must be executed or encountered regardless of the method employed. A task is considered complete if the agent's trajectory contains all the annotated critical states. This approach adapts well to dynamic real-world environments and reduces the false-negative problem prevalent in earlier evaluation strategies.

为缓解该问题，LlamaTouch [197]、WebCanvas [193] 和 WebArena [43] 仅关注序列中的关键节点。关键节点指无论采用何种方法，必须执行或遇到的动作或状态。若代理轨迹包含所有标注的关键状态，则任务视为完成。该方法适应动态现实环境，减少了早期评估策略中普遍存在的假阴性问题。

In summary, trajectory-based methods emphasize the process of task execution by the agent. While they suffer from reliability issues in evaluation outcomes, they enable researchers to design fine-grained metrics, such as accuracy for specific action categories, by leveraging trajectory details.

综上，基于轨迹的方法强调代理的任务执行过程。尽管其评估结果存在可靠性问题，但通过利用轨迹细节，研究者能够设计细粒度指标，如特定动作类别的准确率。

4.3.2 Goal-oriented approaches

4.3.2 目标导向方法

Goal-oriented approaches prioritize assessing whether an agent achieves the final objective, rather than the specific steps taken to do so. For tasks with straightforward evaluation criteria, completion can often be determined by a single signal match. For instance, GUI-WORLD [191] evaluates task success by comparing the similarity between the agent's output text and the ground-truth text. Mobile-Env 188] extends this approach by incorporating multiple signals and defining evaluation rules in a logic expression style. This allows flexibly handling scenarios requiring either the satisfaction of multiple conditions simultaneously or the fulfillment of any single condition.

目标导向方法优先评估代理是否达成最终目标，而非具体采取的步骤。对于评估标准简单的任务，完成情况通常可通过单一信号匹配判定。例如，GUI-WORLD [191] 通过比较代理输出文本与真实文本的相似度来评估任务成功。Mobile-Env [188] 在此基础上引入多重信号，并以逻辑表达式风格定义评估规则，从而灵活处理需同时满足多条件或满足任一条件的场景。

For more complex and diverse tasks, such as system configuration or software operation, relying solely on surface-level signals like UI elements or screenshots often lacks robustness. For example, in a task involving editing

a note and saving it, pressing the ”Save” button might trigger only a brief success notification without causing any noticeable changes in the UI state. Furthermore, surface-level signals can sometimes be inaccessible. For instance, using UIAutomator may fail to dump the view hierarchy for dynamic screen content. To enhance the reliability of success evaluations, MobileAgentBench [41] leverages Android Accessibility Service to capture application events (e.g., click events) as supplementary signals. This ensures a more accurate assessment of task success. Other approaches, such as MOBILE-ENV [188], B-MOCA [202] and AndroidWorld [45], utilize the Android Debug Bridge (ADB) to access underlying system states and generate reward signals. By gaining full access to system resources including the file system, application databases, and system settings, these methods provide a more comprehensive evaluation. Additionally, benchmarks such as OSWorld [187] and Win-dowsAgentArena [199] extract critical components from the system’s final state and design task-specific evaluation scripts. Compared to surface-level signal matching, this strategy achieves significantly higher accuracy in determining task success.

对于更复杂多样的任务，如系统配置或软件操作，仅依赖界面元素或截图等表层信号往往缺乏鲁棒性。例如，在编辑并保存笔记的任务中，点击“保存”按钮可能仅触发短暂的成功通知，而界面状态无明显变化。此外，表层信号有时不可获取，如使用 UIAutomator 可能无法导出动态屏幕内容的视图层级。为提升成功评估的可靠性，MobileAgentBench [41] 利用 Android 辅助功能服务捕获应用事件 (如点击事件) 作为补充信号，确保更准确的任务成功判定。其他方法如 MOBILE-ENV [188]、B-MOCA [202] 和 AndroidWorld [45] 则通过 Android 调试桥 (ADB) 访问底层系统状态并生成奖励信号。通过全面访问文件系统、应用数据库及系统设置等资源，这些方法提供了更全面的评估。此外，OSWorld [187] 和 WindowsAgentArena [199] 等基准从系统最终状态提取关键组件，设计任务专用评估脚本。相比表层信号匹配，该策略在判定任务成功方面准确率显著提升。

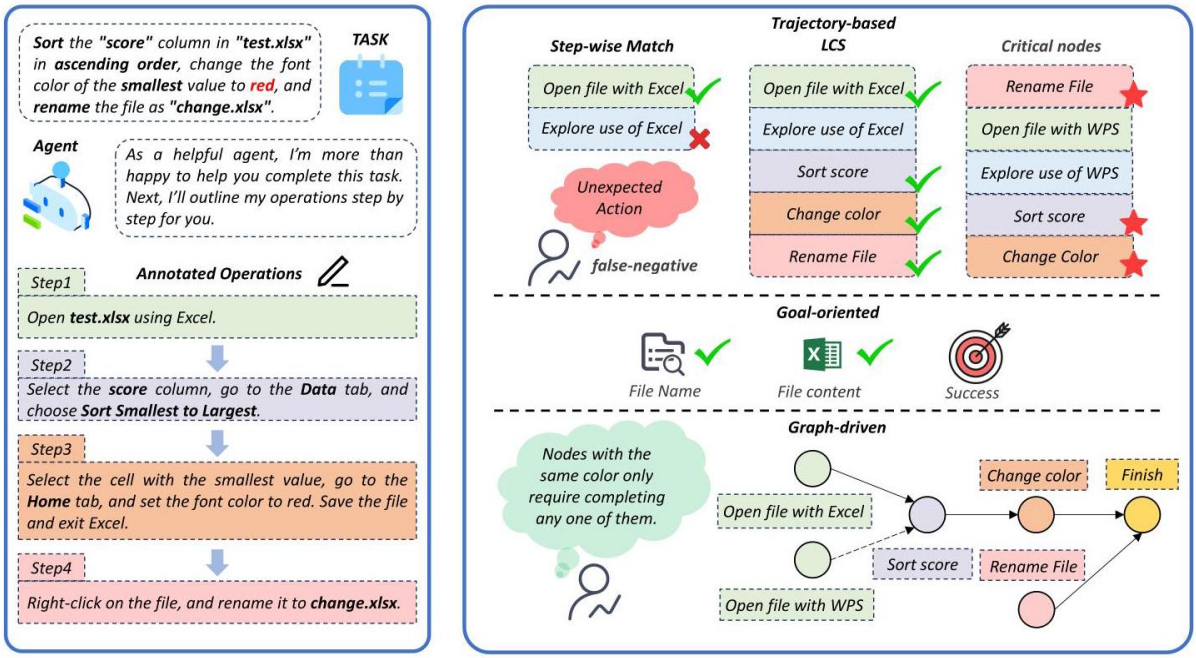

Fig. 9: Comparison between different evaluation strategies. The left shows a sample Excel task with step-by-step operations, while the right contrasts three evaluation approaches - Trajectory-based (matching steps and critical nodes), Goal-oriented (assessing final outcomes), and Graph-driven (representing operations as connected nodes with alternative paths).

图 9: 不同评估策略的比较。左侧展示一个带有逐步操作的 Excel 任务示例，右侧对比三种评估方法——基于轨迹 (匹配步骤和关键节点)、目标导向 (评估最终结果) 和图驱动 (将操作表示为连接节点及备选路径)。

4.3.3 Graph-driven techniques

4.3.3 图驱动技术

Recent research has introduced graph-driven evaluation techniques as a promising middle ground between trajectory-based and goal-oriented approaches. Crab [203] models tasks as directed graphs, encoding both sequential and parallel relationships between sub-nodes. In this framework, multiple paths to the target node elegantly represent alternative solutions, allowing for diverse yet valid execution strategies. The graph-based evaluator systematically decomposes complex tasks into verifiable sub-goals, with each sub-goal represented as a node and associated with a specific validation function that confirms its completion. Similarly, WebVLN [184] applies graph theory to Vision-and-Language Navigation tasks by constructing a navigation graph where webpages form nodes and hyperlinks serve as edges. Task performance is quantitatively assessed by measuring the shortest path distance between the agent's final destination and the target webpage in the navigation graph. By combining the structural rigor of trajectory analysis with the flexibility to accommodate multiple solution paths, graph-driven methods offer a balanced evaluation approach. They provide the granular assessment metrics characteristic of trajectory-based methods while maintaining the solution flexibility inherent in goal-oriented approaches, effectively addressing limitations of both traditional evaluation paradigms.

近期研究提出图驱动评估技术，作为轨迹导向与目标导向方法之间的有力折中方案。Crab [203] 将任务建模为有向图，编码子节点间的顺序与并行关系。在此框架中，通向目标节点的多条路径优雅地表示备选方案，允许多样且有效的执行策略。基于图的评估器系统地将复杂任务分解为可验证的子目标，每个子目标由节点表示，并关联特定的验证函数以确认完成情况。类似地，WebVLN [184] 将图论应用于视觉与语言导航任务，构建导航图，其中网页为节点，超链接为边。通过测量代理最终目的地与目标网页在导航图中的最短路径距离，定量评估任务表现。图驱动方法结合了轨迹分析的结构严谨性与多解路径的灵活性，提供平衡的评估方案。它既具备轨迹导向方法的细粒度评估指标，又保持目标导向方法的解法灵活性，有效克服传统评估范式的局限。

4.4 Challenges

4.4 挑战

Current GUI Agents benchmarks and datasets face several critical challenges that impact their effectiveness and applicability. The evaluation process encounters significant cost barriers in terms of computational resources, time consumption, and financial investment, particularly for complex multi-step tasks requiring specialized models or human judgment. Additionally, existing benchmarks lack sufficient diversity in their content, interface representation, and cultural context, often being limited to English language, optimized interfaces, and regionally-specific examples. Finally, these evaluation environments raise important security and privacy concerns as they require extensive system access and may potentially expose sensitive information.

当前 GUI 代理基准和数据集面临多项关键挑战，影响其有效性和适用性。评估过程在计算资源、时间消耗和资金投入方面存在显著成本障碍，尤其是对需专用模型或人工判断的复杂多步骤任务。此外，现有基准在内容、界面表现和文化背景上缺乏足够多样性，通常局限于英语、优化界面及特定区域示例。最后，这些评估环境涉及广泛系统访问，带来重要的安全与隐私问题，可能暴露敏感信息。

4.4.1 Cost

4.4.1 成本

The evaluation of GUI Agent benchmarks faces significant cost challenges in terms of time, computational resources, and financial investment. For example, completing each task in OSWorld [187] requires approximately 10-20 steps. Assuming that each step involves processing through local models and cloud API calls, taking 10-30 seconds per step, the serial evaluation of hundreds of tasks can take a day or longer. OSWorld [187] mitigates this by allowing parallel simulation on a single machine to accelerate the evaluation process. WindowsAgentArena [199] introduces native cloud parallelization to further reduce memory and CPU demands.

GUI 代理基准的评估面临时间、计算资源和资金投入的显著成本挑战。例如，完成 OSWorld [187] 中的每个任务约需 10-20 步。假设每步需通过本地模型和云端 API 调用处理，耗时 10-30 秒，串行评估数百个任务可能需时一天或更长。OSWorld [187] 通过允许单机并行模拟加速评估过程。WindowsAgentArena [199] 引入原生云端并行化，进一步降低内存和 CPU 需求。

Additionally, some benchmarks involve expensive evaluations, particularly those requiring human judgment, such as ScreenPR [204]. Benchmarks like VideoGUI [189] and GUI-WORLD [191] rely on commercial LLMs for evaluation, adding monetary costs from API calls. Benchmark deployment also introduces a learning curve. User-friendly benchmarks, such as MobileAgentBench [41], offer evaluation parameter configuration with as few as ten lines of code. However, while this may be straightforward for researchers in the AI/ML community, it remains challenging for individuals without a technical background.

此外，一些基准测试涉及昂贵的评估，尤其是那些需要人工判断的，如 ScreenPR [204]。像 VideoGUI [189] 和 GUI-WORLD [191] 这样的基准测试依赖商业大型语言模型 (LLM) 进行评估，增加了 API 调用的费用。基准测试的部署也带来了学习曲线。用户友好的基准测试，如 MobileAgentBench [41]，只需十行代码即可配置评估参数。然而，虽然这对 AI/机器学习社区的研究人员来说可能很简单，但对于没有技术背景的个人来说仍然具有挑战性。

4.4.2 Diversity

4.4.2 多样性

The limited diversity of current datasets and benchmarks poses a significant challenge to developing robust and generalizable GUI Agents. Some datasets provide only textual content, failing to exploit multimodal information. Most datasets are currently limited to English, although multilingual models can be employed for translation. In

Mind2Web [42], the dataset primarily consists of websites frequently visited by U.S. users, which may not fully capture the browsing habits of global users.

当前数据集和基准测试的多样性有限，这对开发稳健且具备泛化能力的 GUI 代理构成了重大挑战。一些数据集仅提供文本内容，未能利用多模态信息。大多数数据集目前仅限于英语，尽管可以使用多语言模型进行翻译。在 Mind2Web [42] 中，数据集主要由美国用户频繁访问的网站组成，可能无法全面反映全球用户的浏览习惯。

AndroidWorld [45] includes only open-source applications with over one million downloads. These popular applications often have highly optimized user interfaces to enhance user experience, offering more features and shortcuts. This makes it more challenging to test agents on less-optimized interfaces. AITW [49] captures UI drift across different versions of Android for Google applications. A promising solution is the environment randomization functionality in B-MoCA [202], which adjusts icon positions and sizes, wallpapers, language settings, and device types to simulate diverse real-world scenarios.

AndroidWorld [45] 仅包含下载量超过一百万的开源应用。这些流行应用通常拥有高度优化的用户界面，以提升用户体验，提供更多功能和快捷方式。这使得在较少优化的界面上测试代理更加困难。AITW [49] 捕捉了谷歌应用在不同 Android 版本间的界面漂移。一个有前景的解决方案是 B-MoCA [202] 中的环境随机化功能，该功能调整图标位置和大小、壁纸、语言设置及设备类型，以模拟多样的真实场景。

4.4.3 Security

4.4.3 安全性

The proliferation of GUI Agent benchmarks and datasets raises critical security and privacy concerns that must be addressed for responsible development and deployment. Collected data may contain sensitive and private information such as personal details, financial records, browsing histories, and authentication credentials. If datasets are misused, they could facilitate malicious purposes, such as bypassing security measures or enabling harmful activities.

GUI 代理基准测试和数据集的激增带来了必须解决的关键安全和隐私问题，以确保负责任的开发和部署。收集的数据可能包含敏感和私密信息，如个人资料、财务记录、浏览历史和认证凭据。如果数据集被滥用，可能助长恶意目的，如绕过安全措施或实施有害活动。

The issue is compounded by the fact that GUI Agents require extensive access privileges to function effectively across applications and platforms. Benchmarks like OS-World [187] and WindowsAgentArena [199] operate with system-level permissions that could be exploited if not properly secured. Furthermore, as agents become more capable of executing complex sequences of actions, the potential for security vulnerabilities increases. For instance, agents designed to automate form-filling could be repurposed for credential harvesting or phishing attacks if their datasets are compromised.

问题因 GUI 代理需要广泛访问权限以跨应用和平台有效运行而加剧。像 OS-World [187] 和 WindowsAgentArena [199] 这样的基准测试以系统级权限运行，若未妥善保护，可能被利用。此外，随着代理执行复杂操作序列的能力增强，安全漏洞的潜在风险也增加。例如，设计用于自动填写表单的代理若数据集被泄露，可能被改用于凭证窃取或网络钓鱼攻击。

Current security measures in benchmarks often focus on isolated environments but lack comprehensive threat modeling for real-world deployment scenarios. The absence of standardized security protocols for GUI Agent evaluation environments represents a significant gap in the field. Future benchmarks must incorporate robust anonymization techniques, permission management systems, and adversarial testing frameworks to ensure that GUI Agents can be evaluated thoroughly without compromising user security or privacy. Additionally, transparent documentation of data collection methodologies and explicit consent mechanisms are essential for maintaining ethical standards in benchmark development.

当前基准测试中的安全措施通常侧重于隔离环境，但缺乏针对真实部署场景的全面威胁建模。GUI 代理评估环境缺乏标准化安全协议是该领域的重要空白。未来基准测试必须整合强有力的匿名化技术、权限管理系统和对抗性测试框架，确保在不危及用户安全或隐私的前提下，能够全面评估 GUI 代理。此外，透明的数据收集方法文档和明确的用户同意机制对于维护基准测试开发的伦理标准至关重要。

5 FUTURE DIRECTIONS AND CHALLENGES

5 未来方向与挑战

We have systematically organized GUI Agents into four core modules and examined existing datasets and benchmarks, we now discuss key challenges and future directions in this field. We explore limitations in data collection and benchmark development (§5.1), challenges in multimodal perception and visual grounding (§5.2), complexities in strategic planning (§5.3), and the potential of reinforcement learning approaches (§5.4) for enhancing GUI Agent capabilities in complex, real-world scenarios.

我们系统地将 GUI 代理划分为四个核心模块，并审视了现有数据集和基准测试，现讨论该领域的关键挑战和未来方向。我们探讨数据收集和基准开发的局限性 (§5.1)、多模态感知与视觉定位的挑战 (§5.2)、策略规划的复杂性 (§5.3)，以及强化学习方法 (§5.4) 在提升 GUI 代理处理复杂真实场景能力中的潜力。

5.1 Data Collection and Benchmark Development

5.1 数据收集与基准开发

Data collection and comprehensive evaluation of GUI Agents remain critical challenges for future research. Current GUI Agent datasets primarily consist of multi-step interaction sequences, but existing datasets often fail to reflect real-world scenarios. For instance, they rarely capture how agents should resume tasks after interruptions, handle unexpected system prompts, or adapt to changing interface states. Future research should focus on developing more realistic multi-step datasets that incorporate these challenging scenarios, such as task resumption after

interruptions, adaptation to dynamic interface changes, and handling of unexpected system responses.

数据收集和对 GUI 代理的全面评估仍是未来研究的关键挑战。当前 GUI 代理数据集主要由多步骤交互序列组成，但现有数据集往往未能反映真实场景。例如，极少捕捉代理在中断后如何恢复任务、处理意外系统提示或适应界面状态变化。未来研究应聚焦于开发更真实的多步骤数据集，涵盖任务中断后的恢复、动态界面变化的适应以及意外系统响应的处理等挑战场景。

The development of efficient methods to gather high-quality, diverse interaction data that accurately captures real-world usage patterns is essential. This includes creating benchmarks that test an agent's ability to maintain task context across interruptions and environmental changes. Equally important is establishing standardized evaluation frameworks that can holistically assess GUI Agent performance across dimensions such as accuracy, robustness, efficiency, and contextual understanding. Addressing these challenges will be crucial for advancing GUI Agent capabilities beyond simplified testing environments to truly functional real-world applications.

开发高效方法以收集高质量、多样化的交互数据，准确反映真实使用模式至关重要。这包括创建测试代理在中断和环境变化中维持任务上下文能力的基准。同样重要的是建立标准化评估框架，能够全面衡量 GUI 代理在准确性、鲁棒性、效率和上下文理解等维度的表现。解决这些挑战对于推动 GUI 代理能力超越简化测试环境，迈向真正功能性真实应用至关重要。

5.2 Multimodal Perception and Visual Grounding

5.2 多模态感知与视觉定位

Current GUI Agents still face significant limitations in screen understanding and visual grounding [8, 46]. The fundamental challenge stems from the unique properties of GUI images—unlike natural images [79], they feature hierarchical layouts with numerous small elements that require precise comprehension. GUI screenshots present complex resolution challenges: while the overall image is large, individual interactive elements like icons and buttons are often extremely small, making accurate detection difficult [8, 46]. This resolution discrepancy frequently leads multimodal language models to hallucinate interface elements or make critical localization errors during navigation tasks, severely limiting their reliability in real-world applications. Future research can focus on developing specialized approaches to address these visual perception challenges. Promising directions include chain-of-thought reasoning to enhance contextual understanding of interface hierarchies and the integration of specialized visual parsing tools that can decompose complex screens into manageable components. Methods like SoM [52] and OmniParser [56] have demonstrated impressive results by tackling these specific limitations. While tool-based approaches offer immediate improvements, the ultimate goal remains advancing toward end-to-end GUI Agents that seamlessly integrate perception and action. Achieving this will require novel architectures and training paradigms specifically designed to handle the unique visual characteristics of GUI environments, potentially incorporating adaptive attention mechanisms that can efficiently process both macro-level layout and micro-level interface details.

