Large Language Model-Brained GUI Agents: A Survey

大语言模型驱动的图形用户界面代理：综述

Chaoyun Zhang, Shilin He, Jiaxu Qian, Bowen Li, Liqun Li, Si Qin, Yu Kang, Minghua Ma, Guyue Liu, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, Qi Zhang
张超云、何世林、钱佳旭、李博文、李立群、秦思、康宇、马明华、刘谷雨、林庆伟、萨拉万 · 拉杰莫汉、张冬梅、张琦

Abstract-Graphical User Interfaces (GUIs) have long been central to human-computer interaction, providing an intuitive and visually-driven way to access and interact with digital systems. Traditionally, automating GUI interactions relied on script-based or rule-based approaches, which, while effective for fixed workflows, lacked the flexibility and adaptability required for dynamic, real-world applications. The advent of Large Language Models (LLMs), particularly multimodal models, has ushered in a new era of GUI automation. They have demonstrated exceptional capabilities in natural language understanding, code generation, task generalization, and visual processing. This has paved the way for a new generation of "LLM-brained" GUI agents capable of interpreting complex GUI elements and autonomously executing actions based on natural language instructions. These agents represent a paradigm shift, enabling users to perform intricate, multi-step tasks through simple conversational commands. Their applications span across web navigation, mobile app interactions, and desktop automation, offering a transformative user experience that revolutionizes how individuals interact with software. This emerging field is rapidly advancing, with significant progress in both research and industry.
摘要 — 图形用户界面（Graphical User Interfaces，GUIs）长期以来一直是人机交互的核心，为访问和与数字系统交互提供了一种直观且可视化驱动的方式。传统上，图形用户界面交互的自动化依赖于基于脚本或基于规则的方法，虽然这些方法对于固定工作流程有效，但缺乏动态现实应用所需的灵活性和适应性。大语言模型（Large Language Models，LLMs），尤其是多模态模型的出现，开启了图形用户界面自动化的新纪元。它们在自然语言理解、代码生成、任务泛化和视觉处理方面展现出卓越的能力。这为新一代“由大语言模型驱动”的图形用户界面代理铺平了道路，这些代理能够解释复杂的图形用户界面元素，并根据自然语言指令自主执行操作。这些代理代表了一种范式转变，使用户能够通过简单的对话式命令执行复杂的多步骤任务。它们的应用涵盖网页导航、移动应用交互和桌面自动化，提供了一种变革性的用户体验，彻底改变了个人与软件交互的方式。这个新兴领域正在迅速发展，在研究和行业领域都取得了重大进展。

To provide a structured understanding of this trend, this paper presents a comprehensive survey of LLM-brained GUI agents, exploring their historical evolution, core components, and advanced techniques. We address critical research questions such as existing GUI agent frameworks, the collection and utilization of data for training specialized GUI agents, the development of large action models tailored for GUI tasks, and the evaluation metrics and benchmarks necessary to assess their effectiveness. Additionally, we examine emerging applications powered by these agents. Through a detailed analysis, this survey identifies key research gaps and outlines a roadmap for future advancements in the field. By consolidating foundational knowledge and state-of-the-art developments, this work aims to guide both researchers and practitioners in overcoming challenges and unlocking the full potential of LLM-brained GUI agents. We anticipate that this survey will serve both as a practical cookbook for constructing LLM-powered GUI agents, and as a definitive reference for advancing research in this rapidly evolving domain.
为了对这一趋势进行系统性的理解，本文对由大语言模型驱动的图形用户界面代理进行了全面综述，探讨了它们的历史演变、核心组件和先进技术。我们探讨了关键的研究问题，例如现有的图形用户界面代理框架、用于训练专门图形用户界面代理的数据收集和利用、为图形用户界面任务量身定制的大型动作模型的开发，以及评估其有效性所需的评估指标和基准。此外，我们还研究了由这些代理驱动的新兴应用。通过详细分析，本综述确定了关键的研究空白，并为该领域未来的发展勾勒了路线图。通过整合基础知识和最新发展，这项工作旨在指导研究人员和从业者克服挑战，充分发挥由大语言模型驱动的图形用户界面代理的潜力。我们预计，本综述既可以作为构建由大语言模型驱动的图形用户界面代理的实用指南，也可以作为该快速发展领域推进研究的权威参考。

The collection of papers reviewed in this survey will be hosted and regularly updated on the GitHub repository: https://github.com/vyokky/LLM-Brained-GUI-Agents-Survey Additionally, a searchable webpage is available at https://aka.ms/gui-agent for easier access and exploration.
本综述中所引用论文的集合将托管在 GitHub 仓库中并定期更新：https://github.com/vyokky/LLM-Brained-GUI-Agents-Survey 此外，还可以通过可搜索的网页 https://aka.ms/gui-agent 更方便地访问和探索。

Index Terms-Large Language Model, Graphical User Interface, AI Agent, Automation, Human-Computer Interaction
关键词 — 大语言模型、图形用户界面、人工智能代理、自动化、人机交互

1 1 INTRODUCTION

2 1 引言

Graphical User Interfaces (GUIs) have been a cornerstone of human-computer interaction, fundamentally transforming how users navigate and operate within digital systems [1]. Designed to make computing more intuitive and accessible, GUIs replaced command-line interfaces (CLIs) [2] with visually driven, user-friendly environments. Through the use of icons, buttons, windows, and menus, GUIs empowered a broader range of users to interact with computers using simple actions such as clicks, typing, and gestures. This shift democratized access to computing, allowing even non-technical users to effectively engage with complex systems. However, GUIs often sacrifice efficiency for

usability, particularly in workflows requiring repetitive or multi-step interactions, where CLIs can remain more streamlined [3].
图形用户界面（Graphical User Interfaces，GUIs）一直是人机交互的基石，从根本上改变了用户在数字系统中的导航和操作方式 [1]。图形用户界面旨在使计算更加直观和易于使用，它用可视化驱动的、用户友好的环境取代了命令行界面（Command - Line Interfaces，CLIs）[2]。通过使用图标、按钮、窗口和菜单，图形用户界面使更广泛的用户能够通过点击、打字和手势等简单操作与计算机进行交互。这一转变使计算的使用更加普及，即使是非技术用户也能有效地与复杂系统进行交互。然而，图形用户界面通常为了可用性而牺牲了效率，特别是在需要重复或多步骤交互的工作流程中，命令行界面可能仍然更加高效 [3]。

While GUIs revolutionized usability, their design, primarily tailored for human visual interaction, poses significant challenges for automation. The diversity, dynamism, and platform-specific nature of GUI layouts make it difficult to develop flexible and intelligent automation tools capable of adapting to various environments. Early efforts to automate GUI interactions predominantly relied on script-based or rule-based methods [4], [5]. Although effective for predefined workflows, these methods were inherently narrow in scope, focusing primarily on tasks such as software testing and robotic process automation (RPA) [6]. Their rigidity required frequent manual updates to accommodate new tasks, changes in GUI layouts, or evolving workflows, limiting their scalability and versatility. Moreover, these approaches lacked the sophistication needed to support dynamic, human-like interactions, thereby constraining their applicability in complex or unpredictable scenarios.
虽然图形用户界面彻底改变了可用性，但其主要为人类视觉交互设计的特点给自动化带来了重大挑战。图形用户界面布局的多样性、动态性和特定平台性质使得开发能够适应各种环境的灵活智能自动化工具变得困难。早期实现图形用户界面交互自动化的努力主要依赖于基于脚本或基于规则的方法 [4]、[5]。虽然这些方法对于预定义的工作流程有效，但本质上适用范围狭窄，主要侧重于软件测试和机器人流程自动化（Robotic Process Automation，RPA）等任务 [6]。它们的僵化性要求频繁手动更新以适应新任务、图形用户界面布局的变化或不断演变的工作流程，限制了它们的可扩展性和通用性。此外，这些方法缺乏支持动态、类人交互所需的复杂性，从而限制了它们在复杂或不可预测场景中的适用性。

The rise of Large Language Models (LLMs) [8, 9], especially those augmented with multimodal capabilities [10], has emerged as a game changer for GUI automation, redefining the the way agents interact with graphical user interfaces. Beginning with models like ChatGPT [11], LLMs have demonstrated extraordinary proficiency in natural language understanding, code generation, and generalization across diverse tasks [8], 12-14. The integration of visual language models (VLMs) has further extended these capabilities, enabling these models to process visual data, such as the intricate layouts of GUIs [15]. This evolution bridges the gap between linguistic and visual comprehension, empowering intelligent agents to interact with GUIs in a more human-like and adaptive manner. By leveraging these advancements, LLMs and VLMs offer transformative potential, enabling agents to navigate complex digital environments, execute tasks dynamically, and revolutionize the field of GUI automation.
大语言模型（LLMs）[8, 9]的兴起，尤其是那些具备多模态能力的模型[10]，已成为图形用户界面（GUI）自动化领域的变革性力量，重新定义了智能体与图形用户界面交互的方式。从ChatGPT[11]等模型开始，大语言模型在自然语言理解、代码生成以及跨多种任务的泛化能力方面展现出了非凡的实力[8, 12 - 14]。视觉语言模型（VLMs）的融入进一步拓展了这些能力，使这些模型能够处理视觉数据，例如图形用户界面的复杂布局[15]。这一发展弥合了语言理解和视觉理解之间的差距，使智能体能够以更接近人类且自适应的方式与图形用户界面进行交互。借助这些进展，大语言模型和视觉语言模型具有变革性潜力，使智能体能够在复杂的数字环境中导航、动态执行任务，并彻底改变图形用户界面自动化领域。

Version: v8 (major update on May 2, 2025)
版本：v8（2025年5月2日重大更新）

Chaoyun Zhang, Shilin He, Jiaxu Qian, Liqun Li, Si Qin, Yu Kang, Minghua Ma, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang and Qi Zhang are with Microsoft. e-mail: {chaoyun.zhang, shilin.he, v-jiaxuqian, liqun.li, si.gin, yu.kang, minghuama, qlin, saravan.rajmohan, dongmeiz, zhang.qi}@microsoft.com.
张朝云、何仕林、钱佳旭、李立群、秦思、康宇、马明华、林庆伟、萨拉万·拉杰莫汉、张冬梅和张琦就职于微软。电子邮件：{chaoyun.zhang, shilin.he, v - jiaxuqian, liqun.li, si.gin, yu.kang, minghuama, qlin, saravan.rajmohan, dongmeiz, zhang.qi}@microsoft.com。

Bowen Li is with Shanghai Artificial Intelligence Laboratory, China. e-mail: libowen.ne@gmail.com.
李博文就职于中国上海人工智能实验室。电子邮件：libowen.ne@gmail.com。

Guyue Liu is with Peking University, China. e-mail: guyue.liu@gmail.com.For any inquiries or discussions, please contact Chaoyun Zhang and Shilin He.
刘谷雨就职于中国北京大学。电子邮件：guyue.liu@gmail.com。如有任何咨询或讨论，请联系张朝云和何仕林。

1. By LLMs, we refer to the general concept of foundation models capable of accepting various input modalities (e.g., visual language models (VLMs), multimodal LLMs (MLLMs)) while producing output exclusively in textual sequences [7].
2. 我们所说的大语言模型，是指能够接受各种输入模态（例如视觉语言模型（VLMs）、多模态大语言模型（MLLMs）），同时仅以文本序列形式输出的基础模型的通用概念[7]。

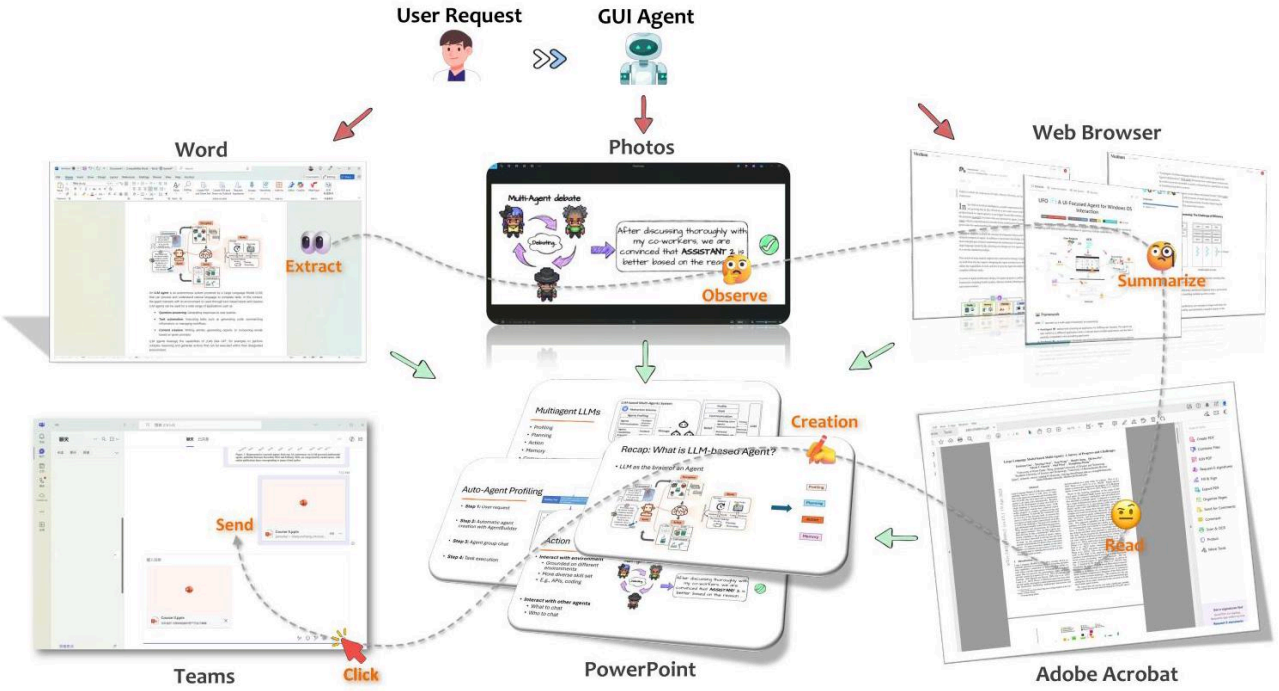

Fig. 1: Illustration of the high-level concept of an LLM-powered GUI agent. The agent receives a user's natural language request and orchestrates actions seamlessly across multiple applications. It extracts information from Word documents, observes content in Photos, summarizes web pages in the browser, reads PDFs in Adobe Acrobat, and creates slides in PowerPoint before sending them through Teams.
图1：由大语言模型驱动的图形用户界面智能体的高层概念示意图。该智能体接收用户的自然语言请求，并在多个应用程序之间无缝协调操作。它从Word文档中提取信息，查看照片中的内容，总结浏览器中的网页内容，在Adobe Acrobat中阅读PDF文件，并在PowerPoint中创建幻灯片，然后通过Teams发送。

2.1 1.1 Motivation for LLM-Brained GUI agents

2.2 1.1 基于大语言模型的图形用户界面智能体的动机

With an LLM serving as its "brain", LLM-powered GUI automation introduces a new class of intelligent agents capable of interpreting a user's natural language requests, analyzing GUI screens and their elements, and autonomously executing appropriate actions. Importantly, these capabilities are achieved without reliance on complex, platform-specific scripts or predefined workflows. These agents, referred to as "LLM-brained GUI agents", can be formally defined as:
以大语言模型作为“大脑”的图形用户界面自动化引入了一类新型智能体，它们能够解读用户的自然语言请求、分析图形用户界面屏幕及其元素，并自主执行适当的操作。重要的是，实现这些能力无需依赖复杂的、特定平台的脚本或预定义的工作流程。这些智能体，即“基于大语言模型的图形用户界面智能体”，可以正式定义为：

Intelligent agents that operate within GUI environments, leveraging LLMs as their core inference and cognitive engine to generate, plan, and execute actions in a flexible and adaptive manner.
在图形用户界面环境中运行的智能体，利用大语言模型作为其核心推理和认知引擎，以灵活和自适应的方式生成、规划和执行操作。

This paradigm represents a transformative leap in GUI automation, fostering dynamic, human-like interactions across diverse platforms. It enables the creation of intelligent, adaptive systems that can reason, make decisions in real-time, and respond flexibly to evolving tasks and environments. We illustrate this high-level concept in Figure 1
这种范式代表了图形用户界面自动化领域的一次变革性飞跃，促进了跨不同平台的动态、类人交互。它使得能够创建智能、自适应的系统，这些系统可以进行推理、实时决策，并灵活应对不断变化的任务和环境。我们在图1中展示了这一高层概念。

Traditional GUI automation are often limited by predefined rules or narrowly focused on specific tasks, constraining their ability to adapt to dynamic environments and diverse applications. In contrast, LLM-powered GUI agents bring a paradigm shift by integrating natural language understanding, visual recognition, and decision-making into a unified framework. This enables them to generalize across a wide range of use cases, transforming task automation and significantly enhancing the intuitiveness and efficiency of human-computer interaction. Moreover, unlike the emerging trend of pure Application Programming Interface (API)-based agents—which depend on APIs that may not always be exposed or accessible-GUI agents leverage the universal nature of graphical interfaces. GUIs offer a general

mechanism to control most software applications, enabling agents to operate in a nonintrusive manner without requiring internal API access. This capability not only broadens the applicability of GUI agents but also empowers external developers to build advanced functionality on top of existing software across diverse platforms and ecosystems. Together, these innovations position GUI agents as a versatile and transformative technology for the future of intelligent automation.
传统的图形用户界面自动化通常受限于预定义规则，或者仅专注于特定任务，这限制了它们适应动态环境和多样化应用的能力。相比之下，基于大语言模型的图形用户界面智能体通过将自然语言理解、视觉识别和决策制定集成到一个统一的框架中，带来了范式转变。这使它们能够在广泛的用例中进行泛化，改变任务自动化方式，并显著提高人机交互的直观性和效率。此外，与新兴的纯基于应用程序编程接口（API）的智能体趋势不同——这类智能体依赖的API可能并不总是公开或可访问的——图形用户界面智能体利用了图形界面的通用性。图形用户界面提供了一种通用机制来控制大多数软件应用程序，使智能体能够以非侵入性的方式运行，而无需访问内部API。这种能力不仅拓宽了图形用户界面智能体的适用性，还使外部开发人员能够在不同平台和生态系统的现有软件基础上构建高级功能。总之，这些创新使图形用户界面智能体成为未来智能自动化领域一种通用且具有变革性的技术。

This new paradigm enables users to control general software systems with conversational commands [16]. By reducing the cognitive load of multi-step GUI operations, LLM-powered agents make complex systems accessible to non-technical users and streamline workflows across diverse domains. Notable examples include SeeAct [17] for web navigation, AppAgent [18] for mobile interactions, and UFO [19] for Windows OS applications. These agents resemble a "virtual assistant" [20] akin to J.A.R.V.I.S. from Iron Man-an intuitive, adaptive system capable of understanding user goals and autonomously performing actions across applications. The futuristic concept of an AI-powered operating system that executes cross-application tasks with fluidity and precision is rapidly becoming a reality [21], [22].
这种新范式使用户能够通过对话式命令控制通用软件系统 [16]。通过减轻多步骤图形用户界面（GUI）操作的认知负担，由大语言模型（LLM）驱动的智能体使非技术用户也能使用复杂系统，并简化了不同领域的工作流程。值得注意的例子包括用于网页导航的SeeAct [17]、用于移动交互的AppAgent [18] 以及用于Windows操作系统应用程序的UFO [19]。这些智能体类似于《钢铁侠》中的贾维斯（J.A.R.V.I.S.）那样的“虚拟助手” [20] —— 一种直观、自适应的系统，能够理解用户目标并自动跨应用程序执行操作。由人工智能驱动的操作系统能够流畅、精确地执行跨应用程序任务的未来概念正在迅速成为现实 [21]、[22]。

Real-world applications of LLM-powered GUI agents are already emerging. For example, Microsoft Power Automate utilizes LLMs to streamline low-code/no-code automation [2] . allowing users to design workflows across Microsoft applications with minimal technical expertise. Integrated AI assistants in productivity software,like Microsoft Copilot [3] are bridging the gap between natural language instructions and operations on application. Additionally, LLM-powered agents show promise for enhancing accessibility [23], potentially allowing visually impaired users to navigate GUIs more effectively by converting natural language commands into executable steps. These developments underscore the timeliness and transformative potential of LLM-powered GUI agents across diverse applications.
由大语言模型驱动的图形用户界面智能体在现实世界中的应用已经开始出现。例如，微软Power Automate利用大语言模型简化低代码/无代码自动化 [2]，使用户只需具备最少的技术知识就能设计跨微软应用程序的工作流程。生产力软件中的集成式人工智能助手，如微软Copilot [3]，正在弥合自然语言指令与应用程序操作之间的差距。此外，由大语言模型驱动的智能体在提高可访问性方面显示出潜力 [23]，有可能通过将自然语言命令转换为可执行步骤，让视障用户更有效地浏览图形用户界面。这些发展凸显了由大语言模型驱动的图形用户界面智能体在不同应用中的及时性和变革潜力。

The convergence of LLMs and GUI automation addresses longstanding challenges in human-computer interaction and introduces new opportunities for intelligent GUI control [24]. This integration has catalyzed a surge in research activity, spanning application frameworks [19], data collection [25], model optimization [15], and evaluation benchmarks [26]. Despite these advancements, key challenges and limitations persist, and many foundational questions remain unexplored. However, a systematic review of this rapidly evolving area is notably absent, leaving a critical gap in understanding.
大语言模型与图形用户界面自动化的融合解决了人机交互中长期存在的挑战，并为智能图形用户界面控制带来了新机遇 [24]。这种融合促使研究活动激增，涵盖了应用框架 [19]、数据收集 [25]、模型优化 [15] 和评估基准 [26] 等方面。尽管取得了这些进展，但关键挑战和局限性仍然存在，许多基础性问题仍未得到探索。然而，对这一快速发展领域的系统综述明显缺失，这在理解上造成了重大空白。

2.3 1.2 Scope of the Survey

2.4 1.2 调查范围

To address this gap, this paper provides a pioneering, comprehensive survey of LLM-brained GUI agents. We cover the historical evolution of GUI agents, provide a step-by-step guide to building these agents, summarize essential and advanced techniques, review notable tools and research related to frameworks, data and models, showcase representative applications, and outline future directions. Specifically, this survey aims to answer the following research questions (RQs):
为了填补这一空白，本文对由大语言模型驱动的图形用户界面智能体进行了开创性的全面调查。我们涵盖了图形用户界面智能体的历史演变，提供了构建这些智能体的分步指南，总结了基本和高级技术，回顾了与框架、数据和模型相关的重要工具和研究，展示了代表性应用，并概述了未来方向。具体而言，本次调查旨在回答以下研究问题（RQs）：

1. RQ1: What is the historical development trajectory of LLM-powered GUI agents? (Section 4)
2. 研究问题1（RQ1）：由大语言模型驱动的图形用户界面智能体的历史发展轨迹是怎样的？（第4节）

2. RQ2: What are the essential components and advanced technologies that form the foundation of LLM-brained GUI agents? (Section 5)
3. 研究问题2（RQ2）：构成由大语言模型驱动的图形用户界面智能体基础的基本组件和先进技术有哪些？（第5节）

3. RQ3: What are the principal frameworks for LLM GUI agents, and what are their defining characteristics? (Section 6)
4. 研究问题3（RQ3）：大语言模型图形用户界面智能体的主要框架有哪些，它们的定义特征是什么？（第6节）

4. RQ4: What are the existing datasets, and how can comprehensive datasets be collected to train optimized LLMs for GUI agents? (Section 7)
5. 研究问题4（RQ4）：现有的数据集有哪些，如何收集全面的数据集来训练适用于图形用户界面智能体的优化大语言模型？（第7节）

5. RQ5: How can the collected data be used to train purpose-built Large Action Models (LAMs) for GUI agents, and what are the current leading models in the field? (Section 8)
6. 研究问题5（RQ5）：如何利用收集到的数据为图形用户界面智能体训练专用的大型动作模型（LAMs），该领域目前的领先模型有哪些？（第8节）

6. RQ6: What metrics and benchmarks are used to evaluate the capability and performance of GUI agents? (Section 9)
7. 研究问题6（RQ6）：用于评估图形用户界面智能体能力和性能的指标和基准有哪些？（第9节）

7. RQ7: What are the most significant real-world applications of LLM-powered GUI agents, and how have they been adapted for practical use? (Section 10)
8. 研究问题7（RQ7）：由大语言模型驱动的图形用户界面智能体在现实世界中最重要的应用有哪些，它们是如何适应实际应用的？（第10节）

8. RQ8: What are the major challenges, limitations, and future research directions for developing robust and intelligent GUI agents? (Section 11)
9. 研究问题8（RQ8）：开发强大而智能的图形用户界面智能体面临的主要挑战、局限性和未来研究方向有哪些？（第11节）

Through these questions, this survey aims to provide a comprehensive overview of the current state of the field, offer a guide for building LLM-brained GUI agents, identify key research gaps, and propose directions for future work. This survey is one of the pioneers to systematically examine the domain of LLM-brained GUI agents, integrating perspectives from LLM advancements, GUI automation, and human-computer interaction.
通过这些问题，本次调查旨在全面概述该领域的当前状态，为构建由大语言模型驱动的图形用户界面智能体提供指南，确定关键研究空白，并提出未来工作的方向。本次调查是系统研究由大语言模型驱动的图形用户界面智能体领域的先驱之一，整合了大语言模型进展、图形用户界面自动化和人机交互等方面的观点。

2.5 1.3 Survey Structure

2.6 1.3 调查结构

The survey is organized as follows, with a structural illustration provided in Figure 2. Section 2 reviews related survey and review literature on LLM agents and GUI automation. Section 3 provides preliminary background on LLMs, LLM agents, and GUI automation. Section 2 traces the evolution of LLM-powered GUI agents. Section 5 introduces key components and advanced technologies within LLM-powered GUI agents, serving as a comprehensive guide. Section 6 presents representative frameworks for LLM-powered GUI agents. Section 7 discusses dataset collection and related data-centric research for optimizing LLMs in GUI agent. Section 8 covers foundational and optimized models for GUI agents. Section 9 outlines evaluation metrics and benchmarks. Section 10 explores real-world applications and use cases. Finally, Section 11 examines current limitations, challenges, and potential future directions, and section 12 conclude this survey. For clarity, a list of abbreviations is provided in Table 1
本次调查的组织如下，图2提供了结构示意图。第2节回顾了关于大语言模型智能体和图形用户界面自动化的相关调查和综述文献。第3节提供了大语言模型、大语言模型智能体和图形用户界面自动化的初步背景知识。第4节追溯了由大语言模型驱动的图形用户界面智能体的演变。第5节介绍了由大语言模型驱动的图形用户界面智能体中的关键组件和先进技术，作为全面指南。第6节介绍了由大语言模型驱动的图形用户界面智能体的代表性框架。第7节讨论了数据集收集以及以数据为中心的相关研究，以优化图形用户界面智能体中的大语言模型。第8节涵盖了图形用户界面智能体的基础模型和优化模型。第9节概述了评估指标和基准。第10节探索了现实世界中的应用和用例。最后，第11节探讨了当前的局限性、挑战和潜在的未来方向，第12节对本次调查进行总结。为清晰起见，表1提供了缩略语列表

3 2 RELATED WORK

4 2 相关工作

The integration of LLMs with GUI agents is an emerging and rapidly growing field of research. Several related surveys and tutorials provide foundational insights and guidance. We provide a brief review of existing overview articles on GUI automation and LLM agents, as these topics closely relate to and inform our research focus. To begin, we provide an overview of representative surveys and books on GUI automation, LLM agents, and their integration, as summarized in Table 2 These works either directly tackle one or two core areas in GUI automation and LLM-driven agents, or provide valuable insights that, while not directly addressing the topic, contribute indirectly to advancing the field. GUI agents, application UI screenshots are equally essential, serving as key inputs for reliable task comprehension and execution.

大语言模型（LLM）与图形用户界面（GUI）代理的集成是一个新兴且快速发展的研究领域。一些相关的综述和教程提供了基础见解和指导。我们简要回顾了现有的关于GUI自动化和大语言模型代理的概述文章，因为这些主题与我们的研究重点密切相关，并为其提供了参考。首先，我们概述了关于GUI自动化、大语言模型代理及其集成的代表性综述和书籍，如表2所示。这些工作要么直接涉及GUI自动化和大语言模型驱动代理的一两个核心领域，要么提供了有价值的见解，虽然没有直接涉及该主题，但间接推动了该领域的发展。对于GUI代理来说，应用程序的用户界面截图同样至关重要，它们是可靠地理解和执行任务的关键输入。

---

2. https://www.microsoft.com/en-us/power-platform/blog/
3. https://www.microsoft.com/en-us/power-platform/blog/

power-automate/revolutionize-the-way-you-work-with-automation-and-ai/ 3. https://copilot.microsoft.com/
power-automate/revolutionize-the-way-you-work-with-automation-and-ai/ 3. https://copilot.microsoft.com/

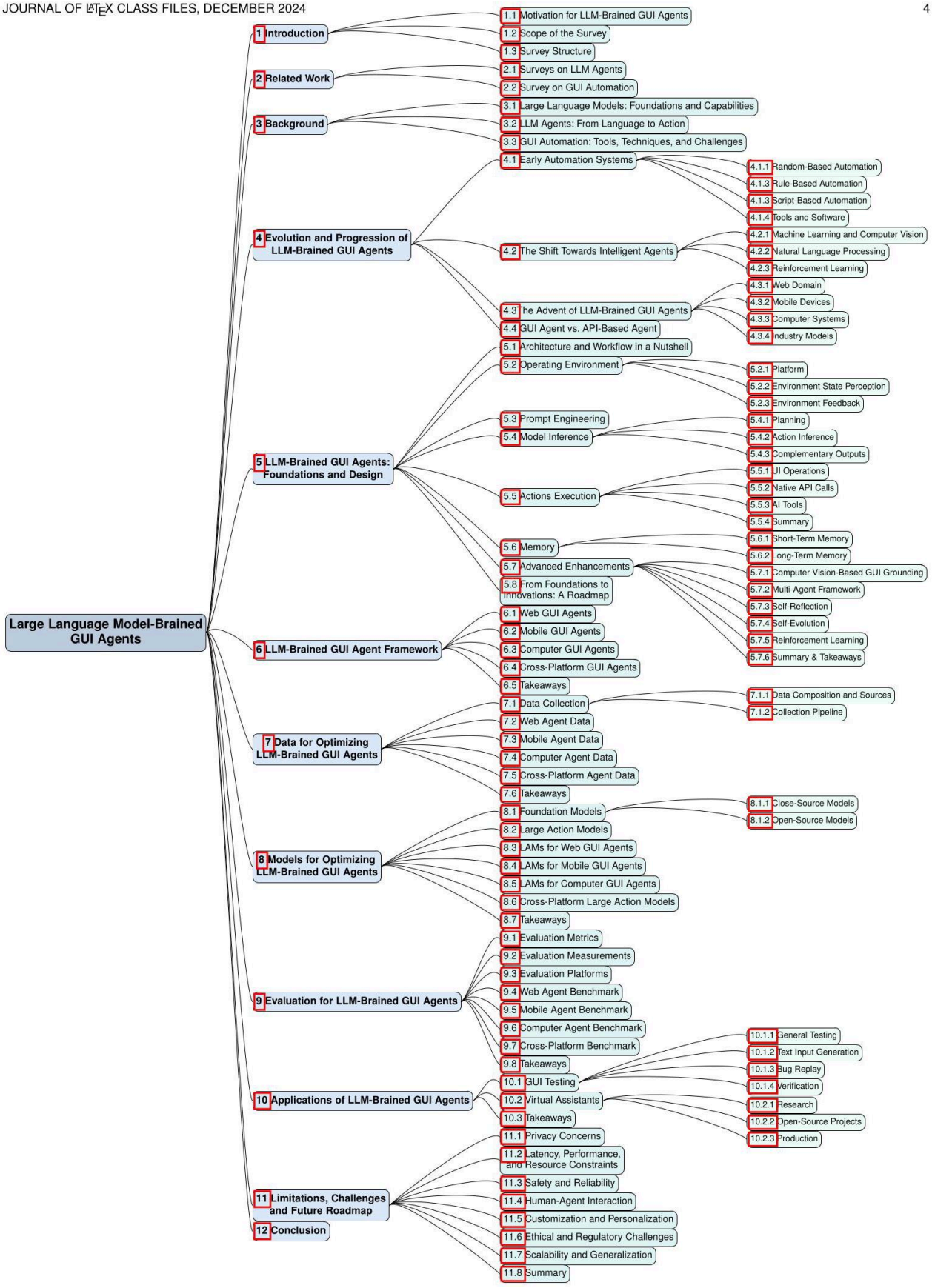

Fig. 2: The structure of the survey on LLM-brained GUI agents.

图2：基于大语言模型的GUI代理综述的结构。

TABLE 1: List of abbreviations in alphabetical order.

表1：按字母顺序排列的缩写列表。

Acronym	Explanation
AI	Artificial Intelligence
AITW	Android in the Wild
AITZ	Android in The Zoo
API	Application Programming Interface
CLI	Command-Line Interface
CLIP	Contrastive Language-Image Pre-Training
CoT	Chain-of-Thought
CSS	Cascading Style Sheets
CUA	Computer-Using Agent
CuP	Completion under Policy
CV	Computer Vision
DOM	Document Object Model
DPO	Direct Preference Optimization
GCC	General Computer Control
GPT	Generative Pre-trained Transformers
GUI	Graphical User Interface
HCI	Human-Computer Interaction
HTML	Hypertext Markup Language
ICL	In-Context Learning
IoU	Intersection over Union
LAM	Large Action Model
LLM	Large Language Model
LSTM	Long Short-Term Memory
LTM	Long-Term Memory
MCTS	Monte Carlo Tree Search
MoE	Mixture of Experts
MDP	Markov Decision Process
MLLM	Multimodal Large Language Model
OCR	Optical Character Recognition
OS	Operation System
RAG	Retrieval-Augmented Generation
ReAct	Reasoning and Acting
RL	Reinforcement Learning
RLHF	Reinforcement Learning from Human Feedback
RNN	Recurrent Neural Network
RPA	Robotic Process Automation
UI	User Interface
UX	User Experience
VAB	VisualAgentBench
VLM	Visual Language Models
ViT	Vision Transformer
VQA	Visual Question Answering
SAM	Segment Anything Model
SoM	Set-of-Mark
STM	Short-Trem Memory

首字母缩写词	解释
人工智能（AI）	人工智能（Artificial Intelligence）
野生安卓（AITW）	野生安卓环境（Android in the Wild）
动物园安卓（AITZ）	受控安卓环境（Android in The Zoo）
应用程序编程接口（API）	应用程序编程接口（Application Programming Interface）
命令行界面（CLI）	命令行界面（Command-Line Interface）
对比语言 - 图像预训练（CLIP）	对比语言 - 图像预训练（Contrastive Language-Image Pre-Training）
思维链（CoT）	思维链（Chain-of-Thought）
层叠样式表（CSS）	层叠样式表（Cascading Style Sheets）
计算机使用代理（CUA）	计算机使用代理（Computer-Using Agent）
策略下完成（CuP）	策略下完成（Completion under Policy）
计算机视觉（CV）	计算机视觉（Computer Vision）
文档对象模型（DOM）	文档对象模型（Document Object Model）
直接偏好优化（DPO）	直接偏好优化（Direct Preference Optimization）
通用计算机控制（GCC）	通用计算机控制（General Computer Control）
生成式预训练变换器（GPT）	生成式预训练变换器（Generative Pre-trained Transformers）
图形用户界面（GUI）	图形用户界面（Graphical User Interface）
人机交互（HCI）	人机交互（Human-Computer Interaction）
超文本标记语言（HTML）	超文本标记语言（Hypertext Markup Language）
上下文学习（ICL）	上下文学习（In-Context Learning）
交并比（IoU）	交并比（Intersection over Union）
大型动作模型（LAM）	大型动作模型（Large Action Model）
大语言模型（LLM）	大语言模型（Large Language Model）
长短期记忆网络（LSTM）	长短期记忆（Long Short-Term Memory）
长期记忆（LTM）	长期记忆（Long-Term Memory）
蒙特卡罗树搜索（MCTS）	蒙特卡罗树搜索（Monte Carlo Tree Search）
专家混合模型（MoE）	专家混合模型（Mixture of Experts）
马尔可夫决策过程（MDP）	马尔可夫决策过程（Markov Decision Process）
多模态大语言模型（MLLM）	多模态大语言模型（Multimodal Large Language Model）
光学字符识别（OCR）	光学字符识别（Optical Character Recognition）
操作系统（OS）	操作系统（Operation System）
检索增强生成（RAG）	检索增强生成（Retrieval-Augmented Generation）
反应式行动（ReAct）	推理与行动（Reasoning and Acting）
强化学习（RL）	强化学习（Reinforcement Learning）
基于人类反馈的强化学习（RLHF）	基于人类反馈的强化学习（Reinforcement Learning from Human Feedback）
循环神经网络（RNN）	循环神经网络（Recurrent Neural Network）
机器人流程自动化（RPA）	机器人流程自动化（Robotic Process Automation）
用户界面（UI）	用户界面（User Interface）
用户体验（UX）	用户体验（User Experience）
视觉智能体基准测试（VAB）	视觉智能体基准测试（VisualAgentBench）
视觉语言模型（VLM）	视觉语言模型（Visual Language Models）
视觉变换器（ViT）	视觉变换器（Vision Transformer）
视觉问答（VQA）	视觉问答（Visual Question Answering）
任意分割模型（SAM）	任意分割模型（Segment Anything Model）
标记集（SoM）	标记集（Set-of-Mark）
短期记忆（STM）	短期记忆（Short-Trem Memory）

4.1 2.1 Survey on GUI Automation

4.2 2.1 GUI自动化综述

GUI automation has a long history and wide applications in industry, especially in GUI testing [27]-[29] and RPA [6] for task automation [42].

GUI自动化在工业界有着悠久的历史和广泛的应用，尤其是在GUI测试[27]-[29]和用于任务自动化的机器人流程自动化（RPA）[6][42]中。

Said et al., [30] provide an overview of GUI testing for mobile applications, covering objectives, approaches, and challenges within this domain. Focusing on Android applications, Li [31] narrows the scope further, while Oksanen et al., 32 explore automatic testing techniques for Windows GUI applications, a key platform for agent operations. Similarly, Moura et al., 72 review GUI testing for web applications, which involves diverse tools, inputs, and methodologies. Deshmukh et al., 33 discuss automated GUI testing for enhancing user experience, an area where LLMs also bring new capabilities. A cornerstone of modern GUI testing is computer vision (CV), which is used to interpret UI elements and identify actionable controls [34]. Yu et al., [35] survey the application of CV in mobile GUI testing, highlighting both its significance and associated challenges. In LLM-powered
Said等人[30]提供了移动应用GUI测试的概述，涵盖了该领域的目标、方法和挑战。Li[31]聚焦于Android应用，进一步缩小了研究范围，而Oksanen等人[32]探讨了Windows GUI应用的自动测试技术，Windows是代理操作的重要平台。同样，Moura等人[72]回顾了Web应用的GUI测试，涉及多样的工具、输入和方法。Deshmukh等人[33]讨论了自动化GUI测试以提升用户体验的研究领域，LLM（大型语言模型）在此也带来了新能力。现代GUI测试的基石是计算机视觉（CV），用于解析用户界面元素并识别可操作控件[34]。Yu等人[35]综述了CV在移动GUI测试中的应用，强调了其重要性及相关挑战。在LLM驱动的

On the other hand, RPA, which focuses on automating repetitive human tasks, also relies heavily on GUI automation for relevant processes. Syed et al., [36] review this field and highlight contemporary RPA themes, identifying key challenges for future research. Chakraborti et al., 37 emphasize the importance of shifting from traditional, script-based RPA toward more intelligent, adaptive paradigms, offering a systematic overview of advancements in this direction. Given RPA's extensive industrial applications, Enriquez et al., [38] and Ribeiro et al., 39 survey the field from an industrial perspective, underscoring its significance and providing a comprehensive overview of RPA methods, development trends, and practical challenges.
另一方面，RPA专注于自动化重复的人类任务，也高度依赖GUI自动化来实现相关流程。Syed等人[36]回顾了该领域，强调了当代RPA的主题，并指出未来研究的关键挑战。Chakraborti等人[37]强调了从传统脚本驱动的RPA向更智能、适应性强的范式转变的重要性，系统性地概述了该方向的进展。鉴于RPA在工业中的广泛应用，Enriquez等人[38]和Ribeiro等人[39]从工业视角对该领域进行了综述，强调其重要性并全面介绍了RPA的方法、发展趋势和实际挑战。

Both GUI testing [40] and RPA [41] continue to face significant challenges in achieving greater intelligence and robustness. LLM-powered GUI agents are poised to play a transformative role in these fields, providing enhanced capabilities and adding substantial value to address these persistent issues.
GUI测试[40]和RPA[41]在实现更高智能化和鲁棒性方面仍面临重大挑战。LLM驱动的GUI代理有望在这些领域发挥变革性作用，提供增强的能力并为解决这些持续存在的问题带来显著价值。

4.3 2.2 Surveys on LLM Agents

4.4 2.2 LLM代理综述

The advent of LLMs has significantly enhanced the capabilities of intelligent agents [43], enabling them to tackle complex tasks previously out of reach, particularly those involving natural language understanding and code generation [44]. This advancement has spurred substantial research into LLM-based agents designed for a wide array of applications [45].
大型语言模型（LLM）的出现显著提升了智能代理的能力[43]，使其能够处理此前难以企及的复杂任务，尤其是涉及自然语言理解和代码生成的任务[44]。这一进展推动了针对广泛应用设计的基于LLM的代理的深入研究[45]。

Both Xie et al., [46] and Wang et al., [47] offer comprehensive surveys on LLM-powered agents, covering essential background information, detailed component breakdowns, taxonomies, and various applications. These surveys serve as valuable references for a foundational understanding of LLM-driven agents, laying the groundwork for further exploration into LLM-based GUI agents. Xie et al., [59] provide an extensive overview of multimodal agents, which can process images, videos, and audio in addition to text. This multimodal capability significantly broadens the scope beyond traditional text-based agents [60]. Notably, most GUI agents fall under this category, as they rely on image inputs, such as screenshots, to interpret and interact with graphical interfaces effectively. Multi-agent frameworks are frequently employed in the design of GUI agents to enhance their capabilities and scalability. Surveys by Guo et al., [48] and Han et al., [49] provide comprehensive overviews of the current landscape, challenges, and future directions in this area. Sun et al., [50] provide an overview of recent methods that leverage reinforcement learning to strengthen multi-agent LLM systems, opening new pathways for enhancing their capabilities and adaptability. These surveys offer valuable insights and guidance for designing effective multi-agent systems within GUI agent frameworks.
Xie等人[46]和Wang等人[47]均提供了关于LLM驱动代理的全面综述，涵盖了基础背景、详细组件解析、分类体系及多种应用。这些综述为理解LLM驱动代理奠定了基础，促进了对基于LLM的GUI代理的进一步探索。Xie等人[59]还对多模态代理进行了广泛介绍，这类代理除了文本外还能处理图像、视频和音频。多模态能力显著拓宽了传统文本代理的应用范围[60]。值得注意的是，大多数GUI代理属于此类，因为它们依赖图像输入（如截图）来有效解析和交互图形界面。多代理框架常被用于GUI代理的设计，以增强其能力和可扩展性。Guo等人[48]和Han等人[49]的综述全面介绍了该领域的现状、挑战及未来方向。Sun等人[50]则概述了利用强化学习强化多代理LLM系统的最新方法，为提升其能力和适应性开辟了新途径。这些综述为设计高效的多代理系统提供了宝贵的见解和指导。

In the realm of digital environments, Wu et al., [61] presents a survey on LLM agents operating in mobile environments, covering key aspects of mobile GUI agents. In a boarder scope, Wang et al., [62] present a survey on the integration of foundation models with GUI agents. Another survey by Gao et al., provides an overview of autonomous
在数字环境领域，Wu等人[61]对移动环境中运行的LLM代理进行了综述，涵盖了移动GUI代理的关键方面。更广泛地，Wang等人[62]综述了

基础模型与GUI代理的整合。Gao等人则提供了跨多种数字平台自主代理的概览

JOURNAL OF LATEX CLASS FILES, DECEMBER 2024 agents operating across various digital platforms [63], highlighting their capabilities, challenges, and applications. All these surveys highlighting emerging trends in this area.
《JOURNAL OF LATEX CLASS FILES, DECEMBER 2024》报道了这些代理的能力、挑战和应用。所有这些综述均强调了该领域的新兴趋势。

TABLE 2: Summary of representative surveys and books on GUI automation and LLM agents. A √symbol indicates that a publication explicitly addresses a given domain, while an ○ symbol signifies that the publication does not focus on the area but offers relevant insights. Publications covering both GUI automation and LLM agents are highlighted for emphasis.
表2：GUI自动化和LLM代理代表性综述及著作汇总。√符号表示该出版物明确涉及相关领域，○符号表示该出版物虽未专注于该领域但提供了相关见解。涵盖GUI自动化和LLM代理两者的出版物以加粗形式突出显示。

Survey	One Sentence Summary	Scope		
		GUI Automation	LLM Agent	LLM Agent + GUI Automation
Li et al., 27]	A book on how to develop an automated GUI testing tool.	✓		
Rodríguez et al., 28	A survey on automated GUI testing in 30 years.	✓		
Arnatovich et al., 29	A survey on automated techniques for mobile functional GUI testing.	✓		
Ivančić et al., [6]	A literature review on RPA.	✓		
Said et al., 30	An overview on mobile GUI testing.	✓		
Li 31	An survey on Android GUI testing.	✓		
Oksanen et al., 32	GUI testing on Windows OS.	✓		
Deshmukh et al., 33	A survey on GUI testing for improving user experience.	✓		
Bajammal et al., 34	A survey on the use of computer vision for software engineering.	✓		
Yu et al., 35	A survey on using computer for mobile app GUI testing.	✓		
Syed et al., 36	A review of contemporary themes and challenges in RPA.	✓		
Chakraborti et al., 37	A review of emerging trends of intelligent process automation.	✓		
Enriquez et al., 38	A scientific and industrial systematic mapping study of RPA.	✓		
Ribeiro et al., 39	A review of combining AI and RPA in industry 4.0.	✓		
Nass et al., 40	Discuss the chanllenges of GUI testing.	✓		
Agostinelli et al., 41	Discuss the research challenges of intelligent RPA.	✓		
Wali et al., 42]	A review on task automation with intelligent agents.	✓		
Zhao et al., 8	A comprehensive survey of LLMs.		✓	
Zhao et al., 44	A survey of LLM-based agents.		✓	
Cheng et al., 44	An overview of LLM-based AI agent.		✓	
Li et al., 45	A survey on personal LLM agents on their capability, efficiency and security.		✓	
Xie et al., [46]	A comprehensive survey of LLM-based agents.		✓	
Wang et al.,. 47	A survey on LLM-based autonomous agents.		✓	
Guo et al., 48	A survey of mult-agent LLM frameworks.		✓	
Han et al., 49	A survey on LLM multi-agent systems, with their challenges and open problems.		✓	
Sun et al., 50	A survey on LLM-based multi-agent reinforcement learning.		✓	
Huang et al., 51]	A survey on planning in LLM agents.		✓	
Aghzal et al., 52	A survey on automated planning in LLMs.		✓	
Zheng et al., 53	Discuss the roadmap of lifelong learning in LLM agents.		✓	
Zhang et al., 54	A survey on the memory of LLM-based agents.		✓	
Shen 13	A survey of the tool usage in LLM agents.		✓	
Chang et al., 55	A survey on evaluation of LLMs.		✓	
Li et al., 56	A survey on benchmarks multimodal applications.		✓	
Li et al., 57	A survey on benchmarking evaluations, applications, and challenges of visual LLMs.		✓	
Huang and Zhang 58	A survey on evaluation of multimodal LLMs.		✓	
Xie et al., 69	A survey on LLM based multimodal agent.		✓	①
Durante et al., 60	A survey of multimodal interaction with AI agents.		✓	0
Wu et al., 611	A survey of foundations and trend on multimodal mobile agents.		✓	✓
Wang et al., 62	A survey on the integration of foundation models with GUI agents.		✓	✓
Gao et al., 63	A survey on autonomous agents across digital platforms.		✓	✓
Dang et al., 64	A survey on GUI agents.		✓	✓
Liu et al., 65	A survey on GUI agent on phone automation.		✓	✓
Hu et al., 66	A survey on MLLM based agents for OS.		✓	✓
Shi et al., 167	A survey of building trustworthy GUI agents.		✓	✓

Ning et al., 68	A survey of agents for Web automation.		✓	✓
Tang et al., 69	A survey of GUI agents powered by (multimodal) LLMs.		✓	✓
Li and Huang et al., 70	A summary of GUI agents powered by foundation models and enhanced through reinforcement learning		✓	✓
Sager et al., 71	A review of AI agent for computer use.	O	✓	✓
Our work	A comprehensive survey on LLM-brained GUI agents, on their foundations, technologies, frameworks, data, models, applications, challenges and future roadmap.	O	✓	✓

调查	一句话总结	范围		
		图形用户界面自动化（GUI Automation）	大语言模型智能体（LLM Agent）	大语言模型智能体 + 图形用户界面自动化（LLM Agent + GUI Automation）
李等人，[27]	一本关于如何开发自动化图形用户界面测试工具的书。	✓		
罗德里格斯等人，[28]	一项关于30年自动化图形用户界面测试的调查。	✓		
阿尔纳托维奇等人，[29]	一项关于移动功能图形用户界面测试自动化技术的调查。	✓		
伊万契奇等人，[6]	一篇关于机器人流程自动化（RPA）的文献综述。	✓		
赛义德等人，[30]	一篇关于移动图形用户界面测试的概述。	✓		
李，[31]	一项关于安卓图形用户界面测试的调查。	✓		
奥克萨宁等人，[32]	Windows操作系统上的图形用户界面测试。	✓		
德什穆克等人，[33]	一项关于为改善用户体验进行图形用户界面测试的调查。	✓		
巴贾马尔等人，[34]	一项关于计算机视觉在软件工程中应用的调查。	✓		
余等人，[35]	一项关于使用计算机进行移动应用图形用户界面测试的调查。	✓		
赛义德等人，[36]	一篇关于机器人流程自动化（RPA）当代主题和挑战的综述。	✓		
查克拉博蒂等人，[37]	一篇关于智能流程自动化新兴趋势的综述。	✓		
恩里克兹等人，[38]	一篇关于机器人流程自动化（RPA）的科学与工业系统映射研究。	✓		
里贝罗等人，[39]	一篇关于工业4.0中人工智能与机器人流程自动化（RPA）结合的综述。	✓		
纳斯等人，[40]	讨论图形用户界面测试的挑战。	✓		
阿戈斯蒂内利等人，[41]	讨论智能机器人流程自动化（RPA）的研究挑战。	✓		
瓦利等人，[42]	一篇关于使用智能体进行任务自动化的综述。	✓		
赵等人，[8]	一项关于大语言模型（LLMs）的全面调查。		✓	
赵等人，[44]	一项关于基于大语言模型（LLM）的智能体的调查。		✓	
程等人，[44]	一篇关于基于大语言模型（LLM）的人工智能智能体的概述。		✓	
李等人，[45]	一项关于个人大语言模型（LLM）智能体的能力、效率和安全性的调查。		✓	
谢等人，[46]	一项关于基于大语言模型（LLM）的智能体的全面调查。		✓	
王等人，[47]	一项关于基于大语言模型（LLM）的自主智能体的调查。		✓	
郭等人，[48]	一项关于多智能体大语言模型（LLM）框架的调查。		✓	
韩等人，[49]	一项关于大语言模型（LLM）多智能体系统及其挑战和开放性问题的调查。		✓	
孙等人，[50]	一项关于基于大语言模型（LLM）的多智能体强化学习的调查。		✓	
黄等人，[51]	一项关于大语言模型（LLM）智能体规划的调查。		✓	

阿格扎尔等人，[52]	一项关于大语言模型（LLMs）中自动化规划的调查。		✓	
郑等人，[53]	讨论大语言模型（LLM）智能体的终身学习路线图。		✓	
张等人，54	基于大语言模型的智能体的记忆研究综述。		✓	
沈13	大语言模型智能体的工具使用研究综述。		✓	
常等人，55	大语言模型评估研究综述。		✓	
李等人，56	多模态应用基准测试研究综述。		✓	
李等人，57	视觉大语言模型的基准评估、应用和挑战研究综述。		✓	
黄和张58	多模态大语言模型评估研究综述。		✓	
谢等人，69	基于大语言模型的多模态智能体研究综述。		✓	①
杜兰特等人，60	与人工智能智能体的多模态交互研究综述。		✓	0
吴等人，611	多模态移动智能体的基础和趋势研究综述。		✓	✓
王等人，62	基础模型与图形用户界面（GUI）智能体集成研究综述。		✓	✓
高等人，63	跨数字平台的自主智能体研究综述。		✓	✓
党等人，64	图形用户界面智能体研究综述。		✓	✓
刘等人，65	手机自动化中的图形用户界面智能体研究综述。		✓	✓
胡等人，66	基于多模态大语言模型的操作系统智能体研究综述。		✓	✓
施等人，167	构建可信图形用户界面智能体研究综述。		✓	✓
宁等人，68	网络自动化智能体研究综述。		✓	✓
唐等人，69	由（多模态）大语言模型驱动的图形用户界面智能体研究综述。		✓	✓
李和黄等人，70	由基础模型驱动并通过强化学习增强的图形用户界面智能体总结		✓	✓
萨格等人，71	计算机使用的人工智能智能体研究综述。	O	✓	✓
我们的工作	对具有大语言模型“大脑”的图形用户界面智能体进行全面综述，涵盖其基础、技术、框架、数据、模型、应用、挑战和未来路线图。	O	✓	✓

Regarding individual components within LLM agents, several surveys provide detailed insights that are especially relevant for GUI agents. Huang et al., [51] examine planning mechanisms in LLM agents, which are essential for executing long-term tasks-a frequent requirement in GUI automation. Zhang et al., [54] explore memory mechanisms, which allow agents to store critical historical information, aiding in knowledge retention and decision-making. Additionally, Shen [13] surveys the use of tools by LLMs (such as APIs and code) to interact effectively with their environments, grounding actions in ways that produce tangible impacts. Further, Chang et al., [55] provide a comprehensive survey on evaluation methods for LLMs, which is crucial for ensuring the robustness and safety of GUI agents. Two additional surveys, [56] and [58], provide comprehensive overviews of benchmarks and evaluation methods specifically tailored to multimodal LLMs. The evaluation also facilitates a feedback loop, allowing agents to improve iteratively based on assessment results. Together, these surveys serve as valuable resources, offering guidance on essential components of LLM agents and forming a foundational basis for LLM-based GUI agents.

关于LLM代理中的各个组成部分，几项综述提供了详细见解，尤其适用于GUI代理。Huang等人[51]研究了LLM代理中的规划机制，这对于执行长期任务至关重要——这是GUI自动化中的常见需求。Zhang等人[54]探讨了记忆机制，使代理能够存储关键的历史信息，有助于知识保留和决策。此外，Shen[13]综述了LLM使用工具（如API和代码）与环境有效交互的方法，使其行为具备实际影响力。Chang等人[55]则提供了关于LLM评估方法的全面综述，这对于确保GUI代理的稳健性和安全性至关重要。另有两篇综述[56]和[58]专门针对多模态LLM的基准和评估方法进行了全面概述。评估还促进了反馈循环，使代理能够基于评估结果迭代改进。综上，这些综述作为宝贵资源，为LLM代理的关键组成部分提供指导，构成了基于LLM的GUI代理的基础。

Compared to existing surveys, our work offers a significantly more comprehensive and up-to-date overview of the LLM-powered GUI agent landscape. We curate and synthesize over 500 references, covering a wide range of topics including foundation models, data sources, system frameworks, benchmarks, evaluation methodologies, and practical deployments. While prior surveys often concentrate on narrower aspects on selected platform (e.g., web, mobile), our survey takes a holistic perspective that spans the full development and deployment lifecycle. Beyond narrative summaries, we also provide consolidated reference tables for each subdomain, enabling readers to quickly categorize and locate relevant works across platforms and research themes—serving as a practical handbook for both researchers and practitioners. Furthermore, we incorporate foundational background material and propose evaluation taxonomies that make the survey accessible to newcomers, addressing gaps in prior work that often assume a high degree of prior familiarity.
与现有综述相比，我们的工作提供了更全面且最新的LLM驱动GUI代理领域概览。我们整理并综合了500多篇文献，涵盖基础模型、数据源、系统框架、基准、评估方法及实际部署等广泛主题。此前的综述多聚焦于特定平台（如网页、移动端）的狭窄方面，而我们的综述则采取全生命周期的整体视角。除了叙述性总结外，我们还为各子领域提供了整合的参考表，方便读者快速分类并定位跨平台及研究主题的相关工作——为研究者和从业者提供实用手册。此外，我们融入了基础背景材料并提出了评估分类法，使综述对新手友好，弥补了以往工作中常假设读者具备较高先验知识的不足。

5 3 BACKGROUND

6 3 背景

The development of LLM-brained GUI agents is grounded in three major advancements: (i) large language models (LLMs) [8], which bring advanced capabilities in natural language understanding and code generation, forming the core intelligence of these agents; (ii) accompanying agent architectures and tools [47] that extend LLM capabilities, bridging the gap between language models and physical environments to enable tangible impacts; and (iii) GUI automation [73], which has cultivated a robust set of tools, models, and methodologies essential for GUI agent functionality. Each of these components has played a critical role in the emergence of LLM-powered GUI agents. In the following subsections, we provide a brief overview of these areas to set the stage for our discussion.
基于LLM的GUI代理的发展依托于三大进展：（i）大型语言模型（LLMs）[8]，具备先进的自然语言理解和代码生成能力，构成这些代理的核心智能；（ii）配套的代理架构和工具[47]，扩展了LLM的能力，弥合语言模型与物理环境之间的鸿沟，实现实际影响；（iii）GUI自动化[73]，培养了一套完善的工具、模型和方法论，是GUI代理功能的关键支撑。这些组成部分共同推动了LLM驱动GUI代理的出现。以下小节将简要介绍这些领域，为后续讨论奠定基础。

6.1 3.1 Large Language Models: Foundations and Capabili- ties

6.2 3.1 大型语言模型：基础与能力

The study of language models has a long and rich history [74], beginning with early statistical language models [75] and smaller neural network architectures [76]. Building on these foundational concepts, recent advancements have focused on transformer-based LLMs, such as the Generative Pre-trained Transformers (GPTs) [77]. These models are pretrained on extensive text corpora and feature significantly larger model sizes, validating scaling laws and demonstrating exceptional capabilities across a wide range of natural language tasks. Beyond their sheer size, these LLMs exhibit enhanced language understanding and generation abilities, as well as emergent properties that are absent in smaller-scale language models 78].
语言模型的研究历史悠久且丰富[74]，始于早期的统计语言模型[75]和较小的神经网络架构[76]。在这些基础概念上，近期进展聚焦于基于Transformer的大型语言模型，如生成式预训练变换器（GPTs）[77]。这些模型在大规模文本语料上预训练，模型规模显著增大，验证了规模定律，并在广泛的自然语言任务中展现出卓越能力。除了规模庞大，这些LLM还表现出增强的语言理解与生成能力，以及小规模语言模型所不具备的涌现特性[78]。

Early neural language models, based on architectures like recurrent neural networks (RNNs) [79] and long short-term memory networks (LSTMs) [80], were limited in both performance and generalization. The introduction of the Transformer model, built on the attention mechanism [81], marked a transformative milestone, establishing the foundational architecture now prevalent across almost all subsequent LLMs. This development led to variations in model structures, including encoder-only models (e.g., BERT [82], RoBERTa [83], ALBERT [84]), decoder-only models (e.g., GPT-1 [85], GPT-2 [86]), and encoder-decoder models (e.g., T5 [87], BART [88]). In 2022, ChatGPT [11] based on GPT-3.5 [89] launched as a groundbreaking LLM, fundamentally shifting perceptions of what language models can achieve. Since then, numerous advanced LLMs have emerged, including GPT-4 [90], LLaMA-3 [91], and Gemini [92], propelling the field into rapid growth. Today's LLMs are highly versatile, with many of them are capable of processing multimodal data and performing a range of tasks, from question answering to code generation, making them indispensable tools in various applications [93]-[96].
早期神经语言模型基于循环神经网络（RNNs）[79]和长短期记忆网络（LSTMs）[80]，在性能和泛化能力上均有限。Transformer模型的引入，基于注意力机制[81]，标志着一个变革性里程碑，奠定了几乎所有后续LLM通用的基础架构。该发展催生了多种模型结构，包括仅编码器模型（如BERT[82]、RoBERTa[83]、ALBERT[84]）、仅解码器模型（如GPT-1[85]、GPT-2[86]）和编码器-解码器模型（如T5[87]、BART[88]）。2022年，基于GPT-3.5[89]的ChatGPT[11]作为突破性LLM发布，根本改变了人们对语言模型能力的认知。此后，众多先进LLM相继出现，包括GPT-4[90]、LLaMA-3[91]和Gemini[92]，推动该领域快速发展。现今的LLM高度多才多艺，许多模型能够处理多模态数据，执行从问答到代码生成的多种任务，成为各类应用中不可或缺的工具[93]-[96]。

The emergence of LLMs has also introduced significant advanced properties that invigorate their applications, making previously challenging tasks, such as natural language-driven GUI agents feasible. These advancements include:
LLM的出现还带来了显著的高级特性，激发了其应用潜力，使得此前具有挑战性的任务，如基于自然语言的GUI代理成为可能。这些进展包括：

1. Few-Shot Learning [77]: Also referred to as in-context learning [97], LLMs can acquire new tasks from a small set of demonstrated examples presented in the prompt during inference, eliminating the need for retraining. This capability is crucial for enabling GUI agents to generalize across different environments with minimal effort.
2. 少样本学习[77]：也称为上下文学习[97]，大型语言模型（LLMs）能够从推理时提示中提供的一小组示例中学习新任务，无需重新训练。这一能力对于使GUI代理能够以最小的努力在不同环境中实现泛化至关重要。

2. Instruction Following [98]: After undergoing instruction tuning, LLMs exhibit a remarkable ability to follow instructions for novel tasks, demonstrating strong generalization skills [89]. This allows LLMs to effectively comprehend user requests directed at GUI agents and to follow predefined objectives accurately.
3. 指令遵循[98]：经过指令微调后，LLMs展现出对新任务指令的卓越遵循能力，表现出强大的泛化能力[89]。这使得LLMs能够有效理解用户针对GUI代理的请求，并准确执行预设目标。

3. Long-Term Reasoning [99]: LLMs possess the ability to plan and solve complex tasks by breaking them down into manageable steps, often employing techniques like chain-of-thought (CoT) reasoning [100], [101]. This capability is essential for GUI agents, as many tasks require multiple steps and a robust planning framework.
4. 长期推理[99]：LLMs具备通过将复杂任务分解为可管理步骤来规划和解决问题的能力，常采用链式思维（chain-of-thought, CoT）推理技术[100],[101]。这一能力对GUI代理尤为重要，因为许多任务需要多步骤和稳健的规划框架。

4. Code Generation and Tool Utilization [102]: LLMs excel in generating code and utilizing various tools, such as APIs [13]. This expertise is vital, as code and tools form the essential toolkit for GUI agents to interact with their environments.
5. 代码生成与工具使用[102]：LLMs擅长生成代码并利用各种工具，如API[13]。这项专长至关重要，因为代码和工具构成了GUI代理与其环境交互的基本工具包。

5. Multimodal Comprehension [10]: Advanced LLMs can integrate additional data modalities, such as images, into their training processes, evolving into multimodal models. This ability is particularly important for GUI agents, which must interpret GUI screenshots presented as images in order to function effectively [103].
6. 多模态理解[10]：先进的LLMs能够将图像等额外数据模态整合进训练过程，发展为多模态模型。这一能力对GUI代理尤为重要，因为它们必须解读以图像形式呈现的GUI截图以实现有效功能[103]。

To further enhance the specialization of LLMs for GUI agents, researchers often fine-tune these models with domain-specific data, such as user requests, GUI screenshots, and action sequences, thereby increasing their customization and effectiveness. In Section 8, we delve into these advanced, tailored models for GUI agents, discussing their unique adaptations and improved capabilities for interacting with graphical interfaces.
为了进一步提升LLMs在GUI代理领域的专业化，研究人员通常使用领域特定数据进行微调，如用户请求、GUI截图和操作序列，从而增强其定制化和效能。在第8节中，我们将深入探讨这些面向GUI代理的高级定制模型，讨论其独特的适应性和改进的图形界面交互能力。

6.3 3.2 LLM Agents: From Language to Action

6.4 3.2 LLM代理：从语言到行动

Traditional AI agents have often focused on enhancing specific capabilities, such as symbolic reasoning or excelling in particular tasks like Go or Chess. In contrast, the emergence of LLMs has transformed AI agents by providing them with a natural language interface, enabling human-like decision-making capabilities, and equipping them to perform a wide variety of tasks and take tangible actions in diverse environments [12], [47], [104], [105]. In LLM agents, if LLMs form the "brain" of a GUI agent, then its accompanying components serve as its "eyes and hands", enabling the LLM to perceive the environment's status and translate its textual output into actionable steps that generate tangible effects [46]. These components transform LLMs from passive information sources into interactive agents that execute tasks on behalf of users, which redefine the role of LLMs from purely text-generative models to systems capable of driving actions and achieving specific goals.
传统的AI代理通常专注于提升特定能力，如符号推理或在围棋、国际象棋等特定任务上的卓越表现。相比之下，LLMs的出现通过提供自然语言接口，赋予AI代理类人决策能力，使其能够执行多样化任务并在多种环境中采取实际行动[12],[47],[104],[105]。在LLM代理中，如果LLMs构成GUI代理的“大脑”，那么其配套组件则是“眼睛和手”，使LLM能够感知环境状态并将文本输出转化为可执行步骤，产生实际效果[46]。这些组件将LLMs从被动的信息源转变为代表用户执行任务的交互式代理，重新定义了LLMs从纯文本生成模型到能够驱动行动并实现特定目标的系统的角色。

In the context of GUI agents, the agent typically perceives the GUI status through screenshots and widget trees [106], then performs actions to mimic user operations (e.g., mouse clicks, keyboard inputs, touch gestures on phones) within the environment. Since tasks can be long-term, effective planning and task decomposition are often required, posing unique challenges. Consequently, an LLM-powered GUI agent usually possess multimodal capabilities [59], a robust planning system [51], a memory mechanism to analyze historical interactions [54], and a specialized toolkit to interact with its environment [27]. We will discuss these tailored designs for GUI agents in detail in Section 5

在GUI代理的语境中，代理通常通过截图和控件树[106]感知GUI状态，然后执行操作以模拟用户行为（如鼠标点击、键盘输入、手机触摸手势）在环境中进行交互。由于任务可能是长期的，通常需要有效的规划和任务分解，这带来了独特挑战。因此，基于LLM的GUI代理通常具备多模态能力[59]、稳健的规划系统[51]、用于分析历史交互的记忆机制[54]以及与环境交互的专用工具包[27]。我们将在第5节详细讨论这些针对GUI代理的定制设计。

6.5 3.3 GUI Automation: Tools, Techniques, and Challenges

6.6 3.3 GUI自动化：工具、技术与挑战

GUI automation has been a critical area of research and application since the early days of GUIs in computing. Initially developed to improve software testing efficiency, GUI automation focused on simulating user actions, such as clicks, text input, and navigation, across graphical applications to validate functionality [30]. Early GUI automation tools were designed to execute repetitive test cases on static interfaces [28]. These approaches streamlined quality assurance processes, ensuring consistency and reducing manual testing time. As the demand for digital solutions has grown, GUI automation has expanded beyond testing to other applications, including RPA [6] and Human-Computer Interaction (HCI) [107]. RPA leverages GUI automation to replicate human actions in business workflows, automating routine tasks to improve operational efficiency. Similarly, HCI research employs GUI automation to simulate user behaviors, enabling usability assessments and interaction studies. In both cases, automation has significantly enhanced productivity and user experience by minimizing repetitive tasks and enabling greater system adaptability [108], [109].

GUI自动化自计算机图形界面诞生之初便是研究和应用的关键领域。最初为提升软件测试效率而开发，GUI自动化侧重于模拟用户操作，如点击、文本输入和导航，以验证图形应用的功能[30]。早期的GUI自动化工具设计用于在静态界面上执行重复测试用例[28]。这些方法简化了质量保证流程，确保一致性并减少人工测试时间。随着数字解决方案需求的增长，GUI自动化已扩展至测试之外的应用，包括机器人流程自动化（RPA）[6]和人机交互（HCI）[107]。RPA利用GUI自动化复制业务流程中的人工操作，自动化常规任务以提升运营效率。同样，HCI研究采用GUI自动化模拟用户行为，支持可用性评估和交互研究。在这两种情况下，自动化显著提升了生产力和用户体验，减少重复任务并增强系统适应性[108],[109]。

Traditional GUI automation methods have primarily depended on scripting and rule-based frameworks [4, [110]. Scripting-based automation utilizes languages such as Python, Java, and JavaScript to control GUI elements programmatically. These scripts simulate a user's actions on the interface, often using tools like Selenium [111] for web-based automation or Autolt [112] and SikuliX [113] for desktop applications. Rule-based approaches, meanwhile, operate based on predefined heuristics, using rules to detect and interact with specific GUI elements based on properties such as location, color, and text labels [4]. While effective for predictable, static workflows [114], these methods struggle to adapt to the variability of modern GUIs, where dynamic content, responsive layouts, and user-driven changes make it challenging to maintain rigid, rule-based automation [115].

传统的GUI自动化方法主要依赖于脚本和基于规则的框架[4, [110]。基于脚本的自动化使用Python、Java和JavaScript等语言以编程方式控制GUI元素。这些脚本模拟用户在界面上的操作，通常使用Selenium[111]进行基于网页的自动化，或使用Autolt[112]和SikuliX[113]进行桌面应用程序的自动化。基于规则的方法则基于预定义的启发式规则，通过元素的位置、颜色和文本标签等属性检测并与特定GUI元素交互[4]。虽然这些方法在可预测的静态工作流程中效果显著[114]，但面对现代GUI中动态内容、响应式布局和用户驱动的变化时，难以维持刚性的基于规则的自动化[115]。

CV has become essential for interpreting the visual aspects of GUIs [35], [116], [117], enabling automation tools to recognize and interact with on-screen elements even as layouts and designs change. CV techniques allow GUI automation systems to detect and classify on-screen elements, such as buttons, icons, and text fields, by analyzing screenshots and identifying regions of interest [118-120]. Optical Character Recognition (OCR) further enhances this capability by extracting text content from images, making it possible for automation systems to interpret labels, error messages, and form instructions accurately [121]. Object detection models add robustness, allowing automation agents to locate GUI elements even when the visual layout shifts [103]. By incorporating CV, GUI automation systems achieve greater resilience and adaptability in dynamic environments.

计算机视觉（CV）已成为解释GUI视觉元素的关键技术[35]，[116]，[117]，使自动化工具能够识别并与屏幕上的元素交互，即使布局和设计发生变化。CV技术通过分析截图并识别感兴趣区域，允许GUI自动化系统检测和分类屏幕元素，如按钮、图标和文本框[118-120]。光学字符识别（OCR）进一步增强了这一能力，通过从图像中提取文本内容，使自动化系统能够准确理解标签、错误信息和表单指令[121]。目标检测模型提升了鲁棒性，使自动化代理即使在视觉布局变化时也能定位GUI元素[103]。通过引入计算机视觉，GUI自动化系统在动态环境中实现了更强的适应性和弹性。

Despite advances, traditional GUI automation methods fall short in handling the complexity and variability of contemporary interfaces. Today's applications often feature dynamic, adaptive elements that cannot be reliably automated through rigid scripting or rule-based methods alone [122], [123]. Modern interfaces increasingly require contextual awareness [124], such as processing on-screen text, interpreting user intent, and recognizing visual cues. These demands reveal the limitations of existing automation frameworks and the need

for more flexible solutions capable of real-time adaptation and context-sensitive responses.
尽管取得了进展，传统GUI自动化方法在处理当代界面的复杂性和多样性方面仍显不足。如今的应用程序通常包含动态、自适应元素，单靠刚性的脚本或基于规则的方法难以实现可靠自动化[122]，[123]。现代界面越来越需要上下文感知能力[124]，例如处理屏幕文本、理解用户意图和识别视觉线索。这些需求暴露了现有自动化框架的局限性，凸显了需要更灵活的解决方案，能够实现实时适应和上下文敏感的响应。

LLMs offer a promising solution to these challenges. With their capacity to comprehend natural language, interpret context, and generate adaptive scripts, LLMs can enable more intelligent, versatile GUI automation [125]. Their ability to process complex instructions and learn from context allows them to bridge the gap between static, rule-based methods and the dynamic needs of contemporary GUIs [126]. By integrating LLMs with GUI agents, these systems gain the ability to generate scripts on-the-fly based on the current state of the interface, providing a level of adaptability and sophistication that traditional methods cannot achieve. The combination of LLMs and GUI agents paves the way for an advanced, user-centered automation paradigm, capable of responding flexibly to user requests and interacting seamlessly with complex, evolving interfaces.
大型语言模型（LLMs）为这些挑战提供了有前景的解决方案。凭借理解自然语言、解析上下文和生成自适应脚本的能力，LLMs能够实现更智能、多功能的GUI自动化[125]。它们处理复杂指令并从上下文中学习的能力，使其能够弥合静态基于规则方法与现代GUI动态需求之间的差距[126]。通过将LLMs与GUI代理集成，这些系统能够根据界面当前状态即时生成脚本，提供传统方法无法达到的适应性和复杂性。LLMs与GUI代理的结合开辟了先进的以用户为中心的自动化范式，能够灵活响应用户请求，并与复杂且不断演变的界面无缝交互。

7 4 EVOLUTION AND PROGRESSION OF LLM- BRAINED GUI AGENTS

8 4 LLM驱动GUI代理的演进与发展

"Rome wasn't built in a day." The development of LLM-brained GUI agents has been a gradual journey, grounded in decades of research and technical progress. Beginning with simple GUI testing scripts and rule-based automation frameworks, the field has evolved significantly through the integration of machine learning techniques, creating more intelligent and adaptive systems. The introduction of LLMs, especially multimodal models, has transformed GUI automation by enabling natural language interactions and fundamentally reshaping how users interact with software applications.
“罗马不是一天建成的。”LLM驱动的GUI代理的发展是一个渐进的过程，基于数十年的研究和技术进步。从简单的GUI测试脚本和基于规则的自动化框架起步，该领域通过引入机器学习技术实现了显著演变，打造出更智能、更具适应性的系统。尤其是多模态大型语言模型（LLMs）的引入，通过支持自然语言交互，彻底改变了GUI自动化，并根本性地重塑了用户与软件应用的交互方式。

As illustrated in Figure 3 prior to 2023 and the emergence of LLMs, work on GUI agents was limited in both scope and capability. Since then, the proliferation of LLM-based approaches has fostered numerous notable developments across platforms including web, mobile, and desktop environments. This surge is ongoing and continues to drive innovation in the field. This section takes you on a journey tracing the evolution of GUI agents, emphasizing key milestones that have brought the field to its present state.
如图3所示，在2023年及LLMs出现之前，GUI代理的研究在范围和能力上都较为有限。此后，基于LLM的方法迅速普及，推动了包括网页、移动和桌面环境在内的多个平台上的诸多重要进展。这一浪潮仍在持续，持续推动该领域的创新。本节将带您回顾GUI代理的发展历程，重点介绍推动该领域达到现状的关键里程碑。

8.1 4.1 Early Automation Systems

8.2 4.1 早期自动化系统

In the initial stages of GUI automation, researchers relied on random-based, rule-based, and script-based strategies. While foundational, these methods had notable limitations in terms of flexibility and adaptability.
在GUI自动化的初期阶段，研究人员依赖随机、基于规则和基于脚本的策略。尽管这些方法奠定了基础，但在灵活性和适应性方面存在显著局限。

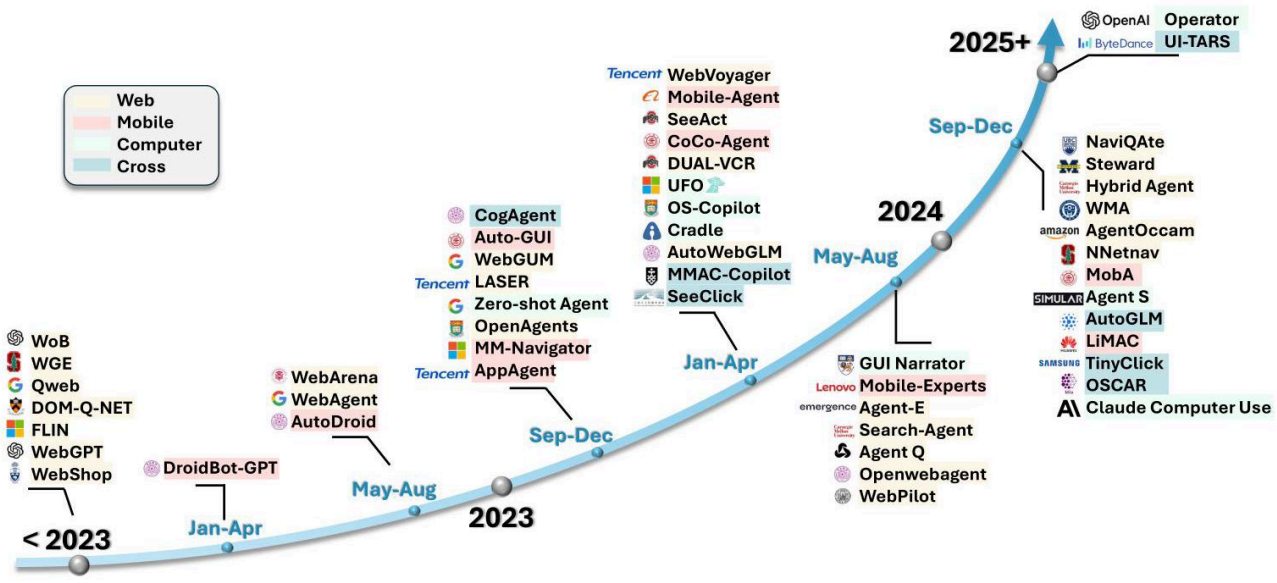

Fig. 3: An overview of GUI agents evolution over years.
图3：GUI代理多年来演进概览。

8.2.1 4.1.1 Random-Based Automation

8.2.2 4.1.1 基于随机的自动化

Random-based automation uses random sequences of actions within the GUI without relying on specific algorithms or structured models using monkey test [127]. This approach was widely used in GUI testing to uncover potential issues by exploring unpredictable input sequences [128]. While effective at identifying edge cases and bugs, random-based methods were often inefficient due to a high number of redundant or irrelevant trials.
基于随机的自动化在GUI中执行随机动作序列，不依赖特定算法或结构化模型，常用猴子测试(monkey test)[127]。该方法广泛应用于GUI测试，通过探索不可预测的输入序列发现潜在问题[128]。虽然在识别边缘情况和缺陷方面有效，但由于大量冗余或无关的尝试，随机方法效率较低。

8.2.3 4.1.2 Rule-Based Automation

8.2.4 4.1.2 基于规则的自动化

Rule-based automation applies predefined rules and logic to automate tasks. In 2001, Memon et al., 129 introduced a planning approach that generated GUI test cases by transforming initial states to goal states through a series of predefined operators. Hellmann et al., [4] (2011) demonstrated the potential of rule-based approaches in exploratory testing, enhancing bug detection. In the RPA domain, SmartRPA [130] (2020) used rule-based processing to automate routine tasks, illustrating the utility of rules for streamlining structured processes.
基于规则的自动化应用预定义的规则和逻辑来自动执行任务。2001年，梅蒙（Memon）等人 [129] 引入了一种规划方法，该方法通过一系列预定义的操作符将初始状态转换为目标状态，从而生成图形用户界面（GUI）测试用例。赫尔曼（Hellmann）等人 [4]（2011年）展示了基于规则的方法在探索性测试中的潜力，提高了错误检测能力。在机器人流程自动化（RPA）领域，SmartRPA [130]（2020年）使用基于规则的处理来自动执行日常任务，说明了规则在简化结构化流程方面的实用性。

8.2.5 4.1.3 Script-Based Automation

8.2.6 4.1.3 基于脚本的自动化

Script-based automation relies on detailed scripts to manage GUI interactions. Tools like jRapture [5] (2000) record and replay Java-based GUI sequences using Java binaries and the JVM, enabling consistent execution by precisely reproducing input sequences. Similarly, DART [131] (2003) automated the GUI testing lifecycle, from structural analysis to test case generation and execution, offering a comprehensive framework for regression testing.
基于脚本的自动化依赖详细的脚本来管理图形用户界面交互。像jRapture [5]（2000年）这样的工具使用Java二进制文件和Java虚拟机（JVM）记录和重放基于Java的图形用户界面序列，通过精确重现输入序列实现一致的执行。同样，DART [131]（2003年）自动化了图形用户界面测试的生命周期，从结构分析到测试用例的生成和执行，为回归测试提供了一个全面的框架。

8.2.7 4.1.4 Tools and Software

8.2.8 4.1.4 工具和软件

A range of software tools were developed for GUI testing and business process automation during this period. Mi-crosoft Power Automate [132] (2019) provides a low-code/no-code environment for creating automated workflows within Microsoft applications. Selenium [133] (2004) supports cross-browser web testing, while Appium [134 (2012) facilitates mobile UI automation. Commercial tools like TestComplete [135] (1999), Katalon Studio [136] (2015), and Ranorex [137] (2007) allow users to create automated tests with cross-platform capabilities.
在此期间，开发了一系列用于图形用户界面测试和业务流程自动化的软件工具。微软Power Automate [132]（2019年）为在微软应用程序中创建自动化工作流提供了一个低代码/无代码环境。Selenium [133]（2004年）支持跨浏览器的网页测试，而Appium [134]（2012年）便于进行移动用户界面自动化。像TestComplete [135]（1999年）、Katalon Studio [136]（2015年）和Ranorex [137]（2007年）等商业工具允许用户创建具有跨平台功能的自动化测试。

Although these early systems were effective for automating specific, predefined workflows, they lacked flexibility and required manual scripting or rule-based logic. Nonetheless, they established the foundations of GUI automation, upon which more intelligent systems were built.
尽管这些早期系统在自动化特定的、预定义的工作流方面很有效，但它们缺乏灵活性，需要手动编写脚本或基于规则的逻辑。尽管如此，它们为图形用户界面自动化奠定了基础，更智能的系统在此基础上得以构建。

8.3 4.2 The Shift Towards Intelligent Agents

8.4 4.2 向智能代理的转变

The incorporation of machine learning marked a major shift towards more adaptable and capable GUI agents. Early milestones in this phase included advancements in machine learning, natural language processing, computer vision, and reinforcement learning applied to GUI tasks.
机器学习的融入标志着向更具适应性和能力的图形用户界面代理的重大转变。这一阶段的早期里程碑包括将机器学习、自然语言处理、计算机视觉和强化学习应用于图形用户界面任务方面的进展。

8.4.1 4.2.1 Machine Learning and Computer Vision

8.4.2 4.2.1 机器学习和计算机视觉

RoScript [110] (2020) was a pioneering system that introduced a non-intrusive robotic testing system for touchscreen applications, expanding GUI automation to diverse platforms. AppFlow [138] (2018) used machine learning to recognize common screens and UI components, enabling modular testing for broad categories of applications. Progress in computer vision also enabled significant advances in GUI testing, with frameworks [117] (2010) automating visual interaction tasks. Humanoid [139] (2019) uses a deep neural network model trained on human interaction traces within the Android system to learn how users select actions based on an app's GUI. This model is then utilized to guide test input generation, resulting in improved coverage and more human-like interaction patterns during testing. Similarly, Deep GUI 140 (2021) applies deep learning techniques to filter out irrelevant parts of the screen, thereby enhancing black-box testing effectiveness in GUI testing by focusing only on significant elements. These approaches demonstrate the potential of deep learning to make GUI testing more efficient and intuitive by aligning it closely with actual user behavior.
RoScript [110]（2020年）是一个开创性的系统，它为触摸屏应用引入了一种非侵入式的机器人测试系统，将图形用户界面自动化扩展到了不同的平台。AppFlow [138]（2018年）使用机器学习来识别常见的屏幕和用户界面组件，为广泛类别的应用程序实现模块化测试。计算机视觉的进展也使图形用户界面测试取得了重大进展，一些框架 [117]（2010年）实现了视觉交互任务的自动化。Humanoid [139]（2019年）使用在安卓系统内的人类交互轨迹上训练的深度神经网络模型，来学习用户如何根据应用程序的图形用户界面选择操作。然后利用该模型来指导测试输入的生成，从而在测试期间提高覆盖率并实现更类似人类的交互模式。同样，Deep GUI [140]（2021年）应用深度学习技术过滤掉屏幕上无关的部分，从而通过仅关注重要元素来提高图形用户界面测试中黑盒测试的有效性。这些方法展示了深度学习通过紧密贴合实际用户行为，使图形用户界面测试更高效、更直观的潜力。

Widget detection, as demonstrated by White et al., 103 (2019), leverages computer vision to accurately identify UI elements, serving as a supporting technique that enables more intelligent and responsive UI automation. By detecting and categorizing interface components, this approach enhances the agent's ability to interact effectively with complex and dynamic GUIs [141].
正如怀特（White）等人 [103]（2019年）所展示的，小部件检测利用计算机视觉准确识别用户界面元素，作为一种支持技术，使更智能、响应更灵敏的用户界面自动化成为可能。通过检测和分类界面组件，这种方法增强了代理与复杂动态图形用户界面有效交互的能力 [141]。

8.4.3 4.2.2 Natural Language Processing

8.4.4 4.2.2 自然语言处理

Natural language processing capabilities introduced a new dimension to GUI automation. Systems like RUSS [142] (2021) and FLIN [143] (2020) allowed users to control GUIs through natural language commands, bridging human language and machine actions. Datasets, such as those in [144] (2020), further advanced the field by mapping natural language instructions to mobile UI actions, opening up broader applications in GUI control. However, these approaches are limited to handling simple natural commands and are not equipped to manage long-term tasks.
自然语言处理能力为图形用户界面自动化引入了一个新维度。像RUSS [142]（2021年）和FLIN [143]（2020年）这样的系统允许用户通过自然语言命令控制图形用户界面，架起了人类语言和机器操作之间的桥梁。像 [144]（2020年）中的数据集，通过将自然语言指令映射到移动用户界面操作，进一步推动了该领域的发展，为图形用户界面控制开辟了更广泛的应用。然而，这些方法仅限于处理简单的自然命令，无法处理长期任务。

8.4.5 4.2.3 Reinforcement Learning

8.4.6 4.2.3 强化学习

The development of environments like World of Bits (WoB) 145 (2017) enabled the training of web-based agents using reinforcement learning (RL). Workflow-guided exploration 146 (2018) improved RL efficiency and task performance. DQT [147] (2024) applied deep reinforcement learning to automate Android GUI testing by preserving widget structures and semantics, while AndroidEnv [148] (2021) offered realistic simulations for agent training on Android. WebShop [149] (2022) illustrated the potential for large-scale web interaction, underscoring the growing sophistication of RL-driven GUI automation.
像比特世界（World of Bits，WoB） [145]（2017年）这样的环境的开发，使得能够使用强化学习（RL）训练基于网页的代理。工作流引导的探索 [146]（2018年）提高了强化学习的效率和任务性能。DQT [147]（2024年）应用深度强化学习来自动化安卓图形用户界面测试，同时保留小部件的结构和语义，而AndroidEnv [148]（2021年）为在安卓系统上进行代理训练提供了逼真的模拟。WebShop [149]（2022年）展示了大规模网页交互的潜力，凸显了强化学习驱动的图形用户界面自动化日益提高的复杂性。

While these machine learning-based approaches were more adaptable than earlier rule-based systems [150], [151], they still struggled to generalize across diverse, unforeseen tasks. Their dependence on predefined workflows and limited adaptability required retraining or customization for new environments, and natural language control was still limited.
虽然这些基于机器学习的方法比早期基于规则的系统更具适应性[150][151]，但它们仍然难以在各种不可预见的任务中实现泛化。它们依赖于预定义的工作流程，适应性有限，在新环境中需要重新训练或定制，并且自然语言控制仍然受限。

8.5 4.3 The Advent of LLM-Brained GUI Agents

8.6 4.3 大语言模型驱动的图形用户界面（GUI）代理的出现

The introduction of LLMs, particularly multimodal models like GPT-40 [93] (2023), has radically transformed GUI automation by allowing intuitive interactions through natural language. Unlike previous approaches that required integration of separate modules, LLMs provide an end-to-end solution for GUI automation, offering advanced capabilities in natural language understanding, visual recognition, and reasoning.
大语言模型（LLM）的引入，特别是像GPT - 40 [93]（2023年）这样的多模态模型，通过允许通过自然语言进行直观交互，从根本上改变了GUI自动化。与之前需要集成独立模块的方法不同，大语言模型为GUI自动化提供了端到端的解决方案，在自然语言理解、视觉识别和推理方面具有先进的能力。

LLMs present several unique advantages for GUI agents, including natural language understanding, multimodal processing, planning, and generalization. These features make LLMs and GUI agents a powerful combination. While there were earlier explorations, 2023 marked a pivotal year for LLM-powered GUI agents, with significant developments across various platforms such as web, mobile, and desktop applications.
大语言模型为GUI代理带来了几个独特的优势，包括自然语言理解、多模态处理、规划和泛化能力。这些特性使大语言模型和GUI代理成为强大的组合。虽然早期有相关探索，但2023年是大语言模型驱动的GUI代理的关键一年，在网页、移动和桌面应用等各种平台上都有显著的发展。

8.6.1 4.3.1 Web Domain

8.6.2 4.3.1 网页领域

The initial application of LLMs in GUI automation was within the web domain, with early studies establishing benchmark datasets and environments [145], [149]. A key milestone was WebAgent [152] (2023), which, alongside WebGUM [153] (2023), pioneered real-world web navigation using LLMs. These advancements paved the way for further developments [17], [154], [155], utilizing more specialized LLMs to enhance web-based interactions.
大语言模型在GUI自动化中的最初应用是在网页领域，早期的研究建立了基准数据集和环境[145][149]。一个关键的里程碑是WebAgent [152]（2023年），它与WebGUM [153]（2023年）一起，开创了使用大语言模型进行现实世界网页导航的先河。这些进展为进一步的发展[17][154][155]铺平了道路，利用更专业的大语言模型来增强基于网页的交互。

8.6.3 4.3.2 Mobile Devices

8.6.4 4.3.2 移动设备

The integration of LLMs into mobile devices began with AutoDroid [156] (2023), which combined LLMs with domain-specific knowledge for smartphone automation. Additional contributions like MM-Navigator [157] (2023), AppAgent [18] (2023), and Mobile-Agent [158] (2023) enabled refined control over smartphone applications. Research has continued to improve accuracy for mobile GUI automation through model fine-tuning [159], [160] (2024).
大语言模型与移动设备的集成始于AutoDroid [156]（2023年），它将大语言模型与特定领域的知识相结合，用于智能手机自动化。其他贡献如MM - Navigator [157]（2023年）、AppAgent [18]（2023年）和Mobile - Agent [158]（2023年）实现了对智能手机应用的精细控制。通过模型微调[159][160]（2024年），移动GUI自动化的准确性研究仍在不断改进。

8.6.5 4.3.3 Computer Systems

8.6.6 4.3.3 计算机系统

For desktop applications, UFO [19] (2024) was one of the first systems to leverage GPT-4 with visual capabilities to fulfill user commands in Windows environments. Cradle [161] (2024) extended these capabilities to software applications and games, while Wu et al., [162] (2024) provided interaction across diverse desktop applications, including web browsers, code terminals, and multimedia tools.
对于桌面应用程序，UFO [19]（2024年）是最早利用具有视觉能力的GPT - 4在Windows环境中执行用户命令的系统之一。Cradle [161]（2024年）将这些能力扩展到软件应用程序和游戏，而Wu等人[162]（2024年）实现了跨多种桌面应用程序的交互，包括网页浏览器、代码终端和多媒体工具。

8.6.7 4.3.4 Industry Models

8.6.8 4.3.4 行业模型

In industry, the Claude 3.5 Sonnet model [163] (2024) introduced a "computer use" feature capable of interacting with desktop environments through UI operations [164]. This signifies the growing recognition of LLM-powered GUI agents as a valuable application in industry, with stakeholders increasingly investing in this technology.
在行业中，Claude 3.5 Sonnet模型[163]（2024年）引入了“计算机使用”功能，能够通过用户界面（UI）操作与桌面环境进行交互[164]。这表明大语言模型驱动的GUI代理作为一种有价值的应用在行业中得到了越来越多的认可，利益相关者也在加大对这项技术的投资。

OpenAI quickly followed up by releasing Operator 165] in 2025, a Computer-Using Agent (CUA) similar to Claude, achieving state-of-the-art performance across various benchmarks. This development underscores the industry's recognition of the value of GUI agents and its growing investment in the field. As interest continues to surge, GUI agent research and development are expected to become increasingly competitive, marking the beginning of a rapidly evolving landscape.
OpenAI迅速跟进，于2025年发布了Operator [165]，这是一个类似于Claude的计算机使用代理（CUA），在各种基准测试中取得了最先进的性能。这一发展凸显了行业对GUI代理价值的认可以及对该领域不断增加的投资。随着兴趣的持续高涨，GUI代理的研发预计将变得越来越具有竞争力，标志着一个快速发展的局面的开始。

Undoubtedly, LLMs have introduced new paradigms and increased the intelligence of GUI agents in ways that were previously unattainable. As the field continues to evolve, we anticipate a wave of commercialization, leading to transformative changes in user interaction with GUI applications.
毫无疑问，大语言模型引入了新的范式，并以以前无法实现的方式提高了GUI代理的智能水平。随着该领域的不断发展，我们预计将迎来一波商业化浪潮，从而给用户与GUI应用程序的交互带来变革性的变化。

8.7 4.4 GUI Agent vs. API-Based Agent

8.8 4.4 GUI代理与基于API的代理

In the field of LLM-powered agents operating within digital environments, the action space can be broadly categorized into two types:
在大语言模型驱动的、在数字环境中运行的代理领域，动作空间大致可分为两类：

1. GUI Agents, which primarily rely on GUI operations (e.g., clicks, keystrokes) to complete tasks.
2. GUI代理，主要依赖GUI操作（如点击、按键）来完成任务。

2. API-Based Agents, which utilize system or application-native APIs to fulfill objectives. We show the principle of both agent types in Figure 4. Each type has distinct advantages, and a deeper understanding of these approaches is critical for designing effective agents.
3. 基于API的代理，利用系统或应用程序原生API来实现目标。我们在图4中展示了这两种代理类型的原理。每种类型都有其独特优势，深入理解这些方法对于设计高效代理至关重要。

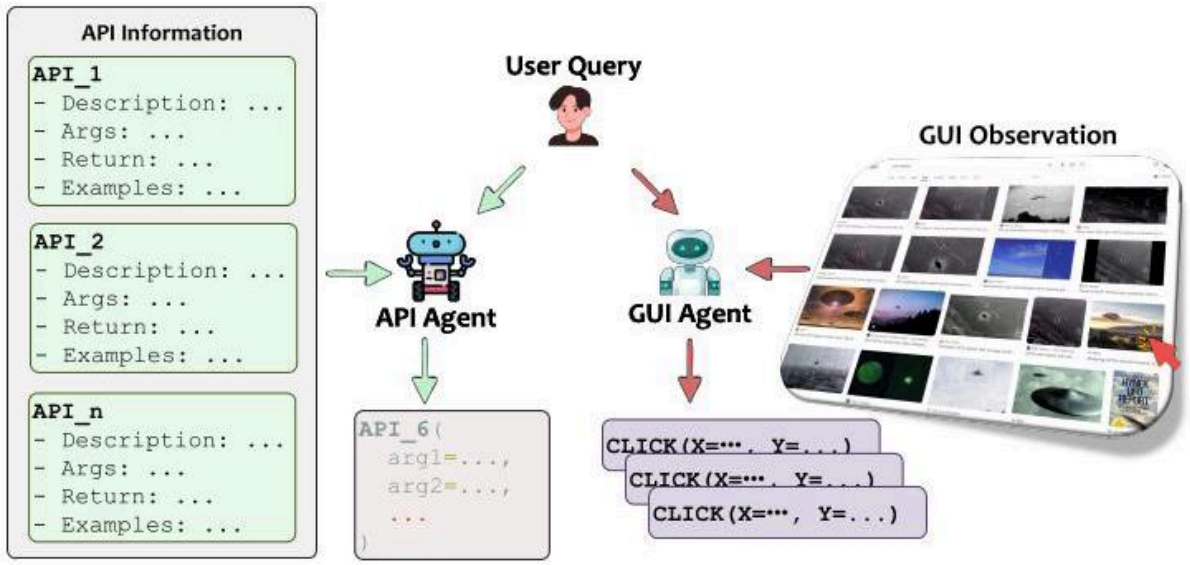

Fig. 4: The comparison between API agent vs. GUI agent.
图4：API代理与GUI代理的比较。

GUI operations provide a universal control interface that can operate across diverse applications using the same action primitives. This makes GUI agents highly generalizable, as they can interact with a wide range of software environments without requiring application-specific adaptations. However, GUI-based interactions are inherently more complex; even simple tasks may require multiple sequential steps, which can increase both the decision-making cost for the agent and the computational resources required for long-term, multi-step workflows. Another key aspect is the transparency of actions in GUI agents. Since GUI agents interact with applications in the same way a human would, by clicking, typing, and navigating through the interface, their actions are inherently more observable and interpretable to users. This transparency fosters better trust and comprehension in agent-computer interactions.
GUI操作提供了一个通用的控制接口，可以使用相同的基本动作跨多种应用程序进行操作。这使得GUI代理具有高度的通用性，能够与广泛的软件环境交互，而无需针对特定应用进行适配。然而，基于GUI的交互本质上更为复杂；即使是简单任务也可能需要多个连续步骤，这会增加代理的决策成本和长期多步骤工作流所需的计算资源。另一个关键方面是GUI代理动作的透明性。由于GUI代理通过点击、输入和界面导航等方式与应用交互，类似于人类操作，其行为对用户来说更易观察和理解。这种透明性有助于增强用户对代理-计算机交互的信任和理解。

In contrast, API-based agents offer a more efficient and direct approach to task completion. By leveraging native APIs, tasks can often be fulfilled with a single, precise call, significantly reducing execution time and complexity. However, these native APIs are often private or restricted to specific applications, limiting accessibility and generalizability. This makes API-based agents less versatile in scenarios where API access is unavailable or insufficient. In addition, API-based agents operate behind the scenes, executing tasks through direct system calls, which, while often more efficient and reliable, can make their operations less visible and harder to debug for end users.
相比之下，基于API的代理提供了一种更高效、直接的任务完成方式。通过利用原生API，任务通常可以通过一次精确调用完成，显著减少执行时间和复杂度。然而，这些原生API往往是私有的或仅限于特定应用，限制了其可访问性和通用性。这使得基于API的代理在API不可用或不足的场景中适用性较差。此外，基于API的代理在后台通过直接系统调用执行任务，虽然通常更高效且可靠，但其操作对终端用户来说较不透明，调试难度较大。

The most effective digital agents are likely to operate in a hybrid manner, combining the strengths of both approaches. Such agents can utilize GUI operations to achieve broad compatibility across software while exploiting native APIs where available to maximize efficiency and effectiveness. These hybrid agents strike a balance between generalization and task optimization, making them a critical focus area in this survey. For a more comprehensive comparison between GUI agents and API agents, please refer to [166].
最有效的数字代理很可能采用混合方式，结合两种方法的优势。这类代理可以利用GUI操作实现广泛的软件兼容性，同时在可用时利用原生API以最大化效率和效果。这些混合代理在通用性和任务优化之间取得平衡，是本综述的关键关注点。有关GUI代理与API代理的更全面比较，请参见文献[166]。

9 5 LLM-BRAINED GUI AGENTS: FOUNDATIONS AND DESIGN

10 5 基于大型语言模型的GUI代理：基础与设计

In essence, LLM-brained GUI agents are designed to process user instructions or requests given in natural language, interpret the current state of the GUI through screenshots or UI element trees, and execute actions that simulate human interaction across various software interfaces [19]. These agents harness the sophisticated natural language understanding, reasoning, and generative capabilities of LLMs to accurately comprehend user intent, assess the GUI context, and autonomously engage with applications across diverse environments, thereby enabling the completion of complex, multi-step tasks. This integration allows them to seamlessly interpret and respond to user requests, bringing adaptability and intelligence to GUI automation.
本质上，基于大型语言模型（LLM）的GUI代理旨在处理以自然语言给出的用户指令或请求，通过截图或UI元素树解读当前GUI状态，并执行模拟人类交互的操作，跨多种软件界面完成任务[19]。这些代理利用LLM强大的自然语言理解、推理和生成能力，准确把握用户意图，评估GUI上下文，并自主与各种环境中的应用交互，从而实现复杂多步骤任务的完成。这种集成使其能够无缝解读和响应用户请求，为GUI自动化带来适应性和智能化。

As a specialized type of LLM agent, most current GUI agents adopt a similar foundational framework, integrating core components such as planning, memory, tool usage, and advanced enhancements like multi-agent collaboration, among others [47]. However, each component must be tailored to meet the specific objectives of GUI agents to ensure adaptability and functionality across various application environments.
作为LLM代理的专门类型，目前大多数GUI代理采用类似的基础框架，集成规划、记忆、工具使用及多代理协作等高级增强组件[47]。然而，每个组件必须针对GUI代理的具体目标进行定制，以确保其在各种应用环境中的适应性和功能性。

In the following sections, we provide an in-depth overview of each component, offering a practical guide and tutorial on building an LLM-powered GUI agent from the ground up. This comprehensive breakdown serves as a cookbook for creating effective and intelligent GUI automation systems that leverage the capabilities of LLMs.
在接下来的章节中，我们将深入概述每个组件，提供构建基于LLM的GUI代理的实用指南和教程。此全面解析可视为创建高效智能GUI自动化系统的配方，充分利用LLM的能力。

10.1 5.1 Architecture and Workflow In a Nutshell

10.2 5.1 架构与工作流程概述

In Figure 5, we present the architecture of an LLM-brained GUI agent, showcasing the sequence of operations from user input to task completion. The architecture comprises several integrated components, each contributing to the agent's ability to interpret and execute tasks based on user-provided natural language instructions. Upon receiving a user request, the agent follows a systematic workflow that includes environment perception, prompt engineering, model inference, action execution, and continuous memory utilization until the task is fully completed.
在图5中，我们展示了基于LLM的GUI代理架构，呈现从用户输入到任务完成的操作序列。该架构包含多个集成组件，各自助力代理根据用户提供的自然语言指令解读并执行任务。代理在接收用户请求后，遵循环境感知、提示工程、模型推理、动作执行及持续记忆利用的系统化工作流程，直至任务完成。

In general, it consists of the following components:
总体而言，其组成包括以下部分：

1. Operating Environment: The environment defines the operational context for the agent, encompassing platforms such as mobile devices, web browsers, and desktop operating systems like Windows. To interact meaningfully, the agent perceives the environment's current state through screenshots, widget trees, or other methods of capturing UI structure [167]. It continuously monitors feedback on each action's impact, adjusting its strategy in real time to ensure effective task progression.
2. 操作环境：环境定义了代理的运行上下文，涵盖移动设备、网页浏览器及Windows等桌面操作系统。为实现有效交互，代理通过截图、小部件树或其他UI结构捕获方法感知环境当前状态[167]。它持续监控每个动作的反馈，实时调整策略，确保任务有效推进。

2. Prompt Engineering: Following environment perception, the agent constructs a detailed prompt to guide the LLM's inference [168]. This prompt incorporates user instructions, processed visual data (e.g., screenshots), UI element layouts, properties, and any additional context relevant to the task. This structured input maximizes the LLM's ability to generate coherent, context-aware responses aligned with the current GUI state.
3. 提示工程：在环境感知之后，代理构建详细提示以引导LLM推理[168]。该提示包含用户指令、处理后的视觉数据（如截图）、UI元素布局、属性及与任务相关的其他上下文。此结构化输入最大化LLM生成连贯且符合当前GUI状态的上下文响应的能力。

3. Model Inference: The constructed prompt is passed to a LLM, the agent's inference core, which produces a sequence of plans, actions and insights required to fulfill the user's request. This model may be a general-purpose LLM or a specialized model fine-tuned with GUI-specific data, enabling a more nuanced understanding of GUI interactions, user flows, and task requirements.
4. 模型推理：构建的提示被传递给大型语言模型（LLM），即代理的推理核心，生成完成用户请求所需的一系列计划、动作和洞见。该模

型可以是通用大型语言模型，也可以是经过GUI特定数据微调的专用模型，从而实现对GUI交互、用户流程和任务需求的更细致理解。

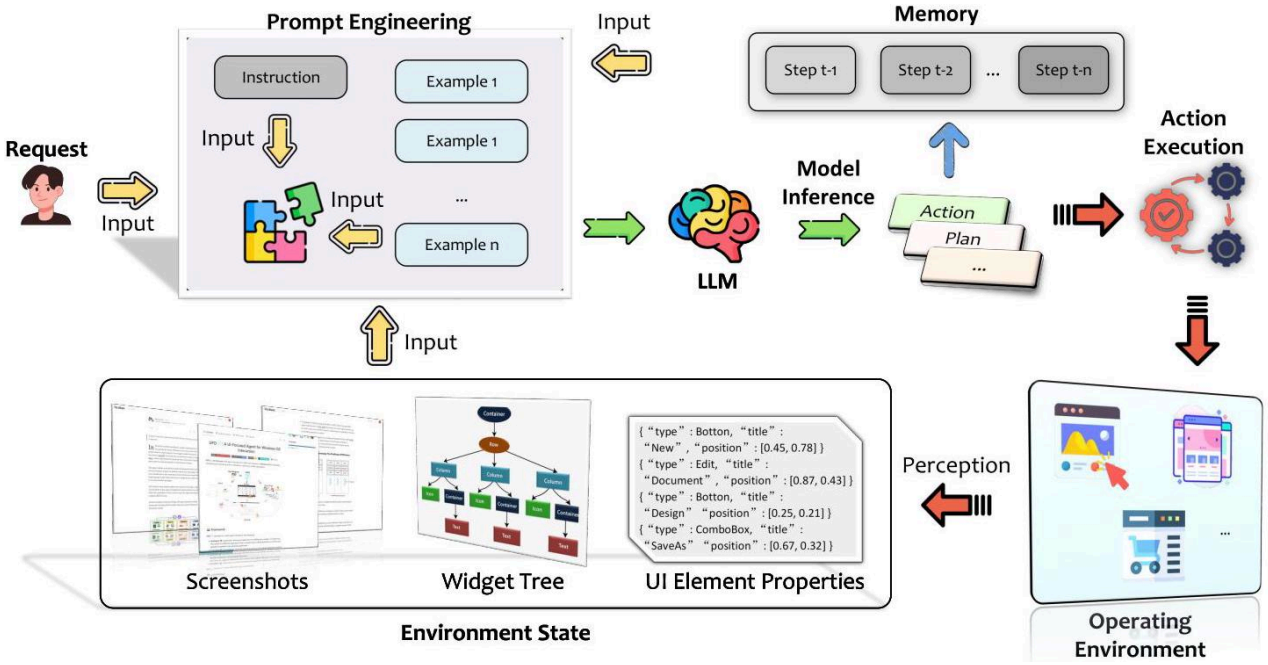

Fig. 5: An overview of the architecture and workflow of a basic LLM-powered GUI agent.
图5：基于大型语言模型的GUI代理的架构与工作流程概览。

4. Actions Execution: Based on the model's inference results, the agent identifies specific actions (such as mouse clicks, keyboard inputs, touchscreen gestures, or API calls) required for task execution [13]. An executor within the agent translates these high-level instructions into actionable commands that impact the GUI directly, effectively simulating human-like interactions across diverse applications and devices.
5. 动作执行：基于模型的推理结果，代理识别执行任务所需的具体动作（如鼠标点击、键盘输入、触摸手势或API调用）[13]。代理内的执行器将这些高级指令转化为直接影响GUI的可操作命令，有效模拟跨多种应用和设备的人类交互行为。

5. Memory: For multi-step tasks, the agent maintains an internal memory to track prior actions, task progress, and environment states [54]. This memory ensures coherence throughout complex workflows, as the agent can reference previous steps and adapt its actions accordingly. An external memory module may also be incorporated to enable continuous learning, access external knowledge, and enhance adaptation to new environments or requirements.
6. 记忆：对于多步骤任务，代理维护内部记忆以跟踪先前动作、任务进展和环境状态[54]。该记忆确保复杂工作流的连贯性，使代理能够参考之前的步骤并相应调整动作。还可引入外部记忆模块，实现持续学习、访问外部知识，并增强对新环境或需求的适应能力。

By iteratively traversing these stages and assembling the foundational components, the LLM-powered GUI agent operates intelligently, seamlessly adapting across various software interfaces and bridging the gap between language-based instruction and concrete action. Each component is critical to the agent's robustness, responsiveness, and capability to handle complex tasks in dynamic environments. In the following subsections, we detail the design and core techniques underlying each of these components, providing a comprehensive guide for constructing LLM-powered GUI agents from the ground up.
通过迭代遍历这些阶段并组装基础组件，基于大型语言模型的GUI代理能够智能运行，顺畅适应各种软件界面，弥合基于语言的指令与具体动作之间的鸿沟。每个组件对代理的稳健性、响应性及处理动态环境中复杂任务的能力至关重要。以下小节详细介绍这些组件的设计与核心技术，为从零构建基于大型语言模型的GUI代理提供全面指导。

10.3 5.2 Operating Environment

10.4 5.2 操作环境

The operating environment for LLM-powered GUI agents encompasses various platforms, such as mobile, web, and desktop operating systems, where these agents can interact with graphical interfaces. Each platform has distinct characteristics that impact the way GUI agents perceive, interpret, and act within it. Examples of GUIs from each platform are shown in Figure 6 This section details the nuances of each platform, the ways agents gather environmental information, and the challenges they face in adapting to diverse operating environments.
基于大型语言模型的GUI代理的操作环境涵盖多种平台，如移动端、网页端和桌面操作系统，这些代理可与图形界面交互。每个平台具有不

同特性，影响GUI代理的感知、理解和行动方式。图6展示了各平台GUI示例。本节详述各平台的细微差别、代理收集环境信息的方式及其适应多样操作环境所面临的挑战。

Fig. 6: Examples of GUIs from web, mobile and computer platforms.
图6：来自网页、移动和计算机平台的GUI示例。

10.4.1 5.2.1 Platform

10.4.2 5.2.1 平台

The operating environment for LLM-powered GUI agents encompasses various platforms, such as mobile, web, and desktop operating systems, where these agents can interact with graphical interfaces. Each platform has distinct characteristics that impact the way GUI agents perceive, interpret, and act within it. Examples of GUIs from each platform are shown in Figure 6. This section details the nuances of each platform, the ways agents gather environmental information, and the challenges they face in adapting to diverse operating environments.
基于大型语言模型的GUI代理的操作环境涵盖多种平台，如移动端、网页端和桌面操作系统，这些代理可与图形界面交互。每个平台具有不同特性，影响GUI代理的感知、理解和行动方式。图6展示了各平台GUI示例。本节详述各平台的细微差别、代理收集环境信息的方式及其适应多样操作环境所面临的挑战。

1. Mobile Platforms: Mobile devices operate within constrained screen real estate, rely heavily on touch interactions [170], and offer varied app architectures (e.g., native vs. hybrid apps). Mobile platforms often use accessibility frameworks, such as Android's Accessibility AP | 171 and iOS's VoiceOver Accessibility Inspector [5] to expose structured information about UI elements. However, GUI agents must handle additional complexities in mobile environments, such as gesture recognition [169], app navigation [172], and platform-specific constraints (e.g., security and privacy permissions) [173], [174].
2. 移动平台：移动设备屏幕空间有限，严重依赖触控交互[170]，且应用架构多样（如原生应用与混合应用）。移动平台通常使用辅助功能框架，如Android的Accessibility API[171]和iOS的VoiceOver辅助检查器[5]，以暴露UI元素的结构化信息。然而，GUI代理在移动环境中还需处理额外复杂性，如手势识别[169]、应用导航[172]及平台特有限制（如安全和隐私权限）[173]，[174]。

TABLE 3: Summary of platform-specific challenges, action spaces, and typical tasks for Web, Mobile, and Computer GUI environments.
表3：Web、移动和计算机GUI环境中平台特定挑战、动作空间及典型任务的总结。

Platform	Typical GUI Challenges	Action Space	Representative Tasks
Mobile	- Constrained screen real estate - Heavy reliance on touch and gesture recognition [169] - App architectures (native vs. hybrid) - Accessibility frameworks (e.g., Android's Accessibility API, iOS VoiceOver) - Platform-specific constraints (permissions, security, privacy)	- Tap, swipe, pinch, and other touch gestures - Virtual keyboard input - In-app navigation (menus, tabs) - Accessing hardware features (camera, GPS)	- App-based login and form filling - Messaging, social media posting - Location-based services and map interactions - Handling push notifications and permission dialogs
Web	- Dynamic and responsive layouts - Asynchronous updates (AJAX, fetch APIs) - HTML/DOM-based structures - Cross-browser inconsistencies	- Click, hover, scroll - DOM-based form filling - Link navigation and element inspection - JavaScript event triggering	- Form completion (registrations, checkouts) - Data extraction/web scraping - Searching and filtering (e.g., e-commerce) - Multi-step web navigation (redirects, pop-ups)
Computer	- Full-fledged OS-level interfaces - Multi-window operations system-level shortcuts - Automation APIs (e.g., Windows UI Automation [32] - Frequent software updates requiring adaptation - Complex, multi-layered software suites	- Mouse click, drag-and-drop - Keyboard shortcuts and text input - Menu navigation, toolbars - Access to multiple application windows	- File management and system settings - Productivity software usage (office suites, IDEs) - Installing/uninstalling applications - Coordinating multi-application workflows

平台	典型的图形用户界面挑战	操作空间	代表性任务
移动端	- 屏幕空间受限 - 强烈依赖触摸和手势识别 [169] - 应用架构（原生与混合） - 辅助功能框架（如 Android 的 Accessibility API，iOS VoiceOver） - 平台特定限制（权限、安全、隐私）	- 轻触、滑动、捏合等触摸手势 - 虚拟键盘输入 - 应用内导航（菜单、标签页） - 访问硬件功能（摄像头、GPS）	- 基于应用的登录和表单填写 - 消息发送、社交媒体发布 - 基于位置的服务和地图交互 - 处理推送通知和权限对话框
网页端	- 动态响应式布局 - 异步更新（AJAX，fetch API） - 基于 HTML/DOM 的结构 - 跨浏览器不一致性	- 点击、悬停、滚动 - 基于 DOM 的表单填写 - 链接导航和元素检查 - JavaScript 事件触发	- 表单填写（注册、结账） - 数据提取/网页抓取 - 搜索和筛选（如电子商务） - 多步骤网页导航（重定向、弹窗）
计算机端	- 完整的操作系统级界面 - 多窗口操作及系统级快捷键 - 自动化 API（如 Windows UI Automation [32]） - 频繁的软件更新需适应 - 复杂多层的软件套件	- 鼠标点击、拖放 - 键盘快捷键和文本输入 - 菜单导航、工具栏 - 访问多个应用窗口	- 文件管理和系统设置 - 办公软件使用（办公套件、集成开发环境） - 安装/卸载应用程序 - 协调多应用工作流程

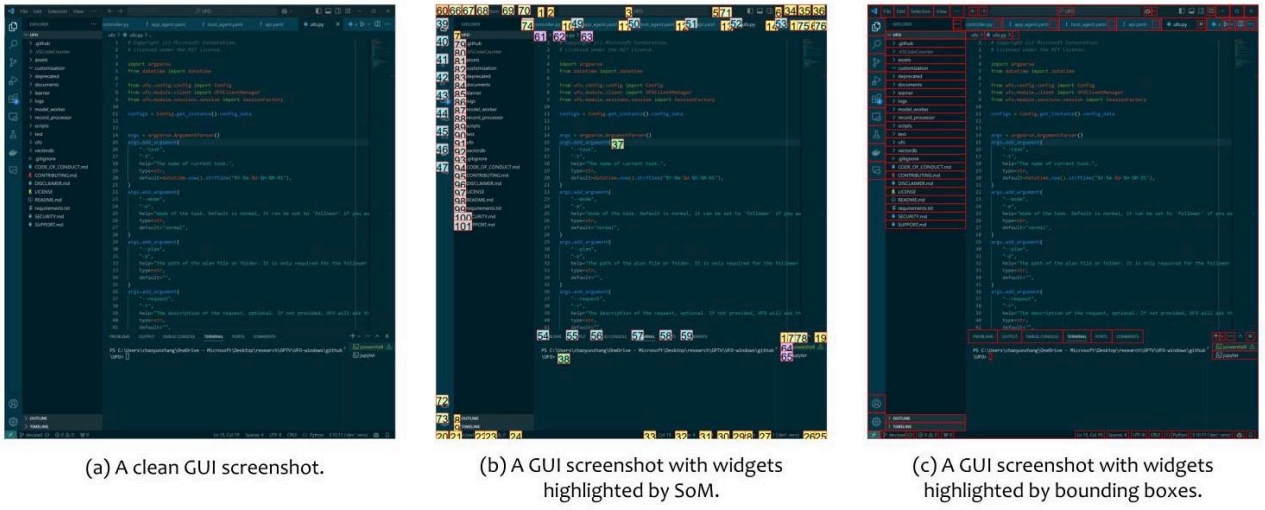

Fig. 7: Examples of different variants of VS Code GUI screenshots.
图7：VS Code图形用户界面（GUI）截图不同变体示例。

2. Web Platforms: Web applications provide a relatively standardized interface, typically accessible through Hy-pertext Markup Language (HTML) and Document Object Model (DOM) structures [175], [176]. GUI agents can leverage HTML attributes, such as

element ID, class, and tag, to identify interactive components. Web environments also present dynamic content, responsive layouts, and asynchronous updates (e.g., AJAX requests) [177], requiring agents to continuously assess the DOM and adapt their actions to changing interface elements.

3. 网络平台：网络应用程序提供了相对标准化的界面，通常可通过超文本标记语言（HTML）和文档对象模型（DOM）结构访问 [175]、[176]。GUI智能体可以利用HTML属性（如元素ID、类和标签）来识别交互组件。网络环境还存在动态内容、响应式布局和异步更新（如AJAX请求） [177]，这要求智能体持续评估DOM并根据不断变化的界面元素调整其操作。

---

4. https://developer.android.com/reference/android/
5. https://developer.android.com/reference/android/

accessibilityservice/AccessibilityService
accessibilityservice/AccessibilityService

5. https://developer.apple.com/documentation/accessibility/
6. https://developer.apple.com/documentation/accessibility/

accessibility-inspector
accessibility-inspector

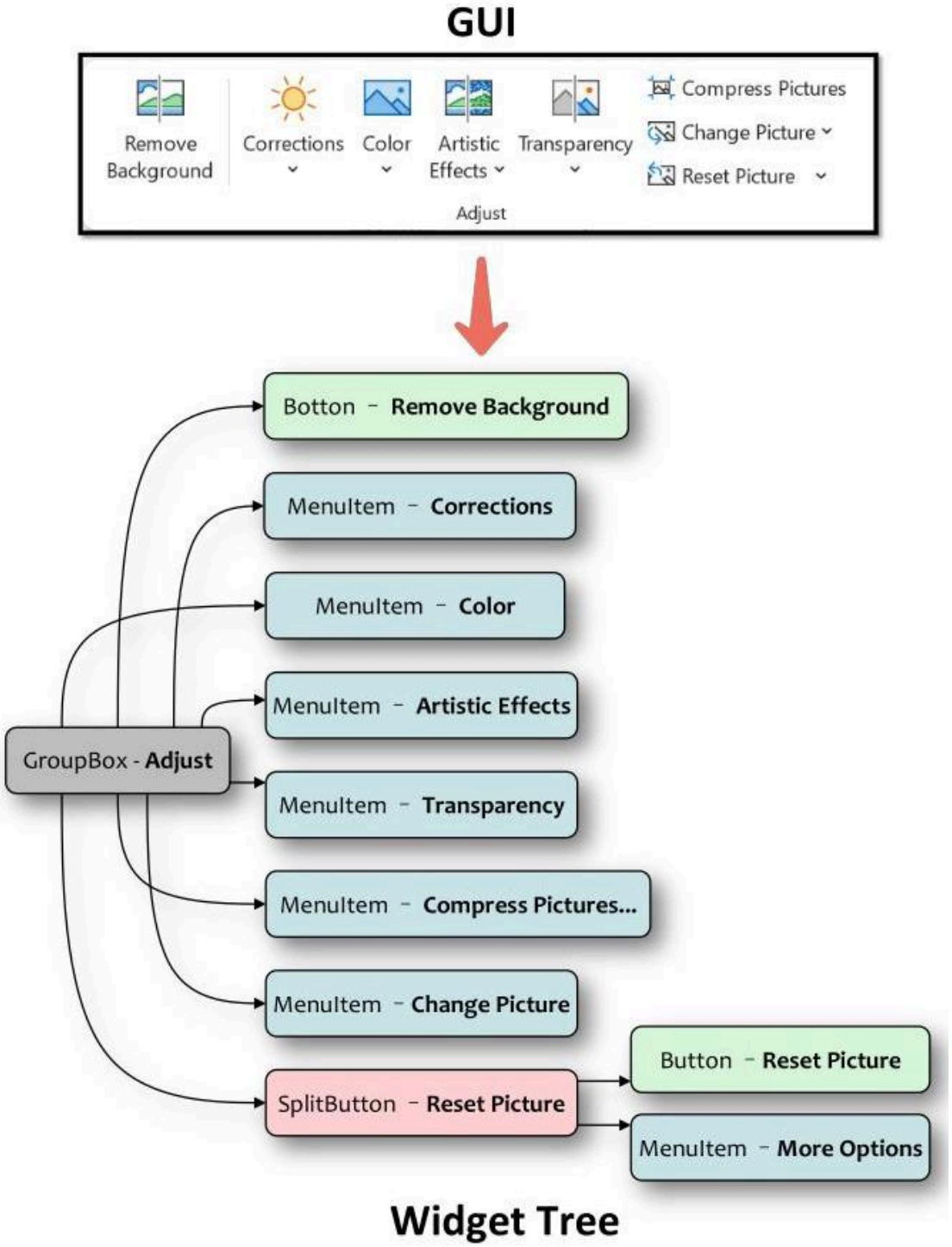

Fig. 8: An example of a GUI and its widget tree.
图8：一个GUI及其小部件树的示例。

3. Computer Platforms: Computer OS platforms, such as Windows, offer full control over GUI interactions. Agents can utilize system-level automation APIs, such as Windows UI Automation [32], to obtain comprehensive UI element data, including type, label, position, and bounding box. These platforms often support a broader set of interaction types, mouse, keyboard, and complex multi-window operations. These enable GUI agents to execute intricate workflows. However, these systems also require sophisticated adaptation for diverse applications, ranging from simple UIs to complex, multi-layered software suites.
4. 计算机平台：计算机操作系统（OS）平台（如Windows）可对GUI交互进行完全控制。智能体可以利用系统级自动化应用程序编程接口（API），如Windows UI自动化 [32]，来获取全面的UI元素数据，包括类型、标签、位置和边界框。这些平台通常支持更广泛的交互类型，如鼠标、键盘和复杂的多窗口操作。这使得GUI智能体能够执行复杂的工作流程。然而，这些系统还需要针对从简单UI到复杂的多层软件套件等各种应用进行复杂的适配。

In summary, the diversity of platforms, spanning mobile, web, and desktop environments, enable GUI agents to deliver broad automation capabilities, making them a generalized solution adaptable across a unified framework. However, each platform presents unique characteristics and constraints at both the system and application levels, necessitating a tailored approach for effective integration. By considering these platform-specific features, GUI agents can be optimized to address the distinctive requirements of each environment, thus enhancing their adaptability and reliability in varied automation scenarios.
综上所述，涵盖移动、网络和桌面环境的平台多样性使GUI智能体能够提供广泛的自动化能力，使其成为一种可在统一框架内适应不同场景的通用解决方案。然而，每个平台在系统和应用层面都呈现出独特的特征和限制，因此需要采用量身定制的方法进行有效集成。通过考虑这些特定于平台的特性，可以对GUI智能体进行优化，以满足每个环境的独特需求，从而提高它们在各种自动化场景中的适应性和可靠性。

10.4.3 5.2.2 Environment State Perception

10.4.4 5.2.2 环境状态感知

Accurately perceiving the current state of the environment is essential for LLM-powered GUI agents, as it directly informs their decision-making and action-planning processes. This perception is enabled by gathering a combination of structured data, such as widget trees, and unstructured data, like screenshots, to capture a complete representation of the interface and its components. In Table 4, we outline key toolkits available for collecting GUI environment data across various platforms, and below we discuss their roles in detail:
准确感知当前环境状态对于由大语言模型（LLM）驱动的GUI智能体至关重要，因为这直接影响它们的决策和行动计划制定过程。这种感知通过收集结构化数据（如小部件树）和非结构化数据（如截图）的组合来实现，以全面呈现界面及其组件。在表4中，我们列出了可用于在各种平台上收集GUI环境数据的关键工具包，下面我们将详细讨论它们的作用：

Widget	Widget Name	Position	Attributes
 Remove Background	Button - 'Remove Background'	L-3810, T128, R-3708, B243	title='Remove Background'; auto_id='PictureBackgroundRemoval'; control_type='Button
 Corrections	MenuItem - 'Corrections'	L-3689, T128, R-3592, B243	title='Corrections'; auto_id='PictureCorrectionsMenu'; control_type='MenuItem'
	MenuItem - 'Color'	L-3589, T128, R-3527, B243	title='Color'; auto_id='PictureColorMenu'; control type='MenuItem'
 Artistic Effects -	MenuItem - 'Artistic Effects'	L-3524, T128, R-3448, B243	title='Artistic Effects'; auto id='PictureArtisticEffectsGallery'; control_type='MenuItem'
	MenuItem - 'Transparency'	L-3445, T128, R-3336, B243	title='Transparency'; auto_id='PictureTransparencyGallery'; control_type='MenuItem'
E. Compress Pictures	Button - 'Compress Pictures...'	L-3333, T128, R-3138, B164	title='Compress Pictures...'; auto id='PicturesCompress'; control type='Button'
	MenuItem - 'Change Picture'	L-3333, T167, R-3149, B203	title='Change Picture'; auto id='PictureChangeMenu'; control_type='MenuItem'
	SplitButton - 'Reset Picture'	L-3333, T206, R-3160, B242	title='Reset Picture'; control type='SplitButton'

控件	控件名称	位置	属性
 删除背景	按钮 - “删除背景”	L-3810, T128, R-3708, B243	title='删除背景'; auto_id='PictureBackgroundRemoval'; control_type='Button
 校正	菜单项 - “校正”	L-3689, T128, R-3592, B243	title='校正'; auto_id='PictureCorrectionsMenu'; control_type='MenuItem'
	菜单项 - “颜色”	L-3589, T128, R-3527, B243	title='颜色'; auto_id='PictureColorMenu'; control type='MenuItem'
 艺术效果 -	菜单项 - “艺术效果”	L-3524, T128, R-3448, B243	title='艺术效果'; auto id='PictureArtisticEffectsGallery'; control_type='MenuItem'
	菜单项 - “透明度”	L-3445, T128, R-3336, B243	title='透明度'; auto_id='PictureTransparencyGallery'; control_type='MenuItem'
压缩图片	按钮 - “压缩图片...”	L-3333, T128, R-3138, B164	title='压缩图片...'; auto id='PicturesCompress'; control type='Button'
	菜单项 - “更改图片”	L-3333, T167, R-3149, B203	title='更改图片'; auto id='PictureChangeMenu'; control_type='MenuItem'
	分割按钮 - “重置图片”	L-3333, T206, R-3160, B242	title='重置图片'; control type='SplitButton'

Fig. 9: Examples of UI element properties in the PowerPoint application for GUI Agent interaction.
图9：PowerPoint应用中用于GUI代理交互的界面元素属性示例。

1. GUI Screenshots: Screenshots provide a visual snapshot of the application, capturing the entire state of the GUI at a given moment. They offer agents a reference for layout, design, and visual content, which is crucial when structural details about UI elements are either limited or unavailable. Visual elements like icons, images, and other graphical cues that may hold important context can be analyzed directly from screenshots. Many platforms have built-in tools to capture screenshots (e.g., Windows Snipping Too 7 macOS Screenshot Utility and Android's MediaProjection AP 9, and screenshots can be enhanced with additional annotations, such as Set-of-Mark (SoM) highlights [178] or bounding boxes 179 around key UI components, to streamline agent decisions. Figure 7 illustrates various screenshots of the VS Code GUI, including a clean version, as well as ones with SoM and bounding boxes that highlight actionable components, helping the agent focus on the most critical areas of the interface.
2. GUI截图：截图提供了应用程序的视觉快照，捕捉了特定时刻GUI的整体状态。它们为代理提供了布局、设计和视觉内容的参考，这在UI元素的结构细节有限或不可用时尤为重要。图标、图像及其他可能包含重要上下文的图形提示等视觉元素，可以直接从截图中进行分析。许多平台内置了截图工具（例如Windows的截图工具、macOS截图实用程序和Android的MediaProjection API），截图还可以通过附加注释进行增强，如关键UI组件周围的标记集（Set-of-Mark，SoM）高亮[178]或边界框[179]，以简化代理的决策过程。图7展示了VS Code GUI的多种截图，包括干净版本，以及带有SoM和边界框突出显示可操作组件的版本，帮助代理聚焦界面中最关键的区域。

2. Widget Trees: Widget trees present a hierarchical view of interface elements, providing structured data about the layout and relationships between components [180]. We show an example of a GUI and its widget tree in Figure 8. By accessing the widget tree, agents can identify attributes such as element type, label, role, and relationships within the interface, all of which are essential for contextual understanding. Tools like Windows UI Automation and macOS's Accessibility AP [10] provide structured views for desktop applications, while Android's Accessibility API and HTML DOM structures serve mobile and web platforms, respectively. This hierarchical data is indispensable for agents to map out logical interactions and make informed choices based on the UI structure.
3. 小部件树：小部件树展示了界面元素的层级视图，提供关于布局和组件之间关系的结构化数据[180]。我们在图8中展示了一个GUI及其小部件树的示例。通过访问小部件树，代理可以识别元素类型、标签、角色及界面内的关系等属性，这些都是实现上下文理解的关键。Windows UI Automation和macOS的Accessibility API提供了桌面应用的结构化视图，而Android的Accessibility API和HTML DOM结构则分别服务于移动和网页平台。这种层级数据对于代理绘制逻辑交互图谱并基于UI结构做出明智选择至关重要。

---

6. https://learn.microsoft.com/en-us/dotnet/framework/ui-automation/ ui-automation-overview
7. https://learn.microsoft.com/zh-cn/dotnet/framework/ui-automation/ui-automation-overview

7. https://support.microsoft.com/en-us/windows/
8. https://support.microsoft.com/en-us/windows/

use-snipping-tool-to-capture%2Dscreenshots%2D00246869%
使用截图工具捕获屏幕截图

2D1843%2D655f%2Df220%2D97299b865f6b
2D1843%2D655f%2Df220%2D97299b865f6b

8. https://support.apple.com/guide/mac-help/ take-a-screenshot-mh26782/mac
9. https://support.apple.com/guide/mac-help/ 截取屏幕截图-mh26782/mac

9. https://developer.android.com/reference/android/media/projection/ MediaProjection
10. https://developer.android.com/reference/android/media/projection/ MediaProjection（媒体投影）

TABLE 4: Key toolkits for collecting GUI environment data.
表4：收集图形用户界面环境数据的关键工具包。

Tool	Platform	Environment	Accessible Information	Highlight	Link
Selenium	Web	Browser platform) (Cross-	DOM elements, HTML structure, CSS properties	Extensive browser support and automation capabilities	https://www.selenium.dev/
Puppeteer	Web	Browser Firefox) (Chrome,	DOM elements, HTML/CSS. network requests	Headless browser automa tion with rich API	https://pptr.dev/
Playwright	Web	Browser platform) (Cross-	DOM elements, HTML/CSS. network interactions	Multi-browser support with automation and testing capabilities	https://playwright.dev/
TestCafe	Web	Browser platform) (Cross-	DOM elements, HTML structure, CSS properties	Easy setup with JavaScript/- TypeScript support	https://testcafe.io/
BeautifulSoup	Web	HTML Parsing	HTML content, DOM elements	Python library for parsing HTML and XML documents	https://www.crummy.com/software/ BeautifulSoup/
Protractor	Web	Browser (Angular)	DOM elements, Angular-specific attributes	Designed for Angular appli cations, integrates with Selenium	https://www.protractortest.org/
WebDriverIO	Web	Browser platform) (Cross-	DOM elements, HTML/CSS network interactions	Highly extensible with a vast plugin ecosystem	https://webdriver.io/
Ghost Inspector	Web	Browser platform) (Cross-	DOM elements, screenshots, test scripts	Cloud-based automated browser testing and monitoring	https://ghostinspector.com/
Cypress	Web	Browser platform) (Cross-	DOM elements, HTML/CSS. network requests	Real-time reloads and interactive debugging	https://www.cypress.io/
UIAutomator	Mobile	Android	UI hierarchy, widget properties, screen content	Native Android UI testing framework	https://developer.android.com/training/ testing/ui-automator
Espresso	Mobile	Android	UI components, view hierarchy, widget properties	Google's native Android UI testing framework	https://developer.android.com/training/ testing/espressc
Android View Hierarchy	Mobile	Android	UI hierarchy, widget properties, layout information	View hierarchy accessible via developer tools	https://developer.android.com/studio/ debua/layout-inspector
iOS Accessibility Inspector	Mobile	iOS	Accessibility tree, UI elements, properties	Tool for inspecting iOS app UI elements	https://developer.apple.com/ documentation/accessibility/ accessibility-inspector
XCUITest	Mobile	iOS	UI elements, accessibility properties, view hierarchy	Apple's iOS UI testing framework	https://developer.apple.com/ documentation/xctest/user interface tests
Flutter Driver	Mobile	Flutter apps	Widget tree, properties, interactions	Automation for Flutter applications	https://flutter.dev/docs/testing
Android's Me- diaProjection API	Mobile	Android	Screenshots, screen recording	Capturing device screen content programmatically	https://developer.android.com/reference/ android/media/projection/MediaProjection
Windows UI Automation	Computer	Windows	Control properties, widget trees, accessibility tree	Native Windows support with OS integration	https://docs.microsoft.com/windows/win32/ winauto/entry-uiauto-win32
Sikuli	Computer	Windows macOS Linux	Screenshots (image recognition), UI elements	Image-based automation using computer vision	http://sikulix.com/

Autolt	Computer	Windows	Window titles, control properties, coordinates	Scripting language for Windows GUI automation	https://www.autoitscript.com/site/autoit/
Inspect.exe	Computer	Windows	UI elements, control properties, accessibility tree	Tool for inspecting Windows UI elements	https://docs.microsoft.com/windows/win32/winauto/inspect-objects
macOS Accessibility API	Computer	macOS	Accessibility tree, UI elements, control properties	macOS support for accessibility and UI automation	https://developer.apple.com/accessibility/
Pywinauto	Computer	Windows	Control properties, UI hierarchy, window information	Python-based Windows GUI automation	https://pwww.electronis.org/docs/latest/
Electron Inspector	Computer	Electron apps	DOM elements, HTML/CSS JavaScript state	Tool for Electron applications	https://www.electronis.org/docs/latest/tutorial/automated-testing
Windows Snipping Tool	Computer	Windows	Screenshots	Tool for capturing screen-shots in Windows	https://www.microsoft.com/en-us/windows/tips/snipping-tool
macOS Screenshot Utility	Computer	macOS	Screenshots, screen recording	Tool for capturing screen shots and recording screen	https://support.apple.com/guide/mac-help/take-a-screenshot-or%2Dscreen-recording%2Dmh26782/mac
AccessKit	Cross-Platform	Various OS	Accessibility tree, control properties, roles	Standardized APIs across platforms	https://github.com/AccessKit/accesskit
Appium	Cross-Platform	Android, iOS, Windows, macOS	UI elements, accessibility properties, gestures	Mobile automation framework	https://appium.io/
Robot Framework	Cross-Platform	Web, Mobile, Desktop	UI elements, DOM, screen-shots	Extensible with various libraries	https://robotframework.org/
Cucumber	Cross-Platform	Web, Mobile, Desktop	Step definitions, UI interactions	BDD framework supporting automation tools	https://scucumber.io/
TestComplete	Cross-Platform	Web, Mobile, Desktop	UI elements, DOM, control properties	Tool with extensive feature set	https://smartbear.com/product/testcomplete/overview/
Katalon Studio	Cross-Platform	Web, Mobile, Desktop	UI elements, DOM, screen-shots	All-in-one automation solution	https://www.katalon.com/
Ranorex	Cross-Platform	Web, Mobile, Desktop	UI elements, DOM, control properties	Tool with strong reporting fea tures	https://www.ranorex.com/
Applitools	Cross-Platform	Web, Mobile, Desktop	Screenshots, visual check points, DOM elements	AI-powered visual testing	https://applitools.com/

工具	平台	环境	可访问信息	高亮	链接
Selenium (Selenium)	网页	浏览器平台）（跨浏览器	DOM元素，HTML结构，CSS属性	广泛的浏览器支持和自动化能力	https://www.selenium.dev/
Puppeteer (Puppeteer)	网页	浏览器 Firefox）（Chrome，	DOM元素，HTML/CSS，网络请求	无头浏览器自动化，提供丰富的API	https://pptr.dev/
Playwright (Playwright)	网页	浏览器平台）（跨浏览器	DOM元素，HTML/CSS，网络交互	多浏览器支持，具备自动化和测试功能	https://playwright.dev/
TestCafe (TestCafe)	网页	浏览器平台）（跨浏览器	DOM元素，HTML结构，CSS属性	易于设置，支持JavaScript/TypeScript	https://testcafe.io/
BeautifulSoup (BeautifulSoup)	网页	HTML解析	HTML内容，DOM元素	用于解析HTML和XML文档的Python库	https://www.crummy.com/software/BeautifulSoup/
Protractor (Protractor)	网页	浏览器（Angular）	DOM元素，Angular特定属性	专为Angular应用设计，集成Selenium	https://www.protractortest.org/
WebDriverIO (WebDriverIO)	网页	浏览器平台）（跨浏览器	DOM元素，HTML/CSS，网络交互	高度可扩展，拥有庞大的插件生态	https://webdriver.io/
Ghost Inspector (Ghost Inspector)	网页	浏览器平台）（跨浏览器	DOM元素，截图，测试脚本	基于云的自动化浏览器测试与监控	https://ghostinspector.com/
Cypress (Cypress)	网页	浏览器平台）（跨浏览器	DOM元素，HTML/CSS，网络请求	实时重载与交互式调试	https://www.cypress.io/
UIAutomator (UIAutomator)	移动端	安卓	UI层级，控件属性，屏幕内容	原生安卓UI测试框架	https://developer.android.com/training/testing/ui-automator
Espresso (Espresso)	移动端	安卓	UI组件，视图层级，控件属性	谷歌原生安卓UI测试框架	https://developer.android.com/training/testing/espressc
安卓视图层级	移动端	安卓	UI层级，控件属性，布局信息	通过开发者工具可访问的视图层级	https://developer.android.com/studio/debug/layout-inspector
iOS辅助功能检查器	移动端	iOS	辅助功能树，UI元素，属性	用于检查iOS应用UI元素的工具	https://developer.apple.com/documentation/accessibility/accessibility-inspector
XCUITest	移动端	iOS	UI元素，辅助功能属性，视图层级	苹果iOS界面测试框架	https://developer.apple.com/documentation/xctest/user interface tests
Flutter驱动	移动端	Flutter应用	组件树，属性，交互	Flutter应用的自动化	https://flutter.dev/docs/testing
安卓MediaProjection API	移动端	安卓	截图，屏幕录制	以编程方式捕获设备屏幕内容	https://developer.android.com/reference/android/media/projection/MediaProjection
Windows UI自动化	计算机	Windows系统	控件属性，组件树，辅助功能树	原生Windows支持及操作系统集成	https://docs.microsoft.com/windows/win32/winauto/entry-uiauto-win32
Sikuli	计算机	Windows macOS Linux	截图（图像识别），UI元素	基于图像的计算机视觉自动化	http://sikulix.com/
AutoIt	计算机	Windows系统	窗口标题，控件属性，坐标	Windows图形界面自动化脚本语言	https://www.autoitscript.com/site/autoit/
Inspect.exe	计算机	Windows系统	UI元素，控件属性，辅助功能树	Windows UI元素检测工具	https://docs.microsoft.com/windows/win32/winauto/inspect-objects

macOS辅助功能API	计算机	macOS系统	辅助功能树，UI元素，控件属性	macOS对辅助功能和UI自动化的支持	https://developer.apple.com/accessibility/
Pywinauto	计算机	Windows系统	控件属性，UI层级，窗口信息	基于Python的Windows图形界面自动化	https://pwww.electronis.org/docs/latest/
Electron检测器	计算机	Electron应用	DOM元素，HTML/CSS JavaScript状态	Electron应用工具	https://www.electronis.org/docs/latest/tutorial/automated-testing
Windows截图工具	计算机	Windows系统	截图	Windows系统截图工具	https://www.microsoft.com/en-us/windows/tips/snipping-tool
macOS截图工具	计算机	macOS系统	截图，屏幕录制	屏幕截图及录屏工具	https://support.apple.com/guide/mac-help/take-a-screenshot-or%2Dscreen-recording%2Dmh26782/mac
AccessKit	跨平台	多种操作系统	辅助功能树，控件属性，角色	跨平台标准化API	https://github.com/AccessKit/accesskit
Appium	跨平台	安卓，iOS，Windows，macOS	UI元素，辅助功能属性，手势	移动自动化框架	https://appium.io/
Robot Framework	跨平台	网页、移动端、桌面端	UI元素、DOM、屏幕截图	可通过多种库进行扩展	https://robotframework.org/
Cucumber	跨平台	网页、移动端、桌面端	步骤定义、UI交互	支持自动化工具的行为驱动开发（BDD）框架	https://scucumber.io/
TestComplete	跨平台	网页、移动端、桌面端	UI元素、DOM、控件属性	功能丰富的工具	https://smartbear.com/product/testcomplete/overview/
Katalon Studio	跨平台	网页、移动端、桌面端	UI元素、DOM、屏幕截图	一体化自动化解决方案	https://www.katalon.com/
Ranorex	跨平台	网页、移动端、桌面端	UI元素、DOM、控件属性	具备强大报告功能的工具	https://www.ranorex.com/
Applitools	跨平台	网页、移动端、桌面端	屏幕截图、视觉检查点、DOM元素	基于人工智能的视觉测试	https://applitools.com/

10. https://developer.apple.com/library/archive/documentation/
11. https://developer.apple.com/library/archive/documentation/

Accessibility/Conceptual/AccessibilityMacOSX/
Accessibility/Conceptual/AccessibilityMacOSX/

3. UI Element Properties: Each UI element in the interface contains specific properties, such as control type, label text, position, and bounding box dimensions, that help agents target the appropriate components. These properties are instrumental for agents to make decisions about spatial relationships (e.g., adjacent elements) and functional purposes (e.g., distinguishing between buttons and text fields). For instance, web applications reveal properties like DOM attributes (id, class, name) and CSS styles that provide context and control information. These attributes assist agents in pinpointing precise elements for interaction, enhancing their ability to navigate and operate within diverse UI environments. Figure 9 illustrates examples of selected UI element properties extracted by the Windows UI Automation API, which support GUI agents in decision-making.
4. UI元素属性：界面中的每个UI元素都包含特定属性，如控件类型、标签文本、位置和边界框尺寸，这些属性帮助代理定位合适的组件。这些属性对于代理判断空间关系（例如相邻元素）和功能用途（例如区分按钮和文本框）至关重要。例如，Web应用程序会显示DOM

属性（id、class、name）和CSS样式等属性，提供上下文和控件信息。这些属性帮助代理精确定位交互元素，增强其在多样UI环境中的导航和操作能力。图9展示了通过Windows UI自动化API提取的选定UI元素属性示例，支持GUI代理的决策过程。

4. Complementary CV Approaches: When structured information is incomplete or unavailable, computer vision techniques can provide additional insights [181]. For instance, OCR allows agents to extract text content directly from screenshots, facilitating the reading of labels, error messages, and instructions [121]. Furthermore, advanced object detection [120] models like SAM (Segment Anything Model) [182], DINO [183] and OmniParser 184 can identify and classify UI components in various layouts, supporting the agent in dynamic environments where UI elements may frequently change. These vision-based methods ensure robustness, enabling agents to function effectively even in settings where standard UI APIs are insufficient. We illustrate an example of this complementary information in Figure 10 and further detail these advanced computer vision approaches in Section 5.7.1
5. 补充的计算机视觉方法：当结构化信息不完整或不可用时，计算机视觉技术可以提供额外的洞见[181]。例如，OCR（光学字符识别）允许代理直接从截图中提取文本内容，便于读取标签、错误信息和指令[121]。此外，先进的目标检测模型如SAM（Segment Anything Model）[182]、DINO[183]和OmniParser[184]能够识别和分类各种布局中的UI组件，支持代理在UI元素频繁变化的动态环境中工作。这些基于视觉的方法确保了鲁棒性，使代理即使在标准UI API不足的情况下也能有效运行。图10展示了此类补充信息的示例，5.7.1节进一步详述了这些先进的计算机视觉方法。

Together, these elements create a comprehensive, multimodal representation of the GUI environment's current state, delivering both structured and visual data. By incorporating this information into prompt construction, agents are empowered to make well-informed, contextually aware decisions without missing critical environmental cues.
这些元素共同构建了GUI环境当前状态的全面多模态表示，提供结构化和视觉数据。通过将这些信息纳入提示构建，代理能够做出信息充分、具上下文感知的决策，不遗漏关键环境线索。

10.4.5 5.2.3 Environment Feedback

10.4.6 5.2.3 环境反馈

Effective feedback mechanisms are essential for GUI agents to assess the success of each action and make informed decisions for subsequent steps. Feedback can take several forms, depending on the platform and interaction type. Figure 11 presents examples of various types of feedback obtained from the environment.
有效的反馈机制对于GUI代理评估每个操作的成功与否并为后续步骤做出明智决策至关重要。反馈形式多样，取决于平台和交互类型。图11展示了从环境中获得的各种反馈示例。

1. Screenshot Update: By comparing before-and-after screenshots, agents can identify visual differences that signify state changes in the application. Screenshot analysis can reveal subtle variations in the interface, such as the appearance of a notification, visual cues, or confirmation messages, that may not be captured by structured data [185].
2. 截图更新：通过比较操作前后的截图，代理可以识别应用状态变化的视觉差异。截图分析能揭示界面中的细微变化，如通知的出现、视觉提示或确认信息，这些可能无法通过结构化数据捕捉[185]。

2. UI Structure Change: After executing an action, agents can detect modifications in the widget tree structure, such as the appearance or disappearance of elements, updates to element properties, or hierarchical shifts [186]. These changes indicate successful interactions (e.g., opening a dropdown or clicking a button) and help the agent determine the next steps based on the updated environment state.
3. UI结构变化：执行操作后，代理可检测控件树结构的修改，如元素的出现或消失、属性更新或层级变动[186]。这些变化表明交互成功（例如打开下拉菜单或点击按钮），帮助代理基于更新的环境状态确定下一步操作。

3. Function Return Values and Exceptions: Certain platforms offer direct feedback on action outcomes through function return values or system-generated exceptions 187 . For example, API responses or JavaScript return values can confirm action success on web platforms, while exceptions or error codes can signal failed interactions, guiding the agent to retry or select an alternative approach.
4. 函数返回值和异常：某些平台通过函数返回值或系统生成的异常提供操作结果的直接反馈[187]。例如，API响应或JavaScript返回值可确认Web平台上的操作成功，而异常或错误代码则指示交互失败，指导代理重试或选择替代方案。

These feedback provided by the environment is crucial for GUI agents to assess the outcomes of their previous actions. This real-time information enables agents to evaluate the effectiveness of their interventions and determine whether to adhere to their initial plans or pivot towards alternative strategies. Through this process of self-reflection, agents can adapt their decision-making, optimizing task execution and enhancing overall performance in dynamic and varied application environments.
环境提供的这些反馈对于GUI代理评估先前操作结果至关重要。实时信息使代理能够评估干预效果，决定是坚持初始计划还是转向替代策略。通过这种自我反思过程，代理可调整决策，优化任务执行，提升在动态多变应用环境中的整体表现。

10.5 5.3 Prompt Engineering

10.6 5.3 提示工程

In the operation of LLM-powered GUI agents, effective prompt construction is a crucial step that encapsulates all necessary information for the agent to generate appropriate responses and execute tasks successfully [168]. After gathering the relevant data from the environment, the agent formulates a comprehensive prompt that combines various components essential for inference by the LLM. Each component serves a specific purpose, and together they enable the agent to execute the user's request efficiently. Figure 12 illustrates a basic example of prompt construction in an LLM-brained GUI agent. The key elements of the prompt are summarized as follows:
在基于大型语言模型（LLM）的GUI代理操作中，有效的提示构建是关键步骤，涵盖代理生成适当响应和成功执行任务所需的所有信息[168]。在收集环境相关数据后，代理制定综合提示，结合多种推理所需的组成部分。每个部分具有特定功能，共同支持代理高效执行用户请求。图12展示了LLM驱动GUI代理中提示构建的基本示例。提示的关键要素总结如下：

1. User Request: This is the original task description provided by the user, outlining the objective and desired outcome. It serves as the foundation for the agent's actions and is critical for ensuring that the LLM understands the context and scope of the task.
2. 用户请求：这是用户提供的原始任务描述，概述目标和期望结果。它是代理行动的基础，对于确保LLM理解任务的上下文和范围至关重要。

2. Agent Instruction: This section provides guidance for the agent's operation, detailing its role, rules to follow, and specific objectives. Instructions clarify what inputs the agent will receive and outline the expected outputs from the LLM, establishing a framework for the inference process. The core agent instructions are usually embedded within the base system prompt of the LLM, with supplementary instructions dynamically injected or updated based on environmental feedback and contextual adaptation.
3. 代理指令：本部分为代理操作提供指导，详细说明其角色、遵循规则和具体目标。指令明确代理将接收的输入和LLM预期输出，建立推理过程的框架。核心代理指令通常嵌入LLM的基础系统提示中，辅以根据环境反馈和上下文适应动态注入或更新的补充指令。

3. Environment States: The agent includes perceived GUI screenshots and UI information, as introduced in Section 5.2.2. This multimodal data may consist of various versions of screenshots (e.g., a clean version and a SoM annotated version) to ensure clarity and mitigate the risk of UI controls being obscured by annotations. This comprehensive representation of the environment is vital for accurate decision-making.
4. 环境状态：代理包括感知到的GUI截图和UI信息，如第5.2.2节所述。这些多模态数据可能包含多个版本的截图（例如，干净版本和带SoM注释的版本），以确保清晰度并减少UI控件被注释遮挡的风险。这种对环境的全面表示对于准确决策至关重要。

4. Action Documents: This component outlines the available actions the agent can take, detailing relevant documentation, function names, arguments, return values, and any other necessary parameters. Providing this 5.4 information equips the LLM with the context needed to select and generate appropriate actions for the task at hand.
5. 操作文档：该组件概述了代理可执行的操作，详细说明相关文档、函数名称、参数、返回值及其他必要参数。提供这些5.4信息使大型语言模型（LLM）具备选择和生成适当操作以完成任务的上下文。

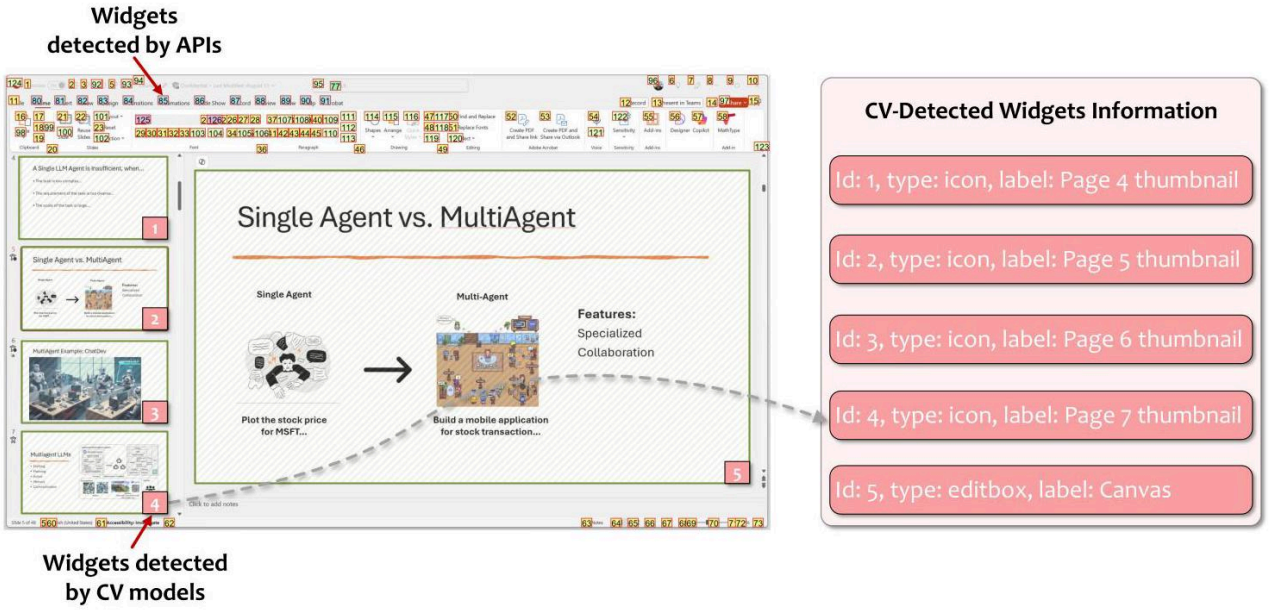

Fig. 10: An example illustrating the use of a CV approach to parse a PowerPoint GUI and detect non-standard widgets, inferring their types and labels.
图10：一个示例，展示如何使用计算机视觉（CV）方法解析PowerPoint GUI并检测非标准控件，推断其类型和标签。

5. Demonstrated Examples: Including example input/output pairs is essential to activate the in-context learning 97 capability of the LLM. These examples help the model comprehend and generalize the task requirements, enhancing its performance in executing the GUI agent's responsibilities.
6. 演示示例：包含输入/输出示例对对于激活LLM的上下文学习能力97至关重要。这些示例帮助模型理解并泛化任务需求，提升其执行GUI代理职责的表现。

6. Complementary Information: Additional context that aids in planning and inference may also be included. This can consist of historical data retrieved from the agent's memory (as detailed in Section 5.6) and external knowledge sources, such as documents obtained through retrieval-augmented generation (RAG) methods 188], [189]. This supplemental information can provide valuable insights that further refine the agent's decision-making processes.
7. 补充信息：还可包含辅助规划和推理的额外上下文。这可能包括从代理记忆中检索的历史数据（详见第5.6节）以及通过检索增强生成（RAG）方法获得的外部知识源[188]，[189]。这些补充信息能提供有价值的见解，进一步优化代理的决策过程。

The construction of an effective prompt is foundational for the performance of LLM-powered GUI agents. By systematically incorporating aforementioned information, the agent ensures that the LLM is equipped with the necessary context and guidance to execute tasks accurately and efficiently.
构建有效的提示词是LLM驱动GUI代理性能的基础。通过系统地整合上述信息，代理确保LLM具备执行任务所需的上下文和指导，从而实现准确高效的操作。

10.7 5.4 Model Inference

10.8 5.4 模型推理

The constructed prompt is submitted to the LLM for inference, where the LLM is tasked with generating both a plan and the specific actions required to execute the user's request. This inference process is critical as it dictates how effectively the GUI agent will perform in dynamic environments. It typically involves two main components: planning and action inference, as well as the generation of complementary outputs. Figure 13 shows an example of the LLM's inference output.
构建好的提示词提交给LLM进行推理，LLM负责生成执行用户请求所需的计划和具体操作。该推理过程至关重要，决定了GUI代理在动态环境中的表现。通常包括两个主要部分：规划与动作推理，以及补充输出的生成。图13展示了LLM推理输出的示例。

10.8.1 5.4.1 Planning

10.8.2 5.4.1 规划

Successful execution of GUI tasks often necessitates a series of sequential actions, requiring the agent to engage in effective planning [52], [190]. Analogous to human cognitive processes, thoughtful planning is essential to organize tasks, schedule actions, and ensure successful completion [51], 191]. The LLM must initially conceptualize a long-term goal while simultaneously focusing on short-term actions to initiate progress toward that goal 192.
成功执行GUI任务通常需要一系列连续动作，要求代理进行有效规划[52]，[190]。类似于人类认知过程，周密的规划对于组织任务、安排操作并确保顺利完成至关重要[51]，[191]。LLM必须首先构思长期目标，同时关注短期动作以推动目标实现[192]。

To effectively navigate the complexity of multi-step tasks, the agent should decompose the overarching task into manageable subtasks and establish a timeline for their execution 193]. Techniques such as CoT reasoning [100] can be employed, enabling the LLM to develop a structured plan that guides the execution of actions. This plan, which can be stored for reference during future inference steps, enhances the organization and focus of the agent's activities.
为有效应对多步骤任务的复杂性，代理应将整体任务分解为可管理的子任务，并制定执行时间表[193]。可采用链式推理（CoT）[100]等技术，使LLM制定结构化计划，指导动作执行。该计划可存储以供后续推理步骤参考，提升代理活动的组织性和专注度。

The granularity of planning may vary based on the nature of the task and the role of the agent [51]. For complex tasks, a hierarchical approach that combines global planning (identifying broad subgoals) with local planning (defining detailed steps for those subgoals) can significantly improve the agent's ability to manage long-term objectives effectively 194].
规划的粒度可能因任务性质和代理角色而异[51]。对于复杂任务，结合全局规划（识别广泛子目标）与局部规划（定义子目标的详细步骤）的分层方法，能显著提升代理有效管理长期目标的能力[194]。

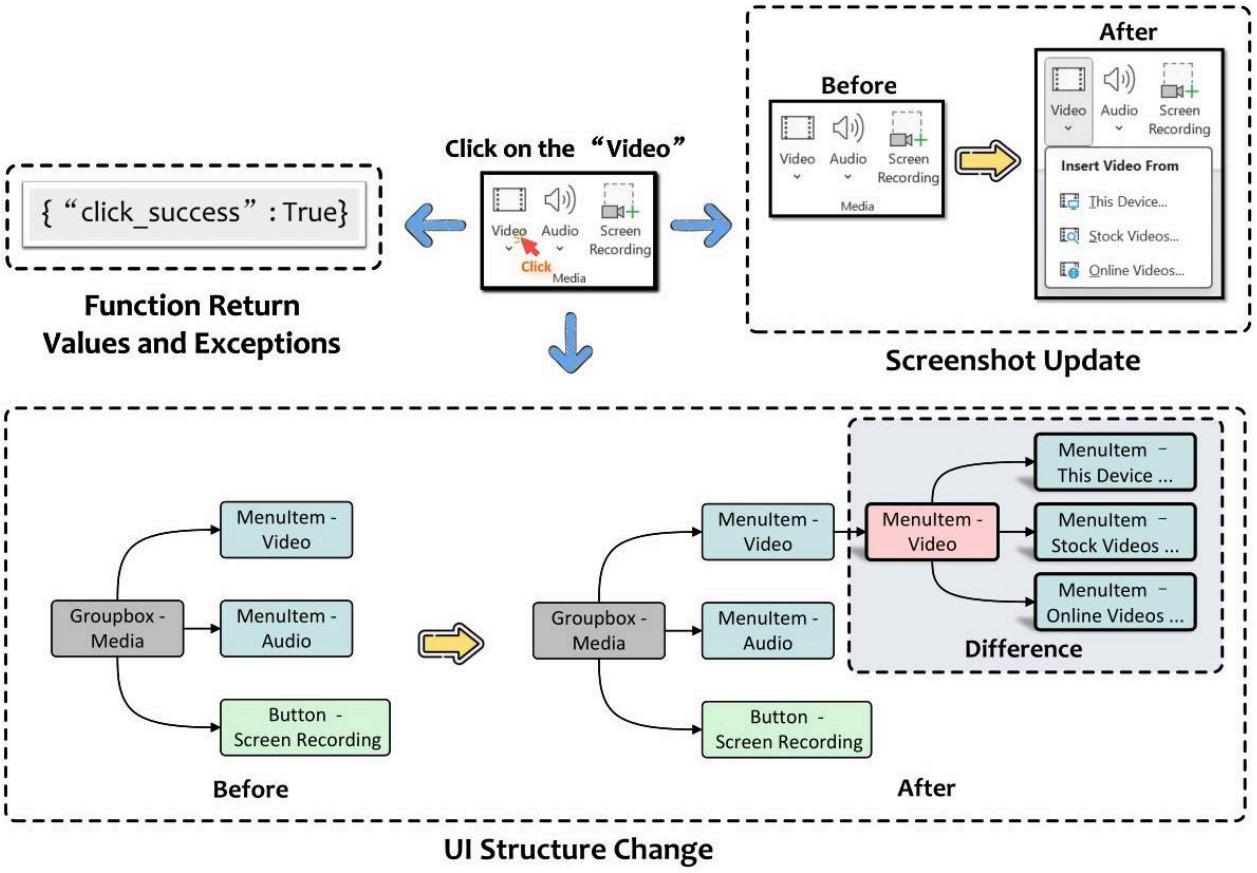

Fig. 11: Examples of various types of feedback obtained from a PowerPoint application environment.
图11：从PowerPoint应用环境获得的各种反馈类型示例。

10.8.3 5.4.2 Action Inference

10.8.4 5.4.2 动作推理

Action inference is the core objective of the inference stage, as it translates the planning into executable tasks. The inferred actions are typically expressed as function call strings, encompassing the function name and relevant parameters. These strings can be readily converted into real-world interactions with the environment, such as clicks, keyboard inputs, mobile gestures, or API calls. A detailed discussion of these action types is presented in Section 5.5
动作推理是推理阶段的核心目标，将规划转化为可执行任务。推断出的动作通常以函数调用字符串形式表达，包含函数名及相关参数。这些字符串可直接转换为与环境的实际交互，如点击、键盘输入、移动手势或API调用。第5.5节对这些动作类型进行了详细讨论。

The input prompt must include a predefined set of actions available for the agent to select from. The agent can choose an action from this set or, if allowed, generate custom code or API calls to interact with the environment [161]. This flexibility can enhance the agent's adaptability to unforeseen circumstances; however, it may introduce reliability concerns, as the generated code may be prone to errors.
输入提示必须包含代理可选的预定义动作集。代理可从中选择动作，或在允许的情况下生成自定义代码或API调用以与环境交互[161]。这种灵活性增强了代理对突发情况的适应能力，但可能带来可靠性问题，因为生成的代码可能存在错误。

10. https://developer.apple.com/library/archive/documentation/ AppleScript/Conceptual/AppleScriptLangGuide/introduction/ASLR_intro.html
11. https://developer.apple.com/library/archive/documentation/ AppleScript/Conceptual/AppleScriptLangGuide/introduction/ASLR_intro.html

---

```
11. https://www.macosxautomation.com/automator/
11. https://www.macosxautomation.com/automator/
```

```
12. https://docs.blender.org/manual/en/latest/sculpt_paint/sculpting/
12. https://docs.blender.org/manual/en/latest/sculpt_paint/sculpting/
```

introduction/gesture tools.html
introduction/gesture tools.html

```
13. https://developer.android.com/reference/android/speech/
13. https://developer.android.com/reference/android/speech/
```

SpeechRecognizer
SpeechRecognizer

```
14. https://developer.apple.com/documentation/sirikit/
14. https://developer.apple.com/documentation/sirikit/
```

```
15. https://pypi.org/project/pyperclip/
15. https://pypi.org/project/pyperclip/
```

```
16. https://clipboardjs.com/
16. https://clipboardjs.com/
```

```
17. https://developer.android.com/develop/sensors-and-location/
17. https://developer.android.com/develop/sensors-and-location/
```

sensors/sensors overview
sensors/sensors overview

```
18. https://learn.microsoft.com/en-us/previous-versions/office/
18. https://learn.microsoft.com/en-us/previous-versions/office/
```

office-365-api/
office-365-api/

```
19. https://developer.android.com/reference
19. https://developer.android.com/reference
```

```
20. https://developer.apple.com/ios/
20. https://developer.apple.com/ios/
```

```
21. https://learn.microsoft.com/en-us/windows/win32/api/
21. https://learn.microsoft.com/en-us/windows/win32/api/
```

```
22. https://developer.apple.com/library/archive/documentation/
22. https://developer.apple.com/library/archive/documentation/
```

Cocoa/Conceptual/CocoaFundamentals/WhatIsCocoa/WhatIsCocoa.
Cocoa/Conceptual/CocoaFundamentals/WhatIsCocoa/WhatIsCocoa.

```
23. https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API
23. https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API
```

```
24. https://axios-http.com/docs/api_intro
24. https://axios-http.com/docs/api_intro
```

```
25. https://platform.openai.com/docs/overview
25. https://platform.openai.com/docs/overview
```

10.8.5 5.4.3 Complementary Outputs

10.8.6 5.4.3 补充输出

In addition to planning and action inference, the LLM can also generate complementary outputs that enhance the agent's capabilities. These outputs may include reasoning processes that clarify the agent's decision-making (e.g., CoT reasoning), messages for user interaction, or communication with other agents or systems, or the status of the task (e.g., continue or finished). The design of these functionalities can be tailored to meet specific needs, thereby enriching the overall performance of the GUI agent.
除了规划和动作推断外，大型语言模型（LLM）还可以生成补充输出，以增强代理的能力。这些输出可能包括阐明代理决策过程的推理过程（例如，链式思维（CoT）推理）、用于用户交互的消息、与其他代理或系统的通信，或任务状态（例如，继续或完成）。这些功能的设计可以根据具体需求进行定制，从而丰富GUI代理的整体性能。

By effectively balancing planning and action inference while incorporating complementary outputs, agents can navigate complex tasks with a higher degree of organization and adaptability.
通过有效平衡规划与动作推断，同时结合补充输出，代理能够以更高的组织性和适应性应对复杂任务。

10.9 5.5 Actions Execution

10.10 5.5 动作执行

Following the inference process, a crucial next step is for the GUI agent to execute the actions derived from the inferred commands within the GUI environment and subsequently gather feedback. Although the term "GUI agent" might suggest a focus solely on user interface actions, the action space can be greatly expanded by incorporating various toolboxes that enhance the agent's versatility. Broadly, the actions available to GUI agents fall into three main categories: (i) UI operations [144], (ii) native API calls [196], and (iii) AI tools [197]. Each category offers unique advantages and challenges, enabling the agent to tackle a diverse range of
在推断过程之后，关键的下一步是GUI代理在GUI环境中执行从推断命令中得出的动作，并随后收集反馈。尽管“GUI代理”一词可能暗示仅关注用户界面动作，但通过整合各种工具箱，可以大幅扩展动作空间，提升代理的多样性。总体而言，GUI代理可用的动作主要分为三类：（i）用户界面操作[144]，（ii）本地API调用[196]，以及（iii）人工智能工具[197]。每类动作各有优势和挑战，使代理能够更有效地处理多样化的

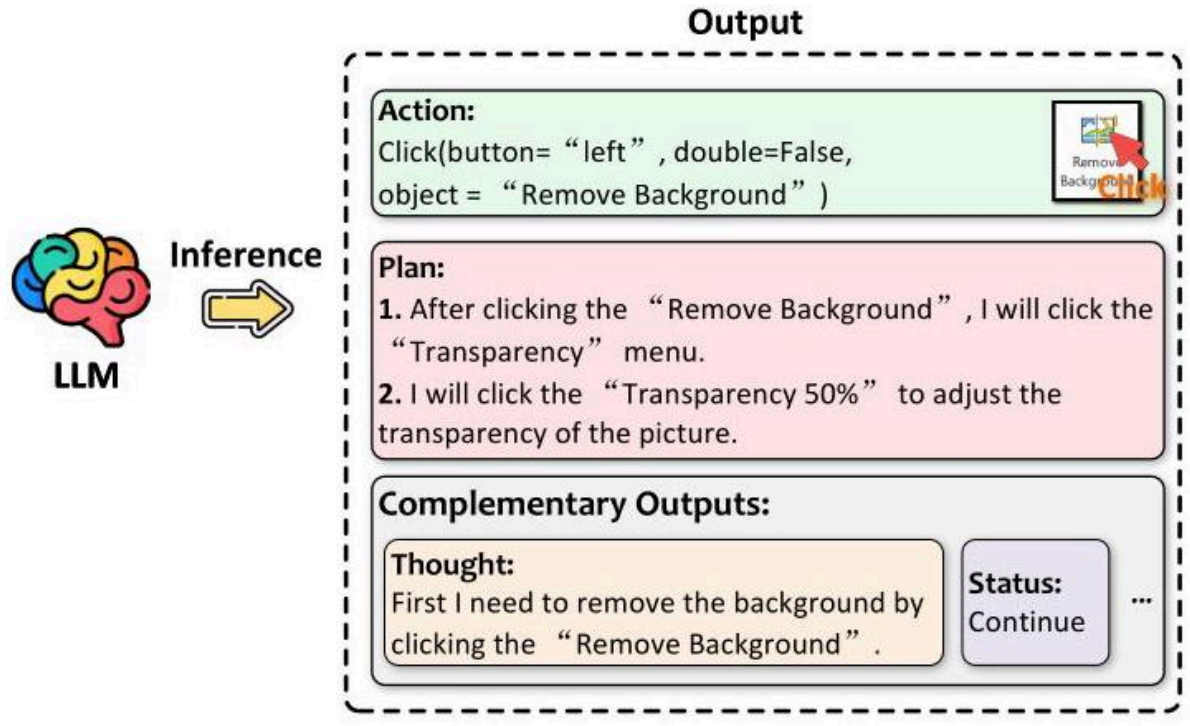

Fig. 13: An example of the LLM's inference output in a GUI agent.
图13：LLM在GUI代理中的推断输出示例。

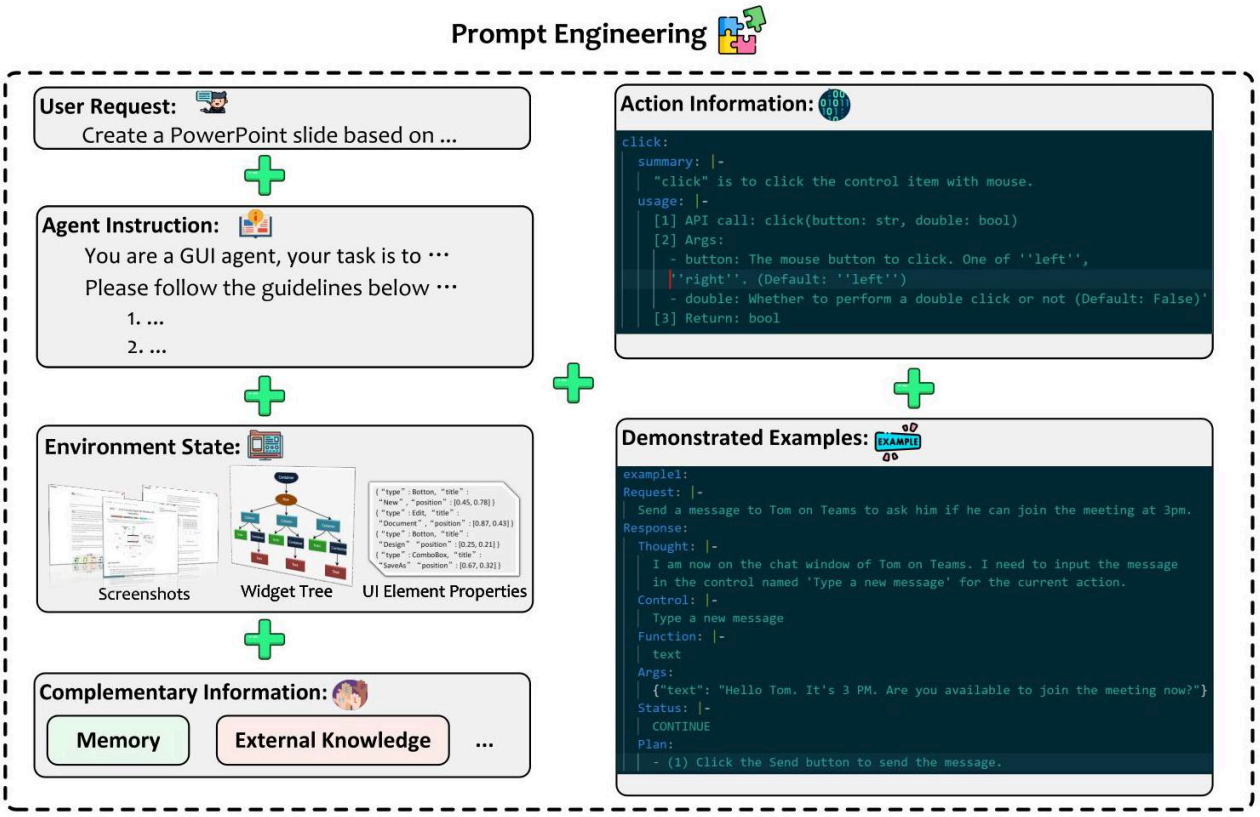

Fig. 12: A basic example of prompt construction in a LLM-brained GUI agent.
图12：基于LLM的GUI代理中提示构建的基本示例。

tasks more effectively. We summarize the various actions commonly used in GUI agents, categorized into distinct types, in Table 5 and provide detailed explanations of each category below.
任务。我们在表5中总结了GUI代理中常用的各种动作，按类别区分，并在下文详细说明每个类别。

10.10.1 5.5.1 UI Operations

10.10.2 5.5.1 用户界面操作

UI operations encompass the fundamental interactions that users typically perform with GUIs in software applications. These operations include various forms of input, such as mouse actions (clicks, drags, hovers), keyboard actions (key presses, combinations), touch actions (taps, swipes), and gestures (pinching, rotating). The specifics of these actions may differ across platforms and applications, necessitating a tailored approach for each environment.
用户界面操作涵盖用户在软件应用的GUI中通常执行的基本交互。这些操作包括各种输入形式，如鼠标动作（点击、拖拽、悬停）、键盘动作（按键、组合键）、触控动作（轻触、滑动）和手势（捏合、旋转）。这些动作的具体细节可能因平台和应用而异，因此需要针对每个环境进行定制。

While UI operations form the foundation of agent interactions with the GUI, they can be relatively slow due to the sequential nature of these tasks. Each operation must be executed step by step, which can lead to increased latency, especially for complex workflows that involve numerous interactions. Despite this drawback, UI operations are crucial for maintaining a broad compatibility across various applications, as they leverage standard user interface elements and interactions.
尽管用户界面操作构成了代理与GUI交互的基础，但由于这些任务的顺序执行特性，操作速度可能较慢。每个操作必须逐步执行，这在涉及大量交互的复杂工作流程中可能导致延迟增加。尽管如此，用户界面操作对于保持跨各种应用的广泛兼容性至关重要，因为它们利用了标准的用户界面元素和交互方式。

10.10.3 5.5.2 Native API Calls

10.10.4 5.5.2 本地API调用

In contrast to UI operations, some applications provide native APIs that allow GUI agents to perform actions more efficiently. These APIs offer direct access to specific functionalities within the application, enabling the agent to execute complex tasks with a single command 198]. For instance, calling the Outlook API allows an agent to send an email in one operation, whereas using UI operations would require a series of steps, such as navigating through menus and filling out forms [199].
与UI操作相比，一些应用程序提供本地API，允许GUI代理更高效地执行操作。这些API提供对应用程序中特定功能的直接访问，使代理能

够通过单一命令执行复杂任务[198]。例如，调用Outlook API可以让代理一次性发送电子邮件，而使用UI操作则需要一系列步骤，如导航菜单和填写表单[199]。

While native APIs can significantly enhance the speed and reliability of action execution, their availability is limited. Not all applications or platforms expose APIs for external use, and developing these interfaces can require substantial effort and expertise. Consequently, while native APIs present a powerful means for efficient task completion, they may not be as generalized across different applications as UI operations.
虽然本地API能显著提升操作执行的速度和可靠性，但其可用性有限。并非所有应用程序或平台都对外开放API，且开发这些接口可能需要大量的努力和专业知识。因此，尽管本地API是高效完成任务的强大手段，但它们在不同应用中的通用性不及UI操作。

TABLE 5: Overview of actions for GUI agents.
表5：GUI代理操作概览。

Action	Category	Original Executor	Examples	Platform	Environment	Toolkit
Mouse actions	UI Operations	Mouse	Click, scroll, hover, drag	Computer	Windows	UI Automation 6, Pywinauto
Mouse actions	UI Operations	Mouse	Click, scroll, hover, drag	Computer	macOS	AppleScript 10, Automator 11
Mouse actions	UI Operations	Mouse	Click, scroll, hover, drag	Web	Browser	Selenium, Puppeteer
Keyboard actions	UI Operations	Keyboard	Typing, key presses, shortcuts	Computer	Windows	UI Automation 6. Pywinauto
Keyboard actions	UI Operations	Keyboard	Typing, key presses, shortcuts	Computer	macOS	AppleScript 10, Automator 1
Keyboard actions	UI Operations	Keyboard	Typing, key presses, shortcuts	Web	Browser	Selenium. Puppeteer
Touch actions	UI Operations	Touchscreen	Tap, swipe, pinch, zoom	Mobile	Android	Appium, UIAu-tomator
Touch actions	UI Operations	Touchscreen	Tap, swipe, pinch, zoom	Mobile	iOS	Appium, XCUITest
Gesture actions	UI Operations	User hand	Rotate, multi-finger gestures	Mobile	Android, iOS	Appium , G tureTools 12
Voice commands	UI Operations	User voice	Speech input, voice commands	Mobile	Android	SpeechRecognize 13
Voice commands	UI Operations	User voice	Speech input, voice commands	Mobile	iOS	SiriKit 14
Clipboard operations	UI Operations	System clipboard	Copy, paste	Cross-platform	Cross-OS	Pyperclip Clipboard.js 16
Screen interactions	UI Operations	User	Screen rotation, shake	Mobile	Android, iOS	Device sensors APIs 17
Shell Commands	Native API Calls	Command Line Interface	File manipulation, system operations, script execution	Computer	Unix/Linux, macOS	Bash, Terminal
Application APIs	Native API Calls	Application APIs	Send email, create document. fetch data	Computer	Windows	Microsoft Office COM APIs 18
Application APIs	Native API Calls	Application APIs	Access calendar. send messages	Mobile	Android	Android SDK APIs 19
Application APIs	Native API Calls	Application APIs	Access calendar, send messages	Mobile	iOS	iOS SDK APIs 20
System APIs	Native API Calls	System APIs	File operations, network requests	Computer	Windows	Win32 API 21
System APIs	Native API Calls	System APIs	File operations, network requests	Computer	macOS	Cocoa APIs 22
Web APIs	Native API Calls	Web Services	Fetch data, submit forms	Web	Browser	Fetch API ^{23} . Axios 24
AI Models	AI Tools	AI Models	Screen understanding, summarization, image generation	Cross-platform	Cross-OS	DALL · E 1951 OpenAI APIs 26

操作	类别	原始执行者	示例	平台	环境	工具包
鼠标操作	用户界面操作	鼠标	点击，滚动，悬停，拖拽	计算机	Windows	UI Automation 6，Pywinauto
鼠标操作	用户界面操作	鼠标	点击，滚动，悬停，拖拽	计算机	macOS	AppleScript 10，Automator 11
鼠标操作	用户界面操作	鼠标	点击，滚动，悬停，拖拽	网页	浏览器	Selenium，Puppeteer
键盘操作	用户界面操作	键盘	输入，按键，快捷键	计算机	Windows	UI Automation 6，Pywinauto
键盘操作	用户界面操作	键盘	输入，按键，快捷键	计算机	macOS	AppleScript 10，Automator 1
键盘操作	用户界面操作	键盘	输入，按键，快捷键	网页	浏览器	Selenium，Puppeteer
触控操作	用户界面操作	触摸屏	点击，滑动，捏合，缩放	移动端	Android	Appium，UIAutomator
触控操作	用户界面操作	触摸屏	点击，滑动，捏合，缩放	移动端	iOS	Appium，XCUITest
手势操作	用户界面操作	用户手势	旋转，多指手势	移动端	Android，iOS	Appium，GestureTools 12
语音命令	用户界面操作	用户语音	语音输入，语音命令	移动端	Android	SpeechRecognize 13
语音命令	用户界面操作	用户语音	语音输入，语音命令	移动端	iOS	SiriKit 14
剪贴板操作	用户界面操作	系统剪贴板	复制，粘贴	跨平台	跨操作系统	Pyperclip，Clipboard.js 16
屏幕交互	用户界面操作	用户	屏幕旋转，摇晃	移动端	Android，iOS	设备传感器API 17
Shell命令	本地API调用	命令行界面	文件操作，系统操作，脚本执行	计算机	Unix/Linux，macOS	Bash，终端
应用程序API	本地API调用	应用程序API	发送邮件，创建文档，获取数据	计算机	Windows	Microsoft Office COM API 18
应用程序API	本地API调用	应用程序API	访问日历，发送消息	移动端	Android	Android SDK API 19
应用程序API	本地API调用	应用程序API	访问日历，发送消息	移动端	iOS	iOS SDK 应用程序接口 20
系统应用程序接口	本地API调用	系统应用程序接口	文件操作，网络请求	计算机	Windows	Win32 应用程序接口 21
系统应用程序接口	本地API调用	系统应用程序接口	文件操作，网络请求	计算机	macOS	Cocoa 应用程序接口 22
Web 应用程序接口	本地API调用	Web 服务	获取数据，提交表单	网页	浏览器	Fetch API ^{23} 。Axios 24
人工智能模型	人工智能工具	人工智能模型	屏幕理解，摘要，图像生成	跨平台	跨操作系统	DALL · E 1951 OpenAI 应用程序接口 26

10.10.5 5.5.3 AI Tools

10.10.6 5.5.3 AI工具

The integration of AI tools into GUI agents represents a transformative advancement in their capabilities. These tools can assist with a wide range of tasks, including content summarization from screenshots or text, document enhancement, image or video generation (e.g., calling ChatGPT [11], DALL-E 195]), and even invoking other agents or Copilot tools for collaborative assistance. The rapid development of generative AI technologies enables GUI agents to tackle complex challenges that were previously beyond their capabilities.
将AI工具集成到GUI代理中代表了其能力的变革性进步。这些工具可以协助完成广泛的任务，包括从截图或文本中提取内容摘要、文档增强、图像或视频生成（例如调用ChatGPT[11]、DALL-E[195]），甚至调用其他代理或Copilot工具进行协作辅助。生成式AI技术的快速发展使GUI代理能够应对此前超出其能力范围的复杂挑战。

By incorporating AI tools, agents can extend their functionality and enhance their performance in diverse contexts. For example, a GUI agent could use an AI summarization tool to quickly extract key information from a lengthy document or leverage an image generation tool to create custom visuals for user presentations. This integration not only streamlines workflows but also empowers agents to deliver high-quality outcomes in a fraction of the time traditionally required.
通过整合AI工具，代理可以扩展其功能并提升在多种场景下的表现。例如，GUI代理可以使用AI摘要工具快速提取冗长文档中的关键信息，

或利用图像生成工具为用户演示创建定制视觉内容。这种集成不仅简化了工作流程，还使代理能够在传统所需时间的一小部分内交付高质量成果。

10.10.7 5.5.4 Summary

10.10.8 5.5.4 总结

An advanced GUI agent should adeptly leverage all three categories of actions: UI operations for broad compatibility, native APIs for efficient execution, and AI tools for enhanced capabilities. This multifaceted approach enables the agent to operate reliably across various applications while maximizing efficiency and effectiveness. By skillfully navigating these action types, GUI agents can fulfill user requests more proficiently, ultimately leading to a more seamless and productive user experience.
一个先进的GUI代理应熟练利用三类操作：广泛兼容的UI操作、高效执行的本地API以及增强能力的AI工具。这种多维度方法使代理能够在各种应用中可靠运行，同时最大化效率和效果。通过巧妙驾驭这些操作类型，GUI代理能够更高效地满足用户需求，最终带来更流畅且富有成效的用户体验。

10.11 5.6 Memory

10.12 5.6 记忆

For a GUI agent to achieve robust performance in complex, multi-step tasks, it must retain memory, enabling it to manage states in otherwise stateless environments. Memory allows the agent to track its prior actions, their outcomes, and the task's overall status, all of which are crucial for informed decision-making in subsequent steps [200]. By establishing continuity, memory transforms the agent from a reactive system into a proactive, stateful one, capable of self-adjustment based on accumulated knowledge. The agent's memory is generally divided into two main types: Short-Term Memory [201] and Long-Term Memory [202]. We show an overview of different types of memory in GUI agents in Table 6
为了使GUI代理在复杂的多步骤任务中表现稳健，必须具备记忆能力，从而在本质上无状态的环境中管理状态。记忆使代理能够追踪先前的操作、其结果及任务的整体状态，这些对于后续步骤的明智决策至关重要[200]。通过建立连续性，记忆将代理从反应式系统转变为主动的、有状态系统，能够基于积累的知识自我调整。代理的记忆通常分为两大类：短期记忆[201]和长期记忆[202]。我们在表6中展示了GUI代理中不同类型记忆的概览。