当前的 GUI 代理在屏幕理解和视觉定位方面仍面临显著限制 [8, 46]。根本挑战源于 GUI 图像的独特属性——与自然图像 [79] 不同，它们具有层级布局，包含大量需要精确理解的小元素。GUI 截图呈现复杂的分辨率挑战: 整体图像较大，但图标和按钮等单个交互元素通常非常小，导致准确检测困难 [8, 46]。这种分辨率差异常使多模态语言模型在导航任务中产生界面元素的幻觉或发生关键定位错误，严重限制了其在实际应用中的可靠性。未来研究可聚焦于开发专门方法以应对这些视觉感知挑战。前景看好的方向包括利用链式思维推理增强对界面层级的上下文理解，以及整合专门的视觉解析工具，将复杂屏幕分解为可管理的组件。诸如 SoM[52] 和 OmniParser[56] 等方法通过解决这些特定限制已展示出令人印象深刻的成果。尽管基于工具的方法能带来即时改进，最终目标仍是实现端到端的 GUI 代理，能够无缝整合感知与动作。实现这一目标需要专门设计的新型架构和训练范式，以处理 GUI 环境独特的视觉特性，可能包括能够高效处理宏观布局和微观界面细节的自适应注意力机制。

5.3 Strategic Planning and Decision Making

5.3 战略规划与决策制定

GUI Agents face critical planning and decision-making challenges when navigating complex interface environments. When unexpected interruptions occur—such as system pop-ups, network errors, or dynamic interface changes—current agents often fail catastrophically, either terminating tasks prematurely or becoming trapped in loops of repeated failed actions. This problem is exacerbated by the unpredictability of real-world interfaces, which frequently update their elements and information architecture, creating a fundamental mismatch between training environments and deployment scenarios.

GUI 代理在复杂界面环境中导航时面临关键的规划和决策挑战。当出现意外中断——如系统弹窗、网络错误或动态界面变化——当前代理常常出现灾难性失败，要么过早终止任务，要么陷入重复失败动作的循环。现实界面的不可预测性加剧了这一问题，界面元素和信息架构频繁更新，导致训练环境与部署场景之间存在根本性不匹配。

Beyond handling interruptions, GUI Agents struggle with fundamental decision-making challenges in uncertain environments. When encountering ambiguous UI elements or multiple viable interaction paths, current agents lack sophisticated reasoning capabilities to determine optimal actions that align with specific user contexts and preferences. The ephemeral nature of modern interfaces further complicates this challenge, as transient elements such as tooltips, modal dialogs, and animated transitions create a constantly shifting interaction landscape that static planning models fail to navigate effectively. These temporal aspects of GUI environments demand more sophisticated understanding of state transitions and timing constraints than current perception frameworks can provide.

除了应对中断，GUI 代理在不确定环境中还面临基本的决策挑战。当遇到模糊的 UI 元素或多条可行的交互路径时，当前代理缺乏复杂的推理能力来确定符合特定用户上下文和偏好的最优动作。现代界面的短暂性进一步加剧了这一挑战，诸如工具提示、模态对话框和动画过渡等瞬时元素构建了不断变化的交互环境，静态规划模型难以有效导航。这些 GUI 环境的时间性特征要求比当前感知框架更复杂的状态转移和时序约束理解。

5.4 Reinforcement Learning for GUI Agents

5.4 GUI 代理的强化学习

With the development of DeepSeek R1 [28], researchers have recognized the potential of reinforcement learning with rule-based rewards. An increasing number of studies building upon the R1 paradigm have achieved notable advances in the field. GUI Agents require multi-step reasoning and decision making in practical scenarios, which inherently aligns with the exploratory nature of reinforcement learning, making it a promising approach for enhancing these agents' capabilities.

随着 DeepSeek R1[28] 的发展，研究者认识到基于规则奖励的强化学习潜力。越来越多基于 R1 范式的研究在该领域取得显著进展。GUI 代理在实际场景中需要多步推理和决策，这与强化学习的探索性质本质契合，使其成为提升代理能力的有前景方法。

Whether using reinforcement learning to train GUI Agents can achieve better performance remains an open question worth exploring. The high-dimensional state space represented by screen images presents computational challenges, while defining appropriate reward functions that balance task completion with user experience considerations remains difficult. Despite these challenges, the integration of reinforcement learning with GUI Agents represents a promising direction that could potentially lead to more adaptive agents capable of handling complex real-world interfaces.

是否利用强化学习训练 GUI 代理能实现更优性能仍是值得探索的开放问题。屏幕图像所代表的高维状态空间带来计算挑战，而定义兼顾任务完成与用户体验的合适奖励函数亦困难重重。尽管如此，将强化学习与 GUI 代理结合代表了一个有潜力的方向，可能促使代理更具适应性，能够处理复杂的现实界面。

6 CONCLUSION

6 结论

In this survey, we have systematically reviewed the recent advancements in GUI Agents research, organizing their architecture into four fundamental modules: perception, exploration, planning, and interaction. We provided detailed explanations of each module, examining how they collectively enable agents to interpret, navigate, and manipulate graphical user interfaces across diverse platforms. Additionally, we explored various application scenarios of GUI Agents and comprehensively described existing benchmarks, highlighting their potential challenges related to computational cost, linguistic and interface diversity, and security concerns. Finally, we identified the remaining challenges in current GUI Agent systems and outlined promising future research directions that could advance this rapidly evolving field.

在本综述中，我们系统回顾了 GUI 代理研究的最新进展，将其架构划分为感知、探索、规划和交互四个基本模块。我们详细阐述了每个模块，探讨它们如何协同使代理能够在多样平台上解读、导航和操作图形用户界面。此外，我们考察了 GUI 代理的多种应用场景，全面描述了现有基准，强调了其在计算成本、语言与界面多样性及安全性方面的潜在挑战。最后，我们指出了当前 GUI 代理系统的剩余难题，并勾勒了有望推动该快速发展领域前进的未来研究方向。

REFERENCES

参考文献

[1] S. Wang, W. Liu, J. Chen, Y. Zhou, W. Gan, X. Zeng, Y. Che, S. Yu, X. Hao, K. Shao, B. Wang, C. Wu, Y. Wang, R. Tang, and J. Hao, ”Gui agents with foundation models: A comprehensive survey,” 2025. [Online]. Available: https://arxiv.org/abs/2411.04890

S. Wang, W. Liu, J. Chen, Y. Zhou, W. Gan, X. Zeng, Y. Che, S. Yu, X. Hao, K. Shao, B. Wang, C. Wu, Y. Wang, R. Tang, and J. Hao, ”Gui agents with foundation models: A comprehensive survey,” 2025. [在线]. 可获取: https://arxiv.org/abs/2411.04890

[2] C. Zhang, S. He, J. Qian, B. Li, L. Li, S. Qin, Y. Kang, M. Ma, G. Liu, Q. Lin, S. Rajmohan, D. Zhang, and Q. Zhang, ”Large language model-brained gui agents: A survey,” 2025. [Online]. Available: https://arxiv.org/abs/2411.18279

C. Zhang, S. He, J. Qian, B. Li, L. Li, S. Qin, Y. Kang, M. Ma, G. Liu, Q. Lin, S. Rajmohan, D. Zhang, and Q. Zhang, ”Large language model-brained gui agents: A survey,” 2025. [在线]. 可获取: https://arxiv.org/abs/2411.18279

[3] D. Nguyen, J. Chen, Y. Wang, G. Wu, N. Park, Z. Hu, H. Lyu, J. Wu, R. Aponte, Y. Xia, X. Li, J. Shi, H. Chen, V. D. Lai, Z. Xie, S. Kim, R. Zhang, T. Yu, M. Tanjim, N. K. Ahmed, P. Mathur, S. Yoon, L. Yao, B. Kveton, T. H. Nguyen, T. Bui, T. Zhou, R. A. Rossi, and F. Dernoncourt, ”Gui agents: A survey,” 2024. [Online]. Available: https://arxiv.org/abs/2412.13501

D. Nguyen, J. Chen, Y. Wang, G. Wu, N. Park, Z. Hu, H. Lyu, J. Wu, R. Aponte, Y. Xia, X. Li, J. Shi, H. Chen, V. D. Lai, Z. Xie, S. Kim, R. Zhang, T. Yu, M. Tanjim, N. K. Ahmed, P. Mathur, S. Yoon, L. Yao, B. Kveton, T. H. Nguyen, T. Bui, T. Zhou, R. A. Rossi, 和 F. Dernoncourt, “Gui agents: A survey(GUI 代理: 综述),”2024. [在线]. 可用: https://arxiv.org/abs/2412.13501

[4] J. Yang, R. Tan, Q. Wu, R. Zheng, B. Peng, Y. Liang, Y. Gu, M. Cai, S. Ye, J. Jang, Y. Deng, L. Liden, and J. Gao, ”Magma: A foundation model for multimodal ai agents,” 2025. [Online]. Available: https://arxiv.org/abs/2502.13130

J. Yang, R. Tan, Q. Wu, R. Zheng, B. Peng, Y. Liang, Y. Gu, M. Cai, S. Ye, J. Jang, Y. Deng, L. Liden, 和 J. Gao, “Magma: A foundation model for multimodal ai agents(Magma: 多模态人工智能代理的基础模型),”2025. [在线]. 可用: https://arxiv.org/abs/2502.13130

[5] J. Wang, H. Xu, J. Ye, M. Yan, W. Shen, J. Zhang, F. Huang, and J. Sang, ”Mobile-Agent: Autonomous Multi-Modal Mobile Device Agent with Visual Perception,” Apr. 2024, arXiv:2401.16158. [Online]. Available: http://arxiv.org/abs/2401.16158

J. Wang, H. Xu, J. Ye, M. Yan, W. Shen, J. Zhang, F. Huang, 和 J. Sang, “Mobile-Agent: Autonomous Multi-Modal Mobile Device Agent with Visual Perception(Mobile-Agent: 具备视觉感知的自主多模态移动设备代理),”2024 年 4 月, arXiv:2401.16158. [在线]. 可用: http://arxiv.org/abs/2401.16158

[6] C. Zhang, Z. Yang, J. Liu, Y. Han, X. Chen, Z. Huang, B. Fu, and G. Yu, ”AppAgent: multimodal agents as smartphone users,” Dec. 2023, arXiv:2312.13771 TLDR: A novel LLM-based multimodal agent framework

designed to operate smartphone applications through a simplified action space, mimicking human-like interactions such as tapping and swiping, thereby broadening its applicability across diverse apps. [Online]. Available: http://arxiv.org/abs/2312.13771

C. Zhang, Z. Yang, J. Liu, Y. Han, X. Chen, Z. Huang, B. Fu, 和 G. Yu, “AppAgent: multimodal agents as smartphone users(AppAgent: 作为智能手机用户的多模态代理),” 2023 年 12 月, arXiv:2312.13771 摘要: 一种新颖的基于大型语言模型 (LLM) 的多模态代理框架，设计用于通过简化的动作空间操作智能手机应用，模拟人类的点击和滑动等交互，从而拓宽其在各种应用中的适用性。[在线]. 可用: http://arxiv.org/abs/2312.13771

[7] Y. Li, C. Zhang, W. Yang, B. Fu, P. Cheng, X. Chen, L. Chen, and Y. Wei, ”AppAgent v2: Advanced Agent for Flexible Mobile Interactions,” Aug. 2024. [Online]. Available: https://arxiv.org/abs/2408.11824v3

Y. Li, C. Zhang, W. Yang, B. Fu, P. Cheng, X. Chen, L. Chen, 和 Y. Wei, “AppAgent v2: Advanced Agent for Flexible Mobile Interactions(AppAgent v2: 灵活移动交互的高级代理),” 2024 年 8 月. [在线]. 可用: https://arxiv.org/abs/2408.11824v3

[8] K. Cheng, Q. Sun, Y. Chu, F. Xu, Y. Li, J. Zhang, and Z. Wu, ”SeeClick: Harnessing GUI Grounding for Advanced Visual GUI Agents,” Feb. 2024, arXiv:2401.10935 TLDR: This work proposes a novel visual GUI agent - SeeClick, which only relies on screenshots for task automation and creates ScreenSpot, the first realistic GUI grounding benchmark that encompasses mobile, desktop, and web environments. [Online]. Available: http://arxiv.org/abs/2401.10935

K. Cheng, Q. Sun, Y. Chu, F. Xu, Y. Li, J. Zhang, 和 Z. Wu, “SeeClick: Harnessing GUI Grounding for Advanced Visual GUI Agents(SeeClick: 利用 GUI 定位实现高级视觉 GUI 代理),” 2024 年 2 月, arXiv:2401.10935 摘要: 本文提出了一种新颖的视觉 GUI 代理——SeeClick，该代理仅依赖截图进行任务自动化，并创建了 ScreenSpot，这是首个涵盖移动端、桌面和网页环境的真实 GUI 定位基准。[在线]. 可用: http://arxiv.org/abs/2401.10935

[9] C. Zhang, L. Li, S. He, X. Zhang, B. Qiao, S. Qin, M. Ma, Y. Kang, Q. Lin, S. Rajmohan, D. Zhang, and Q. Zhang, ”UFO: A UI-Focused Agent for Windows OS Interaction,” May 2024, arXiv:2402.07939 TLDR: UFO, an innovative UI-Focused agent to fulfill user requests tailored to applications on Windows OS, harnessing the capabilities of GPT-Vision is introduced, standing as the first UI agent specifically tailored for task completion within the Windows OS environment. [Online]. Available: http://arxiv.org/abs/2402.07939

C. Zhang, L. Li, S. He, X. Zhang, B. Qiao, S. Qin, M. Ma, Y. Kang, Q. Lin, S. Rajmohan, D. Zhang, 和 Q. Zhang, “UFO: A UI-Focused Agent for Windows OS Interaction(UFO: 面向 Windows 操作系统交互的 UI 聚焦代理),” 2024 年 5 月, arXiv:2402.07939 摘要: 介绍了 UFO，一种创新的 UI 聚焦代理，用于满足 Windows 操作系统中针对应用的用户请求，利用 GPT-Vision 的能力，是首个专门针对 Windows OS 环境中任务完成的 UI 代理。[在线]. 可用: http://arxiv.org/abs/2402.07939

[10] K. Q. Lin, L. Li, D. Gao, Z. Yang, S. Wu, Z. Bai, W. Lei, L. Wang, and M. Z. Shou, ”Showui: One vision-language-action model for gui visual agent,” 2024. [Online]. Available: https://arxiv.org/abs/2411.17465

K. Q. Lin, L. Li, D. Gao, Z. Yang, S. Wu, Z. Bai, W. Lei, L. Wang, 和 M. Z. Shou, “Showui: One vision-language-action model for gui visual agent(Showui: 用于 GUI 视觉代理的视觉-语言-动作一体模型),” 2024. [在线]. 可用: https://arxiv.org/abs/2411.17465

[11] W. Jiang, Y. Zhuang, C. Song, X. Yang, and C. Zhang, ”Appagentx: Evolving gui agents as proficient smartphone users,” 2025. [Online]. Available: https: //arxiv.org/abs/2503.02268

W. Jiang, Y. Zhuang, C. Song, X. Yang, 和 C. Zhang, “Appagentx: Evolving gui agents as proficient smartphone users(Appagentx: 作为熟练智能手机用户进化的 GUI 代理),” 2025. [在线]. 可用: https://arxiv.org/abs/2503.02268

[12] Y. He, J. Jin, S. Xia, J. Su, R. Fan, H. Zou, X. Hu, and P. Liu, ”Pc agent: While you sleep, ai works - a cognitive journey into digital world,” 2024. [Online]. Available: https://arxiv.org/abs/2412.17589

何勇, 金杰, 夏松, 苏杰, 范锐, 邹浩, 胡翔, 刘鹏, ”Pc agent: 当你沉睡时，人工智能在工作——一次数字世界的认知之旅,” 2024. [在线]. 可获取: https://arxiv.org/abs/2412.17589

[13] C. Zhang, S. He, L. Li, S. Qin, Y. Kang, Q. Lin, and D. Zhang, ”Api agents vs. gui agents: Divergence and convergence,” 2025. [Online]. Available: https://arxiv.org/abs/2503.11069

张晨, 何松, 李磊, 秦松, 康宇, 林强, 张东, ”API 代理与 GUI 代理: 分歧与融合,” 2025. [在线]. 可获取: https://arxiv.org/abs/2503.11069

[14] C. Bai, X. Zang, Y. Xu, S. Sunkara, A. Rastogi, J. Chen, and B. A. y. Arcas, ”UIBert: Learning Generic Multimodal Representations for UI Understanding,” Aug. 2021, arXiv:2107.13731. [Online]. Available: http://arxiv.org/abs/2107.13731

白晨, 臧翔, 徐阳, S. Sunkara, A. Rastogi, 陈杰, B. A. y. Arcas, ”UIBert: 学习通用多模态表示以理解用户界面,” 2021 年 8 月, arXiv:2107.13731. [在线]. 可获取: http://arxiv.org/abs/2107.13731

[15] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, ”Llama: Open and efficient foundation language models,” 2023. [Online]. Available: https://arxiv.org/abs/2302.13971

H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, G. Lample, ”Llama: 开放且高效的基础语言模型,” 2023. [在线]. 可获取: https://arxiv.org/abs/2302.13971

[16] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang, B. Hui, L. Ji, M. Li, J. Lin, R. Lin, D. Liu, G. Liu, C. Lu, K. Lu, J. Ma, R. Men, X. Ren, X. Ren, C. Tan, S. Tan, J. Tu, P. Wang, S. Wang, W. Wang, S. Wu, B. Xu, J. Xu, A. Yang, H. Yang, J. Yang, S. Yang, Y. Yao, B. Yu, H. Yuan, Z. Yuan, J. Zhang, X. Zhang, Y. Zhang, Z. Zhang, C. Zhou, J. Zhou, X. Zhou, and T. Zhu, ”Qwen technical report,” 2023. [Online]. Available: https://arxiv.org/abs/2309.16609

白杰, 白松, 褚阳, 崔志, 党凯, 邓翔, 范宇, 葛伟, 韩阳, 黄飞, 惠斌, 纪磊, 李明, 林军, 林锐, 刘东, 刘刚, 陆超, 陆凯, 马军, 门锐, 任翔, 任翔, 谭超, 谭松, 涂军, 王鹏, 王松, 王伟, 吴松, 徐斌, 徐军, 杨安, 杨浩, 杨军, 杨松, 姚阳, 余斌, 袁浩, 袁志, 张军, 张翔, 张阳, 张志, 周超, 周军, 周翔, 朱涛, "Qwen 技术报告," 2023. [在线]. 可获取: https://arxiv.org/abs/2309.16609

[17] OpenAI, "Gpt-4 technical report," 2024. [Online]. Available: https://arxiv.org/abs/2303.08774

OpenAI, "GPT-4 技术报告," 2024. [在线]. 可获取: https://arxiv.org/abs/2303.08774

[18] T. GLM, "Chatglm: A family of large language models from glm-130b to glm-4 all tools," 2024. [Online]. Available: https://arxiv.org/abs/2406.12793

T. GLM, "ChatGLM: 从 GLM-130B 到 GLM-4 All Tools 的大型语言模型家族," 2024. [在线]. 可获取: https://arxiv.org/abs/2406.12793

[19] DeepSeek-AI, "Deepseek-v3 technical report," 2025. [Online]. Available: https://arxiv.org/abs/2412.19437

DeepSeek-AI, "DeepSeek-v3 技术报告," 2025. [在线]. 可获取: https://arxiv.org/abs/2412.19437

[20] P. Wang, S. Bai, S. Tan, S. Wang, Z. Fan, J. Bai, K. Chen, X. Liu, J. Wang, W. Ge, Y. Fan, K. Dang, M. Du, X. Ren, R. Men, D. Liu, C. Zhou, J. Zhou, and J. Lin, "Qwen2- vl: Enhancing vision-language model's perception of the world at any resolution," 2024. [Online]. Available: https://arxiv.org/abs/2409.12191

王鹏, 白松, 谭松, 王松, 范志, 白杰, 陈凯, 刘翔, 王军, 葛伟, 范宇, 党凯, 杜明, 任翔, 门锐, 刘东, 周超, 周军, 林军, "Qwen2-VL: 提升视觉语言模型对任意分辨率世界的感知," 2024. [在线]. 可获取: https://arxiv.org/abs/2409.12191

[21] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," 2023. [Online]. Available: https://arxiv.org/abs/2304.08485

刘浩, 李超, 吴强, 李永健, " 视觉指令调优," 2023. [在线]. 可获取: https://arxiv.org/abs/2304.08485

[22] S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang, H. Zhong, Y. Zhu, M. Yang, Z. Li, J. Wan, P. Wang, W. Ding, Z. Fu, Y. Xu, J. Ye, X. Zhang, T. Xie, Z. Cheng, H. Zhang, Z. Yang, H. Xu, and J. Lin, "Qwen2.5-vl technical report," 2025. [Online]. Available: https://arxiv.org/abs/2502.13923

S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang, H. Zhong, Y. Zhu, M. Yang, Z. Li, J. Wan, P. Wang, W. Ding, Z. Fu, Y. Xu, J. Ye, X. Zhang, T. Xie, Z. Cheng, H. Zhang, Z. Yang, H. Xu, and J. Lin, "Qwen2.5-vl 技术报告," 2025. [在线]. 可用地址: https://arxiv.org/abs/2502.13923

[23] H. Lu, W. Liu, B. Zhang, B. Wang, K. Dong, B. Liu, J. Sun, T. Ren, Z. Li, H. Yang, Y. Sun, C. Deng, H. Xu, Z. Xie, and C. Ruan, "Deepseek-vl: Towards real-world vision-language understanding," 2024. [Online]. Available: https://arxiv.org/abs/2403.05525

H. Lu, W. Liu, B. Zhang, B. Wang, K. Dong, B. Liu, J. Sun, T. Ren, Z. Li, H. Yang, Y. Sun, C. Deng, H. Xu, Z. Xie, and C. Ruan, ”Deepseek-vl: 迈向真实世界的视觉-语言理解,” 2024. [在线]. 可用地址: https://arxiv.org/abs/2403.05525

[24] Q. Ye, H. Xu, G. Xu, J. Ye, M. Yan, Y. Zhou, J. Wang, A. Hu, P. Shi, Y. Shi, C. Li, Y. Xu, H. Chen, J. Tian, Q. Qian, J. Zhang, F. Huang, and J. Zhou, ”mplug-owl: Modularization empowers large language models with multimodality,” 2024. [Online]. Available: https://arxiv.org/abs/2304.14178

Q. Ye, H. Xu, G. Xu, J. Ye, M. Yan, Y. Zhou, J. Wang, A. Hu, P. Shi, Y. Shi, C. Li, Y. Xu, H. Chen, J. Tian, Q. Qian, J. Zhang, F. Huang, and J. Zhou, ”mplug-owl: 模块化赋能大型语言模型的多模态能力,” 2024. [在线]. 可用地址: https://arxiv.org/abs/2304.14178

[25] Z. Chen, J. Wu, W. Wang, W. Su, G. Chen, S. Xing, M. Zhong, Q. Zhang, X. Zhu, L. Lu, B. Li, P. Luo, T. Lu, Y. Qiao, and J. Dai, ”Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks,” 2024. [Online]. Available: https://arxiv.org/abs/2312.14238

Z. Chen, J. Wu, W. Wang, W. Su, G. Chen, S. Xing, M. Zhong, Q. Zhang, X. Zhu, L. Lu, B. Li, P. Luo, T. Lu, Y. Qiao, and J. Dai, ”Internvl: 扩展视觉基础模型并对齐以支持通用视觉-语言任务,” 2024. [在线]. 可用地址: https://arxiv.org/abs/2312.14238

[26] F. Tang, Y. Shen, H. Zhang, S. Chen, G. Hou, W. Zhang, W. Zhang, K. Song, W. Lu, and Y. Zhuang, ”Think twice, click once: Enhancing gui grounding via fast and slow systems,” 2025. [Online]. Available: https://arxiv.org/abs/2503.06470

F. Tang, Y. Shen, H. Zhang, S. Chen, G. Hou, W. Zhang, W. Zhang, K. Song, W. Lu, and Y. Zhuang, ”三思而后点: 通过快慢系统增强图形用户界面定位,” 2025. [在线]. 可用地址: https://arxiv.org/abs/2503.06470

[27] Y. Shen, K. Song, X. Tan, D. Li, W. Lu, and Y. Zhuang, ”Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face,” 2023. [Online]. Available: https://arxiv.org/abs/2303.17580

Y. Shen, K. Song, X. Tan, D. Li, W. Lu, and Y. Zhuang, ”HuggingGPT: 利用 ChatGPT 及其 Hugging Face 伙伴解决 AI 任务,” 2023. [在线]. 可用地址: https://arxiv.org/abs/2303.17580

[28] DeepSeek-AI, D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi, X. Zhang, X. Yu, Y. Wu, Z. F. Wu, Z. Gou, Z. Shao, Z. Li, Z. Gao, A. Liu, B. Xue, B. Wang, B. Wu, B. Feng, C. Lu, C. Zhao, C. Deng, C. Zhang, C. Ruan, D. Dai, D. Chen, D. Ji, E. Li, F. Lin, F. Dai, F. Luo, G. Hao, G. Chen, G. Li, H. Zhang, H. Bao, H. Xu, H. Wang, H. Ding, H. Xin, H. Gao, H. Qu, H. Li, J. Guo, J. Li, J. Wang, J. Chen, J. Yuan, J. Qiu, J. Li, J. L. Cai, J. Ni, J. Liang, J. Chen, K. Dong, K. Hu, K. Gao, K. Guan, K. Huang, K. Yu, L. Wang, L. Zhang, L. Zhao, L. Wang, L. Zhang, L. Xu, L. Xia, M. Zhang, M. Zhang, M. Tang, M. Li, M. Wang, M. Li, N. Tian, P. Huang, P. Zhang, Q. Wang, Q. Chen, Q. Du, R. Ge, R. Zhang, R. Pan, R. Wang, R. J. Chen, R. L. Jin, R. Chen, S. Lu, S. Zhou, S. Chen, S. Ye, S. Wang, S. Yu, S. Zhou, S. Pan, S. S. Li, S. Zhou, S. Wu, S. Ye, T. Yun, T. Pei, T. Sun, T. Wang, W. Zeng, W. Zhao, W. Liu, W. Liang, W. Gao, W. Yu, W. Zhang, W. L. Xiao, W. An, X. Liu, X. Wang, X. Chen, X. Nie, X. Cheng, X. Liu, X. Xie, X. Liu, X. Yang, X. Li, X. Su, X. Lin, X. Q. Li, X. Jin, X. Shen, X. Chen, X. Sun, X. Wang, X. Song, X. Zhou, X. Wang, X. Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. Zhang, Y. Xu, Y. Li, Y. Zhao, Y. Sun, Y. Wang, Y. Yu, Y. Zhang, Y. Shi, Y. Xiong, Y. He, Y. Piao, Y. Wang, Y. Tan, Y. Ma, Y. Liu, Y. Guo, Y. Ou, Y. Wang, Y. Gong, Y. Zou, Y. He, Y. Xiong, Y. Luo, Y. You, Y. Liu, Y. Zhou, Y. X. Zhu, Y.

Xu, Y. Huang, Y. Li, Y. Zheng, Y. Zhu, Y. Ma, Y. Tang, Y. Zha, Y. Yan, Z. Z. Ren, Z. Ren, Z. Sha, Z. Fu, Z. Xu, Z. Xie, Z. Zhang, Z. Hao, Z. Ma, Z. Yan, Z. Wu, Z. Gu, Z. Zhu, Z. Liu, Z. Li, Z. Xie, Z. Song, Z. Pan, Z. Huang, Z. Xu, Z. Zhang, and Z. Zhang, ”Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning,” 2025. [Online]. Available: https://arxiv.org/abs/2501.12948

DeepSeek-AI, D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi, X. Zhang, X. Yu, Y. Wu, Z. F. Wu, Z. Gou, Z. Shao, Z. Li, Z. Gao, A. Liu, B. Xue, B. Wang, B. Wu, B. Feng, C. Lu, C. Zhao, C. Deng, C. Zhang, C. Ruan, D. Dai, D. Chen, D. Ji, E. Li, F. Lin, F. Dai, F. Luo, G. Hao, G. Chen, G. Li, H. Zhang, H. Bao, H. Xu, H. Wang, H. Ding, H. Xin, H. Gao, H. Qu, H. Li, J. Guo, J. Li, J. Wang, J. Chen, J. Yuan, J. Qiu, J. Li, J. L. Cai, J. Ni, J. Liang, J. Chen, K. Dong, K. Hu, K. Gao, K. Guan, K. Huang, K. Yu, L. Wang, L. Zhang, L. Zhao, L. Wang, L. Zhang, L. Xu, L. Xia, M. Zhang, M. Zhang, M. Tang, M. Li, M. Wang, M. Li, N. Tian, P. Huang, P. Zhang, Q. Wang, Q. Chen, Q. Du, R. Ge, R. Zhang, R. Pan, R. Wang, R. J. Chen, R. L. Jin, R. Chen, S. Lu, S. Zhou, S. Chen, S. Ye, S. Wang, S. Yu, S. Zhou, S. Pan, S. S. Li, S. Zhou, S. Wu, S. Ye, T. Yun, T. Pei, T. Sun, T. Wang, W. Zeng, W. Zhao, W. Liu, W. Liang, W. Gao, W. Yu, W. Zhang, W. L. Xiao, W. An, X. Liu, X. Wang, X. Chen, X. Nie, X. Cheng, X. Liu, X. Xie, X. Liu, X. Yang, X. Li, X. Su, X. Lin, X. Q. Li, X. Jin, X. Shen, X. Chen, X. Sun, X. Wang, X. Song, X. Zhou, X. Wang, X. Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. Zhang, Y. Xu, Y. Li, Y. Zhao, Y. Sun, Y. Wang, Y. Yu, Y. Zhang, Y. Shi, Y. Xiong, Y. He, Y. Piao, Y. Wang, Y. Tan, Y. Ma, Y. Liu, Y. Guo, Y. Ou, Y. Wang, Y. Gong, Y. Zou, Y. He, Y. Xiong, Y. Luo, Y. You, Y. Liu, Y. Zhou, Y. X. Zhu, Y. Xu, Y. Huang, Y. Li, Y. Zheng, Y. Zhu, Y. Ma, Y. Tang, Y. Zha, Y. Yan, Z. Z. Ren, Z. Ren, Z. Sha, Z. Fu, Z. Xu, Z. Xie, Z. Zhang, Z. Hao, Z. Ma, Z. Yan, Z. Wu, Z. Gu, Z. Zhu, Z. Liu, Z. Li, Z. Xie, Z. Song, Z. Pan, Z. Huang, Z. Xu, Z. Zhang, and Z. Zhang, “Deepseek-r1: 通过强化学习激励大型语言模型 (LLMs) 推理能力，” 2025 年。[在线]. 可获取:https://arxiv.org/abs/2501.12948

[29] Y. Xu, X. Liu, X. Sun, S. Cheng, H. Yu, H. Lai, S. Zhang, D. Zhang, J. Tang, and Y. Dong, ”AndroidLab: training and systematic benchmarking of android autonomous agents,” Oct. 2024, arXiv:2410.24024. [Online]. Available: http://arxiv.org/abs/2410.24024

Y. Xu, X. Liu, X. Sun, S. Cheng, H. Yu, H. Lai, S. Zhang, D. Zhang, J. Tang, 和 Y. Dong, “Android-Lab: 安卓自主代理的训练与系统性基准测试，” 2024 年 10 月，arXiv:2410.24024。[在线]. 可获取:http://arxiv.org/abs/2410.24024

[30] S. Nong, J. Zhu, R. Wu, J. Jin, S. Shan, X. Huang, and W. Xu, ”MobileFlow: a multimodal LLM for mobile GUI agent,” Aug. 2024, arXiv:2407.04346. [Online]. Available: http://arxiv.org/abs/2407.04346

S. Nong, J. Zhu, R. Wu, J. Jin, S. Shan, X. Huang, 和 W. Xu,“MobileFlow: 一种用于移动 GUI 代理的多模态大语言模型 (LLM)”, 2024 年 8 月, arXiv:2407.04346。[在线]. 可获取:http://arxiv.org/abs/2407.04346

[31] Y. Liu, P. Li, Z. Wei, C. Xie, X. Hu, X. Xu, S. Zhang, X. Han, H. Yang, and F. Wu, ”Infiguiagent: A multi-modal generalist gui agent with native reasoning and reflection,” 2025. [Online]. Available: https://arxiv.org/abs/2501.04575

Y. Liu, P. Li, Z. Wei, C. Xie, X. Hu, X. Xu, S. Zhang, X. Han, H. Yang, 和 F. Wu,“Infiguiagent: 具备本地推理和反思能力的多模态通用 GUI 代理”，2025 年。[在线]. 可获取:https://arxiv.org/abs/2501.04575

[32] H. Wen, Y. Li, G. Liu, S. Zhao, T. Yu, T. J.-J. Li, [44] S. Jiang, Y. Liu, Y. Zhang, and Y. Liu, ”Autodroid: Llm-powered task automation in android,” Mar. 2024. [Online]. Available: http://arxiv.org/abs/2308.15272

H. Wen, Y. Li, G. Liu, S. Zhao, T. Yu, T. J.-J. Li, [44] S. Jiang, Y. Liu, Y. Zhang, 和 Y. Liu, “Autodroid: 基于大语言模型的安卓任务自动化”，2024 年 3 月。[在线]. 可获取:http://arxiv.org/abs/2308.15272

[33] Z. Zhu, H. Tang, Y. Li, K. Lan, Y. Jiang, H. Zhou, Y. Wang, S. Zhang, L. Sun, L. Chen, and K. Yu, ”Moba: A two-level agent system for efficient mobile task automation,” Oct. 2024. [Online]. Available: http://arxiv.org/abs/2410.13757

Z. Zhu, H. Tang, Y. Li, K. Lan, Y. Jiang, H. Zhou, Y. Wang, S. Zhang, L. Sun, L. Chen, 和 K. Yu, “Moba: 一种高效移动任务自动化的双层代理系统”，2024 年 10 月。[在线]. 可获取:http://arxiv.org/abs/2410.13757

[34] S. Agashe, J. Han, S. Gan, J. Yang, A. Li, and X. E. Wang, ”Agent S: an open agentic framework that uses computers like a human,” Oct. 2024, arXiv:2410.08164. [Online]. Available: http://arxiv.org/abs/2410.08164

S. Agashe, J. Han, S. Gan, J. Yang, A. Li, 和 X. E. Wang, “Agent S: 一种像人类一样使用计算机的开放代理框架”，2024 年 10 月，arXiv:2410.08164。[在线]. 可获取:http://arxiv.org/abs/2410.08164

[35] Q. Lu, W. Shao, Z. Liu, F. Meng, B. Li, B. Chen, S. Huang, K. Zhang, Y. Qiao, and P. Luo, ”Gui odyssey: A comprehensive dataset for cross-app gui navigation on mobile devices,” 2024. [Online]. Available: https://arxiv.org/abs/2406.08451

Q. Lu, W. Shao, Z. Liu, F. Meng, B. Li, B. Chen, S. Huang, K. Zhang, Y. Qiao, 和 P. Luo, “Gui odyssey: 面向移动设备跨应用 GUI 导航的综合数据集”, 2024 年。[在线]. 可获取:https://arxiv.org/abs/2406.08451

[36] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, ”Chain-of-thought prompting elicits reasoning in large language models,” Jan. 2023. [Online]. Available: http://arxiv.org/abs/2201.11903

J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, 和 D. Zhou, “链式思维提示激发大型语言模型中的推理能力,”2023 年 1 月。[在线]. 可获取: http://arxiv.org/abs/2201.11903

[37] A. Zeng, M. Attarian, B. Ichter, K. Choromanski, A. Wong, S. Welker, F. Tombari, A. Purohit, M. Ryoo, V. Sindhwani, J. Lee, V. Vanhoucke, and P. Florence, ”Socratic models: Composing zero-shot multimodal reasoning with language,” May 2022. [Online]. Available: http://arxiv.org/abs/2204.00598

A. Zeng, M. Attarian, B. Ichter, K. Choromanski, A. Wong, S. Welker, F. Tombari, A. Purohit, M. Ryoo, V. Sindhwani, J. Lee, V. Vanhoucke, 和 P. Florence, “苏格拉底模型: 通过语言组合实现零样本多模态推理,”2022 年 5 月。[在线]. 可获取: http://arxiv.org/abs/2204.00598

[38] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, ”React: Synergizing reasoning and acting in language models,” Mar. 2023. [Online]. Available: http://arxiv.org/abs/2210.03629

S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, 和 Y. Cao, “React: 在语言模型中协同推理与行动,”2023 年 3 月。[在线]. 可获取: http://arxiv.org/abs/2210.03629

[39] Z. Yang, L. Li, J. Wang, K. Lin, E. Azarnasab, F. Ahmed, Z. Liu, C. Liu, M. Zeng, and L. Wang, ”Mm-react: Prompting chatgpt for multimodal reasoning and action,” Mar. 2023. [Online]. Available: http:

//arxiv.org/abs/2303.11381

Z. Yang, L. Li, J. Wang, K. Lin, E. Azarnasab, F. Ahmed, Z. Liu, C. Liu, M. Zeng, 和 L. Wang, “Mm-react: 为多模态推理与行动提示 ChatGPT,” 2023 年 3 月。[在线]. 可获取: http://arxiv.org/abs/2303.11381

[40] Z. Wu, C. Han, Z. Ding, Z. Weng, Z. Liu, S. Yao, T. Yu, and L. Kong, ”OS-copilot: towards generalist computer agents with self-improvement,” Feb. 2024, arXiv:2402.07456 TLDR: OS-Copilot is introduced, a framework to build generalist agents capable of interfacing with comprehensive elements in an operating system (OS), including the web, code terminals, files, multimedia, and various third-party applications, and FRIDAY, a self-improving embodied agent for automating general computer tasks. [Online]. Available: http://arxiv.org/abs/2402.07456

Z. Wu, C. Han, Z. Ding, Z. Weng, Z. Liu, S. Yao, T. Yu, 和 L. Kong, “OS-copilot: 迈向具备自我提升能力的通用计算机代理,” 2024 年 2 月，arXiv:2402.07456 摘要: 介绍了 OS-Copilot 框架，用于构建能够与操作系统 (OS) 中包括网页、代码终端、文件、多媒体及各种第三方应用等全面元素交互的通用代理，以及 FRIDAY，一种用于自动化通用计算机任务的自我提升具身代理。[在线]. 可获取: http://arxiv.org/abs/2402.07456

[41] L. Wang, Y. Deng, Y. Zha, G. Mao, Q. Wang, T. Min, W. Chen, and S. Chen, ”Mobileagentbench: An efficient and user-friendly benchmark for mobile llm agents,” arXiv preprint arXiv:2406.08184, 2024.

L. Wang, Y. Deng, Y. Zha, G. Mao, Q. Wang, T. Min, W. Chen, 和 S. Chen, “MobileAgentBench: 一个高效且用户友好的移动大型语言模型代理基准测试,” arXiv 预印本 arXiv:2406.08184, 2024 年。

[42] X. Deng, Y. Gu, B. Zheng, S. Chen, S. Stevens, B. Wang, H. Sun, and Y. Su, ”Mind2web: Towards a generalist agent for the web,” Advances in Neural Information Processing Systems, vol. 36, 2024.

X. Deng, Y. Gu, B. Zheng, S. Chen, S. Stevens, B. Wang, H. Sun, 和 Y. Su, “Mind2web: 迈向通用网络代理,” 神经信息处理系统进展, 第 36 卷, 2024 年。

[43] S. Zhou, F. F. Xu, H. Zhu, X. Zhou, R. Lo, A. Sridhar, X. Cheng, T. Ou, Y. Bisk, D. Fried et al., ”Webarena: A realistic web environment for building autonomous agents,” arXiv preprint arXiv:2307.13854, 2023.