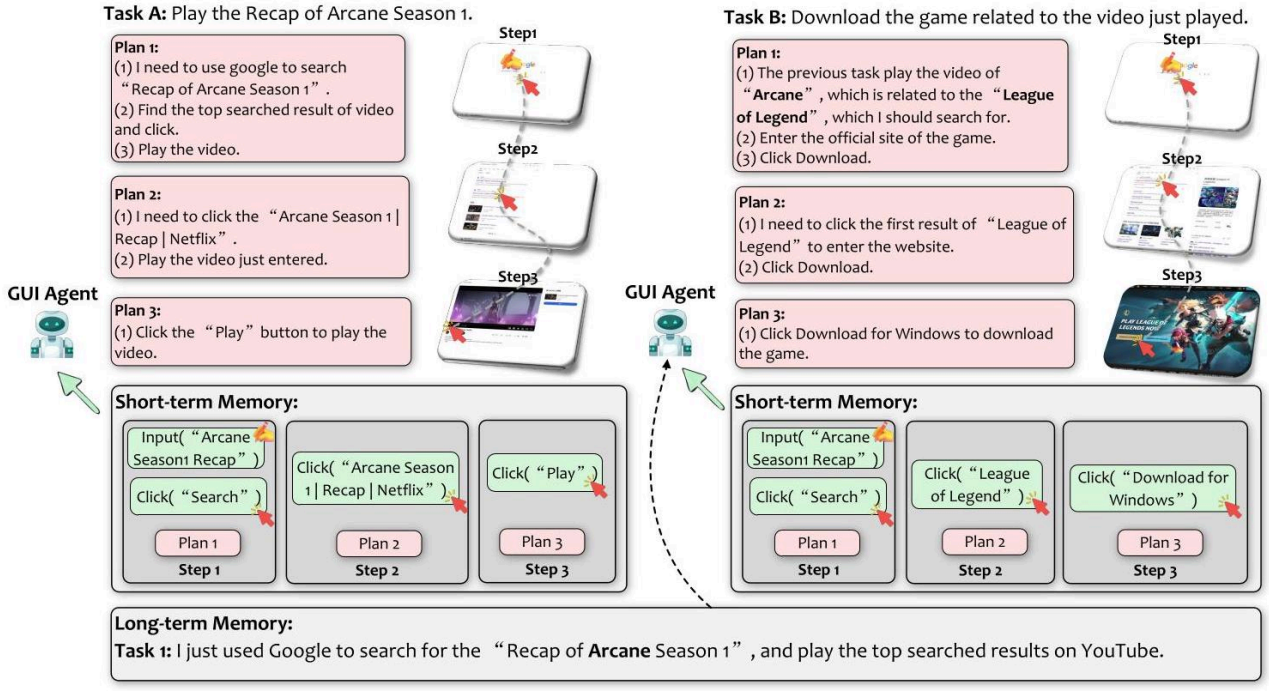

Fig. 14: Illustration of short-term memory and long-term memory in an LLM-brained GUI agent.
图14：具备大型语言模型（LLM）核心的GUI代理中短期记忆与长期记忆的示意图。

TABLE 6: Summary of memory in GUI agents.
表6：GUI代理中记忆的总结。

Memory Element	Memory Type	Description	Storage Medium/Method
Action	Short-term	Historical actions trajectory taken in the environment	In-memory, Context window
Plan	Short-term	Plan passed from previous step	In-memory, Context window
Execution Results	Short-term	Return values, error traces, and other environmental feedback	In-memory, Context window
Environment State	Short-term	Important environment state data, e.g., UI elements	In-memory, Context window
Self-experience	Long-term	Task completion trajectories from historical tasks	Database, Disk
Self-guidance	Long-term	Guidance and rules summarized from historical trajectories	Database, Disk
External Knowledge	Long-term	Other external knowledge sources aiding task completion	External Knowledge Base
Task Success Metrics	Long-term	Metrics from task success or failure rates across sessions	Database, Disk

记忆元素	记忆类型	描述	存储介质/方法
动作	短期	环境中采取的历史动作轨迹	内存中，上下文窗口
计划	短期	来自上一步的计划	内存中，上下文窗口
执行结果	短期	返回值、错误追踪及其他环境反馈	内存中，上下文窗口
环境状态	短期	重要的环境状态数据，例如用户界面元素	内存中，上下文窗口
自我经验	长期	历史任务的完成轨迹	数据库，磁盘
自我指导	长期	从历史轨迹总结的指导和规则	数据库，磁盘
外部知识	长期	辅助任务完成的其他外部知识来源	外部知识库
任务成功指标	长期	跨会话任务成功或失败率的指标	数据库，磁盘

10.12.1 5.6.1 Short-Term Memory

10.12.2 5.6.1 短期记忆

Short-Term Memory (STM) provides the primary, ephemeral context used by the LLM during runtime [203]. STM stores information pertinent to the current task, such as recent plans, actions, results, and environmental states, and continuously updates to reflect the task's ongoing status. This memory is particularly valuable in multi-step tasks, where each decision builds on the previous one, requiring the agent to maintain a clear understanding of the task's trajectory. As illustrated in Figure 14 during the completion of independent tasks, the task trajectory, comprising actions and plans-is stored in the STM. This allows the agent to track task progress effectively and make more informed decisions.
短期记忆（STM）为大型语言模型（LLM）在运行时提供主要的、短暂的上下文[203]。STM存储与当前任务相关的信息，如最近的计划、动作、结果和环境状态，并持续更新以反映任务的进行状态。这种记忆在多步骤任务中尤为重要，因为每个决策都建立在前一个决策之上，要求代理保持对任务轨迹的清晰理解。如图14所示，在完成独立任务时，包含动作和计划的任务轨迹被存储在STM中。这使得代理能够有效跟踪任务进展并做出更明智的决策。

However, STM is constrained by the LLM's context window, limiting the amount of information it can carry forward. To manage this limitation, agents can employ selective memory management strategies, such as selectively discarding or summarizing less relevant details to prioritize the most impactful information. Despite its limited size, STM is essential for ensuring coherent, contextually aware interactions and supporting the agent's capacity to execute complex workflows with immediate, relevant feedback.
然而，STM受限于LLM的上下文窗口，限制了其能够携带的信息量。为应对这一限制，代理可以采用选择性记忆管理策略，例如有选择地丢弃或总结不太相关的细节，以优先保留最重要的信息。尽管容量有限，STM对于确保连贯且具上下文感知的交互至关重要，并支持代理在执行复杂工作流程时获得即时且相关的反馈。

10.12.3 5.6.2 Long-Term Memory

10.12.4 5.6.2 长期记忆

Long-Term Memory (LTM) serves as an external storage repository for contextual information that extends beyond the immediate runtime 204. Unlike STM, which is transient, LTM retains historical task data, including previously completed tasks, successful action sequences, contextual tips, and learned insights. LTM can be stored on disk or in a database, enabling it to retain larger volumes of information than what is feasible within the LLM's immediate context window. In the example shown in Figure 14 when the second task requests downloading a game related to the previous task, the agent retrieves relevant information from its LTM. This enables the agent to accurately identify the correct game, facilitating efficient task completion.
长期记忆（LTM）作为一个外部存储库，用于保存超出即时运行时的上下文信息[204]。与短暂的STM不同，LTM保留历史任务数据，包括先前完成的任务、成功的动作序列、上下文提示和学习到的见解。LTM可以存储在磁盘或数据库中，使其能够保存比LLM即时上下文窗口更大量的信息。如图14所示，当第二个任务请求下载与前一个任务相关的游戏时，代理从其LTM中检索相关信息，从而准确识别正确的游戏，促进高效完成任务。

LTM contributes to the agent's self-improvement over time by preserving examples of successful task trajectories, operational guidelines, and common interaction patterns. When approaching a new task, the agent can leverage RAG techniques to retrieve relevant historical data, which enhances its ability to adapt strategies based on prior success. This is similar to the lifelong learning [53], which makes LTM instrumental in fostering an agent's capacity to "learn" from experience, enabling it to perform tasks with greater accuracy and efficiency

as it accumulates insights across sessions. For instance, [205] provides an illustrative example of using past task trajectories stored in memory to guide and enhance future decision-making, a technique that is highly adaptable for GUI agents. It also enables better personalization by retaining information about previous tasks.
LTM通过保存成功任务轨迹的示例、操作指南和常见交互模式，促进代理随时间的自我提升。在处理新任务时，代理可以利用检索增强生成（RAG）技术检索相关历史数据，增强其基于先前成功调整策略的能力。这类似于终身学习[53]，使LTM成为促进代理“从经验中学习”能力的关键，使其随着跨会话积累的见解，能够更准确高效地执行任务。例如，[205]提供了一个利用存储在记忆中的过去任务轨迹指导和提升未来决策的示例，这一技术对图形用户界面（GUI）代理尤为适用。它还通过保留先前任务的信息，实现更好的个性化。

10.13 5.7 Advanced Enhancements

10.14 5.7 高级增强

While most LLM-brained GUI agents incorporate fundamental components such as perception, planning, action execution, and memory, several advanced techniques have been developed to significantly improve the reasoning and overall capabilities of these agents. Here, we outline shared advancements widely adopted in research to guide the development of more specialized and capable LLM-brained GUI agents.
虽然大多数基于LLM的GUI代理包含感知、规划、动作执行和记忆等基本组件，但已经开发出多种高级技术，显著提升这些代理的推理能力和整体性能。这里我们概述了研究中广泛采用的共享进展，以指导更专业、更强大的基于LLM的GUI代理的开发。

10.14.1 5.7.1 Computer Vision-Based GUI Grounding

10.14.2 5.7.1 基于计算机视觉的GUI定位

Although various tools (Section 4) enable GUI agents to access information like widget location, captions, and properties, certain non-standard GUIs or widgets may not adhere to these tools' protocols [243], rendering their information inaccessible. Additionally, due to permission management, these tools are not always usable. Such incomplete information can present significant challenges for GUI agents, as the LLM may need to independently locate and interact with required widgets by estimating their coordinates to perform actions like clicking—a task that is inherently difficult without precise GUI data.
尽管各种工具（第4节）使GUI代理能够访问控件位置、标题和属性等信息，但某些非标准GUI或控件可能不遵循这些工具的协议[243]，导致其信息无法获取。此外，由于权限管理，这些工具并非总是可用。这种信息不完整对GUI代理构成重大挑战，因为LLM可能需要通过估算控件坐标独立定位并与所需控件交互以执行点击等操作——在缺乏精确GUI数据的情况下，这是一项本质上困难的任务。

CV models offer a non-intrusive solution for GUI grounding directly from screenshots, enabling the detection, localization, segmentation, and even functional estimation of widgets [103], 244-246]. This approach allows agents to interpret the visual structure and elements of the GUI without relying on system-level tools or internal metadata, which may be unavailable or incomplete. CV-based GUI parsing provides agents with valuable insights into interactive components, screen layout, and widget functionalities based solely on visual cues, enhancing their ability to recognize and act upon elements on the screen. Figure 10 provides an illustrative example of how a CV-based GUI parser works. While standard API-based detection captures predefined widgets, the CV model can identify additional elements, such as thumbnails and canvases, which may not have explicit API representations in the PowerPoint interface. This enhances widget recognition, allowing the agent to detect components beyond the scope of API detection. We show an overview of related GUI grounding models and benchmarks in Table 7 8 and 9
计算机视觉（CV）模型提供了一种非侵入式的GUI定位解决方案，直接从截图中实现控件的检测、定位、分割甚至功能估计[103], 244-246]。该方法使代理能够基于视觉线索解读GUI的视觉结构和元素，而无需依赖可能不可用或不完整的系统级工具或内部元数据。基于CV的GUI解析为代理提供了关于交互组件、屏幕布局和控件功能的宝贵见解，提升了其识别和操作屏幕元素的能力。图10展示了基于CV的GUI解析器的示例。标准的基于API的检测捕获预定义控件，而CV模型能够识别额外元素，如缩略图和画布，这些在PowerPoint界面中可能没有明确的API表示。这增强了控件识别能力，使代理能够检测超出API检测范围的组件。相关GUI定位模型和基准的概览见表7、8和9。

A notable example is OmniParser [184], which implements a multi-stage parsing technique involving a fine-tuned model for detecting interactable icons, an OCR module for extracting text, and an icon description model that generates localized semantic descriptions for each UI element. By integrating these components, OmniParser constructs a structured representation of the GUI, enhancing an agent's understanding of interactive regions and functional elements. This comprehensive parsing strategy has shown to significantly improve GPT-4V's screen comprehension and interaction accuracy.
一个显著的例子是OmniParser[184]，它实现了多阶段解析技术，包括用于检测可交互图标的微调模型、用于提取文本的OCR模块以及生成每个UI元素本地语义描述的图标描述模型。通过整合这些组件，OmniParser构建了GUI的结构化表示，增强了代理对交互区域和功能元素的理解。这种综合解析策略显著提升了GPT-4V对屏幕的理解和交互准确性。

Such CV-based GUI grounding layers provide critical grounding information that significantly enhances an agent's ability to interact accurately and intuitively with diverse GUIs. This is particularly beneficial for handling custom or nonstandard elements that deviate from typical accessibility protocols. Additionally, prompting methods like iterative narrowing have shown promise in improving the widget grounding capabilities of VLMs [208]. Together, these approaches pave the way for more adaptable and resilient GUI agents, capable of operating effectively across a broader range of screen environments and application contexts.
基于计算机视觉（CV）的GUI定位层提供了关键的定位信息，显著增强了智能体准确且直观地与多样化GUI交互的能力。这对于处理偏离典型无障碍协议的自定义或非标准元素尤为有益。此外，诸如迭代缩小（iterative narrowing）等提示方法在提升视觉语言模型（VLMs）的小

部件定位能力方面展现出潜力[208]。这些方法共同为更具适应性和鲁棒性的GUI智能体铺平了道路，使其能够在更广泛的屏幕环境和应用场景中高效运行。

Several works have introduced benchmarks to evaluate the GUI grounding capabilities of models and agents. For instance, ScreenSpot [25] serves as a pioneering benchmark designed to assess the GUI grounding performance of LLM-powered agents across diverse platforms, including iOS, Android, macOS, Windows, and web environments. It features a dataset with over 600 screenshots and 1,200 instructions, focusing on complex GUI components such as widgets and icons. This benchmark emphasizes the importance of GUI grounding in enhancing downstream tasks like web automation and mobile UI interaction. Building upon this, ScreenSpot-Pro [241] extends the scope to more professional, high-resolution environments. This evolved version includes 1,581 tasks with high-quality annotations, encompassing domains such as software development, creative tools, CAD, scientific applications, and office productivity. Key features of ScreenSpot-Pro include authentic high-resolution screenshots and meticulous annotations provided by domain experts.
多项研究提出了用于评估模型和智能体GUI定位能力的基准测试。例如，ScreenSpot [25]作为开创性基准，旨在评估基于大型语言模型（LLM）的智能体在iOS、Android、macOS、Windows及网页环境等多平台上的GUI定位表现。其数据集包含600多张截图和1200条指令，重点关注复杂的GUI组件，如小部件和图标。该基准强调GUI定位在提升下游任务（如网页自动化和移动UI交互）中的重要性。在此基础上，ScreenSpot-Pro [241]将范围扩展至更专业的高分辨率环境。该升级版本包含1581个任务，配备高质量注释，涵盖软件开发、创意工具、计算机辅助设计（CAD）、科学应用及办公效率等领域。ScreenSpot-Pro的关键特征包括真实的高分辨率截图和由领域专家提供的细致注释。

These benchmarks provide critical evaluation criteria for assessing GUI grounding capabilities, thereby advancing the development of GUI agents for improved GUI understanding and interaction.
这些基准为评估GUI定位能力提供了关键的评价标准，推动了GUI智能体在GUI理解与交互方面的发展。

10.14.3 5.7.2 Multi-Agent Framework

10.14.4 5.7.2 多智能体框架

The adage "two heads are better than one" holds particular relevance for GUI automation tasks, where a single agent, though capable, can be significantly enhanced within a multi-agent framework [247], [248]. Multi-agent systems leverage the collective intelligence, specialized skills, and complementary strengths of multiple agents to tackle complex tasks more effectively than any individual agent could alone. In the context of GUI agents, multi-agent systems offer advanced capabilities through two primary mechanisms: (i) specialization and (ii) inter-agent collaboration. Figure 15 illustrates an example of how an LLM-powered multi-agent collaborates to create a desk.
“三个臭皮匠，赛过诸葛亮”这一谚语在GUI自动化任务中尤为适用，单个智能体虽具能力，但在多智能体框架中可显著增强[247]，[248]。多智能体系统利用多个智能体的集体智慧、专业技能和互补优势，比单一智能体更有效地完成复杂任务。在GUI智能体的背景下，多智能体系统通过两大机制提供先进能力：（i）专业化和（ii）智能体间协作。图15展示了一个基于大型语言模型（LLM）的多智能体协作创建桌面的示例。

1. Specialization of Agents: In a multi-agent framework, each agent is designed to specialize in a specific role or function, leveraging its unique capabilities to contribute to the overall task. As illustrated in the Figure 15 specialization enables distinct agents to focus on different aspects of the task pipeline. For instance, the "Document Extractor" specializes in extracting relevant content from local documents, such as PDFs, while the "Web Retriever" focuses on gathering additional information from online sources. Similarly, the "Designer" transforms the retrieved information into visually appealing slides, and the "Evaluator" provides feedback to refine and improve the output. This functional separation ensures that each agent becomes highly adept at its designated task, leading to improved efficiency and quality of results 249].
2. 智能体专业化：在多智能体框架中，每个智能体被设计为专注于特定角色或功能，利用其独特能力为整体任务做出贡献。如图15所示，专业化使不同智能体专注于任务流程的不同环节。例如，“文档提取器”专门负责从本地文档（如PDF）中提取相关内容，而“网页检索器”则专注于从在线资源收集额外信息。同样，“设计师”将检索的信息转化为视觉吸引力强的幻灯片，“评估者”则提供反馈以优化和改进输出。这种功能分工确保每个智能体在其指定任务上高度熟练，从而提升效率和结果质量[249]。

TABLE 7: A summary of of GUI grounding models (Part I).
表7：GUI定位模型汇总（第一部分）。

Model/ Benchmark	Platform	Foundation Model	Size	Dataset	Input	Output	Highlight	Link
OmniParser [184]	Mobile, Desktop, and Web	BLIP-2 206] YOLOv8 207]	/	67,000 UI screenshots with bounding box annotations and 7,185 icon-description pairs generated using GPT-4	UI screenshots	IDs, bounding boxes, and descriptions of in-teractable elements	Introduces a purely vision-based screen parsing framework for general UI understanding without external information, significantly improving action prediction accuracy for LLM-driven agents	https://github.com/microsoft/OmniParser
Iterative Narrowing [208]	Mobile, Web, and Desktop	Qwen2-VL and OS-Atlas-Base	/	ScreenSpot 25]	A GUI screenshot and a natural language query	(x,y) coordinates representing the target location in the GUI	Progressively crops regions of the GUI to refine predictions, enhancing precision for GUI ground. ing tasks	https://github. com/ant-8/ GUI-Grounding-via-Iterative-Narro
Iris 209]	Mobile (iOS, Android), Desktop (Windows, macOS), and Web	Qwen-VL 210]	9.6B	850K GUI-specific annotations and 150K vision-language instructions	High-resolution GUI screenshots with natural language instructions	Referring: Generates detailed descriptions of UI elements. Grounding: Locates U elements on the screen.	Information-Sensitive Cropping for efficient handling of high-resolution GUI images, and Self-Refining Dual Learning to iteratively enhance GUI grounding and referring tasks without additional annotations	/
Attention-driven Grounding 211]	Mobile, Web, and Desktop	MiniCPM-Llama3-V 2.5	8.5B	Mind2Web ScreenSpot VisualWebBench 213]	shots textual queries GUI screen-and user	Element localization via bounding boxes, text-to-image mapping for grounding. and actionable descriptions of GUI components	Utilizes attention mechanisms in pre-trained MLLMs without fine-tuning	https: //github.com/ HeimingX/TAG
Aria-UI 214]	Web, Desktop, and Mobile	Aria 215]	3.9B	3.9 million elements and 11.5 million samples	GUI screen-shots, user instructions, and action histories	Pixel coordinates for GUI elements and corresponding actions	A purely vision-based approach avoiding reliance on AXTree-like inputs	https: //ariaui.github.io

UGround 216]	Web, Desktop (Windows, MacOS, Linux), Mobile (Android, iOS)	LLaVA-NeXT-7B 217]	7B	Web-Hybrid and other existing datasets	queries GUI screen-shots, user	Pixel coordinates of GUI elements	A universal GUI grounding model that relies solely on vision, eliminating the need for text-based representations	https: //osu-nlp-group. github.io/ UGround/
GUI-Bee 218]	Web	SeeClick 251. QwenGUI 219], and UIX-7B 220]	7B-13B	NovelScreenSpot	GUI screen-shots, user queries. accessibility tree	GUI element grounding locations, actions and function calls, navigation steps, predicted GUI changes after interaction	Autonomously explores GUI environments, with Q-ICRL optimizing exploration efficiency and enhancing data diversity.	https://qui-bee.github.io

模型/基准	平台	基础模型	规模	数据集	输入	输出	亮点	链接
OmniParser [184]	移动端、桌面端和网页端	BLIP-2 206] YOLOv8 207]	/	67,000张带有边界框标注的UI截图和7,185对由GPT-4生成的图标描述对	UI截图	可交互元素的ID、边界框和描述	提出了一种纯视觉的屏幕解析框架，用于通用UI理解，无需外部信息，显著提升了基于大语言模型(LLM)驱动代理的动作预测准确率	https://github.com/microsoft/OmniParser
迭代缩小法 [208]	移动端、网页端和桌面端	Qwen2-VL 和 OS-Atlas-Base	/	ScreenSpot 25]	一张GUI截图和一条自然语言查询	表示GUI中目标位置的(x,y)坐标	逐步裁剪GUI区域以细化预测，提升GUI定位任务的精度	https://github.com/ant-8/GUI-Grounding-via-Iterative-Narro
Iris 209]	移动端（iOS、Android）、桌面端（Windows、macOS）和网页端	Qwen-VL 210]	9.6B	85万条GUI专用标注和15万条视觉语言指令	高分辨率GUI截图及自然语言指令	指称：生成UI元素的详细描述。定位：在屏幕上定位UI元素。	信息敏感裁剪以高效处理高分辨率GUI图像，自我优化双重学习迭代提升GUI定位和指称任务，无需额外标注	/
基于注意力的定位 211]	移动端、网页端和桌面端	MiniCPM-Llama3-V 2.5	8.5B	Mind2Web ScreenSpot VisualWebBench 213]	截图、文本查询、GUI屏幕和用户	通过边界框实现元素定位，文本到图像映射用于定位，以及GUI组件的可操作描述	利用预训练多模态大语言模型(MLLM)中的注意力机制，无需微调	https://github.com/HeimingX/TAG
Aria-UI 214]	网页端、桌面端和移动端	Aria 215]	3.9B	390万个元素和1150万个样本	GUI截图、用户指令和操作历史	GUI元素的像素坐标及对应操作	纯视觉方法，避免依赖AXTree类输入	https://ariaui.github.io
UGround 216]	网页端、桌面端（Windows、MacOS、Linux）、移动端（Android、iOS）	LLaVA-NeXT-7B 217]	7B	Web-Hybrid及其他现有数据集	查询、GUI截图、用户	GUI元素的像素坐标	一种通用的GUI定位模型，仅依赖视觉，消除对基于文本表示的需求	https://osu-nlp-group.github.io/UGround/
GUI-Bee 218]	网页端	SeeClick 251. QwenGUI 219], 和 UIX-7B 220]	7B-13B	NovelScreenSpot	GUI截图、用户查询、辅助功能树	GUI元素定位位置、操作和函数调用、导航步骤、交互后预测的GUI变化	自主探索GUI环境，Q-ICRL优化探索效率并增强数据多样性。	https://qui-bee.github.io

2. Collaborative Inter-Agent Dynamics: The multi-agent system shown in the Figure 15 exemplifies how agents collaborate dynamically to handle complex tasks. The process begins with the "Document Extractor" and "Web Retriever", which work in parallel to collect information from local and online sources. The retrieved data is communicated to the "Designer", who synthesizes it into a cohesive set of slides. Once the slides are created, the "Evaluator" reviews the output, providing feedback for refinement. These agents share information, exchange context, and operate in a coordinated manner, reflecting a human-like teamwork dynamic. For example, as

depicted, the agents' roles are tightly integrated—each output feeds into the next stage, creating a streamlined workflow that mirrors real-world collaborative environments [19].

3. 协作式多智能体动态：图15所示的多智能体系统展示了智能体如何动态协作以处理复杂任务。该过程始于“文档提取器”和“网络检索器”，它们并行工作，从本地和在线资源收集信息。检索到的数据传递给“设计师”，由其将信息综合成一套连贯的幻灯片。幻灯片创建完成后，“评估者”对输出进行审查，提供改进反馈。这些智能体共享信息、交换上下文，并协调运作，体现了类似人类团队合作的动态。例如，如图所示，智能体的角色紧密集成——每个输出都作为下一阶段的输入，形成了一个流畅的工作流程，反映了现实协作环境[19]。

In such a system, agents can collectively engage in tasks requiring planning, discussion, and decision-making. Through these interactions, the system taps into each agent's domain expertise and latent potential for specialization, maximizing overall performance across diverse, multi-step processes.
在这样的系统中，智能体可以共同参与需要规划、讨论和决策的任务。通过这些交互，系统能够利用每个智能体的领域专长和潜在的专业化能力，最大化在多样化、多步骤流程中的整体性能。

10.14.5 5.7.3 Self-Reflection

10.14.6 5.7.3 自我反思

"A fault confessed is half redressed". In the context of GUI multi-agent systems, self-reflection refers to the agents' capacity to introspectively assess their reasoning, actions, and decisions throughout the task execution process [250]. This capability allows agents to detect potential mistakes, adjust strategies, and refine actions, thereby improving the quality and robustness of their decisions, especially in complex or unfamiliar GUI environments. By periodically evaluating their own performance, self-reflective agents can adapt dynamically to produce more accurate and effective results [251].
“知错能改，善莫大焉”。在GUI多智能体系统中，自我反思指的是智能体在任务执行过程中能够内省性地评估其推理、行为和决策的能力[250]。这一能力使智能体能够发现潜在错误，调整策略，优化行为，从而提升决策的质量和鲁棒性，尤其是在复杂或不熟悉的GUI环境中。通过定期评估自身表现，自我反思的智能体能够动态适应，产生更准确有效的结果[251]。

TABLE 8: A summary of of GUI grounding models (Part II).
表8：GUI基础模型总结（第二部分）。

Model/ Benchmark	Platform	Foundation Model	Size	Dataset	Input	Output	Highlight	Link
RWKV-UI 221]	Web	1.6B	SIGLIP [222], DINOv2 223], SAM [182]	Websight WebUI-7kbal 225 Web2Code 226	High-resolution webpage images	Element grounding, Action prediction, CoT reasoning	Introduces a high-resolution three-encoder architecture with visual prompt engineering and CoT reasoning	/
TRISHUL 227]	Web, Desktop, and Mobile platforms	/ (Training-Free)	/ (Training-Free)	/ (Training-Free)	GUI Screen-shots, user instruction-s/queries. hierarchical screen parsing outputs, OCR-extracted text descriptors	Action grounding, functionality descriptions of GUI elements. GUI referring, and SoMs	Utilizes hierarchical screen parsing and spatially enhanced element descriptions to enhance LVLMs without additional training	/
AutoGUI 228]	Web, Mobile	Qwen-VL-10B 210 SliME-8B 229]	10B / 8B	AutoGUI-704k	GUI screen-shots, User queries	Element functionalities, Element locations	Automatically labels UI elements based on interaction-induced changes, making it scalable and high-quality.	https:// autogui-project. github.io/
Query Inference [230]	Mobile Android	Qwen2-VL-7B-Instruct 231]	7B	UIBERT 232]	GUI shots	Action-oriented queries, Coordinates	Improves reasoning without requiring large-scale training data	https://github.com/ZrW00/ GUIPivot
WinClick 233]	Windows OS	Phi3-Vision 234]	4.2B	WinSpot Benchmark	GUI screen-shots, Natural language instructions	Element locations	The first GUI ground ing model specifically tailored for Windows.	https://github.com/zackhuiliii/ WinSpot
FOCUS 235]	Web, mobile applications, and desktop	Qwen2-VL-2B-Instruct 231]	2B	GUICourse Aguvis-stage1 236] Wave-UI Desktop-UI 238]	GUI screenshot + task instruction	Normalized coordinates (x, y)	A dual-system GUI grounding architecture inspired by human cognition, whic dynamically switches between fast (intuitive) and slow (analytical) grounding modes based on task complexity	https: //github.com/ sugarandgugu/ Focus

UI-E2I-Synth 239]	Web. Windows, and Android	InternVL2-4B and Qwen2-VL-7B	4B and 7B	1.6M screenshots, 9.9M instructions	GUI screenshot	Element coordinates	Introduces a three-stage synthetic data pipeline for GUI grounding with both explicit and implicit instruction synthesis	https: //colmon46. github.io/ i2e-bench-leaderboar
RegionFocus 240]	Web-based and Desktop interfaces	UI-TARS and Qwen2.5-VL	72B	None (test-time only)	GUI screen-shots with a point of interest	Coordinate-based actions	Introduces a visua test-time scaling framework that zooms into salient UI regions and integrates an image as-map mechanism to track history and avoid repeated mistakes—boosting grounding accuracy without model retraining	https://github.com/tiangeluo/ RegionFocus

模型/基准	平台	基础模型	规模	数据集	输入	输出	亮点	链接
RWKV-UI 221]	网页	1.6B	SIGLIP [222], DINOv2 223], SAM [182]	Websight WebUI-7kbal 225 Web2Code 226	高分辨率网页图像	元素定位，动作预测，链式推理（CoT）	引入高分辨率三编码器架构，结合视觉提示工程和链式推理（CoT）	/
TRISHUL 227]	网页、桌面和移动平台	/（无训练）	/（无训练）	/（无训练）	GUI截图，用户指令/查询，分层屏幕解析输出，OCR提取的文本描述	动作定位，GUI元素功能描述，GUI引用及状态模型（SoMs）	利用分层屏幕解析和空间增强元素描述，无需额外训练提升大视觉语言模型（LVLMs）性能	/
AutoGUI 228]	网页，移动端	Qwen-VL-10B 210 SliME-8B 229]	10B / 8B	AutoGUI-704k	GUI截图，用户查询	元素功能，元素位置	基于交互引发的变化自动标注UI元素，实现可扩展且高质量的标注。	https:// autogui-project.github.io/
Query Inference [230]	安卓移动端	Qwen2-VL-7B-Instruct 231]	7B	UIBERT 232]	GUI截图	面向动作的查询，坐标	提升推理能力，无需大规模训练数据	https://github.com/ZrW00/GUIPivot
WinClick 233]	Windows操作系统	Phi3-Vision 234]	4.2B	WinSpot基准	GUI截图，自然语言指令	元素位置	首个专为Windows定制的GUI定位模型。	https://github.com/zackhuiliii/WinSpot
FOCUS 235]	网页、移动应用及桌面	Qwen2-VL-2B-Instruct 231]	2B	GUICourse Aguvis-stage1 236] Wave-UI Desktop-UI 238]	GUI截图+任务指令	归一化坐标（x, y）	受人类认知启发的双系统GUI定位架构，根据任务复杂度动态切换快速（直觉）与慢速（分析）定位模式	https: //github.com/sugarandgugu/ Focus
UI-E2I-Synth 239]	网页、Windows和安卓	InternVL2-4B 和 Qwen2-VL-7B	4B 和 7B	160万截图，990万指令	GUI截图	元素坐标	引入三阶段合成数据流水线，用于GUI定位，结合显式与隐式指令合成	https://colmon46.github.io/i2e-bench-leaderboar
RegionFocus 240]	基于网页和桌面的界面	UI-TARS 和 Qwen2.5-VL	72B	无（仅测试时使用）	带有兴趣点的图形用户界面截图	基于坐标的操作	引入了一种视觉测试时缩放框架，能够放大显著的用户界面区域，并集成图像地图机制以跟踪历史，避免重复错误——在无需模型重新训练的情况下提升定位准确性	https://github.com/tiangeluo/RegionFocus

TABLE 9: A summary of of GUI grounding benchmarks.

表9：GUI定位基准的总结。

Benchmark	Platform	Dataset	Input	Output	Highlight	Link
ScreenSpot 25]	iOS, Android, macOS, and Windows	Over 600 screenshots and 1,200 instructions	GUI screenshots accompanied by user instructions	Bounding boxes or coordinates of actionable GUI elements	A realistic and diverse GUI grounding benchmark covering multiple platforms and a variety of elements	https://github.com/njucckevin/SeeClick
ScreenSpot-Pro 241]	Windows, macOS, Linux	1,581 instruction-screenshot pairs covering 23 applications across 5 industries and 3 operating systems	High-resolution GUI screen-shots paired with natural language instructions	Bounding boxes for locating target UI elements	Introduces a high-resolution benchmark for professional environments	https: //github.com/ likaixin2000/ ScreenSpot-Pro-GUI-Grounding
PixelWeb 242]	Web	100,000 webpages	Rendered webpage screenshots and information	BBox, mask, contour	The first GUI dataset to provide pixel-level annotations—including mask and contour—for web UIs, enabling high-precision GUI grounding and detection tasks	https: //huggingface. co/datasets/ cyberalchemist/ PixelWeb

基准测试	平台	数据集	输入	输出	高亮	链接
ScreenSpot 25]	iOS、Android、macOS 和 Windows	超过600张截图和1200条指令	带有用户指令的图形用户界面（GUI）截图	可操作GUI元素的边界框或坐标	涵盖多平台和多种元素的真实多样化GUI定位基准	https://github.com/njucckevin/SeeClick
ScreenSpot-Pro 241]	Windows、macOS、Linux	涵盖5个行业、3个操作系统中23个应用的1581对指令-截图	高分辨率GUI截图与自然语言指令配对	用于定位目标UI元素的边界框	引入面向专业环境的高分辨率基准	https: //github.com/ likaixin2000/ ScreenSpot-Pro-GUI-Grounding
PixelWeb 242]	网页	100,000个网页	渲染的网页截图及信息	边界框、掩码、轮廓	首个提供像素级注释（包括掩码和轮廓）的网页GUI数据集，支持高精度GUI定位与检测任务	https: //huggingface. co/datasets/ cyberalchemist/ PixelWeb

Task: Create a desk for LLM-based multi-agent system.
任务：为基于大型语言模型（LLM）的多智能体系统创建一张桌子。

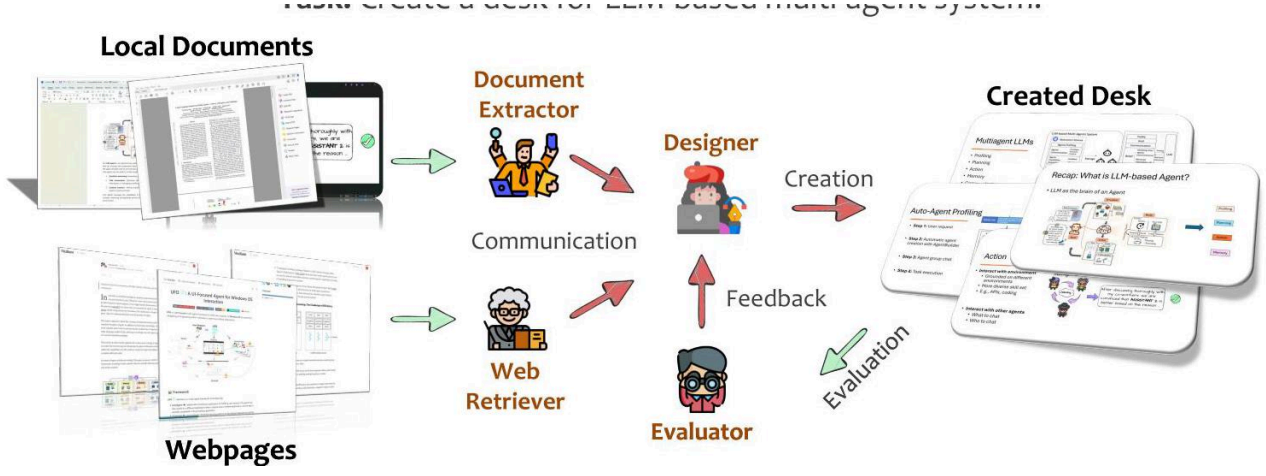

Fig. 15: An example of multi-agent system collaboration in creating a desk.
图15：多智能体系统协作创建桌子的示例。

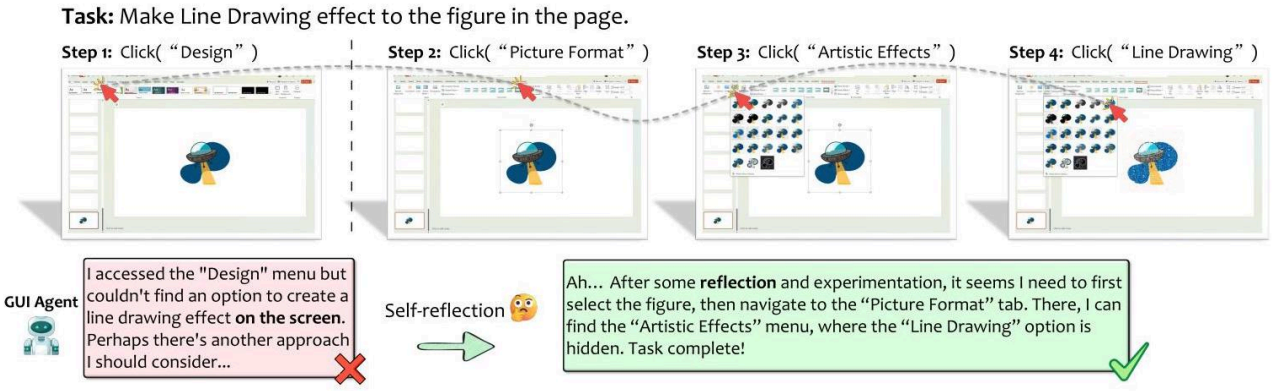

Fig. 16: An example of self-reflection in task completion of an LLM-powered GUI agent.
图16：基于LLM的GUI智能体在完成任务中的自我反思示例。

Self-reflection is particularly critical for GUI agents due to the variable nature of user interfaces and the potential for errors, even in human-operated systems. GUI agents frequently encounter situations that deviate from expectations, such as clicking the wrong button, encountering unexpected advertisements, navigating unfamiliar interfaces, receiving error messages from API calls, or even responding to user feedback on task outcomes. To ensure task success, a GUI agent must quickly reflect on its actions, assess these feedback signals, and adjust its plans to better align with the desired objectives.
自我反思对于GUI智能体尤为关键，因为用户界面的多变性及潜在错误，即使在人类操作系统中也常见。GUI智能体经常遇到偏离预期的情况，如点击错误按钮、遇到意外广告、导航陌生界面、API调用返回错误信息，甚至需要回应用户对任务结果的反馈。为确保任务成功，GUI智能体必须迅速反思自身行为，评估这些反馈信号，并调整计划以更好地符合预期目标。

As illustrated in Figure 16 when the agent initially fails to locate the "Line Drawing" option in the Design menu, self-reflection enables it to reconsider and identify its correct location under Artistic Effects" in the "Picture Format" menu, thereby successfully completing the task.
如图16所示，当智能体最初未能在“设计”菜单中找到“线条绘制”选项时，自我反思使其重新考虑并识别出该选项实际位于“图片格式”菜单下的“艺术效果”中，从而成功完成任务。

In practice, self-reflection techniques for GUI agents typically involve two main approaches: (i) ReAct [252] and (ii) Reflexion [253].
在实践中，GUI智能体的自我反思技术通常包括两种主要方法：（i）ReAct [252] 和（ii）Reflexion [253]。

1. ReAct (Reasoning and Acting): ReAct integrates self-reflection into the agent's action chain by having the agent evaluate each action's outcome and reason about the next best step. In this framework, the agent doesn't simply follow a linear sequence of actions; instead, it adapts dynamically, continuously reassessing its strategy in response to feedback from each action. For example, if a GUI agent attempting to fill a form realizes it has clicked the wrong field, it can adjust by backtracking and selecting the correct element. Through ReAct, the agent achieves higher consistency and accuracy, as it learns to refine its behavior with each completed step.
1）ReAct（推理与行动）：ReAct通过让智能体评估每个动作的结果并推理下一步最佳行动，将自我反思融入智能体的动作链中。在该框架下，智能体不再简单执行线性动作序列，而是动态适应，持续根据每个动作的反馈重新评估策略。例如，当GUI智能体尝试填写表单时发现点击了错误字段，它可以通过回溯并选择正确元素来调整。通过ReAct，智能体实现了更高的一致性和准确性，学会在每一步完成后不断优化行为。

2. Reflexion: Reflexion emphasizes language-based feedback, where agents receive and process feedback from the environment as linguistic input, referred to as self-reflective feedback. This feedback is contextualized and used as input in subsequent interactions, helping the agent to learn rapidly from prior mistakes. For instance, if a GUI agent receives an error message from an application, Reflexion enables the agent to process this message, update its understanding of the interface, and avoid similar mistakes in future interactions. Reflexion's iterative feedback loop promotes continuous improvement and is particularly valuable for GUI agents navigating complex, multi-step tasks. Overall, self-reflection serves as an essential enhancement in GUI multi-agent systems, enabling agents to better navigate the variability and unpredictability of GUI environments. This introspective capability not only boosts individual agent performance but also promotes resilience, adaptability, and long-term learning in a collaborative setting.
2）Reflexion：Reflexion强调基于语言的反馈，智能体将环境反馈作为语言输入进行处理，称为自我反思反馈。该反馈具有上下文信息，并作为后续交互的输入，帮助智能体快速从先前错误中学习。例如，当GUI智能体收到应用程序的错误信息时，Reflexion使其能够处理该信息，更新对界面的理解，避免未来出现类似错误。Reflexion的迭代反馈循环促进持续改进，对于处理复杂多步骤任务的GUI智能体尤为重要。总体而言，自我反思是GUI多智能体系统的重要增强功能，使智能体更好地应对GUI环境的多变性和不可预测性。这种内省能力不仅提升了单个智能体的表现，还促进了协作环境中的韧性、适应性和长期学习。

10.14.7 5.7.4 Self-Evolution

10.14.8 5.7.4 自我进化

Self-evolution [254] is a crucial attribute that GUI agents should possess, enabling them to enhance their performance progressively through accumulated experience. In the context of GUI multi-agent systems, self-evolution allows not only individual agents to improve but also facilitates collective learning and adaptation by sharing knowledge and strategies among agents. During task execution, GUI agents generate detailed action trajectories accompanied by complementary information such as environment states, internal reasoning processes (the agent's thought processes), and evaluation results. This rich data serves as a valuable knowledge base from which GUI agents can learn and evolve. The knowledge extracted from this experience can be categorized into three main areas:
自我进化[254]是GUI智能体应具备的重要特性，使其能够通过积累经验逐步提升性能。在GUI多智能体系统中，自我进化不仅促进个体智能体的改进，还通过智能体间共享知识和策略，推动集体学习与适应。任务执行过程中，GUI智能体生成详细的动作轨迹，并附带环境状态、内部推理过程（智能体的思考过程）及评估结果等补充信息。这些丰富数据构成了宝贵的知识库，供GUI智能体学习和进化。经验中提取的知识主要可分为三大类：

1. Task Trajectories: The sequences of actions executed by agents, along with the corresponding environment states, are instrumental for learning [255]. These successful trajectories can be leveraged in two significant ways. First, they can be used to fine-tune the core LLMs that underpin the agents. Fine-tuning with such domain-specific and task-relevant data enhances the model's ability to generalize and improves performance on similar tasks in the future. Second, these trajectories can be utilized as demonstration examples to activate the in-context learning capabilities of LLMs during prompt engineering. By including examples of successful task executions in the prompts, agents can better understand and replicate the desired behaviors without additional model training.
   1）任务轨迹：智能体执行的动作序列及对应的环境状态对学习至关重要[255]。这些成功轨迹有两大重要用途。首先，可用于微调支撑智能体的核心大型语言模型（LLM）。利用此类领域特定且任务相关的数据进行微调，提升模型的泛化能力和未来类似任务的表现。其次，这些轨迹可作为示范样例，在提示工程中激活LLM的上下文学习能力。通过在提示中包含成功任务执行的示例，智能体无需额外训练即可更好地理解和复制期望行为。

For instance, suppose an agent successfully completes a complex task that involves automating data entry across multiple applications. The recorded action trajectory-comprising the steps taken, decisions made, and contextual cues—can be shared with other agents. These agents can then use this trajectory as a guide when faced with similar tasks, reducing the learning curve and improving efficiency.
例如，假设某智能体成功完成了涉及跨多个应用自动化数据录入的复杂任务。记录的动作轨迹——包括所采取的步骤、做出的决策及上下文线索——可以与其他智能体共享。其他智能体在面对类似任务时，可将此轨迹作为指导，缩短学习曲线，提高效率。

2. Guidance and Rules: From the accumulated experiences, agents can extract high-level rules or guidelines that encapsulate best practices, successful strategies, and lessons learned from past mistakes [256], [257]. Such guidance can be acquired by the LLM itself through trajectory summarization [256], or even via search-based algorithms, such as Monte Carlo Tree Search (MCTS) [257]. This knowledge can be formalized into policies or heuristics that agents consult during decision-making processes, thereby enhancing their reasoning capabilities.
   2）指导与规则：智能体可从积累的经验中提炼出高层次规则或指导方针，涵盖最佳实践、成功策略及从过去错误中汲取的教训[256]，[257]。此类指导可通过轨迹总结由LLM自身获得[256]，或通过基于搜索的算法如蒙特卡洛树搜索（MCTS）[257]获取。该知识可形式化为智能体在决策过程中参考的策略或启发式规则，从而增强其推理能力。

For example, if agents repeatedly encounter errors when attempting to perform certain actions without proper prerequisites (e.g., trying to save a file before specifying a file path), they can formulate a rule to check for these prerequisites before executing the action. This proactive approach reduces the likelihood of errors and improves task success rates.
例如，如果代理在尝试执行某些操作时反复遇到错误，原因是缺少必要的前置条件（例如，在指定文件路径之前尝试保存文件），它们可以制定一条规则，在执行操作前检查这些前置条件。这种主动的方法减少了错误发生的可能性，提高了任务成功率。

3. New Toolkits: Throughout their interactions, GUI agents may discover or develop more efficient methods, tools, or sequences of actions that streamline task execution 161]. These may include optimized API calls, macros, or combinations of UI operations that accomplish tasks more effectively than previous approaches. LLMs can be leveraged to automatically analyze execution trajectories in order to summarize, discover, and generate high-level shortcuts or frequently used fast APIs, which can then be reused for future executions [258]. By incorporating these new tools into their repertoire, agents expand their capabilities and enhance overall efficiency.
4. 新工具包：在交互过程中，GUI代理可能会发现或开发出更高效的方法、工具或操作序列，从而简化任务执行[161]。这些可能包括优化的API调用、宏命令或UI操作的组合，能够比以往方法更有效地完成任务。可以利用大型语言模型（LLMs）自动分析执行轨迹，以总结、发现并生成高级快捷方式或常用快速API，供未来执行时重复使用[258]。通过将这些新工具纳入其技能库，代理扩展了能力，提升了整体效率。

As an example, an agent might find that using a batch processing API can automate repetitive tasks more efficiently than performing individual UI operations in a loop. This new approach can be shared among agents within the multi-agent system, allowing all agents to benefit from the improved method and apply it to relevant tasks.
举例来说，代理可能发现使用批处理API比循环执行单个UI操作更高效地自动化重复任务。这种新方法可以在多代理系统中共享，使所有代理都能受益于改进的方法，并将其应用于相关任务。

Figure 17 illustrates how a GUI agent evolves through task completion. During its operations, the agent adds new capabilities to its skill set, such as an image summarization toolkit, gains insights from reading a paper on creating GUI agents, and stores task trajectories like webpage extraction in its experience pool. When assigned a new task, such as "Learn to make a GUI agent from a GitHub repository", the agent draws on its acquired skills and past experiences to adapt and perform effectively.
图17展示了GUI代理通过完成任务而进化的过程。在操作过程中，代理向其技能集添加了新能力，如图像摘要工具包，从阅读关于创建GUI代理的论文中获得见解，并将网页提取等任务轨迹存储在经验库中。当被分配新任务，如“从GitHub仓库学习制作GUI代理”时，代理会利用其已获得的技能和过往经验进行适应并高效执行。

This dynamic evolution highlights the agent's ability to continually learn, grow, and refine its capabilities. By leveraging past experiences, incorporating new knowledge, and expanding its toolset, GUI agents can adapt to diverse challenges, improve task execution, and significantly enhance the overall performance of the system, fostering a collaborative and ever-improving environment.
这种动态进化凸显了代理持续学习、成长和完善能力的能力。通过利用过去的经验、整合新知识并扩展工具集，GUI代理能够适应多样化的挑战，提升任务执行效果，显著增强系统整体性能，促进协作与持续改进的环境。

10.14.9 5.7.5 Reinforcement Learning

10.14.10 5.7.5 强化学习

Reinforcement Learning (RL) [259] has witnessed significant advancements in aligning LLMs with desired behaviors [260], and has recently been employed in the development of LLM agents [50], [261]. In the context of GUI multi-agent systems, RL offers substantial potential to enhance the performance, adaptability, and collaboration of GUI agents. GUI automation tasks naturally align with the structure of a Markov Decision Process (MDP) [262], making them particularly well-suited for solutions based on RL. In this context, the state corresponds to the environment perception (such as GUI screenshots, UI element properties, and layout configurations), while actions map directly to UI operations, including mouse clicks, keyboard inputs, and API calls. Rewards can be explicitly defined based on various performance metrics, such as task completion, efficiency, and accuracy, allowing the agent to optimize its actions for maximal effectiveness. Figure 18 illustrates an example of MDP modeling for task completion in a GUI agent, where state, action and reward are clearly defined.
强化学习（Reinforcement Learning, RL）[259]在使大型语言模型（LLMs）行为符合预期方面取得了显著进展[260]，并且最近被应用于LLM代理的开发[50],[261]。在GUI多代理系统中，RL具有显著潜力提升GUI代理的性能、适应性和协作能力。GUI自动化任务天然符合马尔可夫决策过程（Markov Decision Process, MDP）[262]的结构，因而特别适合基于RL的解决方案。在此背景下，状态对应环境感知（如GUI截图、UI元素属性和布局配置），动作直接映射到UI操作，包括鼠标点击、键盘输入和API调用。奖励可以基于多种性能指标明确定义，如任务完成度、效率和准确性，使代理能够优化其动作以实现最大效能。图18展示了GUI代理任务完成的MDP建模示例，其中状态、动作和奖励均有明确界定。

By formulating GUI agent interactions as an MDP, we can leverage RL techniques to train agents that learn optimal policies for task execution through trial and error [263]. This approach enables agents to make decisions that maximize cumulative rewards over time, leading to more efficient and effective task completion. For example, an agent learning to automate form filling in a web application can use RL to discover the most efficient sequence of actions to input data and submit the form successfully, minimizing errors and redundant steps. This process helps align the agents more closely with desired behaviors in GUI automation tasks, especially in complex or ambiguous situations where predefined action sequences are insufficient.
通过将GUI代理交互形式化为MDP，我们可以利用RL技术训练代理，通过试错学习最优策略以执行任务[263]。该方法使代理能够做出最大化累积奖励的决策，从而实现更高效、更有效的任务完成。例如，学习自动填写网页表单的代理可以利用RL发现输入数据和提交表单的最优动作序列，减少错误和冗余步骤。此过程有助于使代理在GUI自动化任务中更贴合预期行为，尤其在预定义动作序列不足以应对复杂或模糊情境时表现突出。

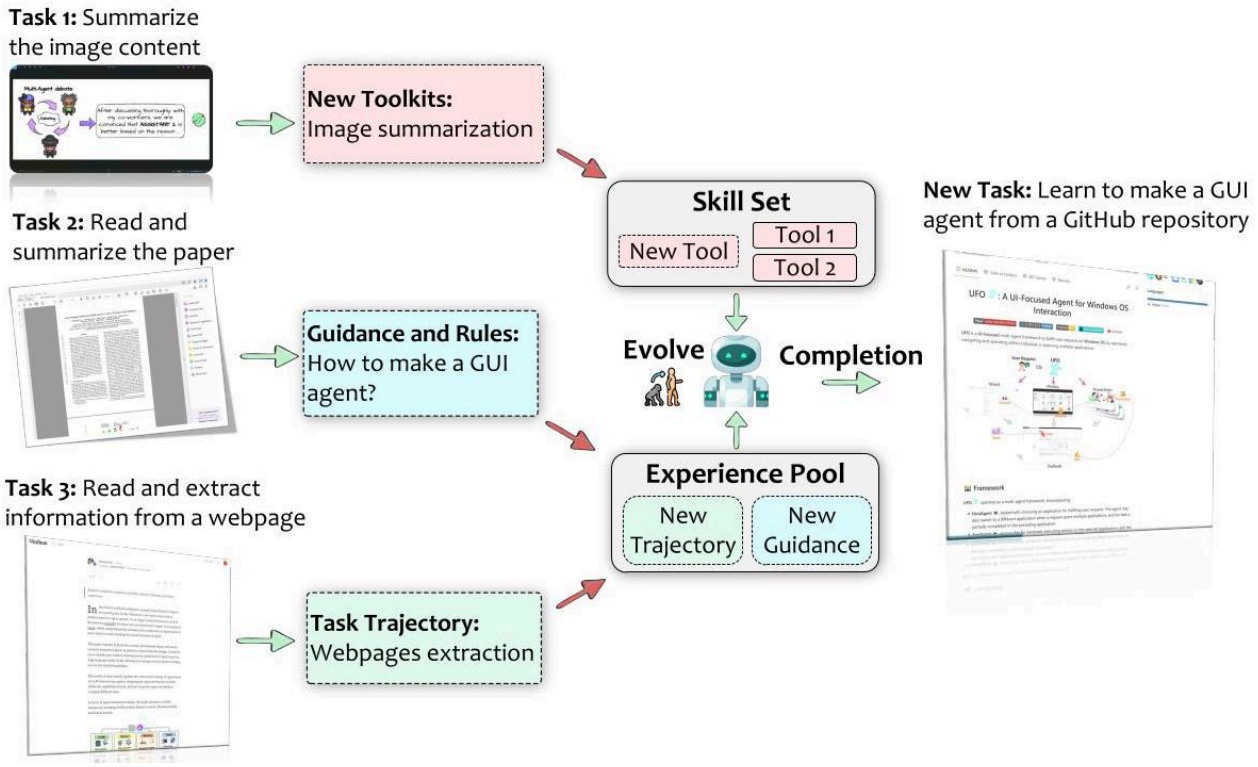

Fig. 17: An example self-evolution in a LLM-powered GUI agent with task completion.
图17：一个基于大型语言模型的GUI代理通过任务完成实现自我进化的示例。

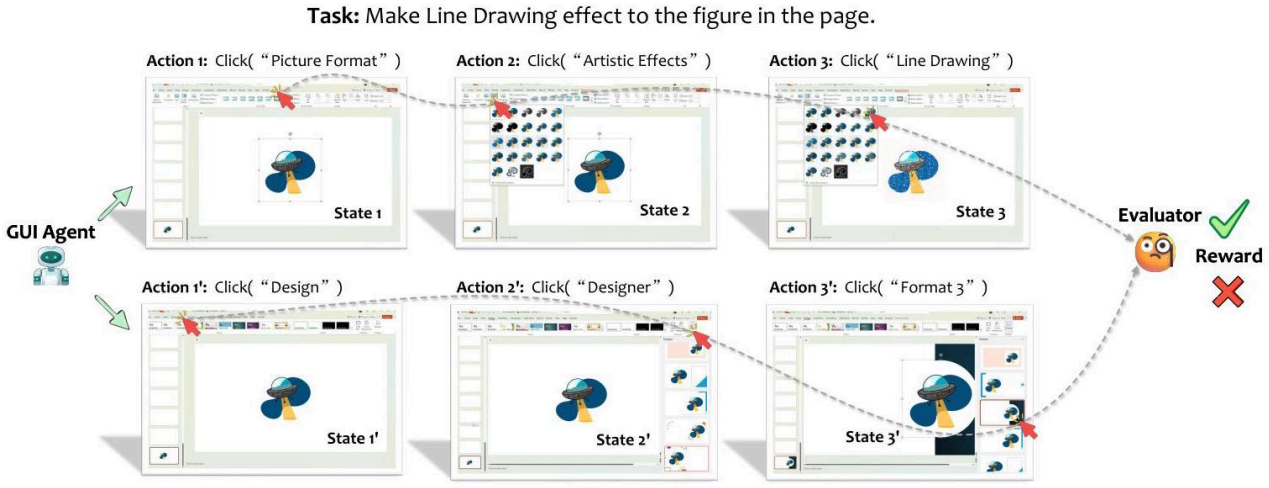

Fig. 18: An example of MDP modeling for task completion in a GUI agent.
图18：GUI代理任务完成的MDP建模示例。

As a representative approach, Bai et al., introduce DigiRL 264 , a two-phase RL framework for training GUI agents in dynamic environments. DigiRL begins with an offline RL phase that uses offline data to initialize the agent model, followed by online fine-tuning, where the model interacts directly with an environment to refine its strategies through live data within an Android learning environment using an LLM evaluator that provides reliable reward signals. This adaptive setting enables the agent to learn and respond effectively to the complexities of dynamic GUIs. Wang et al., propose DistRL 265], an RL fine-tuning pipeline specifically designed for on-device mobile control agents operating within Android. DistRL employs an asynchronous architecture, deploying RL fine-tuned agents across heterogeneous worker devices and environments for decentralized data collection. By leveraging off-policy RL techniques, DistRL enables centralized training with data gathered remotely from diverse environments, significantly enhancing the scalability and robustness of the model. These representative methods illustrate the potential of RL to improve GUI agents, demonstrating how both centralized and distributed RL frameworks can enable more responsive, adaptable, and effective GUI automation models in real-world applications.
作为代表性方法，Bai等人提出了DigiRL[264]，这是一种用于动态环境中训练GUI代理的两阶段RL框架。DigiRL首先进行离线RL阶段，利用离线数据初始化代理模型，随后进行在线微调，模型通过与环境的直接交互，在Android学习环境中利用提供可靠奖励信号的LLM评估

器，通过实时数据优化策略。这种自适应设置使代理能够有效学习并应对动态GUI的复杂性。Wang等人提出了DistRL[265]，这是专为Android设备上移动控制代理设计的RL微调流程。DistRL采用异步架构，在异构工作设备和环境中部署RL微调代理，实现分散式数据收集。通过利用离策略RL技术，DistRL支持基于远程多样环境收集的数据进行集中训练，显著提升模型的可扩展性和鲁棒性。这些代表性方法展示了RL提升GUI代理的潜力，说明集中式和分布式RL框架如何使GUI自动化模型在实际应用中更具响应性、适应性和高效性。

10.14.11 5.7.6 Summary & Takeaways

10.14.12 5.7.6 总结与启示

In conclusion, the advanced techniques significantly enhance the capabilities of LLM-brained GUI agents, making them more versatile, efficient, and adaptive within multi-agent frameworks. Importantly, these techniques are not mutually exclusive-many can be integrated to create more powerful agents. For instance, incorporating self-reflection within a multi-agent framework allows agents to collaboratively improve task strategies and recover from errors. By leveraging these advancements, developers can design LLM-brained GUI agents that are not only adept at automating complex, multi-step tasks but also capable of continuously improving through self-evolution, adaptability to dynamic environments, and effective inter-agent collaboration. Future research is expected to yield even more sophisticated techniques, further extending the scope and robustness of GUI automation.
总之，先进技术显著提升了基于大型语言模型（LLM）的GUI代理的能力，使其在多代理框架中更加多功能、高效且适应性强。重要的是，这些技术并非相互排斥——许多技术可以集成以打造更强大的代理。例如，在多代理框架中引入自我反思机制，使代理能够协作改进任务策略并从错误中恢复。通过利用这些进展，开发者可以设计出不仅擅长自动化复杂多步骤任务，而且能够通过自我进化、适应动态环境及有效的代理间协作不断提升的LLM驱动GUI代理。未来的研究有望带来更为复杂的技术，进一步扩展GUI自动化的范围和鲁棒性。

10.15 5.8 From Foundations to Innovations: A Roadmap

10.16 5.8 从基础到创新：路线图

Building robust, adaptable, and effective LLM-powered GUI agents is a multifaceted process that requires careful integration of several core components. With a solid foundation in architecture, design, environment interaction, and memory, as outlined in Section 5 we now shift our focus to the critical elements required for deploying these agents in practical scenarios. This exploration begins with an in-depth review of state-of-the-art LLM-brained GUI agent frameworks in Section 6 highlighting their advancements and unique contributions to the field. Building on this, we delve into the methodologies for optimizing LLMs for GUI agents, starting with data collection and processing strategies in Section 7 and progressing to model optimization techniques in Section 8 To ensure robust development and validation, we then examine evaluation methodologies and benchmarks in Section 9, which are essential for assessing agent performance and reliability. Finally, we explore a diverse range of practical applications in Section 10 demonstrating the transformative impact of these agents across various domains.
构建稳健、适应性强且高效的LLM驱动GUI代理是一个多方面的过程，需要谨慎整合多个核心组件。基于第5节中架构、设计、环境交互和记忆的坚实基础，我们现在将重点转向在实际场景中部署这些代理所需的关键要素。本节从第6节对最先进的LLM驱动GUI代理框架进行深入回顾开始，重点介绍其进展和对该领域的独特贡献。在此基础上，我们探讨优化GUI代理用LLM的方法论，从第7节的数据收集与处理策略开始，进而到第8节的模型优化技术。为了确保开发和验证的稳健性，我们随后在第9节审视评估方法和基准，这对于评估代理性能和可靠性至关重要。最后，在第10节中，我们探讨了多样的实际应用，展示了这些代理在各领域的变革性影响。

Together, these sections provide a comprehensive roadmap for advancing LLM-brained GUI agents from foundational concepts to real-world implementation and innovation. This roadmap, spanning from foundational components to real-world deployment, encapsulates the essential pipeline required to bring an LLM-powered GUI agent concept from ideation to implementation.
这些章节共同提供了一条全面的路线图，将LLM驱动的GUI代理从基础概念推进到现实世界的实现与创新。这条路线图涵盖了从基础组件到实际部署的全过程，概括了将LLM驱动GUI代理从构想到实现所需的关键流程。

To provide a comprehensive view, we first introduce a taxonomy in Figure 19 which categorizes recent work on LLM-brained GUI agents across frameworks, data, models, evaluation, and applications. This taxonomy serves as a blueprint for navigating the extensive research and development efforts within each field, while acknowledging overlaps among categories where certain models, frameworks, or datasets contribute to multiple aspects of GUI agent functionality.
为了提供全面视角，我们首先在图19中介绍了一个分类法，将近期关于LLM驱动GUI代理的工作按框架、数据、模型、评估和应用进行分类。该分类法作为导航各领域广泛研发工作的蓝图，同时承认类别间的重叠，因为某些模型、框架或数据集对GUI代理功能的多个方面均有贡献。

11 6 LLM-BRAINED GUI AGENT FRAMEWORK

12 6 LLM驱动的GUI代理框架

The integration of LLMs has unlocked new possibilities for constructing GUI agents, enabling them to interpret user requests, analyze GUI components, and autonomously perform actions across diverse environments. By equipping these models with essential components and functionalities, as outlined in Section 5 researchers have created sophisticated frameworks tailored to various platforms and applications. These frameworks represent a rapidly evolving area of research, with each introducing innovative techniques and specialized capabilities that push the boundaries of what GUI agents can achieve.

LLM的整合为构建GUI代理开辟了新可能，使其能够理解用户请求、分析GUI组件，并在多样环境中自主执行操作。通过为这些模型配备第5节中概述的关键组件和功能，研究人员创建了针对不同平台和应用的复杂框架。这些框架代表了一个快速发展的研究领域，每个框架都引入了创新技术和专门能力，推动了GUI代理能力的边界。

We offer a detailed discussion of each framework, examining their foundational design principles, technical advancements, and the specific challenges they address in the realm of GUI automation. By delving into these aspects, we aim to provide deeper insights into how these agents are shaping the future of human-computer interaction and task automation, and the critical role they play in advancing this transformative field.
我们对每个框架进行详细讨论，审视其基础设计原则、技术进展及其在GUI自动化领域所解决的具体挑战。通过深入探讨这些方面，我们旨在提供更深刻的见解，展示这些代理如何塑造人机交互和任务自动化的未来，以及它们在推动这一变革性领域发展中的关键作用。

12.1 6.1 Web GUI Agents

12.2 6.1 网络GUI代理

Advancements in web GUI agents have led to significant strides in automating complex tasks within diverse and dynamic web environments. Recent frameworks have introduced innovative approaches that leverage multimodal inputs, predictive modeling, and task-specific optimizations to enhance performance, adaptability, and efficiency. In this subsection, we first summarize key web GUI agent frameworks in Tables 10 11 and 12 then delve into representative frameworks, highlighting their unique contributions and how they collectively push the boundaries of web-based GUI automation.
网络GUI代理的进步推动了在多样且动态的网络环境中自动化复杂任务的显著发展。近期框架引入了利用多模态输入、预测建模和任务特定优化的创新方法，以提升性能、适应性和效率。本小节首先在表10、11和12中总结关键的网络GUI代理框架，随后深入探讨代表性框架，突出其独特贡献及其如何共同推动基于网络的GUI自动化的边界。

One prominent trend is the integration of multimodal capabilities to improve interaction with dynamic web content. For instance, SeeAct [17] harnesses GPT-4V's multimodal capacities to ground actions on live websites effectively. By leveraging both visual data and HTML structure, SeeAct integrates grounding techniques using image annotations, HTML attributes, and textual choices, optimizing interactions with real-time web content. This approach allows SeeAct to achieve a task success rate of 51.1% on real-time web tasks, highlighting the importance of dynamic evaluation in developing robust web agents.
一个显著趋势是整合多模态能力以改善与动态网页内容的交互。例如，SeeAct [17] 利用GPT-4V的多模态能力，有效地将操作定位于实时网站。通过结合视觉数据和HTML结构，SeeAct采用图像注释、HTML属性和文本选择的定位技术，优化了与实时网页内容的交互。这种方法使SeeAct在实时网页任务中达到51.1%的任务成功率，凸显了动态评估在开发稳健网络代理中的重要性。

Building upon the advantages of multimodal inputs, We-bVoyager 269 advances autonomous web navigation by supporting end-to-end task completion across real-world web environments. Utilizing GPT-4V for both visual (screenshots) and textual (HTML elements) inputs, WebVoyager effectively interacts with dynamic web interfaces, including those with dynamically rendered content and intricate interactive elements. This multimodal capability allows WebVoyager to manage complex interfaces with a success rate notably surpassing traditional text-only methods, setting a new benchmark in web-based task automation.
基于多模态输入的优势，WebVoyager 269推动了自主网页导航，支持在真实网络环境中端到端完成任务。WebVoyager利用GPT-4V处理视觉（截图）和文本（HTML元素）输入，有效应对动态渲染内容和复杂交互元素的网页界面。这种多模态能力使WebVoyager能够管理复杂界面，其成功率显著超过传统的纯文本方法，树立了网络任务自动化的新标杆。

In addition to multimodal integration, some frameworks focus on parsing intricate web structures and generating executable code to navigate complex websites. WebAgent 267 employs a two-tiered model approach by combining HTML-T5 for parsing long, complex HTML documents with Flan-U-PaLM [498] for program synthesis. This modular design enables WebAgent to translate user instructions into executable Python code, autonomously handling complex, real-world websites through task-specific sub-instructions. WebAgent demonstrates a 50% improvement in success rates on real websites compared to traditional single-agent models, showcasing the advantages of integrating HTML-specific parsing with code generation for diverse and dynamic web environments.
除了多模态整合外，一些框架还专注于解析复杂的网页结构并生成可执行代码以导航复杂网站。WebAgent 267采用两层模型方法，结合HTML-T5解析冗长复杂的HTML文档与Flan-U-PaLM [498]进行程序合成。该模块化设计使WebAgent能够将用户指令转化为可执行的Python代码，通过任务特定的子指令自主处理复杂的真实网站。与传统单一代理模型相比，WebAgent在真实网站上的成功率提升了50%，展示了将HTML特定解析与代码生成结合应用于多样且动态网页环境的优势。

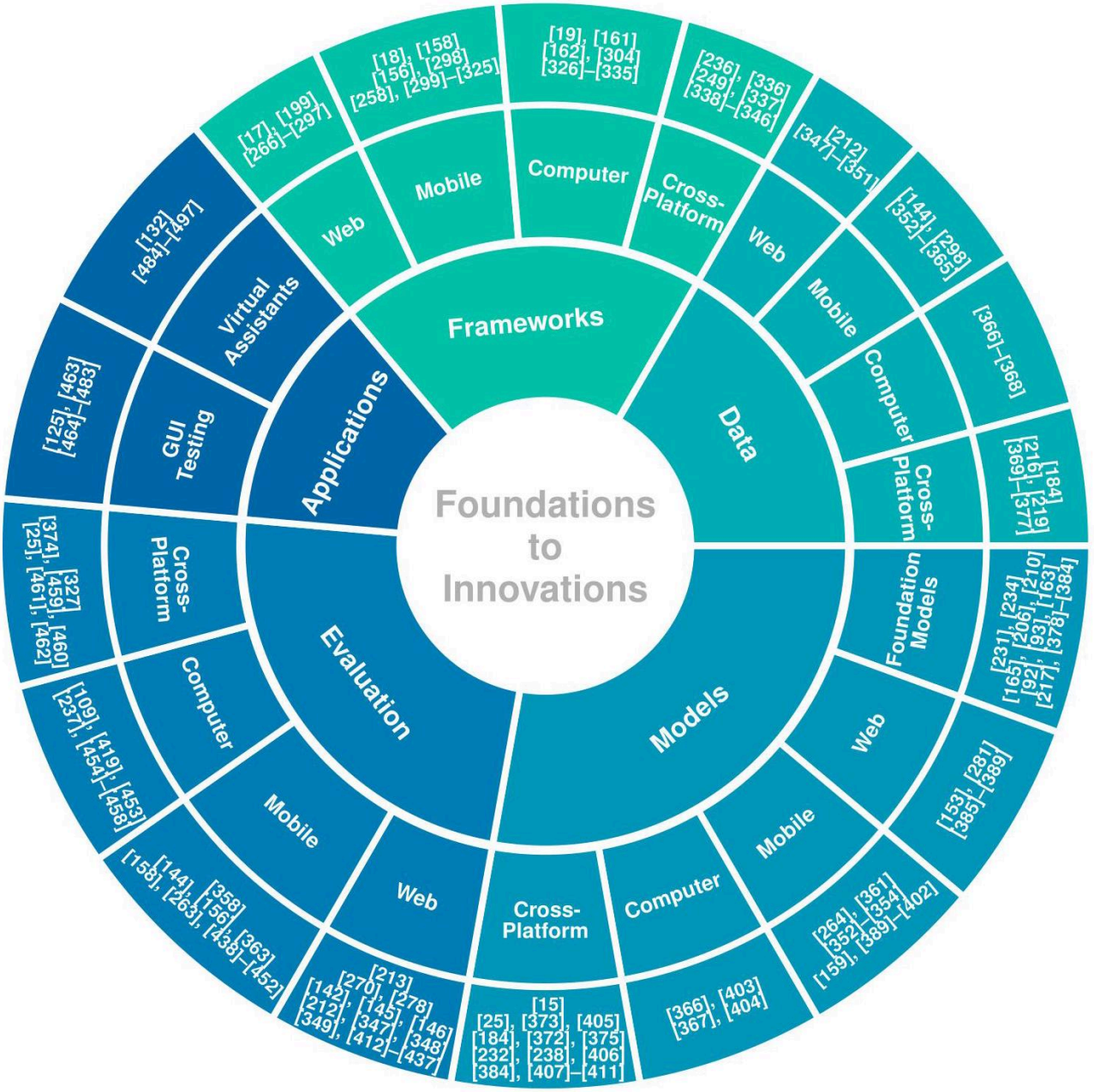

Fig. 19: A Taxonomy of frameworks, data, models, evaluations, and applications: from foundations to innovations in LLM-brained GUI agents.

图19：框架、数据、模型、评估与应用的分类法：从基础到LLM驱动GUI代理的创新。

To enhance decision-making in web navigation, several frameworks introduce state-space exploration and search algorithms. LASER 268 models web navigation as state-space exploration, allowing flexible backtracking and efficient decision-making without requiring extensive in-context examples. By associating actions with specific states and leveraging GPT-4's function-calling feature for state-based action selection, LASER minimizes errors and improves task success, particularly in e-commerce navigation tasks such as WebShop and Amazon. This state-based approach provides a scalable and efficient solution, advancing the efficiency of LLM agents in GUI navigation.

为提升网页导航中的决策能力，若干框架引入了状态空间探索与搜索算法。LASER 268将网页导航建模为状态空间探索，支持灵活回溯和高效决策，无需大量上下文示例。通过将动作与特定状态关联，并利用GPT-4的函数调用功能进行基于状态的动作选择，LASER最大限度减少错误并提升任务成功率，尤其在WebShop和Amazon等电商导航任务中表现突出。该基于状态的方法提供了可扩展且高效的解决方案，推动了LLM代理在GUI导航中的效率提升。

Similarly, Search-Agent 274 innovatively introduces a best-first search algorithm to enhance multi-step reasoning in interactive web environments. By exploring multiple action paths, this approach improves decision-making, achieving up to a 39% increase in success rates across benchmarks like WebArena [412]. Search-Agent's compatibility with existing multimodal LLMs demonstrates the effectiveness of search-based algorithms for complex, interactive web tasks.

类似地，Search-Agent 274创新性地引入了最佳优先搜索算法，以增强交互式网页环境中的多步推理。通过探索多条动作路径，该方法提升

了决策能力，在WebArena [412]等基准测试中成功率提升高达39%。Search-Agent与现有多模态LLM的兼容性证明了基于搜索算法在复杂交互网页任务中的有效性。

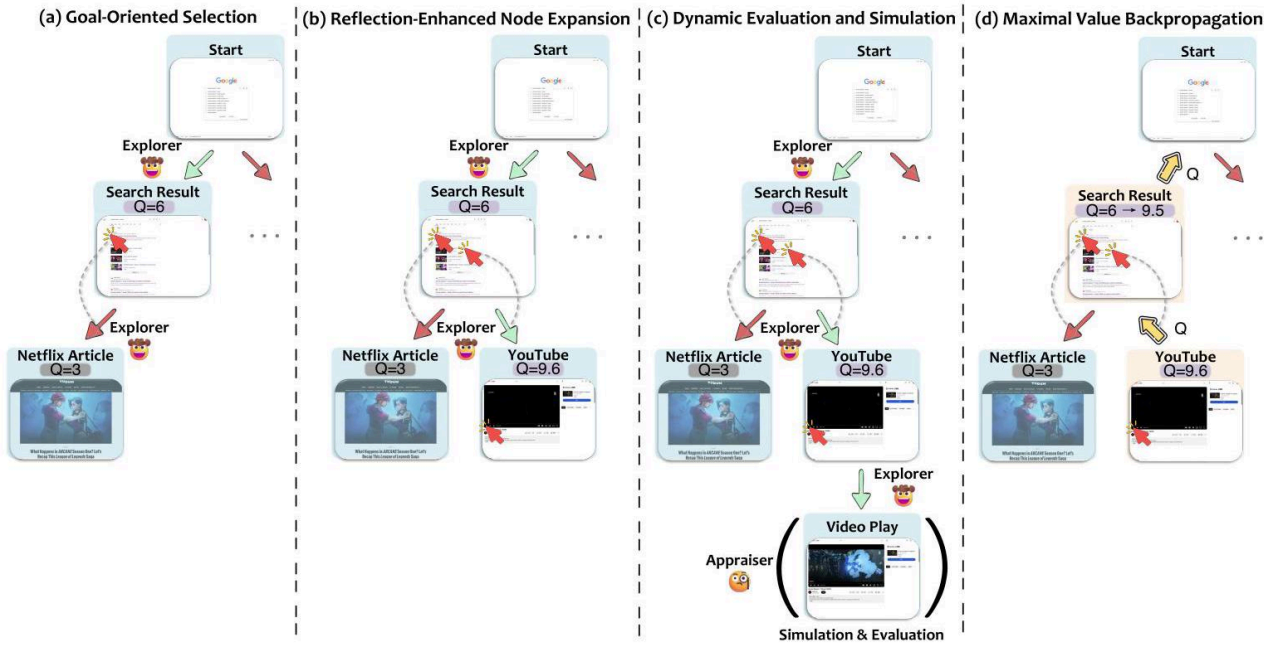

Fig. 20: An illustration of the local optimization stage in WebPilot [275] using MCTS. Figure adapted from the original paper.
图20：WebPilot [275]中使用蒙特卡洛树搜索（MCTS）进行局部优化阶段的示意图。图源自原论文。

Fig. 21: An example illustrating how WebDreamer [282] uses an LLM to simulate the outcome of each action. Figure adapted from the original paper.
图21：示例说明WebDreamer [282]如何利用LLM模拟每个动作的结果。图源自原论文。

Expanding on search-based strategies, WebPilot [275] employs a dual optimization strategy combining global and local Monte Carlo Tree Search (MCTS) [500] to improve adaptability in complex and dynamic environments. As illustrated in Figure 20 WebPilot decomposes overarching tasks into manageable sub-tasks, with each undergoing localized optimization. This approach allows WebPilot to continuously adjust its strategies in response to real-time observations, mimicking human-like decision-making and flexibility. Extensive testing on benchmarks like WebArena [412] and MiniWoB++ [146] demonstrates WebPilot's state-of-the-art performance, showcasing exceptional adaptability compared to existing methods.
在基于搜索的策略基础上，WebPilot [275]采用结合全局与局部蒙特卡洛树搜索（MCTS）[500]的双重优化策略，以提升在复杂动态环境中的适应性。如图20所示，WebPilot将整体任务分解为可管理的子任务，每个子任务进行局部优化。该方法使WebPilot能够根据实时观察持续调整策略，模拟人类般的决策与灵活性。在WebArena [412]和MiniWoB++ [146]等基准测试中的广泛实验表明，WebPilot表现出领先的性能，展现出较现有方法卓越的适应能力。

Furthering the concept of predictive modeling, the WMA [266] introduces a world model to simulate and predict the outcomes of UI interactions. By focusing on transition-based observations, WMA allows agents to simulate action results before committing, reducing unnecessary actions and increasing task efficiency. This predictive capability is particularly effective in long-horizon tasks that require high accuracy, with WMA demonstrating strong performance on benchmarks such as WebArena [412] and Mind2Web [212].
进一步推进预测建模理念，WMA [266]引入了世界模型以模拟和预测用户界面交互的结果。通过关注基于转移的观察，WMA使代理能够在

执行前模拟动作结果，减少不必要的操作并提升任务效率。该预测能力在需要高精度的长时任务中尤为有效，WMA在WebArena [412]和Mind2Web [212]等基准测试中表现优异。

Along similar lines, WebDreamer [282] introduces an innovative use of LLMs for model-based planning in web navigation, as depicted in Figure 21. WebDreamer simulates and evaluates potential actions and their multi-step outcomes using LLMs before execution [506], akin to a "dreamer" that envisions various scenarios. By preemptively assessing the potential value of different plans, WebDreamer selects and executes the plan with the highest expected value. This approach addresses critical challenges in web automation, such as safety concerns and the need for robust decision-making in complex and dynamic environments, demonstrating superiority over reactive agents in benchmarks like VisualWe-bArena [413] and Mind2Web-live [349].
同样，WebDreamer [282]创新性地利用LLM进行基于模型的网页导航规划，如图21所示。WebDreamer在执行前使用LLM模拟并评估潜在动作及其多步结果[506]，类似于“梦想者”预见各种情景。通过预先评估不同方案的潜在价值，WebDreamer选择并执行期望价值最高的方案。该方法解决了网页自动化中的关键挑战，如安全性问题及复杂动态环境下的稳健决策需求，在VisualWebArena [413]和Mind2Web-live [349]等基准测试中优于反应式代理。

TABLE 10: Overview of LLM-brained GUI agent frameworks on web platforms (Part I).
表10：基于LLM的GUI代理框架在网页平台上的概览（第一部分）。

Agent	Platform	Perception	Action	Model	Architecture	Highlight	Link
WMA 266	Web	Accessibility tree from DOM	UI operations, e.g., clock, type, and hover	Llama-3.1-8B-Instruct [91] for predicting observations and GPT-4 for policy modeling	Single-agent with simulation-based observation	Uses a world model to predict state changes before committing actions, improving task success rates and minimizing unnecessary interactions with the environment	https://github.com/kyle8581/WMA-Agents
WebAgent 267]	Web	HTML structure	UI interactions	HTML-T5 for planning and summarization and Flan-U-PaLM 498 for code generation	Two-stage architecture for planning and program synthesis	Leverages specialized LLMs to achieve HTML-based task planning and programmatic action execution	/
LASER 268]	Web	GUI structure of the web environment, with defined states	Defined per state. such as searching, selecting items. navigating pages. and finalizing a purchase	GPT-4	Single-agent	Uses a state-space exploration approach, allowing it to handle novel situations with flexible backtracking	https://github com/Mayer123/ LASER
WebVoyager 269]	Web	Screenshots with numerical labels interactive elements	Standard UI operations	GPT-4V	Single-agent	Integrates visual and textual cues within real-world, rendered web pages, enhancing its ability to navigate com plex web structures	https: //github.com/ MinorJerry/ WebVoyager
AutoWeb-GLM 270]	Web	Simplified HTML OCR for text recognition	UI operations such as clicking, typing, scrolling, and selecting, and advanced APIs like jumping to specific URLs	ChatGLM3-6B 499	Single-agent	Its HTML simplification method for efficient webpage comprehension and its bilingual benchmark	https://github.com/THUDM/AutoWebGLM
OpenAgents 271]	Web	DOM elements	Standard operations. browser-based actions controlled. API calls for tool execution, and structured data manipulation	GPT-4 and Claude [163]	Multi-agent architecture, with distinct agents (Data Agent, Plugins Agent, and Web Agent)	Democratizes access to lan guage agents by providing an open-source, multi-agent framework optimized for real-world tasks	https://github. com/xlang-al OpenAgents
SeeAct [17]	Web	Screenshot images and HTML structure	Standard UI operations	GPT-4V	Single-agent	Its use of GPT-4V's multimodal capabilities to integrate both visual anc HTML information, allowing for more accurate task per formance on dynamic web content	https: //github.com/ OSU-NLP-Group SeeAct
DUAL-VCR 272]	Web	HTML elements and screenshots	Standard UI operations	Flan-T5-base 498	Two-stage single-agent architecture	Dual-view contextualization	/

Agent-E 273]	Web	DOM structure and change observation	Standard UI operations	GPT-4 Turbo	Hierarchical multi-agent architecture, composed of a planner agent and a browser navigation agent	Hierarchical architecture and adaptive DOM perception	https: //github.com/ EmergenceAI/ Agent-E
Search-Agent 274]	Web	Screenshot and text descriptions	Standard UI operations	GPT-4	Single-agent with search	Novel inference-time search algorithm that enhances the agent's ability to per form multi-step planning and decision-making	https: //jykoh.com/ search-agents
R2D2 288]	Web	DOM	Standard UI operations	GPT-40	Single-agent	Dynamically constructs an internal web environment representation for more robust decision-making. The integration of a replay buffer and error analysis reduces navigation errors and improves task completion rates.	https: //github.com/ AmenRa/retriv

代理	平台	感知	动作	模型	架构	重点	链接
WMA 266	网页	来自DOM的无障碍树	用户界面操作，例如时钟、输入和悬停	用于预测观测的Llama-3.1-8B-Instruct [91]和用于策略建模的GPT-4	基于仿真观测的单代理	使用世界模型预测状态变化后再执行动作，提高任务成功率并减少不必要的环境交互	https://github.com/kyle8581/WMA-Agents
WebAgent 267]	网页	HTML结构	用户界面交互	用于规划和摘要的HTML-T5及用于代码生成的Flan-U-PaLM 498	用于规划和程序合成的两阶段架构	利用专用大型语言模型(LLMs)实现基于HTML的任务规划和程序化动作执行	/
LASER 268]	网页	具有定义状态的网页环境GUI结构	按状态定义，如搜索、选择项目、页面导航和完成购买	GPT-4	单代理	采用状态空间探索方法，支持灵活回溯以应对新情况	https://github com/Mayer123/LASER
WebVoyager 269]	网页	带数字标签的截图交互元素	标准用户界面操作	GPT-4V	单代理	整合真实渲染网页中的视觉和文本线索，增强导航复杂网页结构的能力	https: //github.com/ MinorJerry/ WebVoyager
AutoWeb-GLM 270]	网页	简化的HTML光学字符识别(OCR)用于文本识别	用户界面操作如点击、输入、滚动和选择，以及跳转特定URL等高级API	ChatGLM3-6B 499	单代理	其HTML简化方法用于高效网页理解及其双语基准	https://github.com/THUDM/AutoWebGLM
OpenAgents 271]	网页	DOM元素	标准操作，基于浏览器的动作控制，工具执行的API调用及结构化数据操作	GPT-4和Claude [163]	多代理架构，包含不同代理（数据代理、插件代理和网页代理）	通过提供开源多代理框架，优化现实任务，普及语言代理的使用	https://github. com/xlang-al OpenAgents
SeeAct [17]	网页	截图图像和HTML结构	标准用户界面操作	GPT-4V	单代理	利用GPT-4V的多模态能力整合视觉和HTML信息，实现对动态网页内容更准确的任务执行	https: //github.com/ OSU-NLP-Group SeeAct
DUAL-VCR 272]	网页	HTML元素和截图	标准用户界面操作	Flan-T5-base 498	两阶段单代理架构	双视角上下文化	/
Agent-E 273]	网页	DOM结构和变化观察	标准用户界面操作	GPT-4 Turbo	层级多代理架构，由规划代理和浏览器导航代理组成	层级架构和自适应DOM感知	https: //github.com/ EmergenceAI/ Agent-E
Search-Agent 274]	网页	截图和文本描述	标准用户界面操作	GPT-4	带搜索的单智能体	一种新颖的推理时搜索算法，增强智能体执行多步规划和决策的能力	https: //jykoh.com/ search-agents
R2D2 288]	网页	DOM	标准用户界面操作	GPT-40	单代理	动态构建内部网页环境表示，以实现更稳健的决策。回放缓冲区和错误分析的整合减少了导航错误，提高了任务完成率。	https: //github.com/ AmenRa/retriv

Beyond predictive modeling, integrating API interactions into web navigation offers enhanced flexibility and efficiency. The Hybrid Agent [199] combines web browsing and API interactions, dynamically switching between methods based on task requirements. By utilizing API calls for structured data interaction, the Hybrid Agent reduces the time and complexity involved in traditional web navigation, achieving higher accuracy and efficiency in task performance. This hybrid architecture underscores the benefits of integrating both structured API data and human-like browsing capabilities in AI agent systems.