S. Zhou, F. F. Xu, H. Zhu, X. Zhou, R. Lo, A. Sridhar, X. Cheng, T. Ou, Y. Bisk, D. Fried 等, “Webarena: 构建自主代理的真实网络环境,” arXiv 预印本 arXiv:2307.13854, 2023 年。

[44] C. Rawles, S. Clinckemaillie, Y. Chang, J. Waltz, G. Lau, M. Fair, A. Li, W. Bishop, W. Li, F. Campbell-Ajala, D. Toyama, R. Berry, D. Tyamagundlu, T. Lillicrap, and O. Riva, ”AndroidWorld: A Dynamic Benchmarking Environment for Autonomous Agents,” Oct. 2024, arXiv:2405.14573. [Online]. Available: http://arxiv.org/abs/2405.14573

C. Rawles, S. Clinckemaillie, Y. Chang, J. Waltz, G. Lau, M. Fair, A. Li, W. Bishop, W. Li, F. Campbell-Ajala, D. Toyama, R. Berry, D. Tyamagundlu, T. Lillicrap, 和 O. Riva, “AndroidWorld: 自主代理的动态基准测试环境,” 2024 年 10 月，arXiv:2405.14573。[在线]. 可获取: http://arxiv.org/abs/2405.14573

[45] C. Rawles, S. Clinckemaillie, Y. Chang, J. Waltz, G. Lau, M. Fair, A. Li, W. Bishop, W. Li, F. Campbell-Ajala et al., ”Androidworld: A dynamic benchmarking environment for autonomous agents,” arXiv preprint arXiv:2405.14573, 2024.

C. Rawles, S. Clinckemaillie, Y. Chang, J. Waltz, G. Lau, M. Fair, A. Li, W. Bishop, W. Li, F. Campbell-Ajala 等, “AndroidWorld: 自主代理的动态基准测试环境,”arXiv 预印本 arXiv:2405.14573, 2024 年。

[46] K. Li, Z. Meng, H. Lin, Z. Luo, Y. Tian, J. Ma, Z. Huang, and T.-S. Chua, ”Screenspot-pro: Gui grounding for professional high-resolution computer use,” 2025.

K. Li, Z. Meng, H. Lin, Z. Luo, Y. Tian, J. Ma, Z. Huang, 和 T.-S. Chua, “Screenspot-pro: 面向专业高分辨率计算机使用的 GUI 定位,”2025 年。

[47] J. Y. Koh, R. Lo, L. Jang, V. Duvvur, M. C. Lim, P.- Y. Huang, G. Neubig, S. Zhou, R. Salakhutdinov, and D. Fried, ”Visualwebarena: Evaluating multimodal agents on realistic visual web tasks,” arXiv preprint arXiv:2401.13649, 2024.

J. Y. Koh, R. Lo, L. Jang, V. Duvvur, M. C. Lim, P.- Y. Huang, G. Neubig, S. Zhou, R. Salakhutdinov, 和 D. Fried,“Visualwebarena: 在真实视觉网页任务上评估多模态代理,”arXiv 预印本 arXiv:2401.13649, 2024.

[48] H. Wen, Y. Li, G. Liu, S. Zhao, T. Yu, T. J.-J. Li, S. Jiang, Y. Liu, Y. Zhang, and Y. Liu, ”AutoDroid: LLM-powered task automation in android,” Mar. 2024, arXiv:2308.15272 TLDR: This work introduces AutoDroid, a mobile task automation system that can handle arbitrary tasks on any Android application without manual efforts and aims to combine the commonsense knowledge of LLMs and domain-specific knowledge of apps through automated dynamic analysis. [Online]. Available: http://arxiv.org/abs/2308.15272

H. Wen, Y. Li, G. Liu, S. Zhao, T. Yu, T. J.-J. Li, S. Jiang, Y. Liu, Y. Zhang, 和 Y. Liu, “AutoDroid: 基于大型语言模型 (LLM) 的安卓任务自动化,”2024 年 3 月, arXiv:2308.15272 摘要: 本工作介绍了 Auto-Droid，一种移动任务自动化系统，能够无需人工干预处理任何安卓应用上的任意任务，旨在通过自动动态分析结合大型语言模型的常识知识与应用的领域特定知识。[在线]. 可获取:http://arxiv.org/abs/2308.15272

[49] C. Rawles, A. Li, D. Rodriguez, O. Riva, and T. Lilli-crap, ”Androidinthewild: A large-scale dataset for android device control,” Advances in Neural Information Processing Systems, vol. 36, 2024.

C. Rawles, A. Li, D. Rodriguez, O. Riva, 和 T. Lilli-crap, “Androidinthewild: 用于安卓设备控制的大规模数据集,”神经信息处理系统进展 (NeurIPS)，第 36 卷，2024 年。

[50] T. Ding, ”MobileAgent: enhancing mobile control via human-machine interaction and SOP integration,” Jan. 2024, arXiv:2401.04124. [Online]. Available: http://arxiv.org/abs/2401.04124

T. Ding, “MobileAgent: 通过人机交互和标准操作程序 (SOP) 集成增强移动控制,”2024 年 1 月, arXiv:2401.04124。[在线]. 可获取:http://arxiv.org/abs/2401.04124

[51] D. Gao, L. Ji, Z. Bai, M. Ouyang, P. Li, D. Mao, Q. Wu, W. Zhang, P. Wang, X. Guo et al., ”Assist-gui: Task-oriented desktop graphical user interface automation,” arXiv preprint arXiv:2312.13108, 2023.

D. Gao, L. Ji, Z. Bai, M. Ouyang, P. Li, D. Mao, Q. Wu, W. Zhang, P. Wang, X. Guo 等, “Assist-gui: 面向任务的桌面图形用户界面自动化,”arXiv 预印本 arXiv:2312.13108, 2023 年。

[52] J. Yang, H. Zhang, F. Li, X. Zou, C. Li, and J. Gao, ”Set-of-Mark Prompting Unleashes Extraordinary Visual Grounding in GPT-4V,” Nov. 2023, arXiv:2310.11441 TLDR: The experiments show that GPT-4V with SoM in zero-shot setting outperforms the state-of-the-art fully-finetuned referring expression comprehension and segmentation model on RefCOCOg, and the effectiveness of SoM on a wide range of fine-grained vision and multimodal tasks is validated. [Online]. Available: http://arxiv.org/abs/2310.11441

J. Yang, H. Zhang, F. Li, X. Zou, C. Li, 和 J. Gao, “Set-of-Mark 提示释放 GPT-4V 非凡的视觉定位能力,”2023 年 11 月, arXiv:2310.11441 摘要: 实验表明，在零样本设置下，采用 SoM 的 GPT-4V 在 RefCOCOg 数据集上超越了最先进的全微调指代表达理解与分割模型，并验证了 SoM 在广泛细粒度视觉和多模态任务中的有效性。[在线]. 可获取:http://arxiv.org/abs/2310.11441

[53] W. Tan, W. Zhang, X. Xu, H. Xia, Z. Ding, B. Li, B. Zhou, J. Yue, J. Jiang, Y. Li, R. An, M. Qin, C. Zong, L. Zheng, Y. Wu, X. Chai, Y. Bi, T. Xie, P. Gu, X. Li, C. Zhang, L. Tian, C. Wang, X. Wang, B. F. Karlsson, B. An, S. Yan, and Z. Lu, ”Cradle: empowering foundation agents towards general computer control,” Jul. 2024, arXiv:2403.03186 TLDR: This work proposes the General Computer Control (GCC) setting: building foundation agents that can master any computer task by taking only screen images of the computer as input, and producing keyboard and mouse operations as output, similar to human-computer interaction. [Online]. Available: http://arxiv.org/abs/2403.03186

W. Tan, W. Zhang, X. Xu, H. Xia, Z. Ding, B. Li, B. Zhou, J. Yue, J. Jiang, Y. Li, R. An, M. Qin, C. Zong, L. Zheng, Y. Wu, X. Chai, Y. Bi, T. Xie, P. Gu, X. Li, C. Zhang, L. Tian, C. Wang, X. Wang, B. F. Karlsson, B. An, S. Yan, 和 Z. Lu, “Cradle: 赋能基础代理实现通用计算机控制,”2024 年 7 月, arXiv:2403.03186 摘要: 本工作提出了通用计算机控制 (GCC) 设置: 构建基础代理，能够仅通过计算机屏幕图像作为输入，输出键盘和鼠标操作，类似人机交互，掌握任何计算机任务。[在线]. 可获取:http://arxiv.org/abs/2403.03186

[54] W. Hong, W. Wang, Q. Lv, J. Xu, W. Yu, J. Ji, Y. Wang, [65] Z. Wang, Y. Zhang, J. Li, B. Xu, Y. Dong, M. Ding, and J. Tang, ”CogAgent: A Visual Language Model for GUI Agents,” Dec. 2023, arXiv:2312.08914 [cs]. [Online]. Available: http://arxiv.org/abs/2312.08914

W. Hong, W. Wang, Q. Lv, J. Xu, W. Yu, J. Ji, Y. Wang, [65] Z. Wang, Y. Zhang, J. Li, B. Xu, Y. Dong, M. Ding, 和 J. Tang, “CogAgent: 面向图形用户界面代理的视觉语言模型,”2023 年 12 月, arXiv:2312.08914 [计算机科学]. [在线]. 可获取:http://arxiv.org/abs/2312.08914

[55] Y. Qin, Y. Ye, J. Fang, H. Wang, S. Liang, S. Tian, J. Zhang, J. Li, Y. Li, S. Huang, W. Zhong, K. Li, J. Yang, Y. Miao, W. Lin, L. Liu, X. Jiang, Q. Ma, J. Li, X. Xiao, K. Cai, C. Li, Y. Zheng, C. Jin, C. Li, X. Zhou, M. Wang, H. Chen, Z. Li, H. Yang, H. Liu, F. Lin, T. Peng, X. Liu, and G. Shi, ”Ui-tars: Pioneering automated gui interaction with native agents,” 2025. [Online]. Available: https://arxiv.org/abs/2501.12326

秦勇 (Y. Qin)、叶勇 (Y. Ye)、方军 (J. Fang)、王浩 (H. Wang)、梁爽 (S. Liang)、田帅 (S. Tian)、张杰 (J. Zhang)、李军 (J. Li)、李勇 (Y. Li)、黄松 (S. Huang)、钟伟 (W. Zhong)、李凯 (K. Li)、杨杰 (J. Yang)、缪勇 (Y. Miao)、林伟 (W. Lin)、刘磊 (L. Liu)、姜晓 (X. Jiang)、马强 (Q. Ma)、李军 (J. Li)、肖翔 (X. Xiao)、蔡凯 (K. Cai)、李超 (C. Li)、郑勇 (Y. Zheng)、金超 (C. Jin)、李超 (C. Li)、周翔 (X. Zhou)、王明 (M. Wang)、陈浩 (H. Chen)、李志 (Z. Li)、杨浩 (H. Yang)、刘辉 (H. Liu)、林峰 (F. Lin)、彭涛 (T. Peng)、刘翔 (X. Liu)、史刚 (G. Shi)，“Ui-tars: 开创性地使用本地代理实现自动化图形用户界面交互”，2025 年。[在线]. 可获取:https://arxiv.org/abs/2501.12326

[56] Y. Lu, J. Yang, Y. Shen, and A. Awadallah, ”OmniParser for Pure Vision Based GUI Agent,” Aug. 2024, arXiv:2408.00203. [Online]. Available: http://arxiv.org/abs/2408.00203

陆洋 (Y. Lu)、杨杰 (J. Yang)、沈阳 (Y. Shen)、阿瓦达拉 (A. Awadallah)，“OmniParser: 基于纯视觉的图形用户界面代理”，2024 年 8 月，arXiv:2408.00203。[在线]. 可获取:http://arxiv.org/abs/2408.00203

[57] Z. Wang, H. Xu, J. Wang, X. Zhang, M. Yan, J. Zhang, F. Huang, and H. Ji, ”Mobile-agent-e: Self-evolving mobile assistant for complex tasks,” 2025. [Online]. Available: https://arxiv.org/abs/2501.11733

王志 (Z. Wang)、徐浩 (H. Xu)、王军 (J. Wang)、张晓 (X. Zhang)、闫明 (M. Yan)、张杰 (J. Zhang)、黄峰 (F. Huang)、季浩 (H. Ji)，“Mobile-agent-e: 面向复杂任务的自我进化移动助手”，2025 年。[在线]. 可获取:https://arxiv.org/abs/2501.11733

[58] J. Wang, H. Xu, H. Jia, X. Zhang, M. Yan, W. Shen, J. Zhang, F. Huang, and J. Sang, ”Mobile-Agent-v2: Mobile Device Operation Assistant with Effective Navigation via Multi-Agent Collaboration,” Jun. 2024, arXiv:2406.01014 TLDR: The proposed Mobile-Agent-v2, a multi-agent architecture for mobile device operation assistance, comprises three agents: planning agent, decision agent, and reflection agent that generates task progress, making the navigation of history operations more efficient. [Online]. Available: http://arxiv.org/abs/2406.01014

王军 (J. Wang)、徐浩 (H. Xu)、贾浩 (H. Jia)、张晓 (X. Zhang)、闫明 (M. Yan)、沈伟 (W. Shen)、张杰 (J. Zhang)、黄峰 (F. Huang)、桑杰 (J. Sang)，“Mobile-Agent-v2: 通过多代理协作实现高效导航的移动设备操作助手”，2024 年 6 月，arXiv:2406.01014 摘要: 所提出的 Mobile-Agent-v2 是一种用于移动设备操作辅助的多代理架构，包含规划代理、决策代理和反思代理，后者生成任务进展，使历史操作的导航更高效。[在线]. 可获取:http://arxiv.org/abs/2406.01014

[59] W. Zhou, Y. E. Jiang, L. Li, J. Wu, T. Wang, S. Qiu, J. Zhang, J. Chen, R. Wu, S. Wang, S. Zhu, J. Chen, W. Zhang, X. Tang, N. Zhang, H. Chen, P. Cui, and M. Sachan, ”Agents: An open-source framework for autonomous language agents,” 2023. [Online]. Available: https://arxiv.org/abs/2309.07870

周伟 (W. Zhou)、江叶恩 (Y. E. Jiang)、李磊 (L. Li)、吴军 (J. Wu)、王涛 (T. Wang)、邱爽 (S. Qiu)、张杰 (J. Zhang)、陈军 (J. Chen)、吴锐 (R. Wu)、王松 (S. Wang)、朱松 (S. Zhu)、陈军 (J. Chen)、张伟 (W. Zhang)、唐晓 (X. Tang)、张宁 (N. Zhang)、陈浩 (H. Chen)、崔鹏 (P. Cui)、萨昌 (M. Sachan)，“Agents: 一个开源的自主语言代理框架”，2023 年。[在线]. 可获取:https://arxiv.org/abs/2309.07870

[60] N. Shinn, F. Cassano, E. Berman, A. Gopinath, K. Narasimhan, and S. Yao, ”Reflexion: Language agents with verbal reinforcement learning,” Oct. 2023. [Online]. Available: http://arxiv.org/abs/2303.11366

辛恩 (N. Shinn)、卡萨诺 (F. Cassano)、伯曼 (E. Berman)、戈皮纳斯 (A. Gopinath)、纳拉西姆汉 (K. Narasimhan)、姚松 (S. Yao)，“Reflexion: 具备语言强化学习的语言代理”，2023 年 10 月。[在线]. 可获取:http://arxiv.org/abs/2303.11366

[61] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar, ”Voyager: An open-ended embodied agent with large language models,” Oct. 2023. [Online]. Available: http://arxiv.org/abs/2305.16291

王刚 (G. Wang)、谢勇 (Y. Xie)、江勇 (Y. Jiang)、曼德尔卡 (A. Mandlekar)、肖超 (C. Xiao)、朱勇 (Y. Zhu)、范磊 (L. Fan)、阿南德库马尔 (A. Anandkumar)，“Voyager: 基于大型语言模型的开放式具身代理”，2023 年 10 月。[在线]. 可获取:http://arxiv.org/abs/2305.16291

[62] Y. Zhu, S. Qiao, Y. Ou, S. Deng, N. Zhang, S. Lyu, Y. Shen, L. Liang, J. Gu, and H. Chen, ”Knowagent: Knowledge-augmented planning for llm-based agents,” 2024. [Online]. Available: https: //arxiv.org/abs/2403.03101

朱勇 (Y. Zhu)、乔爽 (S. Qiao)、欧阳 (Y. Ou)、邓松 (S. Deng)、张宁 (N. Zhang)、吕松 (S. Lyu)、沈阳 (Y. Shen)、梁磊 (L. Liang)、顾军 (J. Gu)、陈浩 (H. Chen)，“Knowagent: 基于知识增强的 LLM(大型语言模型) 代理规划”，2024 年。[在线]. 可获取:https://arxiv.org/abs/2403.03101

[63] S. Qiao, R. Fang, N. Zhang, Y. Zhu, X. Chen, S. Deng, Y. Jiang, P. Xie, F. Huang, and H. Chen, ”Agent planning with world knowledge model,” 2024. [Online]. Available: https://arxiv.org/abs/2405.14205

乔爽 (S. Qiao)、方锐 (R. Fang)、张宁 (N. Zhang)、朱勇 (Y. Zhu)、陈晓 (X. Chen)、邓松 (S. Deng)、江勇 (Y. Jiang)、谢鹏 (P. Xie)、黄峰 (F. Huang)、陈浩 (H. Chen)，“基于世界知识模型的代理规划”，2024 年。[在线]. 可获取:https://arxiv.org/abs/2405.14205

[64] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou, ”Self-consistency improves chain of thought reasoning in language models,” Mar. 2023. [Online]. Available: http://arxiv.org/abs/2203.11171

王翔 (X. Wang)、魏军 (J. Wei)、舒尔曼斯 (D. Schuurmans)、乐奇 (Q. Le)、池恩 (E. Chi)、纳朗 (S. Narang)、乔德里 (A. Chowdhery)、周丹 (D. Zhou)，“自洽性提升语言模型中的链式思维推理”，2023 年 3 月。[在线]. 可获取:http://arxiv.org/abs/2203.11171

[65] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, ”Tree of thoughts: Deliberate problem solving with large language models,” 2023. [Online]. Available: https://arxiv.org/abs/2305.10601

S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, 和 K. Narasimhan，“思维树 (Tree of thoughts): 利用大型语言模型进行深思熟虑的问题解决，”2023 年。[在线]. 可获取:https://arxiv.org/abs/2305.10601

[66] D. Zhang, S. Zhoubian, Z. Hu, Y. Yue, Y. Dong, and J. Tang, ”Rest-mcts*: Llm self-training via process reward guided tree search,” Sep. 2024. [Online]. Available: http://arxiv.org/abs/2406.03816

D. Zhang, S. Zhoubian, Z. Hu, Y. Yue, Y. Dong, 和 J. Tang, “Rest-mcts*: 通过过程奖励引导的树搜索进行大型语言模型自我训练，”2024 年 9 月。[在线]. 可获取:http://arxiv.org/abs/2406.03816

[67] M. Besta, N. Blach, A. Kubicek, R. Gerstenberger, M. Podstawski, L. Gianinazzi, J. Gajda, T. Lehmann, H. Niewiadomski, P. Nyczyk et al., ”Graph of thoughts: Solving elaborate problems with large language models,”

in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, no. 16, 2024, pp. 17682-17690.