超越预测建模，将API交互整合到网页导航中提供了更高的灵活性和效率。混合代理（Hybrid Agent）[199]结合了网页浏览和API交互，能够根据任务需求动态切换方法。通过利用API调用进行结构化数据交互，混合代理减少了传统网页导航中的时间和复杂性，实现了任务执行的更高准确性和效率。这种混合架构强调了在人工智能代理系统中整合结构化API数据与类人浏览能力的优势。

TABLE 11: Overview of LLM-brained GUI agent frameworks on web platforms (Part II).
表11：基于大型语言模型（LLM）的图形用户界面代理框架在网页平台上的概览（第二部分）。

Agent	Platform	Perception	Action	Model	Architecture	Highlight	Link
ScribeAgent 285]	Web	HTML-DOM	Standard UI operations	Single-agent architecture	Specialized fine-tuning approach using production-scale workflow data to outperform general-purpose LLMs like GPT-4 in web navigation tasks	https://github.com/colonylabs/ScribeAgent	
286]	Web	Screenshots	Standard UI Operations	Claude Sonnet 1631, Qwen2VL-7B 231], and LLaVa-1.6 [217]	A multi-agent architecture involving a task proposer to suggest tasks, an agent policy to perform tasks, and an autonomous evaluator to assess success and provide feedback using RL.	Autonomous skill discovery in real-world environments using task proposers and reward-based evaluation	https://yanqval.github.io/PAE/
WebPilot 275]	Web	Accessibility trees (actrees) and dynamic observations	Standard UI operations	GPT-4	Multi-agent architecture, with Global Optimization and Local Optimization	Dual optimization strategy (Global and Local) with Monte Carlo Tree Search (MCTS) [500], allowing dynamic adaptation to complex, real-world web environments	https://yaoz720.github.io/WebPilot/
Hybrid Agent 199]	Web	Accessibility trees and screenshots	Standard UI operations, API calls, and generating code	GPT-4	Multi-agent system, combining both API and browsing capabilities	Hybrid Agent seamlessly integrates web browsing and API calls	https://github.com/yuegis/API-Based-Agent
AgentOccam 276]	Web	HTML	Standard UI operations	GPT-4	Single-agent	Simple design that optimizes the observation and action spaces	/
NNetnav 277	Web	DOM	Standard UI operations	GPT-4	Single-agent	Trains web agents using synthetic demonstrations, eliminating the need for expensive human input	https: //github.com/MurtyShikhar/ Nnetnav
NaviQAte 2781	Web	Screenshots	Standard UI operations	GPT-4	Single-agent system	Frames web navigation as a question-and-answer task	7
OpenWeb-Agent 279	Web	HTML and screenshots	UI operations. Web APIs, and self-generated code	GPT-4 and AutoWebGLM 270	Modular single-agent	Modular design that allows developers to seamlessly integrate various models to automate web tasks	https://github.com/THUDM/OpenWebAgent/

Steward 280	Web	HTML and screenshots	Standard UI operations	GPT-4	Single-agent	Ability to automate web interactions using natural language instructions	/
WebDreamer 282]	Web	Screenshots combined with SoM. and HTML	Standard UI operations and navigation actions	GPT-40	Model-based single-agent architecture	Pioneers the use of LLMs as world models for planning in complex web environments	https: //github.com/ OSU-NLP-Group/ WebDreamer
Agent Q 281]	Web	DOM for textual input. screenshots for visual feedback	UI interactions. querying the user for help	LLaMA-3 70B 91 for policy learning and execution. GPT-V for visual feedback	Single-agent with MCTS and RL	Combination of MCTS-guided search and self-critique mechanisms enables improvement in reasoning and task execution	https: //github.com/ sentient-engineerir agent-q

代理	平台	感知	动作	模型	架构	亮点	链接
ScribeAgent 285]	网页	HTML-DOM	标准用户界面操作	单代理架构	使用生产规模工作流数据的专门微调方法，在网页导航任务中超越通用大型语言模型（LLMs）如GPT-4	https://github.com/colonylabs/ScribeAgent	
286]	网页	截图	标准用户界面操作	Claude Sonnet 1631, Qwen2VL-7B 231], 以及LLaVa-1.6 [217]	多代理架构，包含任务提议者用于建议任务，代理策略执行任务，以及自主评估者通过强化学习（RL）评估成功并提供反馈	在真实环境中使用任务提议者和基于奖励的评估实现自主技能发现	https://yanqval.github.io/PAE/
WebPilot 275]	网页	辅助功能树（actrees）和动态观察	标准用户界面操作	GPT-4	多代理架构，包含全局优化和局部优化	双重优化策略（全局与局部）结合蒙特卡洛树搜索（MCTS）[500]，实现对复杂真实网页环境的动态适应	https://yaoz720.github.io/WebPilot/
混合代理 199]	网页	辅助功能树和截图	标准用户界面操作、API调用及代码生成	GPT-4	多代理系统，结合API和浏览功能	混合代理无缝整合网页浏览与API调用	https://github.com/yuegis/ API-Based-Agent
AgentOccam 276]	网页	HTML	标准用户界面操作	GPT-4	单代理	简洁设计，优化观察和动作空间	/
NNetnav 277	网页	DOM	标准用户界面操作	GPT-4	单代理	使用合成示范训练网页代理，免除昂贵的人类输入	https: //github.com/MurtyShikhar/ Nnetnav
NaviQAte 2781	网页	截图	标准用户界面操作	GPT-4	单代理系统	将网页导航任务框架化为问答任务	7
OpenWeb-Agent 279	网页	HTML和截图	用户界面操作、网页API及自生成代码	GPT-4和AutoWebGLM 270	模块化单代理	模块化设计，允许开发者无缝集成多种模型以自动化网页任务	https://github.com/THUDM/OpenWebAgent/
Steward 280	网页	HTML和截图	标准用户界面操作	GPT-4	单代理	能够使用自然语言指令自动化网页交互	/
WebDreamer 282]	网页	结合截图、SoM和HTML	标准用户界面操作和导航动作	GPT-40	基于模型的单代理架构	开创性地使用大型语言模型（LLMs）作为复杂网页环境中的世界模型进行规划	https: //github.com/ OSU-NLP-Group/ WebDreamer

Agent Q 281]	网页	用于文本输入的DOM，截图用于视觉反馈	用户界面交互，向用户请求帮助	使用LLaMA-3 70B 91进行策略学习与执行，GPT-V用于视觉反馈	结合蒙特卡洛树搜索（MCTS）与强化学习（RL）的单智能体	蒙特卡洛树搜索（MCTS）引导的搜索与自我批评机制的结合促进了推理和任务执行的提升	https: //github.com/ sentient-engineerir agent-q

Addressing the challenges of complex web structures and cross-domain interactions, AutoWebGLM [270] offers an efficient solution by simplifying HTML to focus on key webpage components, thereby improving task accuracy. Using reinforcement learning and rejection sampling for fine-tuning, AutoWebGLM excels in complex navigation tasks on both English and Chinese sites. Its bilingual dataset and structured action-perception modules make it practical for cross-domain web interactions, emphasizing the importance of efficient handling in diverse web tasks.
针对复杂网页结构和跨域交互的挑战，AutoWebGLM [270] 通过简化HTML以聚焦网页关键组件，提供了一种高效解决方案，从而提升任务准确性。AutoWebGLM采用强化学习和拒绝采样进行微调，在中英文网站的复杂导航任务中表现出色。其双语数据集和结构化的动作-感知模块使其在跨域网页交互中具有实用性，强调了高效处理多样化网页任务的重要性。

ECLAIR [291] represents a pioneering application that replaces traditional RPA with a foundation model-powered GUI agent for enterprise automation. Unlike conventional RPA, which relies on manually programmed rules and rigid scripts, ECLAIR dynamically learns workflows from video demonstrations and textual SOPs (Standard Operating Procedures), significantly reducing setup time and improving adaptability. It operates on enterprise web applications, leveraging GPT- 4V and CogAgent [505] to perceive GUI elements, plan actions, and execute workflows, and validate automatically. By eliminating the high maintenance costs and execution brittleness of RPA, ECLAIR introduces a more flexible and scalable approach to GUI automation. We show a comparison of such agent-based vs. RPA automation in Figure 22. This work establishes an important foundation for LLM-powered GUI automation, demonstrating how multimodal foundation models can bridge the gap between process mining, RPA, and fully autonomous enterprise workflows.
ECLAIR [291] 是一项开创性应用，利用基础模型驱动的GUI代理替代传统的RPA，实现企业自动化。不同于依赖手工编程规则和僵硬脚本的传统RPA，ECLAIR通过视频演示和文本标准操作流程（SOP）动态学习工作流程，显著缩短了部署时间并提升了适应性。它运行于企业网页应用，借助GPT-4V和CogAgent [505]感知GUI元素、规划动作、执行工作流程并自动验证。通过消除RPA的高维护成本和执行脆弱性，ECLAIR引入了一种更灵活且可扩展的GUI自动化方法。图22展示了基于代理与RPA自动化的对比。该工作为基于大语言模型（LLM）的GUI自动化奠定了重要基础，展示了多模态基础模型如何弥合流程挖掘、RPA与全自动企业工作流之间的差距。

In summary, recent frameworks for web GUI agents have made substantial progress by integrating multimodal inputs, predictive models, and advanced task-specific optimizations. These innovations enable robust solutions for real-world tasks, enhancing the capabilities of web-based GUI agents and marking significant steps forward in developing intelligent, adaptive web automation.
总之，近期针对网页GUI代理的框架通过整合多模态输入、预测模型及先进的任务特定优化取得了显著进展。这些创新为现实任务提供了稳健的解决方案，增强了基于网页的GUI代理的能力，标志着智能、自适应网页自动化开发的重要进步。

12.3 6.2 Mobile GUI Agents

12.4 6.2 移动端GUI代理

TABLE 12: Overview of LLM-brained GUI agent frameworks on web platforms (Part III).
表12：基于大语言模型的网页平台GUI代理框架概览（第三部分）。

Agent	Platform	Perception	Action	Model	Architecture	Highlight	Link
Auto-Intent 284]	Web	HTML structure	Standard UI Operations	GPT-3.5, GPT-4, Llama-3 [91] for action inference; Mistral-7B [501] and Flan-T5XL 498] for intent prediction	Single-agent with self-exploration	Introduces a unique self-exploration strategy to generate semantically diverse intent hints	/
AdaptAgent 283]	Web	GUI screen-shots with HTML/DOM structures	Standard UI Operations and Playwright scripts	GPT-40 and Co-gAgent [15]	Single-agent	Adapts to unseen tasks with just 1-2 multimodal human demonstrations	/
WEPO 287]	Web	HTML and DOM	Standard UI Operations	Llama3-8B 91 Mistral-7B [501]. and Gemma-2B 502]	Single-agent architecture.	Incorporates a distance-based sampling mechanism tailored to the DOM tree structure, enhancing preference learning by distinguishing between salient and non-salient web elements with DPO 5031	/
AgentSymbioticWeb 289]		Accessible tree structure of web elements	Standard UI operations	Large LLMs: GPT-40, Claude- 3.5. Small LLMs: LLaMA-3 [91]. DeepSeek-R1 504]	Multi-agent iterative architecture	Introduces an iterative, symbiotic learning process between large and small LLMs for web automation. Enhances both data synthesis and task performance through speculative data synthesis, multi-task learning, and privacy-preserving hybrid modes.	/
LiteWebAgent 292]	Web	DOM, Screen-shots	Standard UI operations, Playwright script	Any LLM and MLLM	Single-agent	First open-source. production-ready web agent integrating tree search for multi-step task execution.	https://github.com/PathOnAI/ LiteWebAgent
ECLAIR 291]	Web	Screenshots	Standard UI operations	GPT-4V,GPT-40, CogAgent 505	Single-agent architecture	Eliminates the high setup costs, brittle execution, and burdensome maintenance associated with traditional RPA by learning from video and text documentation.	https: //github.com/ HazyResearch/ eclair-agents

Dammu et al., 293]	Web	DOM elements, Webpage accessibility attributes	Standard UI operations	Not specified	Single-agent architecture	User-aligned task execution where the agent adapts to individual user preferences in an ethical manner.	/
Plan-and-Act 294]	Web	HTML	Standard UI operations	LLaMA-3.3-70B-Instruct 91]	Two-stage modular architecture: PLANNER + EXECUTOR	Decouples planning from execution in LLM-based GUI agents and introduces a scalable synthetic data generation pipeline to fine-tune each component	/
SkillWeaver 295]	Web	GUI screen-shots and Accessibility Tree	Standard UI operations and high-level skill APIs	GPT-40	Single-agent	Introduces a self-improvement framework for web agents that autonomously discover. synthesize. and refine reusable skill APIs through exploration	https: //github.com/ OSU-NLP-Group/ SkillWeaver
ASI 2961	Web	Webpage Accessibility Tree	Standard GUI actions	Claude-3.5-Sonnet	Single-agent	Introduces programmatic skills that are verified through execution to ensure quality and are used as callable actions to improve efficiency	https://github.com/zorazrw/agent-skill-inductio
Rollback Agent 297]	Web	Accessibility trees	Standard GUI actions	Multi-agent ar chitecture	Multi-module. ReAct-inspired agent architecture	Introduces a modular rollback mechanism that enables multi-step rollback to avoid dead-end states	/

代理	平台	感知	动作	模型	架构	亮点	链接
自动意图 284]	网页	HTML结构	标准用户界面操作	用于动作推断的GPT-3.5、GPT-4、Llama-3 [91]；用于意图预测的Mistral-7B [501]和Flan-T5XL [498]	单代理自我探索	引入独特的自我探索策略以生成语义多样的意图提示	/
AdaptAgent 283]	网页	带有HTML/DOM结构的GUI截图	标准用户界面操作和Playwright脚本	GPT-40和CogAgent [15]	单代理	仅通过1-2次多模态人类示范适应未见任务	/
WEPO 287]	网页	HTML和DOM	标准用户界面操作	Llama3-8B [91]、Mistral-7B [501]和Gemma-2B [502]	单代理架构	结合针对DOM树结构的基于距离的采样机制，通过DPO [503]区分显著与非显著网页元素，提升偏好学习效果	/
AgentSymbioticWeb 289]		网页元素的可访问树结构	标准用户界面操作	大型大语言模型：GPT-40、Claude-3.5；小型大语言模型：LLaMA-3 [91]、DeepSeek-R1 [504]	多代理迭代架构	引入大型与小型大语言模型间的迭代共生学习过程，用于网页自动化。通过推测性数据合成、多任务学习及隐私保护混合模式，提升数据合成与任务性能。	/
LiteWebAgent 292]	网页	DOM，截图	标准用户界面操作，Playwright脚本	任意大语言模型和多模态大语言模型	单代理	首个开源、生产就绪的网页代理，集成树搜索以执行多步骤任务	https://github.com/PathOnAI/LiteWebAgent
ECLAIR 291]	网页	截图	标准用户界面操作	GPT-4V、GPT-40、CogAgent [505]	单代理架构	通过学习视频和文本文档，消除传统RPA的高设置成本、脆弱执行和繁重维护	https://github.com/HazyResearch/eclair-agents
Dammu等人，293]	网页	DOM元素，网页可访问性属性	标准用户界面操作	未指定	单代理架构	用户对齐的任务执行，代理以伦理方式适应个体用户偏好	/
Plan-and-Act 294]	网页	HTML	标准用户界面操作	LLaMA-3.3-70B-Instruct [91]	两阶段模块化架构：规划器 + 执行器	在基于大语言模型的GUI代理中解耦规划与执行，且引入可扩展的合成数据生成流水线以微调各组件	/
SkillWeaver 295]	网页	GUI截图和可访问性树	标准用户界面操作和高级技能API	GPT-40	单代理	引入自我提升框架，网页代理通过探索自主发现、合成和优化可复用技能API	https://github.com/OSU-NLP-Group/SkillWeaver

ASI 296]	网页	网页可访问性树	标准GUI操作	Claude-3.5-Sonnet	单代理	引入通过执行验证的程序化技能，确保质量，并作为可调用动作提升效率	https://github.com/zorazrw/ agent-skill-inductio
回滚代理 297]	网页	辅助功能树	标准GUI操作	多智能体架构	多模块。受ReAct启发的代理架构	引入了一种模块化回滚机制，支持多步回滚以避免死胡同状态	/

The evolution of mobile GUI agents has been marked by significant advancements, leveraging multimodal models, complex architectures, and adaptive planning to address the unique challenges of mobile environments. These agents have progressed from basic interaction capabilities to sophisticated systems capable of dynamic, context-aware operations across diverse mobile applications. We first provide an overview of mobile GUI agent frameworks in Tables 13 14 and 15
移动GUI代理的发展经历了显著的进步，利用多模态模型、复杂架构和自适应规划来应对移动环境的独特挑战。这些代理从基本的交互能力发展为能够在多样化移动应用中进行动态、上下文感知操作的复杂系统。我们首先在表13、14和15中概述了移动GUI代理框架。

Wang et al., 323 pioneer the use of LLMs to enable conversational interaction with mobile UIs, establishing one of the earliest foundations for mobile GUI agents. Their approach involves directly prompting foundation models such as PaLM using structured representations of Android view hierarchies, which are transformed into HTML-like text to better align with the LLM's training distribution. The authors define and evaluate four core tasks, including Screen Summarization, Screen QA, Screen Question Generation, and Instruction-to-UI Mapping-demonstrating that strong performance can be achieved with as few as two prompt examples per task. Emphasizing practicality and accessibility, the work enables rapid prototyping without model fine-tuning, and stands out as a seminal effort in prompt-based evaluation of LLM-powered GUI agents for mobile applications.
Wang等人323开创性地使用大型语言模型（LLMs）实现与移动用户界面的对话交互，奠定了移动GUI代理的早期基础。他们的方法通过结构化表示Android视图层次，将其转换为类似HTML的文本，以更好地匹配LLM的训练分布，直接提示基础模型如PaLM。作者定义并评估了四个核心任务，包括屏幕摘要、屏幕问答、屏幕问题生成和指令到UI映射，展示了每个任务仅用两个提示示例即可实现良好性能。该工作强调实用性和易用性，实现了无需模型微调的快速原型开发，是基于提示的LLM驱动移动GUI代理评估的开创性尝试。

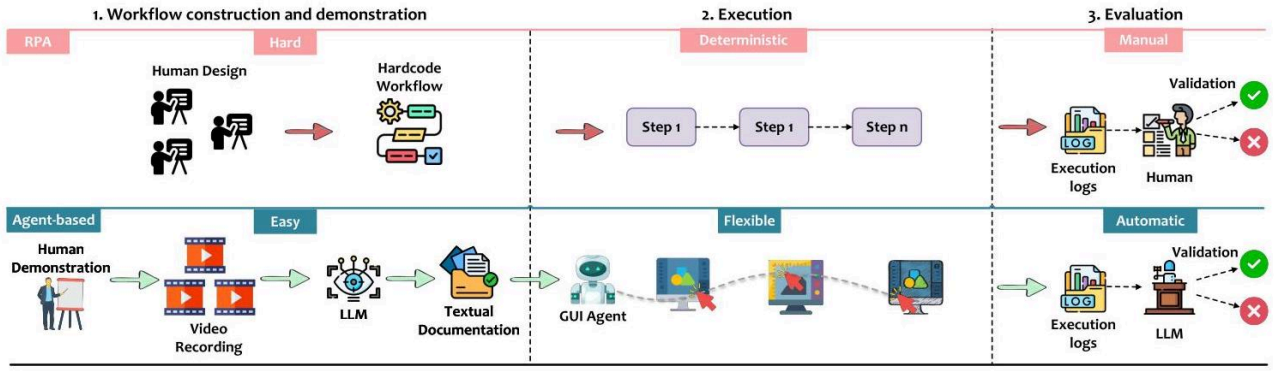

Fig. 22: Comparison of RPA and agent based automation. Figure adapted from [291].
图22：RPA与基于代理的自动化比较。图形改编自[291]。

Early efforts focused on enabling human-like GUI interactions without requiring backend system access. One such pioneering framework is AppAgent [18], which utilizes GPT- 4V's multimodal capabilities to comprehend and respond to both visual and textual information. By performing actions like tapping and swiping using real-time screenshots and structured XML data, AppAgent can interact directly with the GUI across a variety of applications, from social media to complex image editing. Its unique approach of learning through autonomous exploration and observing human demonstrations allows for rapid adaptability to new apps, highlighting the effectiveness of multimodal capabilities in mobile agents.
早期工作集中于实现类人GUI交互，无需访问后端系统。其中一个开创性框架是AppAgent [18]，利用GPT-4V的多模态能力理解并响应视觉和文本信息。通过使用实时截图和结构化XML数据执行点击和滑动等操作，AppAgent能够直接与各种应用的GUI交互，从社交媒体到复杂的图像编辑。其通过自主探索和观察人类示范进行学习的独特方法，实现了对新应用的快速适应，凸显了多模态能力在移动代理中的有效性。

Building upon this foundation, AppAgent-V2 [304] advances the framework by enhancing visual recognition and incorporating structured data parsing. This enables precise, context-aware interactions and the ability to perform complex, multi-step operations across different applications. AppAgent-V2 also introduces safety checks to handle sensitive data and supports cross-app tasks by tracking and adapting to real-time interactions. This progression underscores the importance of advanced visual recognition and structured data processing in improving task precision and safety in real-time mobile environments.
在此基础上，AppAgent-V2 [304]通过增强视觉识别和引入结构化数据解析推进了框架发展。这使得其能够实现精确的上下文感知交互，并执行跨应用的复杂多步骤操作。AppAgent-V2还引入了安全检查以处理敏感数据，并通过跟踪和适应实时交互支持跨应用任务。这一进展强调了先进视觉识别和结构化数据处理在提升实时移动环境任务精度和安全性方面的重要性。

Parallel to these developments, vision-centric approaches emerged to further enhance mobile task automation without relying on app-specific data. For instance, Mobile-Agent [158] leverages OCR, CLIP [509], and Grounding DINO [183] for visual perception. By using screenshots and visual tools, Mobile-Agent performs operations ranging from app navigation to complex multitasking, following instructions iteratively and adjusting for errors through a self-reflective mechanism. This vision-based method positions Mobile-Agent as a versatile and adaptable assistant for mobile tasks.
与此并行，视觉中心方法出现以进一步提升移动任务自动化，且不依赖于特定应用数据。例如，Mobile-Agent [158]利用OCR、CLIP [509]和Grounding DINO [183]进行视觉感知。通过使用截图和视觉工具，Mobile-Agent执行从应用导航到复杂多任务的操作，迭代遵循指令并通过自我反思机制调整错误。这种基于视觉的方法使Mobile-Agent成为多功能且适应性强的移动任务助手。

To address challenges in long-sequence navigation and complex, multi-app scenarios, Mobile-Agent-v2 [306] introduces a multi-agent architecture that separates planning, decision-making, and reflection. By distributing responsibilities among three agents, this framework optimizes task progress tracking, retains memory of task-relevant information, and performs corrective actions when errors occur. Integrated with advanced visual perception tools like Grounding DINO [183] and Qwen-VL-Int4 [210], Mobile-Agent-v2 showcases significant improvements in task completion rates on both Android and Harmony OS, highlighting the potential of multi-agent systems for handling complex mobile tasks.
为解决长序列导航和复杂多应用场景的挑战，Mobile-Agent-v2 [306]引入了多代理架构，分离规划、决策和反思。通过在三个代理间分配职责，该框架优化任务进度跟踪，保留任务相关信息记忆，并在发生错误时执行纠正操作。结合Grounding DINO [183]和Qwen-VL-Int4 [210]等先进视觉感知工具，Mobile-Agent-v2在Android和Harmony OS上的任务完成率显著提升，展示了多代理系统处理复杂移动任务的潜力。

In addition to vision-centric methods, some frameworks focus on translating GUI states into language to enable LLM-based action planning. VisionTasker 299] combines vision-based UI interpretation with sequential LLM task planning by processing mobile UI screenshots into structured natural language. Supported by YOLO-v8 [207] and PaddleOCP [28] for widget detection, VisionTasker allows the agent to automate complex tasks across unfamiliar apps, demonstrating higher accuracy than human operators on certain tasks. This two-stage design illustrates a versatile and adaptable framework, setting a strong precedent in mobile automation.
除了视觉中心方法，一些框架专注于将GUI状态转化为语言，以实现基于LLM的动作规划。VisionTasker [299]结合基于视觉的UI解析与序列化LLM任务规划，通过将移动UI截图处理为结构化自然语言。借助YOLO-v8 [207]和PaddleOCP [28]进行控件检测，VisionTasker使代理能够自动化执行陌生应用中的复杂任务，在某些任务上准确率超过人类操作员。这种两阶段设计展示了一个多功能且适应性强的框架，为移动自动化树立了坚实的先例。

Similarly, DroidBot-GPT [300] showcases an innovative approach by converting GUI states into natural language prompts, enabling LLMs to autonomously decide on action sequences. By interpreting the GUI structure and translating it into language that GPT models can understand, DroidBot-GPT generalizes across various apps without requiring app-specific modifications. This adaptability underscores the transformative role of LLMs in handling complex, multi-step tasks with minimal custom data.
类似地，DroidBot-GPT [300]展示了一种创新方法，通过将GUI状态转换为自然语言提示，使LLM能够自主决定动作序列。通过解析GUI结构并将其转化为GPT模型可理解的语言，DroidBot-GPT实现了跨多种应用的泛化，无需针对特定应用进行修改。这种适应性凸显了LLM在以最少定制数据处理复杂多步骤任务中的变革性作用。

To enhance action prediction and context awareness, advanced frameworks integrate perception and action systems within a multimodal LLM. CoCo-Agent [301] exemplifies this by processing GUI elements like icons and layouts through its Comprehensive Event Perception and Comprehensive Action Planning modules. By decomposing actions into manageable steps and leveraging high-quality data from benchmarks like Android in the Wild (AITW) [358] and META-GUI [357], CoCo-Agent demonstrates its ability to automate mobile tasks reliably across varied smartphone applications.
为提升动作预测和上下文感知，先进框架将感知与动作系统集成于多模态LLM中。CoCo-Agent [301]通过其综合事件感知和综合动作规划模块处理图标和布局等GUI元素。通过将动作分解为可管理步骤，并利用Android in the Wild (AITW) [358]和META-GUI [357]等基准的高质量数据，CoCo-Agent展示了其在多样智能手机应用中可靠自动化移动任务的能力。

Further advancing this integration, CoAT [298] introduces a chain-of-action-thought process to enhance action prediction and context awareness. Utilizing sophisticated models such as GPT-4V and set-of-mark tagging, CoAT addresses the limitations of traditional coordinate-based action recognition. By leveraging the Android-In-The-Zoo (AITZ) dataset it builds, CoAT provides deep context awareness and improves both action prediction accuracy and task completion rates, highlighting its potential for accessibility and user convenience on Android platforms.
进一步推进这一整合，CoAT [298] 引入了链式行动思维（chain-of-action-thought）流程，以增强动作预测和上下文感知。CoAT 利用 GPT-4V 和标记集合（set-of-mark tagging）等复杂模型，解决了传统基于坐标的动作识别的局限性。通过利用其构建的 Android-In-The-

Zoo（AITZ）数据集，CoAT 提供了深度上下文感知，提升了动作预测的准确性和任务完成率，凸显了其在 Android 平台上提升无障碍性和用户便利性的潜力。

---

28. https://github.com/PaddlePaddle/PaddleOCR
29. https://github.com/PaddlePaddle/PaddleOCR

JOURNAL OF LATEX CLASS FILES, DECEMBER 2024

TABLE 13: Overview of LLM-brained GUI agent frameworks on mobile platforms (Part I).
表13：移动平台上基于大型语言模型（LLM）驱动的图形用户界面（GUI）代理框架概览（第一部分）。

Agent	Platform	Perception	Action	Model	Architecture	Highlight	Link
Wang et al., 323]	Android Mobile	Android view hierarchy structure	(1) Screen Question Generation, (2) Screen Summarization, (3) Screen Question Answering, and (4) Mapping Instruction to UI Action	PaLM [507	Single-agent	The first paper to study Screen Question Generation and Screen QA using LLMs	https: //github.com/ google-research google-research tree/master/ llm4mobile
VisionTasker 299]	Android mobile devices	UI screenshots with widget detection and text extraction	UI operations such as tapping, swiping, and entering text	ERNIE Bot 508	Single-agent with vision-based UI understanding and sequential task planning	Vision-based UI understanding approach, which allows it to interpret UI semantics directly from screenshots with out view hierarchy dependencies	https: //github.com/ AkimotoAyako/ VisionTasker
DroidBot-GPT [300]	Android mobile devices	Translates the GUI state information of Android applications into natural language prompts	UI operations, including actions like click, scroll, check, and edit	GPT	Single-agent	Automates Android applica tions without modifications to either the app or the model	https: //github.com/ MobileLLM/ DroidBot-GPT
CoCo-Agent 301]	Android mobile devices	GUI screen-shots, OCR layouts, and historical actions	GUI actions, such as clicking, scrolling, and typing	CLIP 509] for vision encoding and LLaMA-2-chat-7B for language processing	Single-agent	Its dual approach of Comprehensive Environment Perception and Conditional Action Prediction	https://github. com/xbmxb/ CoCo-Agent
	Android mobile devices	GUI screen-shots	GUI operations	BLIP-2 vision encoder 206 with a FLAN-Alpaca [78]	Single-agent with chain-of-action	Its direct interaction with GUI elements. Its chain-of-action mechanism enables it to leverage both past and planned actions	https://github. com/co0elf/ Auto-GUI
MobileGPT 310]	Android mobile devices	Simplified HTML representation	Standard UI operations and navigation actions	GPT-4-turbo for screen understanding and reasoning. GPT-3.5-turbo for slot-filling sub-task parameters	Single-agent architecture augmented by a hierarchical memory structure	Introduces a human-like app memory that allows for task decomposition into modular sub-tasks	https: //mobile-gpt. github.io
MM-Navigator 303]	Mobile iOS and Android	Smartphone screenshots with associated set-of-mark tags	Clickable UI operations	GPT-4V	Single-agent	Using set-of-mark prompting with GPT-4V for precise GUI navigation on smartphones	https://github.com/zzxslp/ MM-Navigator

AppAgent	Android mobile devices	Real-time screenshots and XML files detailing the interactive elements	User-like actions. like Tap, Long press, Swipe, Text input, Back and Exit	GPT-4V	Single-agent	Its ability to perform tasks on any smartphone app us ing a human-like interaction method	https:// appagent-official github.io/
AppAgent-V2 304]	Android mobile devices	GUI screen-shots witl annotated elements, OCR for detecting text and icons. Structured XML metadata	Standard UI Operations: Tap, text input, long press, swipe, back, and stop	GPT-4	Multi-phase architecture witl Exploration Phase and Deployment Phase	Enhances adaptability and precision in mobile environ ments by combining structured data parsing with visual features	/
FedMobileAge 314]	entAndroid mobile devices	GUI Screen-shots	Standard UI operations	Qwen2-VL-Instruct-7B 231]	Multi-agent federated learning	Introduces preserving learning for mobile automation, enabling large-scale without centralized human annotation. privacy-federated training	

代理	平台	感知	动作	模型	架构	高亮	链接
王等人，323]	安卓手机	安卓视图层级结构	(1) 屏幕问题生成，(2) 屏幕摘要，(3) 屏幕问答，(4) 指令映射到UI操作	PaLM [507	单代理	首篇使用大型语言模型（LLMs）研究屏幕问题生成和屏幕问答的论文	https: //github.com/ google-research google-research tree/master/ llm4mobile
VisionTasker 299]	安卓移动设备	带有控件检测和文本提取的UI截图	UI操作，如点击、滑动和输入文本	ERNIE Bot 508	基于视觉的UI理解和顺序任务规划的单代理	基于视觉的UI理解方法，能够直接从截图中解释UI语义，无需视图层级依赖	https: //github.com/ AkimotoAyako/ VisionTasker
DroidBot-GPT [300]	安卓移动设备	将安卓应用的GUI状态信息转换为自然语言提示	UI操作，包括点击、滚动、勾选和编辑等动作	GPT	单代理	无需修改应用或模型即可自动化安卓应用	https: //github.com/ MobileLLM/ DroidBot-GPT
CoCo-Agent 301]	安卓移动设备	GUI截图、OCR布局和历史操作	GUI操作，如点击、滚动和输入	使用CLIP 509]进行视觉编码，LLaMA-2-chat-7B进行语言处理	单代理	其综合环境感知与条件动作预测的双重方法	https://github. com/xbmxb/ CoCo-Agent
	安卓移动设备	GUI截图	GUI操作	BLIP-2视觉编码器 206，配合FLAN-Alpaca [78]	带有动作链的单代理	其与GUI元素的直接交互。动作链机制使其能够利用过去和计划中的动作	https://github. com/co0elf/ Auto-GUI
MobileGPT 310]	安卓移动设备	简化的HTML表示	标准UI操作和导航动作	使用GPT-4-turbo进行屏幕理解和推理，GPT-3.5-turbo用于槽位填充子任务参数	由分层记忆结构增强的单代理架构	引入类人应用记忆，支持任务分解为模块化子任务	https: //mobile-gpt. github.io
MM-Navigator 303]	移动iOS和安卓	带有关联标记集的智能手机截图	可点击的UI操作	GPT-4V	单代理	使用带标记集提示的GPT-4V实现智能手机上的精确GUI导航	https://github.com/zzxslp/ MM-Navigator
AppAgent	安卓移动设备	实时截图和详细交互元素的XML文件	类用户操作，如点击、长按、滑动、文本输入、返回和退出	GPT-4V	单代理	其使用类人交互方式在任何智能手机应用上执行任务的能力	https:// appagent-official github.io/
AppAgent-V2 304]	安卓移动设备	带注释元素的GUI截图，使用OCR检测文本和图标。结构化XML元数据	标准UI操作：点击、文本输入、长按、滑动、返回和停止	GPT-4	包含探索阶段和部署阶段的多阶段架构	通过结合结构化数据解析与视觉特征，提升移动环境中的适应性和精确性	/
FedMobileAge 314]	Android移动设备	图形用户界面截图	标准用户界面操作	Qwen2-VL-Instruct-7B 231]	多智能体联邦学习	引入面向移动自动化的保留学习，实现无需集中人工标注的大规模隐私联邦训练	

Addressing the need for efficient handling of multi-step tasks with lower computational costs, AutoDroid [156] combines LLM-based comprehension with app-specific knowledge. Using an HTML-style GUI representation and a memory-based approach, AutoDroid reduces dependency on extensive LLM queries. Its hybrid architecture of cloud and on-device models enhances responsiveness and accessibility, making AutoDroid a practical solution for diverse mobile tasks. AutoDroid-V2 [305] enhances its predecessor AutoDroid, by utilizing on-device language models to generate and execute multi-step scripts for user task automation. By transforming dynamic and complex GUI elements of mobile apps into structured app documents, it achieves efficient and accurate automation without depending on cloud-based resources. The script-based approach reduces computational overhead by minimizing query frequency, thereby improving task efficiency and addressing the limitations of stepwise agents. This advancement enables privacy-preserving and scalable task automation on mobile platforms.

为满足多步骤任务高效处理且降低计算成本的需求，AutoDroid [156] 结合了基于大语言模型（LLM）的理解能力与应用特定知识。通过采

用类HTML的图形用户界面（GUI）表示和基于记忆的方法，AutoDroid 减少了对大量LLM查询的依赖。其云端与设备端模型的混合架构提升了响应速度和可访问性，使AutoDroid成为多样化移动任务的实用解决方案。AutoDroid-V2 [305] 在前代AutoDroid基础上进行了增强，利用设备端语言模型生成并执行多步骤脚本，实现用户任务自动化。通过将移动应用中动态复杂的GUI元素转化为结构化的应用文档，它实现了高效且准确的自动化，且无需依赖云端资源。基于脚本的方法通过减少查询频率降低了计算开销，从而提升了任务效率，解决了逐步代理的局限性。这一进展使得移动平台上的任务自动化具备隐私保护和可扩展性。

TABLE 14: Overview of LLM-brained GUI agent frameworks on mobile platforms (Part II).
表14：移动平台上基于大语言模型的GUI代理框架概览（第二部分）。

Agent	Platform	Perception	Action	Model	Architecture	Highlight	Link
Prompt2Task 315]	Android mobile devices	GUI structure and layout hierarchy, full-page textual descriptions OCR-based text extraction	Standard UI operations	GPT-4	Multi-agent architecture	Enables UI automation through free-form textual prompts, eliminating the need for users to script automation tasks.	https: //github.com/ PromptRPA Prompt2TaskDataset
ClickAgent 312]	Android Mobile Devices	Screenshots	Standard UI operations	InternVL-2.0 379], TinyClick 337], SeeClick 25	Single-agent	Combines MLLM reasoning with a dedicated UI location model to enhance UI interaction accuracy	https://github.com/Samsung/ ClickAgent
AutoDroid 156]	Android mobile devices	Simplified HTML-style representation	Standard UI operations	GP f-3.5, GPT-4, and Vicuna-7B 510]	Single-agent architecture	Its use of app-specific knowledge and a multi-granularity query optimization module to reduce the computational cost	https: //autodroid-sys. github.io/
AutoDroid-V2 305]	Android Mobile Devices	Structured GUI Representations	Multi-step scripts of standard UI operations and API calls	LIama-3.1-8B 91]	Script-based architecture.	Converts GUI task automation into a script generation problem, enhancing efficiency and task success rates.	/
CoAT [298]	Android mobile devices	Screenshot-based context and semantic information	Standard UI operations	GPT-4V	Single-agent architecture	The integration of a chain-of-action-thought process, which explicitly maps each action to screen descriptions, reasoning steps, and anticipated outcomes	https: //github.com/ ZhangL-HKU/ CoAT
Mobile-Agent [158]	Mobile Android	Screenshots with detection	Standard UI operations	GPT-4V with Grounding DINO 1831 and CLIP 509 for icon detection	Single-agent	Vision-centric approach that eliminates dependency on system-specific data	https://github.com/X-PLUG/ MobileAgent
Mobile-Agent-v2 306	Mobile Android OS and Harmony OS	Screenshots with text, icon recognition, and description	Standard UI operations on mobile phones	GPT-4V with Grounding DINO 183] and Qwen-VL-Int4 511]	Multi-agent architecture with Planning Agent, Decision Agent, and Reflection Agent	Multi-agent architecture enhances task navigation for long-sequence operations	https://github.com/X-PLUG/ MobileAgent
Mobile-Experts 307	Mobile Android	Interface memory and procedural memory	Standard UI operations and code-combined tool formulation	VLMs	Multi-agent framework with double-layer planning	Code-combined tool formulation method and double-layer planning mechanism for collaborative task execution	/

LiMAC 308	Mobile Android	Screenshots and corresponding widget trees	Standard UI operations	Lightweight transformer and fine-tuned VLMs	Single-agent	Balances computational efficiency and natural language understanding	/
MobA 309	Mobile Android	GUI structures. screenshots with annotation	Standard UI operations and API function calls	GPT-4	Two-level agent: a Global Agent and a Local Agent	Two-level agent system that separates task planning and execution into two specialized agents	https://github.com/OpenDFM/MobA
Mobile-Agent-E 311]	Mobile Android	GUI screen-shots, OCR for detecting text and icons	Standard UI operations and APIs	GPT-4o, Claude- 3.5-Sonnet. Gemini-1.5-Pro	Hierarchical Multi-Agent System	Hierarchical multi-agent framework that separates planning from execution for improved long-term reasoning and self-evolution, enabling the system to learn reusable tips and shortcuts	https: //x-plug.github.io/MobileAgent

代理	平台	感知	动作	模型	架构	高亮	链接
Prompt2Task 315]	安卓移动设备	GUI结构与布局层级，基于OCR的全文本描述提取	标准UI操作	GPT-4	多代理架构	通过自由文本提示实现UI自动化，免除用户编写自动化脚本的需求。	https: //github.com/ PromptRPA Prompt2TaskDataset
ClickAgent 312]	安卓移动设备	截图	标准UI操作	InternVL-2.0 379], TinyClick 337], SeeClick 25	单代理	结合多模态大语言模型（MLLM）推理与专用UI定位模型，提高UI交互准确性	https://github.com/Samsung/ ClickAgent
AutoDroid 156]	安卓移动设备	简化的HTML风格表示	标准UI操作	GPT-3.5、GPT-4和Vicuna-7B 510]	单代理架构	利用应用特定知识和多粒度查询优化模块以降低计算成本	https: //autodroid-sys. github.io/
AutoDroid-V2 305]	安卓移动设备	结构化GUI表示	标准UI操作和API调用的多步骤脚本	LIama-3.1-8B 91]	基于脚本的架构。	将GUI任务自动化转化为脚本生成问题，提高效率和任务成功率。	/
CoAT [298]	安卓移动设备	基于截图的上下文和语义信息	标准UI操作	GPT-4V	单代理架构	集成动作-思考链过程，明确映射每个动作到屏幕描述、推理步骤和预期结果	https: //github.com/ ZhangL-HKU/ CoAT
Mobile-Agent [158]	移动安卓	带检测的截图	标准UI操作	结合Grounding DINO 1831和CLIP 509进行图标检测的GPT-4V	单代理	以视觉为中心的方法，消除对系统特定数据的依赖	https://github.com/X-PLUG/ MobileAgent
Mobile-Agent-v2 306	移动安卓操作系统和鸿蒙操作系统	带文本、图标识别和描述的截图	手机上的标准UI操作	结合Grounding DINO 183]和Qwen-VL-Int4 511]的GPT-4V	包含规划代理、决策代理和反思代理的多代理架构	多代理架构增强了长序列操作的任务导航能力	https://github.com/X-PLUG/ MobileAgent
Mobile-Experts 307	移动安卓	界面记忆和程序记忆	标准UI操作与代码结合的工具制定	视觉语言模型（VLMs）	具有双层规划的多代理框架	代码结合工具制定方法和双层规划机制以协同执行任务	/
LiMAC 308	移动安卓	截图及对应的控件树	标准UI操作	轻量级Transformer和微调的视觉语言模型	单代理	平衡计算效率与自然语言理解	/
MobA 309	移动安卓	GUI结构。带注释的截图	标准用户界面操作和API函数调用	GPT-4	两级代理：全局代理和本地代理	将任务规划与执行分离为两个专门代理的两级代理系统	https://github.com/OpenDFM/ MobA
Mobile-Agent-E 311]	移动安卓	GUI截图，使用OCR检测文本和图标	标准用户界面操作和API	GPT-4o，Claude-3.5-Sonnet，Gemini-1.5-Pro	分层多代理系统	分层多代理框架，将规划与执行分离，以提升长期推理和自我进化能力，使系统能够学习可复用的技巧和捷径	https: //x-plug.github. io/MobileAgent

MobileGPT [310] automates tasks on Android devices using a human-like app memory system that emulates the cognitive process of task decomposition-Explore, Select, Derive, and Recall. This approach results in highly efficient and accurate task automation. Its hierarchical memory structure supports modular, reusable, and adaptable tasks and sub-tasks across diverse contexts. MobileGPT demonstrates superior performance over state-of-the-art systems in task success rates, cost efficiency, and adaptability, highlighting its potential for advancing mobile task automation.

MobileGPT [310] 使用类人应用记忆系统自动化安卓设备上的任务，该系统模拟任务分解的认知过程——探索、选择、推导和回忆。这种方法实现了高效且精准的任务自动化。其分层记忆结构支持模块化、可复用且适应性强的任务及子任务，适用于多种场景。MobileGPT 在任务成功率、成本效益和适应性方面表现优于最先进系统，凸显了其推动移动任务自动化发展的潜力。

In a more advanced distributed setting, FedMobileAgent 314] employs a federated learning framework to train mobile automation agents using self-sourced data from users' phone interactions. It addresses the high cost and privacy concerns associated with traditional human-annotated datasets by introducing Auto-Annotation, which leverages vision-language models (VLMs) to infer user intentions from screenshots and actions. The system enables decentralized training through federated learning while preserving user privacy, and its adaptive aggregation method enhances model performance under non-IID data conditions. Experimental results on several mobile benchmarks demonstrate that FedMobileAgent achieves performance comparable to human-annotated models at a fraction of the cost.
在更先进的分布式环境中，FedMobileAgent [314] 采用联邦学习框架，利用用户手机交互的自有数据训练移动自动化代理。它通过引入自动标注（Auto-Annotation），利用视觉-语言模型（VLMs）从截图和操作中推断用户意图，解决了传统人工标注数据集成本高和隐私问题。该系统通过联邦学习实现去中心化训练，同时保护用户隐私，其自适应聚合方法提升了非独立同分布（non-IID）数据条件下的模型性能。多项移动基准实验结果表明，FedMobileAgent 以极低成本实现了与人工标注模型相当的性能。

In summary, mobile GUI agents have evolved significantly, progressing from single-agent systems to complex, multi-agent frameworks capable of dynamic, context-aware operations. These innovations demonstrate that sophisticated architectures, multimodal processing, and advanced planning strategies are essential in handling the diverse challenges of mobile environments, marking significant advancements in mobile automation capabilities.
总之，移动图形用户界面（GUI）代理经历了显著演进，从单一代理系统发展到能够动态感知上下文的复杂多代理框架。这些创新表明，复杂架构、多模态处理和先进规划策略对于应对移动环境的多样化挑战至关重要，标志着移动自动化能力的重大进步。

TABLE 15: Overview of LLM-brained GUI agent frameworks on mobile platforms (Part III).
表15：移动平台上基于大型语言模型（LLM）驱动的GUI代理框架概览（第三部分）。

Agent	Platform	Perception	Action	Model	Architecture	Highlight	Link
ReachAgent 313]	Android mobile devices	GUI Screen-XML document	Standard UI operations	MobileVLM 353]	Single-agent, two-stage training	Divides tasks into subtasks: "Page Reaching" (navigating to the correct screen) and "Page Operation" (performing actions on the screen). using RL with preference-based training to improve long-term task success.	/
Mobile-Agent-V 316]	Mobile Android	Video guidance, XML lierarchy	Standard UI operations	GPT-40	Multi-agent system	Introduces video-guided learning, allowing the agent to acquire operational knowledge efficiently.	https://github.com/X-PLUG/MobileAgent
MobileSteward 317]	Mobile Android	XML layouts, Screenshots	Standard UI interactions, Code execution	GPT-4V, GPT-4o	App-oriented multi-agent framework	Introduces an app-oriented multi-agent framework with self-evolution, overcoming the complexity of cross-app interactions by dynamically recruiting specialized agents.	https://github com/XiaoMi/MobileSteward
AppAgentX 258]	Mobile Android	Screenshots	Standard UI operations	GPT-40	Single-agent architecture	Introduces an evolutionary mechanism that enables dynamic learning from past interactions and replaces inefficient low-level operations with high-level actions	https: //appagentx. github.io/
CHOP 318]	Mobile Android	Screenshots	Standard UI operations	GPT-40	Multi-agent architecture	Introduces a basis subtask framework, where subtasks are predefined based on human task decomposition patterns, ensuring better exe-cutability and efficiency.	https://github.com/Yuqi-Zhou/CHOP
OS-Kairos 319]	Mobile Android	GUI shots	Standard UI operations	OS-Atlas-Pro- 7B and GPT-40	Single-agent with critic-in-the-loop design	Introduces an adaptive interaction framework where each GUI action is paired with a confidence score, dynamically deciding between autonomous execution and human intervention	https: //github.com/ Wuzheng02/ OS-Kairos

V-Droid 320]	Mobile Android	Android Accessibility Tree	Standard UI operations	LLaMA-3.1-8B-Instruct [91]	Verifier-Driven Single-Agent Architecture	Introduces a novel verifier-driven architecture where the LLM does not generate actions directly but instead scores and selects from a finite set of extracted actions, improving task success rates and significantly reducing latency	/
LearnAct 321]	Mobile Android	GUI screen-shots, UI trees, and demonstration trajectories	Standard GUI actions	Gemini-1.5-Pro, UI-TARS-7BSFT, Qwen2-VL-7B	Multi-agent	Introduces a structured. demonstration-based learning pipeline for mobile GUI agents. It addresses long-tail generalization via few-shot demonstrations, achieving substantial performance gains on complex real-world mobile tasks	https: //Igy0404.github io/LearnAct
AndroidGen 322]	Mobile Android	XML UI structure	Standard GUI actions	GLM-4-9B 499 / LLaMA-3-70B 91]	Multi-module single-agent	Innovatively addresses data scarcity for Android agents through a self-improving architecture, a zero human-annotation training pipeline. and effective generalization from easy to hard tasks	https://github.com/THUDM/AndroidGen
Agent-Initiated Interaction 324]	Android Mobile	Accessibility tree and screenshots	Standard GUI operations	Gemini 1.5	Single-agent architecture	Pioneers agent-initiated interaction in mobile UI automation	https: //github.com/ google-research google-research tree/master/ android interaction
Latent State Estimation 325]	Android Mobile	Accessibility tree	Standard GUI operations	PaLM 2	Two-module design with Reasoner and Grounder	First to formalize the estimation of latent UI states using LLMs to support UI automation	7

代理	平台	感知	动作	模型	架构	高亮	链接
ReachAgent 313]	安卓移动设备	GUI屏幕-XML文档	标准用户界面操作	MobileVLM 353]	单代理，两阶段训练	将任务划分为子任务：“页面到达”（导航至正确屏幕）和“页面操作”（在屏幕上执行动作）。使用基于偏好的强化学习（RL）训练以提升长期任务成功率。	/
Mobile-Agent-V 316]	安卓移动端	视频指导，XML层级结构	标准用户界面操作	GPT-40	多代理系统	引入视频引导学习，使代理能够高效获取操作知识。	https://github.com/X-PLUG/MobileAgent
MobileSteward 317]	安卓移动端	XML布局，屏幕截图	标准用户界面交互，代码执行	GPT-4V, GPT-4o	面向应用的多代理框架	引入面向应用的多代理框架，具备自我进化能力，通过动态招募专门代理克服跨应用交互的复杂性。	https://github com/XiaoMi/MobileSteward
AppAgentX 258]	安卓移动端	屏幕截图	标准用户界面操作	GPT-40	单代理架构	引入进化机制，实现从历史交互中动态学习，并用高级动作替代低效的底层操作。	https: //appagentx. github.io/
CHOP 318]	安卓移动端	屏幕截图	标准用户界面操作	GPT-40	多代理架构	引入基础子任务框架，子任务基于人类任务分解模式预定义，确保更好的可执行性和效率。	https://github.com/Yuqi-Zhou/CHOP
OS-Kairos 319]	安卓移动端	GUI截图	标准用户界面操作	OS-Atlas-Pro-7B 和 GPT-40	带有评审者环路设计的单代理	引入自适应交互框架，每个GUI动作配有置信度评分，动态决定自主执行或人工干预。	https: //github.com/ Wuzheng02/OS-Kairos
V-Droid 320]	安卓移动端	安卓辅助功能树	标准用户界面操作	LLaMA-3.1-8B-Instruct [91]	验证者驱动的单代理架构	引入新颖的验证者驱动架构，LLM不直接生成动作，而是对有限提取动作进行评分和选择，提升任务成功率并显著降低延迟。	/
LearnAct 321]	安卓移动端	GUI屏幕截图、UI树和演示轨迹	标准GUI动作	Gemini-1.5-Pro, UI-TARS-7BSFT, Qwen2-VL- 7B	多代理	引入结构化的基于演示的移动GUI代理学习流程，通过少量示范解决长尾泛化问题，在复杂真实移动任务中取得显著性能提升。	https: //Igy0404.github io/LearnAct
AndroidGen 322]	安卓移动端	XML用户界面结构	标准GUI动作	GLM-4-9B 499 / LLaMA-3-70B 91]	多模块单代理	创新性地通过自我改进架构、零人工标注训练流程及从易到难任务的有效泛化，解决安卓代理数据稀缺问题。	https://github.com/THUDM/AndroidGen
代理发起交互 324]	安卓移动端	辅助功能树和截图	标准图形用户界面操作	Gemini 1.5	单代理架构	开创移动UI自动化中代理发起交互的先河	https: //github.com/ google-research google-research tree/master/ android interaction
潜在状态估计 325]	安卓移动端	辅助功能树	标准图形用户界面操作	PaLM 2	包含推理器（Reasoner）和定位器（Grounder）的双模块设计	首次使用大型语言模型（LLMs）形式化潜在UI状态估计以支持UI自动化	7

12.5 6.3 Computer GUI Agents

12.6 6.3 计算机图形用户界面代理

Computer GUI agents have evolved to offer complex automation capabilities across diverse operating systems, addressing challenges such as cross-application interaction, task generalization, and high-level task planning. They have led to the development of sophisticated frameworks capable of handling complex tasks across desktop environments. These agents have evolved from simple automation tools to intelligent systems that leverage multimodal inputs, advanced architectures, and adaptive learning to perform multi-application tasks with high efficiency and adaptability. We provide an overview of computer GUI agent frameworks in Table 16 and 17
计算机图形用户界面代理已经发展出跨多种操作系统的复杂自动化能力，解决了跨应用交互、任务泛化和高级任务规划等挑战。它们促成了能够处理桌面环境中复杂任务的高级框架的发展。这些代理从简单的自动化工具演变为利用多模态输入、先进架构和自适应学习的智能系统，以高效且灵活地执行多应用任务。我们在表16和表17中提供了计算机图形用户界面代理框架的概述。

TABLE 16: Overview of LLM-brained GUI agent frameworks on computer platforms (Part I)..
表16：计算机平台上基于大型语言模型（LLM）驱动的图形用户界面代理框架概述（第一部分）

Agent	Platform	Perception	Action	Model	Architecture	Highlight	Link
UFO [19]	Windows computer	Screenshots with annotated controls, and properties	Standard UI operations with additional customized operations	GPT-Vision	Dual-agent architecture, consisting of a HostAgent (for application selection and global planning) and an AppAgent (for specific task execution within applications)	Its dual-agent system that seamlessly navigates and interacts with multiple applications to fulfill complex user requests in natural language on Windows OS	https: //github.com/ microsoft/UFO
UFO2 334j	Windows desktops	GUI screen-shots and textual control properties list	Unified GUI-API action layer	GPT-40 (and GPT-4V, o1, Gemini-Flash); Vision grounding via OmniParser-v2	Centralized HostAgent with application-specialized AppAgents	Transforms a conventional CUA into an OS-native, pluggable AgentOS with deep Windows integration, hybrid GUI-API actions, vision + UIA perception, speculative multi-action planning, retrieval-augmented knowledge, and a non-intrusive PiP virtual desktop	https: //github.com/ microsoft/UFO/
ScreenAgent 366]	Linux and Windows desktop	Screenshots	Standard UI operations	ScreenAgent model	Single-agent	Integrated planning-acting-reflecting pipeline that simulates a continuous thought process	https: //github.com/ niuzaisheng/ ScreenAgent
OS-Copilot [162]	Linux and MacOS computer	Unified interface that includes mouse and keyboard control, rol. API calls, and Bash or P Python interpreters	Standard operations, Bash and Python commands, as well as API calls	GPT-4	Multi-component architecture involving a planner, configura-tor, actor, and critic modules	Self-directed learning capability, allowing it to adapt to new applications by autonomously generating and refining tools	https: //os-copilot. github.io/
Cradle [161]	Windows computer	Complete screen videos with Grounding DINO 183] and SAM 182 for object detection and localization	Keyboard and mouse actions	GPT-4	Modular single-agent architecture	Its generalizability across various digital environments. allowing it to operate without relying on internal APIs	https: //baai-agents. github.io/ Cradle/
Agent S 326]	Ubuntu and Windows computer	Screenshots and accessibility tree	Standard UI operations and system-level controls	GPT-4 and Claude-3.5 Sonnet 1631	Multi-agent architecture comprising a Manager and Worker structure	Experience-augmented hierarchical planning	https://github.com/simular-ai/ Agent-S

GUI Narrator 327]	Windows computer	High-resolution screenshots	Standard UI operations	GPT-4 and QwenVL-7B 511]	Two-stage architecture, detecting the cursor location and selecting keyframes. then generating action captions	Uses the cursor as a focal point to improve understanding of high-resolution GUI actions	https://showlab.github.io/ GUI-Narrator
PC Agent 329]	Windows Computer	Screenshots and based tracking	Standard UI Operations	Qwen2-VL-72B-Instruct 231] and Molmo 512	A planning agent for decision-making combined with a arounding agent for executing actions.	Human cognition transfer framework, which transforms raw interaction data into cognitive trajectories to enable complex computer tasks.	https: //gair-nlp.github io/PC-Agent/

代理	平台	感知	动作	模型	架构	高亮	链接
不明飞行物 [19]	Windows 电脑	带注释控件和属性的屏幕截图	标准用户界面操作及额外定制操作	GPT-视觉	双代理架构，由HostAgent（用于应用选择和全局规划）和AppAgent（用于应用内具体任务执行）组成	其双代理系统可无缝导航并交互多个应用，以自然语言完成Windows操作系统上的复杂用户请求	https: //github.com/ microsoft/UFO
UFO2 334j	Windows 桌面	图形用户界面屏幕截图及文本控件属性列表	统一图形用户界面-应用程序接口（GUI-API）动作层	GPT-40（及GPT-4V、o1、Gemini-Flash）；通过OmniParser-v2实现视觉定位	集中式HostAgent与应用专用AppAgents	将传统CUA转变为操作系统原生、可插拔的AgentOS，深度集成Windows，混合GUI-API动作，视觉+UIA感知，推测性多动作规划，检索增强知识，以及非侵入式画中画虚拟桌面	https: //github.com/ microsoft/UFO/
ScreenAgent 366]	Linux和Windows 桌面	屏幕截图	标准用户界面操作	ScreenAgent模型	单代理	集成规划-执行-反思流程，模拟连续思维过程	https: //github.com/ niuzaisheng/ ScreenAgent
OS-Copilot [162]	Linux和MacOS电脑	统一界面，包含鼠标和键盘控制、API调用，以及Bash或Python解释器	标准操作、Bash和Python命令，以及API调用	GPT-4	多组件架构，包含规划器、配置器、执行器和评审模块	自主学习能力，允许通过自主生成和优化工具适应新应用	https: //os-copilot. github.io/
Cradle [161]	Windows 电脑	完整屏幕视频，结合Grounding DINO [183]和SAM [182]进行目标检测与定位	键盘和鼠标操作	GPT-4	模块化单代理架构	其在多种数字环境中的泛化能力，无需依赖内部API即可运行	https: //baai-agents. github.io/ Cradle/
Agent S 326]	Ubuntu和Windows 电脑	屏幕截图及辅助功能树	标准用户界面操作及系统级控制	GPT-4和Claude-3.5 Sonnet 1631	多代理架构，包含管理者和工作者结构	经验增强的分层规划	https://github.com/simular-ai/ Agent-S
GUI Narrator 327]	Windows 电脑	高分辨率屏幕截图	标准用户界面操作	GPT-4和QwenVL-7B [511]	两阶段架构，先检测光标位置和选择关键帧，再生成动作描述	以光标为焦点，提升对高分辨率图形用户界面动作的理解	https://showlab.github.io/ GUI-Narrator
PC Agent 329]	Windows 电脑	屏幕截图及基于跟踪	标准用户界面操作	Qwen2-VL-72B-Instruct 231] 和Molmo 512	一个用于决策的规划代理与一个用于执行动作的环绕代理相结合。	人类认知迁移框架，将原始交互数据转化为认知轨迹，以支持复杂的计算机任务。	https: //gair-nlp.github io/PC-Agent/

One significant development in this area is the introduction of multi-agent architectures that enhance task management and execution. For instance, the UI-Focused Agent, UFO [19] represents a pioneering framework specifically designed for the Windows operating system. UFO redefines UI-focused automation through its advanced dual-agent architecture, leveraging GPT-Vision to interpret GUI elements and execute actions autonomously across multiple applications. The framework comprises a HostAgent, responsible for global planning, task decomposition, and application selection, and an AppAgent, tasked with executing assigned subtasks within individual applications, as illustrated in Figure 23. This centralized structure enables UFO to manage complex, multi-application workflows such as aggregating information and generating reports. Similar architectural approach has also been adopted by other GUI agent frameworks [307], [309], [493]. By incorporating safeguards and customizable actions, UFO ensures efficiency and security when handling intricate commands, positioning itself as a cutting-edge assistant for Windows OS. Its architecture, exemplifies dynamic adaptability and robust task-solving capabilities across diverse applications, demonstrating the potential of multi-agent systems in desktop automation.
该领域的一个重要进展是引入了多智能体架构，以增强任务管理和执行能力。例如，面向用户界面（UI）的智能体UFO [19] 是专为Windows操作系统设计的开创性框架。UFO通过其先进的双智能体架构重新定义了面向UI的自动化，利用GPT-Vision解释图形用户界面（GUI）元素并在多个应用程序中自主执行操作。该框架包括一个负责全局规划、任务分解和应用选择的HostAgent，以及一个负责在各个应用中执行分配子任务的AppAgent，如图23所示。该集中式结构使UFO能够管理复杂的多应用工作流，如信息汇总和报告生成。类似的架构方法也被其他GUI智能体框架采用[307]，[309]，[493]。通过引入安全保障和可定制操作，UFO在处理复杂命令时确保了效率和安全，定位为Windows操作系统的前沿助手。其架构体现了跨多样应用的动态适应性和强大的任务解决能力，展示了多智能体系统在桌面自动化中的潜力。

UFO ² 334, the successor to UFO, elevates GUI automation from a vision-only prototype to a deeply integrated, Windows-native AgentOS (Figure 24). It coordinates tasks through a centralized HostAgent, which delegates subtasks to application-specialized AppAgents. A hybrid perception pipeline that fuses Windows UI Automation (UIA) metadata with OmniParser-v2 visual grounding delivers robust control identification even for custom widgets. Via a unified GUI-API action layer, AppAgents preferentially invoke high-level application APIs and fall back to pixel-level clicks only when necessary, cutting both latency and brittleness. A picture-in-picture virtual desktop cleanly isolates agent execution from the user's main session, enabling non-intrusive
UFO ² 334，作为UFO的继任者，将GUI自动化从仅基于视觉的原型提升为深度集成的Windows原生AgentOS（见图24）。它通过集中式HostAgent协调任务，后者将子任务分配给专门针对应用的AppAgents。融合Windows UI自动化（UIA）元数据与OmniParser-v2视觉定位的混合感知管线，实现了对定制控件的稳健识别。通过统一的GUI-API操作层，AppAgents优先调用高级应用API，仅在必要时回退到像素级点击，显著降低了延迟和脆弱性。画中画虚拟桌面将智能体执行与用户主会话清晰隔离，实现非侵入式

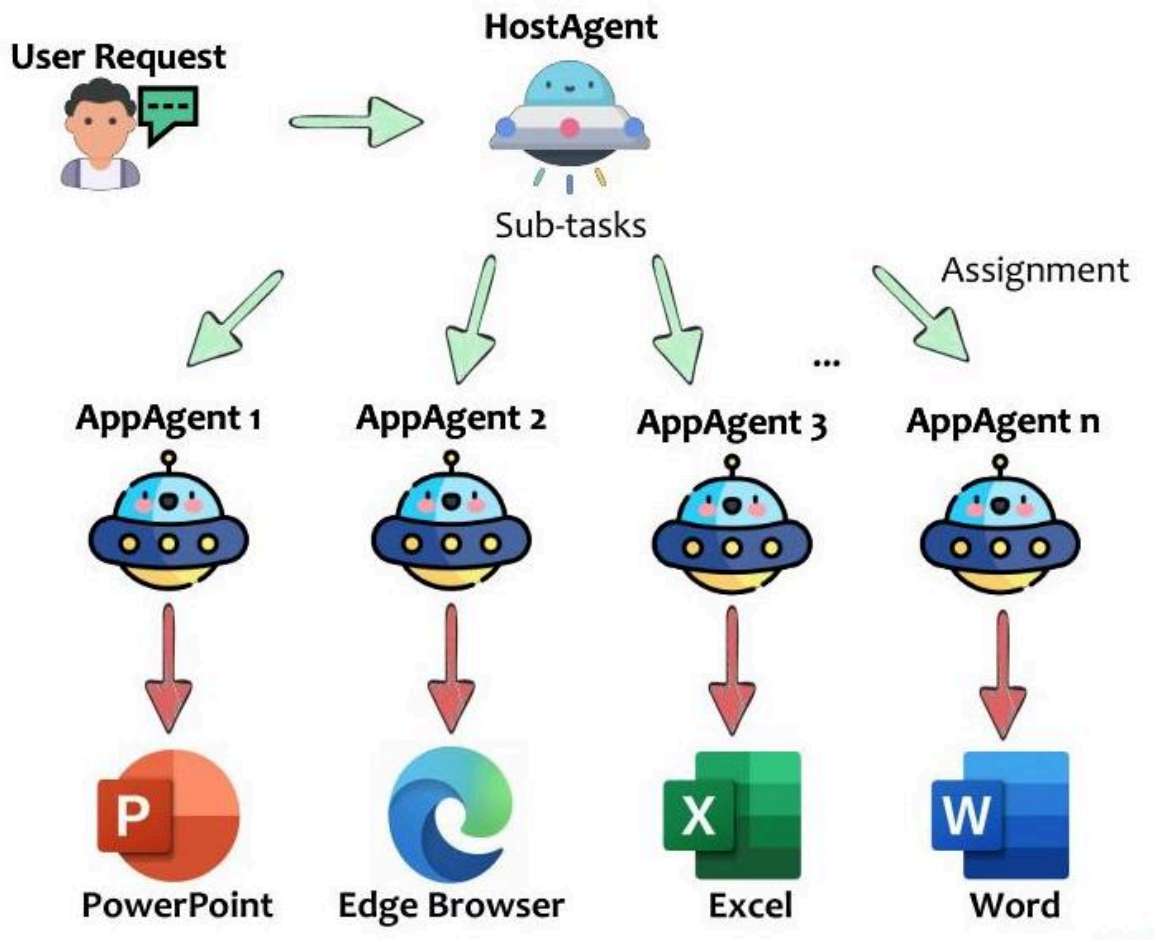

Fig. 23: The multi-agent architecture employed in UFO [19]. Figure adapted from the original paper.
图23：UFO [19]中采用的多智能体架构。图示改编自原论文。

TABLE 17: Overview of LLM-brained GUI agent frameworks on computer platforms (Part II).
表17：计算机平台上基于大型语言模型（LLM）的GUI智能体框架概览（第二部分）。

Agent	Platform	Perception	Action	Model	Architecture	Highlight	Link
Zero-shot Agent 328]	Computer	HTML code and DOM	Standard UI operations	PaLM-2 507]	Single-agent	Zero-shot capability in performing computer control tasks	https: //github.com/ google-research google-research tree/master zero_shot structured reflection
PC-Agent 330]	Windows computers	UI tree, Screen-shots	Standard UI operations	GPT-40	Hierarchica Multi-Agent	PC-Agent's hierarchica multi-agent design enables efficient decomposition of complex PC tasks. Its Active Perception Module enhances fine-grained GUI understanding by combining accessibility structures, OCR, and intention grounding.	https://github com/X-PLUG MobileAgent tree/main PC-Agent
PwP 331]	VSCode-based IDE Computers	Screenshots, File system access, Terminal outputs	Standard UI interactions, File operations, Bash commands, Tools in VSCode	GPT-40, Claude- 3.5 Sonnet, Gemini-1.5	Single-agent architecture	Shifts software engineering agents from API-based tool interactions to direct GUI based computer use, allowing agents to interact with an IDE as a human developer would.	https:// programmingwithpixels com
COLA 332]	Windows computers	GUI structure, properties and screenshots	Standard UI operations and system APIs	GPT-40	Hierarchical Multi-Agent	A dynamic task scheduling mechanism with a plug-and play agent pool, enabling adaptive handling of GUI tasks	https://github.com/Alokia/ COLA-demo
STEVE 333]	Windows Desktop	screen-shots and A11y Tree	Standard UI operations	Qwen2-VL 231 and GPT-40	Single-agent	Introduces a scalable step verification pipeline using GPT-40 to generate binary labels for agent actions, and applies KTO optimization to incorporate both positive and negative actions into agent learning	https://github com/FanbinLu/ STEVE
TaskMind 335]	Windows Computer	Standard GUI actions	GPT-3.5 / GPT-4	Single-agent architecture	Introduces novel task graph representation with cognitive dependencies. enabling LLMs to better generalize demonstrated GUI tasks	https://github.com/ Evennaire/TaskMind	

智能体	平台	感知	动作	模型	架构	高亮	链接
零样本智能体 328]	计算机	HTML代码与DOM	标准用户界面操作	PaLM-2 507]	单智能体	执行计算机控制任务的零样本能力	https://github.com/google-research/google-research/tree/master/zero_shot_structured_reflection
PC-Agent 330]	Windows计算机	用户界面树，屏幕截图	标准用户界面操作	GPT-40	分层多智能体	PC-Agent的分层多智能体设计实现了复杂PC任务的高效分解。其主动感知模块通过结合辅助功能结构、光学字符识别（OCR）和意图定位，增强了细粒度的图形用户界面理解。	https://github com/X-PLUG MobileAgent tree/main PC-Agent
PwP 331]	基于VSCode的集成开发环境计算机	屏幕截图、文件系统访问、终端输出	标准用户界面交互、文件操作、Bash命令、VSCode中的工具	GPT-40, Claude-3.5 Sonnet, Gemini-1.5	单智能体架构	将软件工程智能体从基于API的工具交互转向直接基于图形用户界面的计算机使用，使智能体能够像人类开发者一样与集成开发环境交互。	https://programmingwithpixels com
COLA 332]	Windows计算机	图形用户界面结构、属性和屏幕截图	标准用户界面操作和系统API	GPT-40	分层多智能体	具有即插即用智能体池的动态任务调度机制，实现对图形用户界面任务的自适应处理	https://github.com/Alokia/COLA-demo
STEVE 333]	Windows桌面	屏幕截图和辅助功能树（A11y Tree）	标准用户界面操作	Qwen2-VL 231和 GPT-40	单智能体	引入了可扩展的步骤验证流程，使用GPT-40为智能体动作生成二元标签，并应用KTO优化将正负动作纳入智能体学习中	https://github com/FanbinLu/ STEVE
TaskMind 335]	Windows计算机	标准图形用户界面操作	GPT-3.5 / GPT-4	单智能体架构	引入了具有认知依赖关系的新型任务图表示，增强大型语言模型（LLMs）对演示图形用户界面任务的泛化能力	https://github.com/Evennaire/TaskMind	

multitasking. Runtime performance is further boosted by retrieval-augmented help documents and execution logs, coupled with speculative multi-action planning that executes several steps per single LLM invocation. Tested on 20+ real Windows applications, **UFO**^{2} exceeds Operator [513] and other CUAs by more than 10 percentage points in success rate while halving LLM calls. Because the framework is model-agnostic, swapping GPT-40 for a stronger LLM such as o1 yields additional gains without code changes.

多任务处理。通过检索增强的帮助文档和执行日志，以及执行多步骤的推测性多动作规划（每次调用大型语言模型（LLM）执行多个步骤），进一步提升了运行时性能。在20多个真实Windows应用程序上的测试中，**UFO**^{2}的成功率比Operator [513]及其他计算机用户代理（CUA）高出10个百分点以上，同时将LLM调用次数减半。由于该框架与模型无关，替换GPT-40为更强大的LLM如o1，无需更改代码即可获得额外提升。

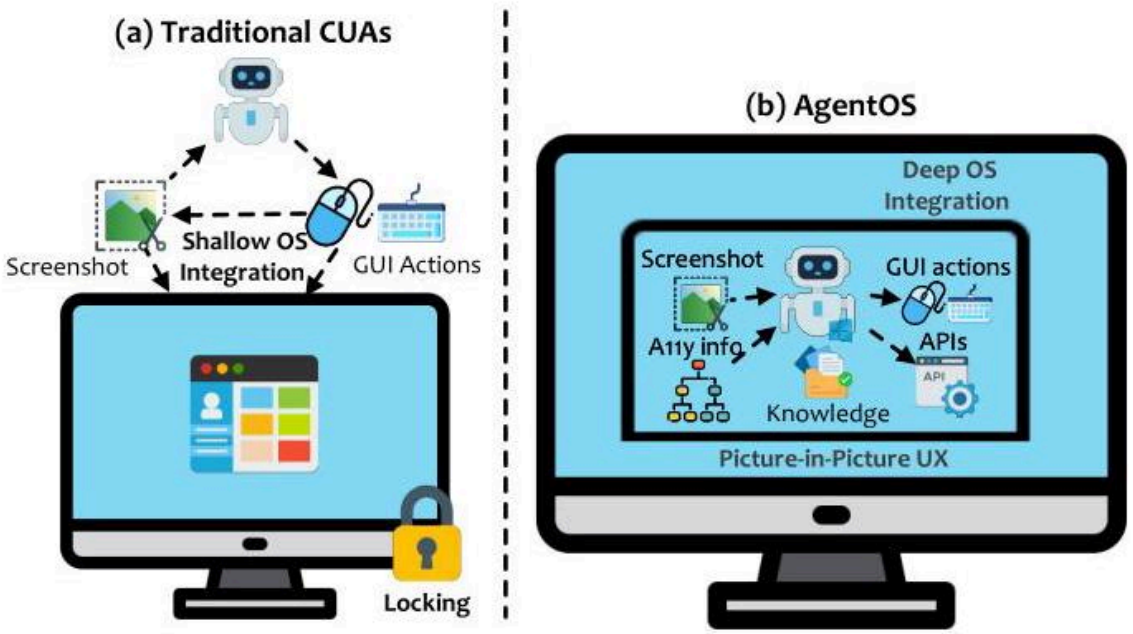

Fig. 24: The comparison of traditional CUAs and the Desktop AgentOS UFO2. Figure adapted from the original paper.
图24：传统计算机用户代理（CUA）与桌面AgentOS UFO2的比较。图示改编自原论文。

Building upon the theme of adaptability and generalist capabilities, Cradle [161] pushes the boundaries of general computer control by utilizing VLMs for interacting with various software, ranging from games to professional applications, without the need for API access. Cradle employs GPT-40 to interpret screen inputs and perform low-level actions, making it versatile across different types of software environments. Its six-module structure, covering functions such as information gathering and self-reflection, enables the agent to execute tasks, reason about actions, and utilize past interactions to inform future decisions. Cradle's capacity to function in dynamic environments, including complex software, marks it as a significant step toward creating generalist agents with broad applicability across desktop environments.
基于适应性和通用能力的主题，Cradle [161]通过利用视觉语言模型（VLM）与各种软件（从游戏到专业应用）交互，推动了通用计算机控制的边界，无需API访问。Cradle采用GPT-40解释屏幕输入并执行底层操作，使其在不同软件环境中具有多功能性。其六模块结构，涵盖信息收集和自我反思等功能，使代理能够执行任务、推理行动并利用过去交互指导未来决策。Cradle在动态环境（包括复杂软件）中的运行能力，标志着向具有广泛桌面环境适用性的通用代理迈出了重要一步。

Extending the capabilities of computer GUI agents to multiple operating systems, OS-Copilot [162] introduces a general-purpose framework designed to operate across Linux and macOS systems. Its notable feature, FRIDAY, showcases the potential of self-directed learning by adapting to various applications and performing tasks without explicit training for each app. Unlike application-specific agents, FRIDAY integrates APIs, keyboard and mouse controls, and command-line operations, creating a flexible platform that can autonomously generate and refine tools as it interacts with new applications. OS-Copilot's ability to generalize across unseen applications, validated by its performance on the GAIA benchmark, provides a foundational model for OS-level agents capable of evolving in complex environments. This demonstrates promising directions for creating adaptable digital assistants that can handle diverse desktop environments and complex task requirements.
OS-Copilot [162]将计算机图形用户界面（GUI）代理的能力扩展到多个操作系统，推出了一个设计用于Linux和macOS系统的通用框架。其显著特性FRIDAY展示了自我导向学习的潜力，能够适应各种应用并执行任务，无需针对每个应用进行显式训练。与特定应用代理不同，FRIDAY整合了API、键盘和鼠标控制以及命令行操作，创建了一个灵活的平台，能够在与新应用交互时自主生成和优化工具。OS-Copilot在GAIA基准测试中的表现验证了其对未见应用的泛化能力，为能够在复杂环境中进化的操作系统级代理提供了基础模型。这展示了创建适应性数字助理以处理多样桌面环境和复杂任务需求的有希望方向。

In the emerging field of LLM-powered GUI agents for desktop environments, Programming with Pixels (PwP) 331 introduces a compelling alternative to traditional tool-based software engineering agents. Rather than relying on predefined API calls, PwP enables agents to interact directly with an IDE using visual perception, keyboard inputs, and mouse clicks, mimicking the way human developers operate within an IDE. This approach allows for generalization beyond predefined APIs, providing a highly expressive environment where agents can execute a wide range of software engineering tasks, including debugging, UI generation, and code editing. Evaluations conducted on PwP-Bench demonstrate that computer-use agents, despite lacking direct access to structured APIs, can match or even surpass traditional tool-based approaches in certain scenarios.
在基于大型语言模型（LLM）的桌面环境GUI代理新兴领域，Programming with Pixels（PwP）331提出了传统基于工具的软件工程代理的有力替代方案。PwP不依赖预定义的API调用，而是使代理通过视觉感知、键盘输入和鼠标点击直接与集成开发环境（IDE）交互，模拟人类开发者在IDE中的操作方式。这种方法允许超越预定义API的泛化，提供了一个高度表达性的环境，使代理能够执行包括调试、用户界面

生成和代码编辑在内的广泛软件工程任务。在PwP-Bench上的评估表明，尽管缺乏对结构化API的直接访问，计算机使用代理在某些场景下能够匹配甚至超越传统基于工具的方法。

In summary, computer GUI agents have evolved significantly, progressing from single-task automation tools to advanced multi-agent systems capable of performing complex, multi-application tasks and learning from interactions. Frameworks like UFO, Cradle, and OS-Copilot illustrate the potential of adaptable, generalist agents in desktop automation, paving the way for the evolution of more intelligent and versatile AgentOS frameworks.
总之，计算机GUI代理经历了显著演进，从单任务自动化工具发展为能够执行复杂多应用任务并从交互中学习的高级多代理系统。UFO、Cradle和OS-Copilot等框架展示了适应性通用代理在桌面自动化中的潜力，为更智能多功能的AgentOS框架的发展铺平了道路。

12.7 6.4 Cross-Platform GUI Agents

12.8 6.4 跨平台GUI代理

Cross-platform GUI agents have emerged as versatile solutions capable of interacting with various environments, from desktop and mobile platforms to more complex systems. These frameworks prioritize adaptability and efficiency, leveraging both lightweight models and multi-agent architectures to enhance cross-platform operability. In this subsection, we first We overview cross-platform GUI agent frameworks in Table 18 then explore key frameworks that exemplify the advancements in cross-platform GUI automation.
跨平台GUI代理作为多功能解决方案出现，能够与从桌面和移动平台到更复杂系统的各种环境交互。这些框架优先考虑适应性和效率，利用轻量级模型和多代理架构提升跨平台操作性。本小节首先在表18中概述跨平台GUI代理框架，然后探讨体现跨平台GUI自动化进展的关键框架。

A significant stride in this domain is represented by Au-toGLM [336], which bridges the gap between web browsing and Android control by integrating large multimodal models for seamless GUI interactions across platforms. AutoGLM introduces an Intermediate Interface Design that separates planning and grounding tasks, improving dynamic decision-making and adaptability. By employing a self-evolving online curriculum with reinforcement learning, the agent learns incrementally from real-world feedback and can recover from errors. This adaptability and robustness make AutoGLM ideal for real-world deployment in diverse user applications, setting a new standard in cross-platform automation and offering promising directions for future research in foundation agents.
该领域的重要进展由AutoGLM [336]代表，它通过整合大型多模态模型，弥合了网页浏览与Android控制之间的差距，实现了跨平台的无缝GUI交互。AutoGLM引入了中间接口设计，将规划与落地任务分离，提升了动态决策和适应能力。通过采用带有强化学习的自我进化在线课程，代理能够从真实反馈中逐步学习并从错误中恢复。这种适应性和鲁棒性使AutoGLM成为多样用户应用中实际部署的理想选择，树立了跨平台自动化的新标准，并为基础代理的未来研究提供了有前景的方向。

While some frameworks focus on integrating advanced models for cross-platform interactions, others emphasize efficiency and accessibility. TinyClick [337] addresses the need for lightweight solutions by focusing on single-turn interactions within GUIs. Utilizing the Florence-2-Base Vision-Language Model, TinyClick executes tasks based on user commands and screenshots with only 0.27 billion parameters. Despite its compact size, it achieves high accuracy-73% on Screenspot [25] and 58.3% on OmniAct [459] —outperforming larger multimodal models like GPT-4V while maintaining efficiency. Its multi-task training and MLLM-based data augmentation enable precise UI element localization, making it suitable for low-resource environments and addressing latency and resource constraints in UI grounding and action execution.
虽然一些框架侧重于整合先进模型以实现跨平台交互，另一些则强调效率和可及性。TinyClick [337] 针对轻量级解决方案的需求，专注于图形用户界面（GUI）中的单轮交互。利用 Florence-2-Base 视觉语言模型（Vision-Language Model），TinyClick 基于用户命令和截图执行任务，参数量仅为2.7亿。尽管体积小巧，它在 Screenspot [25] 上达到73%的高准确率，在 OmniAct [459] 上达到58.3%，表现优于如 GPT-4V 等更大型多模态模型，同时保持高效。其多任务训练和基于多模态大语言模型（MLLM）的数据增强实现了精确的UI元素定位，适用于资源受限环境，解决了UI定位和动作执行中的延迟及资源限制问题。

In addition to lightweight models, multi-agent architectures play a crucial role in enhancing cross-platform GUI interactions. OSCAR 340 exemplifies this approach by introducing a generalist GUI agent capable of autonomously navigating and controlling both desktop and mobile applications. By utilizing a state machine architecture, OSCAR dynamically handles errors and adjusts its actions based on real-time feedback, making it suitable for automating complex workflows guided by natural language. The integration of standardized OS controls, such as keyboard and mouse inputs, allows OSCAR to interact with applications in a generalized manner, improving productivity across diverse GUI environments. Its open-source design promotes broad adoption and seamless integration, offering a versatile tool for cross-platform task automation and productivity enhancement.
除了轻量级模型，多智能体架构在提升跨平台GUI交互中也发挥着关键作用。OSCAR 340 体现了这一方法，推出了一种通用GUI智能体，能够自主导航和控制桌面及移动应用。通过采用状态机架构，OSCAR 动态处理错误并根据实时反馈调整操作，适合自动化由自然语言驱动的复杂工作流程。集成标准化操作系统控件，如键盘和鼠标输入，使OSCAR能够以通用方式与应用交互，提升多样化GUI环境下的生产力。其开源设计促进了广泛采用和无缝集成，提供了一个多功能工具，用于跨平台任务自动化和生产力提升。

Expanding on the concept of multi-agent systems, AgentStore 341 introduces a flexible and scalable framework for integrating heterogeneous agents to automate tasks across operating systems. The key feature of AgentStore is the MetaAgent, which uses the innovative AgentToken strategy to dynamically manage a growing number of specialized agents. By enabling dynamic agent enrollment, the framework fosters adaptability and scalability, allowing both specialized and generalist capabilities to coexist. This multi-agent

architecture supports diverse platforms, including desktop and mobile environments, leveraging multimodal perceptions such as GUI structures and system states. AgentStore's contributions highlight the importance of combining specialization with generalist capabilities to overcome the limitations of previous systems.
在多智能体系统概念基础上，AgentStore 341 引入了一个灵活且可扩展的框架，用于整合异构智能体以实现跨操作系统的任务自动化。AgentStore 的核心特性是 MetaAgent，采用创新的 AgentToken 策略动态管理不断增长的专业智能体数量。通过支持动态智能体注册，该框架促进了适应性和可扩展性，使专业化与通用能力共存。这种多智能体架构支持包括桌面和移动环境在内的多平台，利用多模态感知如GUI结构和系统状态。AgentStore 的贡献凸显了结合专业化与通用能力以克服以往系统局限的重要性。

TABLE 18: Overview of LLM-brained cross-platform GUI agent frameworks.
表18：基于大语言模型（LLM）的跨平台GUI智能体框架概述。

Agent	Platform	Perception	Action	Model	Architecture	Highlight	Link
AutoGLM 336]	Web and Mobile Android	Screenshots with SoM annotation and OCR	Standard UI operations, Native API interactions, and AI-driven actions	ChatGLM 499	Single-agent architecture	Self-evolving online curriculum RL framework, which enables continuous improvement by interacting with real-world environments	https:// xiao9905.github io/AutoGLM/
TinyClick 337]	Web, Mobile, and Windows platforms	GUI screen-shots	Standard UI operations, Native API interactions, and AI-driven actions	Florence-2- Base VLM 514	Single-agent, with single-turn tasks	Compact size (0.27B parameters) with high performance	https: //huggingface co/Samsung/ TinyClick
OSCAR 340]	Desktop and Mobile	Screenshots	Standard UI operations	GPT-4	Single-agent architecture	Ability to adapt to real-time feedback and dynamically adjust its actions	/
AgentStore 341]	Desktop and mobile environments	GUI structures and properties. accessibility trees, screenshots and terminal output etc	Standard UI operations, API calls	GPT-40 an InternVL2-8B 379]	Multi-agent architecture	Dynamically integrate a wide variety of heterogeneous agents, enabling both spe cialized and generalist capabilities	https: //chengyou-jia. github.io/ AgentStore-Home/
MMAC-Copilot 249]	Windows OS Desktop, mobile applications, and game environments	Screenshots	Standard UI operations. Native APIs, and Collaborative multi-agent actions	GPT-4V, SeeClick 25 and Genimi Vision different agents	Multi-agent architecture with Planner, Programmer Viewer, N Mentor Video Analyst, and Librarian	Collaborative multi-agent architecture where agents specialize in specific tasks	/
AGUVIS 236	Web, desktop, and mobile	Image-based observations	Standard UI operations	Fine-tuned Qwen2-VL 231]	Single-agent architecture	Pure vision-based approach for GUI interaction, bypassing textual UI representations and enabling robust cross-platform generalization	https: //aguvis-project. github.io
Ponder Press 342]	Web, Android, iOS Mobile, Windows, and macOS	Purely visual inputs	Standard UI operations	GPT-40 and Claude Sonnet high-level task decomposition, a fine-tuned Qwen2-VL-Instruct 231 for GUI element grounding	Divide-and-conquer architecture	Purely vision-based GUI agent that does not require non-visual inputs	https: //invinciblewyq. github.io/ ponder-press-page

InfiGUIAgent 343]	Mobile, Web, Desktop	Raw screen-shots	Standard UI operations.	Qwen2-VL-2B 231]	Single-agent architecture enhanced by hierarchical reasoning.	Introduces native reasoning skills, such as hierarchical and expectation-reflection reasoning, enabling advanced and adaptive task handling.	https: //github.com/ RealIm-Labs/ InfiGUIAgent
Learn-by-Interact [338]	Web, code development, and desktops	GUI screen-shots with SoM and accessibility tree	Standard UI interactions and code execution	Claude-3.5-Sonnet, Gemini-1.5-Pro CodeGemma-7B, CodeStral-22B	Multi-agent	Introduces autonomous data synthesis process, eliminating the need for human-labeled agentic data	/
CollabUIAgen 339]	sMobile Android, Web	Screenshots, UI trees	Standard UI operations	Qwen2-7B 231], GPT-4	Multi-agent system	A multi-agent reinforcement learning framework that introduces a Credit Re-Assignment (CR) strategy, using LLMs instead of environment specific rewards to enhance performance and generalization.	https: //github.com/ THUNLP-MT/ CollabUIAgents
Agent S2 344]	Ubuntu, Windows, Android	GUI screenshot	Standard UI operations and system APIs	Claude-3.7-Sonnet, Claude-3.5-Sonnet, GPT-40 (for Manager and Worker roles), UI-TARS-72B-DPO, Tesseract OCR, and UNO (for grounding experts)	Compositional multi-agent architecture with a Manager for planning, a Worker for execution, and a Mixture of Grounding experts	Features a Mixture of Grounding technique and Proactive Hierarchical Planning, enabling more accurate grounding and adaptive replanning in long-horizon tasks	https://github.com/simular-ai/ Agent-S
GuidNav 345]	Android and Web	GUI shots screen-	Standard UI operations and system APIs	GPT-40, Gemini 2.0 Flash, Qwen-VL-Plus	Single-agent	Introduces a novel process reward model that provides fine-grained, step-level feedback to enhance GUI task accuracy and success	/

ScaleTrack 346]	Web, Android Mobile, and Desktop Computers	GUI shots screen-	Standard GUI operations	Qwen2-VL-7B	Single-agent	First GUI agent framework to introduce backtracking—learning not only the next action but also historical action sequences	/

代理	平台	感知	动作	模型	架构	高亮	链接
AutoGLM 336]	网页和移动安卓	带有SoM注释和OCR的截图	标准UI操作、本地API交互及AI驱动动作	ChatGLM 499	单代理架构	自我进化的在线课程强化学习框架，能够通过与现实环境交互实现持续改进	https:// xiao9905.github io/AutoGLM/
TinyClick 337]	网页、移动和Windows平台	图形用户界面截图	标准UI操作、本地API交互及AI驱动动作	Florence-2- Base VLM 514	单代理，单轮任务	体积小（0.27B参数）且性能高	https: //huggingface co/Samsung/ TinyClick
OSCAR 340]	桌面和移动端	截图	标准UI操作	GPT-4	单代理架构	具备适应实时反馈并动态调整动作的能力	/
AgentStore 341]	桌面和移动环境	图形用户界面结构和属性、辅助功能树、截图及终端输出等	标准UI操作、API调用	GPT-40 和 InternVL2-8B 379]	多代理架构	动态整合多种异构代理，实现专业化与通用能力兼备	https: //chengyou-jia. github.io/ AgentStore-Home/
MMAC-Copilot 249]	Windows操作系统桌面、移动应用及游戏环境	截图	标准UI操作、本地API及协作多代理动作	GPT-4V、SeeClick 25和 Genimi Vision 不同代理	包含规划者、程序员、查看者、导师、视频分析师和图书管理员的多代理架构	协作多代理架构，代理专注于特定任务	/
AGUVIS 236	网页、桌面和移动端	基于图像的观察	标准UI操作	微调的Qwen2-VL 231]	单代理架构	纯视觉驱动的图形用户界面交互方法，绕过文本UI表示，实现强健的跨平台泛化	https: //aguvis-project. github.io
Ponder Press 342]	网页、安卓、iOS移动端、Windows和macOS	纯视觉输入	标准UI操作	GPT-40和Claude Sonnet进行高级任务分解，微调的Qwen2-VL-Instruct 231用于图形用户界面元素定位	分而治之架构	纯视觉驱动的图形用户界面代理，无需非视觉输入	https: //invinciblewyq. github.io/ ponder-press-page
InfiGUIAgent 343]	移动端、网页、桌面	原始截图	标准UI操作。	Qwen2-VL-2B 231]	通过层级推理增强的单代理架构。	引入本地推理技能，如层级推理和期望反思推理，实现高级且自适应的任务处理。	https: //github.com/ RealIm-Labs/ InfiGUIAgent
交互式学习 [338]	网页、代码开发与桌面环境	带有SoM和辅助功能树的GUI截图	标准用户界面交互与代码执行	Claude-3.5-Sonnet，Gemini-1.5-Pro，CodeGemma-7B，CodeStral-22B	多智能体	引入自主数据合成过程，消除对人工标注智能体数据的需求	/
CollabUIAgen [339]	移动端Android，网页	截图，用户界面树	标准UI操作	Qwen2-7B [231]，GPT-4	多智能体系统	一种多智能体强化学习框架，引入信用重新分配（Credit Re-Assignment, CR）策略，使用大型语言模型（LLMs）替代环境特定奖励，以提升性能和泛化能力。	https://github.com/THUNLP-MT/CollabUIAgents

Agent S2 [344]	Ubuntu，Windows，Android	GUI截图	标准用户界面操作与系统API	Claude-3.7-Sonnet，Claude-3.5-Sonnet，GPT-40（用于管理者和执行者角色），UI-TARS-72B-DPO，Tesseract OCR，UNO（用于定位专家）	组合式多智能体架构，包含负责规划的管理者、负责执行的执行者及多位定位专家的混合体	采用定位专家混合技术与主动分层规划，实现更精准的定位与长远任务中的自适应重新规划	https://github.com/simular-ai/Agent-S
GuidNav [345]	Android与网页	GUI截图	标准用户界面操作与系统API	GPT-40，Gemini 2.0 Flash，Qwen-VL-Plus	单智能体	引入一种新颖的过程奖励模型，提供细粒度的步骤级反馈，以提升GUI任务的准确性和成功率	/
ScaleTrack [346]	网页、Android移动端及桌面计算机	GUI截图	标准GUI操作	Qwen2-VL-7B	单智能体	首个引入回溯机制的GUI智能体框架——不仅学习下一步动作，还学习历史动作序列	/

Further advancing cross-platform GUI interaction, MMAC-Copilot [249] employs a multi-agent, multimodal approach to handle tasks across 3D gaming, office, and mobile applications without relying on APIs. By utilizing specialized agents like Planner, Viewer, and Programmer, MMAC-Copilot collaborates to adapt to the complexities of visually rich environments. Using GPT-4V for visual recognition and OCR for text analysis, it achieves high task completion rates in visually complex environments. The framework's integration with VIBench, a benchmark for non-API applications, underscores its real-world relevance and adaptability. MMAC-Copilot's robust foundation for dynamic interaction across platforms extends applications to industries like gaming, healthcare, and productivity.
进一步推进跨平台GUI交互，MMAC-Copilot [249]采用多智能体、多模态方法处理3D游戏、办公和移动应用中的任务，无需依赖API。通过利用如Planner、Viewer和Programmer等专用智能体，MMAC-Copilot协同适应视觉丰富环境的复杂性。借助GPT-4V进行视觉识别和OCR进行文本分析，在视觉复杂环境中实现高任务完成率。该框架与非API应用基准VIBench的集成，凸显其现实相关性和适应性。MMAC-Copilot为跨平台动态交互奠定了坚实基础，拓展了游戏、医疗和生产力等行业的应用。

AGUVIS 236] leverages a pure vision approach to automate GUI interactions, overcoming limitations of text-based systems like HTML or accessibility trees. Its platform-agnostic design supports web, desktop, and mobile applications while reducing inference costs. AGUVIS employs a two-stage training process: the first focuses on GUI grounding, and the second integrates planning and reasoning within a unified model. This approach delivers state-of-the-art performance in both offline and online scenarios, streamlining decision-making and execution.
AGUVIS [236]利用纯视觉方法自动化GUI交互，克服了基于文本系统如HTML或辅助功能树的局限性。其平台无关设计支持网页、桌面和移动应用，同时降低推理成本。AGUVIS采用两阶段训练过程：第一阶段专注于GUI定位，第二阶段在统一模型中整合规划与推理。该方法在离线和在线场景中均实现了最先进的性能，简化了决策和执行流程。

Agent S2 [344] builds upon its predecessor, Agent S [326], by introducing a hierarchical and compositional framework for GUI agents that integrates generalist models with specialized grounding modules. Departing from monolithic architectures, it employs a Mixture of Grounding (MoG) strategy to delegate fine-grained grounding tasks to expert modules, and adopts Proactive Hierarchical Planning (PHP) to dynamically revise action plans based on evolving observations. Relying solely on GUI screenshots, Agent S2 generalizes effectively across Ubuntu, Windows, and Android platforms. It demonstrates strong scalability and consistently outperforms larger monolithic models by strategically distributing cognitive responsibilities. The design of Agent S2 underscores the advantages of modular architectures for handling long-horizon, high-fidelity GUI interactions.
Agent S2 [344]在其前身Agent S [326]基础上，提出了一个层次化和组合式的GUI智能体框架，将通用模型与专用定位模块结合。它摒弃了单一架构，采用定位混合（Mixture of Grounding, MoG）策略，将细粒度定位任务分配给专家模块，并采用主动层次规划（Proactive Hierarchical Planning, PHP）根据不断变化的观察动态调整行动计划。仅依赖GUI截图，Agent S2在Ubuntu、Windows和Android平台上表现出良好的泛化能力。通过战略性分配认知职责，展现出强大的可扩展性，持续超越更大规模的单一模型。Agent S2的设计凸显了模块化架构在处理长时程、高保真GUI交互中的优势。

In summary, cross-platform GUI agents exemplify the future of versatile automation, offering solutions ranging from lightweight models like TinyClick to sophisticated multi-agent systems such as MMAC-Copilot. Each framework brings unique innovations, contributing to a diverse ecosystem of GUI automation tools that enhance interaction capabilities across varying platforms, and marking significant advancements in cross-platform GUI automation.
总之，跨平台GUI智能体代表了多功能自动化的未来，涵盖从轻量级模型如TinyClick到复杂多智能体系统如MMAC-Copilot的解决方案。每个框架都带来了独特创新，促进了多样化的GUI自动化工具生态系统，提升了不同平台间的交互能力，标志着跨平台GUI自动化的重大进展。

12.9 6.5 Takeaways

12.10 6.5 关键要点

The landscape of GUI agent frameworks has seen notable advancements, particularly in terms of multi-agent architectures, multimodal inputs, and enhanced action sets. These developments are laying the groundwork for more versatile and powerful agents capable of handling complex, dynamic environments. Key takeaways from recent advancements include:
GUI智能体框架领域取得了显著进展，尤其是在多智能体架构、多模态输入和增强动作集方面。这些发展为更通用、更强大的智能体奠定了基础，使其能够应对复杂且动态的环境。近期进展的主要要点包括：

1. Multi-Agent Synergy: Multi-agent systems, such as those in UFO [19] and MMAC-Copilot [249], represent a significant trend in GUI agent development. By assigning specialized roles to different agents within a framework, multi-agent systems can enhance task efficiency, adaptability, and overall performance. As agents take on more complex tasks across diverse platforms, the coordinated use of multiple agents is proving to be a powerful approach, enabling agents to handle intricate workflows with greater precision and speed.
2. 多智能体协同：多智能体系统，如UFO [19]和MMAC-Copilot [249]，是GUI智能体发展的重要趋势。通过在框架内分配不同智能体的专门角色，多智能体系统能够提升任务效率、适应性和整体性能。随着智能体承担跨多平台的复杂任务，多个智能体的协调使用被证明是一种强有力的方法，使智能体能够更精准、更快速地处理复杂工作流。

2. Multimodal Input Benefits: While some agents still rely solely on text-based inputs (e.g., DOM structures or HTML), incorporating visual inputs, such as screenshots, has shown clear performance advantages. Agents like WebVoyager [269] and SeeAct [17] highlight how visual data, combined with textual inputs, provides a richer representation of the environment state, helping agents make better-informed decisions. This integration of multimodal inputs is essential for accurate interpretation in visually complex or dynamic environments where text alone may not capture all necessary context.
3. 多模态输入优势：尽管部分智能体仍仅依赖基于文本的输入（如DOM结构或HTML），但引入视觉输入（如截图）已显示出明显的性能优势。WebVoyager [269]和SeeAct [17]等智能体展示了视觉数据与文本输入结合如何提供更丰富的环境状态表示，帮助智能体做出更明智的决策。在视觉复杂或动态环境中，单靠文本难以捕捉所有必要上下文，多模态输入的整合至关重要。

3. Expanding Action Sets Beyond UI Operations: Recent agents have expanded their action sets beyond standard UI operations to include API calls and AI-driven actions, as seen in Hybrid Agent 199 and AutoWebGLM 270]. Incorporating diverse actions allows agents to achieve higher levels of interaction and task completion, particularly in environments where data can be directly retrieved or manipulated through API calls. This flexibility enhances agent capabilities, making them more efficient and adaptable across a wider range of applications.
4. 扩展动作集超越UI操作：近期智能体扩展了动作集，除了标准UI操作外，还包括API调用和AI驱动动作，如Hybrid Agent [199]和AutoWebGLM [270]所示。多样化动作的引入使智能体能够实现更高级别的交互和任务完成，尤其是在可以通过API直接获取或操作数据的环境中。这种灵活性增强了智能体的能力，使其在更广泛的应用中更高效、更具适应性。

4. Emerging Techniques for Improved Decision-Making: Novel approaches such as world models in WMA [266] and search-based strategies in Search-Agent 274 represent promising directions for more advanced decision-making. World models allow agents to simulate action outcomes, reducing unnecessary interactions and improving efficiency, especially in long-horizon tasks. Similarly, search-based algorithms like best-first and MCTS help agents explore action pathways more effectively, enhancing their adaptability in complex, real-time environments.
5. 改进决策的新兴技术：如WMA [266]中的世界模型和Search-Agent [274]中的基于搜索的策略，代表了更先进决策的有前景方向。世界模型使智能体能够模拟动作结果，减少不必要的交互，提高效率，尤其适用于长时程任务。类似地，基于搜索的算法如最佳优先搜索和蒙特卡洛树搜索（MCTS）帮助智能体更有效地探索动作路径，增强其在复杂实时环境中的适应能力。

5. Toward Cross-Platform Generalization: Cross-platform frameworks, such as AutoGLM [336] and OSCAR [340], underscore the value of generalizability in GUI agent design. These agents are pioneering efforts to create solutions that work seamlessly across mobile, desktop, and web platforms, moving closer to the goal of a one-stop GUI agent that can operate across multiple ecosystems. Cross-platform flexibility will be crucial for agents that aim to assist users consistently across their digital interactions.
6. 跨平台泛化方向：跨平台框架，如AutoGLM[336]和OSCAR[340]，强调了GUI代理设计中泛化能力的重要性。这些代理是旨在实现移动端、桌面端和网页端无缝协作的开创性尝试，逐步接近能够跨多个生态系统运行的一站式GUI代理目标。跨平台的灵活性对于那些希望在用户的数字交互中持续提供帮助的代理来说至关重要。

6. Pure Vision-Based Agent: To enable universal GUI control, pure vision-based frameworks have emerged as a prominent solution. These agents rely solely on screenshots for decision-making, eliminating the need for access to metadata such as widget trees or element properties. Notable work like AGUVIS [236] exemplifies this approach. While pure vision-based methods offer greater generalizability and bypass system API limitations, they require strong "grounding" capabilities to accurately locate and interact with UI elements—an ability often lacking in many foundational models. Fine-tuning models specifically for visual grounding and GUI understanding, or integrating GUI parsing techniques like OmniParser [184], can address this challenge and enhance the agent's ability to perform precise interactions.

7. 纯视觉驱动代理：为实现通用的GUI控制，纯视觉驱动框架成为一种突出解决方案。这类代理完全依赖截图进行决策，无需访问诸如控件树或元素属性等元数据。AGUVIS[236]等重要工作即为此类方法的典范。虽然纯视觉方法具备更强的泛化能力并绕过了系统API的限制，但它们需要强大的“定位”能力以准确识别和操作UI元素——这一能力在许多基础模型中往往缺失。通过针对视觉定位和GUI理解进行模型微调，或结合如OmniParser[184]的GUI解析技术，可以解决这一挑战，提升代理执行精确交互的能力。

The field of GUI agents is moving towards multi-agent architectures, multimodal capabilities, diverse action sets, and novel decision-making strategies. These innovations mark significant steps toward creating intelligent, adaptable agents capable of high performance across varied and dynamic environments. The future of GUI agents lies in the continued refinement of these trends, driving agents towards broader applicability and more sophisticated, human-like interactions across platforms.
GUI代理领域正朝着多代理架构、多模态能力、多样化动作集和新颖决策策略发展。这些创新标志着向构建智能、适应性强且能在多变环境中高效表现的代理迈出了重要步伐。GUI代理的未来在于持续完善这些趋势，推动代理实现更广泛的适用性和更复杂、更具人性化的跨平台交互。

13 7 Data for Optimizing LLM-Brained GUI AGENTS

14 7 用于优化基于大语言模型的GUI代理的数据

In the previous section, we explored general frameworks for LLM-brained GUI agents, most of which rely on foundational LLMs such as GPT-4V and GPT-4o. However, to elevate these agents' performance and efficiency, optimizing their "brain", the underlying model is crucial. Achieving this often involves fine-tuning foundational models using large-scale, diverse, and high-quality contextual GUI datasets [515], which are specifically curated to enable these models to excel in GUI-specific tasks. Collecting such datasets, particularly those rich in GUI screenshots, metadata, and interactions, necessitates an elaborate process of data acquisition, filtering, and preprocessing, each requiring substantial effort and resources [516].
在上一节中，我们探讨了基于大语言模型（LLM）的GUI代理的一般框架，这些代理大多依赖于GPT-4V和GPT-4o等基础LLM。然而，为了提升这些代理的性能和效率，优化其“大脑”即底层模型至关重要。通常这需要利用大规模、多样且高质量的上下文GUI数据集[515]对基础模型进行微调，这些数据集专门策划以使模型在GUI特定任务中表现优异。收集此类数据集，尤其是包含丰富GUI截图、元数据和交互信息的数据集，需经过复杂的数据采集、筛选和预处理过程，每一步都需投入大量人力和资源[516]。

As GUI agents continue to gain traction, researchers have focused on assembling datasets that represent a broad spectrum of platforms and capture the diverse intricacies of GUI environments. These datasets are pivotal in training models that can generalize effectively, thanks to their coverage of varied interfaces, workflows, and user interactions. To ensure comprehensive representation, innovative methodologies have been employed to collect and structure these data assets. In the sections that follow, we detail an end-to-end pipeline for data collection and processing tailored to training GUI-specific LLMs. We also examine significant datasets from various platforms, providing insights into their unique features, the methodologies used in their creation, and their potential applications in advancing the field of LLM-brained GUI agents.
随着GUI代理的不断普及，研究者们致力于组建涵盖广泛平台并捕捉GUI环境多样细节的数据集。这些数据集对于训练能够有效泛化的模型至关重要，因为它们涵盖了多样的界面、工作流程和用户交互。为确保全面代表性，研究中采用了创新方法来收集和构建这些数据资产。接下来的章节中，我们将详细介绍针对GUI专用LLM训练的数据采集与处理端到端流程，并审视来自不同平台的重要数据集，解析其独特特征、构建方法及其在推动基于LLM的GUI代理领域发展中的潜力。

14.1 7.1 Data Collection

14.2 7.1 数据采集

Data is pivotal in training a purpose-built GUI agent, yet gathering it requires substantial time and effort due to the task's complexity and the varied environments involved.
数据在训练专用GUI代理中至关重要，但由于任务复杂且环境多样，数据采集过程需要大量时间和精力。

14.2.1 7.1.1 Data Composition and Sources

14.2.2 7.1.1 数据组成与来源

The essential data components for GUI agent training closely align with the agent's perception and inference requirements discussed in Sections 5.2.2 and 5.4. At a high level, this data comprises:
GUI代理训练所需的核心数据成分与第5.2.2节和5.4节中讨论的代理感知与推理需求高度契合。总体而言，这些数据包括：

1. User Instructions: These provide the task's overarching goal, purpose, and specific details, typically in natural language, offering a clear target for the agent to accomplish, e.g., "change the font size of all text to 12".
2. 用户指令：提供任务的总体目标、目的及具体细节，通常以自然语言形式呈现，为代理设定明确的完成目标，例如“将所有文本的字体大小改为12”。

2. Environment Perception: This typically includes GUI screenshots, often with various visual augmentations, as well as optional supplementary data like widget trees and UI element properties to enrich the context. This should encompass both the static

assessment of environment states (Section 5.2.2) and the dynamic environment feedback that captures post-action changes (Section 5.2.3), thereby providing sufficient contextual information.

3. 环境感知：通常包括GUI截图，常伴有各种视觉增强手段，以及可选的补充数据如控件树和UI元素属性以丰富上下文。这应涵盖环境状态的静态评估（第5.2.2节）和捕捉动作后变化的动态环境反馈（第5.2.3节），从而提供充分的上下文信息。

3. Task Trajectory: This contains the critical action sequence required to accomplish the task, along with supplementary information, such as the agent's plan. A trajectory usually involves multiple steps and actions to navigate through the task.
4. 任务轨迹：包含完成任务所需的关键动作序列及辅助信息，如代理的计划。轨迹通常涉及多步骤、多动作以完成任务。

While user instructions and environmental perception serve as the model's input, the expected model output is the task trajectory. This trajectory's action sequence is then grounded within the environment to complete the task.
用户指令和环境感知作为模型输入，而模型预期输出则是任务轨迹。该轨迹中的动作序列随后需在环境中落地执行以完成任务。

For user instructions, it is crucial to ensure that they are realistic and reflective of actual user scenarios. Instructions can be sourced in several ways: (i) directly from human designers, who can provide insights based on real-world applications; (ii) extracted from existing, relevant datasets if suitable data is available; (iii) sourcing from public materials, such as websites, application help documentation, and other publicly available resources; and (iv) generated by LLMs, which can simulate a broad range of user requests across different contexts. Additionally, LLMs can be employed for data augmentation [517], increasing both the quality and diversity of instructions derived from the original data.
对于用户指令，确保其真实且反映实际用户场景至关重要。指令可以通过多种方式获取：(i) 直接来自人类设计者，他们能基于真实应用提供见解；(ii) 从现有相关数据集中提取，前提是有合适的数据；(iii) 来源于公开材料，如网站、应用帮助文档及其他公开资源；(iv) 由大型语言模型（LLMs）生成，能够模拟不同语境下的广泛用户请求。此外，LLMs还可用于数据增强[517]，提升从原始数据中获得指令的质量和多样性。

For gathering environment perception data, various toolkits—such as those discussed in Section 5.2.2 can be used to capture the required GUI data. This can be done within an environment emulator (e.g., Android Studio Emulator [29] Selenium WebDriver [30] Windows Sandbox [31]) or by directly interfacing with a real environment to capture the state of GUI elements, including screenshots, widget trees, and other metadata essential for the agent's operation.
为了收集环境感知数据，可以使用多种工具包——如第5.2.2节中讨论的工具包——来捕获所需的GUI数据。这可以在环境模拟器中完成（例如，Android Studio模拟器[29] Selenium WebDriver[30] Windows Sandbox[31]），也可以通过直接与真实环境交互来捕获GUI元素的状态，包括截图、小部件树及其他对代理操作至关重要的元数据。

Collecting task trajectories, which represent the agent's action sequence to complete a task, is often the most challenging aspect. Task trajectories need to be accurate, executable, and well-validated. Collection methods include (i) using programmatically generated scripts, which define action sequences for predefined tasks, providing a highly controlled data source; (ii) employing human annotators, who complete tasks in a crowdsourced manner with each step recorded, allowing for rich, authentic action data; and (iii) leveraging model or agent bootstrapping [518], where an existing LLM or GUI agent attempts to complete the task and logs its actions, though this method may require additional validation due to potential inaccuracies. All these methods demand considerable effort, reflecting the complexities of gathering reliable, task-accurate data for training GUI agents.
收集任务轨迹，即代理完成任务的动作序列，通常是最具挑战性的部分。任务轨迹需准确、可执行且经过充分验证。收集方法包括：(i) 使用程序生成的脚本，定义预设任务的动作序列，提供高度可控的数据源；(ii) 雇佣人工标注者，以众包方式完成任务并记录每一步，获得丰富且真实的动作数据；(iii) 利用模型或代理自举[518]，即现有LLM或GUI代理尝试完成任务并记录动作，但此方法可能因潜在不准确性需额外验证。所有方法均需大量投入，反映了收集可靠且任务准确数据以训练GUI代理的复杂性。

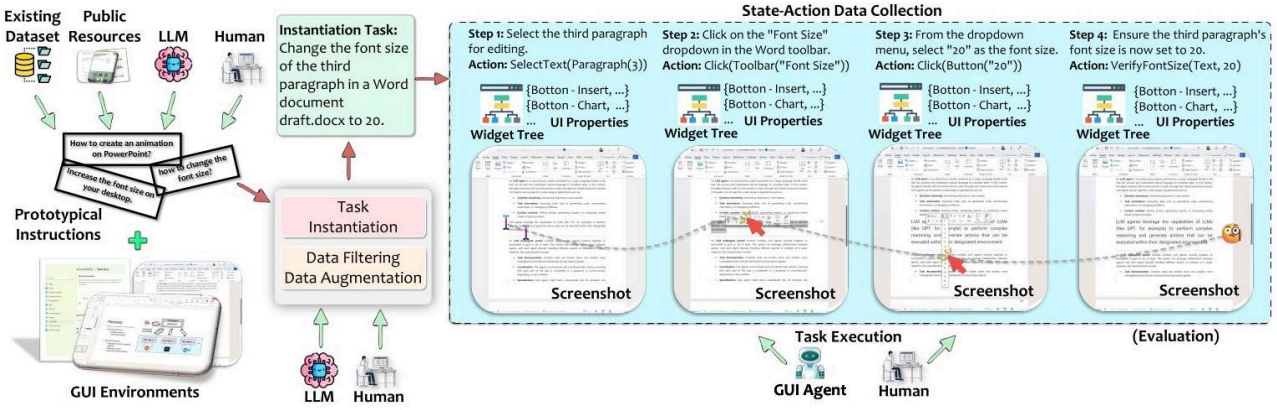

Fig. 25: A complete pipeline for data collection for training a GUI agent model.
图25：用于训练GUI代理模型的数据收集完整流程。

14.2.3 7.1.2 Collection Pipeline

14.2.4 7.1.2 收集流程

Figure 25 presents a complete pipeline for data collection aimed at training a GUI agent model. The process begins with gathering initial user instructions, which may come from various aforementioned sources. These instructions are typically prototypical, not yet tailored or grounded to a specific environment [374]. For instance, an instruction like "how to change the font size?" from a general website lacks specificity and doesn't align with the concrete requests a user might make within a particular application. To address this, an instantiation step is required [374], where instructions are contextualized within a specific environment, making them more actionable. For example, the instruction might be refined to "Change the font size of the third paragraph in a Word document of draft.docx to 20.", giving it a clear, environment-specific goal. This instantiation process can be conducted either manually by humans or programmatically with an LLM.
图25展示了用于训练GUI代理模型的数据收集完整流程。流程始于收集初始用户指令，指令可能来自上述多种来源。这些指令通常是原型性的，尚未针对特定环境进行定制或落地[374]。例如，来自通用网站的“如何更改字体大小？”指令缺乏具体性，且不符合用户在特定应用中的具体请求。为此，需要进行实例化步骤[374]，将指令置于特定环境中，使其更具可操作性。例如，指令可细化为“将draft.docx文档中第三段的字体大小改为20”，赋予明确的环境特定目标。该实例化过程可由人工手动完成，也可通过LLM程序化实现。

Following instantiation, instructions may undergo a filtering step to remove low-quality data, ensuring only relevant and actionable instructions remain. Additionally, data augmentation techniques can be applied to expand and diversify the dataset, improving robustness. Both of these processes can involve human validation or leverage LLMs for efficiency.
实例化后，指令可能经过过滤步骤以剔除低质量数据，确保仅保留相关且可执行的指令。此外，可应用数据增强技术以扩展和多样化数据集，提高鲁棒性。上述过程均可结合人工验证或利用LLM提升效率。

Once instruction refinement is complete, task trajectories and environment perceptions are collected simultaneously. As actions are performed within the environment, each step is logged, providing a record of the environment's state and the specific actions taken. After a full task trajectory is recorded, an evaluation phase is necessary to identify and remove any failed or inaccurate sequences, preserving the quality of the dataset. By iterating this pipeline, a high-quality dataset of GUI agent data can be compiled, which is crucial for training optimized models.
指令精炼完成后，任务轨迹和环境感知数据将同步收集。在环境中执行动作时，记录每一步，提供环境状态及具体动作的日志。完整任务轨迹记录后，需进行评估阶段，识别并剔除失败或不准确的序列，保证数据集质量。通过迭代该流程，可汇编高质量的GUI代理数据集，这对训练优化模型至关重要。

In the following sections, we review existing GUI agent datasets across various platforms, offering insights into current practices and potential areas for improvement.
在后续章节中，我们将回顾各平台现有的GUI代理数据集，提供当前实践的见解及潜在改进方向。

14.3 7.2 Web Agent Data

14.4 7.2 网络代理数据

Web-based GUI agents demand datasets that capture the intricate complexity and diversity of real-world web interactions. These datasets often encompass varied website structures, including DOM trees and HTML content, as well as multi-step task annotations that reflect realistic user navigation and interaction patterns. Developing agents that can generalize across different websites and perform complex tasks requires comprehensive datasets that provide rich contextual information. We provide an overview of web-based GUI agents dataset in Table 19
基于网络的GUI代理需要捕捉真实网络交互复杂性和多样性的数据集。这些数据集通常涵盖多样的网站结构，包括DOM树和HTML内容，以及反映真实用户导航和交互模式的多步骤任务注释。开发能够跨不同网站泛化并执行复杂任务的代理，需依赖提供丰富上下文信息的综合数据集。我们在表19中概述了基于网络的GUI代理数据集。

Building upon this need, several significant datasets have been developed to advance web-based GUI agents. Unlike traditional datasets focusing on narrow, predefined tasks, Mind2Web [212] represents a significant step forward by emphasizing open-ended task descriptions, pushing agents to interpret high-level goals independently. It offers over 2,350 human-annotated tasks across 137 diverse websites, capturing complex interaction patterns and sequences typical in web navigation. This setup aids in evaluating agents' generalization across unseen domains and serves as a benchmark for language grounding in web-based GUIs, enhancing adaptability for real-world applications.
基于此需求，多个重要数据集已被开发以推动基于网络的GUI代理发展。与传统聚焦于狭窄预定义任务的数据集不同，Mind2Web[212]代表了重要进展，强调开放式任务描述，推动代理独立理解高层目标。该数据集涵盖137个多样网站上的2350多个人工标注任务，捕捉了网络导航中典型的复杂交互模式和序列。此设置有助于评估代理在未见领域的泛化能力，并作为网络GUI语言落地的基准，提升其在实际应用中的适应性。

Similarly, WebVLN [347] expands on web GUI tasks by combining navigation with question-answering. It provides agents with text-based queries that guide them to locate relevant web pages and extract information. By leveraging both HTML and visual content from websites, WebVLN aligns with real-world challenges of web browsing. This dataset is particularly valuable for researchers aiming to develop agents capable of complex, human-like interactions in GUI-driven web spaces.
类似地，WebVLN [347] 通过将导航与问答结合，扩展了网页GUI任务。它为代理提供基于文本的查询，指导其定位相关网页并提取信息。

通过利用网站的HTML和视觉内容，WebVLN契合了网页浏览的现实挑战。该数据集对于旨在开发能够在GUI驱动的网页空间中进行复杂类人交互的代理的研究人员尤为宝贵。

Moreover, WebLINX 348 focuses on conversational GUI agents, particularly emphasizing real-world web navigation through multi-turn dialogue. Featuring over 2,300 expert demonstrations across 155 real-world websites, WebLINX creates a rich environment with DOM trees and screenshots for training and evaluating agents capable of dynamic, user-guided navigation tasks. This dataset promotes agent generalization across new sites and tasks, with comprehensive action and dialogue data that provide insights into enhancing agent responsiveness in realistic web-based scenarios.
此外，WebLINX 348 专注于对话式GUI代理，特别强调通过多轮对话实现现实网页导航。WebLINX包含来自155个真实网站的2300多次专家演示，构建了包含DOM树和截图的丰富环境，用于训练和评估能够执行动态用户引导导航任务的代理。该数据集促进代理在新网站和任务上的泛化，提供了全面的动作和对话数据，有助于提升代理在真实网页场景中的响应能力。

---

29. https://developer.android.com/studio
30. https://developer.android.com/studio

30. https://www.selenium.dev/
31. https://www.selenium.dev/

31. https://learn.microsoft.com/en-us/windows/security/ application-security/application-isolation/windows-sandbox/
32. https://learn.microsoft.com/en-us/windows/security/ application-security/application-isolation/windows-sandbox/

windows-sandbox-overview
windows-sandbox-overview

MultiUI [220] is a large-scale dataset designed to enhance GUI agents' text-rich visual understanding. It comprises 7.3 million multimodal instruction samples collected from 1 million websites, covering key web UI tasks such as element grounding, action prediction, and interaction modeling. Unlike traditional datasets that rely on raw HTML, MultiUI utilizes structured accessibility trees to generate high-quality multimodal instructions. Models trained on MultiUI demonstrate substantial performance improvements, achieving a 48% gain on VisualWebBench [213] and a 19.1% increase in element accuracy on Mind2Web [212].
MultiUI [220] 是一个大规模数据集，旨在增强GUI代理的文本丰富视觉理解能力。它包含从100万个网站收集的730万多条多模态指令样本，涵盖元素定位、动作预测和交互建模等关键网页UI任务。与依赖原始HTML的传统数据集不同，MultiUI利用结构化的辅助功能树生成高质量的多模态指令。在MultiUI上训练的模型表现显著提升，在VisualWebBench [213]上性能提升48%，在Mind2Web [212]上的元素准确率提升19.1%。

JOURNAL OF LATEX CLASS FILES, DECEMBER 2024

TABLE 19: Overview of datasets for optimizing LLMs tailored for web GUI agents.
表19：针对网页GUI代理优化大语言模型(LLM)的数据集概览。

Dataset	Platform	Source	Content	Scale	Collection Method	Highlight	Link
Mind2Web 212]	Web	Crowdsourced	Task descriptions, action sequences, webpage snapshots	2,350 from websites 137	Human demonstrations	Develops generalist web agents with diverse user interactions on real-world websites	https: //osu-nlp-group. github.io/ Mind2Web/
Mind2Web-Live 349]	Web	Sampled and reannotated from the Mind2Web 212	Textual descriptions, intermediate evaluation states. action sequences, and metadata, GUI screenshots	542 tasks, with 4,550 detailed annotation steps.	Annotated by human experts.	Emphasis on dynamic evaluation using “key nodes”, which represent critical intermediate states in web tasks.	https: //huggingface. co/datasets/ iMeanAI/ Mind2Web-Live
WebVLN 347]	Web	Human-designed, LLM-generated	Text instructions, plans, GUI screen-shots, HTML content	8,990 navigation paths, 14,825 QA pairs	WebVLN simulator, LLM-generated QA pairs	Vision-and-language navigation for human-like web browsing	https://github.com/WebVLN/ WebVLN
WebLINX 348]	Web	From human experts	Conversational interactions, action sequences, DOM and screenshots	2,337 demonstrations with over 100,000 interactions	Annotated by human experts	The first large-scale dataset designed to evaluate agents in real-world conversational web navigation	https: //mcgill-nlp. github.io/ weblinx/
AgentTrek 350]	Web	Web tutorials	Task metadata. step-by-step instructions, action sequences, visual observations, reproducible native traces	4,902 trajectories	VLM agent guided by tutorials, with Playwright capturing the traces	Synthesizes high-quality trajectory data by leveraging web tutorials	/
MultiUI 220]	Web	Combination of human-designed instructions and automated extraction from web structures	Textual task descriptions, plans, action sequences. GUI screenshots. accessibility trees, bounding box annotations	7.3 million instruction samples from 1 million websites	LLMs and Playwright	Supports a broad range of UI-related tasks, including GUI understanding, action prediction, and element grounding.	https: //neulab.github. io/MultiUI/
Explorer 290]	Web	Popular URLs with systematic web exploration by LLMs	Textual task descriptions. Action sequences, GUI screenshots, Accessibility trees, HTML content	94K successful web trajectories. 49K unique URLs, 720K screenshots	Generated by a multi-agent LLM pipeline	Largest-scale web trajectory dataset to date; dynamically explores web pages to create contextually relevant tasks	/

InSTA [351]	Web	Automatically generated by LLMs across 1M websites from Common Crawl	Web navigation tasks in natural language, task plans and action sequences, HTML-based observations converted to markdown, and evaluations from LLM-based judges	150,000 tasks across 150,000 websites	Generated by LLMs using the Playwright API and filtered by LLM-based judges	Presents fully automated three-stage data generation pipeline—task generation, action execution, and evaluation—using only language models without any human annotations	https:// data-for-agents github.io

数据集	平台	来源	内容	规模	收集方法	亮点	链接
Mind2Web 212]	网页	众包	任务描述、操作序列、网页快照	来自137个网站的2,350个样本	人工演示	开发具备多样用户交互能力的通用网页代理，应用于真实网站	https: //osu-nlp-group. github.io/ Mind2Web/
Mind2Web-Live 349]	网页	从Mind2Web 212中抽样并重新标注	文本描述、中间评估状态、操作序列及元数据，GUI截图	542个任务，包含4,550个详细标注步骤	由人工专家标注	强调使用“关键节点”进行动态评估，这些节点代表网页任务中的关键中间状态	https: //huggingface. co/datasets/ iMeanAI/ Mind2Web-Live
WebVLN 347]	网页	人工设计，LLM生成	文本指令、计划、GUI截图、HTML内容	8,990条导航路径，14,825对问答对	WebVLN模拟器，LLM生成的问答对	面向类人网页浏览的视觉与语言导航	https://github.com/WebVLN/ WebVLN
WebLINX 348]	网页	来自人工专家	对话交互、操作序列、DOM和截图	2,337次演示，超过100,000次交互	由人工专家标注	首个用于评估真实对话网页导航代理的大规模数据集	https: //mcgill-nlp. github.io/ weblinx/
AgentTrek 350]	网页	网页教程	任务元数据、逐步指令、操作序列、视觉观察、可复现的原生轨迹	4,902条轨迹	由教程指导的视觉语言模型代理，使用Playwright捕获轨迹	通过利用网页教程合成高质量轨迹数据	/
MultiUI 220]	网页	结合人工设计指令与网页结构自动提取	文本任务描述、计划、操作序列，GUI截图，辅助功能树，边界框标注	来自100万个网站的730万条指令样本	大型语言模型（LLMs）与Playwright	支持广泛的用户界面相关任务，包括GUI理解、操作预测和元素定位	https: //neulab.github. io/MultiUI/
Explorer 290]	网页	通过大型语言模型系统性探索的热门网址	文本任务描述、操作序列、GUI截图、辅助功能树、HTML内容	94K条成功网页轨迹，49K个独立网址，720K张截图	由多代理大型语言模型流水线生成	迄今最大规模的网页轨迹数据集；动态探索网页以创建上下文相关任务	/
InSTA [351]	网页	由大型语言模型自动生成，覆盖Common Crawl的100万个网站	自然语言网页导航任务、任务计划与操作序列，基于HTML的观察转换为markdown格式，以及基于大型语言模型评审的评估	覆盖150,000个任务和150,000个网站	由大型语言模型使用Playwright API生成，并由大型语言模型评审过滤	展示了完全自动化的三阶段数据生成流程——任务生成、操作执行和评估——仅使用语言模型，无需任何人工标注	https:// data-for-agents github.io

InSTA [351] is an Internet-scale dataset for training GUI-based web agents, generated entirely through an automated LLM pipeline without human annotations. It covers 150k diverse websites sourced from Common Crawl and includes rich web navigation tasks, trajectories in Playwright API calls, and evaluations using LLM-based judges. The dataset highlights strong generalization capabilities and data efficiency, significantly outperforming human-collected datasets like Mind2Web [212] and WebLINX 348 in zero-shot and low-resource settings. InSTA represents a key advancement in scalable data curation for LLM-powered GUI agents, offering unprecedented coverage across real-world web interfaces.

InSTA [351] 是一个用于训练基于GUI的网页代理的互联网规模数据集，完全通过自动化的大型语言模型（LLM）流水线生成，无需人工注释。它涵盖了来自Common Crawl的15万个多样化网站，包含丰富的网页导航任务、Playwright API调用轨迹以及基于LLM评审的评估。该

数据集展示了强大的泛化能力和数据效率，在零样本和低资源环境下显著优于人工收集的数据集如Mind2Web [212]和WebLINX 348。InSTA代表了面向LLM驱动GUI代理的可扩展数据策划的关键进展，提供了前所未有的真实网页界面覆盖。

Collectively, these datasets represent essential resources that enable advancements in web agent capabilities, supporting the development of adaptable and intelligent agents for diverse web applications.
这些数据集共同构成了推动网页代理能力进步的关键资源，支持开发适应性强且智能的代理，以应对多样化的网页应用。

14.5 7.3 Mobile Agent Data

14.6 7.3 移动代理数据

Mobile platforms are critical for GUI agents due to the diverse range of apps and unique user interactions they involve. To develop agents that can effectively navigate and interact with mobile interfaces, datasets must offer a mix of single and multi-step tasks, focusing on natural language instructions, UI layouts, and user interactions. We first overview mobile GUI agents dataset in Tables 20 and 21.
移动平台对GUI代理至关重要，因为其涉及多样的应用和独特的用户交互。为了开发能够有效导航和交互移动界面的代理，数据集必须包含单步和多步任务，重点关注自然语言指令、UI布局和用户交互。我们首先在表20和表21中概述移动GUI代理数据集。

An early and foundational contribution in this domain is the Rico dataset [355], which provides over 72,000 unique UI screens and 10,811 user interaction traces from more than 9,700 Android apps. Rico has been instrumental for tasks such as UI layout similarity, interaction modeling, and perceptual modeling, laying the groundwork for mobile interface understanding and GUI agent development.
该领域早期且基础性的贡献是Rico数据集[355]，提供了超过72,000个独特的UI屏幕和来自9,700多个安卓应用的10,811条用户交互轨迹。Rico在UI布局相似性、交互建模和感知建模等任务中发挥了重要作用，为移动界面理解和GUI代理开发奠定了基础。

Building upon the need for grounding natural language instructions to mobile UI actions, PIXELHELP [144] introduces a dataset specifically designed for this purpose. It includes multi-step instructions, screenshots, and structured UI element data, enabling detailed analysis of how verbal instructions can be converted into mobile actions. This dataset has significant applications in accessibility and task automation, supporting agents that autonomously execute tasks based on verbal cues.
基于将自然语言指令映射到移动UI操作的需求，PIXELHELP [144]引入了专门为此设计的数据集。它包含多步指令、截图和结构化UI元素数据，支持详细分析口头指令如何转化为移动操作。该数据集在无障碍和任务自动化方面具有重要应用，支持代理基于口头提示自主执行任务。

Further expanding the scope, the Android in the Wild (AITW) dataset [358] offers one of the most extensive collections of natural device interactions. Covering a broad spectrum of Android applications and diverse UI states, AITW captures multi-step tasks emulating real-world device usage. Collected through interactions with Android emulators, it includes both screenshots and action sequences, making it ideal for developing GUI agents that navigate app interfaces without relying on app-specific APIs. Due to its scale and diversity, AITW has become a widely used standard in the field.
进一步扩展范围的是Android in the Wild (AITW)数据集[358]，提供了最广泛的自然设备交互集合之一。涵盖了广泛的安卓应用和多样的UI状态，AITW捕捉了模拟真实设备使用的多步任务。通过安卓模拟器交互收集，包含截图和操作序列，非常适合开发无需依赖特定应用API即可导航应用界面的GUI代理。凭借其规模和多样性，AITW已成为该领域广泛使用的标准。

In addition, META-GUI 357 provides a unique dataset for mobile task-oriented dialogue systems by enabling direct interaction with mobile GUIs, bypassing the need for API-based controls. This approach allows agents to interact across various mobile applications using multi-turn dialogues and GUI traces, broadening their capabilities in real-world applications without custom API dependencies. The dataset's support for complex interactions and multi-turn dialogue scenarios makes it valuable for building robust conversational agents.
此外，META-GUI 357提供了一个独特的数据集，用于移动任务导向对话系统，通过直接与移动GUI交互，绕过API控制需求。这种方法允许代理通过多轮对话和GUI轨迹跨多种移动应用交互，拓展了其在真实应用中的能力，无需定制API依赖。该数据集支持复杂交互和多轮对话场景，对于构建稳健的对话代理极具价值。

Recently, MobileViews 364 emerged as the largest mobile screen dataset to date, offering over 600,000 screenshot-view hierarchy pairs from 20,000 Android apps. Collected with an LLM-enhanced app traversal tool, it provides a high-fidelity resource for mobile GUI agents in tasks such as screen summarization, tappability prediction, and UI component identification. Its scale and comprehensive coverage of screen states make MobileViews a key resource for advancing mobile GUI agent capabilities.
近期，MobileViews 364成为迄今为止最大的移动屏幕数据集，提供了来自20,000个安卓应用的超过60万对截图与视图层级。通过LLM增强的应用遍历工具收集，为移动GUI代理在屏幕摘要、可点击性预测和UI组件识别等任务中提供了高保真资源。其规模和全面的屏幕状态覆盖使MobileViews成为推动移动GUI代理能力提升的关键资源。

Collectively, mobile platforms currently boast the richest set of datasets due to their versatile tools, emulator support, and diverse use cases, reflecting the demand for high-quality, adaptive GUI agents in mobile applications.
总体而言，移动平台因其多样化工具、模拟器支持和丰富用例，拥有最丰富的数据集，反映了对高质量、适应性强的移动GUI代理的需求。

14.7 7.4 Computer Agent Data

14.8 7.4 计算机代理数据

In contrast to mobile and web platforms, the desktop domain for GUI agents has relatively fewer dedicated datasets, despite its critical importance for applications like productivity tools and enterprise software. However, notable efforts have been made to support the development and evaluation of agents designed for complex, multi-step desktop tasks. We show related dataset for computer GUI agents in Table 22
与移动和网页平台相比，桌面领域的GUI代理专用数据集相对较少，尽管其对生产力工具和企业软件等应用至关重要。然而，已有显著努力支持设计用于复杂多步桌面任务的代理的开发和评估。我们在表22中展示了相关的计算机GUI代理数据集。

A significant contribution in this area is ScreenA-gent [366], a dedicated dataset and model designed to facilitate GUI control in Linux and Windows desktop environments. ScreenAgent provides a comprehensive pipeline that enables agents to perform multi-step task execution autonomously, encompassing planning, action, and reflection phases. By leveraging annotated screenshots and detailed action sequences, it allows for high precision in UI element positioning and task completion, surpassing previous models in accuracy. This dataset is invaluable for researchers aiming to advance GUI agent capabilities in the desktop domain, enhancing agents' decision-making accuracy and user interface interactions.
该领域的重要贡献是ScreenAgent [366]，这是一个专门设计用于Linux和Windows桌面环境GUI控制的数据集和模型。ScreenAgent提供了一个完整的流水线，使代理能够自主执行多步任务，涵盖规划、执行和反思阶段。通过利用带注释的截图和详细的操作序列，实现了UI元素定位和任务完成的高精度，超越了以往模型的准确性。该数据集对旨在提升桌面领域GUI代理能力的研究人员极具价值，增强了代理的决策准确性和用户界面交互能力。

The LAM [367] is specifically designed to train and evaluate Large Action Models (LAMs) for GUI environments, bridging natural language task understanding and action execution. It comprises two core components: Task-Plan data, detailing user tasks with step-by-step plans, and Task-Action data, translating these plans into executable GUI actions. Sourced from application documentation, WikiHow articles, and Bing search queries, the dataset is enriched and structured using GPT-4. Targeting the Windows OS, with a focus on automating tasks in Microsoft Word, it includes 76,672 task-plan pairs and 2,688 task-action trajectories, making it one of the largest collections for GUI-based action learning. Data quality is ensured through a robust validation pipeline that combines LLM-based instantiation, GUI interaction testing, and manual review. Each entry is complemented with GUI screenshots and metadata, enabling models to learn both high-level task planning and low-level execution. The dataset's modular design supports fine-tuning for specific GUI tasks and serves as a replicable framework for building datasets in other environments, marking a significant contribution to advancing GUI-based automation.
LAM [367] 专门设计用于训练和评估图形用户界面（GUI）环境中的大型动作模型（Large Action Models，LAMs），实现自然语言任务理解与动作执行的桥接。它包含两个核心组成部分：任务-计划数据，详细描述用户任务及其逐步计划；任务-动作数据，将这些计划转化为可执行的GUI操作。数据来源包括应用文档、WikiHow文章和必应搜索查询，数据集通过GPT-4进行丰富和结构化。该数据集以Windows操作系统为目标，重点自动化Microsoft Word中的任务，包含76,672对任务-计划数据和2,688条任务-动作轨迹，是基于GUI动作学习的最大数据集之一。通过结合基于大型语言模型（LLM）的实例化、GUI交互测试和人工审核的严格验证流程，确保数据质量。每条数据均配有GUI截图和元数据，使模型能够学习高层次的任务规划和底层执行。数据集的模块化设计支持针对特定GUI任务的微调，并作为构建其他环境数据集的可复制框架，标志着GUI自动化领域的重要进展。

Although the desktop domain has fewer datasets compared to mobile and web, efforts like ScreenAgent and LAMs highlight the growing interest and potential for developing sophisticated GUI agents for computer systems.
尽管桌面领域的数据集数量少于移动和网页领域，但像ScreenAgent和LAMs这样的项目凸显了开发复杂计算机系统GUI代理的日益增长的兴趣和潜力。

14.9 7.5 Cross-Platform Agent Data

14.10 7.5 跨平台代理数据

Cross-platform datasets play a pivotal role in developing versatile GUI agents that can operate seamlessly across mobile, computer, and web environments. Such datasets support generalizability and adaptability, enabling agents to handle varied interfaces and tasks in real-world applications. We provide an overview of related dataset for cross-platform GUI agents in Table 23 and 24.
跨平台数据集在开发能够无缝运行于移动、计算机和网页环境的多功能GUI代理中起着关键作用。这类数据集支持泛化能力和适应性，使代理能够处理现实应用中多样的界面和任务。我们在表23和表24中提供了相关跨平台GUI代理数据集的概览。

One significant contribution is ScreenAI [375], which extends the scope of data collection to include both mobile and desktop interfaces. Covering tasks such as screen annotation, question-answering, and navigation, ScreenAI offers hundreds of millions of annotated samples. Its comprehensive scale and mixed-platform coverage make it a robust foundation for GUI agents that need to manage complex layouts and interactions across diverse interfaces. By emphasizing element recognition and screen summarization, ScreenAI advances the development of multi-platform GUI agents capable of handling varied visual structures.
一个重要贡献是ScreenAI [375]，其数据收集范围扩展至移动和桌面界面。涵盖屏幕注释、问答和导航等任务，ScreenAI提供了数亿条带注释

的样本。其全面的规模和多平台覆盖使其成为需要管理复杂布局和多样交互的GUI代理的坚实基础。通过强调元素识别和屏幕摘要，ScreenAI推动了能够处理多样视觉结构的多平台GUI代理的发展。

TABLE 20: Overview of datasets for optimizing LLMs tailored for mobile GUI agents (Part I).
表20：针对移动GUI代理优化大型语言模型（LLM）数据集概览（第一部分）。

Dataset	Platform	Source	Content	Scale	Collection Method	Highlight	Link
VGA [354]	Android Mobile	Rico 355	GUI screenshots, task descriptions, action sequences, bounds, layout, and functions of GUI elements	63.8k instances, 22.3k instruction-following data pairs, 41.4k conversation data pairs	Generated by GPT-4 models	Prioritizes visual content to reduce inaccuracies	https: //github.com/ Linziyang1999/ Vision% 2DGUI% 2Dassistant
Rico 355]	Android Mobile	Gathered from real Android apps on Google Play Store	Textual data, screenshots. action sequences, UI structure, annotated l representations	72,219 unique UI screens, 10,811 user interaction traces	automated exploration	Crowdsourcing, Comprehensive dataset for mobile UI design, interaction modeling, layout generation	http://www.interactionmining org/
PixelHelp 144]	Android Mobile	Human, web "How-to", Rico UI corpus synthetic	Natural language instructions, action sequences, GUI screenshots. structured UI data	187 multi-step instructions, 295,476 synthetic singlestep commands	Human annotation and synthetic generation	Pioneering method for grounding natural language instructions to executable mobile UI actions	https: //github.com/ google-research google-research tree/master/ seg2act
MoTIF 356]	Android Mobile	Human-written	Natural language instructions, action sequences, GUI screenshots. structured UI data	6,100 tasks across 125 Android apps	Human annotation	Task feasibility prediction for interactive GUI in mobile apps	https: //github.com/ aburns4/MoTIF
META-GUI 357]	Android Mobile	SMCalFlow 519	Dialogues, action sequences, screenshots. Android view hierarchies	1,125 dialogues and 4,684 turns	Human annotation	Task-oriented dialogue system for mobile GUI without relying on back-end APIs	https: //x-lance.github. io/META-GU
AITW 358	Android Mobile	Human-generated instructions. LLM-generated prompts	Natural language instructions, screenshots. observation-action pairs	715,142 episodes and 30,378 unique instructions	Human raters using Android emulators	Large-scale dataset for device control research with extensive app and UI diversity	https: //github.com/ google-research google-research tree/master/ android in the wild
GUI-Xplore 365]	Mobile Android	Combination of automated exploration and manual design	Exploration videos. textual tasks, QA pairs, view hierarchies, GUI screen-shots, action sequences, and GUI transition graphs	312 apps, 115 hours of video, 32,569 QA pairs, 41,293 actions, about 200 pages per app	Automated and human exploration	Introduces an exploration-based pretraining paradigm that provides rich app-specific priors through video data	https: //github.com/ 921112343/ GUI-Xplore

数据集	平台	来源	内容	规模	收集方法	亮点	链接
VGA [354]	安卓手机	Rico 355	GUI截图、任务描述、操作序列、边界、布局及GUI元素功能	63.8千个实例，22.3千条指令跟随数据对，41.4千条对话数据对	由GPT-4模型生成	优先考虑视觉内容以减少错误	https: //github.com/ Linziyang1999/ Vision% 2DGUI% 2Dassistant
Rico 355]	安卓手机	收集自Google Play商店的真实安卓应用	文本数据、截图、操作序列、UI结构、带注释的表示	72,219个独特UI界面，10,811条用户交互轨迹	自动化探索	众包，面向移动UI设计、交互建模、布局生成的综合数据集	http://www.interactionmining org/
PixelHelp 144]	安卓手机	人工，网页“操作指南”，Rico UI语料合成	自然语言指令、操作序列、GUI截图、结构化UI数据	187条多步骤指令，295,476条合成单步命令	人工标注与合成生成	将自然语言指令映射到可执行移动UI操作的开创性方法	https: //github.com/ google-research google-research tree/master/ seg2act
MoTIF 356]	安卓手机	人工编写	自然语言指令、操作序列、GUI截图、结构化UI数据	涵盖125个安卓应用的6,100个任务	人工标注	移动应用交互式GUI的任务可行性预测	https: //github.com/ aburns4/MoTIF
META-GUI 357]	安卓手机	SMCalFlow 519	对话、操作序列、截图、安卓视图层级	1,125个对话，4,684轮交互	人工标注	无需依赖后端API的移动GUI任务导向对话系统	https: //x-lance.github. io/META-GU
AITW 358	安卓手机	人工生成指令，LLM生成提示	自然语言指令、截图、观察-操作对	715,142个情节，30,378条独特指令	使用安卓模拟器的人工评分员	面向设备控制研究的大规模数据集，涵盖丰富的应用和UI多样性	https: //github.com/ google-research google-research tree/master/ android in the wild
GUI-Xplore 365]	移动安卓	自动探索与人工设计相结合	探索视频、文本任务、问答对、视图层级、GUI截图、操作序列及GUI转换图	312个应用，115小时视频，32,569个问答对，41,293个操作，约每个应用200页	自动与人工探索	引入基于探索的预训练范式，通过视频数据提供丰富的应用特定先验	https: //github.com/ 921112343/ GUI-Xplore

Building upon the need for evaluating visual foundation models across environments, VisualAgentBench [374] is a groundbreaking cross-platform benchmark designed to assess GUI agents in both mobile and web settings. It emphasizes interaction-focused tasks, using environments like Android Virtual Device and WebArena-Lite [412] to evaluate and improve agent responses to GUI layouts and user interface actions. The dataset's innovative collection method, which combines program-based solvers and large multimodal model bootstrapping, facilitates robust training trajectories that enhance adaptability and error recovery in GUI agent tasks.
基于对跨环境评估视觉基础模型的需求，VisualAgentBench [374] 是一个开创性的跨平台基准，旨在评估移动端和网页环境中的GUI代理。它强调以交互为核心的任务，利用Android虚拟设备和WebArena-Lite [412]等环境来评估和提升代理对GUI布局及用户界面操作的响应能力。该数据集创新性地结合了基于程序的求解器和大型多模态模型引导的收集方法，促进了稳健的训练路径，增强了GUI代理任务中的适应性和错误恢复能力。

Furthermore, GUI-World [371] spans multiple platforms, including desktop, mobile, and XR environments, with over 12,000 annotated videos. Designed to address the challenges of dynamic and sequential GUI tasks, GUI-World allows researchers to benchmark GUI agent capabilities across diverse interfaces. By providing detailed action sequences and QA pairs, it sets a high standard for evaluating agents in complex, real-world scenarios.
此外，GUI-World [371] 涵盖桌面、移动和XR环境，拥有超过12,000个带注释的视频。该数据集旨在应对动态和序列化GUI任务的挑战，使研究人员能够在多样化界面中对GUI代理能力进行基准测试。通过提供详细的操作序列和问答对，它为在复杂真实场景中评估代理设定了高标准。

Additionally, xLAM [372] contributes significantly to actionable agent development by providing a unified dataset format designed to support multi-turn interactions, reasoning, and function-calling tasks. Sourced from datasets like WebShop 425], ToolBench [520], and AgentBoard [521], xLAM standardizes data formats across diverse environments, addressing the common issue of inconsistent data structures that hinder agent training and cross-environment compatibility. By offering a consistent structure, xLAM enhances the adaptability and error detection capabilities of GUI agents, allowing for more seamless integration and performance across different applications.
另外，xLAM [372] 对可操作代理开发贡献显著，提供了统一的数据集格式，支持多轮交互、推理和函数调用任务。其数据来源包括WebShop [425]、ToolBench [520]和AgentBoard [521]等数据集，xLAM标准化了不同环境中的数据格式，解决了阻碍代理训练和跨环境兼容性的常见数据结构不一致问题。通过提供一致的结构，xLAM提升了GUI代理的适应性和错误检测能力，实现了不同应用间更顺畅的集成和性能表现。

OS-Genesis [369] adopts a reverse task synthesis approach for the Android and web platforms. It leverages GPT- 4o to interactively explore the environment and generate instructions in a reverse manner. This process constructs high-quality, diverse GUI trajectories without relying on human annotations or predefined tasks. By eliminating these dependencies, OS-Genesis achieves scalable and efficient training for GUI agents while significantly enhancing the diversity and quality of the generated data.
OS-Genesis [369] 采用逆向任务合成方法，针对Android和网页平台。它利用GPT-4o进行交互式环境探索并以逆向方式生成指令。该过程无需依赖人工标注或预定义任务，即可构建高质量、多样化的GUI轨迹。通过消除这些依赖，OS-Genesis 实现了GUI代理的可扩展高效训练，同时显著提升了生成数据的多样性和质量。

TABLE 21: Overview of datasets for optimizing LLMs tailored for mobile GUI agents (Part II).
表21：针对移动GUI代理优化大型语言模型(LLM)的数据集概览（第二部分）。

Dataset	Platform	Source	Content	Scale	Collection Method	Highlight	Link
GUI Odyssey 359	Android Mobile	Human designers, GPT-4	Textual tasks, plans, action sequences, GUI screenshots	7,735 episodes across 201 apps	Human demonstrations	Focuses on cross-app navigation tasks on mobile devices	https: //github.com/ OpenGVLab/ GUI-Odyssey
Amex 360	Android Mobile	Human-designed, ChatGPT-generated	Text tasks, action sequences, high-res screenshots with multi-level annotations	104,000 screen-shots, 1.6 million interactive elements, 2,946 instructions	Human annotations, autonomous scripts	Multi-level, large-scale annotations supporting complex mobile GUI tasks	https: //yuxiangchai. github.io/ AMEX/
Ferret-UI 352]	iOS, Android Mobile	Spotlight dataset, GPT-4	Text tasks, action plans, GUI element annotations, bounding boxes	40,000 elementary tasks, 10,000 advanced tasks	GPT-generated	Benchmark for UI-centric tasks with adjustable screen aspect ratios	https: //github.com/ apple/ml-ferret
AITZ [298]	Android Mobile	AITW 358	Screen-action pairs, action descriptions	18,643 screen-action pairs across 70+ apps, episodes	GPT- 4V, icon detection models	Structured "Chain-of-Action-Thought" enhancing GUI navigation	https: //github.com/ IMNearth/CoAT
Octo-planner [361]	Android Mobile	GPT-4 generated	Text tasks, decomposed plans, action sequences	1,000 points	GPT-4 generated	Optimized for task planning with GUI actions	https: //huggingface. co/NexaAIDev/ octopus-planning
E-ANT 362]	Android tiny-apps	Human behaviors	Task descriptions, screenshots. action sequences. page element data	40,000+ traces, 10,000 action intents	Human annotation	First large-scale Chinese dataset for GUI navigation with real human interactions	/
Mobile3M 353]	Android Mobile	Real-world interactions, simulations	UI screenshots. XML documents. action sequences	3,098,786 pages, 20,138,332 actions	Simulation algorithm	Large-scale Chinese mobile GUI dataset with unique navigation graph	https: //github.com/ Meituan-AutoMD MobileVLM
AndroidLab 363]	Android Mobile	Human design, LLM self-exploration. academic datasets	Text instructions, action sequences, XML data, screen-shots	10.5k traces. 94.3k steps	Human annotation, LLM self-exploration	XML-based interaction data with unified action space	https://github.com/THUDM/ Android-Lab
MobileViews 364]	Android Mobile	LLM-enhanced app traversal tool	Screenshot-view hierarchy pairs	600,000 screenshots. VH pairs from 20,000+ apps	LLM-enhanced crawler	Largest open-source mobile screen dataset	https: //huggingface. co/datasets/ mllmTeam/ MobileViews
FedMABench 452	Android Mobile	AndroidControl 515], AITW 358	Textual task descriptions. action sequences. and GUI screenshots	6 dataset series with over 30 subsets	Inferred from existing Android datasets	The first dataset designed to benchmark federated mobile GUI agents	https://github.com/wwh0411/ FedMABench

数据集	平台	来源	内容	规模	收集方法	亮点	链接
GUI Odyssey 359	安卓手机	人工设计者，GPT-4	文本任务、计划、操作序列、GUI截图	覆盖201个应用的7,735个情节	人工演示	聚焦移动设备上的跨应用导航任务	https: //github.com/ OpenGVLab/ GUI-Odyssey
Amex 360	安卓手机	人工设计，ChatGPT生成	文本任务、操作序列、高分辨率截图及多层注释	104,000张截图，160万个交互元素，2,946条指令	人工注释，自主脚本	支持复杂移动GUI任务的多层大规模注释	https: //yuxiangchai. github.io/ AMEX/
Ferret-UI 352]	iOS，安卓手机	Spotlight数据集，GPT-4	文本任务、操作计划、GUI元素注释、边界框	40,000个基础任务，10,000个高级任务	GPT生成	支持可调屏幕长宽比的UI中心任务基准	https: //github.com/ apple/ml-ferret
AITZ [298]	安卓手机	AITW 358	屏幕-操作对，操作描述	覆盖70+应用的18,643个屏幕-操作对，情节	GPT-4V，图标检测模型	结构化“行动思维链”(Chain-of-Action-Thought)增强GUI导航	https: //github.com/ IMNearth/CoAT
Octo-planner [361]	安卓手机	GPT-4生成	文本任务、分解计划、操作序列	1,000个点	GPT-4生成	针对带GUI操作的任务规划优化	https: //huggingface. co/NexaAIDev/ octopus-planning
E-ANT 362]	安卓小应用	人工行为	任务描述、截图、操作序列、页面元素数据	40,000+轨迹，10,000操作意图	人工注释	首个具备真实人类交互的中文大规模GUI导航数据集	/
Mobile3M 353]	安卓手机	真实交互，模拟	UI截图，XML文档，操作序列	3,098,786页，20,138,332次操作	模拟算法	具备独特导航图的大规模中文移动GUI数据集	https: //github.com/ Meituan-AutoMD MobileVLM
AndroidLab 363]	安卓手机	人工设计，LLM自我探索，学术数据集	文本指令、操作序列、XML数据、截图	10.5k轨迹，94.3k步骤	人工注释，LLM自我探索	基于XML的交互数据，统一操作空间	https://github.com/THUDM/ Android-Lab
MobileViews 364]	安卓手机	基于大语言模型（LLM）的应用遍历工具	截图与视图层级配对	60万张截图。来自2万多个应用的视图层级配对	基于大语言模型（LLM）的爬虫	最大规模开源移动屏幕数据集	https: //huggingface. co/datasets/ mllmTeam/ MobileViews
FedMABench 452	安卓手机	AndroidControl 515], AITW 358	文本任务描述、操作序列及图形用户界面（GUI）截图	6个数据集系列，包含30多个子集	基于现有Android数据集推断而来	首个用于评测联邦移动GUI代理的数据集	https://github.com/wwh0411/ FedMABench

Collectively, these cross-platform datasets contribute to building multi-platform GUI agents, paving the way for agents that can seamlessly navigate and perform tasks across different interfaces, fostering more generalized and adaptable systems.
这些跨平台数据集共同促进了多平台图形用户界面（GUI）代理的构建，为能够无缝导航并执行不同界面任务的代理铺平了道路，推动了更通用且适应性强的系统的发展。

14.11 7.6 Takeaways

14.12 7.6 关键要点

Data collection and curation for LLM-powered GUI agents is an intensive process, often requiring substantial human involvement, particularly for generating accurate action sequences and annotations. While early datasets were limited in scale and task diversity, recent advancements have led to large-scale, multi-platform datasets that support more complex and realistic GUI interactions. Key insights from these developments include:
用于大型语言模型（LLM）驱动的GUI代理的数据收集与整理是一个密集的过程，通常需要大量人工参与，尤其是在生成准确的操作序列和注释方面。尽管早期数据集在规模和任务多样性上有限，近期的进展已催生了支持更复杂且真实GUI交互的大规模多平台数据集。这些发展的主要见解包括：

1. Scale and Diversity: High-quality, large-scale data is essential for training robust GUI agents capable of handling diverse UI states and tasks. Datasets like MobileViews 364 and ScreenAI 375 illustrate the importance of vast and varied data to accommodate the dynamic nature of mobile and desktop applications, enhancing the agent's resilience across different environments.
2. 规模与多样性：高质量、大规模的数据对于训练能够处理多样化用户界面状态和任务的鲁棒GUI代理至关重要。诸如MobileViews 364和ScreenAI 375等数据集展示了庞大且多样化数据的重要性，以适应移动和桌面应用的动态特性，提升代理在不同环境中的适应能力。

2. Cross-Platform Flexibility: Cross-platform datasets such as VisualAgentBench 374 and GUI-World 371 underscore the value of

generalizability, enabling agents to perform consistently across mobile, web, and desktop environments. This cross-platform adaptability is a crucial step towards creating one-stop solutions where a single GUI agent can operate seamlessly across multiple platforms.

3. 跨平台灵活性：跨平台数据集如VisualAgentBench 374和GUI-World 371强调了泛化能力的价值，使代理能够在移动端、网页和桌面环境中保持一致的表现。这种跨平台适应性是实现一站式解决方案的关键步骤，使单一GUI代理能够无缝运行于多个平台。

3. Automated Data Collection: Al-driven data collection tools, as exemplified by OmniParser [184] and Mo-bileViews 364, showcase the potential to significantly reduce manual efforts and accelerate scalable dataset creation. By automating the annotation process, these tools pave the way for more efficient data pipelines, moving towards a future where AI supports AI by expediting data gathering and labeling for complex GUI interactions.
4. 自动化数据收集：以OmniParser [184]和MobileViews 364为代表的人工智能驱动数据收集工具展示了显著减少人工工作量并加速可扩展数据集创建的潜力。通过自动化注释过程，这些工具为更高效的数据管道铺路，迈向由AI辅助AI，加快复杂GUI交互数据采集与标注的未来。

TABLE 22: Overview of datasets for optimizing LLMs tailored for computer GUI agents.
表22：针对计算机GUI代理优化大型语言模型的数据集概览。

Dataset	Platform	Source	Content	Scale	Collection Method	Highlight	Link
ScreenAgent 366]	Linux, Windows OS	Human-designed	GUI screenshots, action sequences	273 task sessions. 3,005 training screenshots, 898 test screenshots	Human annotation	VLM-based agent across multiple desktop environments	https: //github.com/ niuzaisheng/ ScreenAgent
LAM 367]	Windows OS	Application documentation, WikiHow articles, Bing search queries	Task descriptions in natural language, step-by-step plans, action sequences, GUI screenshots	76,672 task-plan pairs, 2,192 task-action trajectories	Instantiated using GPT- 4, with actions tested and validated in the Windows environment using UFO 19	Provides structured pipeline for collecting validating, and augmenting data, enabling high-quality training for action-oriented AI models.	https://github.com/microsoft/ UFO/tree/main/ dataflow
DeskVision 368]	Windows macOS, and Linux desktops	Internet	GUI screenshots with annotated bounding boxes for UI elements and detailed region captions	54,855 screenshots with 303,622 UI element annotations	UI elements detected using OmniParser and PaddleOCR	The first large-scale, open-source dataset focusing on real-world desktop GUI scenarios across operating systems	/

数据集	平台	来源	内容	规模	收集方法	亮点	链接
ScreenAgent 366]	Linux、Windows 操作系统	人工设计	图形用户界面（GUI）截图，操作序列	273 个任务会话。3,005 张训练截图，898 张测试截图	人工标注	基于视觉语言模型（VLM）的代理，适用于多桌面环境	https: //github.com/ niuzaisheng/ ScreenAgent
LAM 367]	Windows 操作系统	应用文档、WikiHow 文章、必应搜索查询	自然语言任务描述、逐步计划、操作序列、GUI 截图	76,672 对任务-计划，2,192 条任务-操作轨迹	使用 GPT-4 实例化，操作在 Windows 环境中通过 UFO 19 测试和验证	提供结构化流程用于收集、验证和增强数据，实现面向操作的高质量 AI 模型训练。	https://github.com/microsoft/ UFO/tree/main/ dataflow
DeskVision 368]	Windows、macOS 和 Linux 桌面	互联网	带有 UI 元素标注边界框和详细区域说明的 GUI 截图	54,855 张截图，包含 303,622 个 UI 元素标注	使用 OmniParser 和 PaddleOCR 检测 UI 元素	首个聚焦跨操作系统真实桌面 GUI 场景的大规模开源数据集	/

TABLE 23: Overview of datasets for optimizing LLMs tailored for cross-platform GUI agents (Part I).
表23：用于优化针对跨平台图形用户界面代理的大型语言模型（LLM）数据集概览（第一部分）。

Dataset	Platform	Source	Content	Scale	Collection Method	Highlight	Link
Visual-AgentBench [374]	Android Mobile, Web	VAB-Mobile: Android Virtual Device, VAB-WebArena-Lite: WebArena 413]	Task instructions, action sequences, screen observations	VAB-Mobile: 1,213 trajectories, 10,175 steps; VAB-WebArena-Lite: 1,186 trajectories, 9,522 steps	Program-based solvers. agent bootstrapping, human demonstrations	Systematic evaluation of VLM as a visual foundation agent across multiple scenarios	https://github.com/THUDM/ VisualAgentBench
GUICourse 219]	Android Mobile, Web	Web scraping, simulation, manual design	GUI screenshots, action sequences, OCR tasks, QA pairs	10 million website page-annotation pairs, 67,000 action instructions	LLM-based auto-annotation, crowd-sourcing	Dataset suite for enhancing VLM GUI navigation on web and mobile platforms	https://github.com/yiye3/ GUICourse
GUI-World [371]	OS, Mobile, Web, XR	Student workers, YouTube instructional videos	GUI videos, human-annotated keyframes, captions, data, action sequences	12,000 videos, 83,176 frames	Human annotation	Designed dynamic, sequential GUI tasks with video data	https: //gui-world. github.io/
ScreenAl 375]	Android, iOS, Desktop/Web	Crawling apps and webpages, synthetic QA	Screen annotation, screen QA, navigation, summarization	Annotation: hundreds of millions; QA: tens of millions; Navigation: millions	Model, human annotation	Comprehensive pretraining and fine-tuning for GUI tasks across platforms	https://github.com/google% 2Dresearch% 2Ddatasets/ screen annotation
OmniParser 184]	Web, Desktop, Mobile	Popular webpages	UI screenshots, bounding boxes, icon descriptions, OCR-derived text	67,000+ screenshots. 7,000 icon-description pairs	Finetuned detection model, OCR, human descriptions	Vision-based parsing of UI screen-shots into structured elements	https://github.com/microsoft/ OmniParser

数据集	平台	来源	内容	规模	收集方法	亮点	链接
Visual-AgentBench [374]	安卓手机，网页	VAB-Mobile：安卓虚拟设备（Android Virtual Device），VAB-WebArena-Lite：WebArena 413]	任务指令，动作序列，屏幕观察	VAB-Mobile：1,213条轨迹，10,175步；VAB-WebArena-Lite：1,186条轨迹，9,522步	基于程序的求解器，代理引导，人类示范	作为视觉基础代理（visual foundation agent）在多场景下对视觉语言模型（VLM）的系统评估	https://github.com/THUDM/VisualAgentBench
GUICourse 219]	安卓手机，网页	网页爬取，模拟，手工设计	GUI截图，动作序列，OCR任务，问答对	1000万网页页面标注对，67,000条动作指令	基于大语言模型（LLM）的自动标注，众包	用于提升视觉语言模型（VLM）在网页和移动平台GUI导航能力的数据集套件	https://github.com/yiye3/GUICourse
GUI-World [371]	操作系统，移动端，网页，扩展现实（XR）	学生工，YouTube教学视频	GUI视频，人类标注关键帧，字幕，数据，动作序列	12,000个视频，83,176帧	人工标注	设计了动态的、连续的GUI任务并配有视频数据	https: //gui-world. github.io/
ScreenAl 375]	安卓，iOS，桌面/网页	爬取应用和网页，合成问答	屏幕标注，屏幕问答，导航，摘要	标注：数亿条；问答：数千万条；导航：数百万条	模型，人工标注	跨平台GUI任务的全面预训练与微调	https://github.com/google%2Dresearch% 2Ddatasets/ screen annotation
OmniParser 184]	网页，桌面，移动端	热门网页	UI截图，边界框，图标描述，OCR提取文本	67,000+截图，7,000个图标描述对	微调检测模型，OCR，人工描述	基于视觉的UI截图解析为结构化元素	https://github.com/microsoft/OmniParser

4. Unified Data Formats and Protocols: xLAM's unified data format is an essential innovation that improves compatibility across diverse platforms [372], addressing a significant bottleneck in cross-platform GUI agent development. Establishing standardized protocols or action spaces for data collection, particularly given the varied data formats, action spaces, and environment representations across platforms, will be vital in furthering agent generalization and consistency.
5. 统一数据格式与协议：xLAM的统一数据格式是一项重要创新，提升了不同平台间的兼容性[372]，解决了跨平台GUI代理开发中的关键瓶颈。鉴于各平台间数据格式、动作空间和环境表示的多样性，建立标准化的数据采集协议或动作空间对于推动代理的泛化能力和一致性至关重要。

In summary, the evolving landscape of datasets for LLM-powered GUI agents spans multiple platforms, with each dataset addressing unique challenges and requirements specific to its environment. These foundational resources are key to enabling agents to understand complex UIs, perform nuanced interactions, and improve generalization across diverse applications. The push towards cross-platform adaptability, automated data collection, and standardized data formats will continue to shape the future of GUI agents.
总之，面向大型语言模型(LLM)驱动的GUI代理的数据集不断发展，涵盖多个平台，每个数据集针对其环境的独特挑战和需求。这些基础资源是使代理理解复杂用户界面、执行细致交互并提升跨应用泛化能力的关键。推动跨平台适应性、自动化数据采集和标准化数据格式将持续塑造GUI代理的未来。

TABLE 24: Overview of datasets for optimizing LLMs tailored for cross-platform GUI agents (Part II).
表24：针对跨平台GUI代理优化大型语言模型(LLM)的数据集概览（第二部分）。

Dataset	Platform	Source	Content	Scale	Collection Method	Highlight	Link
Web-Hybrid 216]	Web, Android Mobile	Web-synthetic data	Screenshots, text-based referring expressions, coordinates on GUIs	10 million GUI elements, 1.3 million screenshots	Rule-based synthesis, LLMs for referring expressions	Largest dataset for GUI visual grounding	https: //osu-nlp-group. github.io/ UGround/
GUIDE 370]	Computer and Web	Direct submissions from businesses and survey responses	Task descriptions, GUI screenshots. action sequences, CoT reasoning, spatial grounding	N/A	Collected through NEXTAG, an automated annotation tool	Integrates images, action sequences, task descriptions, and spatial grounding into a unified dataset	https://github.com/superagi/GUIDE
xLAM 372]	Web and tools used	Synthesized data, and existing dataset	Textual tasks, action sequences, function-calling data	60,000 data points	Collected using AI models with human verification steps	Provides a unified format across diverse environments, enhancing generalizability and error detection for GUI agents	https: //github.com/ SalesforceAIResearch xLAM
Insight-UI 373]	iOS, Android, Windows, Linux, Web	Common Crawl corpus	Textual tasks, plans, action sequences, GUI screenshots	434,000 episodes, 1,456,000 images	Automatic simulations performed by a browser API	Instruction-free paradigm and entirely auto-generated	/
OS-Genesis 369]	Web and Android	Reverse tas synthesis, where the GUI environment is explored interactively without predefined tasks or human annotations.	High-level instructions, low-level instructions, action sequences, and environment states.	1,000 synthesized trajectories.	Model-based interaction-driven approach with GPT-40.	Reverses the conventional task-driven collection process by enabling exploration-first trajectory synthesis.	https: //giushisun. github.io/ OS-Genesis-Home/
Navi-plus 377]	Web and Android	AndroidControl 515] and Mind2Web 212]	Task descriptions, GUI action trajectories, low-level step instructions, screenshots, and followup ASK/SAY interaction pairs	/	LLM-automated with human validation	Introduces a Self-Correction GUI Navigation task featuring the novel ASK action for recovering missing information	/
Explorer 376]	Web and Android	Automated traversal of real websites and Android apps	UI screenshots, bounding boxes of interactable elements, screen similarity labels, and user actions	KhanAcademy (Web): 2,841 interactables, 378 screen similarity samples; Spotify (Android): 1,207 interactables, 451 screen similarity samples	Automated tools, HTML parsing, Accessibility Tree	Platform-independent, supports auto-labeling, and enables trace recording and voice-controlled GUI navigation	https://github.com/varnelis/Explorer

数据集	平台	来源	内容	规模	收集方法	亮点	链接
Web-混合 216]	网页，安卓移动端	网页合成数据	截图、基于文本的指代表达、GUI上的坐标	1000万个GUI元素，130万张截图	基于规则的合成，使用大型语言模型(LLMs)生成指代表达	最大的GUI视觉定位数据集	https: //osu-nlp-group. github.io/ UGround/
GUIDE 370]	计算机和网页	来自企业的直接提交和调查问卷回复	任务描述、GUI截图、动作序列、链式推理(CoT)、空间定位	不适用	通过NEXTAG自动标注工具收集	将图像、动作序列、任务描述和空间定位整合为统一数据集	https://github.com/superagi/GUIDE
xLAM 372]	网页及所用工具	合成数据及现有数据集	文本任务、动作序列、函数调用数据	6万个数据点	使用AI模型收集并辅以人工验证	提供跨多环境的统一格式，提升GUI代理的泛化能力和错误检测	https: //github.com/ SalesforceAIResearch xLAM
Insight-UI 373]	iOS、安卓、Windows、Linux、网页	Common Crawl语料库	文本任务、计划、动作序列、GUI截图	434,000个剧集，1,456,000张图片	由浏览器API自动模拟执行	无指令范式，完全自动生成	/
OS-Genesis 369]	网页和安卓	逆向任务合成，通过交互式探索GUI环境，无预设任务或人工标注	高级指令、低级指令、动作序列和环境状态	1000条合成轨迹	基于模型的交互驱动方法，使用GPT-4	颠覆传统任务驱动收集流程，实现先探索后合成轨迹	https: //giushisun. github.io/ OS-Genesis-Home/
Navi-plus 377]	网页和安卓	AndroidControl 515] 和 Mind2Web 212]	任务描述、GUI动作轨迹、低级步骤指令、截图及后续ASK/SAY交互对	/	大型语言模型自动生成，辅以人工验证	引入自我纠正GUI导航任务，新增ASK动作用于恢复缺失信息	/
Explorer 376]	网页和安卓	自动遍历真实网站和安卓应用	UI截图、可交互元素的边界框、屏幕相似度标签及用户操作	KhanAcademy（网页）：2841个可交互元素，378个屏幕相似度样本；Spotify（安卓）：1207个可交互元素，451个屏幕相似度样本	自动化工具、HTML解析、辅助功能树	平台无关，支持自动标注，支持轨迹记录和语音控制GUI导航	https://github.com/varnelis/Explorer

15 8 Models for Optimizing LLM-Brained GUI AGENTS

16 8 用于优化大型语言模型驱动GUI代理的模型

LLMs act as the "brain" of GUI agents, empowering them to interpret user intents, comprehend GUI screens, and execute actions that directly impact their environments. While several existing foundation models are robust enough to serve as this core, they can be further fine-tuned and optimized to evolve into Large Action Models (LAMs)-specialized models tailored to improve the performance and efficiency of GUI agents. These LAMs bridge the gap between general-purpose capabilities and the specific demands of GUI-based interactions.
大型语言模型（LLMs）作为GUI代理的“大脑”，赋能其解读用户意图、理解GUI界面并执行直接影响环境的操作。虽然现有多种基础模型已足够强大，可作为这一核心，但它们仍可通过微调和优化，演进为专门提升GUI代理性能与效率的大动作模型（Large Action Models，LAMs）。这些LAMs弥合了通用能力与基于GUI交互的特定需求之间的差距。

In this section, we first introduce the foundation models that currently form the backbone of GUI agents, highlighting their strengths and limitations. We then delve into the concept of LAMs, discussing how these models are fine-tuned with GUI-specific datasets to enhance their adaptability, accuracy, and action-orientation in GUI environments. Through this exploration, we illustrate the progression from general-purpose LLMs to purpose-built LAMs, laying the foundation for advanced, intelligent GUI agents.
本节首先介绍当前构成GUI代理骨干的基础模型，重点分析其优势与局限。随后深入探讨LAMs的概念，说明如何通过GUI特定数据集对这

些模型进行微调，以提升其在GUI环境中的适应性、准确性和动作导向性。通过此过程，展示从通用LLMs向专用LAMs的演进，为先进智能GUI代理奠定基础。

16.1 8.1 Foundation Models

16.2 8.1 基础模型

Foundation models serve as the core of LLM-powered GUI agents, providing the essential capabilities for understanding and interacting with graphical user interfaces. Recent advancements in both close-source and open-source MLLMs have significantly enhanced the potential of GUI agents, offering improvements in efficiency, scalability, and multimodal reasoning. This subsection explores these foundation models, highlighting their innovations, contributions, and suitability for GUI agent applications. For a quick reference, Table 25 presents an overview of the key models and their characteristics.
基础模型作为LLM驱动GUI代理的核心，提供理解和交互图形用户界面的基本能力。近来，闭源与开源多模态大型语言模型（MLLMs）的进展显著提升了GUI代理的潜力，在效率、可扩展性和多模态推理方面带来改进。本小节探讨这些基础模型，突出其创新、贡献及在GUI代理应用中的适用性。表25为关键模型及其特性提供了快速参考。

16.2.1 8.1.1 Close-Source Models

16.2.2 8.1.1 闭源模型

While proprietary models are not openly available for customization, they offer powerful capabilities that can be directly utilized as the "brain" of GUI agents.
尽管专有模型不开放定制，但其强大能力可直接用作GUI代理的“大脑”。

Among these, GPT-4V [378] and GPT-4o [93] are most commonly used in existing GUI agent frameworks due to their strong abilities, as discussed in Section 6. GPT- 4V represents a significant advancement in multimodal AI, combining text and image analysis to expand the functionality of traditional LLMs. Its ability to understand and generate responses based on both textual and visual inputs makes it well-suited for GUI agent tasks that require deep multimodal reasoning. Although its deployment is limited due to safety and ethical considerations, GPT-4V underscores the potential of foundation models to revolutionize GUI agent development with enhanced efficiency and flexibility.
其中，GPT-4V [378] 和 GPT-4o [93] 因其强大能力，在现有GUI代理框架中最为常用，如第6节所述。GPT-4V代表了多模态人工智能的重大进展，结合文本与图像分析，扩展了传统LLMs的功能。其基于文本和视觉输入理解并生成响应的能力，使其非常适合需要深度多模态推理的GUI代理任务。尽管因安全和伦理考虑部署受限，GPT-4V凸显了基础模型通过提升效率和灵活性，革新GUI代理开发的潜力。

Similarly, GPT-4o offers a unified multimodal autoregressive architecture capable of processing text, audio, images, and video. This model excels in generating diverse outputs efficiently, achieving faster response times at lower costs compared to its predecessors. Its rigorous safety and alignment practices make it reliable for sensitive tasks, positioning it as a robust tool for intelligent GUI agents that require comprehensive multimodal comprehension.
同样，GPT-4o提供统一的多模态自回归架构，能处理文本、音频、图像和视频。该模型在高效生成多样化输出方面表现出色，响应速度更快且成本更低于前代产品。其严格的安全和对齐措施使其在敏感任务中可靠，定位为需要全面多模态理解的智能GUI代理的强大工具。

The Gemini model family [92] advances multimodal AI modeling by offering versions tailored for high-complexity tasks, scalable performance, and on-device efficiency. Notably, the Nano models demonstrate significant capability in reasoning and coding tasks despite their small size, making them suitable for resource-constrained devices. Gemini's versatility and efficiency make it a compelling choice for powering GUI agents that require both performance and adaptability.
Gemini模型家族[92]推动多模态AI建模，提供针对高复杂度任务、可扩展性能及设备端效率的版本。尤其是Nano模型，尽管体积小巧，却在推理和编码任务中展现出显著能力，适合资源受限设备。Gemini的多样性和高效性使其成为需要兼顾性能与适应性的GUI代理的有力选择。

Emphasizing industry investment in GUI automation, Claude 3.5 Sonnet (Computer Use) introduces a pioneering approach by utilizing a vision-only paradigm for desktop task automation [163], [164]. It leverages real-time screenshots to observe the GUI state and generate actions, eliminating the need for metadata or underlying GUI structure. This model effectively automates GUI tasks by interpreting the screen, moving the cursor, clicking buttons, and typing text. Its unique architecture integrates a ReAct-based [252] reasoning paradigm with selective observation, reducing computational overhead by observing the environment only when necessary. Additionally, Claude 3.5 maintains a history of GUI screenshots, enhancing task adaptability and enabling dynamic interaction with software environments in a humanlike manner. Despite challenges in handling dynamic interfaces and error recovery, this model represents a significant step forward in creating general-purpose GUI agents. Its development highlights substantial industry investment in this area, indicating a growing focus on leveraging LLMs for advanced GUI automation.
强调行业对GUI自动化的投入，Claude 3.5 Sonnet（计算机使用）通过采用纯视觉范式实现桌面任务自动化，开创了新方法[163]，[164]。它利用实时截图观察GUI状态并生成操作，无需元数据或底层GUI结构。该模型通过解读屏幕、移动光标、点击按钮和输入文本，有效自动化GUI任务。其独特架构结合了基于ReAct[252]的推理范式与选择性观察，仅在必要时观察环境，降低计算开销。此外，Claude 3.5维护GUI截图历史，增强任务适应性，实现类人动态交互。尽管在处理动态界面和错误恢复方面存在挑战，该模型代表了通用GUI代理创建的重要进展。其开发体现了行业在该领域的重大投入，显示出利用LLMs推动高级GUI自动化的日益关注。

The Operator model [165], [513], developed by OpenAI, represents a new frontier in Computer-Using Agents (CUA), akin to Claude 3.5 Sonnet (Computer Use). Designed to interact with GUI environments through LLM-powered reasoning and vision capabilities, Operator builds upon GPT-40, integrating reinforcement learning to navigate and execute tasks across digital interfaces such as browsers, forms, and applications. By perceiving screenshots, interpreting UI elements, and performing actions via a virtual cursor and keyboard, Operator enables the automation of complex GUI-based workflows, including online purchases, email management, and document editing. Notably, Operator excels in understanding and manipulating digital environments, establishing itself as a powerful tool for human-computer interaction automation. Its exceptional performance on various benchmarks underscores its leading capabilities in GUI-based task automation.
Operator模型[165]，[513]由OpenAI开发，代表计算机使用代理（CUA）的新前沿，类似于Claude 3.5 Sonnet（计算机使用）。该模型设计用于通过LLM驱动的推理和视觉能力与GUI环境交互，基于GPT-4o，集成强化学习以导航并执行浏览器、表单和应用等数字界面上的任务。通过感知截图、解读UI元素并通过虚拟光标和键盘执行操作，Operator实现复杂GUI工作流的自动化，包括在线购物、邮件管理和文档编辑。值得注意的是，Operator在理解和操控数字环境方面表现卓越，确立了其作为人机交互自动化强大工具的地位。其在多项基准测试中的优异表现凸显了其在基于GUI任务自动化领域的领先能力。

16.2.3 8.1.2 Open-Source Models

16.2.4 8.1.2 开源模型

Open-source models provide flexibility for customization and optimization, allowing developers to tailor GUI agents with contextual data and deploy them on devices with limited resources.
开源模型提供了定制和优化的灵活性，使开发者能够结合上下文数据定制图形用户界面（GUI）代理，并将其部署在资源有限的设备上。

The Qwen-VL series [210] is notable for its fine-grained visual understanding and multimodal capabilities. With a Vision Transformer-based visual encoder and the Qwen-7B language model [511], it achieves state-of-the-art results on vision-language benchmarks while supporting multilingual interactions. Its efficiency and open-source availability, along with quantized versions for resource efficiency, make it suitable for developing GUI agents that require precise visual comprehension.
Qwen-VL系列[210]以其细粒度视觉理解和多模态能力著称。该系列采用基于视觉变换器（Vision Transformer）的视觉编码器和Qwen-7B语言模型[511]，在视觉-语言基准测试中取得了最先进的成果，同时支持多语言交互。其高效性和开源特性，以及为资源效率设计的量化版本，使其适合开发需要精确视觉理解的GUI代理。

Building upon this, Qwen2-VL [231] introduces innovations like Naive Dynamic Resolution and Multimodal Rotary Position Embedding, enabling efficient processing of diverse modalities including extended-length videos. The scalable versions of Qwen2-VL balance computational efficiency and performance, making them adaptable for both on-device applications and complex multimodal tasks in GUI environments.
在此基础上，Qwen2-VL[231]引入了朴素动态分辨率（Naive Dynamic Resolution）和多模态旋转位置编码（Multimodal Rotary Position Embedding）等创新，能够高效处理包括长视频在内的多样模态。Qwen2-VL的可扩展版本在计算效率和性能之间取得平衡，适用于设备端应用及GUI环境中的复杂多模态任务。

InternVL-2 [379], [380] combines a Vision Transformer with a Large Language Model to handle text, images, video, and medical data inputs. Its progressive alignment strategy and availability in various sizes allow for flexibility in deployment. By achieving state-of-the-art performance in complex multimodal tasks, InternVL-2 demonstrates powerful capabilities that are valuable for GUI agents requiring comprehensive multimodal understanding.
InternVL-2[379]，[380]结合了视觉变换器和大型语言模型，能够处理文本、图像、视频及医疗数据输入。其渐进式对齐策略和多种尺寸版本提供了部署灵活性。通过在复杂多模态任务中实现最先进性能，InternVL-2展现了强大的能力，适用于需要全面多模态理解的GUI代理。

Advancing efficient integration of visual and linguistic information, CogVLM [381] excels in cross-modal tasks with a relatively small number of trainable parameters. Its ability to deeply integrate visual and language features while preserving the full capabilities of large language models makes it a cornerstone for GUI agent development, especially in applications where resource efficiency is critical.
CogVLM[381]在视觉与语言信息的高效整合方面取得突破，凭借较少的可训练参数在跨模态任务中表现出色。其能够深度融合视觉与语言特征，同时保留大型语言模型的全部能力，使其成为GUI代理开发的基石，尤其适用于资源效率至关重要的应用场景。

TABLE 25: Overview of foundation models for LLM-brained GUI agents.
表25：基于大型语言模型（LLM）驱动的GUI代理基础模型概览。

Model	Modality	Model Size	Architecture	Training Methods	Highlights	Open-Source	Link
	Text, audio, image, and video	-	Multimodal autoregressive architecture	Pre-trained on a mix of public data, further trained for align ment with human preferences and safety considerations	Unified multimodal architecture that seamlessly processes and generates outputs across text audio, image, and video, offering faster and more cost-effective operation than its predecessors	No	/
GPT-4V 378]	Text and image	-	-	Pre-trained on a large dataset of text and image data, followed by fine-tuning with reinforcement learning from human feedback (RLHF)	Notable for its multimodal ca pabilities, allowing it to analyze and understand images along side text	No	/
Gemini 9	Text, image, audio, and video	Nano versions: 1.8B/3.25B	Enhanced Transformer decoders	Large-scale pre-training on multimodal data, followed by supervised fine-tuning, reward modeling, and RLHF	Achieves state-of-the-art performance across multimodal tasks. including a groundbreaking 90% on the MMLU benchmark, and demonstrates capacity for on-device deployment with small model sizes	No	/
Claude 3.5 Sonnet (Computer Use) 163. 164]	Text and image	-	ReAct-based reasoning	-	Pioneering role in GUI automation as the first public beta model to utilize a vision-only paradigm for desktop task automation	No	/
Operator [165], [513]	Text and Image	-	Built on GPT-4o	Supervised learning and reinforcement learning	Trained to use a computer like a human, achieving remarkable performance on benchmarks	No	/
Qwen-VL 210]	Text and image	9.6B	A Vision Transformer (ViT) 522] as the visual encoder, with a large language model based on the Qwen-7B architecture	Two stages of pre-training and a final stage of instruction fine-tuning	Achieves state-of-the-art per formance on vision-language benchmarks and supports fine-grained visual understanding	Yes	httpss://github.com/QwenLM/Qwen-VL
Qwen2-VL 231]	Text, image, and video	2B/7B/72B	ViT [522] as the vision encoder, paired with the Qwen2 series of language models	The ViT is trained with image text pairs; all parameters are unfrozen for broader multimodal learning with various datasets; fine-tuning the LLM on instruction datasets	Introduces Naive Dynamic Res olution for variable resolution image processing and Multimodal Rotary Position Embedding for enhanced multimodal integration	Yes	httpss://github.com/QwenLM/Qwen2-VL
InternVL-2 379], [380]	Text, image, video, and medical data	1B/2B/4B/ 8B/26B/40B	ViT as the vision encoder and a LLM as the language component	Progressive alignment strategy starting with coarse data and moving to fine data	Demonstrates powerful capabilities in handling complex multimodal tasks with various model sizes	Yes	httpss://internvl.github.io/blog/2024-07-02-InternVL-2 0/

CoqVLM 381]	Text and image	17B	A ViT encoder, a two-layer MLP adapter, a pre-trained large language model, and a visual expert module	Stage 1 focuses on image captioning; Stage 2 combines image captioning and referring expression comprehension tasks	Achieves deep integration of visual and language features while preserving the full capabilities of large language models	Yes	httpss://github.com/THUDM/CogVLM
	Text and image	7B/13B	Decoder-only architecture based on the Vicuna model, combined with a visual encoder	A combination of supervised training and additional instruction tuning	Ability to handle free-form region inputs via its hybrid region representation, enabling versatile spa tial understanding and grounding	Yes	httpss: //github.com/apple/ml-ferret
	Text and image	7B/13B	A vision encoder (CLIP ViT-L/14), a language decoder (Vicuna)	Pre-training using filtered image-text pairs, fine-tuning with a multimodal instruction-following dataset	Its lightweight architecture enables quick experimentation demonstrating capabilities close to GPT-4 in multimodal reason ing	Yes	httpss://llava-vl.github.io
LLaVA-1.5 383]	Text and image	7B/13B	A vision encoder (CLIP-ViT) and an encoder-decoder LLM architecture (e.g., Vicuna or LLaMA)	Pre-training on vision-language alignment with image-text pairs; visual instruction tuning with specific task-oriented data	Notable for its data efficiency and scaling to high-resolution image inputs	Yes	httpss://llava-vl github.io
BLIP-2 206]	Text and image	3.4B/12.1B	A frozen image encoder. a lightweight Querying Transformer to bridge the modality gap, and a frozen large language model	Vision-language representation learning: trains the Q-Former with a frozen image encoder; Vision-to-language generative learning: connects the Q-Former to a frozen LLM to enable image-to-text generation	Achieves state-of-the-art performance on various vision-language tasks with a compute efficient strategy by leveraging frozen pre-trained models	Yes	httpss://github.com/salesforce LAVIS/tree main/projects/ blip2
Phi-3.5-Vision 234]	Text and image	4.2B	Image encoder: CLIP ViT-L/14 to process visual inputs, and transformer decoder based on the Phi-3.5 mini model for textual outputs	Pre-training on a combination of interleaved image-text datasets synthetic OCR data, chart/table comprehension data, and text-only data; supervised fine-tuning using large-scale multimodal and text datasets; Direct Preference Optimization (DPO) to improve alignment, safety, and multimodal task performance	Excels in reasoning over visual and textual inputs, demonstrat ing competitive performance on single-image and multi-image tasks while being compact	Yes	httpss://github.com/microsoft/ Phi-3CookBook tree/main

模型	模态	模型规模	架构	训练方法	亮点	开源	链接
	文本、音频、图像和视频	-	多模态自回归架构	在混合公共数据上进行预训练，随后针对与人类偏好和安全考虑的对齐进行进一步训练	统一的多模态架构，能够无缝处理并生成文本、音频、图像和视频的输出，运行速度更快且成本更低于其前代模型	否	/
GPT-4V 378]	文本和图像	-	-	在大量文本和图像数据集上预训练，随后通过人类反馈强化学习（RLHF）进行微调	以其多模态能力著称，能够同时分析和理解图像与文本	否	/
Gemini 9	文本、图像、音频和视频	Nano版本：1.8B/3.25B	增强型Transformer解码器	在多模态数据上进行大规模预训练，随后进行监督微调、奖励建模和RLHF	在多模态任务中实现了最先进的性能，包括在MMLU基准测试中取得突破性的90%，并展示了小模型尺寸下的设备端部署能力	否	/
Claude 3.5 Sonnet（计算机使用）163. 164]	文本和图像	-	基于ReAct的推理	-	作为首个采用纯视觉范式进行桌面任务自动化的公开测试模型，在GUI自动化领域具有开创性作用	否	/
Operator [165], [513]	文本和图像	-	基于GPT-4o构建	监督学习和强化学习	训练其像人类一样使用计算机，在基准测试中表现出色	否	/
Qwen-VL 210]	文本和图像	9.6B	采用视觉Transformer（ViT）522]作为视觉编码器，基于Qwen-7B架构的大型语言模型	两阶段预训练及最终的指令微调阶段	在视觉-语言基准测试中达到最先进性能，支持细粒度视觉理解	是	httpss://github.com/QwenLM/Qwen-VL
Qwen2-VL 231]	文本、图像和视频	2B/7B/72B	ViT [522]作为视觉编码器，配合Qwen2系列语言模型	ViT使用图文对进行训练；所有参数均解冻以支持多样数据集的广泛多模态学习；对大型语言模型进行指令数据集微调	引入了用于可变分辨率图像处理的朴素动态分辨率（Naive Dynamic Resolution）和用于增强多模态融合的多模态旋转位置编码（Multimodal Rotary Position Embedding）	是	httpss://github.com/QwenLM/Qwen2-VL
InternVL-2 379], [380]	文本、图像、视频及医疗数据	1B/2B/4B/8B/26B/40B	ViT作为视觉编码器，LLM作为语言组件	采用渐进式对齐策略，从粗略数据开始，逐步过渡到精细数据	展示了在处理复杂多模态任务时，具备多种模型规模的强大能力	是	httpss://internvl.github.io/blog/2024-07-02-InternVL-20/

CoqVLM 381]	文本和图像	17B	包含ViT编码器、两层MLP适配器、预训练大型语言模型及视觉专家模块	第一阶段专注于图像描述；第二阶段结合图像描述与指代表达理解任务	实现了视觉与语言特征的深度融合，同时保留了大型语言模型的全部能力	是	httpss://github.com/THUDM/CogVLM
	文本和图像	7B/13B	基于Vicuna模型的仅解码器架构，结合视觉编码器	结合监督训练和额外的指令微调	通过其混合区域表示能力处理自由形式的区域输入，实现多样的空间理解与定位	是	httpss://github.com/apple/ml-ferret
	文本和图像	7B/13B	视觉编码器（CLIP ViT-L/14）、语言解码器（Vicuna）	使用过滤后的图文对进行预训练，利用多模态指令跟随数据集进行微调	其轻量级架构支持快速实验，展现出接近GPT-4多模态推理能力	是	httpss://llava-vl. github.io
LLaVA-1.5 383]	文本和图像	7B/13B	一个视觉编码器（CLIP-ViT）和一个编码器-解码器大型语言模型架构（例如，Vicuna 或 LLaMA）	基于图文对进行视觉-语言对齐的预训练；使用特定任务导向数据进行视觉指令微调	以数据效率高和支持高分辨率图像输入而著称	是	httpss://llava-vl github.io
BLIP-2 206]	文本和图像	3.4B/12.1B	一个冻结的图像编码器，一个轻量级查询变换器（Querying Transformer）用于桥接模态差异，以及一个冻结的大型语言模型	视觉-语言表示学习：使用冻结的图像编码器训练Q-Former；视觉到语言生成学习：将Q-Former连接到冻结的LLM，实现图像到文本的生成	通过利用冻结的预训练模型，以计算效率高的策略在多种视觉-语言任务上实现了最先进的性能	是	httpss://github. com/salesforce LAVIS/tree main/projects/ blip2
Phi-3.5-Vision 234]	文本和图像	4.2B	图像编码器：使用CLIP ViT-L/14处理视觉输入，基于Phi-3.5迷你模型的变换器解码器生成文本输出	在交错的图文数据集、合成OCR数据、图表/表格理解数据和纯文本数据的组合上进行预训练；使用大规模多模态和文本数据集进行监督微调；采用直接偏好优化（Direct Preference Optimization, DPO）提升对齐、安全性和多模态任务性能	在视觉和文本输入的推理方面表现出色，在单图像和多图像任务中展现出竞争力的性能，同时模型结构紧凑	是	httpss://github. com/microsoft/ Phi-3CookBook tree/main

Enhancing spatial understanding and grounding, Ferret 382 offers an innovative approach tailored for GUI agents. By unifying referring and grounding tasks within a single framework and employing a hybrid region representation, it provides precise interaction with graphical interfaces. Its robustness against object hallucinations and efficient architecture make it ideal for on-device deployment in real-time GUI applications.
为了增强空间理解和定位能力，Ferret 382 提供了一种针对图形用户界面（GUI）代理的创新方法。通过在单一框架内统一指代和定位任务，并采用混合区域表示，它实现了与图形界面的精确交互。其对对象幻觉的鲁棒性和高效架构使其非常适合在实时GUI应用中进行设备端部署。

The LLaVA model [217] integrates a visual encoder with a language decoder, facilitating efficient alignment between modalities. Its lightweight projection layer and modular design enable quick experimentation and adaptation, making it suitable for GUI agents that require fast development cycles and strong multimodal reasoning abilities. Building on this, LLaVA-1.5 383 introduces a novel MLP-based cross-modal connector and scales to high-resolution image inputs, achieving impressive performance with minimal training data. Its data efficiency and open-source availability pave the way for widespread use in GUI applications requiring detailed visual reasoning.
LLaVA模型[217]集成了视觉编码器与语言解码器，促进了多模态之间的高效对齐。其轻量级投影层和模块化设计支持快速实验和适应，适合需要快速开发周期和强大多模态推理能力的GUI代理。在此基础上，LLaVA-1.5 383引入了新颖的基于多层感知机（MLP）的跨模态连接

器，并扩展到高分辨率图像输入，凭借极少的训练数据实现了出色性能。其数据效率和开源特性为需要细致视觉推理的GUI应用的广泛使用铺平了道路。

BLIP-2 [206] employs a compute-efficient strategy by leveraging frozen pre-trained models and introducing a lightweight Querying Transformer. This design allows for state-of-the-art performance on vision-language tasks with fewer trainable parameters. BLIP-2's modularity and efficiency make it suitable for resource-constrained environments, highlighting its potential for on-device GUI agents.
BLIP-2 [206]采用了计算高效策略，通过利用冻结的预训练模型并引入轻量级查询变换器（Querying Transformer）。该设计使其在视觉-语言任务上以更少的可训练参数实现了最先进的性能。BLIP-2的模块化和高效性使其适合资源受限环境，凸显了其在设备端GUI代理中的潜力。

Finally, Phi-3.5-Vision 234 achieves competitive performance in multimodal reasoning within a compact model size. Its innovative training methodology and efficient integration of image and text understanding make it a robust candidate for GUI agents that require multimodal reasoning and on-device inference without the computational overhead of larger models.
最后，Phi-3.5-Vision 234在紧凑模型尺寸下实现了多模态推理的竞争性能。其创新的训练方法和图像与文本理解的高效整合，使其成为需要多模态推理和设备端推断且避免大型模型计算负担的GUI代理的有力候选。

In summary, both close-source and open-source foundation models have significantly advanced the capabilities of LLM-powered GUI agents. While proprietary models offer powerful out-of-the-box performance, open-source models provide flexibility for customization and optimization, enabling tailored solutions for diverse GUI agent applications. The innovations in multimodal reasoning, efficiency, and scalability across these models highlight the evolving landscape of foundation models, paving the way for more intelligent and accessible GUI agents.
总之，闭源和开源基础模型均显著提升了基于大型语言模型（LLM）的GUI代理能力。虽然专有模型提供了强大的开箱即用性能，开源模型则为定制和优化提供了灵活性，使得针对多样化GUI代理应用的解决方案成为可能。这些模型在多模态推理、效率和可扩展性方面的创新，彰显了基础模型不断演进的格局，为更智能、更易用的GUI代理铺平了道路。

16.3 8.2 Large Action Models

16.4 8.2 大动作模型

While general-purpose foundation LLMs excel in capabilities like multimodal understanding, task planning, and tool utilization, they often lack the specialized optimizations required for GUI-oriented tasks. To address this, researchers have introduced Large Action Models (LAMs)—foundation LLMs fine-tuned with contextual, GUI-specific datasets (as outlined in Section 7) to enhance their action-driven capabilities. These models represent a significant step forward in refining the "brain" of GUI agents for superior performance.
尽管通用基础大型语言模型（LLM）在多模态理解、任务规划和工具使用等能力上表现出色，但它们往往缺乏针对GUI任务所需的专门优化。为此，研究人员提出了大动作模型（Large Action Models，LAMs）——基于上下文的GUI特定数据集（详见第7节）微调的基础LLM，以增强其动作驱动能力。这些模型代表了提升GUI代理“大脑”性能的重要进展。

In the realm of GUI agents, LAMs provide several transformative advantages:
在GUI代理领域，LAMs带来了若干变革性优势：

1. Enhanced Action Orientation: By specializing in action-oriented tasks, LAMs enable accurate interpretation of user intentions and generation of precise action sequences. This fine-tuning ensures that LAMs can seamlessly align their outputs with GUI operations, delivering actionable steps tailored to user requests.
2. 增强的动作导向性：通过专注于动作导向任务，LAMs能够准确解读用户意图并生成精确的动作序列。此微调确保LAMs能无缝对齐其输出与GUI操作，提供针对用户请求量身定制的可执行步骤。

2. Specialized Planning for Long, Complex Tasks: LAMs excel in devising and executing intricate, multi-step workflows. Whether the tasks span multiple applications or involve interdependent operations, LAMs leverage their training on extensive action sequence datasets to create coherent, long-term plans. This makes them ideal for productivity-focused tasks requiring sophisticated planning across various tools.
3. 针对长且复杂任务的专门规划：LAMs擅长设计和执行复杂的多步骤工作流。无论任务跨越多个应用还是涉及相互依赖的操作，LAMs都能利用其在大规模动作序列数据集上的训练，制定连贯的长期计划。这使其非常适合需要跨多工具进行复杂规划的生产力任务。

3. Improved GUI Comprehension and Visual Grounding: Training on datasets that incorporate GUI screenshots allows LAMs to advance their abilities in detecting, localizing, and interpreting UI components such as buttons, menus, and forms. By utilizing visual cues instead of relying solely on structured UI metadata, LAMs become highly adaptable, performing effectively across diverse software environments.
4. 改进的GUI理解与视觉定位：通过训练包含GUI截图的数据集，LAMs提升了检测、定位和解读UI组件（如按钮、菜单和表单）的能力。利用视觉线索而非仅依赖结构化UI元数据，LAMs具备高度适应性，能在多样化软件环境中高效运行。

4. Efficiency through Model Size Reduction: Many LAMs are built on smaller foundational models—typically around 7 billion parameters—that are optimized for GUI-specific tasks. This compact, purpose-driven design reduces computational overhead, enabling efficient operation even in resource-constrained environments, such as on-device inference.