M. Besta, N. Blach, A. Kubicek, R. Gerstenberger, M. Podstawski, L. Gianinazzi, J. Gajda, T. Lehmann, H. Niewiadomski, P. Nyczyk 等,“思维图 (Graph of thoughts): 利用大型语言模型解决复杂问题,”发表于第 38 卷第 16 期《美国人工智能协会会议论文集》(AAAI Conference on Artificial Intelligence),2024 年,页码 17682-17690。

[68] Z. Zhang and A. Zhang, ”You Only Look at Screens: Multimodal Chain-of-Action Agents,” Jun. 2024, arXiv:2309.11436. [Online]. Available: http: //arxiv.org/abs/2309.11436

Z. Zhang 和 A. Zhang, “你只需看屏幕: 多模态行动链代理 (Multimodal Chain-of-Action Agents),” 2024 年 6 月,arXiv:2309.11436。[在线]. 可获取:http://arxiv.org/abs/2309.11436

[69] J. Zhang, J. Wu, Y. Teng, M. Liao, N. Xu, X. Xiao, Z. Wei, and D. Tang, ”Android in the Zoo: Chain-of-Action-Thought for GUI Agents,” Jul. 2024, arXiv:2403.02713 TLDR: This work presents Chain-of-Action-Thought (dubbed CoAT), which takes the description of the previous actions, the current screen, and more importantly the action thinking of what actions should be performed and the outcomes led by the chosen action. [Online]. Available: http://arxiv.org/abs/2403.02713

J. Zhang, J. Wu, Y. Teng, M. Liao, N. Xu, X. Xiao, Z. Wei, 和 D. Tang, “动物园中的安卓: 图形用户界面代理的行动思维链 (Chain-of-Action-Thought),”2024 年 7 月,arXiv:2403.02713 摘要: 本工作提出了行动思维链 (简称 CoAT),该方法结合了先前动作的描述、当前屏幕信息,以及对应执行动作及其结果的思考。[在线]. 可获取:http://arxiv.org/abs/2403.02713

[70] Y. Dong, X. Zhu, Z. Pan, L. Zhu, and [8 Y. Yang, ”Villageragent: A graph-based multi-agent framework for coordinating complex task dependencies in minecraft,” Jun. 2024. [Online]. Available: http://arxiv.org/abs/2406.05720

Y. Dong, X. Zhu, Z. Pan, L. Zhu, 和 Y. Yang, “村民代理 (Villageragent): 基于图的多代理框架,用于协调《我的世界》(Minecraft) 中复杂任务依赖关系, ”2024 年 6 月。[在线]. 可获取:http://arxiv.org/abs/2406.05720

[71] R. Niu, J. Li, S. Wang, Y. Fu, X. Hu, X. Leng, H. Kong, Y. Chang, and Q. Wang, ”ScreenAgent: A Vision Language Model-driven Computer Control Agent,” arXiv e-prints, p. arXiv:2402.07945, Feb. 2024.

R. Niu, J. Li, S. Wang, Y. Fu, X. Hu, X. Leng, H. Kong, Y. Chang, 和 Q. Wang, “ScreenAgent: 一种由视觉语言模型驱动的计算机控制代理,”arXiv 电子预印本,编号 arXiv:2402.07945,2024 年 2 月。

[72] X. Ma, Z. Zhang, and H. Zhao, ”CoCo-agent: a comprehensive cognitive MLLM agent for smartphone GUI automation,” in Findings of the Association for Computational Linguistics: ACL 2024, L.-W. Ku, A. Martins, and V. Srikumar, Eds. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 9097-9110, tLDR: A Comprehensive Cognitive LLM Agent, CoCo-Agent, with two novel approaches, comprehensive environment perception (CEP) and conditional action prediction (CAP), to systematically improve the GUI automation performance. [Online]. Available: https://aclanthology.org/2024.findings-acl.539

X. Ma, Z. Zhang, 和 H. Zhao, “CoCo-agent: 一种面向智能手机图形用户界面自动化的综合认知多模态大型语言模型代理,”发表于 2024 年计算语言学协会 (ACL) 研究成果集，L.-W. Ku, A. Martins, 和 V. Srikumar 编，泰国曼谷: 计算语言学协会，2024 年 8 月，页码 9097-9110。摘要: 提出了一种综合认知大型语言模型代理 CoCo-Agent，采用两种新方法——综合环境感知 (CEP) 和条件动作预测 (CAP)，系统性提升图形用户界面自动化性能。[在线]. 可获取:https://aclanthology.org/2024.findings-acl.539

[73] H. He, W. Yao, K. Ma, W. Yu, Y. Dai, H. Zhang, Z. Lan, and D. Yu, ”WebVoyager: Building an End-to-End Web Agent with Large Multimodal Models,” arXiv e-prints, p. arXiv:2401.13919, Jan. 2024.

H. He, W. Yao, K. Ma, W. Yu, Y. Dai, H. Zhang, Z. Lan, 和 D. Yu, “WebVoyager: 基于大型多模态模型构建端到端网页代理,”arXiv 电子预印本，编号 arXiv:2401.13919，2024 年 1 月。

[74] K. Ma, H. Zhang, H. Wang, X. Pan, W. Yu, and D. Yu, ”LASER: LLM Agent with State-Space Exploration for Web Navigation,” arXiv e-prints, p. arXiv:2309.08172, Sep. 2023.

K. Ma, H. Zhang, H. Wang, X. Pan, W. Yu, 和 D. Yu, “LASER: 具备状态空间探索能力的网页导航大型语言模型代理,”arXiv 电子预印本，编号 arXiv:2309.08172，2023 年 9 月。

[75] Z. Wu, Z. Wu, F. Xu, Y. Wang, Q. Sun, C. Jia, K. Cheng, Z. Ding, L. Chen, P. P. Liang, and Y. Qiao, ”OS-ATLAS: a foundation action model for generalist GUI agents,” Oct. 2024, arXiv:2410.23218 version: 1. [Online]. Available: http://arxiv.org/abs/2410.23218

吴志强, 吴志强, 徐飞, 王勇, 孙强, 贾超, 程凯, 丁志, 陈磊, 梁鹏鹏, 乔勇, ”OS-ATLAS: 通用图形用户界面代理的基础动作模型,” 2024 年 10 月, arXiv:2410.23218 版本: 1. [在线]. 可获取: http://arxiv.org/abs/2410.23218

[76] H. Li, J. Su, Y. Chen, Q. Li, and Z. Zhang, ”Sheetcopilot: Bringing software productivity to the next level through large language models,” Oct. 2023. [Online]. Available: http://arxiv.org/abs/2305.19308

李浩, 苏杰, 陈颖, 李强, 张志, ”Sheetcopilot: 通过大型语言模型提升软件生产力至新高度,” 2023 年 10 月. [在线]. 可获取: http://arxiv.org/abs/2305.19308

[77] B. Zheng, B. Gou, J. Kil, H. Sun, and Y. Su, ”GPT- 4V(ision) is a Generalist Web Agent, if Grounded,” Mar. 2024, arXiv:2401.01614. [Online]. Available: http://arxiv.org/abs/2401.01614

郑博, 苟博, Kil J., 孙浩, 苏勇, ”GPT-4V(ision) 是通用网页代理，前提是有基础支持,” 2024 年 3 月, arXiv:2401.01614. [在线]. 可获取: http://arxiv.org/abs/2401.01614

[78] S. Zhou, F. F. Xu, H. Zhu, X. Zhou, R. Lo, A. Sridhar, X. Cheng, T. Ou, Y. Bisk, D. Fried, U. Alon, and G. Neubig, ”WebArena: A Realistic Web Environment for Building Autonomous Agents,” Apr. 2024, arXiv:2307.13854 TLDR: This paper builds an environment for language-guided agents that is highly realistic and reproducible, and creates an environment with fully functional websites from four common domains: e-commerce, social forum discussions, collaborative software development, and content management. [Online]. Available: http: //arxiv.org/abs/2307.13854

周松, 徐飞飞, 朱浩, 周翔, 罗瑞, Sridhar A., 程翔, 欧涛, Bisk Y., Fried D., Alon U., Neubig G., ”WebArena: 构建自主代理的真实网页环境,” 2024 年 4 月, arXiv:2307.13854 摘要: 本文构建了一个高度真实且可复现的语言引导代理环境，创建了包含四个常见领域 (电子商务、社交论坛讨论、协作软件开发和内容管理) 的全功能网站环境。[在线]. 可获取: http://arxiv.org/abs/2307.13854

[79] Z. Li, K. You, H. Zhang, D. Feng, H. Agrawal, X. Li, M. P. S. Moorthy, J. Nichols, Y. Yang, and Z. Gan, ”Ferret-UI 2: mastering universal user interface understanding across platforms,” Oct. 2024, arXiv:2410.18967. [Online]. Available: http://arxiv.org/abs/2410.18967

李志, 游凯, 张浩, 冯东, Agrawal H., 李翔, Moorthy M. P. S., Nichols J., 杨勇, 甘志, ”Ferret-UI 2: 跨平台掌握通用用户界面理解,” 2024 年 10 月, arXiv:2410.18967. [在线]. 可获取: http://arxiv.org/abs/2410.18967

[80] C. Rawles, A. Li, D. Rodriguez, O. Riva, and T. Lillicrap, ”Android in the Wild: A Large-Scale Dataset for Android Device Control,” Oct. 2023, arXiv:2307.10088 TLDR: A dataset for device-control research, Android in the Wild (AITW), which is orders of magnitude larger than current datasets, and contains multi-step tasks that require semantic understanding of language and visual context. [Online]. Available: http://arxiv.org/abs/2307.10088

Rawles C., 李安, Rodriguez D., Riva O., Lillicrap T., ”Android in the Wild: 大规模安卓设备控制数据集,” 2023 年 10 月, arXiv:2307.10088 摘要: 一个用于设备控制研究的数据集 Android in the Wild (AITW)，规模远超现有数据集，包含需要语言语义理解和视觉上下文的多步骤任务。[在线]. 可获取: http://arxiv.org/abs/2307.10088

[81] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann et al., ”Palm: Scaling language modeling with pathways,” Journal of Machine Learning Research, vol. 24, no. 240, pp. 1-113, 2023.

Chowdhery A., Narang S., Devlin J., Bosma M., Mishra G., Roberts A., Barham P., Chung H. W., Sutton C., Gehrmann S. 等, ”Palm: 通过路径 (Pathways) 扩展语言模型规模,” 机器学习研究杂志, 第 24 卷, 第 240 期, 页 1-113, 2023 年.

[82] D. Gao, L. Ji, Z. Bai, M. Ouyang, P. Li, D. Mao, Q. Wu, W. Zhang, P. Wang, X. Guo, H. Wang, L. Zhou, and M. Z. Shou, ”ASSISTGUI: task-oriented desktop graphical user interface automation,” Jan. 2024, arXiv:2312.13108 TLDR: An advanced Actor-Critic Embodied Agent framework is proposed, which incorporates a sophisticated GUI parser driven by an LLM-agent and an enhanced reasoning mechanism adept at handling lengthy procedural tasks that outshine existing methods in performance. [Online]. Available: http://arxiv.org/abs/2312.13108

高东, 纪力, 白志, 欧阳明, 李鹏, 毛东, 吴强, 张伟, 王鹏, 郭翔, 王浩, 周磊, 寿明哲, ”ASSISTGUI: 面向任务的桌面图形用户界面自动化,” 2024 年 1 月, arXiv:2312.13108 摘要: 提出了一种先进的演员-评论家 (Actor-Critic) 具身代理框架，结合由大型语言模型驱动的复杂 GUI 解析器和增强的推理机制，擅长处理长流程任务，性能优于现有方法。[在线]. 可获取: http://arxiv.org/abs/2312.13108

[83] OpenAI, ”ChatGPT,” https://openai.com/research/ chatgpt/, 2021.

OpenAI, ”ChatGPT,” https://openai.com/research/chatgpt/, 2021 年.

[84] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, J. Zhu, and L. Zhang, ”Grounding dino: Marrying dino with grounded pretraining for open-set object detection,” 2024. [Online]. Available: https://arxiv.org/abs/2303.05499

刘爽, 曾志, 任涛, 李飞, 张浩, 杨杰, 江强, 李超, 杨杰, 苏浩, 朱军, 张磊, ”Grounding dino: 将 dino 与有基础预训练结合用于开放集目标检测,” 2024 年. [在线]. 可获取: https://arxiv.org/abs/2303.05499

[85] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, ”Segment anything,” 2023. [Online]. Available: https://arxiv.org/abs/2304.02643

A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, 和 R. Girshick, “Segment anything(任意分割),”2023. [在线]. 可获取: https://arxiv.org/abs/2304.02643

[86] Y. Du, C. Li, R. Guo, X. Yin, W. Liu, J. Zhou, Y. Bai, Z. Yu, Y. Yang, Q. Dang, and H. Wang, ”Pp-ocr: A practical ultra lightweight ocr system,” 2020. [Online]. Available: https://arxiv.org/abs/2009.09941

Y. Du, C. Li, R. Guo, X. Yin, W. Liu, J. Zhou, Y. Bai, Z. Yu, Y. Yang, Q. Dang, 和 H. Wang, “Pp-ocr: 一个实用的超轻量级光学字符识别系统,”2020. [在线]. 可获取: https://arxiv.org/abs/2009.09941

[87] OpenAI, ”Operator system card,” January 2025, released on January 23, 2025.

OpenAI, “操作系统卡片,”2025 年 1 月，2025 年 1 月 23 日发布。

[88] Z. Li, K. You, H. Zhang, D. Feng, H. Agrawal, X. Li, M. P. S. Moorthy, J. Nichols, Y. Yang, and Z. Gan, ”Ferret-ui 2: Mastering universal user interface understanding across platforms,” 2024. [Online]. Available: https://arxiv.org/abs/2410.18967

Z. Li, K. You, H. Zhang, D. Feng, H. Agrawal, X. Li, M. P. S. Moorthy, J. Nichols, Y. Yang, 和 Z. Gan, “Ferret-ui 2: 跨平台通用用户界面理解的掌握,”2024. [在线]. 可获取: https://arxiv.org/abs/2410.18967

[89] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, ”Attention is all you need,” 2023. [Online]. Available: https://arxiv.org/abs/1706.03762

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, 和 I. Polosukhin, “Attention is all you need(注意力机制即一切),”2023. [在线]. 可获取: https://arxiv.org/abs/1706.03762

[90] L. Sun, X. Chen, L. Chen, T. Dai, Z. Zhu, and K. Yu, ”META-GUI: Towards Multi-modal Conversational Agents on Mobile GUI,” in Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Y. Goldberg, Z. Kozareva, and Y. Zhang, Eds. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 6699-6712. [Online]. Available: https://aclanthology.org/2022.emnlp-main.449

L. Sun, X. Chen, L. Chen, T. Dai, Z. Zhu, 和 K. Yu, “META-GUI: 面向移动图形用户界面的多模态对话代理,”载于 2022 年自然语言处理实证方法会议论文集，Y. Goldberg, Z. Kozareva, 和 Y. Zhang 编，阿布扎比，阿拉伯联合酋长国: 计算语言学协会，2022 年 12 月，第 6699-6712 页。[在线]. 可获取: https://aclanthology.org/2022.emnlp-main.449

[91] S. Ren, K. He, R. Girshick, and J. Sun, ”Faster r-cnn: Towards real-time object detection with region proposal networks,” 2016. [Online]. Available: https://arxiv.org/abs/1506.01497

S. Ren, K. He, R. Girshick, 和 J. Sun, “Faster r-cnn: 基于区域建议网络的实时目标检测,”2016. [在线]. 可获取: https://arxiv.org/abs/1506.01497

[92] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, ”Bert: Pre-training of deep bidirectional transformers for language understanding,” 2019. [Online]. Available: https://arxiv.org/abs/1810.04805

J. Devlin, M.-W. Chang, K. Lee, 和 K. Toutanova, “Bert: 用于语言理解的深度双向变换器预训练,”2019. [在线]. 可获取: https://arxiv.org/abs/1810.04805

[93] P. Wang, S. Bai, S. Tan, S. Wang, Z. Fan, J. Bai, K. Chen, X. Liu, J. Wang, W. Ge et al., ”Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution,” arXiv preprint arXiv:2409.12191, 2024.

P. Wang, S. Bai, S. Tan, S. Wang, Z. Fan, J. Bai, K. Chen, X. Liu, J. Wang, W. Ge 等, “Qwen2-vl: 提升视觉语言模型对任意分辨率世界的感知,”arXiv 预印本 arXiv:2409.12191, 2024.

[94] H. Liu, C. Li, Q. Wu, and Y. J. Lee, ”Visual instruction tuning,” Advances in neural information processing systems, vol. 36, 2024.

H. Liu, C. Li, Q. Wu, 和 Y. J. Lee, “视觉指令调优,”神经信息处理系统进展，卷 36，2024。

[95] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, ”An image is worth 16x16 words: Transformers for image recognition at scale,” 2021. [Online]. Available: https://arxiv.org/abs/2010.11929

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, 和 N. Houlsby, “一张图像胜过 16x16 个词: 大规模图像识别的变换器,”2021. [在线]. 可获取: https://arxiv.org/abs/2010.11929

[96] Y. Huang, T. Lv, L. Cui, Y. Lu, and F. Wei, ”LayoutImv3: Pre-training for document ai with unified text and image masking,” 2022. [Online]. Available: https://arxiv.org/abs/2204.08387

Y. Huang, T. Lv, L. Cui, Y. Lu, 和 F. Wei, “LayoutImv3: 统一文本与图像掩码的文档人工智能预训练,”2022. [在线]. 可获取: https://arxiv.org/abs/2204.08387

[97] X. Bi, D. Chen, G. Chen, S. Chen, D. Dai, C. Deng, H. Ding, K. Dong, Q. Du, Z. Fu et al., ”Deepseek llm: Scaling open-source language models with longter-mism,” arXiv preprint arXiv:2401.02954, 2024.

X. Bi, D. Chen, G. Chen, S. Chen, D. Dai, C. Deng, H. Ding, K. Dong, Q. Du, Z. Fu 等, “Deepseek llm: 扩展开源语言模型的长期记忆,”arXiv 预印本 arXiv:2401.02954, 2024。

[98] X. Liu, B. Qin, D. Liang, G. Dong, H. Lai, H. Zhang, H. Zhao, I. L. Iong, J. Sun, J. Wang, J. Gao, J. Shan, K. Liu, S. Zhang, S. Yao, S. Cheng, W. Yao, W. Zhao, X. Liu, X. Liu, X. Chen, X. Yang, Y. Yang, Y. Xu, Y. Yang, Y. Wang, Y. Xu, Z. Qi, Y. Dong, and J. Tang, ”AutoGLM: Autonomous Foundation Agents for GUIs,” Oct. 2024, arXiv:2411.00820. [Online]. Available: http://arxiv.org/abs/2411.00820

刘翔, 秦斌, 梁丹, 董刚, 赖浩, 张浩, 赵浩, I. L. Iong, 孙杰, 王军, 高杰, 单军, 刘凯, 张帅, 姚帅, 程帅, 姚伟, 赵伟, 刘翔, 刘翔, 陈翔, 杨翔, 杨洋, 许洋, 杨洋, 王洋, 许洋, 齐志, 董洋, 唐军, ”Auto-GLM: 用于图形用户界面 (GUIs) 的自主基础代理,” 2024 年 10 月, arXiv:2411.00820. [在线]. 可获取: http://arxiv.org/abs/2411.00820

[99] T. GLM, A. Zeng, B. Xu, B. Wang, C. Zhang, D. Yin, D. Zhang, D. Rojas, G. Feng, H. Zhao et al., ”Chatglm: A family of large language models from glm-130b to glm-4 all tools,” arXiv preprint arXiv:2406.12793, 2024.

T. GLM, A. 曾, B. 许, B. 王, C. 张, D. 尹, D. 张, D. 罗哈斯, G. 冯, H. 赵等, ”Chatglm: 从 glm-130b 到 glm-4 all tools 的大型语言模型家族,” arXiv 预印本 arXiv:2406.12793, 2024 年.