5. 通过模型规模缩减实现效率：许多LAMs基于较小的基础模型——通常约70亿参数——针对GUI特定任务进行了优化。这种紧凑且目标明确的设计降低了计算开销，使其即使在资源受限环境（如设备端推断）中也能高效运行。

As illustrated in Figure 26, the process of developing a purpose-built LAM for GUI agents begins with a robust, general-purpose foundation model, ideally with VLM capabilities. Fine-tuning these models on comprehensive, specialized GUI datasets—including user instructions, widget trees, UI properties, action sequences, and annotated screen-shots-transforms them into optimized LAMs, effectively equipping them to serve as the "brain" of GUI agents.
如图26所示，针对GUI代理开发专用LAM的过程始于一个强大的通用基础模型，理想情况下具备视觉语言模型（VLM）能力。通过在包含用户指令、小部件树、UI属性、动作序列和带注释截图的全面专业GUI数据集上微调，这些模型被转化为优化的LAMs，有效装备其作为GUI代理“大脑”。

This optimization bridges the gap between planning and execution. A general-purpose LLM might provide only textual plans or abstract instructions in response to user queries, which may lack precision. In contrast, a LAM-empowered GUI agent moves beyond planning to actively and intelligently execute tasks on GUIs. By interacting directly with application interfaces, these agents perform tasks with remarkable precision and adaptability. This paradigm shift marks the evolution of GUI agents from passive task planners to active, intelligent executors.
这种优化弥合了规划与执行之间的差距。通用LLM可能仅对用户查询提供文本计划或抽象指令，缺乏精确性。相比之下，具备LAM能力的GUI代理不仅规划，还能主动智能地执行GUI上的任务。通过直接与应用界面交互，这些代理以卓越的精度和适应性完成任务。这一范式转变标志着GUI代理从被动任务规划者向主动智能执行者的演进。

16.5 8.3 LAMs for Web GUI Agents

16.6 8.3 面向Web GUI代理的LAMs

In the domain of web-based GUI agents, researchers have developed specialized LAMs that enhance interaction and navigation within web environments. These models are tailored to understand the complexities of web GUIs, including dynamic content and diverse interaction patterns. We present an analysis of LAMs tailored for web GUI agents in Table 26
在基于网页的GUI代理领域，研究人员开发了专门的语言-动作模型（LAMs），以增强网页环境中的交互和导航能力。这些模型针对网页GUI的复杂性进行了定制，包括动态内容和多样的交互模式。我们在表26中展示了针对网页GUI代理的LAMs分析。

Building upon the need for multimodal understanding, WebGUM [153] integrates HTML understanding with visual perception through temporal and local tokens. It leverages Flan-T5 [498] for instruction fine-tuning and ViT [522] for visual inputs, enabling it to process both textual and visual information efficiently. This multimodal grounding allows WebGUM to generalize tasks effectively, significantly outperforming prior models on benchmarks like MiniWoB++ [146] and WebShop [425]. With its data-efficient design and capacity for multi-step reasoning, WebGUM underscores the importance of combining multimodal inputs in enhancing GUI agent performance.
基于多模态理解的需求，WebGUM [153]通过时间和局部标记整合了HTML理解与视觉感知。它利用Flan-T5 [498]进行指令微调，采用ViT [522]处理视觉输入，使其能够高效处理文本和视觉信息。这种多模态基础使WebGUM能够有效泛化任务，在MiniWoB++ [146]和WebShop [425]等基准测试中显著超越先前模型。凭借其数据高效设计和多步推理能力，WebGUM强调了结合多模态输入以提升GUI代理性能的重要性。

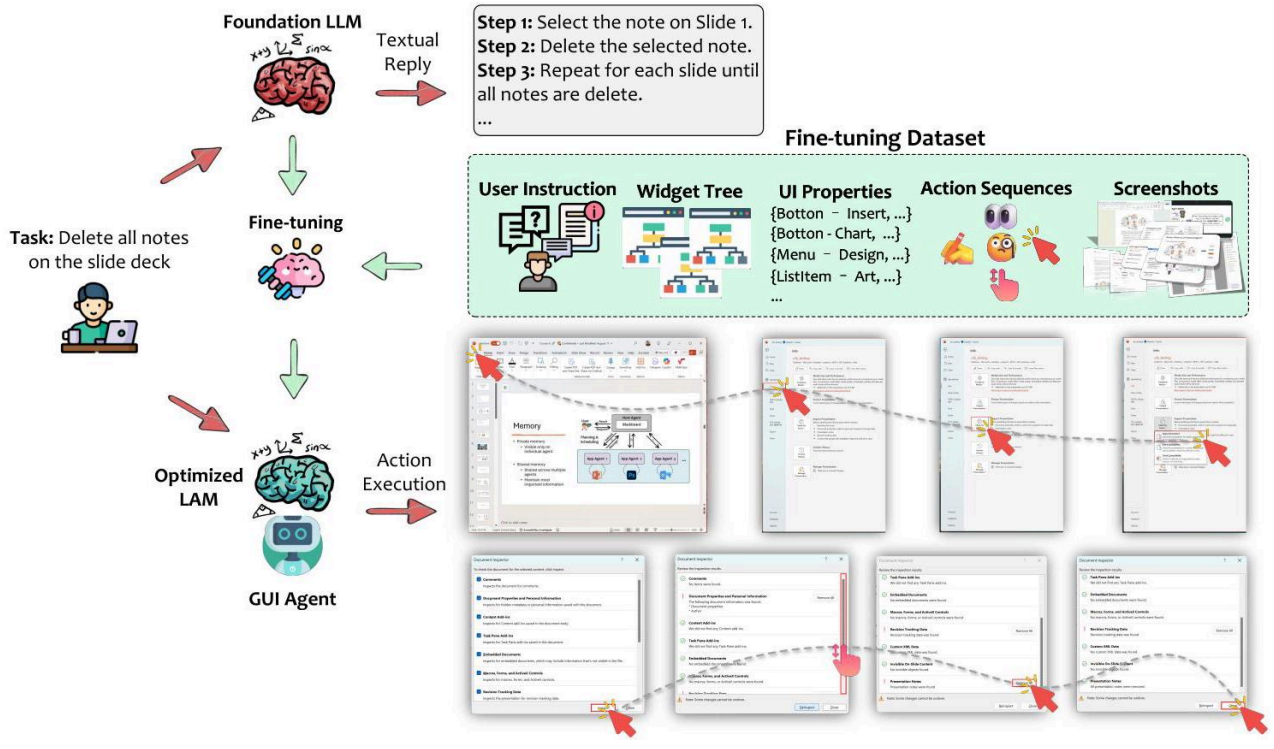

Fig. 26: The evolution from foundation LLMs to GUI agent-optimized LAM with fine-tuning.
图26：从基础大型语言模型（LLMs）到经过微调优化的GUI代理专用语言-动作模型（LAM）的演进。

Addressing the challenge of multi-step reasoning and planning in GUI environments, researchers have introduced frameworks that incorporate advanced search and learning mechanisms. For instance, Agent Q [281] employs MCTS combined with self-critique mechanisms and Direct Preference Optimization (DPO) [503] to improve success rates in complex tasks such as product search and reservation booking. By fine-tuning the LLaMA-3 70B model [91] to process HTML DOM representations and generate structured action plans, thoughts, and environment-specific commands, this framework showcases the power of integrating reasoning, search, and iterative fine-tuning for autonomous agent development.
针对GUI环境中多步推理与规划的挑战，研究人员引入了结合高级搜索与学习机制的框架。例如，Agent Q [281]采用蒙特卡洛树搜索（MCTS）结合自我批判机制和直接偏好优化（DPO）[503]，提升了产品搜索和预订等复杂任务的成功率。通过微调LLaMA-3 70B模型[91]以处理HTML DOM表示并生成结构化的行动计划、思考过程及环境特定指令，该框架展示了推理、搜索与迭代微调整合在自主代理开发中的强大能力。

Leveraging smaller models for efficient web interaction, GLAINTEL [385] demonstrates that high performance can be achieved without large computational resources. Utilizing the Flan-T5 [498] model with 780M parameters, it focuses on dynamic web environments like simulated e-commerce platforms. The model incorporates RL to optimize actions such as query formulation and navigation, effectively integrating human demonstrations and unsupervised learning. Achieving results comparable to GPT-4-based methods at a fraction of the computational cost, GLAINTEL underscores the potential of reinforcement learning in enhancing web-based GUI agents for task-specific optimization.
GLAINTEL [385]利用较小模型实现高效网页交互，证明无需大量计算资源也能达到高性能。该模型采用参数量为7.8亿的Flan-T5 [498]，专注于动态网页环境如模拟电商平台。模型结合强化学习优化查询构建和导航等动作，有效整合人类示范与无监督学习。GLAINTEL以远低于GPT-4方法的计算成本取得了相当的结果，凸显了强化学习在提升基于网页的GUI代理任务特定优化中的潜力。

To enable continuous improvement and generalization across varied web domains, OpenWebVoyager [387] combines imitation learning with an iterative exploration-feedback-optimization cycle. Leveraging large multimodal models like Idefics2-8B [523], it performs autonomous web navigation tasks. By training on diverse datasets and fine-tuning using trajectories validated by GPT-4 feedback, the agent addresses real-world complexities without relying on synthetic environments. This approach significantly advances GUI agent frameworks by demonstrating the capability to generalize across varied web domains and tasks.
为实现跨多样网页领域的持续改进与泛化，OpenWebVoyager [387]结合模仿学习与迭代探索-反馈-优化循环。利用大型多模态模型如Idefics2-8B [523]，其执行自主网页导航任务。通过在多样数据集上训练并利用GPT-4反馈验证的轨迹进行微调，该代理应对了真实世界的复杂性，避免依赖合成环境。此方法显著推动了GUI代理框架的发展，展示了跨多样网页领域和任务泛化的能力。

Moreover, tackling challenges such as sparse training data and policy distribution drift, WebRL 388 introduces a self-evolving curriculum and robust reward mechanisms for training LLMs as proficient web agents. By dynamically generating tasks based on the agent's performance, WebRL fine-tunes models like Llama-3.1 [91] and GLM-4 [499], achieving significant success rates in web-based tasks within the WebArena environment. This framework outperforms both proprietary APIs and other open-source models, highlighting the effectiveness of adaptive task generation and sustained learning improvements in developing advanced GUI agents.
此外，针对训练数据稀缺和策略分布漂移等挑战，WebRL 388引入了自我进化课程和稳健奖励机制，用于训练大型语言模型成为高效网页代理。通过基于代理表现动态生成任务，WebRL微调了Llama-3.1 [91]和GLM-4 [499]，在WebArena环境中的网页任务中取得显著成功率。该框架超越了专有API和其他开源模型，凸显了自适应任务生成和持续学习改进在开发先进GUI代理中的有效性。

These advancements in LAMs for web GUI agents illustrate the importance of integrating multimodal inputs, efficient model designs, and innovative training frameworks to enhance agent capabilities in complex web environments.
这些针对网页GUI代理的语言-动作模型进展，体现了整合多模态输入、高效模型设计及创新训练框架在提升复杂网页环境中代理能力的重要性。

16.7 8.4 LAMs for Mobile GUI Agents

16.8 8.4 移动GUI代理的语言-动作模型

Mobile platforms present unique challenges for GUI agents, including diverse screen sizes, touch interactions, and resource constraints. Researchers have developed specialized LAMs to address these challenges, enhancing interaction and navigation within mobile environments. We present an overview of LAMs tailored for mobile GUI agents in Table 27 and 28
移动平台为GUI代理带来了独特挑战，包括多样的屏幕尺寸、触控交互和资源限制。研究人员开发了专门的语言-动作模型以应对这些挑战，提升移动环境中的交互和导航能力。我们在表27和28中展示了针对移动GUI代理的语言-动作模型概览。

TABLE 26: An overview of GUI-optimized models on web platforms.
表26：网页平台上针对GUI优化模型的概览。

Model	Platform	Foundation Model	Size	Input	Output	Dataset	Highlights	Link
Agent Q 281]	Web	LLaMA-3 70B 91J	70B	HTML DOM representations	Plans, thoughts, actions, and actio explanations	WebShop benchmark and OpenTable dataset	Combines Monte Carlo Tree Search (MCTS) with self-critique mechanisms, leveraging reinforcement learning to achieve exceptional performance	https://github.com/sentient%2Dengineering/ agent-q
GLAINTEL 385]	Web	Flan-T5 498]	780M	User instructions and observations of webpage state	GUI actions	1.18M real-world products, 12,087 crowd-sourced natural language intents, 1,010 human demonstrations	Efficient use of smaller LLMs, and integration of RL and human demonstrations for robust performance	/
WebN-T5 386	Web	T5 87]	-	HTML and DOM with screenshots	Hierarchical navigation plans and GUI interactions	MiniWoB++, 13,000 human-made demonstrations	Combines supervised learning and reinforcement learning to address limitations of previous models in memorization and generalization	/
OpenWeb-We Voyager 387]		ldefics2- 8b-instruct 523]	8B	GUI screenshots, accessibility trees	Actions on GUI, planning and thought, answers queries	Mind2Web WebVoyager datasets and generated queries for real-world web navigation	Combining imitation learning with a feedback loop for continuous improvement	https: //github.com/ MinorJerry/ OpenWebVoyager
WebRL 388	Web	Llama-3.1 91 and GLM-4 524]	8B/9B/ 70B	Task instructions, action history, HTML content	Actions. element identifiers. explanations or notes	WebArena-Lite	Introduces a self-evolving online curriculum reinforcement learning framework, which dynamically generates tasks based on past failures and adapts to the agent's skill level	https://github.com/THUDM/ WebRL

WebGUM 153]	Web	Flan-T5 498] and Vision Transformer (ViT) 522]	3B	HTML, screenshots, interaction history. instructions	Web navigation actions and freeform text	MiniWoB++ and WebShop benchmarks	Integrates temporal and local multimodal perception, combining HTML and visual tokens, and uses an instruction-finetuned language model for enhanced reasoning and task generalization	https://console.cloud.google com/storage/ browser/ gresearch/ weblim

模型	平台	基础模型（Foundation Model）	规模	输入	输出	数据集	亮点	链接
Agent Q 281]	网页	LLaMA-3 70B 91J	70B	HTML DOM 表示	计划、思考、行动及行动解释	WebShop 基准和 OpenTable 数据集	结合蒙特卡洛树搜索（Monte Carlo Tree Search, MCTS）与自我批评机制，利用强化学习实现卓越性能	https://github.com/sentient% 2Dengineering/ agent-q
GLAINTEL 385]	网页	Flan-T5 498]	780M	用户指令与网页状态观察	图形用户界面（GUI）操作	118万真实产品，12,087条众包自然语言意图，1,010个人工示范	高效利用小型大型语言模型（LLM），结合强化学习与人工示范以实现稳健性能	/
WebN-T5 386	网页	T5 87]	-	HTML 和 DOM 及截图	分层导航计划与图形用户界面交互	MiniWoB++，13,000个人工示范	结合监督学习与强化学习，解决先前模型在记忆与泛化上的局限	/
OpenWeb- We Voyager 387]		ldefics2- 8b- instruct 523]	8B	图形用户界面截图，辅助功能树	图形用户界面操作、规划与思考，回答查询	Mind2Web WebVoyager 数据集及生成查询，用于真实网页导航	结合模仿学习与反馈循环，实现持续改进	https: //github.com/ MinorJerry/ OpenWebVoyager
WebRL 388	网页	Llama-3.1 91 和 GLM-4 524]	8B/9B/ 70B	任务指令、操作历史、HTML 内容	操作、元素标识符、解释或注释	WebArena-Lite	引入自我进化的在线课程强化学习框架，基于过去失败动态生成任务，并适应代理技能水平	https://github.com/THUDM/ WebRL
WebGUM 153]	网页	Flan-T5 498] 和视觉变换器（Vision Transformer, ViT）522]	3B	HTML、截图、交互历史、指令	网页导航操作与自由文本	MiniWoB++ 和 WebShop 基准	整合时序与局部多模态感知，结合 HTML 与视觉标记，使用指令微调语言模型以增强推理与任务泛化能力	https://console.cloud.google com/storage/ browser/ gresearch/ weblim

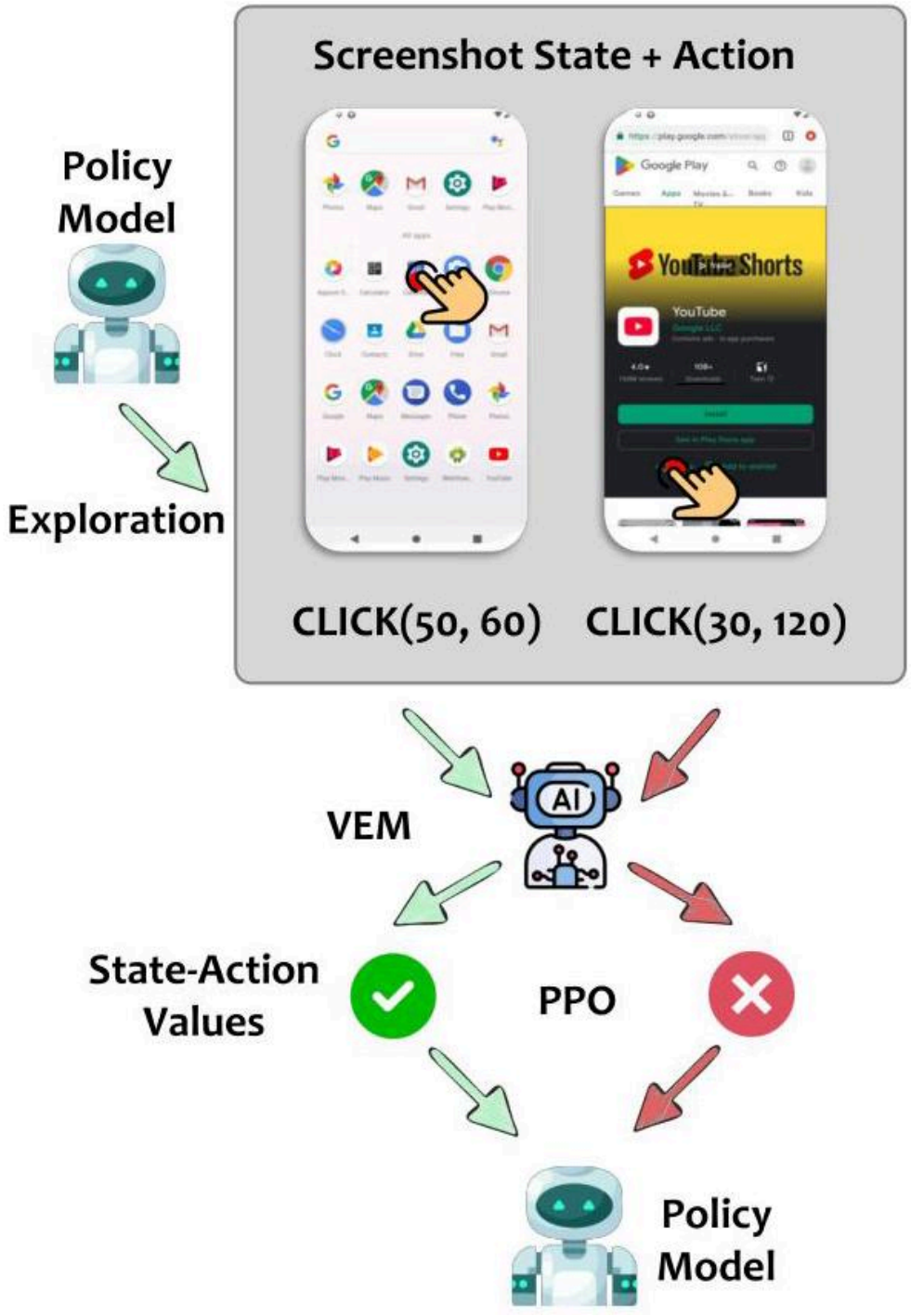

Fig. 27: The PPO training process of VEM 399. Figure adapted from the original paper.
图27：VEM 399的PPO训练过程。图示改编自原论文。

Focusing on detailed UI understanding, MobileVLM [353] introduces an advanced vision-language model designed specifically for mobile UI manipulation tasks. Built on Qwen-VL-Chat [210], it incorporates mobile-specific pretraining tasks for intra- and inter-UI comprehension. By leveraging the Mobile3M dataset—a comprehensive corpus of 3 million UI pages and interaction traces organized into directed graphs-the model excels in action prediction and navigation tasks. MobileVLM's novel two-stage pretraining framework significantly enhances its adaptability to mobile UIs, outperforming existing VLMs in benchmarks like ScreenQA [532] and Auto-UI [302]. This work highlights the effectiveness of tailored pretraining in improving mobile GUI agent performance.
专注于细致的用户界面理解，MobileVLM [353] 引入了一种专为移动UI操作任务设计的先进视觉语言模型。该模型基于Qwen-VL-Chat [210]，融合了针对移动设备的UI内部及跨UI理解的预训练任务。通过利用Mobile3M数据集——一个包含300万UI页面及交互轨迹、以有向图形式组织的综合语料库——该模型在动作预测和导航任务中表现出色。MobileVLM新颖的两阶段预训练框架显著提升了其对移动UI的适应能力，在ScreenQA [532]和Auto-UI [302]等基准测试中优于现有视觉语言模型。该研究凸显了定制预训练在提升移动图形用户界面代理性能方面的有效性。

Addressing the need for robust interaction in dynamic environments, DigiRL 264 presents a reinforcement learning-based framework tailored for training GUI agents in Android environments. By leveraging offline-to-online RL, DigiRL adapts to real-world stochasticity, making it suitable for diverse, multi-step tasks. Unlike prior models reliant on imitation learning, DigiRL autonomously learns from interaction data, refining itself to recover from errors and adapt to new scenarios. The use of a pre-trained Vision-Language Model with 1.3 billion parameters enables efficient processing of GUI screenshots and navigation commands. Its performance on the AITW dataset demonstrates a significant improvement over baseline methods, positioning DigiRL as a benchmark in the development of intelligent agents optimized for complex GUI interactions.
针对动态环境中对稳健交互的需求，DigiRL 264提出了一个基于强化学习的框架，专门用于训练Android环境下的GUI代理。通过离线到在线的强化学习方法，DigiRL适应现实世界的随机性，适合多样化的多步骤任务。不同于依赖模仿学习的先前模型，DigiRL自主从交互数据中学习，能够自我优化以纠正错误并适应新场景。其采用了一个拥有13亿参数的预训练视觉语言模型，能够高效处理GUI截图和导航指令。在AITW数据集上的表现显著优于基线方法，使DigiRL成为优化复杂GUI交互智能代理开发的标杆。

Both Digi-Q [398] and VEM [399] investigate the use of offline RL to enhance the performance of GUI agents without requiring direct interaction with the environment. Digi-Q employs temporal-difference learning to train a Q-function offline and derives policies through a Best-of-N selection strategy based on the predicted Q-values. Similarly, VEM introduces an environment-free RL framework tailored for training LLM-powered GUI agents using PPO. It directly estimates state-action values from offline data by fine-tuning with annotated value data from GPT-40, thereby enabling policy training without real-time execution in a GUI environment. At inference time, only the policy model is utilized. Figure 27 illustrates the overall architecture of VEM. The study further demonstrates that offline RL with structured credit assignment can achieve performance comparable to interactive RL models. Overall, VEM offers a scalable and layout-agnostic approach for training GUI agents while minimizing interaction costs. Both works underscore the potential of offline RL for GUI agent training.
Digi-Q [398]和VEM [399]均探讨了利用离线强化学习提升GUI代理性能的方法，无需与环境直接交互。Digi-Q采用时序差分学习离线训练Q函数，并通过基于预测Q值的N选优策略导出策略。类似地，VEM提出了一个无环境强化学习框架，专为使用PPO训练基于大型语言模型（LLM）的GUI代理设计。它通过微调来自GPT-40的带注释价值数据，直接从离线数据估计状态-动作值，从而实现无需实时GUI环境执行的策略训练。推理时仅使用策略模型。图27展示了VEM的整体架构。研究进一步表明，带有结构化信用分配的离线强化学习能够达到与交互式强化学习模型相当的性能。总体而言，VEM提供了一种可扩展且与布局无关的GUI代理训练方法，同时最大限度降低了交互成本。两项工作均强调了离线强化学习在GUI代理训练中的潜力。

TABLE 27: An overview of GUI-optimized models on mobile platforms (Part I).
表27：移动平台上GUI优化模型概览（第一部分）。

Model	Platform	Foundation Model	Size	Input	Output	Dataset	Highlights	Link
Mobile-VLM 353]	Mobile Android	Qwen-VL-Chat 210]	9.8B	Screenshots and structured XML documents	Action predictions. navigation steps, and element locations	Mobile3M, includes 3 million UI pages, 20+ million actions, and XML data structured as directed graphs	Mobile-specific pretraining tasks that enhance intra- and inter-UI understanding, with a uniquely large and graph-structured Chinese UI dataset (Mobile3M)	https://github.com/XiaoMi/mobilevlm
Octo-planner 361]	Mobile devices	Phi-3 N Mini 234]	3.8B	User queries and available function descriptions	Execution steps	1,000 data samples generated using GPT-4	Optimized for resource-constrained devices to ensure low latency, privacy, and offline functionality	https: //huggingface. co/NexaAIDev/ octopus-planning
DigiRL 264]	Mobile Android	AutoUI 302]	1.3B	Screenshots	GUI actions	AiTW	Offline-to-online reinforcement learning, bridging gaps in static and dynamic environments	https: //github.com/ DigiRL-agent/ digirl
LVG 390]	Mobile Android	Swin Transformer [525] and BERT 82]	-	UI screenshots and free-form language expressions	Bounding box coordinates	UIBert dataset and synthetic dataset	Unifies detection and grounding tasks through layout-guided contrastive learning	/
Ferret-UI 352	Android and iPhone platforms	Ferret | 382]	7B/13B	Raw screen pixels, sub-images divided for finer resolution, bounding boxes and regional annotations	Widget bounding boxes, text from OCR tasks descriptions of UI elements or overall screen functionality, UI interaction actions	Generated from RICO (for Android) and AMP (for iPhone)	Multi-platform support with high-resolution adaptive image encoding	https://github.com/apple/ml-ferret/tree/ main/ferretui
Octopus 391]	Mobile devices	CodeLlama-7B 5261 Google Gemma 2B 502	7B, 2B	API documentation examples	Function names with arguments for API calls	RapidAPI Hub	Use of conditional masking to enforce correct output formatting	/
Octopus v2 392]	Edge devices	Gemma-2B 502]	2B	User queries and descriptions of available functions	Function calls with precise parameters	20 Android APIs. with up to 1,000 data points generated for training	Functional tokenization strategy. which assigns unique tokens to function calls, significantly reducing the context length required for accurate prediction	/

Octopus v3 393]	Edge devices	CLIP-based model and a causal language model	Less than 1 billion parameters	Queries and commands, images and functional tokens	Functional tokens for actions	Leveraged from Octopus v2 [392]	Introduction of functional tokens for multimodal applications enables the representation of any function as a token, enhancing the model's flexibility	/
Octopus v4 394]	Serverless cloud-based platforms and edge devices	17 models	Varies	User queries	Domain-specific answers, actions	Synthetic datasets similar to Octopus v2	Graph-based framework integrating multiple specialized models for optimized performance	https://github.com/NexaAI/octopus-v4

模型	平台	基础模型	规模	输入	输出	数据集	亮点	链接
移动端-VLM 353]	移动安卓	Qwen-VL-Chat 210]	9.8B	截图和结构化XML文档	动作预测、导航步骤和元素位置	Mobile3M，包含300万UI页面，超过2000万动作，以及以有向图形式结构化的XML数据	针对移动端的预训练任务，增强UI内部及跨UI理解，拥有独特的大规模图结构中文UI数据集（Mobile3M）	https://github.com/XiaoMi/mobilevlm
Octo-planner 361]	移动设备	Phi-3 N Mini 234]	3.8B	用户查询和可用功能描述	执行步骤	使用GPT-4生成的1000个数据样本	针对资源受限设备优化，确保低延迟、隐私保护和离线功能	https: //huggingface.co/NexaAlDev/ octopus-planning
DigiRL 264]	移动安卓	AutoUI 302]	1.3B	截图	图形用户界面动作	AiTW	离线到在线的强化学习，弥合静态与动态环境的差距	https: //github.com/ DigiRL-agent/ digirl
LVG 390]	移动安卓	Swin Transformer [525] 和 BERT 82]	-	UI截图和自由形式语言表达	边界框坐标	UIBert数据集和合成数据集	通过布局引导的对比学习统一检测和定位任务	/
Ferret-UI 352	安卓和iPhone平台	Ferret | 382]	7B/13B	原始屏幕像素，细分子图以获得更高分辨率，边界框和区域注释	控件边界框，OCR提取文本，UI元素描述或整体屏幕功能，UI交互动作	基于RICO（安卓）和AMP（iPhone）生成	多平台支持，具备高分辨率自适应图像编码	https://github.com/apple/ ml-ferret/tree/ main/ferretui
Octopus 391]	移动设备	CodeLlama-7B 5261 Google Gemma 2B 502	7B，2B	API文档示例	带参数的API调用函数名	RapidAPI中心	使用条件掩码以确保输出格式正确	/
Octopus v2 392]	边缘设备	Gemma-2B 502]	2B	用户查询和可用功能描述	带精确参数的函数调用	20个安卓API，生成最多1000个数据点用于训练	功能标记化策略，为函数调用分配唯一标记，显著减少准确预测所需的上下文长度	/
Octopus v3 393]	边缘设备	基于CLIP的模型和因果语言模型	参数少于10亿	查询与命令，图像与功能标记	用于操作的功能标记	借鉴自 Octopus v2 [392]	引入多模态应用的功能标记，使任何功能都能表示为标记，提升模型的灵活性	/
Octopus v4 394]	无服务器云平台与边缘设备	17 个模型	各不相同	用户查询	特定领域的答案与操作	类似于 Octopus v2 的合成数据集	基于图的框架，整合多个专用模型以优化性能	https://github.com/NexaAI/octopus-v4

To enhance GUI comprehension and reduce reliance on textual data, VGA 354 employs fine-tuned vision-language models that prioritize image-based cues such as shapes, colors, and positions. Utilizing the RICO [355] dataset for training, VGA is tailored for Android GUIs and employs a two-stage fine-tuning process to align responses with both visual data and human intent. The model excels in understanding GUI layouts, predicting design intents, and facilitating precise user interactions. By outperforming existing models like GPT- 4V in GUI comprehension benchmarks, VGA sets a new standard for accuracy and efficiency in mobile GUI agents.
为了增强图形用户界面（GUI）的理解能力并减少对文本数据的依赖，VGA 354采用了微调的视觉-语言模型，重点关注形状、颜色和位置等基于图像的线索。利用RICO [355]数据集进行训练，VGA专为安卓GUI设计，采用两阶段微调过程，使模型响应与视觉数据及人类意图相匹配。该模型在理解GUI布局、预测设计意图和促进精确用户交互方面表现出色。通过在GUI理解基准测试中超越GPT-4V等现有模型，VGA为移动GUI代理树立了新的准确性和效率标准。

In the context of lightweight and efficient models, UINav [395] demonstrates a practical system for training neural agents to automate UI tasks on mobile devices. It balances accuracy, generalizability, and computational efficiency through macro actions and an error-driven demonstration collection process. UINav uses a compact encoder-decoder architecture and SmallBERT [528] for text and screen element encoding, making it suitable for on-device inference. A key innovation is its ability to generalize across diverse tasks and apps with minimal demonstrations, addressing key challenges in UI automation with a versatile framework.
在轻量高效模型的背景下，UINav [395]展示了一个实用系统，用于训练神经代理自动化移动设备上的用户界面任务。它通过宏动作和基于错误的示范收集过程，在准确性、泛化能力和计算效率之间取得平衡。UINav采用紧凑的编码器-解码器架构和SmallBERT [528]进行文本及屏幕元素编码，适合设备端推理。其关键创新在于能够通过极少的示范实现跨多样任务和应用的泛化，解决了UI自动化中的核心挑战，构建了一个多功能框架。

UI-R1 [401] introduces a RL-based training paradigm aimed at enhancing GUI action prediction for multimodal large language models (MLLMs). The resulting model, UI-R1-3B, fine-tunes Qwen2.5-VL-3B using a novel rule-based reward function that jointly evaluates action type correctness and click coordinate accuracy, while also enabling o1-style [533] chain-of-thought (CoT) reasoning through structured tags. UI-R1 relies on only 136 high-quality samples selected via a three-stage filtering strategy. Despite this limited supervision, UI-R1-3B achieves significant improvements on both in-domain and out-of-domain benchmarks. By leveraging Group Relative Policy Optimization (GRPO) [534], the framework aligns policy optimization with the goals of GUI grounding and task execution. UI-R1 establishes a scalable and data-efficient approach for training GUI agents via RL and paves the way for lightweight yet effective agent design. Its methodology has also been successfully extended to cross-platform agents [410], [411], demonstrating strong generalization capabilities.
UI-R1 [401]提出了一种基于强化学习（RL）的训练范式，旨在提升多模态大语言模型（MLLM）对GUI动作的预测能力。最终模型UI-R1-3B通过一种新颖的基于规则的奖励函数微调Qwen2.5-VL-3B，该函数联合评估动作类型的正确性和点击坐标的准确性，同时通过结构化的标签实现o1风格[533]的链式思维（CoT）推理。UI-R1仅依赖通过三阶段筛选策略选出的136个高质量样本。尽管监督有限，UI-R1-3B在内域和外域基准测试中均取得显著提升。通过利用群体相对策略优化（GRPO）[534]，该框架将策略优化与GUI定位及任务执行目标对齐。UI-R1建立了一种可扩展且数据高效的RL训练GUI代理方法，并为轻量且高效的代理设计铺平了道路。其方法也成功扩展至跨平台代理[410]，[411]，展现出强大的泛化能力。

TABLE 28: An overview of GUI-optimized models on mobile platforms (Part II).
表28：移动平台上GUI优化模型概览（第二部分）。

Model	Platform	Foundation Model	Size	Input	Output	Dataset	Highlights	Link
VGA 354]	Mobile Android	LLaVA-v1.6-mistral-7B 217]	7B	GUI screenshots with with positional, visual, and hierarchical data	Actions and function calls, descriptions of GUI components, navigation and task planning	63.8k-image dataset constructed from the RICO	Minimizes hallucinations in GUI comprehension by employing an image-centric fine-tuning approach, ensuring balanced attention between text and visual content	https: //github.com/ Linziyang1999/ VGA% 2Dvisual% 2DGUI% 2Dassistant
MobileFlow Mobile 159]	phones	Qwen-VL-Chat [210]	21B	GUI screenshots with OCR textual information and bounding boxes	GUI actions and question answering	70k manually labeled business-specific data spanning 10 business sectors, and datasets like RefCOCO, ScreenQA, Flickr30K	Hybrid visual encoder capable of variable-resolution input and Mixture of Experts (MoE) [527] for enhanced performance and efficiency	7
UINav 395]	Mobile Android	SmallBERT 528]	Agent model: 320k, Referee model: 430k, Small-BERT model: 17.6MB	UI elements, utterance, screen representation	Predicted actions and element to act upon	43 tasks across 128 Android apps and websites, collecting 3,661 demonstrations	Introduces a macro action framework and an error-driven demonstration collection process, significantly reducing training effort while enabling robust task performance with small, efficient models suitable for mobile devices	/
AppVLM 397]	Android mobile devices	Paligemma-3B-896 529]	3B	Annotated screen-shots with bounding boxes and UI labels	GUI actions	AndroidControl 515], AndroidWorld 440]	A lightweight model that achieves near-GPT-40 performance in Android control tasks while being ${10} \times$ faster and more resource-efficient.	/

VSC-RL 396]	Mobile Android	AutoUI 302 Gemini-1.5-Pro	/	Screenshots	GUI actions	AitW	Addresses sparse-reward, long-horizon tasks for RL by autonomously breaking a complicated goal into subgoals	https:// ai-agents-2030 github.io/ VSC-RL
Diai-Q 398]	Mobile Android	LLaVA-1.5 383	7B	GUI screenshots	GUI actions, Q-values	AitW 358]	Introduces a VLM-based Q-function for GUI agent training, enabling reinforcement learning without online interactions.	https: //github.com/ DigiRL-agent/ digiq
VEM 399]	Mobile Android	Qwen2VL 231]	7B	GUI screenshots	GUI actions, Q-values	AitW 358]	Unlike traditional RL methods that require environment interactions, VEM enables training purely on offline data with a Value Environment Model.	https://github.com/microsoft/ GUI-Agent-RL
MP-GUI 400]	Mobile Android	InternViT-300M and InternLM2.5-7B-chat 530]	8B	GUI screenshots	Natural language output, element grounding, captioning, and semantic navigation	680K mixed-modality dataset	Introduces a tri-perceiver architecture that models textual, graphical, and spatial modalities to enhance GUI reasoning	https://github.com/Big Taige/ MP-GUI
UI-R1 401]	Mobile Android	Qwen2.5-VL-3B	3B	GUI screenshots	Reasoning text and GUI actions	ScreenSpo and An-droidControl	Introduces a rule-based reinforcement learning approach using GRPO to enhance reasoning and action prediction in GUI tasks with only 136 examples	https: //github.com/ III6gg/UI-R1
ViMo 402]	Mobile Android	Pre-trained Stable Diffusion model [531]	/	Current GUI image. user action (in natural language), GUI text representation	GUI text representation of the next state and reconstructed full GUI image (visual prediction of the next screen)	Android Control and AITW	First GUI world model that predicts future visual GUI states	https:// ai-agents-2030. github.io/ViMo/

模型	平台	基础模型（Foundation Model）	规模	输入	输出	数据集	亮点	链接
VGA 354]	移动安卓	LLaVA-v1.6-mistral-7B 217]	7B	带有位置、视觉和层级数据的GUI截图	操作和函数调用，GUI组件描述，导航与任务规划	由RICO构建的63.8k图像数据集	通过采用以图像为中心的微调方法，最大限度减少GUI理解中的幻觉，确保文本与视觉内容之间的注意力平衡	https: //github.com/ Linziyang1999/ VGA% 2Dvisual% 2DGUI% 2Dassistant
MobileFlow 移动 159]	手机	Qwen-VL-Chat [210]	21B	带有OCR文本信息和边界框的GUI截图	GUI操作与问答	涵盖10个行业的7万条手工标注业务专用数据，以及RefCOCO、ScreenQA、Flickr30K等数据集	支持可变分辨率输入的混合视觉编码器及专家混合（MoE）[527]，提升性能与效率	7
UINav 395]	移动安卓	SmallBERT 528]	代理模型：320k，裁判模型：430k，Small-BERT模型：17.6MB	UI元素、话语、屏幕表示	预测操作及目标元素	涵盖128个安卓应用和网站的43个任务，收集了3,661个示范	引入宏操作框架和基于错误驱动的示范收集流程，大幅减少训练工作量，同时实现适合移动设备的小型高效模型的稳健任务表现	/
AppVLM 397]	安卓移动设备	Paligemma-3B-896 529]	3B	带有边界框和UI标签的注释截图	GUI操作	AndroidControl 515], AndroidWorld 440]	一款轻量级模型，在安卓控制任务中达到接近GPT-4性能，同时速度更快且资源更高效。	/
VSC-RL 396]	移动安卓	AutoUI 302 Gemini-1.5-Pro	/	截图	GUI操作	AitW	通过自主将复杂目标拆分为子目标，解决稀疏奖励和长时程强化学习任务	https:// ai-agents-2030 github.io/ VSC-RL
Diai-Q 398]	移动安卓	LLaVA-1.5 383	7B	GUI截图	GUI操作，Q值	AitW 358]	引入基于视觉语言模型（VLM）的Q函数用于GUI代理训练，实现无需在线交互的强化学习。	https: //github.com/ DigiRL-agent/ digiq
VEM 399]	移动安卓	Qwen2VL 231]	7B	GUI截图	GUI操作，Q值	AitW 358]	不同于传统需要环境交互的强化学习方法，VEM通过价值环境模型实现纯离线数据训练。	https://github.com/microsoft/ GUI-Agent-RL
MP-GUI 400]	移动安卓	InternViT-300M 和 InternLM2.5-7B-chat 530]	8B	GUI截图	自然语言输出、元素定位、图注和语义导航	68万混合模态数据集	引入三感知器架构，建模文本、图形和空间模态，增强GUI推理能力	https://github.com/Big Taige/ MP-GUI
UI-R1 401]	移动安卓	Qwen2.5- VL-3B	3B	GUI截图	推理文本与GUI操作	ScreenSpo 和 AndroidControl	介绍了一种基于规则的强化学习方法，使用GRPO（Guided Reinforcement Policy Optimization）在仅有136个样本的情况下提升GUI任务中的推理和动作预测能力	https: //github.com/ III6gg/UI-R1

ViMo 402]	移动安卓	预训练的 Stable Diffusion模型 [531]	/	当前GUI图像，用户操作（自然语言描述），GUI文本表示	下一状态的GUI文本表示及重建的完整GUI图像（下一屏幕的视觉预测）	Android Control 和 AITW	首个预测未来视觉GUI状态的GUI世界模型	https:// ai-agents-2030. github.io/ViMo/

In addition to action models, ViMo [402] introduces a novel generative visual world model for GUI agents, aimed at improving App agent decision-making by predicting the next GUI state as an image rather than a textual description. A key innovation of ViMo is the Symbolic Text Representation (STR), which replaces GUI text regions with structured placeholders to facilitate accurate and legible text synthesis. This decoupled design allows the system to handle GUI graphics generation using a fine-tuned diffusion model, and text generation through an LLM, thereby achieving high visual fidelity and semantic precision. ViMo significantly boosts both GUI prediction quality and downstream agent performance, with a reported 29.14% relative improvement in GUI generation metrics and enhanced planning accuracy for long-horizon tasks. As a forward simulator, ViMo represents a crucial advancement toward reliable world models for mobile GUI agents, supporting more effective decision evaluation and trajectory planning in visual environments.

除了动作模型外，ViMo [402] 引入了一种新颖的生成式视觉世界模型，用于GUI代理，旨在通过预测下一个GUI状态的图像而非文本描述来提升应用代理的决策能力。ViMo的一项关键创新是符号文本表示（Symbolic Text Representation，STR），它用结构化占位符替代GUI文本区域，以便实现准确且清晰的文本合成。这种解耦设计使系统能够通过微调的扩散模型处理GUI图形生成，通过大型语言模型（LLM）进行文本生成，从而实现高视觉保真度和语义精确性。ViMo显著提升了GUI预测质量和下游代理性能，据报道在GUI生成指标上相对提升了29.14%，并增强了长远任务的规划准确性。作为一个前向模拟器，ViMo代表了面向移动GUI代理的可靠世界模型的重要进展，支持在视觉环境中更有效的决策评估和轨迹规划。

16.9 8.5 LAMs for Computer GUI Agents

16.10 8.5 计算机GUI代理的语言动作模型（LAMs）

TABLE 29: An overview of GUI-optimized models on computer platforms.
表29：计算机平台上针对GUI优化模型的概览。

Model	Platform	Foundation Model	Size	Input	Output	Dataset	Highlights	Link
Screen-Agent 366]	Linux and Windows desktop	CogAgent [15]	18B	GUI screenshots	Mouse and keyboard actions	273 task sessions	Comprehensive pipeline of planning, acting, and reflecting to handle real computer screen operations autonomously	https: //github.com/ niuzaisheng/ ScreenAgent
Octopus 403]	Desktop	MPT-7B 535 and CLIP ViT-L/14 509]	7B	Visual images, scene graphs containing objects relations, environment messages	Executable action code and plans	OctoGibson: 476 tasks with structured initial and goal states; OctoMC: 40 tasks across biomes; OctoGTA: 25 crafted tasks spanning different game settings	Incorporates reinforcement learning with environmental feedback	https: //choiszt.github. io/Octopus/
LAM 367	Windows OS	Mistral-7B 501]	7B	Task requests in natural language, application environmental data	Plans, actions	76,672 task-plan pairs, 2,192 task-action trajectories	The LAM model bridges the gap between planning and action execution in GUI environments. It introduces a multi-phase training pipeline combining task planning, imitation learning, self-boosting exploration, and reward-based optimization for robust action-oriented performance.	https://github.com/microsoft/ UFO/tree/main/ dataflow
ScreenLL 404]	MDesktop	LLaVA	7B, 13B	GUI screenshots	Predicted GUI actions	High-resolution YouTube tutorials	Introduces a novel stateful screen schema to compactly represent GUI interactions over time, enabling fine-grained understanding and accurate action prediction	/

模型	平台	基础模型（Foundation Model）	规模	输入	输出	数据集	亮点	链接
屏幕-代理 366]	Linux和Windows桌面	CogAgent [15]	18B	图形用户界面截图	鼠标和键盘操作	273个任务会话	规划、执行与反思的综合流程，能够自主处理真实计算机屏幕操作	https: //github.com/niuzaisheng/ScreenAgent
Octopus 403]	桌面	MPT-7B 535和CLIP ViT-L/14 509]	7B	视觉图像、包含对象关系的场景图、环境信息	可执行的动作代码和计划	OctoGibson：476个具有结构化初始和目标状态的任务；OctoMC：跨生物群系的40个任务；OctoGTA：涵盖不同游戏设置的25个精心设计任务	结合环境反馈的强化学习	https: //choiszt.github.io/Octopus/
LAM 367	Windows操作系统	Mistral-7B 501]	7B	自然语言的任务请求，应用环境数据	计划，动作	76,672对任务-计划对，2,192条任务-动作轨迹	LAM模型弥合了图形用户界面环境中规划与动作执行的鸿沟。它引入了多阶段训练流程，结合任务规划、模仿学习、自我增强探索和基于奖励的优化，实现了稳健的面向动作的性能。	https://github.com/microsoft/UFO/tree/main/ dataflow
ScreenLL 404]	MDesktop	LLaVA	7B，13B	图形用户界面截图	预测的图形用户界面动作	高清YouTube教程	引入了一种新颖的有状态屏幕模式，用于紧凑地表示随时间变化的图形用户界面交互，实现细粒度理解和精准动作预测	/

For desktop and laptop environments, GUI agents must handle complex applications, multitasking, and varied interaction modalities. Specialized LAMs for computer GUI agents enhance capabilities in these settings, enabling more sophisticated task execution. We overview of LAMs for computer GUI agents across in Table 29.
对于桌面和笔记本环境，GUI代理必须处理复杂的应用、多任务以及多样的交互方式。专门针对计算机GUI代理的LAM（大动作模型）增强了这些环境下的能力，实现更复杂的任务执行。表29中概述了计算机GUI代理的LAM。

Integrating planning, acting, and reflecting phases, ScreenAgent 366 is designed for autonomous interaction with computer screens. Based on CogAgent [15], it is fine-tuned using the ScreenAgent Dataset, providing comprehensive GUI interaction data across diverse tasks. With inputs as screenshots and outputs formatted in JSON for mouse and keyboard actions, ScreenAgent achieves precise UI element localization and handles continuous multi-step tasks. Its capability to process real-time GUI interactions using a foundation model sets a new benchmark for LLM-powered GUI agents, making it an ideal reference for future research in building more generalized intelligent agents.
ScreenAgent 366整合了规划、执行和反思阶段，旨在实现与计算机屏幕的自主交互。基于CogAgent [15]，通过ScreenAgent数据集进行微调，提供涵盖多样任务的全面GUI交互数据。其输入为截图，输出为鼠标和键盘操作的JSON格式，ScreenAgent实现了精确的UI元素定位并能处理连续的多步骤任务。其利用基础模型处理实时GUI交互的能力，为基于大语言模型（LLM）的GUI代理树立了新标杆，是未来构建更通用智能代理研究的理想参考。

Bridging high-level planning with real-world manipulation, Octopus [403] represents a pioneering step in embodied vision-language programming. Leveraging the MPT-7B [535] and CLIP ViT-L/14 [509], Octopus integrates egocentric and bird's-eye views for visual comprehension, generating executable action code. Trained using the OctoVerse suite, its datasets encompass richly annotated environments like OmniGibson, Minecraft, and GTA-V, covering routine and reasoning-intensive tasks. Notably, Octopus innovates through Reinforcement Learning with Environmental Feedback, ensuring adaptive planning and execution. Its vision-dependent functionality offers seamless task generalization in unseen scenarios, underscoring its capability as a unified model for embodied agents operating in complex GUI environments.
Octopus [403]作为具身视觉语言编程的开创性成果，桥接了高层规划与现实操作。利用MPT-7B [535]和CLIP ViT-L/14 [509]，Octopus融合了第一人称视角和俯视图进行视觉理解，生成可执行的动作代码。通过OctoVerse套件训练，其数据集涵盖了如OmniGibson、Minecraft和GTA-V等丰富注释环境，涵盖常规及推理密集型任务。值得注意的是，Octopus通过环境反馈的强化学习实现了自适应规划与执行。其依赖视觉的功能在未知场景中实现了无缝任务泛化，凸显了其作为复杂GUI环境中具身代理统一模型的能力。

Wang et al., [367] present a comprehensive overview of LAMs, a new paradigm in AI designed to perform tangible actions in GUI environments, using UFO [19] at Windows OS as a case study platform. Built on the Mistral-7B [501] foundation, LAMs advance beyond traditional LLMs by integrating task planning with actionable outputs. Leveraging structured inputs from tools like the UI Automation (UIA) API, LAMs generate executable steps for dynamic planning and adaptive responses. A multi-phase training strategy—encompassing task-plan pretraining, imitation learning, self-boosting exploration, and reinforcement learning—ensures robustness and accuracy. Evaluations on real-world GUI tasks highlight LAMs' superior task success rates compared to standard models. This innovation establishes a foundation for intelligent GUI agents capable of transforming user requests into real-world actions, driving significant progress in productivity and automation.
Wang等人[367]全面综述了LAMs，这是一种旨在GUI环境中执行具体操作的新型AI范式，以Windows操作系统上的UFO [19]为案例平台。基于Mistral-7B [501]基础，LAMs超越传统LLM，整合任务规划与可执行输出。利用UI自动化（UIA）API等工具的结构化输入，LAMs生成动态规划和自适应响应的可执行步骤。多阶段训练策略涵盖任务规划预训练、模仿学习、自我增强探索和强化学习，确保了鲁棒性和准确性。真实GUI任务评测显示，LAMs的任务成功率优于标准模型。这一创新为智能GUI代理奠定了基础，能够将用户请求转化为现实操作，推动生产力和自动化的显著进步。

These developments in computer GUI agents highlight the integration of advanced visual comprehension, planning, and action execution, paving the way for more sophisticated and capable desktop agents.
这些计算机GUI代理的发展凸显了先进视觉理解、规划与动作执行的整合，为更复杂且功能强大的桌面代理铺平了道路。

16.11 8.6 Cross-Platform Large Action Models

16.12 8.6 跨平台大动作模型

To achieve versatility across various platforms, cross-platform LAMs have been developed, enabling GUI agents to operate seamlessly in multiple environments such as mobile devices, desktops, and web interfaces. We provide an analysis of LAMs tailored for cross-platform GUI agents in Table 30 and 31
为实现跨多平台的通用性，开发了跨平台LAM，使GUI代理能够在移动设备、桌面和网页界面等多种环境中无缝运行。表30和31中提供了针对跨平台GUI代理的LAM分析。

CogAgent [15] stands out as an advanced visual language model specializing in GUI understanding and navigation across PC, web, and Android platforms. Built on CogVLM [381], it incorporates a novel high-resolution cross-module to process GUI screenshots efficiently, enabling detailed comprehension of GUI elements and their spatial relationships. Excelling in tasks requiring OCR and GUI grounding, CogAgent achieves state-of-the-art performance on benchmarks like Mind2Web [212] and AITW [358]. Its ability to generate accurate action plans and interface with GUIs positions it as a pivotal step in developing intelligent agents optimized for GUI environments. CogAgent has further evolved into its beta version, GLM-PC [505], offering enhanced control capabilities.
CogAgent [15]作为一款先进的视觉语言模型，专注于PC、网页和Android平台的GUI理解与导航。基于CogVLM [381]，其引入了新颖的高分辨率跨模块以高效处理GUI截图，实现对GUI元素及其空间关系的细致理解。CogAgent在需要OCR和GUI定位的任务中表现卓越，在Mind2Web [212]和AITW [358]等基准测试中达到最先进水平。其生成准确动作计划并与GUI接口交互的能力，使其成为开发优化GUI环境智能代理的重要一步。CogAgent已进一步发展为测试版GLM-PC [505]，提供了增强的控制能力。

TABLE 30: An overview of GUI-optimized models on cross-platform agents (Part I).
表30：跨平台代理中针对GUI优化模型概览（第一部分）。

Model	Platform	Foundation Model	Size	Input	Output	Dataset	Highlights	Link
RUIG 405]	Mobile and desktop	Swin Transformer [525 and BART 88]	decoder layers	UI screenshots and text instructions	Bounding box predictions in linguistic form	MoTIF dataset and Ri-coSCA dataset for mobile UI data and Common Crawl for desktop UI data	Innovatively uses policy gradients to improve the spatial decoding in the pixel-to-sequence paradigm	/
CogAgen	PC, web, and Android platforms	CogVLM-17B 381]	18B	GUI screen-shots combined with OCR-derived text	Task plans, action sequences, and textual descriptions	CCS400K, text recognition datasets: 80M synthetic text images, visual grounding datasets and GUI dataset Mind2Web and AiTW	High-resolution cross-module to balance computational efficiency and high-resolution input processing	https://github com/THUDM/ CogVLM
SeeClick	iOS, Android, macOS, Windows, and web	Qwen-VL 210]	9.6B	GUI screenshots and textual instructions	GUI actions and element locations for interaction	300k webpages with text and icons, RICO, and data from LLaVA	Ability to perform GUI tasks purely from screenshots and its novel GUI grounding pre-training approach	https://github com/njucckevin/ SeeClick
ScreenAl 375]	Mobile, desktop, and tablet Uls	PaLI-3 [536]	5B	GUI screen-shots with OCR text, image captions, and other visual elements	Text-based answers questions, screen annotations with bounding box coordinates and labels. navigation instructions. summaries of screen content	262M mobile web screenshots and 54M mobile app screenshots	Unified representation of UIs and infographics, combining visual and textual elements	https://github com/kyegomez/ ScreenAl
V-Zen 408]	Computers and Web	Vicuna-7B 510], DINO 183],EVA-2-CLIP 537]	7B	Text, GUI Images	Action Prediction. GUI Bounding Box	GUIDE 370]	Dual-resolution visual encoding for precise GUI grounding and task executior	https: //github.com/ abdur75648/ V-Zen
Ferret-UI 406]	iPhone, Android, iPad, Web, Ap-pleTV	Vicuna- 13B 510], Gemma- 2B 502 Llama3-8B 91]	Vicuna-13B 510], Gemma-2B 502] Llama3-8B 91]	UI screenshots annotated bounding boxes and labels for UI widgets, OCR detected text and bounding boxes text elements. source HTML hierarchy trees for web data	Descriptions of UI elements, widget classification OCR, tapability and text/widget location. interaction instructions and multi-round interaction-based QA	Core-set, GroundUI-18k, GUIDE, Spotlight	Multi-platform support with high-resolution adaptive image encoding	/

ShowUI 238]	Websites, desktops, and mobile phones	Phi-3.5-Vision 234]	4.2B	GUI screenshots with OCR for text-based UI elements and visual grounding for icons and widgets	GUI actions. navigation element location	ScreenSpot, RICO, GUIEnv, GUIAct, AiTW, AiTZ, GUI-World	Interleaved Vision-Language Action approach, allowing seamless navigation, grounding, and understanding of GUI environments	https://github com/showlab ShowUI
OS-ATLAS 232]	Windows, macOS, Linux, Android, and the web	InternVL-2 379 and Qwen2-VL 210]	4B/7B	GUI screenshots	GUI actions	AndroidControl, SeeClick, and others annotated with GPT-4 over 13 million GUI elements and 2.3 million screenshots	The first foundation action model designed for generalist GUI agents, supporting cross-platform GUI tasks, and introducing a unified action space	https://osatlas.github.io/
xLAM 372	Diverse environments	Mistral-7B 501 and DeepSeek-Coder-7B 538	Range from 1B to 8×22B	Unified function-calling data formats	Function calls, thought processes	Synthetic and augmented data. including over 60,000 high-quality samples generated using APIGen from 3,673 APIs across 21 categories	Excels in function-calling tasks by leveraging unified and scalable data pipelines	https: //github.com/ SalesforceAIResear xLAM
SpiritSigh 384]	Web, Android, Windows Desktop	InternVL 379]	2B, 8B, and 26B	GUI screenshots	GUI actions	AitW [358], Common-Crawl websites, and custom annotations	Introduces a Universal Block Parsing (UBP) method to resolve positional ambiguity in high-resolution visual inputs.	https: //hzhiyuan. github.io/ SpiritSight-Agent

模型	平台	基础模型（Foundation Model）	规模	输入	输出	数据集	亮点	链接
RUIG 405]	移动端和桌面端	Swin Transformer [525 和 BART 88]	解码器层	用户界面截图和文本指令	以语言形式的边界框预测	用于移动UI数据的MoTIF数据集和RicoSCA数据集，以及用于桌面UI数据的Common Crawl	创新性地使用策略梯度提升像素到序列范式中的空间解码能力	/
CogAgen	PC、网页和安卓平台	CogVLM- 17B 381]	18B	结合OCR提取文本的GUI截图	任务计划、动作序列和文本描述	CCS400K，文本识别数据集：8000万合成文本图像，视觉定位数据集以及GUI数据集Mind2Web和AiTW	高分辨率跨模块设计，平衡计算效率与高分辨率输入处理	https://github com/THUDM/ CogVLM
SeeClick	iOS、安卓、macOS、Windows和网页	Qwen-VL 210]	9.6B	GUI截图和文本指令	GUI操作和交互元素位置	30万网页含文本和图标，RICO，以及来自LLaVA的数据	能够仅凭截图执行GUI任务及其新颖的GUI定位预训练方法	https://github com/njucckevin/ SeeClick
ScreenAI 375]	移动端、桌面端和平板UI	PaLI-3 [536]	5B	带OCR文本、图像说明及其他视觉元素的GUI截图	基于文本的问答，带边界框坐标和标签的屏幕注释，导航指令，屏幕内容摘要	2.62亿移动网页截图和5400万移动应用截图	统一表示UI和信息图，结合视觉与文本元素	https://github com/kyegomez/ ScreenAI
V-Zen 408]	计算机和网页	Vicuna-7B 510], DINO 183], EVA- 2-CLIP 537]	7B	文本，GUI图像	动作预测，GUI边界框	GUIDE 370]	双分辨率视觉编码，实现精确的GUI定位和任务执行	https: //github.com/ abdur75648/ V-Zen
Ferret-UI 406]	iPhone、安卓、iPad、网页、AppleTV	Vicuna- 13B 510], Gemma-2B 502 Llama3-8B 91]	Vicuna-13B 510], Gemma-2B 502] Llama3-8B 91]	带注释边界框和标签的UI截图，OCR检测文本及其边界框文本元素，网页数据的源HTML层级树	UI元素描述、小部件分类OCR、可点击性及文本/小部件位置，交互指令及多轮基于交互的问答	核心集、GroundUI-18k、GUIDE、Spotlight	多平台支持，具备高分辨率自适应图像编码	/
ShowUI 238]	网站、桌面和手机	Phi-3.5- Vision 234]	4.2B	带OCR文本的GUI截图，用于文本型UI元素及图标和小部件的视觉定位	GUI操作，导航元素位置	ScreenSpot、RICO、GUIEnv、GUIAct、AiTW、AiTZ、GUI-World	交织视觉-语言动作方法，实现GUI环境的无缝导航、定位和理解	https://github com/showlab ShowUI
OS-ATLAS 232]	Windows、macOS、Linux、Android 和网页	InternVL-2 379 和 Qwen2-VL 210]	4B/7B	图形用户界面截图	图形用户界面操作	AndroidControl、SeeClick 等，使用GPT-4 标注了超过1300万个图形用户界面元素和230万张截图	首个为通用图形用户界面代理设计的基础动作模型，支持跨平台图形用户界面任务，并引入统一动作空间	https://osatlas.github.io/
xLAM 372	多样化环境	Mistral-7B 501 和 DeepSeek-Coder-7B 538	范围从1B到8×22B	统一的函数调用数据格式	函数调用、思考过程	合成和增强数据，包括使用 APIGen 从21个类别的3673个API生成的超过6万条高质量样本	通过利用统一且可扩展的数据管道，在函数调用任务中表现出色	https: //github.com/ SalesforceAIResear xLAM

SpiritSigh 384]	网页、Android、Windows 桌面	InternVL 379]	2B、8B 和 26B	图形用户界面截图	图形用户界面操作	AitW [358]、Common-Crawl 网站和自定义标注	引入通用块解析（Universal Block Parsing，UBP）方法以解决高分辨率视觉输入中的位置歧义问题。	https: //hzhiyuan. github.io/ SpiritSight-Agent

Focusing on universal GUI understanding, Ferret-UI 2 406 from Apple is a state-of-the-art multimodal large language model designed to master UI comprehension across diverse platforms, including iPhones, Android devices, iPads, web, and AppleTV. By employing dynamic high-resolution image encoding, adaptive gridding, and high-quality multimodal training data generated through GPT-4, it outperforms its predecessor and other competing models in UI referring, grounding, and interaction tasks. Ferret-UI 2's advanced datasets and innovative training techniques ensure high accuracy in spatial understanding and user-centered interactions, setting a new benchmark for cross-platform UI adaptability and performance.
专注于通用图形用户界面（GUI）理解，Apple的Ferret-UI 2 406是一款最先进的多模态大型语言模型，旨在掌握包括iPhone、Android设备、iPad、网页和AppleTV在内的多平台UI理解。通过采用动态高分辨率图像编码、自适应网格划分以及由GPT-4生成的高质量多模态训练数据，其在UI指代、定位和交互任务中表现优于其前身及其他竞争模型。Ferret-UI 2的先进数据集和创新训练技术确保了空间理解和以用户为中心的交互的高准确性，树立了跨平台UI适应性和性能的新标杆。

TABLE 31: An overview of GUI-optimized models on cross-platform agents (Part II).
表31：跨平台代理上针对GUI优化模型的概述（第二部分）。

Model	Platform	Foundation Model	Size	Input	Output	Dataset	Highlights	Link
Falcon-UI 373	iOS, Android, Windows, Linux, Web	Qwen2-VL-7B	7B	Screenshots of GUI with node information and annotations for visible elements	GUI actions and coordinates or bounding boxes for interaction elements	Insight-UI dataset, further fine-tuned on datasets such as AITW, AITZ, Android Control, and Mind2Web	Decouples GUI context comprehension from instruction-following tasks, leveraging an instruction-free pretraining approach.	/
UI-TARS 407]	Web, Desktop (Windows, ma-cOS), and Mobile (Android)	Qwen-2-VL 7B and 72B 231]	7B 72B	GUI screen-shots	GUI actions	GUI screenshots and metadata collected from websites, apps, and operating systems; action trace datasets from various GUI agent benchmarks; 6M GUI tutorials for reasoning enhancement; multiple open-source datasets	Pure vision-based perception with standardized GUI actions across platforms (Web, Mobile, Desktop).	https://github.com/bytedance/UI-TARS
Magma 409]	Web, Mobile, Desktop, Robotics	LLaMA-3-8B [91], ConvNeXt-Xxlarge 539	8.6B	GUI screenshots, textual task descriptions	GUI actions, robotic manipulation	UI, robotics data, human instructional videos	Jointly trains on heterogeneous datasets, enabling generalization across digital and physical tasks	https: //microsoft. github.io/ Magma/
GUI-R1 410]	Windows, Linux, MacOS, Android, and Web	QwenVL2.5 540]	3B a 7B	GUI shots	Reasoning text and GUI actions	Mixture of 3K high-quality samples	first framework to apply rule-based reinforcement learning (RFT) to high-level GUI tasks across platforms.	https://github.com/ritzz-ai/GUI-R1.git
InfiGUI-R1 411]	Web, Desktop, and Android	Qwen2.5-VL-3B-Instruct	3B	GUI screenshots, Accessibility Tree	Reasoning text and GUI actions	Diverse dataset mixture	Two-stage training framework Actor2Reasoner: (1) Reasoning Injection via Spatial Reasoning Distillation, and (2) Deliberation Enhancement via Reinforcement Learning with Sub-goal Guidance and Error Recovery Scenario Construction	https: //github.com/ RealIm-Labs/ InfiGUI-R1

Task Generalization 389	Web and Android (Mobile)	Qwen2-VL-7B-Instruct 231]	7B	GUI screen-shots	Thoughts and grounded coordinate-based actions	11 domain datasets with 56K GUI trajectory samples	Introduces mid-training on diverse non-GUI reasoning tasks (particularly math and code) to substantially enhance GUI agent planning capabilities	https://github.com/hkust-nlp/GUIMid

模型	平台	基础模型（Foundation Model）	规模	输入	输出	数据集	亮点	链接
Falcon-UI 373	iOS、Android、Windows、Linux、Web	Qwen2-VL- 7B	7B	带有节点信息和可见元素注释的GUI截图	GUI操作及交互元素的坐标或边界框	Insight-UI数据集，进一步在AITW、AITZ、Android Control和Mind2Web等数据集上微调	将GUI上下文理解与指令执行任务解耦，采用无指令预训练方法。	/
UI-TARS 407]	Web、桌面（Windows、macOS）和移动端（Android）	Qwen-2-VL 7B和72B 231]	7B 72B	GUI截图	GUI操作	从网站、应用和操作系统收集的GUI截图及元数据；来自多个GUI代理基准的操作轨迹数据集；用于推理增强的600万GUI教程；多个开源数据集	基于纯视觉的感知，跨平台（Web、移动、桌面）标准化GUI操作。	https://github.com/bytedance/UI-TARS
Magma 409]	Web、移动端、桌面、机器人	LLaMA-3- 8B [91]，ConvNeXt-Xxlarge 539	8.6B	GUI截图，文本任务描述	GUI操作，机器人操控	UI、机器人数据、人类指导视频	联合训练异构数据集，实现数字与物理任务的泛化	https://microsoft.github.io/Magma/
GUI-R1 410]	Windows、Linux、MacOS、Android和Web	QwenVL2.5 540]	3B和7B	GUI截图	推理文本和GUI操作	3000个高质量样本混合	首个将基于规则的强化学习（RFT）应用于跨平台高级GUI任务的框架。	https://github.com/ritzz-ai/GUI-R1.git
InfiGUI-R1 411]	Web、桌面和Android	Qwen2.5- VL-3B-Instruct	3B	GUI截图，辅助功能树	推理文本和GUI操作	多样化数据集混合	两阶段训练框架Actor2Reasoner：（1）通过空间推理蒸馏注入推理，（2）通过带子目标引导和错误恢复场景构建的强化学习提升深思能力	https: //github.com/ RealIm-Labs/ InfiGUI-R1
任务泛化 389	Web和Android（移动端）	Qwen2-VL-7B-Instruct 231]	7B	GUI截图	思考与基于坐标的具体操作	11个领域数据集，含56000条GUI轨迹样本	引入多样非GUI推理任务（尤其是数学和代码）的中期训练，显著提升GUI代理的规划能力	https://github.com/hkust-nlp/GUIMid

Advancing GUI automation, ShowUI 238 introduces a pioneering Vision-Language-Action model that integrates high-resolution visual inputs with textual understanding to perform grounding, navigation, and task planning. Optimized for web, desktop, and mobile environments, ShowUI leverages the Phi- 3.5-vision-instruct backbone and comprehensive datasets to achieve robust results across benchmarks like ScreenSpot [25] and GUI-Odyssey [359]. Its ability to process multi-frame and dynamic visual inputs alongside JSON-structured output actions highlights its versatility. With innovations in interleaved image-text processing and function-calling capabilities, ShowUI sets a new standard for LLM-powered GUI agents.

推动GUI自动化，ShowUI 238引入了一种开创性的视觉-语言-动作（Vision-Language-Action）模型，该模型将高分辨率视觉输入与文本理解相结合，以实现定位、导航和任务规划。ShowUI针对网页、桌面和移动环境进行了优化，利用Phi-3.5-vision-instruct骨干网络和全面的数

据集，在ScreenSpot [25]和GUI-Odyssey [359]等基准测试中取得了稳健的成果。其处理多帧和动态视觉输入以及JSON结构化输出动作的能力凸显了其多功能性。通过交错图文处理和函数调用功能的创新，ShowUI为基于大型语言模型（LLM）的GUI代理树立了新标准。

Addressing the need for a unified action space, OS-ATLAS 232 introduces a foundational action model specifically designed for GUI agents across platforms like Windows, macOS, Linux, Android, and the web. By leveraging a massive multi-platform dataset and implementing a unified action space, OS-ATLAS achieves state-of-the-art performance in GUI grounding and out-of-distribution generalization tasks. Its scalable configurations adapt to varying computational needs while maintaining versatility in handling natural language instructions and GUI elements. As a powerful open-source alternative to commercial solutions, OS-ATLAS marks a significant step toward democratizing access to advanced GUI agents.
为解决统一动作空间的需求，OS-ATLAS 232引入了一种基础动作模型，专为Windows、macOS、Linux、Android及网页等多平台GUI代理设计。通过利用庞大的多平台数据集并实现统一动作空间，OS-ATLAS在GUI定位和分布外泛化任务中达到了最先进的性能。其可扩展配置适应不同计算需求，同时保持处理自然语言指令和GUI元素的多样性。作为商业解决方案的强大开源替代品，OS-ATLAS标志着向普及先进GUI代理迈出了重要一步。

Magma [409] is a foundation model for multimodal AI agents that integrates LLMs with vision and action understanding to complete UI navigation and robotic manipulation tasks. Unlike previous models optimized for either UI automation or robotics, Magma jointly trains on a heterogeneous dataset (about 39M samples) spanning UI screenshots, web navigation, robot trajectories, and instructional videos. It employs SoM and Trace-of-Mark techniques, which enhance action grounding and prediction by labeling actionable elements in GUI environments and tracking motion traces in robotic tasks.
Magma [409]是一种多模态AI代理的基础模型，将大型语言模型（LLM）与视觉和动作理解相结合，以完成UI导航和机器人操作任务。不同于以往专注于UI自动化或机器人领域的模型，Magma在一个异构数据集（约3900万样本）上联合训练，涵盖UI截图、网页导航、机器人轨迹和教学视频。它采用了SoM和Trace-of-Mark技术，通过标注GUI环境中的可操作元素和追踪机器人任务中的运动轨迹，增强了动作定位和预测能力。

UI-TARS [407] is an advanced, vision-based Large Action Model (LAM) optimized for multi-platform GUI agents. Unlike traditional approaches, it relies solely on GUI screenshots for perception, eliminating the need for structured representations. By incorporating a unified action space, UI-TARS enables seamless execution across Web, Windows, macOS, and Android environments. Built on Qwen-2-VL, it is trained on 6 million GUI tutorials, large-scale screenshot datasets, and multiple open-source benchmarks. A key innovation of UI-TARS is its System-2 reasoning capability, which allows it to generate explicit reasoning steps before executing actions, enhancing decision-making in dynamic environments. Additionally, it employs an iterative self-improvement framework, refining its performance through reflection-based learning. Experimental results demonstrate that UI-TARS outperforms existing models, including GPT-40 and Claude, in task execution benchmarks.
UI-TARS [407]是一款先进的基于视觉的大动作模型（LAM），针对多平台GUI代理进行了优化。不同于传统方法，它仅依赖GUI截图进行感知，免去了对结构化表示的需求。通过引入统一动作空间，UI-TARS实现了在网页、Windows、macOS和Android环境中的无缝执行。基于Qwen-2-VL构建，训练数据包括600万GUI教程、大规模截图数据集及多个开源基准。UI-TARS的一大创新是其系统2推理能力，能够在执行动作前生成明确的推理步骤，提升动态环境中的决策质量。此外，它采用迭代自我改进框架，通过反思学习不断优化性能。实验结果表明，UI-TARS在任务执行基准测试中优于包括GPT-40和Claude在内的现有模型。

These cross-platform LAMs demonstrate the potential of unified models that can adapt to diverse environments, enhancing the scalability and applicability of GUI agents in various contexts.
这些跨平台的大动作模型展示了统一模型适应多样环境的潜力，提升了GUI代理在各种场景中的可扩展性和适用性。

16.13 8.7 Takeaways

16.14 8.7 关键总结

The exploration of LAMs for GUI agents has revealed several key insights that are shaping the future of intelligent interaction with graphical user interfaces:
对GUI代理大动作模型的探索揭示了若干关键见解，这些见解正在塑造图形用户界面智能交互的未来：

1. Smaller Models for On-Device Inference: Many of the optimized LAMs are built from smaller foundational models, often ranging from 1 billion to 7 billion parameters. This reduction in model size enhances computational efficiency, making it feasible to deploy these models on resource-constrained devices such as mobile phones and edge devices. The ability to perform on-device inference without relying on cloud services addresses privacy concerns and reduces latency, leading to a more responsive user experience.
2. 用于设备端推理的小型模型：许多优化后的大动作模型基于较小的基础模型，参数规模通常在10亿至70亿之间。模型规模的缩减提升了计算效率，使得在资源受限的设备如手机和边缘设备上部署成为可能。无需依赖云服务即可进行设备端推理，解决了隐私问题并降低了延迟，带来更流畅的用户体验。

2. Enhanced GUI Comprehension Reduces Reliance on Structured Data: Models like VGA 354 and OmniParser [184] emphasize the importance of visual grounding and image-centric fine-tuning to reduce dependency on structured UI metadata. By improving GUI comprehension directly from visual inputs, agents become more adaptable to different software environments, including those where structured data may be inaccessible or inconsistent.

3. 增强的GUI理解减少对结构化数据的依赖：如VGA 354和OmniParser [184]等模型强调视觉定位和以图像为中心的微调的重要性，以降低对结构化UI元数据的依赖。通过直接从视觉输入提升GUI理解，代理能够更好地适应不同软件环境，包括那些结构化数据不可用或不一致的场景。

3. Reinforcement Learning Bridges Static and Dynamic Environments: The application of reinforcement learning in models like DigiRL [264] demonstrates the effectiveness of bridging static training data with dynamic real-world environments. This approach allows agents to learn from interactions, recover from errors, and adapt to changes, enhancing their robustness and reliability in practical applications.
4. 强化学习桥接静态与动态环境：如DigiRL [264]等模型中强化学习的应用展示了将静态训练数据与动态现实环境相结合的有效性。这种方法使代理能够通过交互学习、从错误中恢复并适应变化，增强了其实用中的鲁棒性和可靠性。

4. Unified Function-Calling Enhances Interoperability: Efforts to standardize data formats and function-calling mechanisms, as seen in models like xLAM [372], facilitate multi-turn interactions and reasoning across different platforms. This unification addresses compatibility issues and enhances the agent's ability to perform complex tasks involving multiple APIs and services.
5. 统一函数调用提升互操作性：如xLAM [372]等模型中对数据格式和函数调用机制的标准化努力，促进了跨平台的多轮交互和推理。这种统一解决了兼容性问题，增强了代理执行涉及多个API和服务的复杂任务的能力。

5. Inference-Time Computing and Reasoning Models: Recent work highlights the importance of inference-time computing, where models plan, reason, and decompose tasks on the fly without architectural changes. Techniques such as extended context windows and chain-of-thought prompting (e.g., "o1-style" reasoning) enable more robust, long-horizon decision-making. UI-R1 [401], GUI-R1 [410] and InfiGUI-R1 [411] are pioneering efforts in this direction. There is also growing interest in rule-based rewards and cost functions that guide inference-time behavior, integrating explicit heuristics to improve the stability, interpretability, and generalization of GUI agents.
5）推理时计算与推理模型：近期研究强调了推理时计算的重要性，即模型在不改变架构的情况下，能够即时规划、推理和分解任务。诸如扩展上下文窗口和链式思维提示（例如“o1风格”推理）等技术，使得长远决策更加稳健。UI-R1 [401]、GUI-R1 [410] 和 InfiGUI-R1 [411] 是该方向的开创性工作。基于规则的奖励和成本函数也日益受到关注，这些函数引导推理时行为，整合显式启发式方法以提升GUI代理的稳定性、可解释性和泛化能力。

The advancements in LAMs for GUI agents highlight a trend toward specialized, efficient, and adaptable models capable of performing complex tasks across various platforms. By focusing on specialization, multimodal integration, and innovative training methodologies, researchers are overcoming the limitations of general-purpose LLMs. These insights pave the way for more intelligent, responsive, and user-friendly GUI agents that can transform interactions with software applications.
GUI代理中基于大模型（LAMs）的进展凸显了向专用、高效且适应性强的模型发展的趋势，这些模型能够跨多平台执行复杂任务。通过聚焦专业化、多模态整合和创新训练方法，研究者们正在克服通用大型语言模型（LLMs）的局限性。这些洞见为更智能、响应迅速且用户友好的GUI代理铺平了道路，能够革新软件应用的交互方式。

17 9 EVALUATION FOR LLM-BRAINED GUI AGENTS

18 9 基于大型语言模型的GUI代理评估

In the domain of GUI agents, evaluation is crucial for enhancing both functionality and user experience [56], [58] and should be conducted across multiple aspects. By systematically assessing these agents' effectiveness across various tasks, evaluation tonly gauges their performance in different dimensions but also provides a framework for their continuous improvement [541]. Furthermore, it encourages in vation by identifying areas for potential development, ensuring that GUI agents evolve in tandem with advancements in LLMs and align with user expectations.
在GUI代理领域，评估对于提升功能性和用户体验至关重要[56]，[58]，应涵盖多个方面。通过系统地评估这些代理在不同任务中的有效性，评估不仅衡量其在各维度的表现，还为其持续改进提供框架[541]。此外，评估通过识别潜在发展领域，促进创新，确保GUI代理与大型语言模型的进步同步演进，并符合用户期望。

As illustrated in Figure 28 when a GUI agent completes a task, it produces an action sequence, captures screenshots, extracts UI structures, and logs the resulting environment states. These outputs serve as the foundation for evaluating the agent's performance through various metrics and measurements across diverse platforms. In the subsequent sections, we delve into these evaluation methodologies, discussing the metrics and measurements used to assess GUI agents comprehensively. We also provide an overview of existing benchmarks tailored for GUI agents across different platforms, highlighting their key features and the challenges they address.
如图28所示，当GUI代理完成任务时，会生成动作序列、截取屏幕截图、提取UI结构并记录环境状态。这些输出构成了通过多种指标和测量方法评估代理性能的基础。接下来的章节中，我们将深入探讨这些评估方法，讨论用于全面评估GUI代理的指标和测量手段。同时，我们还将概述针对不同平台GUI代理的现有基准测试，突出其关键特性及所解决的挑战。

18.1 9.1 Evaluation Metrics

18.2 9.1 评估指标

Evaluating GUI agents requires robust and multidimensional metrics to assess their performance across various dimensions, including accuracy, efficiency, and compliance (e.g., safety). In a typical benchmarking setup, the GUI agent is provided with a natural language instruction as input and is expected to auto mously execute actions until the task is completed. During this process, various assets can be collected, such as the sequence of actions taken by the agent, step-wise observations (e.g., DOM or HTML structures), screenshots, runtime logs, final states, and execution time. These assets enable evaluators to determine whether the task has been completed successfully and to analyze the agent's performance. In this section, we summarize the key evaluation metrics commonly used for benchmarking GUI agents. te that different research works may use different names for these metrics, but with similar calculations. We align their names in this section.

评估GUI代理需要稳健且多维的指标，以衡量其在准确性、效率和合规性（如安全性）等多个维度的表现。在典型的基准测试中，GUI代理接收自然语言指令作为输入，需自主执行动作直至任务完成。过程中可收集多种数据，如代理执行的动作序列、逐步观察（如DOM或HTML结构）、屏幕截图、运行日志、最终状态及执行时间。这些数据使评估者能够判断任务是否成功完成，并分析代理表现。本节总结了常用于GUI代理基准测试的关键评估指标。需注意，不同研究可能对这些指标命名不同，但计算方法相似。我们在本节统一命名。

1. Step Success Rate: Completing a task may require multiple steps. This metric measures the ratio of the number of steps that are successful over the total steps within a task. A high step success rate indicates precise and accurate execution of granular steps, which is essential for the reliable performance of tasks involving multiple steps [212], [349], [358].
   1）步骤成功率：完成任务可能需要多个步骤。该指标衡量任务中成功步骤数与总步骤数的比率。高步骤成功率表明对细粒度步骤的精准执行，这对于多步骤任务的可靠完成至关重要[212]，[349]，[358]。

2. Turn Success Rate: A turn indicates a single interaction between the user and the agent. A turn may consist of multiple steps, and completing a task may consist of multiple turns. This metric measures the ratio of turns that successfully address the request in that interaction over all turns. It focuses on the agent's ability to understand and fulfill user expectations during interactive or dialog-based tasks, ensuring the agent's responsiveness and reliability across iterative interactions, particularly in tasks requiring dynamic user-agent communication [155], 348].
   2）回合成功率：回合指用户与代理之间的一次交互。一个回合可能包含多个步骤，完成任务可能涉及多个回合。该指标衡量成功满足该交互请求的回合数与总回合数的比率。它关注代理在交互或对话任务中理解并满足用户期望的能力，确保代理在迭代交互中的响应性和可靠性，尤其适用于需要动态用户-代理沟通的任务[155]，[348]。

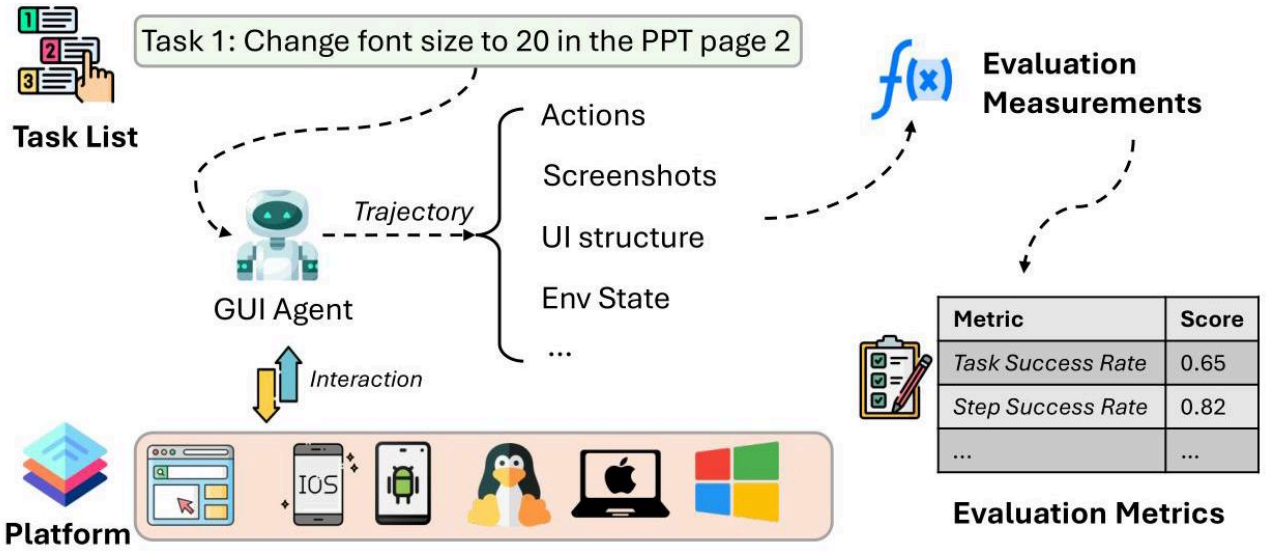

Fig. 28: An illustrative example of evaluation of task completion by a GUI agent.
图28：GUI代理任务完成评估的示例说明。

3. Task Success Rate: Task success rate measures the successful task completion over all tasks set in the benchmark. It evaluates whether the final task completion state is achieved while ig ring the intermediate steps. This metric provides an overall measure of end-to-end task completion, reflecting the agent's ability to handle complex workflows holistically [419], [425], [438].
   3）任务成功率：任务成功率衡量基准中所有任务的成功完成比例。它评估最终任务完成状态是否达成，而忽略中间步骤。该指标提供端到端任务完成的整体衡量，反映代理整体处理复杂工作流的能力[419]，[425]，[438]。

4. Efficiency Score: Efficiency score evaluates how effectively the agent completes tasks while considering resource consumption,

execution time, or total steps the agent might take. This metric can be broken down into the following sub-metrics:
4）效率得分：效率得分评估代理在完成任务时的资源消耗、执行时间或总步骤数等方面的有效性。该指标可细分为以下子指标：

- Time Cost: Measures the time taken to complete tasks.
- 时间成本：衡量完成任务所用时间。
- Resource Cost: Measures the memory/CPU/GPU usage to complete tasks.
- 资源成本：衡量完成任务所用的内存/CPU/GPU资源。
- LLM Cost: Evaluates the computational or monetary cost of LLM calls used during task execution.
- 大模型成本：评估任务执行过程中调用大型语言模型（LLM）的计算或金钱成本。
- Step Cost: Measures the total steps required to complete tasks.
- 步骤成本：衡量完成任务所需的总步骤数。

Depending on the specific metrics used, the efficiency score can be interpreted differently in different papers [442], [444].
根据所使用的具体指标，不同论文中效率得分的解释可能有所不同[442]，[444]。

5. Completion under Policy: This metric measures the rate at which tasks are completed successfully while adhering to policy constraints. It ensures that the agent complies with user-defined or organizational rules, such as security, ethical, safety, privacy, or business guidelines, during task execution. This metric is particularly relevant for applications where compliance is as critical as task success 416 .
   5）遵守策略的完成率：该指标衡量在遵守策略约束的前提下任务成功完成的比率。它确保代理在执行任务时遵守用户定义或组织规定的规则，如安全、伦理、安全性、隐私或业务指南。该指标对于合规性与任务成功同等重要的应用尤为关键[416]。

6. Risk Ratio: Similar to the previous metric, the risk ratio evaluates the potential risk associated with the agent's actions during task execution. It identifies vulnerabilities, errors, or security concerns that could arise during task handling. A lower risk ratio indicates higher trustworthiness and reliability, while a higher ratio may suggest areas needing improvement to minimize risks and enhance robustness [416].
   6）风险比率：与前一指标类似，风险比率评估代理在执行任务过程中可能带来的潜在风险。它识别任务处理过程中可能出现的漏洞、错误或安全隐患。较低的风险比率表明更高的可信度和可靠性，而较高的比率则可能提示需要改进的方面，以降低风险并增强稳健性[416]。

The implementation of metrics in each GUI agent benchmark might vary depending on the platform and the task formulation. In all tables in this section, we mapped the original metrics used in the benchmarks, which may possess different names, to the categories that we defined above.
每个GUI代理基准测试中指标的实现可能因平台和任务定义而异。在本节所有表格中，我们将基准测试中使用的原始指标（可能名称不同）映射到上述定义的类别。

18.3 9.2 Evaluation Measurements

18.4 9.2 评估测量

To effectively evaluate GUI agents, various measurement techniques are employed to assess their accuracy and alignment with expected outputs. These measurements validate different aspects of agent performance, ranging from textual and visual correctness to interaction accuracy and system state awareness, using code, models, and even agents as evaluators [26]. Below, we summarize key measurement approaches used in benchmarking GUI agents. Based on these measurements, the evaluation metrics defined beforehand can be calculated accordingly.
为了有效评估GUI代理，采用多种测量技术来评估其准确性及与预期输出的一致性。这些测量验证代理性能的不同方面，涵盖文本和视觉的正确性、交互准确性及系统状态感知，评估者可使用代码、模型甚至其他代理[26]。以下总结了用于GUI代理基准测试的关键测量方法。基于这些测量，可相应计算预先定义的评估指标。

1. Text Match: This measurement evaluates whether the text-based outputs of the agent match the expected results. For example, whether a target product name is reached when the agent is browsing an e-commerce website. It can involve different levels of strictness, including:
   1）文本匹配：该测量评估代理的文本输出是否与预期结果相符。例如，代理在浏览电商网站时是否达到了目标产品名称。文本匹配可包含不同严格度级别，包括：

- Exact Match: Ensures the output perfectly matches the expected result.
- 精确匹配：确保输出与预期结果完全一致。
- Partial or Fuzzy Match: Allows for approximate matches, which are useful for handling mi r variations such as typos or sy nyms.
- 部分或模糊匹配：允许近似匹配，适用于处理拼写错误或同义词等变体。
- Semantic Similarity: Measures deeper alignment in semantic meaning using techniques like cosine similarity of text embeddings or

other semantic similarity measures.