[100] Y. Xu, Z. Wang, J. Wang, D. Lu, T. Xie, A. Saha, D. Sahoo, T. Yu, and C. Xiong, ”Aguvis: Unified pure vision agents for autonomous gui interaction,” 2024. [Online]. Available: https://arxiv.org/abs/2412.04454

许洋, 王志, 王军, 卢丹, 谢涛, A. 萨哈, D. 萨胡, T. 余, C. 熊, ”Aguvis: 用于自主图形用户界面交互的统一纯视觉代理,” 2024 年. [在线]. 可获取: https://arxiv.org/abs/2412.04454

[101] Y. Yang, Y. Wang, D. Li, Z. Luo, B. Chen, C. Huang, and J. Li, ”Aria-ui: Visual grounding for gui instructions,” 2024. [Online]. Available: https://arxiv.org/abs/2412.16256

杨洋, 王洋, 李丹, 罗志, 陈斌, 黄超, 李军, ”Aria-ui: 图形用户界面指令的视觉定位,” 2024 年. [在线]. 可获取: https://arxiv.org/abs/2412.16256

[102] M. Deitke, C. Clark, S. Lee, R. Tripathi, Y. Yang, J. S. Park, M. Salehi, N. Muennighoff, K. Lo, L. Soldaini, J. Lu, T. Anderson, E. Bransom, K. Ehsani, H. Ngo, Y. Chen, A. Patel, M. Yatskar, C. Callison-Burch, A. Head, R. Hendrix, F. Bastani, E. VanderBilt, N. Lambert, Y. Chou, A. Chheda, J. Sparks, S. Skjonsberg, M. Schmitz, A. Sarnat, B. Bischoff, P. Walsh, C. Newell, P. Wolters, T. Gupta, K.-H. Zeng, J. Borchardt, D. Groeneveld, C. Nam, S. Lebrecht, C. Wittlif, C. Schoenick, O. Michel, R. Krishna, L. Weihs, N. A. Smith, H. Hajishirzi, R. Girshick, A. Farhadi, and A. Kembhavi, ”Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models,” 2024. [Online]. Available: https://arxiv.org/abs/2409.17146

M. Deitke, C. Clark, S. Lee, R. Tripathi, Y. Yang, J. S. Park, M. Salehi, N. Muennighoff, K. Lo, L. Soldaini, J. Lu, T. Anderson, E. Bransom, K. Ehsani, H. Ngo, Y. Chen, A. Patel, M. Yatskar, C. Callison-Burch, A. Head, R. Hendrix, F. Bastani, E. VanderBilt, N. Lambert, Y. Chou, A. Chheda, J. Sparks, S. Skjonsberg, M. Schmitz, A. Sarnat, B. Bischoff, P. Walsh, C. Newell, P. Wolters, T. Gupta, K.-H. Zeng, J. Borchardt, D. Groeneveld, C. Nam, S. Lebrecht, C. Wittlif, C. Schoenick, O. Michel, R. Krishna, L. Weihs, N. A. Smith, H. Hajishirzi, R. Girshick, A. Farhadi, A. Kembhavi, ”Molmo 和 Pixmo: 用于最先进视觉-语言模型的开放权重和开放数据,” 2024 年. [在线]. 可获取: https://arxiv.org/abs/2409.17146

[103] L. Zheng, R. Wang, X. Wang, and B. An, ”Synapse: Trajectory-as-exemplar prompting with memory for computer control,” 2024. [Online]. Available: https://arxiv.org/abs/2306.07863

郑磊, 王锐, 王翔, 安斌, ”Synapse: 以轨迹为示例的带记忆计算机控制提示,” 2024 年. [在线]. 可获取: https://arxiv.org/abs/2306.07863

[104] M. Kim, V. Bursztyn, E. Koh, S. Guo, and S.-w. Hwang, ”RaDA: Retrieval-augmented web agent planning with LLMs,” Bangkok, Thailand, pp. 13511-13525, Aug. 2024. [Online]. Available: https://aclanthology.org/2024.findings-acl.802

M. Kim, V. Bursztyn, E. Koh, S. Guo, S.-w. Hwang, ”RaDA: 基于大型语言模型的检索增强网页代理规划,” 泰国曼谷, 第 13511-13525 页, 2024 年 8 月. [在线]. 可获取: https://aclanthology.org/2024.findings-acl.802

[105] A. Zhao, D. Huang, Q. Xu, M. Lin, Y.-J. Liu, and G. Huang, ”Expel: Llm agents are experiential learners,” 2023. [Online]. Available: https://arxiv.org/abs/2308.10144

赵安, 黄丹, 徐强, 林明, 刘一杰, 黄刚, ”Expel: 大型语言模型代理的体验式学习者,” 2023 年. [在线]. 可获取: https://arxiv.org/abs/2308.10144

[106] Y. Fu, D.-K. Kim, J. Kim, S. Sohn, L. Logeswaran, K. Bae, and H. Lee, ”Autoguide: Automated generation and selection of state-aware guidelines for large language model agents,” 2024. [Online]. Available: https://arxiv.org/abs/2403.08978

傅洋, 金东奎, 金俊, 孙胜, L. Logeswaran, K. Bae, H. Lee, ”Autoguide: 大型语言模型代理的状态感知指南自动生成与选择,” 2024 年. [在线]. 可获取: https://arxiv.org/abs/2403.08978

[107] X. Zhu, Y. Chen, H. Tian, C. Tao, W. Su, C. Yang, G. Huang, B. Li, L. Lu, X. Wang, Y. Qiao, Z. Zhang, and J. Dai, ”Ghost in the minecraft: Generally capable agents for open-world environments via large language models with text-based knowledge and memory,” 2023. [Online]. Available: https://arxiv.org/abs/2305.17144

X. Zhu, Y. Chen, H. Tian, C. Tao, W. Su, C. Yang, G. Huang, B. Li, L. Lu, X. Wang, Y. Qiao, Z. Zhang, 和 J. Dai, “Minecraft 中的幽灵: 通过基于文本的知识和记忆，利用大型语言模型实现开放世界环境中的通用智能体，” 2023 年。[在线]. 可获取:https://arxiv.org/abs/2305.17144

[108] W. Zhou, Y. E. Jiang, P. Cui, T. Wang, Z. Xiao, Y. Hou, R. Cotterell, and M. Sachan, ”Recurrentgpt: Interactive generation of (arbitrarily) long text,” 2023. [Online]. Available: https://arxiv.org/abs/2305.13304

W. Zhou, Y. E. Jiang, P. Cui, T. Wang, Z. Xiao, Y. Hou, R. Cotterell, 和 M. Sachan, “Recurrentgpt: 交互式生成 (任意) 长文本,”2023 年。[在线]. 可获取:https://arxiv.org/abs/2305.13304

[109] C. Qian, S. Liang, Y. Qin, Y. Ye, X. Cong, Y. Lin, Y. Wu, Z. Liu, and M. Sun, ”Investigate-consolidate-exploit: A general strategy for inter-task agent self-evolution,” 2024. [Online]. Available: https://arxiv.org/abs/2401.13996

C. Qian, S. Liang, Y. Qin, Y. Ye, X. Cong, Y. Lin, Y. Wu, Z. Liu, 和 M. Sun, “调查-整合-利用: 一种通用的跨任务智能体自我进化策略,”2024 年。[在线]. 可获取:https://arxiv.org/abs/2401.13996

[110] H. Su, W. Shi, J. Kasai, Y. Wang, Y. Hu, M. Ostendorf, W. tau Yih, N. A. Smith, L. Zettlemoyer, and T. Yu, ”One embedder, any task: Instruction-finetuned text embeddings,” 2023. [Online]. Available: https://arxiv.org/abs/2212.09741

H. Su, W. Shi, J. Kasai, Y. Wang, Y. Hu, M. Ostendorf, W. tau Yih, N. A. Smith, L. Zettlemoyer, 和 T. Yu, “一个嵌入器, 任意任务: 指令微调文本嵌入,”2023 年。[在线]. 可获取:https://arxiv.org/abs/2212.09741

[111] S. Min, M. Lewis, L. Zettlemoyer, and H. Hajishirzi, ”Metaicl: Learning to learn in context,” May 2022. [Online]. Available: http://arxiv.org/abs/2110.15943

S. Min, M. Lewis, L. Zettlemoyer, 和 H. Hajishirzi, “Metaicl: 上下文中学习如何学习,”2022 年 5 月。[在线]. 可获取:http://arxiv.org/abs/2110.15943

[112] Y. Xia, T. Yu, Z. He, H. Zhao, J. McAuley, and S. Li, ”Aligning as debiasing: Causality-aware alignment via reinforcement learning with interventional feedback,” in Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), K. Duh, H. Gomez, and S. Bethard, Eds. Mexico City, Mexico: Association for Computational Linguistics, Jun. 2024, pp. 4684-4695. [Online]. Available: https: //aclanthology.org/2024.naacl-long.262

Y. Xia, T. Yu, Z. He, H. Zhao, J. McAuley, 和 S. Li, “将对齐视为去偏: 通过带有干预反馈的强化学习实现因果感知对齐,”发表于 2024 年北美计算语言学协会人类语言技术会议论文集 (第一卷: 长文), K. Duh, H. Gomez, 和 S. Bethard 编。墨西哥城, 墨西哥: 计算语言学协会, 2024 年 6 月, 第 4684-4695 页。[在线]. 可获取:https://aclanthology.org/2024.naacl-long.262

[113] J. Yuan, D. Du, H. Zhang, Z. Di, and U. Naseem, ”Reversal of thought: Enhancing large language models with preference-guided reverse reasoning warm-up,” 2024. [Online]. Available: https://arxiv.org/abs/2410.12323

J. Yuan, D. Du, H. Zhang, Z. Di, 和 U. Naseem, “逆向思维: 通过偏好引导的逆向推理预热增强大型语言模型,”2024 年。[在线]. 可获取:https://arxiv.org/abs/2410.12323

[114] X. Feng, Z. Wan, M. Wen, S. M. McAleer, Y. Wen, W. Zhang, and J. Wang, ”Alphazero-like tree-search can guide large language model decoding and training,” Feb. 2024. [Online]. Available: http://arxiv.org/abs/2309.17179

X. Feng, Z. Wan, M. Wen, S. M. McAleer, Y. Wen, W. Zhang, 和 J. Wang, “类似 AlphaZero 的树搜索可指导大型语言模型的解码与训练,”2024 年 2 月。[在线]. 可获取:http://arxiv.org/abs/2309.17179

[115] R. Ding, C. Zhang, L. Wang, Y. Xu, M. Ma, W. Zhang, S. Qin, S. Rajmohan, Q. Lin, and D. Zhang, ”Everything of thoughts: Defying the law of penrose triangle for thought generation,” Feb. 2024. [Online]. Available:

http://arxiv.org/abs/2311.04254

R. Ding, C. Zhang, L. Wang, Y. Xu, M. Ma, W. Zhang, S. Qin, S. Rajmohan, Q. Lin, 和 D. Zhang, “思维万象: 挑战彭罗斯三角定律的思维生成,”2024 年 2 月。[在线]. 可获取:http://arxiv.org/abs/2311.04254

[116] Z. Gao, B. Niu, X. He, H. Xu, H. Liu, A. Liu, X. Hu, and L. Wen, ”Interpretable contrastive monte carlo tree search reasoning,” Oct. 2024. [Online]. Available: http://arxiv.org/abs/2410.01707

Z. Gao, B. Niu, X. He, H. Xu, H. Liu, A. Liu, X. Hu, 和 L. Wen, “可解释的对比蒙特卡洛树搜索推理,”2024 年 10 月。[在线]. 可获取:http://arxiv.org/abs/2410.01707

[117] A. Zhou, K. Yan, M. Shlapentokh-Rothman, H. Wang, and Y.-X. Wang, ”Language agent tree search unifies reasoning acting and planning in language models,” Jun. 2024. [Online]. Available: http://arxiv.org/abs/2310.04406

A. Zhou, K. Yan, M. Shlapentokh-Rothman, H. Wang, 和 Y.-X. Wang, “语言智能体树搜索统一了语言模型中的推理、行动与规划,”2024 年 6 月。[在线]. 可获取:http://arxiv.org/abs/2310.04406

[118] S. Hao, Y. Gu, H. Ma, J. J. Hong, Z. Wang, D. Z. Wang, and Z. Hu, ”Reasoning with language model is planning with world model,” Oct. 2023. [Online]. Available: http://arxiv.org/abs/2305.14992

S. Hao, Y. Gu, H. Ma, J. J. Hong, Z. Wang, D. Z. Wang, 和 Z. Hu, “用语言模型推理即是用世界模型规划,”2023 年 10 月。[在线]. 可获取:http://arxiv.org/abs/2305.14992

[119] H. Sun, M. Haider, R. Zhang, H. Yang, J. Qiu, M. Yin, M. Wang, P. Bartlett, and A. Zanette, ”Fast best-of-n decoding via speculative rejection,” Oct. 2024. [Online]. Available: http://arxiv.org/abs/2410.20290

H. Sun, M. Haider, R. Zhang, H. Yang, J. Qiu, M. Yin, M. Wang, P. Bartlett, 和 A. Zanette, “通过推测性拒绝实现快速的最佳 n 解码”，2024 年 10 月。[在线]. 可获取: http://arxiv.org/abs/2410.20290

[120] P. Wang, L. Li, Z. Shao, R. X. Xu, D. Dai, Y. Li, D. Chen, Y. Wu, and Z. Sui, ”Math-shepherd: Verify and reinforce llms step-by-step without human annotations,” Feb. 2024. [Online]. Available: http://arxiv.org/abs/2312.08935

P. Wang, L. Li, Z. Shao, R. X. Xu, D. Dai, Y. Li, D. Chen, Y. Wu, 和 Z. Sui, “Math-shepherd: 无需人工标注的逐步验证与强化大型语言模型 (LLMs)” ，2024 年 2 月。[在线]. 可获取: http://arxiv.org/abs/2312.08935

[121] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe, ”Let’s verify step by step,” May 2023. [Online]. Available: http://arxiv.org/abs/2305.20050

H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, 和 K. Cobbe, “让我们一步步验证”，2023 年 5 月。[在线]. 可获取: http://arxiv.org/abs/2305.20050

[122] Y. Shang, Y. Li, F. Xu, and Y. Li, ”Synergy-of-thoughts: Eliciting efficient reasoning in hybrid language models,” Aug. 2024. [Online]. Available: http://arxiv.org/abs/2402.02563

Y. Shang, Y. Li, F. Xu, 和 Y. Li, “思维协同: 在混合语言模型中引导高效推理”，2024 年 8 月。[在线]. 可获取: http://arxiv.org/abs/2402.02563

[123] A. Setlur, C. Nagpal, A. Fisch, X. Geng, J. Eisenstein, R. Agarwal, A. Agarwal, J. Berant, and A. Kumar, ”Rewarding progress: Scaling automated process verifiers for llm reasoning,” Oct. 2024. [Online]. Available: http://arxiv.org/abs/2410.08146

A. Setlur, C. Nagpal, A. Fisch, X. Geng, J. Eisenstein, R. Agarwal, A. Agarwal, J. Berant, 和 A. Kumar, “奖励进展: 扩展自动化过程验证器以支持大型语言模型推理”，2024 年 10 月。[在线]. 可获取: http://arxiv.org/abs/2410.08146

[124] L. Zhang, A. Hosseini, H. Bansal, M. Kazemi, A. Kumar, and R. Agarwal, ”Generative verifiers: Reward modeling as next-token prediction,” Oct. 2024. [Online]. Available: http://arxiv.org/abs/2408.15240

L. Zhang, A. Hosseini, H. Bansal, M. Kazemi, A. Kumar, 和 R. Agarwal, “生成式验证器: 将奖励建模视为下一个词预测”，2024 年 10 月。[在线]. 可获取: http://arxiv.org/abs/2408.15240

[125] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica, ”Judging llm-as-a-judge with mt-bench and chatbot arena,” Dec. 2023. [Online]. Available: http://arxiv.org/abs/2306.05685

L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, 和 I. Stoica, “用 MT-Bench 和 Chatbot Arena 评估‘大型语言模型作为裁判’”，2023 年 12 月。[在线]. 可获取: http://arxiv.org/abs/2306.05685

[126] Z. Qi, M. Ma, J. Xu, L. L. Zhang, F. Yang, and M. Yang, ”Mutual reasoning makes smaller llms stronger problem-solvers,” Aug. 2024. [Online]. Available: http://arxiv.org/abs/2408.06195

Z. Qi, M. Ma, J. Xu, L. L. Zhang, F. Yang, 和 M. Yang, “相互推理使小型大型语言模型成为更强的问题解决者”，2024 年 8 月。[在线]. 可获取: http://arxiv.org/abs/2408.06195

[127] D. Zhou, N. Schärli, L. Hou, J. Wei, N. Scales, X. Wang, D. Schuurmans, C. Cui, O. Bousquet, Q. Le, and E. Chi, ”Least-to-most prompting enables complex reasoning in large language models,” Apr. 2023. [Online]. Available: http://arxiv.org/abs/2205.10625

D. Zhou, N. Schärli, L. Hou, J. Wei, N. Scales, X. Wang, D. Schuurmans, C. Cui, O. Bousquet, Q. Le, 和 E. Chi, “由少及多提示法促进大型语言模型的复杂推理”，2023 年 4 月。[在线]. 可获取: http://arxiv.org/abs/2205.10625

[128] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I.

Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom, ”Llama 2: Open foundation and fine-tuned chat models,” Jul. 2023. [Online]. Available: http://arxiv.org/abs/2307.09288

H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom, ”Llama 2: 开放基础模型与微调聊天模型,” 2023 年 7 月。[在线]. 可获取: http://arxiv.org/abs/2307.09288

[129] X. Huang, W. Liu, X. Chen, X. Wang, H. Wang, D. Lian, Y. Wang, R. Tang, and E. Chen, ”Understanding the planning of llm agents: A survey,” 2024. [Online]. Available: https://arxiv.org/abs/2402.02716

X. Huang, W. Liu, X. Chen, X. Wang, H. Wang, D. Lian, Y. Wang, R. Tang, 和 E. Chen, ” 理解大型语言模型代理的规划: 一项综述,” 2024 年。[在线]. 可获取: https://arxiv.org/abs/2402.02716

[130] W. Yuan, R. Y. Pang, K. Cho, X. Li, S. Sukhbaatar, J. Xu, and J. Weston, ”Self-rewarding language models,” Feb. 2024. [Online]. Available: http://arxiv.org/abs/2401.10020

W. Yuan, R. Y. Pang, K. Cho, X. Li, S. Sukhbaatar, J. Xu, 和 J. Weston, ” 自我奖励语言模型,” 2024 年 2 月。[在线]. 可获取: http://arxiv.org/abs/2401.10020

[131] J. Huang, Q. She, W. Jiang, H. Wu, Y. Hao, T. Xu, and F. Wu, ”Qdmr-based planning-and-solving prompting for complex reasoning tasks.”

J. Huang, Q. She, W. Jiang, H. Wu, Y. Hao, T. Xu, 和 F. Wu, ” 基于 QDMR 的规划与求解提示方法用于复杂推理任务.”

[132] E. Zelikman, Y. Wu, J. Mu, and N. D. Goodman, ”Star: Bootstrapping reasoning with reasoning,” May 2022. [Online]. Available: http://arxiv.org/abs/2203.14465

E. Zelikman, Y. Wu, J. Mu, 和 N. D. Goodman, “Star: 用推理引导推理的自举方法,”2022 年 5 月。[在线]. 可获取:http://arxiv.org/abs/2203.14465

[133] A. Hosseini, X. Yuan, N. Malkin, A. Courville, A. Sordoni, and R. Agarwal, ”V-star: Training verifiers for self-taught reasoners,” Aug. 2024. [Online]. Available: http://arxiv.org/abs/2402.06457

A. Hosseini, X. Yuan, N. Malkin, A. Courville, A. Sordoni, 和 R. Agarwal, “V-star: 为自学推理者训练验证器,”2024 年 8 月。[在线]. 可获取:http://arxiv.org/abs/2402.06457

[134] A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang, S. Gupta, B. P. Majumder, K. Hermann, S. Welleck, A. Yazdanbakhsh, and P. Clark, ”Self-refine: Iterative refinement with self-feedback,” May 2023. [Online]. Available: http://arxiv.org/abs/2303.17651

A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang, S. Gupta, B. P. Majumder, K. Hermann, S. Welleck, A. Yazdanbakhsh, 和 P. Clark, “Self-refine: 基于自我反馈的迭代精炼,”2023 年 5 月。[在线]. 可获取:http://arxiv.org/abs/2303.17651

[135] Z. Yuan, H. Yuan, C. Li, G. Dong, K. Lu, C. Tan, C. Zhou, and J. Zhou, ”Scaling relationship on learning mathematical reasoning with large language models,” Sep. 2023. [Online]. Available: http://arxiv.org/abs/2308.01825

Z. Yuan, H. Yuan, C. Li, G. Dong, K. Lu, C. Tan, C. Zhou, 和 J. Zhou, “大规模语言模型中数学推理学习的规模关系,”2023 年 9 月。[在线]. 可获取:http://arxiv.org/abs/2308.01825

[136] E. Zelikman, G. Harik, Y. Shao, V. Jayasiri, N. Haber, and N. D. Goodman, ”Quiet-star: Language models can teach themselves to think before speaking,” Mar. 2024. [Online]. Available: http://arxiv.org/abs/2403.09629

E. Zelikman, G. Harik, Y. Shao, V. Jayasiri, N. Haber, 和 N. D. Goodman, “Quiet-star: 语言模型能够自我学习先思考后发言,”2024 年 3 月。[在线]. 可获取:http://arxiv.org/abs/2403.09629

[137] X. Zhang, C. Du, T. Pang, Q. Liu, W. Gao, and M. Lin, ”Chain of preference optimization: Improving chain-of-thought reasoning in llms,” Oct. 2024. [Online]. Available: http://arxiv.org/abs/2406.09136

X. Zhang, C. Du, T. Pang, Q. Liu, W. Gao, 和 M. Lin, “偏好链优化: 提升大语言模型中的链式思维推理,”2024 年 10 月。[在线]. 可获取:http://arxiv.org/abs/2406.09136

[138] C. Andukuri, J.-P. Fränken, T. Gerstenberg, and N. D. Goodman, ”Star-gate: Teaching language models to ask clarifying questions,” Aug. 2024. [Online]. Available: http://arxiv.org/abs/2403.19154

C. Andukuri, J.-P. Fränken, T. Gerstenberg, 和 N. D. Goodman, “Star-gate: 教语言模型提出澄清性问题,”2024 年 8 月。[在线]. 可获取:http://arxiv.org/abs/2403.19154

[139] Z. Zhang, A. Zhang, M. Li, H. Zhao, G. Karypis, and A. Smola, ”Multimodal chain-of-thought reasoning in language models,” May 2024. [Online]. Available: http://arxiv.org/abs/2302.00923

Z. Zhang, A. Zhang, M. Li, H. Zhao, G. Karypis, 和 A. Smola, “语言模型中的多模态链式思维推理,”2024 年 5 月。[在线]. 可获取:http://arxiv.org/abs/2302.00923

[140] Y. Wu, P. Zhang, W. Xiong, B. Oguz, J. C. Gee, and Y. Nie, ”The role of chain-of-thought in complex vision-language reasoning task,” Nov. 2023. [Online]. Available: http://arxiv.org/abs/2311.09193

Y. Wu, P. Zhang, W. Xiong, B. Oguz, J. C. Gee, 和 Y. Nie, “链式思维在复杂视觉-语言推理任务中的作用,”2023 年 11 月。[在线]. 可获取:http://arxiv.org/abs/2311.09193

[141] P. Lu, S. Mishra, T. Xia, L. Qiu, K.-W. Chang, S.-C. Zhu, O. Tafjord, P. Clark, and A. Kalyan, ”Learn to explain: Multimodal reasoning via thought chains for science question answering.”

P. Lu, S. Mishra, T. Xia, L. Qiu, K.-W. Chang, S.-C. Zhu, O. Tafjord, P. Clark, 和 A. Kalyan, “学会解释: 通过思维链进行多模态推理以回答科学问题。”

[142] D. Mondal, S. Modi, S. Panda, R. Singh, and G. S. Rao, ”Kam-cot: Knowledge augmented multimodal chain-of-thoughts reasoning,” Jan. 2024. [Online]. Available: http://arxiv.org/abs/2401.12863

D. Mondal, S. Modi, S. Panda, R. Singh, 和 G. S. Rao, “Kam-cot: 知识增强的多模态链式思维推理,” 2024 年 1 月。[在线]. 可获取:http://arxiv.org/abs/2401.12863

[143] C. Mitra, B. Huang, T. Darrell, and R. Herzig, ”Compositional chain-of-thought prompting for large multimodal models.”

C. Mitra, B. Huang, T. Darrell, 和 R. Herzig, “大规模多模态模型的组合式链式思维提示。”

[144] H. Shao, S. Qian, H. Xiao, G. Song, Z. Zong, L. Wang, Y. Liu, and H. Li, ”Visual cot: Advancing multi-modal language models with a comprehensive dataset and benchmark for chain-of-thought reasoning,” Nov. 2024. [Online]. Available: http://arxiv.org/abs/2403.16999

H. Shao, S. Qian, H. Xiao, G. Song, Z. Zong, L. Wang, Y. Liu, 和 H. Li, “Visual cot: 通过全面的数据集和基准推动多模态语言模型的链式思维推理,”2024 年 11 月。[在线]. 可获取:http://arxiv.org/abs/2403.16999

[145] Z. Li, R. Luo, J. Zhang, M. Qiu, and Z. Wei, ”Vocot: Unleashing visually grounded multi-step reasoning in large multi-modal models,” May 2024. [Online]. Available: http://arxiv.org/abs/2405.16919

Z. Li, R. Luo, J. Zhang, M. Qiu, 和 Z. Wei, “Vocot: 释放大规模多模态模型中视觉基础的多步推理能力,”2024 年 5 月。[在线]. 可获取:http://arxiv.org/abs/2405.16919

[146] D. Zhang, J. Yang, H. Lyu, Z. Jin, Y. Yao, M. Chen, and J. Luo, ”Cocot: Contrastive chain-of-thought prompting for large multimodal models with multiple image inputs,” Jan. 2024. [Online]. Available: http://arxiv.org/abs/2401.02582

张丹, 杨杰, 吕浩, 金志, 姚颖, 陈明, 罗军, ”Cocot: 用于多图像输入的大型多模态模型的对比链式思维提示,” 2024 年 1 月. [在线]. 可获取: http://arxiv.org/abs/2401.02582

[147] G. Zheng, B. Yang, J. Tang, H.-Y. Zhou, and S. Yang, ”Ddcot: Duty-distinct chain-of-thought prompting for multimodal reasoning in language models,” Oct. 2023. [Online]. Available: http://arxiv.org/abs/2310.16436

郑刚, 杨斌, 唐杰, 周海洋, 杨松, ”Ddcot: 用于语言模型多模态推理的职责区分链式思维提示,” 2023 年 10 月. [在线]. 可获取: http://arxiv.org/abs/2310.16436

[148] B. Luan, H. Feng, H. Chen, Y. Wang, W. Zhou, and H. Li, ”Textcot: Zoom in for enhanced multimodal text-rich image understanding,” Apr. 2024. [Online]. Available: http://arxiv.org/abs/2404.09797

栾斌, 冯浩, 陈浩, 王勇, 周伟, 李浩, ”Textcot: 聚焦以增强多模态文本丰富图像理解,” 2024 年 4 月. [在线]. 可获取: http://arxiv.org/abs/2404.09797

[149] C. Tan, J. Wei, Z. Gao, L. Sun, S. Li, R. Guo, B. Yu, and S. Z. Li, ”Boosting the power of small multimodal reasoning models to match larger models with self-consistency training.”

谭超, 魏军, 高志, 孙磊, 李爽, 郭锐, 余斌, 李双志, ” 通过自洽训练提升小型多模态推理模型的能力以匹配大型模型.”

[150] J. Qi, M. Ding, W. Wang, Y. Bai, Q. Lv, W. Hong, B. Xu, L. Hou, J. Li, Y. Dong, and J. Tang, ”Cogcom: Train large vision-language models diving into details through chain of manipulations,” May 2024. [Online]. Available: http://arxiv.org/abs/2402.04236

齐军, 丁明, 王伟, 白洋, 吕强, 洪伟, 徐斌, 侯磊, 李军, 董阳, 唐杰, ”Cogcom: 通过操作链深入细节训练大型视觉-语言模型,” 2024 年 5 月. [在线]. 可获取: http://arxiv.org/abs/2402.04236

[151] S. Menon, R. Zemel, and C. Vondrick, ”Whiteboard-of-thought: Thinking step-by-step across modalities,” Jun. 2024. [Online]. Available: http://arxiv.org/abs/ 2406.14562

梅农, 泽梅尔, 冯德里克, ” 思维白板: 跨模态逐步思考,” 2024 年 6 月. [在线]. 可获取: http://arxiv.org/abs/2406.14562

[152] L. Wei, W. Wang, X. Shen, Y. Xie, Z. Fan, X. Zhang, Z. Wei, and W. Chen, ”Mc-cot: A modular collaborative cot framework for zero-shot medical-vqa with llm and mllm integration,” Oct. 2024. [Online]. Available: http://arxiv.org/abs/2410.04521

魏磊, 王伟, 沈翔, 谢阳, 范志, 张翔, 魏志, 陈伟, ”Mc-cot: 用于零样本医学视觉问答的模块化协作链式思维框架，结合大语言模型和多模态大语言模型,” 2024 年 10 月. [在线]. 可获取: http://arxiv.org/abs/2410.04521

[153] D. Xin, X. Tan, K. Shen, Z. Ju, D. Yang, Y. Wang, S. Takamichi, H. Saruwatari, S. Liu, J. Li, and S. Zhao, ”Rall-e: Robust codec language modeling with chain-of-thought prompting for text-to-speech synthesis,” May 2024. [Online]. Available: http://arxiv.org/abs/2404.03204

辛东, 谭翔, 沈凯, 鞠志, 杨东, 王勇, 高桥, 猿渡, 刘松, 李军, 赵松, ”Rall-e: 结合链式思维提示的鲁棒编解码语言模型用于文本到语音合成,” 2024 年 5 月. [在线]. 可获取: http://arxiv.org/abs/2404.03204

[154] H. Gong and B. Veluri, ”Seamlessexpressivelm: Speech language model for expressive speech-to-speech translation with chain-of-thought,” May 2024. [Online]. Available: http://arxiv.org/abs/2405.20410

龚浩, 维鲁里, ”Seamlessexpressivelm: 用于富有表现力的语音到语音翻译的语音语言模型，结合链式思维,” 2024 年 5 月. [在线]. 可获取: http://arxiv.org/abs/2405.20410

[155] K. Hu, Z. Chen, C.-H. H. Yang, P. Zelasko, O. Hrinchuk, V. Lavrukhin, J. Balam, and B. Ginsburg, ”Chain-of-thought prompting for speech translation,” Sep. 2024. [Online]. Available: http://arxiv.org/abs/2409.11538

胡凯, 陈志, 杨承翰, 泽拉斯科, 赫林丘克, 拉夫鲁金, 巴拉姆, 金斯堡, ” 链式思维提示用于语音翻译,” 2024 年 9 月. [在线]. 可获取: http://arxiv.org/abs/2409.11538

[156] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. K. Li, Y. Wu, and D. Guo,

”Deepseekmath: Pushing the limits of mathematical reasoning in open language models,” 2024. [Online]. Available: https://arxiv.org/abs/2402.03300

邵志, 王鹏, 朱强, 徐睿, 宋杰, 毕翔, 张浩, 张明, 李永康, 吴洋, 郭东, ”Deepseekmath: 推动开放语言模型中数学推理的极限,” 2024 年. [在线]. 可获取: https://arxiv.org/abs/2402.03300

[157] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, ”Proximal policy optimization algorithms,” 2017. [Online]. Available: https://arxiv.org/abs/1707.06347

舒尔曼, 沃尔斯基, 达里瓦尔, 拉德福, 克里莫夫, ” 近端策略优化算法,” 2017 年. [在线]. 可获取: https://arxiv.org/abs/1707.06347

[158] T. Xie, Z. Gao, Q. Ren, H. Luo, Y. Hong, B. Dai, J. Zhou, K. Qiu, Z. Wu, and C. Luo, ”Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning,” 2025. [Online]. Available: https://arxiv.org/abs/2502.14768

谢涛, 高志, 任强, 罗浩, 洪阳, 戴斌, 周军, 邱凯, 吴志, 罗成, ”Logic-rl: 结合基于规则的强化学习释放大型语言模型推理能力,” 2025 年. [在线]. 可获取: https://arxiv.org/abs/2502.14768

[159] H. Song, J. Jiang, Y. Min, J. Chen, Z. Chen, W. X. Zhao, L. Fang, and J.-R. Wen, ”R1-searcher: Incentivizing the search capability in llms via reinforcement learning,” 2025. [Online]. Available: https://arxiv.org/abs/2503.05592

H. Song, J. Jiang, Y. Min, J. Chen, Z. Chen, W. X. Zhao, L. Fang, 和 J.-R. Wen, ”R1-searcher: 通过强化学习激励大语言模型 (LLMs) 的搜索能力,” 2025 年。[在线]. 可获取:https://arxiv.org/abs/2503.05592

[160] W. Huang, B. Jia, Z. Zhai, S. Cao, Z. Ye, F. Zhao, Z. Xu, Y. Hu, and S. Lin, ”Vision-r1: Incentivizing reasoning capability in multimodal large language models,” 2025. [Online]. Available: https://arxiv.org/abs/2503.06749

W. Huang, B. Jia, Z. Zhai, S. Cao, Z. Ye, F. Zhao, Z. Xu, Y. Hu, 和 S. Lin, ”Vision-r1: 激励多模态大语言模型的推理能力,” 2025 年。[在线]. 可获取:https://arxiv.org/abs/2503.06749

[161] Y. Peng, G. Zhang, M. Zhang, Z. You, J. Liu, Q. Zhu, K. Yang, X. Xu, X. Geng, and X. Yang, ”Lmm-r1: Empowering 3b lmms with strong reasoning abilities through two-stage rule-based rl,” 2025. [Online]. Available: https://arxiv.org/abs/2503.07536

Y. Peng, G. Zhang, M. Zhang, Z. You, J. Liu, Q. Zhu, K. Yang, X. Xu, X. Geng, 和 X. Yang, ”Lmm-r1: 通过两阶段基于规则的强化学习赋能 3B 参数大语言模型强推理能力,” 2025 年。[在线]. 可获取:https://arxiv.org/abs/2503.07536

[162] T. Masterman, S. Besen, M. Sawtell, and A. Chao, ”The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey,” Apr. 2024. [Online]. Available: http://arxiv.org/abs/2404.11584

T. Masterman, S. Besen, M. Sawtell, 和 A. Chao, ” 新兴人工智能代理架构在推理、规划和工具调用中的全景: 一项综述,” 2024 年 4 月。[在线]. 可获取:http://arxiv.org/abs/2404.11584

[163] K. Cheng, Q. Sun, Y. Chu, F. Xu, Y. Li, J. Zhang, and Z. Wu, ”Seeclick: Harnessing gui grounding for advanced visual gui agents,” 2024. [Online]. Available: https://arxiv.org/abs/2401.10935

K. Cheng, Q. Sun, Y. Chu, F. Xu, Y. Li, J. Zhang, 和 Z. Wu, ”Seeclick: 利用 GUI 定位技术打造高级视觉 GUI 代理,” 2024 年。[在线]. 可获取:https://arxiv.org/abs/2401.10935

[164] G. Baechler, S. Sunkara, M. Wang, F. Zubach, H. Mansoor, V. Etter, V. Cărbune, J. Lin, J. Chen, and A. Sharma, ”Screenai: A vision-language model for ui and infographics understanding,” 2024. [Online]. Available: https://arxiv.org/abs/2402.04615

G. Baechler, S. Sunkara, M. Wang, F. Zubach, H. Mansoor, V. Etter, V. Cărbune, J. Lin, J. Chen, 和 A. Sharma, ”Screenai: 用于界面和信息图理解的视觉-语言模型,” 2024 年。[在线]. 可获取:https://arxiv.org/abs/2402.04615

[165] P. Shaw, M. Joshi, J. Cohan, J. Berant, P. Pasupat, H. Hu, U. Khandelwal, K. Lee, and K. Toutanova, ”From pixels to ui actions: Learning to follow instructions via graphical user interfaces,” 2023. [Online]. Available: https://arxiv.org/abs/2306.00245

P. Shaw, M. Joshi, J. Cohan, J. Berant, P. Pasupat, H. Hu, U. Khandelwal, K. Lee, 和 K. Toutanova, ” 从像素到界面操作: 通过图形用户界面学习执行指令,” 2023 年。[在线]. 可获取:https://arxiv.org/abs/2306.00245

[166] Y. Qin, E. Zhou, Q. Liu, Z. Yin, L. Sheng, R. Zhang, Y. Qiao, and J. Shao, ”Mp5: A multimodal open-ended embodied system in minecraft via active perception,” 2024. [Online]. Available: https://arxiv.org/abs/2312.07472

Y. Qin, E. Zhou, Q. Liu, Z. Yin, L. Sheng, R. Zhang, Y. Qiao, 和 J. Shao, ”Mp5: 通过主动感知实现的 Minecraft 多模态开放式具身系统,” 2024 年。[在线]. 可获取:https://arxiv.org/abs/2312.07472

[167] J. Wang, H. Xu, H. Jia, X. Zhang, M. Yan, W. Shen, J. Zhang, F. Huang, and J. Sang, ”Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration,” Jun. 2024. [Online]. Available: http://arxiv.org/abs/2406.01014

J. Wang, H. Xu, H. Jia, X. Zhang, M. Yan, W. Shen, J. Zhang, F. Huang, 和 J. Sang, ”Mobile-agent-v2: 通过多代理协作实现高效导航的移动设备操作助手,” 2024 年 6 月。[在线]. 可获取:http://arxiv.org/abs/2406.01014

[168] Y. Hu, Y. Cai, Y. Du, X. Zhu, X. Liu, Z. Yu, Y. Hou, S. Tang, and S. Chen, ”Self-evolving multi-agent collaboration networks for software development,” 2024. [Online]. Available: https://arxiv.org/abs/2410.16946

Y. Hu, Y. Cai, Y. Du, X. Zhu, X. Liu, Z. Yu, Y. Hou, S. Tang, 和 S. Chen, ” 自我进化的多代理协作网络用于软件开发,” 2024 年。[在线]. 可获取:https://arxiv.org/abs/2410.16946

[169] A. Yan, Z. Yang, W. Zhu, K. Lin, L. Li, J. Wang, J. Yang, Y. Zhong, J. McAuley, J. Gao, Z. Liu, and L. Wang, ”GPT-4V in Wonderland: Large Multimodal Models for Zero-Shot Smartphone GUI Navigation,” arXiv e-prints, p. arXiv:2311.07562, Nov. 2023.

A. Yan, Z. Yang, W. Zhu, K. Lin, L. Li, J. Wang, J. Yang, Y. Zhong, J. McAuley, J. Gao, Z. Liu, 和 L. Wang, ”GPT-4V 在奇境: 用于零样本智能手机 GUI 导航的大型多模态模型,” arXiv 电子预印本，编号 arXiv:2311.07562，2023 年 11 月。

[170] H. Lai, X. Liu, I. L. Iong, S. Yao, Y. Chen, P. Shen, H. Yu, H. Zhang, X. Zhang, Y. Dong, and J. Tang, ”AutoWebGLM: A Large Language Model-based Web Navigating Agent,” Oct. 2024, arXiv:2404.03648 TLDR: Inspired by human browsing patterns, an HTML simplification algorithm is designed to represent webpages, preserving vital information succinctly and bootstrap the model by reinforcement learning and rejection sampling to further facilitate webpage comprehension, browser operations, and efficient task decomposition by itself. [Online]. Available: http://arxiv.org/abs/2404.03648

H. Lai, X. Liu, I. L. Iong, S. Yao, Y. Chen, P. Shen, H. Yu, H. Zhang, X. Zhang, Y. Dong, 和 J. Tang, ”AutoWebGLM: 基于大型语言模型的网页导航代理,” 2024 年 10 月, arXiv:2404.03648 摘要: 受人类浏览模式启发，设计了一种 HTML 简化算法以表示网页，简洁地保留关键信息，并通过强化学习和拒绝采样引导模型，进一步促进网页理解、浏览器操作及高效任务分解。

[171] J. Kil, C. H. Song, B. Zheng, X. Deng, Y. Su, and W.-L. Chao, ”Dual-View Visual Contextualization for Web Navigation,” arXiv e-prints, p. arXiv:2402.04476, Feb. 2024.

J. Kil, C. H. Song, B. Zheng, X. Deng, Y. Su, 和 W.-L. Chao, ” 双视角视觉上下文化用于网页导航,” arXiv 电子预印本, p. arXiv:2402.04476, 2024 年 2 月。

[172] S. Agashe, J. Han, S. Gan, J. Yang, A. Li, and X. E. Wang, ”Agent S: An Open Agentic Framework that Uses Computers Like a Human,” arXiv e-prints, p. arXiv:2410.08164, Oct. 2024.

S. Agashe, J. Han, S. Gan, J. Yang, A. Li, 和 X. E. Wang, ”Agent S: 一个像人类一样使用计算机的开放代理框架,” arXiv 电子预印本, p. arXiv:2410.08164, 2024 年 10 月。

[173] Z. Zhang and A. Zhang, ”You Only Look at Screens: Multimodal Chain-of-Action Agents,” arXiv e-prints, p. arXiv:2309.11436, Sep. 2023.

Z. Zhang 和 A. Zhang, ” 你只需看屏幕: 多模态行动链代理,” arXiv 电子预印本, p. arXiv:2309.11436, 2023 年 9 月。

[174] J. Zhang, J. Wu, Y. Teng, M. Liao, N. Xu, X. Xiao, Z. Wei, and D. Tang, ”Android in the Zoo: Chain-of-Action-Thought for GUI Agents,” arXiv e-prints, p. arXiv:2403.02713, Mar. 2024.