- 语义相似度：利用文本嵌入的余弦相似度或其他语义相似度度量，衡量更深层次的语义对齐。

Text Match is widely applied in tasks involving textual selections, data entry, or natural language responses.
文本匹配广泛应用于涉及文本选择、数据输入或自然语言响应的任务中。

2. Image Match: Image Match focuses on validating whether the agent acts or stops on the expected page (e.g., webpage, app UI), or selects the right image. It involves comparing screenshots, selected graphical elements, or visual outcomes against ground truth images using image similarity metrics or visual question answering (VQA) methods. This measurement is particularly crucial for tasks requiring precise visual identification.
   2）图像匹配：图像匹配侧重验证代理是否在预期页面（如网页、应用UI）执行操作或停止，或是否选择了正确的图像。它通过图像相似度指标或视觉问答（VQA）方法，将截图、选定的图形元素或视觉结果与真实图像进行比较。该测量对需要精确视觉识别的任务尤为重要。

3. Element Match: This measurement checks whether specific widget elements (e.g., those in HTML, DOM, or application UI hierarchies) interacted with by the agent align with the expected elements. These may include:
   3）元素匹配：该测量检查代理交互的特定控件元素（如HTML、DOM或应用UI层级中的元素）是否与预期元素一致。这些元素可能包括：

- HTML Tags and Attributes: Ensuring the agent identifies and interacts with the correct structural elements.
- HTML标签和属性：确保代理识别并交互正确的结构元素。

- URLs and Links: Validating navigation-related elements.
- URL和链接：验证导航相关元素。

- DOM Hierarchies: Confirming alignment with expected DOM structures in dynamic or complex web interfaces.
- DOM层级结构：确认与动态或复杂网页界面中预期DOM结构的一致性。

- UI Controls and Widgets: Verifying interactions with platform-specific controls such as buttons, sliders, checkboxes, dropdown menus, or other GUI components in desktop and mobile applications.
- 用户界面控件和组件：验证与平台特定控件（如按钮、滑块、复选框、下拉菜单或其他桌面和移动应用中的GUI组件）的交互。

- Accessibility Identifiers: Utilizing accessibility identifiers or resource IDs in mobile platforms like Android and iOS to ensure correct element selection.
- 辅助功能标识符：利用移动平台（如Android和iOS）中的辅助功能标识符或资源ID，确保正确选择元素。

- View Hierarchies: Assessing alignment with expected view hierarchies in mobile applications, similar to DOM hierarchies in web applications.
- 视图层级结构：评估移动应用中视图层级结构是否符合预期，类似于网页应用中的DOM层级。

- System Controls and APIs: Ensuring correct interaction with operating system controls or APIs, such as file dialogs, system menus, or tifications in desktop environments.
- 系统控件和API：确保与操作系统控件或API（如文件对话框、系统菜单或桌面环境中的通知）的正确交互。

Element Match ensures robust interaction with user interface components across different platforms during task execution.
元素匹配确保在任务执行过程中跨不同平台与用户界面组件的稳健交互。

4. Action Match: This measurement assesses the accuracy of the agent's actions, such as clicks, scrolls, or keystrokes, by comparing them against an expected sequence. It involves:
5. 动作匹配：该指标评估代理执行动作（如点击、滚动或按键）的准确性，通过与预期序列进行比较。其内容包括：

- Action Accuracy: Validates that each action (including action type and its arguments) is performed correctly (e.g., clicking the correct button, typing the right input).
- 动作准确性：验证每个动作（包括动作类型及其参数）是否正确执行（例如，点击正确的按钮，输入正确的内容）。

- Action Sequence Alignment: Ensures actions occur in the correct order to meet task requirements.
- 动作序列对齐：确保动作按正确顺序发生，以满足任务要求。

- Location Prediction: Checks that spatial actions, such as mouse clicks or touch gestures, target the intended regions of the interface.
- 位置预测：检查空间动作（如鼠标点击或触摸手势）是否针对界面预期区域。

Action Match is vital for evaluating step-wise correctness in task completion.
动作匹配对于评估任务完成过程中的逐步正确性至关重要。

5. State Information: State Information captures runtime data related to the system's environment during task execution. It provides

insights into contextual factors that may influence the agent's behavior, such as:

6. 状态信息：状态信息捕获任务执行期间系统环境的运行时数据，提供可能影响代理行为的上下文因素洞见，例如：

- Application State: Information about the state of the application being interacted with (e.g., open files, active windows, saved files in given locations).
- 应用状态：关于所交互应用状态的信息（例如，打开的文件、活动窗口、特定位置保存的文件）。
- System Logs: Detailed logs recording the agent's decisions and interactions.
- 系统日志：记录代理决策和交互的详细日志。
- Environment Variables: Contextual data about the operating system or runtime environment.
- 环境变量：关于操作系统或运行时环境的上下文数据。

This measurement is valuable for debugging, performance analysis, and ensuring reliability under diverse conditions.
该指标对于调试、性能分析及确保在多样条件下的可靠性具有重要价值。

Each of these measurement techniques contributes to a comprehensive evaluation framework, ensuring that the agent t only completes tasks but does so with precision, efficiency, and adaptability. Together, they help build trust in the agent's ability to perform reliably in real-world scenarios while maintaining compliance with policy constraints.
这些测量技术共同构建了一个全面的评估框架，确保代理t不仅完成任务，而且以精准、高效和适应性强的方式执行。它们共同助力建立对代理在现实场景中可靠执行能力的信任，同时保持对策略约束的遵守。

18.5 9.3 Evaluation Platforms

18.6 9.3 评估平台

Evaluating GUI agents requires diverse platforms to capture the varying environments in which these agents operate. The platforms span web, mobile, and desktop environments, each with unique characteristics, challenges, and tools for evaluation. This section summarizes the key aspects of these platforms and their role in benchmarking GUI agents.
评估GUI代理需要多样化的平台，以涵盖这些代理所运行的不同环境。平台涵盖网页、移动和桌面环境，每种环境具有独特的特性、挑战和评估工具。本节总结了这些平台的关键方面及其在GUI代理基准测试中的作用。

1. Web: Web platforms are among the most common environments for GUI agents, reflecting their prevalence in everyday tasks such as browsing, form filling, and data scraping. Key characteristics of web platforms for evaluation include:
2. 网页：网页平台是GUI代理最常见的环境之一，反映了它们在浏览、填写表单和数据抓取等日常任务中的广泛应用。网页平台评估的主要特征包括：

- Dynamic Content: Web applications often involve dynamic elements generated through JavaScript, AJAX, or similar tech logies, requiring agents to handle asynchro us updates effectively.
- 动态内容：网页应用通常包含通过JavaScript、AJAX或类似技术生成的动态元素，要求代理能够有效处理异步更新。
- Diverse Frameworks: The variety of web tech logies (e.g., HTML, CSS, JavaScript frameworks) demands robust agents capable of interacting with a range of interface designs and structures.
- 多样化框架：各种网页技术（如HTML、CSS、JavaScript框架）要求代理具备与多种界面设计和结构交互的能力。
- Tools and Libraries: Evaluation often uses tools such as Selenium, Puppeteer, or Playwright to emulate browser interactions, collect runtime information, and compare outcomes against expected results.
- 工具和库：评估通常使用Selenium、Puppeteer或Playwright等工具来模拟浏览器交互、收集运行时信息，并将结果与预期进行比较。
- Accessibility Compliance: Metrics like WCAG (Web Content Accessibility Guidelines) adherence can also be evaluated to ensure inclusivity.
- 无障碍合规性：还可以评估如WCAG（Web内容无障碍指南）等指标，以确保包容性。

2. Mobile: Mobile platforms, particularly Android and iOS, pose unique challenges for GUI agents due to their constrained interfaces and touch-based interactions. Evaluating agents on mobile platforms involves:
3. 移动：移动平台，尤其是Android和iOS，由于其受限的界面和基于触摸的交互，为GUI代理带来了独特挑战。移动平台上的代理评估包括：

- Screen Size Constraints: Agents must adapt to limited screen real estate, ensuring interactions remain accurate and efficient.
- 屏幕尺寸限制：代理必须适应有限的屏幕空间，确保交互的准确性和高效性。
- Touch Gestures: Evaluating the agent's ability to simulate gestures such as taps, swipes, and pinches is essential.
- 触摸手势：评估代理模拟点击、滑动和捏合等手势的能力至关重要。
- Platform Diversity: Android devices vary significantly in terms of screen sizes, resolutions, and system versions, while iOS offers more standardized conditions.

- 平台多样性：Android设备在屏幕尺寸、分辨率和系统版本上差异显著，而iOS则提供更为标准化的条件。
- Evaluation Tools: Tools like Appium and Espresso (for Android) or XCTest (for iOS) and emulators are commonly used for testing and evaluation.
- 评估工具：常用的测试和评估工具包括Appium和Espresso（针对Android）或XCTest（针对iOS）及模拟器。

3. Desktop: Desktop platforms provide a richer and more complex environment for GUI agents, spanning multiple operating systems such as Windows, macOS, and Linux. Evaluations on desktop platforms often emphasize:
4. 桌面：桌面平台为GUI代理提供了更丰富、更复杂的环境，涵盖Windows、macOS和Linux等多种操作系统。桌面平台的评估通常强调：

- Application Diversity: Agents must handle a wide range of desktop applications, including productivity tools, web browsers, and custom enterprise software.
- 应用多样性：代理必须处理各种桌面应用，包括生产力工具、网页浏览器和定制企业软件。
- Interaction Complexity: Desktop interfaces often include advanced features such as keyboard shortcuts, drag-and-drop, and context menus, which agents must handle correctly.
- 交互复杂性：桌面界面通常包含键盘快捷键、拖放和上下文菜单等高级功能，代理必须正确处理这些功能。
- Cross-Platform Compatibility: Evaluations may involve ensuring agents can operate across multiple operating systems and versions.
- 跨平台兼容性：评估可能涉及确保代理能在多个操作系统及其不同版本上运行。
- Automation Frameworks: Tools such as Windows UI Automation, macOS Accessibility APIs, and Linux's AT-SPI are used to automate and monitor agent interactions.
- 自动化框架：使用如Windows UI自动化、macOS辅助功能API和Linux的AT-SPI等工具来自动化和监控代理交互。
- Resource Usage: Memory and CPU usage are significant metrics, particularly for long-running tasks or resource-intensive applications.
- 资源使用：内存和CPU使用率是重要指标，尤其针对长时间运行的任务或资源密集型应用。

Each platform presents distinct challenges and opportunities for evaluating GUI agents. Web platforms emphasize scalability and dynamic interactions, mobile platforms focus on touch interfaces and performance, and desktop platforms require handling complex workflows and cross-application tasks. Some benchmarks are cross-platform, requiring agents to be robust, adaptable, and capable of generalizing across different environments.
每个平台在评估GUI代理时都面临独特的挑战和机遇。网页平台强调可扩展性和动态交互，移动平台侧重触控界面和性能，桌面平台则需处理复杂工作流程和跨应用任务。一些基准测试是跨平台的，要求代理具备强健性、适应性及跨环境泛化能力。

All the metrics, measurements, and platforms discussed are essential for a comprehensive evaluation of GUI agents across multiple aspects. Most existing benchmarks rely on them for evaluation. In what follows, detail these benchmarks for GUI agents selectively.
上述所有指标、测量方法和平台对于全面评估GUI代理的多方面性能至关重要。大多数现有基准测试依赖这些指标进行评估。以下内容将有选择地详细介绍这些GUI代理基准测试。

18.7 9.4 Web Agent Benchmarks

18.8 9.4 网页代理基准测试

Evaluating GUI agents in web environments necessitates benchmarks that capture the complexities and nuances of web-based tasks. Over the years, several benchmarks have been developed, each contributing unique perspectives and challenges to advance the field. We first provide an overview of these benchmarks in Tables 32 33 34 and 35
在网页环境中评估GUI代理需要能够捕捉网页任务复杂性和细微差别的基准测试。多年来，开发了若干基准测试，各自为推动该领域发展贡献了独特视角和挑战。我们首先在表32、33、34和35中概述这些基准测试。

One of the pioneering efforts in this domain is Mini-WoB++ [145], [146], focusing on assessing reinforcement learning agents on web-based GUI tasks. It introduces realistic interaction scenarios, including clicking, typing, and navigating web elements, and leverages workflow-guided exploration (WGE) to improve efficiency in environments with sparse rewards. Agents are evaluated based on success rates, determined by their ability to achieve final goal states, highlighting adaptability and robustness across various complexities.
该领域的先驱之一是Mini-WoB++ [145]，[146]，专注于评估基于网页GUI任务的强化学习代理。它引入了真实的交互场景，包括点击、输入和网页元素导航，并利用工作流引导探索（Workflow-Guided Exploration, WGE）提升在稀疏奖励环境中的效率。代理通过成功率评估，即其达成最终目标状态的能力，突出其在不同复杂度下的适应性和鲁棒性。

Building upon the need for more realistic environments, Mind2Web [212] represents a significant advancement by enabling agents to handle real-world HTML environments rather than simplified simulations. Established after the advent of LLMs [157], it offers a large dataset of over 2,000 tasks spanning multiple domains, presenting challenges from basic actions to complex multi-page workflows. The benchmark emphasizes end-to-end task performance through metrics like Element Accuracy and Task Success Rate, encouraging rigorous evaluation of agents.
基于对更真实环境的需求，Mind2Web [212]实现了重大进展，使代理能够处理真实HTML环境，而非简化模拟。该基准在大型语言模型

（LLMs）[157]出现后建立，提供了涵盖多个领域的2000多个任务数据集，挑战涵盖基础操作到复杂多页工作流程。该基准通过元素准确率和任务成功率等指标强调端到端任务表现，鼓励对代理进行严格评估。

Extending Mind2Web's capabilities, MT-Mind2Web 414 introduces conversational web navigation, requiring sophisticated interactions that span multiple turns with both users and the environment. This advanced benchmark includes 720 web navigation conversation sessions with 3,525 instruction and action sequence pairs, averaging five user-agent interactions per session, thereby testing agents' conversational abilities and adaptability.
扩展Mind2Web功能的MT-Mind2Web 414引入了对话式网页导航，要求代理与用户及环境进行多轮复杂交互。该高级基准包含720个网页导航对话会话，包含3525对指令与动作序列，平均每会话五次用户-代理交互，测试代理的对话能力和适应性。

To further enhance realism, WebArena [412] sets a new standard with its realistic web environment that mimics genuine human interactions. Featuring 812 tasks across multiple domains, it requires agents to perform complex, long-horizon interactions over multi-tab web interfaces. By focusing on functional correctness rather than surface-level matches, WebArena promotes thorough assessment of agents' practical abilities.
为进一步提升真实性，WebArena [412]以其模拟真实人类交互的网页环境树立了新标准。涵盖812个跨领域任务，要求代理在多标签网页界面上执行复杂的长时交互。WebArena侧重功能正确性而非表面匹配，促进对代理实际能力的深入评估。

Recognizing the importance of multimodal capabilities, VisualWebArena, an extension of WebArena [412], was designed to assess agents on realistic visually grounded web tasks. Comprising 910 diverse tasks in domains like Clas-sifieds, Shopping, and Reddit, it adds new visual functions for measuring open-ended tasks such as visual question answering and fuzzy image matching, thereby challenging agents in multimodal understanding.
鉴于多模态能力的重要性，VisualWebArena作为WebArena [412]的扩展，设计用于评估代理在真实视觉基础网页任务中的表现。包含910个多样化任务，涉及分类广告、购物和Reddit等领域，新增视觉功能用于测量开放式任务，如视觉问答和模糊图像匹配，挑战代理的多模态理解能力。

Similarly, VideoWebArena [421] focuses on evaluating agents' abilities to comprehend and interact with video content on the web. It presents 74 videos across 2,021 tasks, challenging agents in video-based information retrieval, contextual reasoning, and skill application. This benchmark highlights critical deficiencies in current models, emphasizing the need for advancements in agentic reasoning and video comprehension.
类似地，VideoWebArena [421]聚焦评估代理理解和交互网页视频内容的能力。它提供74个视频，涵盖2021个任务，挑战代理在基于视频的信息检索、上下文推理和技能应用方面的表现。该基准揭示当前模型的关键不足，强调代理推理和视频理解能力的提升需求。

Complementing this, VisualWebBench [213] offers a multimodal benchmark that assesses understanding, OCR, grounding, and reasoning across website, element, and action levels. Spanning 1.5K samples from real-world websites, it identifies challenges such as poor grounding and subpar OCR with low-resolution inputs, providing a crucial evaluation perspective distinct from general multimodal benchmarks.
作为补充，VisualWebBench [213]提供了一个多模态基准，评估网站、元素和动作层面的理解、光学字符识别（OCR）、定位和推理能力。涵盖1500个来自真实网站的样本，揭示了如定位不准确和低分辨率输入导致的OCR性能不佳等挑战，提供了与通用多模态基准不同的重要评估视角。

Beyond the challenges of multimodality, understanding agents' resilience to environmental distractions is crucial. EnvDistraction 422 introduces a benchmark that evaluates the faithfulness of multimodal GUI agents under n-malicious distractions, such as pop-ups and recommendations. The study demonstrates that even advanced agents are prone to such distractions, revealing vulnerabilities that necessitate robust multimodal perception for reliable automation.
除了多模态挑战，理解代理对环境干扰的抗干扰能力也至关重要。EnvDistraction 422引入了一个基准，评估多模态GUI代理在恶意干扰（如弹窗和推荐）下的忠实度。研究表明，即使是先进代理也易受此类干扰，暴露出需要强健多模态感知以实现可靠自动化的脆弱性。

Focusing on safety and trustworthiness, ST-WebAgentBench 416 takes a unique approach by emphasizing the management of unsafe behaviors in enterprise settings. It features a human-in-the-loop system and a policy-driven hierarchy, introducing the Completion under Policy (CuP) metric to evaluate agents' compliance with organizational, user, and task-specific policies. This benchmark operates in web environments using BrowserGym 426] and includes 235 tasks with policies addressing various safety dimensions, providing a comprehensive framework for evaluating agents in enterprise scenarios.
聚焦安全性和可信度，ST-WebAgentBench 416采用独特方法，强调企业环境中不安全行为的管理。它包含人机交互系统和基于策略的层级结构，引入“策略下完成度”（Completion under Policy, CuP）指标，评估代理对组织、用户及任务特定策略的遵守情况。该基准在使用BrowserGym [426]的网页环境中运行，包含235个任务，涵盖多种安全维度的策略，为企业场景下代理评估提供了全面框架。

Addressing the automation of enterprise software tasks, WorkArena 420 offers a benchmark emphasizing tasks commonly performed within the Service w platform. With 19,912 unique instances across 33 tasks, it highlights the significant performance gap between current state-of-the-art agents and human capabilities in enterprise UI automation, setting a trajectory for future in vation.
针对企业软件任务的自动化，WorkArena 420 提供了一个基准，重点关注在 Service w 平台上常见的任务。该基准包含 33 个任务中的 19,912 个独特实例，突显了当前最先进代理与人类在企业用户界面自动化能力之间的显著性能差距，为未来创新指明了方向。

TABLE 32: Overview of web GUI agent benchmarks (Part I).
表 32：网页图形用户界面代理基准概览（第一部分）。

Benchmark	Platform	Year	Live	Highlight	Data Size	Metric	Measurement	Link
MiniWoB++ 145 146]	Web	2017	Yes	Evaluates agents on basic web interactions like clicking, typing, and form navigation.	100 web interaction tasks	Task Rate Success	Element Match	https://github.com/Farama%2DFoundation/ miniwob%2Dplusplus
142]	Web	2021	No	Uses ThingTalk for mapping natural language to web actions, enabling precise web-based task execution in real HTML environments.	741 instructions	Task Success Rate	Text Match, Element Match	https: //github.com/ xnancy/russ
WebShop	Web	2022	Yes	Simulates e-commerce navigation with real-world products, challenging agents with instruction comprehension, multi-page navigation, and strategic exploration.	12,087 instructions	Task Success Rate, Step Success Rate"	Text Match	https: //webshop-pnlp.github.io/
Mind2Web 21	Web	2023	No	Tests adaptability on real-world, dynamic websites across domains.	2,000 tasks	Step Success Rate, Task Success Rate	Element Match, Action Match	https: //github.com/ OSU-NLP-Group/ Mind2Web
Mind2Web-Live 349	Web	2024	Yes	Provides intermediate action tracking for realistic task assessment, along with an updated Mind2Web-Live dataset and tools for an tation.	542 tasks	Step Success Rate, Task Success Rate, Efficiency Score	Element Match, Text Match, trajectory length	https: //huggingface.co/datasets/ iMeanAI/ Mind2Web-Live
Mind2Web-Live-Abstracted 278	Web	2024	Yes	Abstract the descriptions by omitting task-specific details and user input information in Mind2Web-Live, which are more streamlined and less time-consuming to compose.	104 samples	Task Success Rate, Efficiency Score	Text Match, Image Match, Element Match, Path Length	https: //anonymous.4open.science/ r/navigate
WebArena 412	Web	2023	Yes	Simulates realistic, multi-tab browsing on Docker-hosted websites, focusing on complex, long-horizon tasks that mirror real online interactions.	812 long-horizon tasks	Step Success Rate	Text Match	https: //webarena.dev/
VisualWebArena 413]	Web	2024	Yes	Assesses multimodal agents on visually grounded tasks, requiring both visual and textual interaction capabilities in web environments.	910 tasks	Step Success Rate	Text Match, Image Match	https: //jykoh.com/vwa
MT-Mind2Web 414	Web	2024	No	Introduces conversational web navigation with multiturn interactions, supported by a specialized multi-turn web dataset.	720 sessions/3525 instructions	Step Success Rate, Turn Success Rate	Element Match, Action Match	https://github.com/magicgh/self-map

基准测试	平台	年份	在线	亮点	数据规模	指标	测量方式	链接
MiniWoB++ 145 146]	网页	2017	是	评估代理在基本网页交互上的表现，如点击、输入和表单导航。	100个网页交互任务	任务成功率	元素匹配	https://github.com/Farama% 2DFoundation/ miniwob% 2Dplusplus
142]	网页	2021	否	使用ThingTalk将自然语言映射到网页操作，实现真实HTML环境中精确的网页任务执行。	741条指令	任务成功率	文本匹配，元素匹配	https: //github.com/ xnancy/russ
WebShop	网页	2022	是	模拟电商导航，包含真实商品，考验代理的指令理解、多页导航和策略探索能力。	12,087条指令	任务成功率，步骤成功率"	文本匹配	https: //webshop-pnlp. github.io/
Mind2Web 21	网页	2023	否	测试代理在跨领域动态真实网站上的适应能力。	2,000个任务	步骤成功率，任务成功率	元素匹配，动作匹配	https: //github.com/ OSU-NLP-Group/ Mind2Web
Mind2Web-Live 349	网页	2024	是	提供中间动作跟踪以实现真实任务评估，并附带更新的Mind2Web-Live数据集及注释工具。	542个任务	步骤成功率，任务成功率，效率评分	元素匹配，文本匹配，轨迹长度	https: //huggingface. co/datasets/ iMeanAI/ Mind2Web-Live
Mind2Web-Live-抽象版 278	网页	2024	是	通过省略任务细节和用户输入信息对Mind2Web-Live描述进行抽象，更加简洁且节省编写时间。	104个样本	任务成功率，效率评分	文本匹配，图像匹配，元素匹配，路径长度	https: //anonymous. 4open.science/ r/navigate
WebArena 412	网页	2023	是	模拟Docker托管网站上的真实多标签浏览，聚焦复杂且长时程任务，反映真实在线交互。	812个长时程任务	步骤成功率	文本匹配	https: //webarena.dev/
VisualWebArena 413]	网页	2024	是	评估多模态代理在视觉基础任务上的表现，要求具备网页环境中的视觉和文本交互能力。	910个任务	步骤成功率	文本匹配，图像匹配	https: //jykoh.com/vwa
MT-Mind2Web 414	网页	2024	否	引入多轮对话式网页导航，支持多轮交互，配备专门的多轮网页数据集。	720个会话/3525条指令	步骤成功率，回合成功率	元素匹配，动作匹配	https://github.com/magicgh/ self-map

BrowserGym [426] builds ecosystem designed for web agent research. It unifies various benchmarks like Mini-WoB(++) [146], WebArena [412], and WorkArena [420] under a single framework, addressing the issue of fragmentation in web agent evaluation. By leveraging standardized observation and action spaces, it enables consistent and reproducible experiments. BrowserGym's extensible architecture make it a vital tool for developing and testing GUI-driven agents powered by LLMs, significantly accelerating in vation in web automation research.
BrowserGym [426] 构建了一个专为网页代理研究设计的生态系统。它将 Mini-WoB(++) [146]、WebArena [412] 和 WorkArena [420] 等多个基准统一到单一框架下，解决了网页代理评估中的碎片化问题。通过利用标准化的观察和动作空间，实现了一致且可复现的实验。BrowserGym 的可扩展架构使其成为开发和测试由大型语言模型（LLMs）驱动的图形用户界面（GUI）代理的重要工具，显著加速了网页自动化研究中的创新。

In the realm of interacting with live websites, WebOlym-pus 424 introduces an open platform that enables web agents to interact with live websites through a Chrome extension-based interface. Supporting diverse tasks and integrating a safety monitor to prevent harmful actions, it promotes safer automation of web-based tasks and provides a critical tool for evaluating agent performance in realistic scenarios.
在与实时网站交互领域，WebOlympus 424 引入了一个开放平台，使网页代理能够通过基于 Chrome 扩展的界面与实时网站交互。该平台支持多样化任务，并集成了安全监控器以防止有害操作，促进了网页任务的更安全自动化，并为在真实场景中评估代理性能提供了关键工具。

Collectively, these benchmarks have significantly contributed to advancing the evaluation of web-based GUI agents, each addressing different aspects such as realism, multimodality, safety, and enterprise applicability. Their developments reflect the evolving challenges and requirements in creating sophisticated agents capable of complex web interactions.
这些基准共同显著推动了基于网页的 GUI 代理评估的发展，各自针对现实性、多模态、安全性和企业适用性等不同方面。它们的发展反映了创建能够进行复杂网页交互的高级代理所面临的不断演变的挑战和需求。

18.9 9.5 Mobile Agent Benchmarks

18.10 9.5 移动代理基准

Evaluating GUI agents on mobile platforms presents unique challenges due to the diversity of interactions and the complexity of mobile applications. Several benchmarks have been developed to address these challenges, each contributing to the advancement of mobile agent evaluation. We first provide an analysis for these mobile benchmarks in Tables 36 and 37
在移动平台上评估 GUI 代理面临独特挑战，因交互多样性和移动应用复杂性。为应对这些挑战，开发了多个基准，每个都推动了移动代理

评估的进步。我们首先在表36和表37中对这些移动基准进行分析。

An early effort in this domain is PIXELHELP [144], which focuses on grounding natural language instructions to actions on mobile user interfaces. Addressing the significant challenge of interpreting and executing complex, multi-step tasks, PIXELHELP provides a comprehensive dataset pairing English instructions with human-performed actions on a mobile UI emulator. It comprises 187 multi-step instructions across four task categories, offering a robust resource for evaluating models on task accuracy through metrics like Complete Match and Partial Match.
该领域的早期工作之一是 PIXELHELP [144]，其重点是将自然语言指令映射到底层移动用户界面操作。针对解释和执行复杂多步骤任务的重大挑战，PIXELHELP 提供了一个全面的数据集，将英文指令与人在移动 UI 模拟器上的操作配对。该数据集包含187条跨四个任务类别的多步骤指令，为通过“完全匹配”和“部分匹配”等指标评估模型的任务准确性提供了坚实资源。

TABLE 33: Overview of web GUI agent benchmarks (Part II).
表33：网页 GUI 代理基准概览（第二部分）。

Benchmark	Platform	Year	Live	Highlight	Data Size	Metric	Measurement	Link
MMInA 415	Web	2024	Yes	Tests multihop, multimodal tasks on real-world websites, requiring agents to handle cross-page information extraction and reasoning for complex tasks.	1,050 tasks	Step Success Rate, Task Success Rate	Text Match, Element Match	https://mmina.cliangyu.com/
AutoWebBench 270]	Web	2024	No	Bilingual web browsing benchmark with 10,000 browsing traces, supporting evaluation across language-specific environments.	10,000 traces	Step Success Rate, Efficiency Score	Element Match, Action Match, Time	https://github.com/THUDM/AutoWebGLM
WorkArena 420]	Web	2024	Yes	Focuses on real-world enterprise software interactions, targeting tasks frequently performed by knowledge workers	19,912 unique task instances	Task Success Rate, Efficiency Score, Completion under Policy, Turn Success Rate	Element Match, Text Match, Execution-based Validation	https: //github.com/ ServiceNow/ WorkArena
VideoWebArena 421]	Web	2024	Yes	Focuses on long-context multimodal agents using video tutorials for task completion	74 videos amounting to approximately 4 hours, with 2,021 tasks in total	Task Success Rate. Intermediate Intent Success Rate, Efficiency Scores	Element Match, State Information, Exact and Fuzzy Text Matches	https://github.com/jiang0/videowebarena
EnvDistraction 422]	Web	2024	No	Evaluates the "faithfulness" of multimodal GUI agents by assessing their susceptibility to environmental distractions, such as pop-ups, fake search results, or misleading recommendations	1,198 tasks	Task Success Rate	Text Match, Element Match. State Information	https://github.com/xbmxb/EnvDistraction
WebVLN-v1 347]	Web	2024	No	Combines navigation and question-answering on shopping sites, integrating visual and textual content for unified web interaction evaluation.	8,990 paths and 14,825 QA pairs	Task Success Rate, Efficiency Score	Element Match, Path Length, Trajectory Length	https://github.com/WebVLN/WebVLN
WEBLINX 348	Web	2024	No	Focuses on conversational navigation, requiring agents to follow multi-turn user instructions in realistic, dialogue-based web tasks.	100k interactions	Turn Success Rate	Element Match, Text Match, Action Match	https: //mcgill-nlp. github.io/weblinx/

ST-WebAgentBench 416]	Web	2024	Yes	Evaluates policy-driven safety in web agents, using the Completion under Policy metric to ensure compliance in enterprise-like environments.	235 tasks	Task Success Rate, Completion under Policy (CuP), Risk Ratio	Element Match, Action Match. Text Match	https: //sites.google. com/view/ st-webagentbench home
CompWoB 417]	Web	2023	No	Tests agents on sequential, compositional tasks that require state management across multiple steps, simulating real-world automation scenarios.	50 compositional tasks	Task Success Rate	Element Match	https: //github.com/ google-research google-research tree/master/ compositional rl/compwob

基准测试	平台	年份	实时	亮点	数据规模	指标	测量方式	链接
MMInA 415	网页	2024	是	测试多跳、多模态任务在真实网站上的表现，要求代理能够处理跨页面的信息提取和复杂任务的推理。	1,050 个任务	步骤成功率，任务成功率	文本匹配，元素匹配	https://mmina.cliangyu.com/
AutoWebBench 270]	网页	2024	否	双语网页浏览基准，包含10,000条浏览轨迹，支持跨语言环境的评估。	10,000 条轨迹	步骤成功率，效率得分	元素匹配，动作匹配，时间	https://github.com/THUDM/ AutoWebGLM
WorkArena 420]	网页	2024	是	聚焦真实企业软件交互，针对知识工作者常执行的任务。	19,912 个独特任务实例	任务成功率，效率得分，策略完成率，回合成功率	元素匹配，文本匹配，基于执行的验证	https: //github.com/ ServiceNow/ WorkArena
VideoWebArena 421]	网页	2024	是	聚焦使用视频教程完成任务的长上下文多模态代理。	74 个视频，总时长约4小时，共2,021个任务	任务成功率，中间意图成功率，效率得分	元素匹配，状态信息，精确及模糊文本匹配	https://github.com/jiang0/ videowebarena
EnvDistraction 422]	网页	2024	否	通过评估多模态GUI代理对环境干扰（如弹窗、虚假搜索结果或误导性推荐）的敏感性，衡量其“忠实度”。	1,198 个任务	任务成功率	文本匹配，元素匹配，状态信息	https://github.com/xbmxb/ EnvDistraction
WebVLN-v1 347]	网页	2024	否	结合购物网站上的导航与问答，整合视觉和文本内容，进行统一的网页交互评估。	8,990 条路径和14,825 对问答对	任务成功率，效率得分	元素匹配，路径长度，轨迹长度	https://github.com/WebVLN/ WebVLN
WEBLINX 348	网页	2024	否	聚焦对话式导航，要求代理在真实对话式网页任务中遵循多轮用户指令。	10万次交互	回合成功率	元素匹配，文本匹配，动作匹配	https: //mcgill-nlp. github.io/ weblinx/
ST-WebAgentBench 416]	网页	2024	是	评估基于策略的网页代理安全性，使用策略完成率（Completion under Policy）指标确保企业环境中的合规性。	235 个任务	任务成功率，策略完成率（CuP），风险比	元素匹配，动作匹配，文本匹配	https: //sites.google. com/view/ st-webagentbench home
CompWoB 417]	网页	2023	否	测试代理在需要跨多步骤状态管理的顺序组合任务中的表现，模拟真实自动化场景。	50 个组合任务	任务成功率	元素匹配	https://github.com/google-research/google-research/tree/master/compositional_rl/compwob

Building upon the need for systematic evaluation, AN-DROIDLAB [363] establishes a comprehensive framework for Android-based auto mous agents. It introduces both an action space and operational modes that support consistent evaluations for text-only and multimodal models. By providing XML and SoM operation modes, ANDROIDLAB allows LLMs and LMMs to simulate real-world interactions in equivalent environments. The benchmark includes 138 tasks across nine apps, encompassing typical Android functionalities, and evaluates agents using metrics such as Success Rate and Reversed Redundancy.

基于系统评估的需求，AN-DROIDLAB [363] 建立了一个针对基于安卓的自动化代理的综合框架。它引入了动作空间和操作模式，支持对纯文本和多模态模型进行一致性评估。通过提供 XML 和 SoM 操作模式，ANDROIDLAB 使大型语言模型（LLMs）和大型多模态模型（LMMs）能够在等效环境中模拟真实交互。该基准包含涵盖典型安卓功能的九个应用中的138个任务，并使用成功率和逆冗余等指标评估代理表现。

To further challenge agents in handling both API and UI operations, Mobile-Bench [442] offers an in vative approach by combining these elements within a realistic Android environment. Its multi-app setup and three distinct task categories test agents' capabilities in handling simple and complex mobile interactions, pushing beyond traditional single-app scenarios. The evaluation leverages CheckPoint metrics, assessing agents at each key action step, providing insights into planning and decision-making skills.
为了进一步挑战代理在处理 API 和 UI 操作方面的能力，Mobile-Bench [442] 通过在真实安卓环境中结合这些元素，提供了一种创新方法。其多应用设置和三类不同任务测试代理处理简单和复杂移动交互的能力，突破了传统单应用场景。评估利用 CheckPoint 指标，在每个关键动作步骤对代理进行评估，提供对规划和决策能力的洞察。

Emphasizing safety in mobile device control, MobileSafe-tyBench 443 provides a structured evaluation framework that prioritizes both helpfulness and safety. It rigorously tests agents across common mobile tasks within an Android emulator, focusing on layered risk assessment, including legal compliance and privacy. A distinctive feature is its indirect prompt injection test to probe agent robustness. The evaluation ensures agents are scored on practical success while managing risks, advancing research in LLM reliability and secure auto mous device control.
MobileSafetyBench 443 强调移动设备控制的安全性，提供了一个结构化的评估框架，优先考虑实用性和安全性。它在安卓模拟器中严格测试代理在常见移动任务中的表现，重点关注分层风险评估，包括法律合规和隐私保护。其独特之处在于通过间接提示注入测试代理的鲁棒性。评估确保代理在实际成功的同时管理风险，推动大型语言模型可靠性和安全自动移动设备控制的研究进展。

Expanding the scope to multiple languages and application scenarios, SPA-BENCH 444 introduces an extensive benchmark for smartphone agents. It assesses both single-app and cross-app tasks in a plug-and-play framework that supports seamless agent integration. With a diverse task collection across Android apps, including system and third-party apps, SPA-BENCH offers a realistic testing environment measuring agent capabilities in understanding UIs and handling app navigation through metrics like success rate, efficiency, and resource usage.
SPA-BENCH 444 扩展到多语言和多应用场景，推出了一个针对智能手机代理的广泛基准。它评估单应用和跨应用任务，采用即插即用框架支持无缝集成代理。通过涵盖系统和第三方安卓应用的多样任务集合，SPA-BENCH 提供了一个真实的测试环境，衡量代理理解用户界面和处理应用导航的能力，指标包括成功率、效率和资源使用。

TABLE 34: Overview of web GUI agent benchmarks (Part III).
表34：网页图形用户界面代理基准概览（第三部分）。

Benchmark	Platform	Year	Live	Highlight	Data Size	Metric	Measurement	Link
TURKING BENCH 418]	Web	2024	Yes	Uses natural HTML tasks from crowdsourcing to assess interaction skills with real-world web layouts and elements.	32.2K instances	ask Success Rate	Text Match, Element Match, Image Match	https: //turkingbench. github.io
VisualWebBench 213]	Web	2024	No	Provides a fine-grained assessment of multimodal large language models (MLLMs) on web-specific tasks	1,534 instances from 139 real websites across 87 subdomains	Task Success Rate, Turn Success Rate, Efficiency Metrics	Text Match, Image Match, Element Match, Action Match	https:// visualwebbench. github.io/
WONDERBREAD 423]	Web	2024	No	Focuses on business process management (BPM) tasks like documentation, knowledge transfer, and process improvement	2,928 human demonstrations across 598 distinct workflows	Task Success Rate, Step Success Rate. Efficiency Score, Completion under Policy	Text Match, Action Match, State Information	https: //github.com/ HazyResearch/ wonderbread
WebOlympus 424]	Web	2024	Yes	An open platform for web agents that simplifies running demos, evaluations, and data collection for web agents on live websites	50 tasks	Task Success e. Step Success Rate	Action Match	/
BrowserGym 426]	Web	2024	Yes	Provides a unified, extensible, and open-source environment for evaluating web agents with standardized APIs and observations.	Benchmarks include Mini-WoB(++) with 125 tasks, WebArena with 812 tasks, and WorkArena with up to 341 tasks per level.	Task Success Rate, Step Success Rate, Turn Success Rate, Efficiency Metrics.	Text-based matching and element match.	https: //github.com/ ServiceNow/ BrowserGym
WebWalkerQA 427]	Web	2025	Yes	Benchmarks the capacity of LLMs to handle deep. structured, and realistic web-based navigation and reasoning tasks.	680 high-quality QA pairs.	Task Success Rate, Efficiency Score.	Text Match, Action Match.	https: //github.com/ Alibaba-NLP/ WebWalker
WebGames 428	Web	2025	Yes	A comprehensive benchmark designed to evaluate the capabilities of general-purpose web-browsing AI agents through 50+ interactive challenges. It uniquely provides a hermetic testing environment with verifiable ground-truth solutions.	50+ challenges	Task \$k Success Rate	Action Match	https: //github.com/ convergence-ai/ webgames

基准测试	平台	年份	在线	亮点	数据规模	指标	测量方式	链接
TURKING BENCH 418]	网页	2024	是	利用众包的自然HTML任务评估与真实网页布局和元素的交互能力。	32.2K 条实例	任务成功率	文本匹配，元素匹配，图像匹配	https: //turkingbench. github.io
VisualWebBench 213]	网页	2024	否	对多模态大语言模型（MLLMs）在网页特定任务上的细粒度评估	来自139个真实网站、87个子域的1,534条实例	任务成功率，回合成功率，效率指标	文本匹配，图像匹配，元素匹配，动作匹配	https:// visualwebbench. github.io/
WONDERBREAD 423]	网页	2024	否	聚焦于业务流程管理（BPM）任务，如文档编制、知识传递和流程改进	涵盖598个不同工作流的2,928个人工示范	任务成功率，步骤成功率，效率评分，政策下完成度	文本匹配，动作匹配，状态信息	https: //github.com/ HazyResearch/ wonderbread
WebOlympus 424]	网页	2024	是	一个开放平台，简化了在真实网站上运行网页代理的演示、评估和数据收集	50个任务	任务成功率，步骤成功率	动作匹配	/
BrowserGym 426]	网页	2024	是	提供统一、可扩展且开源的环境，通过标准化API和观察接口评估网页代理。	基准包括Mini-WoB(++)的125个任务，WebArena的812个任务，以及WorkArena每级最多341个任务。	任务成功率，步骤成功率，回合成功率，效率指标。	基于文本的匹配和元素匹配。	https: //github.com/ ServiceNow/ BrowserGym
WebWalkerQA 427]	网页	2025	是	评测大语言模型（LLMs）处理深度、结构化且真实网页导航与推理任务的能力。	680对高质量问答对。	任务成功率，效率评分。	文本匹配，动作匹配。	https: //github.com/ Alibaba-NLP/ WebWalker
WebGames 428	网页	2025	是	一个综合基准，设计用于通过50多个交互式挑战评估通用网页浏览AI代理的能力。它独特地提供了一个封闭测试环境，具备可验证的标准答案。	50多个挑战	任务成功率	动作匹配	https: //github.com/ convergence-ai/ webgames

Focusing on efficient and user-friendly evaluation, Mo-bileAgentBench 446 presents a benchmark tailored for agents on Android devices. It offers a fully auto mous testing process, leveraging final UI state matching and real-time app event tracking. With 100 tasks across 10 open-source Android applications categorized by difficulty, it accommodates multiple paths to success, enhancing reliability and applicability. Comprehensive metrics, including task success rate, efficiency, latency, and token cost, provide insights into agent performance.

MobileAgentBench 446专注于高效且用户友好的评估，提出了一个针对Android设备上代理的基准测试。它提供了全自动测试流程，利用最终UI状态匹配和实时应用事件跟踪。涵盖10个开源Android应用中的100个任务，按难度分类，支持多路径成功，提升了可靠性和适用性。全面的指标包括任务成功率、效率、延迟和令牌成本，提供了对代理性能的深入洞察。

Complementing these efforts, LlamaTouch [445] introduces a benchmark and testbed for mobile UI task automation in real-world Android environments. Emphasizing essential state an tation, it enables precise evaluation of tasks regardless of execution path variability or dynamic UI elements. With 496 tasks spanning 57 unique applications, LlamaTouch demonstrates scalability and fidelity through advanced matching techniques, integrating pixel-level screenshots and textual screen hierarchies, reducing false negatives and supporting diverse task complexities.

作为补充，LlamaTouch [445]引入了一个面向真实Android环境中移动UI任务自动化的基准和测试平台。强调关键状态注释，使得无论执行路径变化或动态UI元素如何，都能实现精确的任务评估。包含57个独特应用中的496个任务，LlamaTouch通过先进的匹配技术展示了其可扩展性和准确性，结合像素级截图和文本屏幕层级，减少了误判，支持多样化任务复杂度。

Zhao et al., introduce GTArena [447], a formalized framework and benchmark designed to advance auto mous GUI testing agents. GTArena provides a standardized evaluation environment tailored for multimodal large language models. Central to its design is the vel Transition Tuple data structure, which systematically captures and analyzes GUI defects. The benchmark assesses three core tasks—test intention generation, task execution, and defect detection—using a diverse dataset comprising real-world, artificially injected, and synthetic defects, establishing GTArena as a pioneering benchmark for GUI testing agents.

赵等人提出了GTArena [447]，一个规范化的框架和基准，旨在推动自动GUI测试代理的发展。GTArena提供了一个为多模态大型语言模型量身定制的标准化评估环境。其设计核心是状态转换元组（Transition Tuple）数据结构，系统地捕捉和分析GUI缺陷。该基准评估三大核心任务——测试意图生成、任务执行和缺陷检测，使用包含真实、人工注入及合成缺陷的多样化数据集，确立了GTArena作为GUI测试代理的开创性基准。

Collectively, these benchmarks have significantly advanced the evaluation of mobile-based GUI agents, addressing challenges in task complexity, safety, efficiency, and scalability. Their contributions are instrumental in developing more capable and reliable agents for mobile platforms.
总体而言，这些基准显著推动了基于移动端GUI代理的评估，解决了任务复杂性、安全性、效率和可扩展性等挑战。它们的贡献对于开发更强大且可靠的移动平台代理具有重要意义。

TABLE 35: Overview of web GUI agent benchmarks (Part IV).
表35：网页GUI代理基准概览（第四部分）。

Benchmark	Platform	Year	Live	Highlight	Data Size	Metric	Measurement	Link
SafeArena 429]	Web	2025	Yes	The first benchmark specifically designed to evaluate the deliberate misuse of web agents by testing their ability to complete both safe and harmful tasks.	500 tasks	Task Success Rate, Completion under Policy, Risk Ratio	Text Match, State Information	https: //safearena. github.io
WABER [430]	Web	2025	Yes	Introduces two new evaluation metrics—Efficiency and Reliability—that go beyond standard success rate measurements	655 tasks	Task Success Rate, Efficiency Score	Action Match, State Information	https: //github.com/ SumanKNath/ WABER
Online-Mind2Web 431]	Web	2025	Yes	A real-world online evaluation benchmark designed to reflect actual user interactions with live web interfaces	300 tasks from 136 websites	Task Success Rate, Efficiency Score	Image Match, Action Match, State Information, LLM-as-a-Judge Evaluation	https: //github.com/ OSU-NLP-Group/ Online-Mind2Web
AgentDAM 432]	Web	2025	Yes	The first benchmark to evaluate privacy leakage risks in multimodal, realistic web environments using agentic models	246 human-annotated test cases	Task Success Rate, Risk Ratio	Action Match, Text Match	https: //github.com/ facebookresearch/ ai-agent-privacy
AgentRewardBenchWeb 433		2025	No	The first benchmark to rigorously evaluate LLM-based judges against human expert annotations across multiple web agent tasks	1,302 trajectories, 351 tasks	Task Success Rate, Completion under Policy	Image Match, Element/State Match	https:// agent-reward-bench github.io
RealWebAssist 434	Web	2025	No	The first benchmark to evaluate long-horizon web assistance using real-world users' sequential instructions expressed in natural and often ambiguous language	1,885 instructions	Task Success Rate, Step Success Rate. Efficiency Score	Action Match	https: //scai.cs.jhu. edu/projects/ RealWebAssist/
435]	Web	2025	Yes	Fully deterministic, high-fidelity replicas of real-world websites (e.g., Airbnb, Amazon, Gmail), enabling safe, reproducible, and configurable testing for multi-turn GUI-based agents	112 tasks across 11 deterministic websites	Task Success Rate	Text Match, Action Match, State Information Match	https: //github.com/ agi-inc/agisdk
BEARCUBS 436	Web	2025	Yes	Emphasizes interaction with live web pages and includes multimodal tasks (e.g., video. audio, 3D) that cannot be solved by text-only methods, addressing limitations of prior benchmarks relying on static or simulated environments	111 questions	Task Success Rate, Efficiency Score	Text Match, Action Match	https: //bear-cubs. github.io

WASP 437]	Web	2025	Yes	The first end-to-end benchmark for evaluating web agents' security under realistic prompt injection attacks, simulating attacker capabilities in live sandboxed web environments	84 test cases	Task Success Rate. Completion under Policy, Risk Ratio	Action Match, State Information	/

基准测试	平台	年份	在线	亮点	数据规模	指标	测量方式	链接
SafeArena 429]	网页	2025	是	首个专门设计用于评估网络代理恶意滥用的基准，通过测试其完成安全和有害任务的能力。	500个任务	任务成功率、策略下完成率、风险比	文本匹配、状态信息	https: //safearena. github.io
WABER [430]	网页	2025	是	引入了两个新的评估指标——效率和可靠性，超越了标准成功率的测量	655个任务	任务成功率、效率得分	动作匹配、状态信息	https: //github.com/ SumanKNath/ WABER
Online-Mind2Web 431]	网页	2025	是	一个反映真实用户与在线网页界面交互的真实世界在线评估基准	来自136个网站的300个任务	任务成功率、效率得分	图像匹配、动作匹配、状态信息、以大语言模型(LLM)作为评判	https: //github.com/ OSU-NLP-Group/ Online-Mind2Web
AgentDAM 432]	网页	2025	是	首个使用代理模型评估多模态现实网络环境中隐私泄露风险的基准	246个人工标注测试用例	任务成功率、风险比	动作匹配、文本匹配	https: //github.com/ facebookresearch/ ai-agent-privacy
AgentRewardBenchWeb 433		2025	否	首个严格评估基于大语言模型(LLM)的评判与人类专家注释在多种网络代理任务中的表现的基准	1,302条轨迹，351个任务	任务成功率、策略下完成率	图像匹配、元素/状态匹配	https:// agent-reward-bench github.io
RealWebAssist 434	网页	2025	否	首个使用真实用户以自然且常含歧义语言表达的连续指令，评估长时程网页辅助的基准	1,885条指令	任务成功率、步骤成功率、效率得分	动作匹配	https: //scai.cs.jhu. edu/projects/ RealWebAssist/
435]	网页	2025	是	完全确定性、高保真度的真实网站（如Airbnb、亚马逊、Gmail）复制品，支持多轮基于GUI的代理的安全、可复现和可配置测试	11个确定性网站上的112个任务	任务成功率	文本匹配、动作匹配、状态信息匹配	https: //github.com/ agi-inc/agisdk
BEARCUBS 436	网页	2025	是	强调与实时网页的交互，包含多模态任务（如视频、音频、3D），解决了以往依赖静态或模拟环境的基准无法解决的问题	111个问题	任务成功率、效率得分	文本匹配、动作匹配	https: //bear-cubs. github.io
WASP 437]	网页	2025	是	首个端到端基准，用于评估网络代理在真实提示注入攻击下的安全性，模拟攻击者在沙箱网页环境中的能力	84个测试用例	任务成功率、策略下完成率、风险比	动作匹配、状态信息	/

18.11 9.6 Computer Agent Benchmarks

18.12 9.6 计算机代理基准测试

Evaluating GUI agents on desktop computers involves diverse applications and complex workflows. Several benchmarks have been developed to assess agents' capabilities in these environments, each addressing specific challenges and advancing the field. We overview benchmarks for computer GUI agents in Table 38

在桌面计算机上评估图形用户界面（GUI）代理涉及多样的应用程序和复杂的工作流程。已经开发了若干基准测试来评估这些环境中代理的能力，每个基准针对特定挑战并推动该领域的发展。我们在表38中概述了计算机GUI代理的基准测试。

An early benchmark in this domain is Act2Cap [327], which focuses on capturing and narrating GUI actions in video formats using a cursor as a pivotal visual guide. Act2Cap emphasizes the detailed nuances of GUI interactions, particularly cursor-based actions like clicks and drags, essential for advancing automation capabilities in GUI-intensive tasks. It includes a substantial dataset of 4,189 samples across various Windows GUI environments, employing metrics based on element-wise Intersection over Union to evaluate semantic accuracy and temporal and spatial precision.

该领域的早期基准之一是Act2Cap [327]，其重点是使用光标作为关键视觉引导，以视频格式捕捉和叙述GUI操作。Act2Cap强调GUI交互的细节，特别是基于光标的操作如点击和拖拽，这对于推进GUI密集型任务的自动化能力至关重要。它包含了一个涵盖多种Windows GUI环境的4,189个样本的大型数据集，采用基于元素级交并比（Intersection over Union）的指标来评估语义准确性以及时间和空间精度。

To provide a scalable and genuine computer environment for multimodal agents, OSWorld [419] introduces a pioneering framework that supports task setup, execution-based evaluation, and interactive learning across multiple operating systems, including Ubuntu, Windows, and macOS. OSWorld serves as a unified environment that mirrors the complexity and diversity of real-world computer use, accommodating arbitrary applications and open-ended computer tasks. It includes a comprehensive suite of 369 tasks on Ubuntu and 43 tasks on Windows, utilizing execution-based evaluation metrics like success rate for rigorous assessment.
为了为多模态代理提供一个可扩展且真实的计算机环境，OSWorld [419]引入了一个开创性框架，支持任务设置、基于执行的评估以及跨多个操作系统（包括Ubuntu、Windows和macOS）的交互式学习。OSWorld作为一个统一环境，反映了现实计算机使用的复杂性和多样性，支持任意应用程序和开放式计算任务。它包含了Ubuntu上的369个任务和Windows上的43个任务，利用基于执行的评估指标如成功率进行严格评估。

Building on OSWorld, WindowsArena [453] adapts the framework to create over 150 diverse tasks specifically for the Windows operating system. Focusing on multi-modal, multi-step tasks, it requires agents to demonstrate abilities in planning, screen understanding, and tool usage within a real Windows environment. Addressing the challenge of slow evaluation times, WindowsArena enables parallelized deployment in the Azure cloud, drastically reducing evaluation time and allowing for comprehensive testing across various applications and web domains.
基于OSWorld，WindowsArena [453]调整该框架，专门为Windows操作系统创建了150多个多样化任务。该基准聚焦于多模态、多步骤任务，要求代理展示规划、屏幕理解和工具使用能力，均在真实的Windows环境中完成。针对评估时间较长的挑战，WindowsArena支持在Azure云中并行部署，大幅缩短评估时间，并允许在各种应用和网页领域进行全面测试。

TABLE 36: Overview of mobile GUI agent benchmarks (Part I).
表36：移动GUI代理基准测试概览（第一部分）。

Benchmark	Platform	Year	Live	Highlight	Data Size	Metric	Measurement	Link
AndroidEnv 263	Android	2021	Yes	Provides an open-source platform based on the Android ecosystem with over 100 tasks across approximately 30 apps, focusing on reinforcement learning for various Android interactions.	100+ tasks	NA	NA	https: //github.com/ google-deepmind/ android_env
PIXELHELP 144]	Android	2020	No	Includes a corpus of natural language instructions paired with UI actions across four task categories, aiding in grounding language to UI interactions.	187 multistep instructions	Step Success Rate	Element Match, Action Match	https: //github.com/ google-research google-research tree/master/ seq2act
Mobile-Env 438]	Android	2024	Yes	Comprehensive toolkit for Android GUI benchmarks to enable controlled evaluations of real-world app interactions.	224 tasks	Task Success Rate, Step Success Rate	Text Match, Element Match, Image Match, State Information	https://github.com/X-LANCE/ Mobile-Env
B-MOCA [439]	Android	2024	Yes	Benchmarks mobile device control agents on realistic tasks, incorporating UI layout and language randomization to evaluate generalization capabilities.	131 tasks	ask Success Rate	Element Match, State Information	https://b-moca.github.io/
AndroidWorld 440]	Android	2024	Yes	Offers a dynamic Android environment, allowing for diverse natural language instruction testing.	116 tasks	Task sk Success Rate	State Information	https: //github.com/ google-research android world
Mobile-Eval 158]	Android	2024	Yes	Benchmark based on mainstream Android apps, and designed to test common mobile interactions.	30 instructions	Task Success Rate, Step Success Rate, Efficiency Score	Text Match, Path Length	https://github.com/X-PLUG/ MobileAgent
DroidTask [156]	Android	2024	Yes	Android Task Automation benchmark supports exploration and task recording in real apps with corresponding GUI action traces.	158 tasks	Step Success Rate, Task Success Rate	Element Match, Action Match	https: //github.com/ MobileLLM/ AutoDroid
AITW 358	Android	2023	No	A large-scale dataset, which is partly inspired by PIXEL-HELP, covering diverse Android interactions.	715,142 episodes	Task Success Rate, Step Success Rate	Action Match	https: //github.com/ google-research google-research tree/master/ android in the wild
AndroidArena 441]	Android	2024	Yes	Focuses on daily cross-app and constrained tasks within the Android ecosystem, providing single-app and multi-app interaction scenarios.	221 tasks	Task Success Rate, Step Success Rate, Efficiency Score	Action Match, Path Length	https: //github.com/ AndroidArenaAgeni AndroidArena

ANDROIDLAB 363]	Android	2024	Yes	Provides a structured evaluation framework with 138 tasks across nine apps, supporting both text-only and multimodal agent evaluations on Android.	138 tasks	Task Success Rate, Step Success Rate Efficiency Score	Element Match, Image Match	https://github.com/THUDM/ Android-Lab
GTArena 447]	Mobile applications	2024	No	Introduces a Transition Tuple for GUI defects, enabling large-scale defect dataset creation and reproducible, end-to-end automated testing.	10,000+ GUI display and GUI interactions	Task Success Rate, Step Success Rate	Text Match, Element Match, Action Match	https: //github.com/ ZJU-ACES-ISE/ ChatUITest

基准测试	平台	年份	实时	亮点	数据规模	指标	测量方式	链接
AndroidEnv 263	安卓	2021	是	提供基于Android生态系统的开源平台，涵盖约30个应用中的100多个任务，重点研究各种Android交互的强化学习（Reinforcement Learning）。	100+ 任务	无	无	https: //github.com/ google-deepmind/ android_env
PIXELHELP 144]	安卓	2020	否	包含一组自然语言指令与UI操作配对的语料，涵盖四类任务，有助于将语言与UI交互进行绑定。	187条多步骤指令	步骤成功率	元素匹配，动作匹配	https: //github.com/ google-research google-research tree/master/ seq2act
Mobile-Env 438]	安卓	2024	是	面向Android图形用户界面基准的综合工具包，支持对真实应用交互进行受控评估。	224 任务	任务成功率，步骤成功率	文本匹配，元素匹配，图像匹配，状态信息	https://github.com/X-LANCE/ Mobile-Env
B-MOCA [439]	安卓	2024	是	在真实任务中对移动设备控制代理进行基准测试，结合UI布局和语言随机化以评估泛化能力。	131 任务	任务成功率	元素匹配，状态信息	https://b-moca.github.io/
AndroidWorld 440]	安卓	2024	是	提供动态Android环境，支持多样的自然语言指令测试。	116 任务	任务成功率	状态信息	https: //github.com/ google-research android world
Mobile-Eval 158]	安卓	2024	是	基于主流Android应用的基准，设计用于测试常见移动交互。	30条指令	任务成功率，步骤成功率，效率评分	文本匹配，路径长度	https://github.com/X-PLUG/ MobileAgent
DroidTask [156]	安卓	2024	是	Android任务自动化基准，支持在真实应用中探索和任务录制，并附带相应的GUI操作轨迹。	158 任务	步骤成功率，任务成功率	元素匹配，动作匹配	https: //github.com/ MobileLLM/ AutoDroid
AITW 358	安卓	2023	否	一个大规模数据集，部分灵感来源于PIXEL-HELP，涵盖多样的Android交互。	715,142 个回合	任务成功率，步骤成功率	动作匹配	https: //github.com/ google-research google-research tree/master/ android in the wild
AndroidArena 441]	安卓	2024	是	聚焦Android生态系统中的日常跨应用及受限任务，提供单应用和多应用交互场景。	221 任务	任务成功率，步骤成功率，效率评分	动作匹配，路径长度	https: //github.com/ AndroidArenaAgeni AndroidArena
ANDROIDLAB 363]	安卓	2024	是	提供结构化评估框架，涵盖九个应用中的138个任务，支持Android上的纯文本及多模态代理评估。	138个任务	任务成功率，步骤成功率，效率评分	元素匹配，图像匹配	https://github.com/THUDM/ Android-Lab
GTArena 447]	移动应用	2024	否	引入了用于GUI缺陷的转换元组（Transition Tuple），实现大规模缺陷数据集的构建及可复现的端到端自动化测试。	10,000+ GUI显示和GUI交互	任务成功率，步骤成功率	文本匹配，元素匹配，动作匹配	https: //github.com/ ZJU-ACES-ISE/ ChatUITest

Focusing on office automation tasks, OF-FICEBENCH 455] introduces a groundbreaking framework for benchmarking LLM agents in realistic office workflows. Simulating intricate workflows across multiple office applications like Word, Excel, and Email within a Linux Docker environment, it evaluates agents' proficiency in cross-application automation. The benchmark challenges agents with complex tasks at varying difficulty levels, demanding adaptability to different complexities and use cases. Customized metrics assess operation accuracy and decision-making, providing critical insights into agents' capabilities in managing multi-application office scenarios.

专注于办公自动化任务，OF-FICEBENCH 455] 引入了一个开创性的框架，用于在真实办公流程中对大型语言模型（LLM）代理进行基准测试。该框架在 Linux Docker 环境中模拟了跨多个办公应用程序（如 Word、Excel 和电子邮件）的复杂工作流程，评估代理在跨应用自动化中的能力。该基准测试通过不同难度级别的复杂任务挑战代理，要求其适应不同的复杂性和使用场景。定制的指标评估操作准确性和决策能力，提供了代理在管理多应用办公场景中的关键能力洞察。

Addressing the automation of data science and engineering workflows, Spider2-V [454] offers a distinctive benchmark. It features 494 real-world tasks across 20 enterprise-level applications, spanning the entire data science workflow from data warehousing to visualization. Assessing agents' abilities to handle both code generation and complex GUI interactions within authentic enterprise software environments on Ubuntu, it employs a multifaceted evaluation method that includes information-based validation, file-based comparison, and execution-based verification.
针对数据科学与工程工作流的自动化，Spider2-V [454] 提供了一个独特的基准测试。它涵盖了 20 个企业级应用中的 494 个真实任务，涵盖从数据仓储到可视化的整个数据科学工作流程。该基准评估代理在 Ubuntu 上真实企业软件环境中处理代码生成和复杂图形用户界面（GUI）交互的能力，采用包括基于信息的验证、基于文件的比较和基于执行的验证在内的多维度评估方法。

TABLE 37: Overview of mobile GUI agent benchmarks (Part II).
表 37：移动 GUI 代理基准测试概览（第二部分）。

Benchmark	Platform	Year	Live	Highlight	Data Size	Metric	Measurement	Link
A3 448	Mobile Android	2025	Yes	Introduces a novel business-level LLM-based evaluation process, significantly reducing human labor and coding expertise requirements.	201 tasks across 21 widely used apps.	Task Success Rate.	Element Match, Action Match.	https: //yuxiangchai. github.io/ Android-Agent-Arena
LlamaTouch 445	Mobile Android	2024	Yes	Enables faithful and scalable evaluations for mobile UI task automation by matching task execution traces against annotated essential states	496 tasks covering 57 unique Android applications	Task Success Rate, Step Success Rate, Efficiency Score	Text Action Match, State Information Match	https: //github.com/ LlamaTouch/ LlamaTouch
MobileAgentBenc 446]	Mobile An droid	2024	Yes	Provides a fully autonomous evaluation process on real Android devices and flexibility in judging success conditions across multiple paths to completion	100 tasks across 10 open-source Android applications	Task Success Rate, Efficiency Score, Latency, Token Cost	State Information (UI State Matching)	https:// mobileagentbench. github.io/
Mobile-Bench 442]	Android	2024	Yes	Supports both UI and API-based actions in multi-app scenarios, testing agents on single and multi-task structures with a checkpoint-based evaluation approach.	832 entries (200+ tasks)	Task Success Rate, Step Success Rate, Efficiency Score	Action Match, Path Length	https://github com/XiaoMi/ MobileBench
Mobile Safety Bench 443]	Android	2024	Yes	Prioritizes safety evaluation in mobile control tasks, with distinct tasks focused on helpfulness, privacy, and legal compliance.	100 tasks	Task Success Rate, Risk Mitigation Success	Action Match with Safety Considered, Element Match, State Information	https:// mobilesafetybench. github.io/
SPA-BENCH 444]	Android	2024	Yes	Extensive evaluation framework supporting single-app and cross-app tasks in English and Chinese, providing a plug-and-play structure for diverse task scenarios.	340 tasks	Task Success Rate, Step Success Rate, Efficiency Score	Action Match, State Information, Time Spent, API Cost	https: //spa-bench. github.io
SPHINX 449]	Android	2025	Yes	Provides a fully automated benchmarking suite and introduces a multi-dimensional evaluation framework.	284 common tasks.	Task Success Rate, Efficiency Score Completion under Policy, Turn Success Rate.	Text Match, Image Match, Element Match, Action Match.	7
AEIA-MN 45 450]	Mobile Android	2025	Yes	Introduces the Active Environment Injection Attack (AEIA) framework that actively manipulates environmental elements (e.g., notifications) in mobile operating systems to mislead multimodal LLM-powered agents.	61 tasks (Android-World) 45 tasks (AppAgent)	Task Success Rate, Risk Ratio, Efficiency Score	Text Match, State Information, Action Match	/

AutoEval 451]	Mobile Ar droid	2025	Yes	Introduces a fully autonomous evaluation framework for mobile agents, eliminating the need for manual task reward signal definition and extensive evaluation code development.	93 tasks	Task Success Rate	Action Match, State Information	/
LearnGUI 321]	Mobile Android	2025	Yes	The first benchmark to systematically study few-shot demonstration-based learning in mobile GUI agents, featuring both offline and online task environments	Offline: 2,252 tasks with k-shot variants across 44 apps; Online: 101 interactive tasks across 20 apps	Task Success Rate	Action Match	https: //lgy0404.github. io/LearnAct

基准测试	平台	年份	实时	亮点	数据规模	指标	测量方式	链接
A3 448	移动端安卓	2025	是	引入了一种新颖的基于大语言模型（LLM）的业务级评估流程，显著减少了对人工劳动和编码专业知识的需求。	涵盖21个广泛使用应用的201个任务。	任务成功率。	元素匹配，动作匹配。	https: //yuxiangchai.github.io/ Android-Agent-Arena
LlamaTouch 445	移动端安卓	2024	是	通过将任务执行轨迹与标注的关键状态匹配，实现对移动UI任务自动化的真实且可扩展的评估。	涵盖57个独特安卓应用的496个任务	任务成功率，步骤成功率，效率得分	文本动作匹配，状态信息匹配	https: //github.com/LlamaTouch/LlamaTouch
MobileAgentBenc 446]	移动安卓	2024	是	提供在真实安卓设备上的全自动评估流程，并在多路径完成条件下灵活判断成功标准	涵盖10个开源安卓应用的100个任务	任务成功率，效率得分，延迟，令牌成本	状态信息（UI状态匹配）	https:// mobileagentbench.github.io/
Mobile-Bench 442]	安卓	2024	是	支持多应用场景中的UI和API动作，采用基于检查点的评估方法测试单任务和多任务结构的代理。	832条目（200+任务）	任务成功率，步骤成功率，效率得分	动作匹配，路径长度	https://github com/XiaoMi/MobileBench
Mobile Safety Bench 443]	安卓	2024	是	优先考虑移动控制任务中的安全评估，设有专注于帮助性、隐私和法律合规的不同任务。	100个任务	任务成功率，风险缓解成功率	考虑安全性的动作匹配，元素匹配，状态信息	https:// mobilesafetybench.github.io/
SPA-BENCH 444]	安卓	2024	是	支持单应用和跨应用任务的广泛评估框架，涵盖英语和中文，提供多样任务场景的即插即用结构。	340个任务	任务成功率，步骤成功率，效率得分	动作匹配，状态信息，耗时，API成本	https: //spa-bench.github.io
SPHINX 449]	安卓	2025	是	提供全自动基准测试套件，并引入多维度评估框架。	284个常见任务。	任务成功率，效率得分，策略下完成率，回合成功率。	文本匹配，图像匹配，元素匹配，动作匹配。	7
AEIA-MN 45 450]	移动端安卓	2025	是	引入主动环境注入攻击（Active Environment Injection Attack，AEIA）框架，主动操控移动操作系统中的环境元素（如通知），误导多模态大语言模型（LLM）驱动的代理。	61个任务（Android-World）45个任务（AppAgent）	任务成功率，风险比，效率得分	文本匹配，状态信息，动作匹配	/
AutoEval 451]	移动安卓	2025	是	引入全自动移动代理评估框架，免除手动定义任务奖励信号和大量评估代码开发的需求。	93个任务	任务成功率	动作匹配，状态信息	/
LearnGUI 321]	移动端安卓	2025	是	首个系统性研究基于少量示范学习的移动GUI代理基准，涵盖离线和在线任务环境	离线：44个应用中包含k次示例变体的2,252个任务；在线：20个应用中的101个交互式任务	任务成功率	动作匹配	https: //lgy0404.github.io/LearnAct

In the realm of productivity software, AssistGUI [109] provides a pioneering framework for evaluating agents' capabilities. It introduces an Actor-Critic Embodied Agent framework capable of complex hierarchical task planning, GUI parsing, and action generation. The dataset includes diverse tasks across design, office work, and system settings, supported by project files for reproducibility. By emphasizing outcome-driven evaluation with pixel-level precision and procedural adherence, AssistGUI highlights the potential and limitations of current LLM-based agents in managing intricate desktop software workflows.

在生产力软件领域，AssistGUI [109] 提供了一个开创性的框架，用于评估智能体的能力。它引入了一个具备复杂层级任务规划、GUI解析和动作生成能力的演员-评论家具身智能体（Actor-Critic Embodied Agent）框架。该数据集涵盖设计、办公和系统设置等多样化任务，并配备项目文件以保证可复现性。通过强调基于结果的评估，结合像素级精度和流程遵循，AssistGUI 突出了当前基于大型语言模型（LLM）智能体在管理复杂桌面软件工作流程中的潜力与局限。

WorldGUI [456] is a benchmark designed to evaluate GUI agents under dynamic conditions on the Windows platform. Unlike previous static benchmarks, it introduces varied initial states to simulate real-world interactions across both desktop and web applications. Rather than always starting from a fixed default state, agents must adapt to changing UI layouts, user interactions, system settings, and pre-existing conditions, requiring robust adaptability to perform effectively. The benchmark comprises 315 tasks spanning 10 popular software applications and incorporates instructional videos, project files, and multiple pre-action scenarios, providing a comprehensive and realistic evaluation framework for assessing an agent's ability to handle complex task execution.
WorldGUI [456] 是一个旨在评估Windows平台下GUI智能体在动态条件下表现的基准测试。不同于以往的静态基准，它引入了多样的初始状态，以模拟桌面和网页应用中的真实交互。智能体不再总是从固定的默认状态开始，而必须适应不断变化的界面布局、用户交互、系统设置和预先存在的条件，要求具备强大的适应能力以有效执行任务。该基准包含315个任务，涵盖10款流行软件，配备教学视频、项目文件及多种预动作场景，提供了一个全面且真实的评估框架，用于衡量智能体处理复杂任务执行的能力。

TABLE 38: Overview of computer GUI agent benchmarks.
表38：计算机GUI智能体基准测试概览。

Benchmark	Platform	Year	Live	Highlight	Data Size	Metric	Measurement	Link
OSWorld 419]	Linux, Windows, ma-cOS	2024	Yes	Scalable, real computer environment for multimodal agents, supporting task setup, execution-based evaluation, and interactive learning across Ubuntu, Windows, and macOS.	369 Ubuntu tasks, 43 Windows tasks	Task Rate sk Success	Execution-based State Information (such 1 internal file interpretation, permission management)	https://os-world.github.io/
Windows Agent Arena [453]	Windows	2024	Yes	Adaptation of OSWorld focusing exclusively on the Windows OS with diverse multi-step tasks, enabling agents to use a wide range of applications and tools.	154 tasks	Task Success Rate	Same as OS-World, scalable with cloud parallelization	https: //microsoft. github.io/ WindowsAgentArena
OmniACT 459]	macOS, Linux, Windows	2024	No	Assesses agents' capability to generate executable programs for computer tasks across desktop and web applications in various OS environments, prioritizing multimodal challenges.	9,802 data points	Task Success Rate, Step Success Rate	Action Match	https: //huggingface. co/datasets/ Writer/omniact
VideoGUI 460]	Windows	2024	No	Focuses on visual-centric tasks from instructional videos, emphasizing planning and action precision in applications like Adobe Photoshop and Premiere Pro	178 tasks, 463 subtasks	Task sk Success Rate	State Information, Action Match	https://showlab.github.io/ videogui
Spider2-V 454	Linux	2024	Yes	Benchmarks agents across data science and engineering workflows in authentic enterprise software environments, covering tasks from data ingestion to visualization.	494 tasks	Task Rate Success	Action Match, State Information	https: //spider2-v. github.io
Act2Cap 327]	Windows	2024	Yes	Emphasizes GUI action narration using cursor-based prompts in video format, covering a variety of GUI interactions like clicks, typing, and dragging.	4,189 samples	Step Success Rate	Element Match	https://showlab.github.io/ GUI-Narrator
OFFICEBENCH 455]	Linux	2024	Yes	Tests cross-application automation in office work-flows with complex multistep tasks across applications like Word and Excel, assessing operational integration in realistic scenarios.	300 tasks	Task Success Rate	Action Match, Text Match, State Information	https://github.com/zlwang-cs/ OfficeBench

AssistGUI [109]	Windows	2024	Yes	The first benchmark focused on task-oriented desktop GUI automation	100 tasks from 9 popular applications	Task Success Rate, Efficiency Score	Element Match, Action Match	https://showlab github.io/assistgui/
WorldGUI 456]	Windows	2025	Yes	First GUI benchmark designed to evaluate dynamic GUI interactions by incorporating various initial states.	315 total tasks from 10 Windows applications	Task Success Rate, Efficiency Score	Image Match, Element Match. Action Match	/
	Desktop (Windows, Linux)	2025	No	The first large-scale benchmark specifically designed for desktop GUI agents	8,227 query-label pairs in total	Rate SECESS	Action Match, Text Match	https://uivision.github.io
Computer Agent Arena [458]	Windows, Ubuntu, macOS	2025	Yes	The first large-scale, open-ended evaluation platform for multimodal LLM-based agents in real desktop computing environments	User-proposed tasks	Task Success Rate	Human evalua tors	https://arena.xlang.ai/

基准测试	平台	年份	实时	亮点	数据规模	指标	测量方式	链接
OSWorld 419]	Linux、Windows、macOS	2024	是	可扩展的多模态智能体真实计算机环境，支持任务设置、基于执行的评估及跨Ubuntu、Windows和macOS的交互式学习。	369个Ubuntu任务，43个Windows任务	任务成功率	基于执行的状态信息（如内部文件解析、权限管理）	https://os-world.github.io/
Windows Agent Arena [453]	Windows	2024	是	OSWorld的Windows专版，聚焦多步骤多样任务，使智能体能够使用广泛的应用程序和工具。	154个任务	任务成功率	与OSWorld相同，支持云端并行扩展	https://microsoft.github.io/WindowsAgentArena
OmniACT 459]	macOS、Linux、Windows	2024	否	评估智能体在多操作系统环境下为桌面和网页应用生成可执行程序的能力，重点考察多模态挑战。	9,802条数据点	任务成功率，步骤成功率	动作匹配	https://huggingface.co/datasets/Writer/omniact
VideoGUI 460]	Windows	2024	否	聚焦来自教学视频的视觉中心任务，强调在Adobe Photoshop和Premiere Pro等应用中的规划与动作精准性	178个任务，463个子任务	任务成功率	状态信息，动作匹配	https://showlab.github.io/videogui
Spider2-V 454	Linux	2024	是	在真实企业软件环境中对智能体进行数据科学与工程工作流的基准测试，涵盖从数据摄取到可视化的任务。	494个任务	任务成功率	动作匹配，状态信息	https://spider2-v.github.io
Act2Cap 327]	Windows	2024	是	强调使用基于光标提示的视频格式进行GUI动作叙述，涵盖点击、输入、拖拽等多种GUI交互。	4,189个样本	步骤成功率	元素匹配	https://showlab.github.io/GUI-Narrator
OFFICEBENCH 455]	Linux	2024	是	测试跨应用办公自动化，包含Word和Excel等复杂多步骤任务，评估现实场景中的操作集成能力。	300个任务	任务成功率	动作匹配，文本匹配，状态信息	https://github.com/zlwang-cs/OfficeBench
AssistGUI [109]	Windows	2024	是	首个聚焦任务导向桌面GUI自动化的基准测试	来自9个流行应用的100个任务	任务成功率，效率评分	元素匹配，动作匹配	https://showlab.github.io/assistgui/

WorldGUI 456]	Windows	2025	是	首个设计用于评估动态GUI交互的基准，涵盖多种初始状态。	来自10个Windows应用程序的315个总任务	任务成功率，效率评分	图像匹配，元素匹配，动作匹配	/
	桌面（Windows，Linux）	2025	否	首个专为桌面GUI代理设计的大规模基准测试	共8,227个查询-标签对	评分SECESS	动作匹配，文本匹配	https://uivision.github.io
计算机代理竞技场 [458]	Windows，Ubuntu，macOS	2025	是	首个面向真实桌面计算环境中基于多模态大语言模型（LLM）代理的开放式大规模评估平台	用户提出的任务	任务成功率	人工评估者	https://arena.xlang.ai/

Computer Agent Arena [458] presents a new paradigm for benchmarking LLM-based GUI agents through live, user-configured desktop environments. Unlike traditional static datasets, it provides an interactive cloud-based infrastructure where agents are evaluated on tasks spanning web browsing, programming, and productivity using real applications like Google Docs, VSCode, and Slack. Its innovation lies in using head-to-head agent comparisons, human judgment, and Elo-based ranking to evaluate general-purpose digital agents in realistic settings. The benchmark supports Windows and Ubuntu, with MacOS support planned, and allows customized task scenarios with diverse software and website setups. By enabling crowdsourced evaluations and planning open-source releases, it fosters community-driven improvements and robust comparisons.
Computer Agent Arena [458] 提出了一种通过实时、用户配置的桌面环境对基于大型语言模型（LLM）的图形用户界面（GUI）代理进行基准测试的新范式。不同于传统的静态数据集，它提供了一个交互式的云端基础设施，代理在使用真实应用程序如Google Docs、VSCode和Slack执行涵盖网页浏览、编程和生产力的任务时进行评估。其创新之处在于通过代理间的直接对抗比较、人类评判和基于Elo的排名方法，在真实场景中评估通用数字代理。该基准支持Windows和Ubuntu系统，计划支持MacOS，并允许通过多样化的软件和网站配置定制任务场景。通过支持众包评估和计划开源发布，促进社区驱动的改进和稳健的比较。

Collectively, these benchmarks provide comprehensive evaluation frameworks for GUI agents on desktop platforms, addressing challenges in task complexity, cross-application automation, scalability, and fidelity. Their contributions are instrumental in advancing the development of sophisticated agents capable of complex interactions in desktop environments.
这些基准共同为桌面平台上的GUI代理提供了全面的评估框架，解决了任务复杂性、跨应用自动化、可扩展性和真实性等挑战。它们的贡献对于推动能够在桌面环境中进行复杂交互的高级代理的发展具有重要作用。

18.13 9.7 Cross-Platform Agent Benchmarks

18.14 9.7 跨平台代理基准

To develop GUI agents capable of operating across multiple platforms, cross-platform benchmarks are essential. These benchmarks challenge agents to adapt to different environments and interfaces, evaluating their versatility and robustness. We provide an overview of benchmarks for cross-platform GUI agents in Tables 39.
为了开发能够跨多个平台运行的GUI代理，跨平台基准测试至关重要。这些基准挑战代理适应不同环境和界面，评估其多样性和鲁棒性。我们在表39中提供了跨平台GUI代理基准的概览。

Addressing this need, VisualAgentBench (VAB) 374 represents a pioneering benchmark for evaluating GUI and multimodal agents across a broad spectrum of realistic, interactive tasks. Encompassing platforms such as Web (WebArena-Lite [412]), Android (VAB-Mobile [363]), and game environments, VAB focuses on vision-based interaction and high-level decision-making tasks. The benchmark employs a multi-level data collection strategy involving human demonstrations, program-based solvers, and model bootstrapping. Evaluation metrics concentrate on success rates, ensuring comprehensive performance assessments in tasks like navigation and content modification, thereby filling a significant gap in benchmarking standards for GUI-based LLM agents.
针对这一需求，VisualAgentBench（VAB）374代表了一个开创性的基准，用于评估GUI和多模态代理在广泛的真实交互任务中的表现。涵盖了Web（WebArena-Lite [412]）、Android（VAB-Mobile [363]）及游戏环境等平台，VAB聚焦于基于视觉的交互和高层次决策任务。该基准采用多层次数据收集策略，包括人类示范、基于程序的求解器和模型自举。评估指标集中于成功率，确保对导航和内容修改等任务的全面性能评估，从而填补了基于GUI的LLM代理基准标准中的重要空白。

Complementing this, CRAB 461 introduces an in vative benchmark by evaluating multimodal language model agents in cross-environment interactions. It uniquely supports seamless multi-device task execution, evaluating agents in scenarios where tasks span both Ubuntu Linux and Android environments. By introducing a graph-based evaluation method that breaks down tasks into sub-goals and accommodates multiple correct paths to completion, CRAB provides a nuanced assessment of planning, decision-making, and adaptability. Metrics such as Completion Ratio, Execution Efficiency, Cost Efficiency, and Success Rate offer comprehensive insights into agent performance.
作为补充，CRAB 461通过评估多模态语言模型代理在跨环境交互中的表现，提出了创新的基准。它独特地支持无缝的多设备任务执行，评

估代理在涵盖Ubuntu Linux和Android环境的场景中的表现。通过引入基于图的评估方法，将任务分解为子目标并允许多条正确路径完成，CRAB对规划、决策和适应性提供了细致的评估。完成率、执行效率、成本效率和成功率等指标为代理性能提供了全面洞察。

Focusing on GUI grounding for cross-platform visual agents, ScreenSpot [25] offers a comprehensive benchmark emphasizing tasks that rely on interpreting screenshots rather than structured data. ScreenSpot includes over 600 screen-shots and 1,200 diverse instructions spanning mobile (iOS, Android), desktop (macOS, Windows), and web platforms. It evaluates click accuracy and localization precision by measuring how effectively agents can identify and interact with GUI elements through visual cues alone. By challenging models with a wide variety of UI elements, ScreenSpot addresses real-world complexities, making it an essential resource for evaluating visual GUI agents across varied environments.
专注于跨平台视觉代理的GUI定位，ScreenSpot [25]提供了一个全面的基准，强调依赖于截图而非结构化数据的任务。ScreenSpot包含600多张截图和1200条多样化指令，涵盖移动端（iOS、Android）、桌面端（macOS、Windows）和网页平台。它通过测量代理识别和通过视觉线索与GUI元素交互的能力，评估点击准确率和定位精度。通过挑战模型处理各种UI元素，ScreenSpot应对了现实世界的复杂性，成为评估跨多样环境视觉GUI代理的重要资源。

These cross-platform benchmarks collectively advance the development of GUI agents capable of operating seamlessly across multiple platforms. By providing comprehensive evaluation frameworks, they are instrumental in assessing and enhancing the versatility and adaptability of agents in diverse environments.
这些跨平台基准共同推动了能够无缝跨多个平台运行的GUI代理的发展。通过提供全面的评估框架，它们在评估和提升代理在多样环境中的多功能性和适应性方面发挥了关键作用。

18.15 9.8 Takeaways

18.16 9.8 结论

The evolution of GUI agent benchmarks reflects a broader shift towards more realistic, interactive, and comprehensive evaluation environments. This section highlights key trends and future directions in the benchmarking of LLM-brained GUI agents.
GUI代理基准的发展反映了向更真实、交互性更强且更全面的评估环境的广泛转变。本节重点介绍了基于大型语言模型的GUI代理基准测试中的关键趋势和未来方向。

1. Towards More Interactive and Realistic Environments: Recent advancements in GUI agent benchmarking emphasize the transition from synthetic scenarios to more interactive and realistic environments. This shift is evident in the use of simulators, Docker containers, and real-world applications to create "live" environments that better mimic genuine user interactions. Such environments t only provide a more accurate assessment of agent capabilities but also pose new challenges in terms of performance and robustness.
2. 向更具交互性和真实性的环境发展：近期GUI代理基准测试的进展强调了从合成场景向更具交互性和真实性环境的转变。这一转变体现在使用模拟器、Docker容器和真实应用程序创建“实时”环境，更好地模拟真实用户交互。这类环境不仅提供了对代理能力更准确的评估，也在性能和鲁棒性方面带来了新的挑战。

2. Cross-Platform Benchmarks: The emergence of cross-platform benchmarks that encompass mobile, web, and desktop environments represents a significant step towards evaluating the generalizability of GUI agents. However, these benchmarks introduce fundamental challenges unique to each platform. A unified interface for accessing platform-specific information, such as HTML and DOM structures, could substantially streamline the benchmarking process and reduce implementation efforts. Future work should focus on standardizing these interfaces to facilitate seamless agent evaluation across diverse environments.
3. 跨平台基准：涵盖移动端、网页和桌面环境的跨平台基准的出现，是评估GUI代理泛化能力的重要一步。然而，这些基准引入了各平台独有的根本性挑战。统一访问平台特定信息（如HTML和DOM结构）的接口，能够显著简化基准测试流程并减少实现工作量。未来工作应聚焦于标准化这些接口，以促进代理在多样环境中的无缝评估。

3. Increased Human Interaction and Realism: There is a growing trend towards incorporating more human-like interactions in benchmarks, as seen in multi-turn and conversational scenarios. These setups mirror real-world use cases more closely, thereby providing a rigorous test of an agent's ability to handle dynamic, iterative interactions. As GUI agents become more sophisticated, benchmarks must continue to evolve to include these nuanced interaction patterns, ensuring agents can operate effectively in complex, human-centric environments.
4. 增强的人机交互和真实性：基准测试中越来越多地融入类人交互，如多轮和对话场景。这些设置更贴近真实使用案例，从而严格考验代理处理动态、迭代交互的能力。随着GUI代理日益复杂，基准测试必须持续演进，纳入这些细腻的交互模式，确保代理能在复杂且以人为中心的环境中有效运行。

4. Scalability and Automation Challenges: Scalability remains a significant concern in benchmarking GUI agents. The creation of realistic tasks and the development of evaluation methods for individual cases often require substantial human effort. Automation of these processes could alleviate some of the scalability issues, enabling more extensive and efficient benchmarking. Future research should explore automated task generation and evaluation techniques to enhance scalability.
5. 可扩展性与自动化挑战：可扩展性仍然是GUI代理基准测试中的一个重要问题。创建真实任务和开发针对个案的评估方法通常需要大量人工投入。自动化这些过程可以缓解部分可扩展性问题，从而实现更广泛和高效的基准测试。未来的研究应探索自动任务生成和评估技术，以提升可扩展性。

5. Emphasis on Safety, Privacy, and Compliance: There is a table trend towards evaluating GUI agents on safety, privacy, and compliance metrics. These considerations are increasingly important as agents are integrated into sensitive and regulated domains. Encouraging this trend will help ensure that agents t only perform tasks effectively but also adhere to necessary legal and ethical standards. Future benchmarks should continue to expand on these dimensions, incorporating evaluations that reflect real-world compliance and data security requirements.
6. 强调安全性、隐私和合规性：在GUI代理的评估中，安全性、隐私和合规性指标正成为一个显著趋势。随着代理被应用于敏感和受监管领域，这些考量变得愈发重要。鼓励这一趋势有助于确保代理不仅高效完成任务，还能遵守必要的法律和伦理标准。未来的基准测试应继续扩展这些维度，纳入反映现实合规性和数据安全要求的评估内容。

The landscape of GUI agent benchmarking is rapidly evolving to meet the demands of increasingly complex and interactive environments. By embracing cross-platform evaluations, fostering human-like interactions, addressing scalability challenges, and prioritizing safety and compliance, the community can pave the way for the next generation of sophisticated GUI agents. Continued in vation and collaboration will be essential in refining benchmarks to ensure they accurately capture the multifaceted capabilities of modern agents, ultimately leading to more intuitive and effective human-computer interactions.
GUI代理基准测试的格局正在迅速演变，以满足日益复杂和交互性强的环境需求。通过采用跨平台评估、促进类人交互、解决可扩展性挑战以及优先考虑安全与合规，社区能够为下一代先进的GUI代理铺平道路。持续的创新与合作对于完善基准测试至关重要，以确保其准确反映现代代理的多方面能力，最终实现更直观、高效的人机交互。

TABLE 39: Overview of cross-platform GUI agent benchmarks.
表39：跨平台GUI代理基准测试概览。

Benchmark	Platform	Year	Live	Highlight	Data Size	Metric	Measurement	Link
VisualAgent Bench [374]	Web, Android, Game, Virtual Embodied	2024	Yes	First benchmark designed for visual foundation agents across GUI and multimodal tasks, focusing on vision-centric interactions in Android, web, and game environments.	4,482 trajectories	Task Rate Success	Text Match	https://github.com/THUDM/VisualAgentBench/
SPR Benchmark 462]	Mobile, Web, Operating Systems	2024	Yes	Evaluates GUI screen readers' ability to describe both content and layout information	650 screen-shots annotated with 1,500 target points and regions	Task Success Rate, Efficiency Score	Text Match, Element Match	/
AgentStudio 237]	Windows, Linux, macOS	2024	Yes	Open toolkit for creating and benchmarking general-purpose virtual agents, supporting complex interactions across diverse software applications.	NA	Step Success Rate	Action Match, State Information, Image Match	https:// computer-agents. github.io/ agent-studio/
CRAB 461	Linux, Android	2024	Yes	Cross-environment benchmark evaluating agents across mobile and desktop devices, using a graph-based evaluation method to handle multiple correct paths and task flexibility.	120 tasks	Step Success Rate, Efficiency Score	Action Match	https: //github.com/ crab-benchmark
ScreenSpot 25]	iOS, Android, macOS, Windows, Web	2024	No	Vision-based GUI benchmark with pre-trained GUI grounding, assessing agents' ability to interact with GUI elements across mobile. desktop, and web platforms using only screenshots.	1,200 instructions	Step Success Rate	Action Match	https://github.com/njucckevin/SeeClick

基准测试	平台	年份	实时	亮点	数据规模	指标	测量方式	链接
VisualAgent基准[374]	网页，安卓，游戏，虚拟具身	2024	是	首个针对视觉基础代理（visual foundation agents）设计的基准，涵盖GUI和多模态任务，聚焦安卓、网页及游戏环境中的视觉交互。	4,482条轨迹	任务成功率	文本匹配	https://github.com/THUDM/VisualAgentBench/
SPR基准[462]	移动端，网页，操作系统	2024	是	评估GUI屏幕阅读器描述内容和布局信息的能力	650张截图，标注了1,500个目标点和区域	任务成功率，效率得分	文本匹配，元素匹配	/
AgentStudio[237]	Windows，Linux，macOS	2024	是	用于创建和评测通用虚拟代理的开源工具包，支持跨多种软件应用的复杂交互。	无	步骤成功率	动作匹配，状态信息，图像匹配	https://computer-agents.github.io/agent-studio/
CRAB[461]	Linux，安卓	2024	是	跨环境基准，评估移动端和桌面设备上的代理，采用基于图的评估方法以处理多条正确路径和任务灵活性。	120个任务	步骤成功率，效率得分	动作匹配	https://github.com/crab-benchmark
ScreenSpot[25]	iOS，安卓，macOS，Windows，网页	2024	否	基于视觉的GUI基准，具备预训练的GUI定位能力，评估代理仅通过截图在移动端、桌面和网页平台与GUI元素交互的能力。	1,200条指令	步骤成功率	动作匹配	https://github.com/njucckevin/SeeClick

19 10 APPLICATIONS OF LLM-BRAINED GUI AGENTS

20 10 具备大型语言模型智能的图形用户界面代理的应用

As LLM-brained GUI agents continue to mature, a growing number of applications leverage this concept to create more intelligent, user-friendly, and natural language-driven interfaces. These advancements are reflected in research papers, open-source projects, and industry solutions. Typical applications encompass (i) GUI testing, which has transitioned from traditional script-based approaches to more intuitive, natural language-based interactions, and (ii) virtual assistants, which automate users' daily tasks in a more adaptive and responsive manner through natural language interfaces.

随着具备大型语言模型（LLM）智能的图形用户界面（GUI）代理不断成熟，越来越多的应用利用这一概念，打造更智能、更友好且以自然语言驱动的界面。这些进展体现在研究论文、开源项目和行业解决方案中。典型应用包括：(i) GUI测试，已从传统的基于脚本的方法转向更直观的自然语言交互；(ii) 虚拟助手，通过自然语言界面以更适应性和响应性的方式自动化用户的日常任务。

20.1 10.1 GUI Testing

20.2 10.1 GUI测试

GUI testing evaluates a software application's graphical user interface to ensure compliance with specified requirements, functionality, and user experience standards. It verifies interface elements like buttons, menus, and windows, as well as their responses to user interactions. Initially conducted manually, GUI testing evolved with the advent of automation tools such as Selenium and Appium, enabling testers to automate repetitive tasks, increase coverage, and reduce testing time [35], [543]. However, LLM-powered GUI agents have introduced a paradigm shift, allowing non-experts to test GUIs intuitively through natural language interfaces. These agents cover diverse scenarios, including general testing, input generation, and bug reproduction, without the need for traditional scripting [543].