J. Zhang, J. Wu, Y. Teng, M. Liao, N. Xu, X. Xiao, Z. Wei, 和 D. Tang, ”Android 动物园:GUI 代理的行动思维链,” arXiv 电子预印本, p. arXiv:2403.02713, 2024 年 3 月。

[175] T. Shi, A. Karpathy, L. Fan, J. Hernandez, and P. Liang, ”World of bits: An open-domain platform for web-based agents,” in International Conference on Machine Learning. PMLR, 2017, pp. 3135-3144.

T. Shi, A. Karpathy, L. Fan, J. Hernandez, 和 P. Liang, ” 比特世界: 一个面向网页代理的开放域平台,” 载于国际机器学习会议。PMLR, 2017, 页 3135-3144。

[176] B. Deka, Z. Huang, C. Franzen, J. Hibschman, D. Afer-gan, Y. Li, J. Nichols, and R. Kumar, ”Rico: A mobile app dataset for building data-driven design applications,” in Proceedings of the 30th annual ACM symposium on user interface software and technology, 2017, pp. 845-854.

B. Deka, Z. Huang, C. Franzen, J. Hibschman, D. Afergan, Y. Li, J. Nichols, 和 R. Kumar, ”Rico: 用于构建数据驱动设计应用的移动应用数据集,” 载于第 30 届年度 ACM 用户界面软件与技术研讨会论文集, 2017, 页 845-854。

[177] E. Z. Liu, K. Guu, P. Pasupat, T. Shi, and P. Liang, ”Reinforcement learning on web interfaces using workflow-guided exploration,” arXiv preprint arXiv:1802.08802, 2018.

E. Z. Liu, K. Guu, P. Pasupat, T. Shi, 和 P. Liang, ” 基于工作流引导探索的网页界面强化学习,” arXiv 预印本 arXiv:1802.08802, 2018 年。

[178] Y. Li, J. He, X. Zhou, Y. Zhang, and J. Baldridge, ”Mapping natural language instructions to mobile ui action sequences,” arXiv preprint arXiv:2005.03776, 2020.

Y. Li, J. He, X. Zhou, Y. Zhang, 和 J. Baldridge, ” 将自然语言指令映射到移动 UI 动作序列,” arXiv 预印本 arXiv:2005.03776, 2020 年。

[179] X. Chen, Z. Zhao, L. Chen, D. Zhang, J. Ji, A. Luo, Y. Xiong, and K. Yu, ”Websrc: A dataset for web-based structural reading comprehension,” arXiv preprint arXiv:2101.09465, 2021.

X. Chen, Z. Zhao, L. Chen, D. Zhang, J. Ji, A. Luo, Y. Xiong, 和 K. Yu, ”Websrc: 一个面向网页结构化阅读理解的数据集,” arXiv 预印本 arXiv:2101.09465, 2021 年。

[180] A. Burns, D. Arsan, S. Agrawal, R. Kumar, K. Saenko, and B. A. Plummer, ”A dataset for interactive vision-language navigation with unknown command feasibility,” in European Conference on Computer Vision. Springer, 2022, pp. 312-328.

A. Burns, D. Arsan, S. Agrawal, R. Kumar, K. Saenko, 和 B. A. Plummer, ” 一个用于交互式视觉语言导航且命令可行性未知的数据集,” 载于欧洲计算机视觉会议。Springer, 2022, 页 312-328。

[181] L. Sun, X. Chen, L. Chen, T. Dai, Z. Zhu, and K. Yu, ”Meta-gui: Towards multi-modal conversational agents on mobile gui,” arXiv preprint arXiv:2205.11029, 2022.

L. Sun, X. Chen, L. Chen, T. Dai, Z. Zhu, 和 K. Yu, ”Meta-gui: 迈向移动 GUI 上的多模态对话代理,” arXiv 预印本 arXiv:2205.11029, 2022 年。

[182] S. Yao, H. Chen, J. Yang, and K. Narasimhan, ”Web-shop: Towards scalable real-world web interaction with grounded language agents,” Advances in Neural Information Processing Systems, vol. 35, pp. 20744- 20757, 2022.

S. Yao, H. Chen, J. Yang, 和 K. Narasimhan, ”Web-shop: 迈向具备基础语言能力的可扩展真实网页交互代理,” 神经信息处理系统进展, 第 35 卷, 页 20744-20757, 2022 年。

[183] Y. Deng, X. Zhang, W. Zhang, Y. Yuan, S.-K. Ng, and T.-S. Chua, ”On the multi-turn instruction following for conversational web agents,” arXiv preprint arXiv:2402.15057, 2024.

邓毅，张晓，张伟，袁毅，吴世康，蔡天石，“关于对话式网页代理的多轮指令跟随”，arXiv 预印本 arXiv:2402.15057，2024 年。

[184] Q. Chen, D. Pitawela, C. Zhao, G. Zhou, H.-T. Chen, and Q. Wu, ”Webvln: Vision-and-language navigation on websites,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, no. 2, 2024, pp. 1165-1173.

陈强，皮塔韦拉，赵晨，周刚，陈海涛，吴强，“Webvln: 基于视觉与语言的网页导航”，发表于《美国人工智能协会会议论文集》，第 38 卷，第 2 期，2024 年，第 1165-1173 页。

[185] Z. Zhang, S. Tian, L. Chen, and Z. Liu, ”Mmina: Benchmarking multihop multimodal internet agents,” arXiv preprint arXiv:2404.09992, 2024.

张志, 田松, 陈磊, 刘志, “Mmina: 多跳多模态互联网代理基准测试”, arXiv 预印本 arXiv:2404.09992, 2024 年。

[186] J. Liu, Y. Song, B. Y. Lin, W. Lam, G. Neubig, Y. Li, and X. Yue, ”Visualwebbench: How far have multimodal llms evolved in web page understanding and grounding?” arXiv preprint arXiv:2404.05955, 2024.

刘杰，宋阳，林博洋，林伟，纽比格，李阳，岳翔，“Visualwebbench: 多模态大语言模型在网页理解与定位方面的发展进展”，arXiv 预印本 arXiv:2404.05955，2024 年。

[187] T. Xie, D. Zhang, J. Chen, X. Li, S. Zhao, R. Cao, T. J. Hua, Z. Cheng, D. Shin, F. Lei et al., ”Os-world: Benchmarking multimodal agents for open-ended tasks in real computer environments,” arXiv preprint arXiv:2404.07972, 2024.

谢涛，张东，陈军，李翔，赵帅，曹锐，华天杰，程志，申东，雷飞等，“Os-world: 真实计算机环境中多模态代理的开放式任务基准”，arXiv 预印本 arXiv:2404.07972，2024 年。

[188] D. Zhang, H. Xu, Z. Zhao, L. Chen, R. Cao, and K. Yu, ”Mobile-env: an evaluation platform and benchmark for llm-gui interaction,” arXiv preprint arXiv:2305.08144, 2023.

张东，徐浩，赵志，陈磊，曹锐，余凯，“Mobile-env: 大语言模型与图形用户界面交互的评测平台与基准”，arXiv 预印本 arXiv:2305.08144，2023 年。

[189] K. Q. Lin, L. Li, D. Gao, Q. WU, M. Yan, Z. Yang, L. Wang, and M. Z. Shou, ”Videogui: A benchmark for gui automation from instructional videos,” arXiv preprint arXiv:2406.10227, 2024.

林启强，李磊，高东，吴强，闫明，杨志，王磊，寿明哲，“Videogui: 基于教学视频的图形用户界面自动化基准”，arXiv 预印本 arXiv:2406.10227，2024 年。

[190] W. Chen, J. Cui, J. Hu, Y. Qin, J. Fang, Y. Zhao, C. Wang, J. Liu, G. Chen, Y. Huo et al., ”Guicourse: From general vision language models to versatile gui agents,” arXiv preprint arXiv:2406.11317, 2024.

陈伟，崔军，胡军，秦阳，方军，赵阳，王超，刘军，陈刚，霍阳等，“Guicourse: 从通用视觉语言模型到多功能图形用户界面代理”，arXiv 预印本 arXiv:2406.11317，2024 年。

[191] D. Chen, Y. Huang, S. Wu, J. Tang, L. Chen, Y. Bai, Z. He, C. Wang, H. Zhou, Y. Li et al., ”Gui-world: A dataset for gui-oriented multimodal llm-based agents,” arXiv preprint arXiv:2406.10819, 2024.

陈东，黄勇，吴帅，唐军，陈磊，白洋，何志，王超，周浩，李阳等，“Gui-world: 面向图形用户界面的多模态大语言模型代理数据集”，arXiv 预印本 arXiv:2406.10819，2024 年。

[192] Q. Lu, W. Shao, Z. Liu, F. Meng, B. Li, B. Chen, S. Huang, K. Zhang, Y. Qiao, and P. Luo, ”Gui odyssey: A comprehensive dataset for cross-app gui navigation on mobile devices,” arXiv preprint arXiv:2406.08451, 2024.

陆强，邵伟，刘志，孟飞，李斌，陈斌，黄松，张凯，乔阳，罗鹏，“Gui odyssey: 移动设备跨应用图形用户界面导航的综合数据集”，arXiv 预印本 arXiv:2406.08451，2024 年。

[193] Y. Pan, D. Kong, S. Zhou, C. Cui, Y. Leng, B. Jiang, H. Liu, Y. Shang, S. Zhou, T. Wu et al., ”Webcanvas: Benchmarking web agents in online environments,” arXiv preprint arXiv:2406.12373, 2024.

潘勇，孔东，周松，崔超，冷毅，姜斌，刘浩，尚阳，周松，吴涛等，“Webcanvas: 在线环境中网页代理的基准测试”，arXiv 预印本 arXiv:2406.12373，2024 年。

[194] R. Cao, F. Lei, H. Wu, J. Chen, Y. Fu, H. Gao, X. Xiong, H. Zhang, Y. Mao, W. Hu et al., ”Spider2-v: How far are multimodal agents from automating data science and engineering workflows?” arXiv preprint arXiv:2407.10956, 2024.

曹锐，雷飞，吴浩，陈军，傅阳，高浩，熊翔，张浩，毛阳，胡伟等，“Spider2-v: 多模态代理距离自动化数据科学与工程工作流还有多远?”，arXiv 预印本 arXiv:2407.10956，2024 年。

[195] R. Kapoor, Y. P. Butala, M. Russak, J. Y. Koh, K. Kam-ble, W. Alshikh, and R. Salakhutdinov, ”Omniact: A dataset and benchmark for enabling multimodal generalist autonomous agents for desktop and web,” arXiv preprint arXiv:2402.17553, 2024.

卡普尔，布塔拉，鲁萨克，柯俊英，甘布尔，阿尔希赫，萨拉胡丁诺夫，“Omniact: 支持桌面与网页多模态通用自主代理的数据集与基准”，arXiv 预印本 arXiv:2402.17553，2024 年。

[196] Y. Chai, S. Huang, Y. Niu, H. Xiao, L. Liu, D. Zhang, P. Gao, S. Ren, and H. Li, ”Amex: Android multi-annotation expo dataset for mobile gui agents,” arXiv preprint arXiv:2407.17490, 2024.

Y. Chai, S. Huang, Y. Niu, H. Xiao, L. Liu, D. Zhang, P. Gao, S. Ren, 和 H. Li, “Amex: 面向移动 GUI 代理的 Android 多注释展示数据集，”arXiv 预印本 arXiv:2407.17490, 2024。

[197] L. Zhang, S. Wang, X. Jia, Z. Zheng, Y. Yan, L. Gao, Y. Li, and M. Xu, ”Llamatouch: A faithful and scalable testbed for mobile ui automation task evaluation,” arXiv preprint arXiv:2404.16054, 2024.

L. Zhang, S. Wang, X. Jia, Z. Zheng, Y. Yan, L. Gao, Y. Li, 和 M. Xu, “Llamatouch: 一个忠实且可扩展的移动 UI 自动化任务评测平台,” arXiv 预印本 arXiv:2404.16054, 2024。

[198] M. Xing, R. Zhang, H. Xue, Q. Chen, F. Yang, and Z. Xiao, ”Understanding the weakness of large language model agents within a complex android environment,” in Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2024, pp. 6061-6072.

M. Xing, R. Zhang, H. Xue, Q. Chen, F. Yang, 和 Z. Xiao, “理解复杂 Android 环境下大型语言模型代理的弱点,” 发表于第 30 届 ACM SIGKDD 知识发现与数据挖掘大会论文集，2024，页 6061-6072。

[199] R. Bonatti, D. Zhao, F. Bonacci, D. Dupont, S. Abdali, Y. Li, Y. Lu, J. Wagle, K. Koishida, A. Bucker et al., ”Windows agent arena: Evaluating multi-modal os agents at scale,” arXiv preprint arXiv:2409.08264, 2024.

R. Bonatti, D. Zhao, F. Bonacci, D. Dupont, S. Abdali, Y. Li, Y. Lu, J. Wagle, K. Koishida, A. Bucker 等,“Windows 代理竞技场: 大规模多模态操作系统代理评测,” arXiv 预印本 arXiv:2409.08264, 2024。

[200] X. H. Lù, Z. Kasner, and S. Reddy, ”Weblinx: Real-world website navigation with multi-turn dialogue,” arXiv preprint arXiv:2402.05930, 2024.

X. H. Lù, Z. Kasner, 和 S. Reddy, “Weblinx: 基于多轮对话的真实网站导航, ” arXiv 预印本 arXiv:2402.05930, 2024。

[201] L. Zheng, Z. Huang, Z. Xue, X. Wang, B. An, and S. Yan, ”Agentstudio: A toolkit for building general virtual agents,” arXiv preprint arXiv:2403.17918, 2024.

L. Zheng, Z. Huang, Z. Xue, X. Wang, B. An, 和 S. Yan, “Agentstudio: 构建通用虚拟代理的工具包,” arXiv 预印本 arXiv:2403.17918, 2024。

[202] J. Lee, T. Min, M. An, D. Hahm, H. Lee, C. Kim, and K. Lee, ”Benchmarking mobile device control agents across diverse configurations,” arXiv preprint arXiv:2404.16660, 2024.

J. Lee, T. Min, M. An, D. Hahm, H. Lee, C. Kim, 和 K. Lee,“跨多配置的移动设备控制代理基准测试,” arXiv 预印本 arXiv:2404.16660, 2024。

[203] T. Xu, L. Chen, D.-J. Wu, Y. Chen, Z. Zhang, X. Yao, Z. Xie, Y. Chen, S. Liu, B. Qian et al., ”Crab: Cross-environment agent benchmark for multimodal language model agents,” arXiv preprint arXiv:2407.01511, 2024.

T. Xu, L. Chen, D.-J. Wu, Y. Chen, Z. Zhang, X. Yao, Z. Xie, Y. Chen, S. Liu, B. Qian 等，“Crab: 面向多模态语言模型代理的跨环境代理基准,” arXiv 预印本 arXiv:2407.01511, 2024。

[204] Y. Fan, L. Ding, C.-C. Kuo, S. Jiang, Y. Zhao, X. Guan, J. Yang, Y. Zhang, and X. E. Wang, ”Read anywhere pointed: Layout-aware gui screen reading with tree-of-lens grounding,” arXiv preprint arXiv:2406.19263, 2024.

Y. Fan, L. Ding, C.-C. Kuo, S. Jiang, Y. Zhao, X. Guan, J. Yang, Y. Zhang, 和 X. E. Wang,“Read anywhere pointed: 基于树状透镜定位的布局感知 GUI 屏幕阅读,”arXiv 预印本 arXiv:2406.19263, 2024。

[205] J. Chen, D. Yuen, B. Xie, Y. Yang, G. Chen, Z. Wu, L. Yixing, X. Zhou, W. Liu, S. Wang, K. Zhou, R. Shao, L. Nie, Y. Wang, J. Hao, J. Wang, and K. Shao, ”Spa-bench: A comprehensive benchmark for smartphone agent evaluation,” 2025. [Online]. Available: https://arxiv.org/abs/2410.15164

J. Chen, D. Yuen, B. Xie, Y. Yang, G. Chen, Z. Wu, L. Yixing, X. Zhou, W. Liu, S. Wang, K. Zhou, R. Shao, L. Nie, Y. Wang, J. Hao, J. Wang, 和 K. Shao,“Spa-bench: 智能手机代理评测的综合基准,”2025。[在线]。可访问:https://arxiv.org/abs/2410.15164

[206] W. Li, W. Bishop, A. Li, C. Rawles, F. Campbell-Ajala, D. Tyamagundlu, and O. Riva, ”On the effects of data scale on computer control agents,” arXiv preprint arXiv:2406.03679, 2024.

W. Li, W. Bishop, A. Li, C. Rawles, F. Campbell-Ajala, D. Tyamagundlu, 和 O. Riva, “数据规模对计算机控制代理影响的研究,”arXiv 预印本 arXiv:2406.03679, 2024。

[207] J. Liu, T. Ou, Y. Song, Y. Qu, W. Lam, C. Xiong, W. Chen, G. Neubig, and X. Yue, ”Harnessing web-page uis for text-rich visual understanding,” arXiv preprint arXiv:2410.13824, 2024.

J. Liu, T. Ou, Y. Song, Y. Qu, W. Lam, C. Xiong, W. Chen, G. Neubig, 和 X. Yue, “利用网页 UI 进行文本丰富的视觉理解,”arXiv 预印本 arXiv:2410.13824, 2024。

[208] K. Zhao, J. Song, L. Sha, H. Shen, Z. Chen, T. Zhao, X. Liang, and J. Yin, ”Gui testing arena: A unified benchmark for advancing autonomous gui testing agent,” 2024. [Online]. Available: https: //arxiv.org/abs/2412.18426

K. Zhao, J. Song, L. Sha, H. Shen, Z. Chen, T. Zhao, X. Liang, 和 J. Yin, “Gui testing arena: 推进自主 GUI 测试代理的统一基准,”2024。[在线]。可访问:https://arxiv.org/abs/2412.18426

[209] Y. Chai, H. Li, J. Zhang, L. Liu, G. Liu, G. Wang, S. Ren, S. Huang, and H. Li, ”A3: Android agent arena for mobile gui agents,” 2025. [Online]. Available: https://arxiv.org/abs/2501.01149

Y. Chai, H. Li, J. Zhang, L. Liu, G. Liu, G. Wang, S. Ren, S. Huang, 和 H. Li, “A3: 面向移动 GUI 代理的 Android 代理竞技场,”2025 年。[在线]. 可获取:https://arxiv.org/abs/2501.01149

[210] J. Wu, W. Yin, Y. Jiang, Z. Wang, Z. Xi, R. Fang, L. Zhang, Y. He, D. Zhou, P. Xie, and F. Huang, ”Webwalker: Benchmarking llms in web traversal,” 2025. [Online]. Available: https://arxiv.org/abs/2501.07572

J. Wu, W. Yin, Y. Jiang, Z. Wang, Z. Xi, R. Fang, L. Zhang, Y. He, D. Zhou, P. Xie, 和 F. Huang, “Webwalker:Web 遍历中大型语言模型 (LLMs) 的基准测试, ”2025 年。[在线]. 可获取:https://arxiv.org/abs/2501.07572

[211] H. H. Zhao, D. Gao, and M. Z. Shou, ”Worldgui: Dynamic testing for comprehensive desktop gui automation,” 2025. [Online]. Available: https://arxiv.org/abs/2502.08047

H. H. Zhao, D. Gao, 和 M. Z. Shou, “Worldgui: 面向综合桌面 GUI 自动化的动态测试,”2025 年。[在线]. 可获取:https://arxiv.org/abs/2502.08047

[212] S. Nayak, X. Jian, K. Q. Lin, J. A. Rodriguez, M. Kalsi, R. Awal, N. Chapados, M. T. Özsu, A. Agrawal, D. Vazquez, C. Pal, P. Taslakian, S. Gella, and S. Rajeswar, ”Ui-vision: A desktop-centric gui benchmark for visual perception and interaction,” 2025. [Online]. Available: https://arxiv.org/abs/2503.15661

S. Nayak, X. Jian, K. Q. Lin, J. A. Rodriguez, M. Kalsi, R. Awal, N. Chapados, M. T. Özsu, A. Agrawal, D. Vazquez, C. Pal, P. Taslakian, S. Gella, 和 S. Rajeswar, “Ui-vision: 面向视觉感知与交互的桌面中心 GUI 基准,”2025 年。[在线]. 可获取:https://arxiv.org/abs/2503.15661