GUI测试评估软件应用的图形用户界面，以确保其符合指定的需求、功能和用户体验标准。它验证界面元素如按钮、菜单和窗口，以及它们对用户交互的响应。最初以手工方式进行，随着Selenium和Appium等自动化工具的出现，GUI测试得以自动化重复任务，提高覆盖率并缩短测试时间[35]，[543]。然而，基于LLM的GUI代理引入了范式转变，使非专业人员能够通过自然语言界面直观地测试GUI。这些代理涵盖多种场景，包括通用测试、输入生成和缺陷复现，无需传统脚本[543]。

Figure 29 and illustrates the use of an LLM-powered GUI agent to test font size adjustment on Windows OS. With only a natural language test case description, the agent autonomously performs the testing by executing UI operations, navigating through the settings menu, and leveraging its screen understanding capabilities to verify the final outcome of font size adjustment. This approach dramatically reduces the effort required for human or script-based testing. Next, we detail the GUI testing works powered by GUI agents, and first provide an overview Tables 40 41 and 42
图29展示了使用基于LLM的GUI代理测试Windows操作系统字体大小调整的过程。仅凭自然语言的测试用例描述，代理便能自主执行测试，完成UI操作，导航设置菜单，并利用其屏幕理解能力验证字体大小调整的最终效果。这种方法大幅减少了人工或基于脚本测试所需的工作量。接下来，我们将详细介绍由GUI代理驱动的GUI测试工作，首先提供表40、41和42的概览。

20.2.1 10.1.1 General Testing

20.2.2 10.1.1 通用测试

Early explorations demonstrated how LLMs like GPT-3 could automate GUI testing by interpreting natural language test cases and programmatically executing them. For example, one approach integrates GUI states with GPT-3 prompts, leveraging tools like Selenium and OpenCV to reduce manual scripting and enable black-box testing [542]. Building on this, a subsequent study employed GPT-4 and Selenium WebDriver for web application testing, achieving superior branch coverage compared to traditional methods like monkey testing [463]. These advances highlight how LLMs simplify GUI testing workflows while significantly enhancing coverage and efficiency.
早期探索展示了如何利用GPT-3等LLM通过解释自然语言测试用例并以程序化方式执行，实现GUI测试自动化。例如，一种方法将GUI状态与GPT-3提示结合，利用Selenium和OpenCV等工具减少手工脚本编写，实现黑盒测试[542]。在此基础上，后续研究采用GPT-4和Selenium WebDriver进行网页应用测试，较传统的猴子测试方法实现了更优的分支覆盖率[463]。这些进展凸显了LLM简化GUI测试流程，同时显著提升覆盖率和效率的能力。

Further pushing boundaries, GPTDroid reframed GUI testing as an interactive Q&A task. By extracting structured semantic information from GUI pages and leveraging memory mechanisms for long-term exploration, it increased activity coverage by 32%, uncovering critical bugs with remarkable precision [125]. This approach underscores the potential of integrating conversational interfaces with memory for comprehensive app testing. For Android environments, DROIDAGENT introduced an intent-driven testing framework. It automates task generation and execution by perceiving GUI states in JSON format and using LLMs for realistic task planning. Its ability to set high-level goals and achieve superior feature coverage demonstrates how intent-based testing can transform functional verification in GUI applications [464].
进一步突破，GPTDroid将GUI测试重新定义为交互式问答任务。通过从GUI页面提取结构化语义信息并利用记忆机制进行长期探索，其活动覆盖率提升了32%，并以卓越的精度发现了关键缺陷[125]。该方法强调了结合对话界面与记忆机制以实现全面应用测试的潜力。在Android环境中，DROIDAGENT引入了基于意图的测试框架。它通过感知JSON格式的GUI状态并利用LLM进行真实任务规划，实现任务生成与执行自动化。其设定高层目标并实现优异特征覆盖率的能力，展示了基于意图的测试如何变革GUI应用的功能验证[464]。

TABLE 40: Overview of GUI-testing with LLM-powered GUI agents (Part I).
表40：基于LLM的GUI代理进行GUI测试概览（第一部分）。

Project	Category	Platform	Model	Perception	Action	Scenario	Highlight	Link
Daniel and Anne 542]	General testing	General-purpose platforms	GPT-3	GUI structure and state	Standard U operations	Automates the software testing process using natural language test cases	Applies GPT-3's language understanding capabilities to GUI-based software testing. enabling natural interaction through text-based test case descriptions.	https: //github.com/ neuroevolution% 2Dai/ SoftwareTestingLanguageModel:
Daniel and Anne 463]	General testing	Web platforms	GPT-4	HTML DOM structure	Standard I operations	Automated GUI testing to enhance branch coverage and efficiency	Performs end-to-end GUI testing using GPT-4's natural language understanding and reasoning capabilities.	https: //github.com/ Software TestingLLMs/ WebtestingWithLLMs
GPTDroid [125]	General testing	Mobile Android	GPT-3.5	UI view hierarchy files	Standard UI operations and compound actions	Automates GUI testing to improve testing coverage and detect bugs efficiently	Formulates GUI testing as a Q& A task, utilizing LLM capabilities to provide human-like interaction.	https: //github.com/ franklinbill GPTDroid
DROID-AGENT 464]	General testing	Mobile Android	GPT-3.5, GPT-4	JSON representation of the GUI state	Standard UI operations, higher-level APIs, and custom actions	Semantic, intent-driven automation of GUI testing	Autonomously generates and executes high-level, realistic tasks for Android GUI testing based on app-specific functionalities.	https://github.com/coinse/ droidagent
AUITest-Agent 465]	General testing	Mobile Android	GPT-4	GUI screenshots, UI hierarchy files, CV-enhanced techniques like Vision-UI	Standard operations	Automated functional testing of GUIs	Features dynamic agent organization for step-oriented testing and a multi-source data extraction strategy for precise function verification.	https://github.com/bz-lab/ AUITestAgent
VisionDroid 466]	General testing	Mobile Android	GPT-4	GUI screenshots with annotated bounding boxes, View hierarchy files	Standard UI operations	Identifies n crash bugs	Integrates vision-driven prompts and GUI text alignment with vision-language models to enhance understanding of G contexts and app logic.	https: //github.com/ testtestA6/ VisionDroid

AXNav 467]	Accessibility testing	iOS mobile devices	GPT-4	GUI screenshots, UI element detection model, and OCR	Gestures, capturing screenshots, and highlighting potential accessibility issues	Automates accessibility testing workflows. including testing features like VoiceOver, Dynamic T Type, Bold Text, and Button Shapes	Adapts to natural language test instructions and generates annotated videos to visually and interactively review accessibility test results.	/
LLMigrate 473]	General testing	Mobile Android	GPT-4o	DOM and screenshots	Standard UI operations	Automates the transfer of usage-based UI tests between Android apps	Leverages multimodal LLMs to perform UI test transfers without requiring source code access	/

项目	类别	平台	模型	感知	动作	场景	亮点	链接
Daniel 和 Anne 542]	通用测试	通用平台	GPT-3	GUI结构和状态	标准U操作	使用自然语言测试用例自动化软件测试流程	应用GPT-3的语言理解能力于基于GUI的软件测试，通过基于文本的测试用例描述实现自然交互。	https: //github.com/ neuroevolution% 2Dai/ SoftwareTestingLanguageModel:
Daniel 和 Anne 463]	通用测试	网页平台	GPT-4	HTML DOM 结构	标准I操作	自动化GUI测试以提升分支覆盖率和效率	利用GPT-4的自然语言理解和推理能力执行端到端GUI测试。	https: //github.com/ Software TestingLLMs/ WebtestingWithLLMs
GPTDroid [125]	通用测试	移动 Android	GPT-3.5	UI视图层级文件	标准UI操作及复合动作	自动化GUI测试以提升测试覆盖率并高效发现缺陷	将GUI测试形式化为问答任务，利用大型语言模型(LLM)能力实现类人交互。	https: //github.com/ franklinbill GPTDroid
DROID-AGENT 464]	通用测试	移动 Android	GPT-3.5, GPT-4	GUI状态的JSON表示	标准UI操作、高级API及自定义动作	基于语义和意图驱动的GUI测试自动化	基于应用特定功能，自动生成并执行高层次、真实场景的Android GUI测试任务。	https://github.com/coinse/ droidagent
AUITest-Agent 465]	通用测试	移动 Android	GPT-4	GUI截图、UI层级文件、基于计算机视觉的增强技术如Vision-UI	标准操作	GUI的自动化功能测试	具备动态代理组织以支持步骤导向测试及多源数据提取策略，实现精准功能验证。	https://github.com/bz-lab/ AUITestAgent
VisionDroid 466]	通用测试	移动 Android	GPT-4	带注释边界框的GUI截图，视图层级文件	标准UI操作	识别崩溃缺陷	结合视觉驱动的提示和GUI文本对齐，利用视觉语言模型增强对GUI上下文和应用逻辑的理解。	https: //github.com/ testtestA6/ VisionDroid
AXNav 467]	无障碍测试	iOS移动设备	GPT-4	GUI截图、UI元素检测模型及光学字符识别(OCR)	手势操作、截图捕捉及突出潜在无障碍问题	自动化无障碍测试流程，包括测试VoiceOver、动态字体、粗体文本和按钮形状等功能	适应自然语言测试指令，生成带注释视频以视觉化和交互式地审查无障碍测试结果。	/
LLMigrate 473]	通用测试	移动 Android	GPT-4o	DOM和截图	标准UI操作	自动化基于使用情况的UI测试在Android应用间的迁移	利用多模态大语言模型（LLMs）执行UI测试迁移，无需访问源代码	/

ProphetAgent [482] introduces a novel approach to LLM-powered GUI testing by automatically synthesizing Android application test scripts from natural language descriptions. Departing from previous methods that directly apply LLMs to GUI screenshots or app behaviors, ProphetAgent builds a Clustered UI Transition Graph (CUTG) enriched with semantic annotations. This structured representation enables more accurate mapping between natural language test steps and GUI operations, leading to significant

improvements in completion rate (78.1%) and action accuracy (83.3%). The system employs a dual-agent architecture: SemanticAgent handles semantic annotation, while GenerationAgent generates executable scripts. ProphetAgent demonstrates strong scalability and real-world applicability—reducing tester workload by over 70% at ByteDance. Its performance underscores the effectiveness of combining LLMs with explicit semantic knowledge graphs in GUI-based environments.
ProphetAgent [482] 提出了一种利用大型语言模型（LLM）驱动的图形用户界面（GUI）测试的新方法，通过自然语言描述自动合成安卓应用测试脚本。不同于以往直接将LLM应用于GUI截图或应用行为的方法，ProphetAgent 构建了一个带有语义注释的聚类UI转换图（Clustered UI Transition Graph，CUTG）。这种结构化表示实现了自然语言测试步骤与GUI操作之间更精准的映射，显著提升了完成率（78.1%）和操作准确率（83.3%）。系统采用双代理架构：SemanticAgent负责语义注释，GenerationAgent负责生成可执行脚本。ProphetAgent 展现了强大的可扩展性和实际应用价值——在字节跳动减少了超过70%的测试人员工作量。其性能凸显了将LLM与显式语义知识图结合应用于基于GUI环境的有效性。

AUITestAgent extended the capabilities of LLM-powered GUI testing by bridging natural language-driven requirements and GUI functionality [465]. Employing multi-modal analysis and dynamic agent organization, it efficiently executes both simple and complex testing instructions. This framework highlights the value of combining multi-source data extraction with robust language models to automate functional testing in commercial apps. Incorporating vision-based methods, VisionDroid redefined GUI testing by aligning screenshots with textual contexts to detect non-crash bugs [466]. This innovation ensures application reliability by identifying logical inconsistencies and exploring app functionalities that conventional methods often overlook.
AUITestAgent 扩展了LLM驱动GUI测试的能力，通过连接自然语言驱动的需求与GUI功能[465]。采用多模态分析和动态代理组织，能够高效执行简单和复杂的测试指令。该框架强调了结合多源数据提取与强大语言模型在商业应用功能测试自动化中的价值。VisionDroid 结合基于视觉的方法，通过将截图与文本上下文对齐，重新定义了GUI测试以检测非崩溃类缺陷[466]。这一创新通过识别逻辑不一致性和探索传统方法常忽视的应用功能，保障了应用的可靠性。

Accessibility testing has also benefited from LLM-powered agents. AXNav addresses challenges in iOS accessibility workflows, automating tests for features like VoiceOver and Dynamic Type using natural language instructions and pixel-based models. Its ability to generate annotated videos for interactive review positions AXNav as a scalable and user-friendly solution for accessibility testing [467].
辅助功能测试同样受益于LLM驱动的代理。AXNav 针对iOS辅助功能工作流中的挑战，利用自然语言指令和基于像素的模型自动化测试VoiceOver和动态字体等功能。其生成带注释的视频以供交互式审查的能力，使AXNav成为一种可扩展且用户友好的辅助功能测试解决方案[467]。

TABLE 41: Overview of GUI-testing with LLM-powered GUI agents (Part II).
表41：基于LLM驱动GUI代理的GUI测试概述（第二部分）。

Project	Category	Platform	Model	Perception	Action	Scenario	Highlight	Link
Cui et al., 468]	Test input generation	Mobile Android	GPT-3.5, GPT-4	GUI structures and contextual information	Entering text inputs	Generating and validating text inputs for Android applications	Demonstrates the effectiveness of various LLMs in generating context-aware text inputs, improving UI test coverage, and identifying previously unreported bugs.	/
QTypist 469]	Test input generation	Mobile Android	GPT-3	UI hierarchy files	Generates semantic text inputs	Automates mobile GUI testing by generating appropriate text inputs	Formulates text input generation as a cloze-style fill-in-the-blank language task.	/
Crash-Translator 470]	Bug replay	Mobile Android	GPT-3	Crash-related stack trace information and GUI structure	Standard UI operations	Automates the reproduction of mobile application crashes	Leverages LLMs for iterative GUI navigation and crash reproduction from stack traces, integrating a reinforcement learning-based scoring system to optimize exploration steps.	https://github.com/wuchiuwong/Crash Translator
AdbGPT 471]	Bug replay	Mobile Android	GPT-3.5	GUI structure and hierarchy	Standard U operations	Automates bug reproduction extracting S2R (Steps to Reproduce) entities	Combines prompt engineering with few-shot learning and chain-of-thought reasoning to leverage LLMs for GUI-based tasks.	https://github.com/sidongfeng/AdbGPT
MagicWand 472]	Verrification	Mobile Android	GPT-4V	UI screen-shots and hierarchical UI control tree	Standard U operations	Automates the verification of "How-to" instructions from a search engine	Features a three-stage process: extracting instructions, executing them in a simulated environment, and reranking search results based on execution outcomes.	/
UXAgent 474	Usability testing for web design	Web	Self-designed	Simplified HTML representations	Standard L operations	Automated usability testing of web applications	ables LLM-powered automated usability testing by simulating thousands of user interactions. collecting both qualitative and quantitative data, and providing researchers with early feedback before real-user studies.	https://uxagent.hailab.io
Guardian 475	GUI Testing	Mobile Android	GPT-3.5	GUI structure, Properties	Standard L operations	Autonomously explores mobile applications. interacting with the UI to validate functionalities.	Improves LLM-driven UI testing by offloading planning tasks to an external runtime system.	/
Test-Agent 476	GUI Testing	Android, iOS, Harmony OS	Not Mentioned	GUI screenshots, UI structure information	Standard 【 operations	Cross-platform mobile testing	Eliminates the need for pre-written test scripts by leveraging LLMs and multimodal perception to generate and execute test cases automatically.	/

VLM-Fuzz 477]	GUI Testing	Android (Mobile)	GPT-4o	GUI screenshots and structure information	Standard U operations. system-level actions	Automated Android testing detection of crashes and bugs	Integrates vision-language reasoning with heuristic-based depth-first search (DFS) to systematically explore complex Android UIs, achieving significantly higher code coverage	/

项目	类别	平台	模型	感知	动作	场景	亮点	链接
Cui 等人，468]	测试输入生成	移动端 Android	GPT-3.5，GPT-4	GUI 结构与上下文信息	输入文本	为 Android 应用生成并验证文本输入	展示了多种大型语言模型（LLMs）在生成上下文感知文本输入、提升界面测试覆盖率及发现未报告缺陷方面的有效性。	/
QTypist 469]	测试输入生成	移动端 Android	GPT-3	UI 层级文件	生成语义文本输入	通过生成合适的文本输入实现移动 GUI 测试自动化	将文本输入生成任务形式化为填空式语言任务（cloze-style）。	/
Crash-Translator 470]	缺陷重现	移动端 Android	GPT-3	与崩溃相关的堆栈跟踪信息及 GUI 结构	标准 UI 操作	实现移动应用崩溃的自动重现	利用大型语言模型（LLMs）进行迭代式 GUI 导航和基于堆栈跟踪的崩溃重现，结合基于强化学习的评分系统优化探索步骤。	https://github.com/wuchiuwong/ Crash Translator
AdbGPT 471]	缺陷重现	移动端 Android	GPT-3.5	GUI 结构与层级	标准 UI 操作	自动化缺陷重现，提取重现步骤（S2R）实体	结合提示工程、少量示例学习及链式思维推理，利用大型语言模型（LLMs）完成基于 GUI 的任务。	https://github.com/sidongfeng/ AdbGPT
MagicWand 472]	验证	移动端 Android	GPT-4V	UI 截图及层级 UI 控件树	标准 UI 操作	自动验证搜索引擎中的“操作指南”说明	采用三阶段流程：提取指令、在模拟环境中执行、并根据执行结果重新排序搜索结果。	/
UXAgent 474	网页设计的可用性测试	网页	自主设计	简化的 HTML 表示	标准 UI 操作	网页应用的自动化可用性测试	通过模拟数千次用户交互，收集定性和定量数据，利用大型语言模型（LLMs）实现自动化可用性测试，为研究人员在真实用户研究前提供早期反馈。	https://uxagent.hailab.io
Guardian 475	GUI 测试	移动端 Android	GPT-3.5	GUI 结构，属性	标准 UI 操作	自主探索移动应用，交互界面以验证功能。	通过将规划任务外包给外部运行时系统，提升基于大型语言模型（LLM）的界面测试效果。	/
Test-Agent 476	GUI 测试	Android、iOS、Harmony OS	未提及	GUI 截图，UI 结构信息	标准 UI 操作	跨平台移动测试	通过利用大型语言模型（LLMs）和多模态感知，自动生成并执行测试用例，免除预先编写测试脚本的需求。	/
VLM-Fuzz 477]	GUI 测试	安卓（移动端）	GPT-4o	图形用户界面截图及结构信息	标准U操作。系统级动作	自动化安卓测试中的崩溃和缺陷检测	结合视觉-语言推理与基于启发式的深度优先搜索（DFS），系统性地探索复杂安卓用户界面，实现显著更高的代码覆盖率	/

20.2.3 10.1.2 Text Input generation

20.2.4 10.1.2 文本输入生成

In the realm of text input generation, Cui et al., demonstrated how GPT-3.5 and GPT-4 could enhance Android app testing by generating context-aware text inputs for UI fields [468]. By systematically evaluating these inputs across multiple apps, they revealed the potential of LLMs in improving test coverage and detecting unique bugs with minimal manual intervention. Similarly, QTypist formulated text input generation as a fill-in-the-blank task, leveraging LLMs to improve activity and page coverage by up to 52% [469].

在文本输入生成领域，崔等人展示了GPT-3.5和GPT-4如何通过为UI字段生成上下文感知的文本输入来增强Android应用测试[468]。通过对多个应用系统地评估这些输入，他们揭示了大型语言模型（LLMs）在提升测试覆盖率和以最小人工干预检测独特缺陷方面的潜力。同样，QTypist将文本输入生成视为填空任务，利用LLMs将活动和页面覆盖率提升了最多52%[469]。

20.2.5 10.1.3 Bug Replay

20.2.6 10.1.3 缺陷重现

For bug reproduction, CrashTranslator automated the reproduction of crashes from stack traces by integrating reinforcement learning with LLMs. Its iterative navigation and crash prediction steps significantly reduced debugging time and outperformed state-of-the-art methods [470]. Meanwhile, AdbGPT demonstrated how few-shot learning and chain-of-thought reasoning could transform textual bug reports into actionable GUI operations. By dynamically inferring GUI actions, AdbGPT provided an efficient and lightweight solution for bug replay [471].
在缺陷重现方面，CrashTranslator通过将强化学习与LLMs结合，实现了从堆栈跟踪自动重现崩溃。其迭代导航和崩溃预测步骤显著缩短了调试时间，且优于最先进的方法[470]。与此同时，AdbGPT展示了如何利用少样本学习和链式思维推理，将文本缺陷报告转化为可执行的GUI操作。通过动态推断GUI动作，AdbGPT提供了一种高效且轻量级的缺陷重现解决方案[471]。

BugCraft 478 leverages LLM-powered GUI agents to automate bug reproduction in games, specifically targeting the open-ended and complex environment of Minecraft. It employs GPT-40 as the inference engine, integrating textual bug reports, visual GUI understanding through OmniParser 184 , and external knowledge from the Minecraft Wiki to generate and execute structured reproduction steps. Actions are carried out via a custom Macro API, enabling robust interaction with both the game's GUI and environment. BugCraft's ability to translate unstructured bug descriptions into executable in-game behaviors highlights the strong potential of vision-enhanced LLM agents for advancing software testing and debugging.
BugCraft 478 利用基于LLM的GUI代理自动化游戏中的缺陷重现，特别针对Minecraft这一开放且复杂的环境。它采用GPT-40作为推理引擎，整合文本缺陷报告、通过OmniParser 184实现的视觉GUI理解以及来自Minecraft Wiki的外部知识，生成并执行结构化的重现步骤。操作通过定制的宏API执行，实现了与游戏GUI和环境的稳健交互。BugCraft将非结构化缺陷描述转化为可执行的游戏内行为，凸显了视觉增强型LLM代理在推动软件测试与调试方面的强大潜力。

TABLE 42: Overview of GUI-testing with LLM-powered GUI agents (Part III).
表42：基于LLM的GUI代理进行GUI测试概览（第三部分）。

Project	Category	Platform	Model	Perception	Action	Scenario	Highlight	Link
BugCraft 478]	Bug Reproduction	Windows Computer	BugCraft based on GPT-40	GUI screenshots	Standard UI operations	Automatically reproduces crash bugs in Minecraft by reading user-submitted bug reports, generating structured steps, and executing them to cause a crash	First end-to-end framework that automates crash bug reproduction in a complex open-world game (Minecraft) using LLM-driven agents, vision-based UI parsing, and structured action execution	https:// bugcraft2025. github.io/
ReuseDroic 479]	GUI Testing	Mobile Android	ReuseDroid based on GPT-40	GUI screenshots and widget properties	Standard UI operations	Migrates GUI test cases between Android a that share similar functionality but differ in operational logic	Leverages visual contexts and dynamic feedback mechanisms to significantly boost migration success rates compared to prior mapping- and LLM-based methods	/
SeeAct-ATA and PinATA 480]	GUI Testing	Web	SeeAct [17]	GUI structure and DOM	Standard U operations	Automates manual to-end (E2E) web application testing	First open-source attempt to adapt LLM-powered Autonomous Web Agents into Autonomous Test Agents (ATA) for web testing	/
GERALLT 481]	GUI Testing	Desktop (Windows/Linux	GPT-40	GUI screenshots and structure information	Standard L operations	Finds unintuitive behavior, inconsistencies, and functional errors in GUIs without predefined test scripts	Pioneers LLM-driven testing on real-world desktop GUI applications (not web or mobile), combining structured GUI parsing with LLM-based control and evaluation	https://github.com/DLR-SC/ GERALLT

ProphetAge 482]	gentGUI Testing	Android Mobile	and Gener-ationAgent using foundation models (GPT-40)	SemanticAgent XML UI trees	Executable UI test scripts	Automates GUI test case generation from natural language for regression and compatibility testing in mobile apps	Innovatively combines LLM reasoning with a semantically enriched GUI graph (CUTG), significantly improving GUI test synthesis performance and efficiency over state-of-the-art tools	https: //github.com/ prophetagent/ Home
Agent for User 483]	GUI Testing	Android Mobile	GPT-4	XML view hierarchy	Standard operations	Automated testing of multiuser interactive features	Introduces a multi-agent LLM framework where each agent simulates a user on a virtual device	/

项目	类别	平台	模型	感知	动作	场景	亮点	链接
BugCraft 478]	缺陷复现	Windows 电脑	基于 GPT-40 的 BugCraft	GUI 截图	标准用户界面操作	通过读取用户提交的缺陷报告，生成结构化步骤并执行，从而自动复现 Minecraft 中的崩溃缺陷	首个端到端框架，利用大型语言模型（LLM）驱动的代理、基于视觉的用户界面解析和结构化动作执行，实现复杂开放世界游戏（Minecraft）中崩溃缺陷的自动复现	https:// bugcraft2025.github.io/
ReuseDroic 479]	GUI 测试	安卓移动端	基于 GPT-40 的 ReuseDroid	GUI 截图及控件属性	标准用户界面操作	在功能相似但操作逻辑不同的 Android 应用间迁移 GUI 测试用例	利用视觉上下文和动态反馈机制，显著提升迁移成功率，优于以往基于映射和大型语言模型的方法	/
SeeAct-ATA 和 PinATA 480]	GUI 测试	网页	SeeAct [17]	GUI 结构和 DOM	标准用户操作	自动化手动端到端（E2E）网页应用测试	首个开源尝试，将大型语言模型驱动的自主网页代理（Autonomous Web Agents）转化为自主测试代理（ATA）用于网页测试	/
GERALLT 481]	GUI 测试	桌面（Windows/Linux）	GPT-40	GUI 截图及结构信息	标准用户操作	无需预定义测试脚本，发现 GUI 中不直观的行为、不一致性及功能错误	开创性地将大型语言模型驱动的测试应用于真实桌面 GUI 应用（非网页或移动端），结合结构化 GUI 解析与基于 LLM 的控制和评估	https://github.com/DLR-SC/ GERALLT
ProphetAge 482]	智能 GUI 测试	安卓移动端	基于基础模型（GPT-40）的生成代理	语义代理 XML UI 树	可执行的 UI 测试脚本	自动从自然语言生成 GUI 测试用例，用于移动应用的回归和兼容性测试	创新性地结合大型语言模型推理与语义丰富的 GUI 图（CUTG），显著提升 GUI 测试合成的性能和效率，优于现有先进工具	https: //github.com/prophetagent/ Home
用户代理 483]	GUI 测试	安卓移动端	GPT-4	XML 视图层级	标准操作	多用户交互功能的自动化测试	引入多代理大型语言模型框架，每个代理模拟虚拟设备上的用户	/

20.2.7 10.1.4 Verification

20.2.8 10.1.4 验证

Finally, as a novel application in testing, MagicWand showcased the potential of LLMs in automating "How-to" verifications. By extracting, executing, and refining instructions from search engines, it addressed critical challenges in user-centric task automation, improving the reliability of GUI-driven workflows 472
最后，作为测试中的一种新颖应用，MagicWand展示了大型语言模型（LLM）在自动化“操作指南”验证中的潜力。通过从搜索引擎提取、执行和优化指令，它解决了以用户为中心的任务自动化中的关键挑战，提高了基于图形用户界面（GUI）工作流的可靠性 472

In summary, LLM-powered GUI agents have revolutionized GUI testing by introducing natural language-driven methods, vision-based alignment, and automated crash reproduction. These innovations have enhanced test coverage, efficiency, and accessibility, setting new benchmarks for intelligent GUI testing frameworks.
总之，基于LLM的GUI代理通过引入自然语言驱动的方法、基于视觉的对齐和自动化崩溃复现，革新了GUI测试。这些创新提升了测试覆盖率、效率和可访问性，为智能GUI测试框架树立了新的标杆。

20.3 10.2 Virtual Assistants

20.4 10.2 虚拟助手

Virtual assistants, such as Sir 32 are AI-driven applications that help users by performing tasks, answering questions, and executing commands across various platforms, including web browsers, mobile phones, and computers. Initially, these assistants were limited to handling simple commands via voice or text input, delivering rule-based responses or running fixed workflows similar to RPA. They focused on basic tasks, such as setting alarms or checking the weather.
虚拟助手，如Siri 32，是基于人工智能的应用程序，帮助用户执行任务、回答问题并在各种平台（包括网页浏览器、手机和电脑）上执行命令。最初，这些助手仅限于通过语音或文本输入处理简单命令，提供基于规则的响应或运行类似机器人流程自动化（RPA）的固定工作流，主要处理诸如设置闹钟或查询天气等基础任务。

With advancements in LLMs and agents, virtual assistants have evolved significantly. They now support more complex, context-aware interactions on device GUIs through textual or voice commands and provide personalized responses, catering to diverse applications and user needs on various platforms. This progression has transformed virtual assistants from basic utilities into intelligent, adaptive tools capable of managing intricate workflows and enhancing user productivity across platforms. Figure 30 presents a conceptual example of a GUI agent-powered virtual assistant on a smartphone [33] . In this scenario, the agent enables users to interact through chat, handling tasks such as setting up a screenshot shortcut on their behalf. This feature is particularly beneficial for users unfamiliar with the phone's functionalities, simplifying complex tasks into conversational commands.
随着LLM和代理技术的发展，虚拟助手显著进化。它们现在支持通过文本或语音命令在设备GUI上进行更复杂、具上下文感知的交互，并提供个性化响应，满足各种平台上多样化的应用和用户需求。这一进步使虚拟助手从基础工具转变为智能、适应性强的工具，能够管理复杂工作流并提升跨平台用户生产力。图30展示了基于GUI代理的智能手机虚拟助手的概念示例[33]。在此场景中，代理允许用户通过聊天交互，处理如为用户设置截图快捷方式等任务。该功能对不熟悉手机功能的用户尤为有益，将复杂任务简化为对话式命令。

---

32. https://www.apple.com/siri/
33. https://www.apple.com/siri/

33. The application and scenario depicted in the figure are conceptual and fabricated. They do not reflect the actual functionality of any specific smartphone. Readers should consult the phone manual or official guidance for accurate information on AI assistant capabilities.
34. 图中所示的应用和场景为概念性和虚构内容，不反映任何特定智能手机的实际功能。读者应参考手机手册或官方指导以获取关于AI助手功能的准确信息。

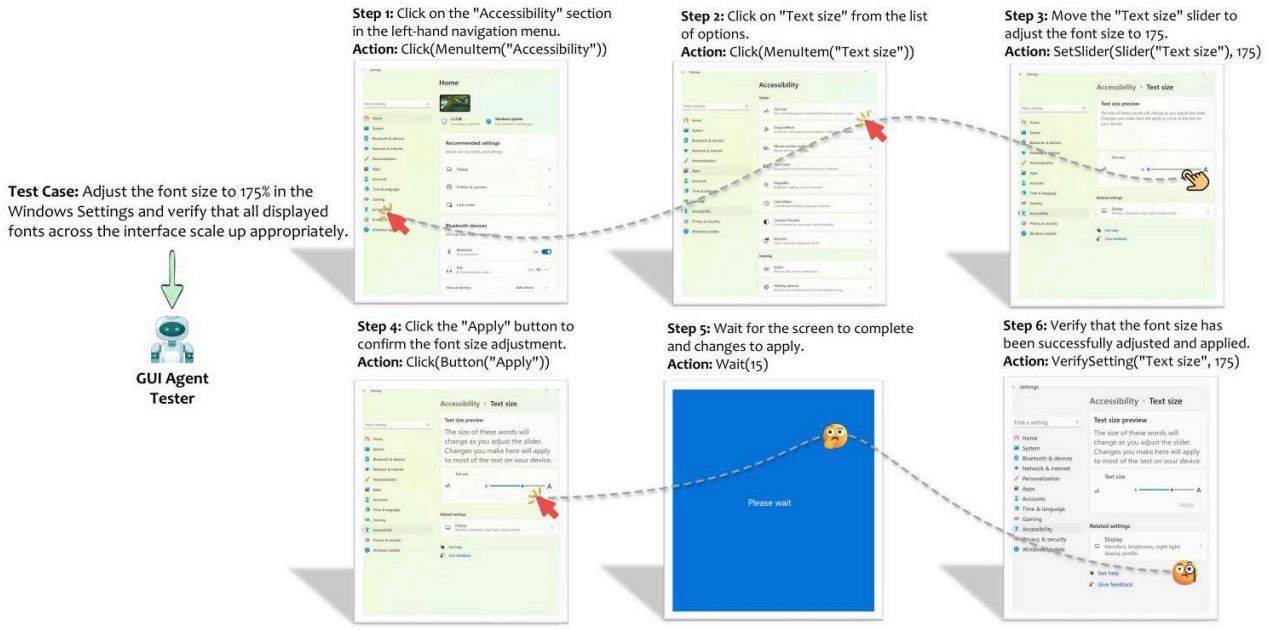

Fig. 29: An example of testing font size adjustment using an LLM-powered GUI agent.
图29：使用基于LLM的GUI代理测试字体大小调整的示例。

To explore more real-world applications of virtual assistants powered by GUI agents, we provide an overview of advancements across research, open-source initiatives, and production-level applications, as summarized in Table 43 and 44
为了探讨更多基于GUI代理的虚拟助手在现实中的应用，我们概述了研究、开源项目和生产级应用的进展，详见表43和44。

20.4.1 10.2.1 Research

20.4.2 10.2.1 研究

Recent research efforts have significantly advanced the capabilities of virtual assistants by integrating LLM-powered GUI agents, enabling more intelligent and adaptable interactions within various applications.
近期研究通过整合基于LLM的GUI代理，显著提升了虚拟助手的能力，实现了更智能、更具适应性的多应用交互。

Firstly, the integration of LLMs into GUI-based automation has been explored to enhance business process automation. For instance, [485] introduces Agentic Process Automation through the development of ProAgent, which automates both the creation and execution of workflows in GUI environments. By utilizing agents like ControlAgent and DataAgent, it supports complex actions such as dynamic branching and report generation in applications like Slack and Google Sheets. This approach transcends traditional RPA by enabling flexible, intelligent workflows, significantly reducing the need for manual intervention and highlighting the transformative potential of LLM-powered agents in virtual assistants.
首先，LLM与基于GUI的自动化集成被用于增强业务流程自动化。例如，[485]提出了通过ProAgent开发的Agentic流程自动化，能够自动创建和执行GUI环境中的工作流。通过使用ControlAgent和DataAgent等代理，它支持Slack和Google Sheets等应用中的动态分支和报告生成等复杂操作。这种方法超越了传统RPA，实现了灵活智能的工作流，显著减少了人工干预，凸显了基于LLM代理在虚拟助手中的变革潜力。

Building upon the idea of integrating LLMs with GUI environments, researchers have focused on mobile platforms to automate complex tasks. LLMPA [486] is a pioneering framework that leverages LLMs to automate multi-step tasks within mobile applications like Alipay. It interacts directly with app GUIs, mimicking human actions such as clicks and typing, and employs UI tree parsing and object detection for precise environment understanding. A unique controllable calibration module ensures logical action execution, demonstrating the potential of LLM-powered virtual assistants to handle intricate workflows and real-world impact in assisting users with diverse tasks.
基于将LLM与GUI环境集成的理念，研究者聚焦于移动平台以自动化复杂任务。LLMPA [486]是一个开创性框架，利用LLM自动化支付宝等移动应用中的多步骤任务。它直接与应用GUI交互，模拟点击和输入等人类操作，并通过UI树解析和对象检测实现精确环境理解。独特的可控校准模块确保逻辑动作执行，展示了基于LLM的虚拟助手处理复杂工作流和实际应用的潜力。

Similarly, the automation of smartphone tasks through natural language prompts has been addressed by PromptRPA [490]. Utilizing a multi-agent framework, it automates tasks within smartphone GUI environments, tackling challenges like interface updates and user input variability. Advanced perception methods, including OCR and hierarchical GUI analysis, are employed to understand and interact with mobile interfaces. By supporting real-time feedback and iterative improvements, PromptRPA underscores the importance of user-centered design in LLM-driven virtual assistants.
类似地，PromptRPA [490]通过自然语言提示实现了智能手机任务的自动化。该多代理框架自动化智能手机GUI环境中的任务，应对界面更新和用户输入多样性等挑战。采用OCR和分层GUI分析等先进感知方法理解并交互移动界面。通过支持实时反馈和迭代改进，PromptRPA强调了以用户为中心设计在LLM驱动虚拟助手中的重要性。

In the realm of accessibility, LLM-powered GUI agents have been instrumental in enhancing user experience for individuals with disabilities. For example, VizAbility [484] enhances the accessibility of data visualizations for blind and low-vision users. By combining structured chart navigation with LLM-based conversational interactions, users can ask natural language queries and receive insights on chart content and trends. Leveraging frameworks like OII [34] and chart specifications such as Vega-Lite 35 VizAbility allows exploration of visual data without direct visual perception, addressing real-world accessibility challenges in GUIs.
在无障碍领域，基于LLM的GUI代理在提升残障用户体验方面发挥了重要作用。例如，VizAbility [484]增强了盲人和低视力用户对数据可视化的无障碍访问。通过结合结构化图表导航和基于LLM的对话交互，用户可用自然语言查询并获得图表内容和趋势的洞察。利用OII [34]框架和Vega-Lite 35等图表规范，VizAbility使用户无需直接视觉感知即可探索视觉数据，解决了GUI中的实际无障碍挑战。

Furthermore, addressing the needs of older adults, EasyAsk [491] serves as a context-aware in-app assistant that enhances usability for non-technical users. By integrating multi-modal inputs, combining natural voice queries and touch interactions with GUI elements, it generates accurate and contextual tutorial searches. EasyAsk demonstrates how GUI agents can enhance accessibility by integrating contextual information and interactive tutorials, empowering users to navigate smartphone functions effectively.
此外，为满足老年人的需求，EasyAsk [491] 作为一种具备上下文感知能力的应用内助手，提升了非技术用户的可用性。通过整合多模态输入，结合自然语音查询与触控交互及图形用户界面（GUI）元素，它能够生成准确且具上下文相关性的教程搜索。EasyAsk 展示了图形用户界面代理如何通过整合上下文信息和交互式教程来增强无障碍性，使用户能够有效地操作智能手机功能。

34. https://mitvis.github.io/olli/
35. https://mitvis.github.io/olli/

35. https://vega.github.io/
36. https://vega.github.io/

JOURNAL OF LATEX CLASS FILES, DECEMBER 2024

TABLE 43: Overview of virtual assistants with LLM-powered GUI agents (Part I).
表43：基于大型语言模型（LLM）驱动的图形用户界面代理虚拟助手概览（第一部分）。

Project	Type	Platform	Model	Perception	Action	Scenario	Highlight	Link
ProAgent 485]	Research	Web and Desktop	GPT-4	Task descriptions and structured application data	Standard UI operations and dynamic branching	Automates business processes such as data analysis, report generation, and notifications via GUI-based tools	Introduces dynamic workflows where agents interpret and execute tasks flexibly, surpassing traditional RPA systems	https: //github.com/ OpenBMB/ ProAgent
LLMPA 486]	Research	Mobile (Android)	AntLLM-10b	UI tree structures, visual modeling, and text extraction modules	Standard UI operations	Automates user interactions within mobile apps, such as ticket booking	Integrates LLM reasoning capabilities with a modular design that supports task decomposition, object detection, and robust action prediction in GUI environments	/
VizAbility 484]	Research	Desktop	GPT-4V	Keyboard-navigable tree views	Navigates chart structures and generates answers	Assists blind and low-vision users in exploring and understanding data visualizations	Integrates structured chart navigation with LLM-powered conversational capabilities, enabling visually impaired users to query in natural language	https://dwr.bc.edu/vizability/
GPTVoice-Tasker 487]	Research	Mobile (Android)	GPT-4	Android Accessibility Tree	Standard UI operations	Automates user interactions on mobile devices through voice commands	Integrates LLMs for natural command interpretation and real-time GUI interactions, using a graph-based local database to record and replicate interactions	https: //github.com/ vuminhduc796 GPTVoiceTasker
AutoTask 488]	Research	Mobile (Android)	GPT-4	Android Accessibility Tree	Standard UI operations	Automates multistep tasks on mobile devices	Operates without predefined scripts or configurations, autonomously exploring GUI environments	https: //github.com/ BowenBryanWan AutoTask
AssistEditor 489]	Research	Windows	UniVTG 544]	GUI elements, user requirements, and video data	Standard UI operations	Automates video editing workflows	Employs a multi-agent collaboration framework where agents specialize in roles to integrate user requirements into video editing workflows	7

PromptRPA 490]	Research	Mobile (Android)	GPT- 4 and GPT-3.5 Turbo	Layout hierarchy and screenshots with OCR	Standard U operations and application-level functionalities	Automates smartphone tasks and creates interactive tutorials	Integrates user feedback loops for continuous improvement, addressing interface evolution and task variability	/
EasyAsk 491]	Research	Mobile (Android)	GPT-4	Android Accessibility Tree	Highlights specific UI elements for user interaction	Assists older adults in learning and navigating smartphone functions through in-app interactive tutorials	Combines voice and touch inputs, supplementing incomplete or ambiguous queries with in-app contextual information	/
WebNav 497]	Research	Web	Gemini 2.0 Flash Thinking	Standard UI operations	GUI screenshots and DOM	Assistive technology fo visually impaired users, enabling voice-based navigation complex websites	Combines a ReAct-style reasoning loop, real-time DOM labeling, and voice-driven interaction to support intelligent web navigation for visually impaired users	/

项目	类型	平台	模型	感知	动作	场景	亮点	链接
ProAgent 485]	研究	网页和桌面	GPT-4	任务描述和结构化应用数据	标准用户界面操作和动态分支	通过基于图形用户界面（GUI）的工具自动化业务流程，如数据分析、报告生成和通知	引入动态工作流，代理能够灵活解释和执行任务，超越传统的机器人流程自动化（RPA）系统	https: //github.com/ OpenBMB/ ProAgent
LLMPA 486]	研究	移动端（安卓）	AntLLM-10b	用户界面树结构、可视化建模和文本提取模块	标准用户界面操作	自动化移动应用中的用户交互，如票务预订	结合大型语言模型（LLM）推理能力，采用模块化设计，支持任务分解、对象检测和图形用户界面环境中的稳健动作预测	/
VizAbility 484]	研究	桌面端	GPT-4V	键盘可导航的树视图	导航图表结构并生成答案	帮助盲人和低视力用户探索和理解数据可视化	结合结构化图表导航与大型语言模型驱动的对话功能，使视障用户能够以自然语言查询	https://dwr.bc.edu/vizability/
GPTVoice-Tasker 487]	研究	移动端（安卓）	GPT-4	安卓辅助功能树	标准用户界面操作	通过语音命令自动化移动设备上的用户交互	集成大型语言模型，实现自然命令解析和实时图形用户界面交互，使用基于图的本地数据库记录并复现交互	https: //github.com/ vuminhduc796 GPTVoiceTasker
AutoTask 488]	研究	移动端（安卓）	GPT-4	安卓辅助功能树	标准用户界面操作	自动化移动设备上的多步骤任务	无需预定义脚本或配置，能够自主探索图形用户界面环境	https: //github.com/ BowenBryanWan AutoTask
AssistEditor 489]	研究	Windows	UniVTG 544]	图形用户界面元素、用户需求和视频数据	标准用户界面操作	自动化视频编辑工作流程	采用多代理协作框架，代理各司其职，将用户需求整合进视频编辑流程	7
PromptRPA 490]	研究	移动端（安卓）	GPT-4 和 GPT-3.5 Turbo	布局层级和带有光学字符识别（OCR）的截图	标准用户界面操作和应用级功能	自动化智能手机任务并创建交互式教程	集成用户反馈循环，实现持续改进，应对界面演变和任务多样性	/
EasyAsk 491]	研究	移动端（安卓）	GPT-4	安卓辅助功能树	突出显示特定用户界面元素以便用户交互	通过应用内交互式教程帮助老年人学习和使用智能手机功能	结合语音和触摸输入，利用应用内上下文信息补充不完整或模糊的查询	/
WebNav 497]	研究	网页	Gemini 2.0 快速思考	标准用户界面操作	图形用户界面截图和文档对象模型（DOM）	为视障用户提供辅助技术，实现基于语音的复杂网站导航	结合ReAct风格的推理循环、实时DOM标注和语音驱动交互，支持视障用户智能网页导航	/

Voice interaction has also been a focus area, with tools like GPTVoiceTasker [487] facilitating hands-free interaction with Android GUIs through natural language commands. It bridges the gap between voice commands and GUI-based actions using real-time semantic extraction and a hierarchical representation of UI elements. By automating multi-step tasks and learning from user behavior, it enhances task efficiency and reduces cognitive load, highlighting the transformative potential of LLMs in improving accessibility and user experience in mobile environments.

语音交互也是一个重点领域，诸如GPTVoiceTasker [487]等工具通过自然语言命令实现了对Android图形用户界面（GUI）的免提操作。它利用实时语义提取和UI元素的层次化表示，弥合了语音命令与基于GUI的操作之间的差距。通过自动化多步骤任务并从用户行为中学习，它提升了任务效率，减轻了认知负担，凸显了大型语言模型（LLMs）在提升移动环境中无障碍性和用户体验方面的变革潜力。

Expanding on voice-powered interactions, AutoTask 488 enables virtual assistants to execute multi-step tasks in GUI environments without predefined scripts. It autonomously explores and learns from mobile GUIs, effectively combining voice command interfaces with dynamic action engines to interact with GUI elements. Utilizing trial-and-error and experience-driven learning, AutoTask adapts to unknown tasks and environments, showcasing its potential in enhancing voice-driven virtual assistants for hands-free interactions.

在语音驱动交互的基础上，AutoTask 488使虚拟助手能够在GUI环境中执行多步骤任务，无需预设脚本。它自主探索并学习移动GUI，有效

结合语音命令接口与动态动作引擎以与GUI元素交互。通过试错和经验驱动的学习，AutoTask适应未知任务和环境，展示了其在增强语音驱动虚拟助手实现免提交互方面的潜力。

Finally, in the domain of creative workflows, AssistEdi-tor 489 exemplifies a multi-agent framework for automating video editing tasks. By interacting with GUI environments, it autonomously performs complex workflows using dialogue systems and video understanding models to bridge user intent with professional editing tasks. The innovative use of specialized agents ensures efficient task distribution and execution, demonstrating the practical application of LLM-powered GUI agents in real-world scenarios and expanding automation into creative domains.
最后，在创意工作流程领域，AssistEditor 489展示了一个用于自动化视频编辑任务的多代理框架。通过与GUI环境交互，它利用对话系统和视频理解模型自主执行复杂工作流程，将用户意图与专业编辑任务连接起来。专用代理的创新使用确保了任务的高效分配与执行，展示了基于大型语言模型的GUI代理在现实场景中的实际应用，并将自动化扩展到创意领域。

These research endeavors collectively showcase significant advancements in LLM-powered GUI agents, highlighting their potential to transform virtual assistants into intelligent, adaptable tools capable of handling complex tasks across various platforms and user needs.
这些研究工作共同展示了基于大型语言模型的GUI代理的显著进展，凸显了其将虚拟助手转变为智能、适应性强的工具的潜力，能够处理跨平台和多样用户需求的复杂任务。

20.4.3 10.2.2 Open-Source Projects

20.4.4 10.2.2 开源项目

TABLE 44: Overview of virtual assistants with LLM-powered GUI agents (Part II).
表44：基于大型语言模型的GUI代理虚拟助手概览（第二部分）。

Project	Type	Platform	Model	Perception	Action	Scenario	Highlight	Link
OpenAdapt 492]	Open-source	Desktop	LLM, VLM (e.g., GPT-4, ACT-1)	Screenshots with CV tools for GUI parsing	Standard UI operations	Automates repetitive tasks across industries	Learns task automation by observing user interactions, eliminating manual scripting	https: //github.com/ OpenAdaptAI/ OpenAdapt
AgentSea 493]	Open-source	Desktop and Web	LLM, VLM	Screenshots with CV tools for GUI parsing	Standard UI operations	Automates tasks within GUI environments	Offers a modular toolkit adhering to the UNIX philosophy, allowing developers to create custom AI agents for diverse GUI environments	https://www. agentsea.ai/
Open Interpreter 494]	Open-source	Desktop, Web, Mobile (Android)	LLM	System perception via command-line	Shell commands, code, and native APIs	Automates tasks, conducts data analysis, manages files, and controls web browsers for research	Executes code locally, providing full access to system resources and libraries, overcoming limitations of cloud-based services	https: //github.com/ OpenInterpreter/ open-interpreter
MultiOn 495]	Production	Web	LLM	|	Standard UI operations	Automates web-based tasks	Performs autonomous web actions via natural language commands	https://www.multion.ai/
YOYO Agent in MagicOS 496]	Production	Mobile (Magi-cOS 9.0)	MagicLM	GUI context	Executes in-app and cross-app operations	Automates daily tasks, enhancing productivity	Leverages MagicLM to understand and execute complex tasks across applications, learning user habits to provide personalized assistance	/
Power Automate 132]	Production	Windows	LLM, VLM	Records user interactions with the GUI	Standard UI operations	Automates repetitive tasks and streamlines work-flows	Translates natural language descriptions of desired automations into executable workflows	https://learn.microsoft.com/en-us/ power-automate/ desktop-flows/ create% 2Dflow-using% 2Dai-recorder

Eko 545]	Production	Web browsers and computer environments	ChatGPT and Claude 3.5	Visual-Interactive Element Perception (VIEP) technology for interacting with GUI elements.	Standard UI operations.	Automates tasks by handling diverse workflows.	Decomposes natural language task descriptions into executable workflows, enabling seamless integration of natural language and programming logic in agent design.	https: //eko.fellou.ai/

项目	类型	平台	模型	感知	动作	场景	亮点	链接
OpenAdapt 492]	开源	桌面	大型语言模型(LLM)、视觉语言模型(VLM)（例如，GPT-4，ACT-1）	使用计算机视觉工具对图形用户界面(GUI)进行截图解析	标准用户界面操作	自动化跨行业的重复性任务	通过观察用户交互学习任务自动化，免除手动编写脚本	https: //github.com/ OpenAdaptAI/ OpenAdapt
AgentSea 493]	开源	桌面和网页	大型语言模型(LLM)、视觉语言模型(VLM)	使用计算机视觉工具对图形用户界面(GUI)进行截图解析	标准用户界面操作	在图形用户界面环境中自动化任务	提供遵循UNIX哲学的模块化工具包，允许开发者为多样化的GUI环境创建定制AI代理	https://www. agentsea.ai/
Open Interpreter 494]	开源	桌面、网页、移动端（安卓）	大型语言模型(LLM)	通过命令行实现系统感知	Shell命令、代码及本地API	自动化任务，进行数据分析，管理文件，控制网页浏览器以支持研究	本地执行代码，全面访问系统资源和库，突破云服务的限制	https: //github.com/ OpenInterpreter/ open-interpreter
MultiOn 495]	生产环境	网页	大型语言模型(LLM)	|	标准用户界面操作	自动化基于网页的任务	通过自然语言命令执行自主网页操作	https://www.multion.ai/
MagicOS中的YOYO代理 496]	生产环境	移动端（MagicOS 9.0）	MagicLM	图形用户界面上下文	执行应用内及跨应用操作	自动化日常任务，提高生产力	利用MagicLM理解并执行跨应用复杂任务，学习用户习惯以提供个性化辅助	/
Power Automate 132]	生产环境	Windows	大型语言模型(LLM)、视觉语言模型(VLM)	记录用户与图形用户界面的交互	标准用户界面操作	自动化重复任务，优化工作流程	将自然语言描述的自动化需求转化为可执行的工作流	https://learn.microsoft.com/en-us/ power-automate/ desktop-flows/ create% 2Dflow-using% 2Dai-recorder
Eko 545]	生产环境	网页浏览器及计算机环境	ChatGPT和Claude 3.5	用于交互图形用户界面元素的视觉交互元素感知(VIEP)技术	标准用户界面操作	通过处理多样化工作流实现任务自动化	将自然语言任务描述分解为可执行工作流，实现自然语言与编程逻辑在代理设计中的无缝融合	https: //eko.fellou.ai/

In addition to research prototypes, open-source projects have contributed substantially to the development and accessibility of LLM-brained GUI agents, enabling wider adoption and customization.
除了研究原型，开源项目也为大型语言模型（LLM）驱动的图形用户界面（GUI）代理的发展和普及做出了重要贡献，促进了更广泛的采用和定制。

One such project is OpenAdapt [492], an open-source framework that utilizes large multimodal models to automate tasks by observing and replicating user interactions within GUI environments. It captures screenshots and records user inputs, employing computer vision techniques to understand and execute standard UI operations. Designed to streamline workflows across various industries, OpenAdapt learns from user demonstrations, thereby reducing the need for manual scripting and showcasing adaptability in GUI-based task

automation.
其中一个项目是OpenAdapt [492]，这是一个开源框架，利用大型多模态模型通过观察和复制用户在GUI环境中的交互来自动化任务。它捕捉屏幕截图并记录用户输入，采用计算机视觉技术理解并执行标准的用户界面操作。OpenAdapt旨在简化各行业的工作流程，通过学习用户演示，减少手动脚本编写的需求，展示了在基于GUI的任务自动化中的适应能力。

Similarly, AgentSea [493] offers a comprehensive and modular toolkit for creating intelligent agents that can navigate and interact with various GUI environments across multiple platforms. Its flexibility is particularly beneficial for developing virtual assistants capable of automating complex tasks within applications, enhancing user productivity. By adhering to the UNIX philosophy, AgentSea ensures that each tool is specialized, promoting ease of use and extensibility. Its open-source nature fosters community collaboration and innovation in AI-driven GUI automation.
类似地，AgentSea [493]提供了一个全面且模块化的工具包，用于创建能够在多个平台的各种GUI环境中导航和交互的智能代理。其灵活性特别适合开发能够在应用程序内自动化复杂任务的虚拟助手，从而提升用户生产力。遵循UNIX哲学，AgentSea确保每个工具专注于特定功能，促进易用性和可扩展性。其开源特性促进了社区协作和AI驱动GUI自动化的创新。

Open Interpreter [494] further exemplifies the potential of open-source contributions by leveraging large language models to execute code locally. Users can interact with their computer's GUI through natural language commands, supporting multiple programming languages and operating across various platforms. By facilitating tasks such as data analysis, web automation, and system management, Open Interpreter provides unrestricted access to system resources and libraries, enhancing flexibility and control. Its customization capabilities make it a valuable asset for users aiming to streamline operations through AI-powered virtual assistance.
Open Interpreter [494]进一步展示了开源贡献的潜力，通过利用大型语言模型在本地执行代码。用户可以通过自然语言命令与计算机的GUI交互，支持多种编程语言并跨多个平台运行。Open Interpreter通过支持数据分析、网页自动化和系统管理等任务，提供了对系统资源和库的无限制访问，增强了灵活性和控制力。其定制能力使其成为希望通过AI驱动的虚拟助手简化操作的用户的宝贵工具。

These open-source projects not only advance the state of LLM-powered GUI agents but also democratize access to intelligent virtual assistants, enabling developers and users to tailor solutions to specific needs and applications.
这些开源项目不仅推动了LLM驱动GUI代理的技术进步，也实现了智能虚拟助手的普及，使开发者和用户能够根据具体需求和应用定制解决方案。

20.4.5 10.2.3 Production

20.4.6 10.2.3 生产应用

The integration of LLM-brained GUI agents into production environments demonstrates their practical viability and impact on enhancing user experiences in commercial applications.
将LLM驱动的GUI代理集成到生产环境中，展示了其实际可行性及其在提升商业应用用户体验方面的影响。

Power Automate 132 exemplifies an AI-powered GUI agent that enhances user interaction with desktop applications. By allowing users to describe tasks in natural language while recording actions, it translates these descriptions into automated workflows, effectively bridging the gap between user intent and execution. Its ability to record and replicate user actions within the GUI streamlines the automation of repetitive tasks, making it a valuable tool for increasing efficiency and highlighting advancements in user-friendly automation solutions.
Power Automate 132是一个AI驱动的GUI代理示例，它增强了用户与桌面应用的交互。通过允许用户用自然语言描述任务并记录操作，它将这些描述转化为自动化工作流，有效地弥合了用户意图与执行之间的差距。其在GUI中记录和复制用户操作的能力简化了重复任务的自动化，是提升效率和展示用户友好自动化解决方案进步的宝贵工具。

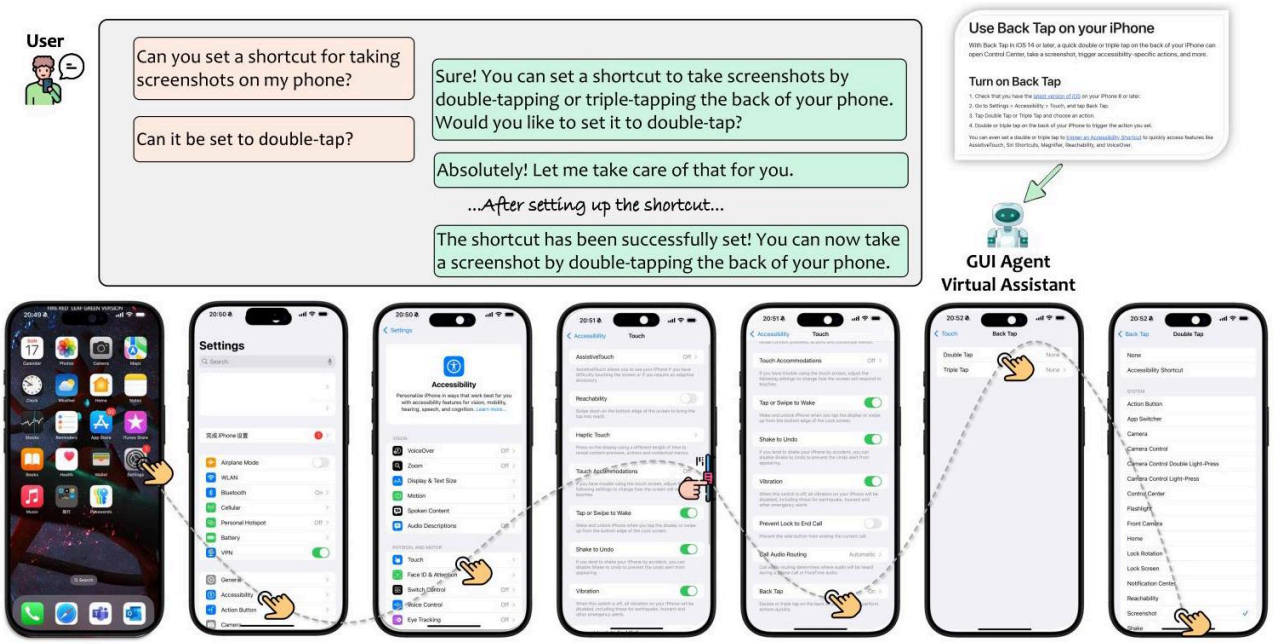

Fig. 30: A conceptual example of a GUI agent-powered virtual assistant on a smartphone.
图30：智能手机上由GUI代理驱动的虚拟助手的概念示例。

In the realm of web interactions, MultiOn [495] serves as a personal AI agent that autonomously interacts with web-based GUIs to execute user-defined tasks. Leveraging large language models, it interprets natural language commands and translates them into precise web actions, effectively automating complex or repetitive tasks. MultiOn's approach to perceiving and manipulating web elements enables seamless functioning across various web platforms, enhancing user productivity and streamlining web interactions.
在网页交互领域，MultiOn [495]作为个人AI代理，能够自主与基于网页的GUI交互以执行用户定义的任务。利用大型语言模型，它解读自然语言命令并将其转化为精确的网页操作，有效自动化复杂或重复的任务。MultiOn对网页元素的感知和操作方法使其能够在各种网页平台上无缝运行，提升用户生产力并简化网页交互。

On mobile platforms, the YOYO Agent in MagicOS [496] exemplifies an LLM-powered GUI agent operating within the MagicOS 9.0 interface. Utilizing Honor's MagicLM, it comprehends and executes user commands across various applications, learning from user behavior to offer personalized assistance. This integration demonstrates how large language models can enhance virtual assistants, enabling them to perform complex tasks within GUI environments and improving user experience and productivity on mobile devices.
在移动平台上，MagicOS [496]中的YOYO Agent是一个LLM驱动的GUI代理，运行于MagicOS 9.0界面。它利用荣耀的MagicLM，理解并执行跨多应用的用户命令，通过学习用户行为提供个性化辅助。该集成展示了大型语言模型如何增强虚拟助手，使其能够在GUI环境中执行复杂任务，提升移动设备上的用户体验和生产力。

Eko 545] serves as a prime example of a versatile and efficient tool for developing intelligent agents capable of interacting with GUIs across various platforms. Its integration with multiple LLMs and the innovative Visual-Interactive Element Perception (VIEP) technology highlight its capability to perform complex tasks through natural language instructions. Eko's comprehensive tool support make it a valuable resource for developers aiming to create customizable and production-ready agent-based workflows. By facilitating seamless interaction within GUI environments, Eko exemplifies the advancements in virtual assistants powered by LLMs.
Eko [545]是一个多功能且高效的工具示例，用于开发能够跨多个平台与GUI交互的智能代理。其集成了多种大型语言模型和创新的视觉交互元素感知（Visual-Interactive Element Perception，VIEP）技术，彰显了通过自然语言指令执行复杂任务的能力。Eko对工具的全面支持使其成为开发者创建可定制且适合生产环境的基于代理的工作流的宝贵资源。通过促进GUI环境中的无缝交互，Eko体现了LLM驱动虚拟助手的技术进步。

These production-level implementations highlight the practical applications and benefits of LLM-brained GUI agents in enhancing automation, productivity, and user engagement across different platforms and industries.
这些生产级实现突显了LLM驱动GUI代理在提升自动化、生产力和用户参与度方面的实际应用和优势，涵盖了不同平台和行业。

20.5 10.3 Takeaways

20.6 10.3 结论

The application of LLM-brained GUI agents has ushered in new capabilities and interfaces for tasks such as GUI testing and virtual assistance, introducing natural language interactions, enhanced automation, and improved accessibility across platforms. These agents are transforming the way users interact with software applications by simplifying complex tasks and making technology more accessible. However, despite these advancements, LLM-brained GUI agents are still in their infancy, and several challenges need to be addressed for them to reach maturity. Key insights from recent developments include:
LLM驱动的GUI代理的应用为GUI测试和虚拟助手等任务带来了新的能力和界面，引入了自然语言交互、增强的自动化以及跨平台的可访问性。这些代理正在改变用户与软件应用的交互方式，通过简化复杂任务使技术更加易用。然而，尽管取得了这些进展，LLM驱动的GUI代理仍处于初期阶段，仍需解决若干挑战以实现成熟。近期发展的关键见解包括：

1. Natural Language-Driven Interactions: LLM-powered GUI agents have enabled users to interact with applications using natural language, significantly lowering the barrier to entry for non-expert users. In GUI testing, tools like GPTDroid [125] and AUITestAgent [465] allow testers to specify test cases and requirements in plain language, automating the execution and verification processes. Similarly, virtual assistants like LLMPA [486] and ProAgent [485] interpret user commands to perform complex tasks, showcasing the potential of natural language interfaces in simplifying user interactions across platforms.
2. 自然语言驱动的交互：基于大型语言模型（LLM）的图形用户界面（GUI）代理使用户能够使用自然语言与应用程序交互，显著降低了非专业用户的使用门槛。在GUI测试中，像GPTDroid [125]和AUITestAgent [465]这样的工具允许测试人员用通俗语言指定测试用例和需求，实现执行和验证过程的自动化。同样，虚拟助手如LLMPA [486]和ProAgent [485]能够解析用户命令以执行复杂任务，展示了自然语言界面在简化跨平台用户交互方面的潜力。

2. Enhanced Automation of Complex Tasks: These agents have demonstrated the ability to automate multistep and intricate workflows without the need for manual scripting. Projects like AutoTask [488] and GPTVoic-eTasker [487] autonomously explore and interact with GUI environments, executing tasks based on high-level goals or voice commands. In GUI testing, agents have improved coverage and efficiency by automating the generation of test inputs and reproducing bugs from textual descriptions, as seen in CrashTranslator [470] and AdbGPT [471].
3. 复杂任务的增强自动化：这些代理展示了无需手动编写脚本即可自动化多步骤复杂工作流的能力。项目如AutoTask [488]和GPTVoiceTasker [487]能够自主探索和交互GUI环境，基于高层目标或语音命令执行任务。在GUI测试中，代理通过自动生成测试输入和根据文本描述复现缺陷，提高了覆盖率和效率，典型案例包括CrashTranslator [470]和AdbGPT [471]。

3. Multimodal Perception and Interaction: Integrating visual and textual inputs has enhanced the agents' understanding of GUI contexts, leading to better decision-making and interaction accuracy. Agents like VizAbil-ity [484] and OpenAdapt [492] utilize screenshots, UI trees, and OCR to perceive the environment more comprehensively. This multimodal approach is crucial for applications that require precise identification and manipulation of GUI elements, especially in dynamic or visually complex interfaces.
4. 多模态感知与交互：整合视觉和文本输入增强了代理对GUI上下文的理解，提升了决策和交互的准确性。代理如VizAbility [484]和OpenAdapt [492]利用截图、UI树和光学字符识别（OCR）更全面地感知环境。这种多模态方法对于需要精确识别和操作GUI元素的应用尤为关键，尤其是在动态或视觉复杂的界面中。

4. Improved Accessibility and User Experience: LLM-brained GUI agents have contributed to making technology more accessible to users with disabilities or limited technical proficiency. Tools like VizAbility [484] aid blind and low-vision users in understanding data visualizations, while EasyAsk [491] assists older adults in navigating smartphone functions. By tailoring interactions to the needs of diverse user groups, these agents enhance inclusivity and user experience.
5. 改善无障碍性和用户体验：基于LLM的GUI代理有助于提升技术对残障用户或技术水平有限用户的可及性。工具如VizAbility [484]帮助盲人和低视力用户理解数据可视化，而EasyAsk [491]辅助老年人使用智能手机功能。通过针对不同用户群体的需求定制交互，这些代理增强了包容性和用户体验。

LLM-brained GUI agents are transforming the landscape of GUI interaction and automation by introducing natural language understanding, enhanced automation capabilities, and improved accessibility. While they are still in the early stages of development, the ongoing advancements and emerging applications hold great promise for the future. Continued research and innovation are essential to overcome current challenges and fully realize the potential of these intelligent agents across diverse domains and platforms.
基于LLM的GUI代理通过引入自然语言理解、增强的自动化能力和改进的无障碍性，正在改变GUI交互和自动化的格局。尽管它们仍处于早期发展阶段，但持续的进展和新兴应用展现出巨大的未来潜力。持续的研究和创新对于克服当前挑战、充分实现这些智能代理在多领域和多平台的潜能至关重要。

21 11 Limitations, Challenges and Future ROADMAP

22 11 限制、挑战与未来路线图

Despite significant advancements in the development of LLM-brained GUI agents, it is important to acknowledge that this field is still in its infancy. Several technical challenges and limitations hinder their widespread adoption in real-world applications. Addressing these issues is crucial to enhance the agents' effectiveness, safety, and user acceptance. In this section, we outline key limitations and propose future research directions to overcome these challenges, providing concrete examples to illustrate each point.
尽管基于LLM的GUI代理取得了显著进展，但必须承认该领域仍处于起步阶段。若干技术挑战和限制阻碍了其在实际应用中的广泛采用。解决这些问题对于提升代理的有效性、安全性和用户接受度至关重要。本节将概述主要限制并提出未来研究方向，以克服这些挑战，并通过具体示例加以说明。

22.1 11.1 Privacy Concerns

22.2 11.1 隐私问题

Privacy is a critical concern uniquely intensified in the context of LLM-powered GUI agents. These agents often require access to sensitive user data—such as screenshots, interaction histories, personal credentials, and confidential documents—to effectively perceive and interact with the GUI environment. In many cases, this data must be transmitted to remote servers for model inference, especially when relying on cloud-based LLMs [546]-[548]. Such deployments raise significant privacy risks, including data breaches, unauthorized access, and misuse of personal information. These concerns are further amplified when sensitive inputs are routed through third-party APIs or processed off-device, creating compliance and security vulnerabilities that can deter real-world adoption.
隐私是LLM驱动的GUI代理中特别突出的关键问题。这些代理通常需要访问敏感用户数据——如截图、交互历史、个人凭证和机密文件——以有效感知和交互GUI环境。在许多情况下，这些数据必须传输到远程服务器进行模型推理，尤其是在依赖云端LLM时[546]-[548]。此类部署带来了重大隐私风险，包括数据泄露、未经授权访问和个人信息滥用。当敏感输入通过第三方API或在设备外处理时，这些风险进一步加剧，造成合规和安全漏洞，阻碍实际应用的推广。

For instance, a GUI agent tasked with managing a user's email inbox may need to read, classify, and respond to messages containing highly personal or confidential content. Offloading this processing to the cloud introduces risks of exposure, prompting hesitation among users and organizations due to potential privacy violations [432], [549], [550]. Compared to traditional LLM applications, GUI agents operate at a finer granularity of user activity and often require broader system access, making privacy-preserving deployment strategies a critical and domain-specific challenge.
例如，负责管理用户邮箱的GUI代理可能需要读取、分类并回复包含高度个人或机密内容的邮件。将此类处理任务卸载到云端增加了暴露风险，导致用户和组织因潜在隐私违规而犹豫不决[432]，[549]，[550]。与传统LLM应用相比，GUI代理操作的用户活动粒度更细，且通常需要更广泛的系统访问权限，使得隐私保护部署策略成为一项关键且领域特定的挑战。

Potential Solutions: To mitigate privacy concerns, future research should focus on enabling on-device inference, where the language model operates directly on the user's device without uploading personal data [551], [552]. Achieving this requires advancements in model compression techniques 553, on-device optimization [554], and efficient inference algorithms 555 to accommodate the computational limitations of user devices. In addition, frameworks must incorporate data redaction, secure communication channels, and explicit scoping of data usage within the agent's context. Furthermore, integration with system-level privacy controls and user consent mechanisms (e.g., runtime permission dialogs or sandboxed execution) is essential for deployment in regulated domains.
潜在解决方案：为缓解隐私问题，未来研究应聚焦于实现设备端推理，即语言模型直接在用户设备上运行，无需上传个人数据[551]，[552]。这需要在模型压缩技术[553]、设备端优化[554]和高效推理算法[555]方面取得进展，以适应用户设备的计算限制。此外，框架必须包含数据脱敏、安全通信通道和明确限定代理上下文中的数据使用范围。此外，集成系统级隐私控制和用户同意机制（如运行时权限对话框或沙箱执行）对于在受监管领域的部署至关重要。

From the technical perspective, implementing privacy-preserving techniques like federated learning [556], differential privacy [557], and homomorphic encryption [558] can enhance data security while allowing the model to learn from user data. Furthermore, developers of GUI agents should collaborate with privacy policymakers to ensure that user data and privacy are appropriately protected [559]. They should make the data handling processes transparent to users, clearly informing them about what data are being transmitted and how they are used, and obtain explicit user consent 560
从技术角度来看，实施诸如联邦学习（federated learning）[556]、差分隐私（differential privacy）[557]和同态加密（homomorphic encryption）[558]等隐私保护技术，可以在保障模型从用户数据中学习的同时增强数据安全性。此外，GUI代理的开发者应与隐私政策制定者合作，确保用户数据和隐私得到适当保护[559]。他们应向用户透明展示数据处理流程，明确告知哪些数据被传输及其用途，并获得用户的明确同意560

22.3 11.2 Latency, Performance, and Resource Constraints

22.4 11.2 延迟、性能与资源限制

One challenge that is particularly salient for GUI agents-distinct from general LLM applications is the issue of latency in interactive, multi-step execution environments. Since GUI agents rely on large language models to plan and issue actions, their computational demands can lead to high latency and slow response times, which directly impact user experience [561]. This is especially critical in time-sensitive or interactive scenarios, where delays in action execution can cause user frustration or even trigger unintended system behavior. Unlike single-shot LLM tasks, GUI agents typically operate over extended sequences of steps, making latency cumulative and more disruptive over time. The problem is further amplified in on-device deployments, where computational resources are limited. For example, running an LLM-powered agent within a mobile app may result in sluggish performance or rapid battery depletion, significantly undermining usability on resource-constrained platforms 562-564]. These concerns are uniquely pronounced in GUI agents due to their need for real-time perception, decision-making, and UI control in dynamic environments [563].
GUI代理面临的一个特别突出的问题——区别于一般大型语言模型（LLM）应用——是交互式多步骤执行环境中的延迟问题。由于GUI代理依赖大型语言模型来规划和发出操作指令，其计算需求可能导致高延迟和响应缓慢，直接影响用户体验[561]。这在时间敏感或交互性强的场景中尤为关键，操作执行的延迟可能引发用户不满甚至导致系统行为异常。与单次调用的LLM任务不同，GUI代理通常在较长的步骤序列中运行，使得延迟累积并随时间加剧干扰。在设备端部署时，计算资源有限，这一问题更为突出。例如，在移动应用中运行基于LLM的代理可能导致性能迟缓或电池快速耗尽，严重影响资源受限平台的可用性[562-564]。由于GUI代理需要在动态环境中实现实时感知、决策和界面控制，这些问题尤为显著[563]。

Potential Solutions: Future work should aim to reduce inference latency by optimizing model architectures for speed and efficiency [565]. Techniques such as model distillation can create smaller, faster models without substantially compromising performance [566]. Leveraging hardware accelerators like GPUs, TPUs, or specialized AI chips, and exploring parallel processing methods can enhance computational efficiency [567]. Implementing incremental inference and caching mechanisms may also improve responsiveness by reusing computations where applicable [568]. Additionally, research into model optimization and compression techniques, such as pruning [569] and quantization [553] can produce lightweight models suitable for deployment on resource-constrained devices. Exploring edge computing [552] and distributed inference [570] can help distribute the computational load effectively.
潜在解决方案：未来工作应致力于通过优化模型架构以提升速度和效率来降低推理延迟[565]。模型蒸馏等技术可在不显著损失性能的前提下，生成更小更快的模型[566]。利用GPU、TPU或专用AI芯片等硬件加速器，并探索并行处理方法，可提升计算效率[567]。实现增量推理和缓存机制，通过复用计算结果也有助于提高响应速度[568]。此外，研究模型优化和压缩技术，如剪枝[569]和量化[553]，可打造适合资源受限设备部署的轻量级模型。探索边缘计算[552]和分布式推理[570]有助于有效分担计算负载。

Moreover, GUI agents should collaborate with application developers to encourage them to expose high-level native APIs for different functionalities [198], [199], which combine several UI operations into single API calls. By integrating these APIs into the GUI agent, tasks can be completed with fewer steps, making the process much faster and reducing cumulative latency.
此外，GUI代理应与应用开发者合作，鼓励其为不同功能暴露高级原生API[198]，[199]，将多个UI操作合并为单次API调用。通过将这些API集成到GUI代理中，任务可用更少步骤完成，显著加快流程并减少累计延迟。

22.5 11.3 Safety and Reliability

22.6 11.3 安全性与可靠性

The real-world actuation capabilities of GUI agents introduce unique and significant safety and reliability risks beyond those faced by general-purpose LLMs. Because GUI agents can directly manipulate user interfaces-clicking buttons, deleting files, submitting forms, or initiating system-level operations-errors in action generation can have irreversible consequences [548], [571]. These may include data corruption, accidental message dispatches, application crashes, or unauthorized access to sensitive system components [572], [573]. Such risks are compounded by the inherent uncertainty and non-determinism in LLM outputs: agents may hallucinate actions, misinterpret UI contexts, or behave inconsistently across sessions [574-578]. For example, an agent automating financial transactions could mistakenly execute the wrong transfer, leading to material losses. Furthermore, GUI agents expose a broader attack surface than traditional LLM applications—they are susceptible to black-box adversarial attacks that could manipulate their inputs or exploit their decision policies [579].
GUI代理在现实世界中的执行能力带来了超出通用LLM的独特且重大安全与可靠性风险。由于GUI代理能直接操作用户界面——点击按钮、删除文件、提交表单或启动系统级操作——操作生成错误可能导致不可逆后果[548]，[571]。这些后果可能包括数据损坏、误发消息、应用崩溃或未经授权访问敏感系统组件[572]，[573]。LLM输出固有的不确定性和非确定性加剧了这些风险：代理可能产生幻觉操作、误解UI上下文或在不同会话中表现不一致[574-578]。例如，自动化金融交易的代理可能错误执行转账，造成实质性损失。此外，GUI代理暴露的攻击面比传统LLM应用更广——它们易受黑盒对抗攻击，攻击者可能操控输入或利用其决策策略[579]。

Unlike passive language models, GUI agents operate within dynamic software ecosystems where incorrect actions can propagate across applications or escalate into systemwide disruptions. Integration challenges also arise, including compatibility with evolving UI frameworks, user permission boundaries, and software-specific safety constraints, and malicious attacks [580], [581]. These concerns, coupled with the lack of interpretability and formal guarantees, contribute to skepticism and reluctance from users and developers alike. Addressing safety and reliability in GUI agents thus requires not only robust model behavior but also runtime safeguards 582], rollback mechanisms, and

interface-aware verification techniques tailored specifically to this interaction paradigm.
与被动语言模型不同，GUI代理运行于动态软件生态系统中，错误操作可能跨应用传播或引发系统级故障。集成挑战还包括与不断演进的UI框架兼容、用户权限边界、软件特定安全约束及恶意攻击[580]，[581]。这些问题，加之缺乏可解释性和形式化保障，导致用户和开发者普遍持怀疑态度和谨慎态度。因而，保障GUI代理的安全性和可靠性不仅需模型行为稳健，还需运行时安全措施[582]、回滚机制及针对该交互范式的界面感知验证技术。

Potential Solutions: Ensuring safety and reliability necessitates robust error detection and handling mechanisms [583]. Future research should focus on integrating validation steps that verify the correctness of inferred actions before execution [584]. Developing formal verification methods [585], implementing exception handling routines [586], and establishing rollback procedures [587] are essential for preventing and mitigating the impact of errors. Additionally, incorporating permission management [588-591] to limit the agent's access rights can prevent unauthorized or harmful operations.
潜在解决方案：确保安全性和可靠性需健全的错误检测与处理机制[583]。未来研究应聚焦于集成验证步骤，确认推断操作的正确性后再执行[584]。开发形式化验证方法[585]、实现异常处理流程[586]及建立回滚程序[587]，对防止和减轻错误影响至关重要。此外，纳入权限管理[588-591]以限制代理访问权限，可防止未经授权或有害操作。

Furthermore, creating standardized interaction protocols can facilitate smoother and safer integration with various applications and systems [592]. Ensuring that agents comply with security best practices, such as secure authentication and authorization protocols [593], is essential.
此外，制定标准化交互协议有助于实现与各类应用和系统的更顺畅、更安全集成[592]。确保代理遵守安全最佳实践，如安全认证和授权协议[593]，同样至关重要。

22.7 11.4 Human-Agent Interaction

22.8 11.4 人机交互

Human-agent interaction introduces distinct challenges in the context of GUI agents, where the agent and user operate within the same dynamic interface. Any user intervention-such as moving the mouse, altering window states, or modifying inputs—can inadvertently interfere with the agent's ongoing execution, potentially causing conflicts, unintended actions, or breakdowns in task flow [594], [595]. Designing robust collaboration protocols that govern when the agent should yield control, pause execution, or defer to the user is a non-trivial problem specific to GUI-based automation.
在人机交互中，GUI代理面临独特挑战，因为代理和用户在同一动态界面内操作。任何用户干预——如移动鼠标、改变窗口状态或修改输入——都可能无意中干扰代理的执行，导致冲突、意外操作或任务流程中断[594]，[595]。设计健壮的协作协议以规范代理何时应让出控制权、暂停执行或让步给用户，是GUI自动化中特有的复杂问题。

Further complicating this interaction is the ambiguity of user instructions. Natural language commands may be vague, under-specified, or context-dependent, leading to misinterpretations or incomplete task plans. GUI agents may also encounter runtime uncertainties—such as unexpected pop-ups, missing inputs, or conflicting UI states—that require them to seek user clarification or feedback [19], [596]. Determining when and how an agent should request user input-whether for disambiguation, permission, or verification-is critical for ensuring both reliability and user trust [67], [597], [598].
用户指令的模糊性进一步加剧了这种交互。自然语言命令可能含糊、不完整或依赖上下文，导致误解或任务计划不全。GUI代理还可能遇到运行时不确定性——如意外弹窗、缺失输入或UI状态冲突——需要向用户寻求澄清或反馈[19]，[596]。确定代理何时以及如何请求用户输入——无论是为了解歧义、获得许可还是验证——对确保系统可靠性和用户信任至关重要[67]，[597]，[598]。

This challenge is exemplified in the fabricated scenario shown in Figure 31 where a GUI agent is instructed to send an email to "Tom." The agent must first prompt the user to log in securely, protecting credentials by avoiding automated input. It then encounters ambiguity when multiple contacts named "Tom" are found, and resolves it by prompting the user to select the intended recipient. Finally, before dispatching the email, the agent requests explicit confirmation, recognizing that email-sending is a non-reversible action with privacy implications [19]. Although the task appears simple, it reflects the complexity of real-world human-GUI agent collaboration, involving privacy preservation, ambiguity resolution, and intentionality confirmation [599]. These are not generic LLM issues, but domain-specific challenges rooted in shared interaction with software interfaces—underscoring the need for new design paradigms around shared control, interruption handling, and proactive clarification in GUI agent systems.
这一挑战在图31所示的虚构场景中得以体现：GUI代理被指示向“Tom”发送邮件。代理首先必须提示用户安全登录，避免自动输入以保护凭证。随后，当发现多个名为“Tom”的联系人时，代理通过提示用户选择目标收件人来解决歧义。最后，在发送邮件前，代理请求明确确认，认识到邮件发送是不可逆且涉及隐私的操作[19]。尽管任务看似简单，但它反映了现实中人机GUI代理协作的复杂性，涵盖隐私保护、歧义解决和意图确认[599]。这些问题非通用大型语言模型（LLM）问题，而是基于软件界面共享交互的领域特定挑战——强调了围绕共享控制、中断处理和主动澄清的新设计范式在GUI代理系统中的必要性。

Potential Solutions: Emphasizing user-centered design 600 principles can address user needs and concerns, providing options for customization and control over the agent's behavior [596]. Equipping agents with the ability to engage in clarification dialogues when user instructions are unclear can enhance task accuracy [601]. Natural language understanding components can detect ambiguity and prompt users for additional information. For instance, the agent could ask, "There are two contacts named John. Do you mean John Smith or John Doe?" Incorporating human-in-the-loop systems allows for human intervention during task execution, enabling users to guide or correct the

agent's decisions when necessary [602]. Developing adaptive interaction models that facilitate seamless collaboration between humans and agents is essential. Additionally, providing transparency and explainability in the agent's reasoning processes can build user trust and improve cooperation [67, 603], 604].
潜在解决方案：强调以用户为中心的设计原则[600]，可满足用户需求和关注点，提供定制和控制代理行为的选项[596]。赋予代理在用户指令不明确时进行澄清对话的能力，可提升任务准确性[601]。自然语言理解组件能够检测歧义并提示用户补充信息。例如，代理可能询问：“有两个名为John的联系人，您指的是John Smith还是John Doe？”引入人机协同系统允许在任务执行过程中进行人工干预，使用户在必要时引导或纠正代理决策[602]。开发适应性交互模型以促进人机无缝协作至关重要。此外，提供代理推理过程的透明性和可解释性，有助于建立用户信任并改善合作[67]，[603]，[604]。

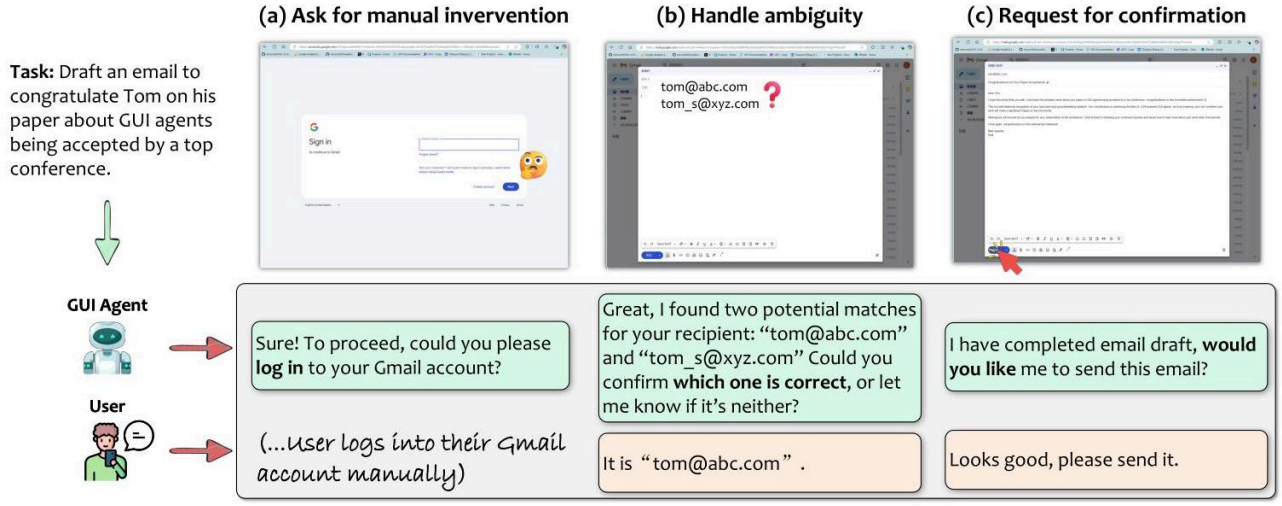

Fig. 31: An illustrative example of human-agent interaction for completing an email sending request.
图31：完成发送邮件请求的人机代理交互示例。

Lastly, developing a virtual desktop environment for the agent to operate in-one that connects to the user's main desktop session without disrupting their workflow, can significantly enhance the user experience (UX) in human-agent interaction. The picture-in-picture mode implemented in UFO2 334 demonstrates this concept in practice, as illustrated in Figure 32 By allowing the agent to run within a resizable and movable virtualized desktop, users can easily minimize or reposition the agent window as needed. This flexibility improves both the usability and the overall UX of interacting with GUI-based agents.
最后，开发一个供代理操作的虚拟桌面环境——该环境连接用户的主桌面会话且不干扰其工作流程——可显著提升人机交互的用户体验（UX）。UFO2[334]中实现的画中画模式展示了这一概念的实际应用，如图32所示。通过允许代理在可调整大小和可移动的虚拟桌面内运行，用户可以根据需要轻松最小化或重新定位代理窗口。这种灵活性提升了GUI代理的可用性和整体用户体验。

22.9 11.5 Customization and Personalization

22.10 11.5 定制化与个性化

Effective GUI agents must go beyond generic task completion and provide experiences that are personalized to individual users, adapting to their unique workflows, preferences, and behavioral patterns [45], [605]. Unlike general LLM applications that operate in isolated prompts or conversations, GUI agents work across software environments where user interaction styles can vary significantly. A one-size-fits-all agent may fail to align with how a particular user edits documents, navigates interfaces, or organizes tasks—resulting in friction, inefficiency, or user frustration [606].
高效的GUI代理必须超越通用任务完成，提供针对个别用户个性化的体验，适应其独特的工作流程、偏好和行为模式[45]，[605]。不同于在孤立提示或对话中运行的一般大型语言模型（LLM）应用，GUI代理跨越软件环境，用户的交互风格差异显著。千篇一律的代理可能无法契合特定用户的文档编辑、界面导航或任务组织方式，导致摩擦、低效或用户挫败感[606]。

For instance, a GUI agent assisting with document editing must learn the user's preferred tone, formatting conventions, and vocabulary. Without this contextual understanding, the agent may offer irrelevant suggestions or enforce formatting inconsistent with the user's intent. Personalization in GUI agents thus requires longitudinal learning, where the agent continually adapts based on prior interactions, fine-tunes its behavior to match user expectations, and preserves consistency across sessions [607].
例如，协助文档编辑的GUI代理必须学习用户偏好的语气、格式规范和词汇。缺乏这种上下文理解，代理可能提供无关建议或强制执行与用户意图不符的格式。GUI代理的个性化因此需要长期学习，代理基于先前交互持续适应，微调行为以匹配用户期望，并在会话间保持一致性[607]。

However, this introduces new challenges. The high variability in user preferences-especially in free-form GUI environments—makes it difficult to define universal personalization strategies. Moreover, collecting and leveraging user-specific data must be done responsibly, raising critical concerns around privacy, data retention, and on-device learning. Striking a balance between effective customization and user trust is particularly important for GUI agents, which often operate over sensitive documents, personal applications, or system-level interfaces.
然而，这也带来了新挑战。用户偏好的高度多样性——尤其是在自由形式的GUI环境中——使得定义通用个性化策略变得困难。此外，收集和利用用户特定数据必须负责任地进行，涉及隐私保护、数据保留和设备端学习等关键问题。在处理敏感文档、个人应用或系统级界面时，平衡有效定制与用户信任对GUI代理尤为重要。

Potential Solutions: Future research should focus on developing mechanisms for user modeling [608] and preference learning [609], enabling agents to tailor their actions to individual users. Techniques such as reinforcement learning from user feedback [610], collaborative filtering [611], and context-aware computing [612] can help agents learn user preferences over time. Ensuring that personalization is achieved without compromising privacy is essential [613], potentially through on-device learning and anonymized data processing. In a more futuristic, cyberpunk-inspired scenario, agents may inversely generate GUIs tailored to users' needs, enabling greater customization and personalization [614].
潜在解决方案：未来的研究应聚焦于开发用户建模（user modeling）[608]和偏好学习（preference learning）[609]机制，使代理能够根据个体用户定制其行为。诸如基于用户反馈的强化学习（reinforcement learning）[610]、协同过滤（collaborative filtering）[611]和上下文感知计算（context-aware computing）[612]等技术，可以帮助代理随着时间推移学习用户偏好。确保个性化实现的同时不损害隐私至关重要[613]，这可能通过设备端学习和匿名数据处理来实现。在更具未来感、赛博朋克风格的场景中，代理甚至可能反向生成针对用户需求定制的图形用户界面（GUI），实现更高程度的定制化和个性化[614]。

22.11 11.6 Ethical and Regulatory Challenges

22.12 11.6 伦理与监管挑战

LLM-powered GUI agents raise distinct ethical and regulatory concerns due to their ability to perform real-world actions across software interfaces. Unlike traditional LLMs, these agents can autonomously trigger operations, manipulate data, and interact with sensitive applications—amplifying risks around accountability, fairness, and user consent | 548 | 615 |- 618.
基于大型语言模型（LLM）的GUI代理因其能够跨软件界面执行真实操作而引发独特的伦理和监管问题。与传统LLM不同，这些代理可以自主触发操作、操控数据并与敏感应用交互——这加剧了责任归属、公平性和用户同意等方面的风险 | 548 | 615 |- 618。

A key concern is bias inherited from training data, which can lead to unfair behavior in sensitive workflows. For example, a GUI agent assisting in hiring may unknowingly exhibit gender or racial bias 619 , 620 . These risks are harder to audit at the GUI level due to limited traceability across multi-application actions. Regulatory compliance adds further complexity. GUI agents often operate across domains with strict data protection laws, but lack standardized mechanisms for logging actions or securing user consent. This makes it challenging to meet legal and ethical standards, especially when agents act in opaque or background contexts. Addressing these issues requires tailored solutions for GUI agents, including permission controls, runtime confirmations, and transparent activity logs-ensuring safe, fair, and compliant deployment across diverse environments.
一个关键问题是训练数据中继承的偏见，可能导致在敏感工作流程中表现出不公平行为。例如，协助招聘的GUI代理可能无意中表现出性别或种族偏见 619 , 620 。由于多应用操作的可追溯性有限，这些风险在GUI层面更难审计。监管合规性进一步增加了复杂性。GUI代理通常跨越受严格数据保护法律约束的领域，但缺乏标准化的操作日志记录或用户同意保障机制。这使得在代理以不透明或后台方式行动时，满足法律和伦理标准变得尤为困难。解决这些问题需要针对GUI代理的定制化方案，包括权限控制、运行时确认和透明的活动日志，确保在多样化环境中的安全、公平和合规部署。

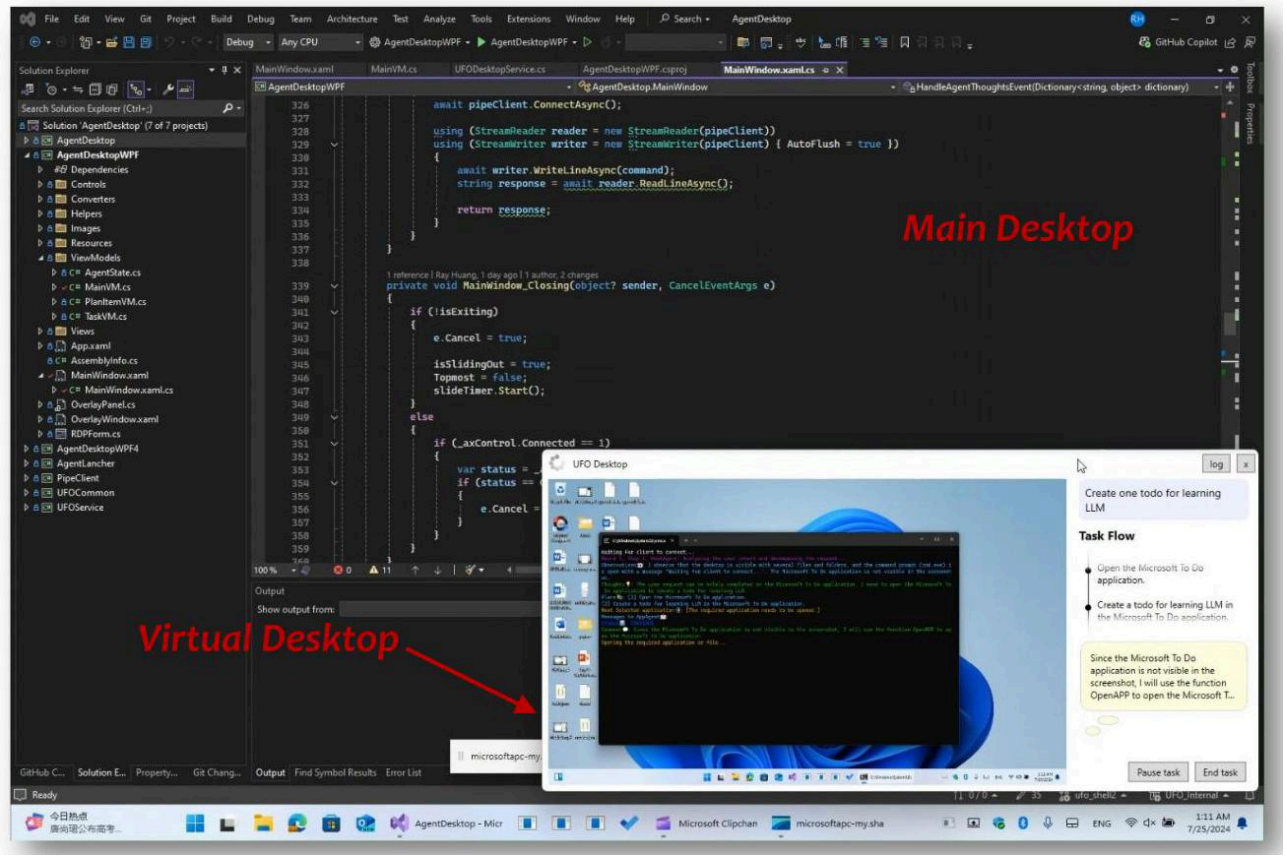

Fig. 32: The Picture-in-Picture interface in UFO^2 : a virtual desktop window enabling non-disruptive automation. Figure adapted from 334.
图32：UFO^{2}中的画中画界面：一个虚拟桌面窗口，实现非干扰式自动化。图示改编自334。

Potential Solutions: Addressing these concerns requires establishing clear ethical guidelines and regulatory frameworks for the development and use of GUI agents 621. Future work should focus on creating mechanisms for auditing and monitoring agent behavior [622] to ensure compliance with ethical standards and legal requirements 623]. Incorporating bias detection and mitigation strategies in language models can help prevent discriminatory or unfair actions 624]. Providing users with control over data usage and clear information about the agent's capabilities can enhance transparency and trust.
潜在解决方案：应对这些问题需要建立明确的伦理准则和GUI代理开发及使用的监管框架 621。未来工作应聚焦于创建审计和监控代理行为的机制[622]，以确保符合伦理标准和法律要求 623]。在语言模型中引入偏见检测和缓解策略，有助于防止歧视性或不公平行为 624]。为用户提供对数据使用的控制权及关于代理能力的明确信息，可以增强透明度和信任。

22.13 11.7 Scalability and Generalization

22.14 11.7 可扩展性与泛化能力

GUI agents often struggle to scale beyond specific applications or environments, limiting their generalization. Each software interface features unique layouts, styles, and interaction patterns—even common UI elements like pop-up windows can vary widely 625]. These variations make it difficult to design agents that operate robustly across platforms without retraining or fine-tuning.
GUI代理常常难以超越特定应用或环境进行扩展，限制了其泛化能力。每个软件界面都具有独特的布局、风格和交互模式——即使是常见的UI元素如弹出窗口也差异巨大 625]。这些差异使得设计能够跨平台稳健运行的代理变得困难，通常需要重新训练或微调。

A further challenge is the dynamic nature of real-world GUIs. Frequent changes due to software updates, A/B testing, or interface redesigns—such as repositioned buttons or modified widget hierarchies—can easily break previously functional agents. For example, an agent trained on one version of a word processor may fail when the layout changes, or when deployed on a different program with similar functionality but a different interface structure. Even when GUIs share visual similarities, agents often fail to generalize without additional exploration or adaptation [626]. This lack of robustness restricts deployment in practical settings and increases the cost of maintenance, requiring frequent updates or retraining to stay aligned with evolving environments 627-629]. Overcoming this challenge remains critical for developing truly scalable and adaptable GUI agents.
另一个挑战是现实世界GUI的动态特性。由于软件更新、A/B测试或界面重设计导致的频繁变化——如按钮位置调整或控件层级修改——很容易使之前正常工作的代理失效。例如，在某版本文字处理器上训练的代理，界面布局变化后可能无法正常工作，或在功能相似但界面结构不同的程序上部署时失败。即使GUI在视觉上相似，代理往往也难以在没有额外探索或适应的情况下泛化[626]。这种鲁棒性不足限制了实际

应用部署，并增加了维护成本，需要频繁更新或重新训练以适应不断变化的环境 627-629]。克服这一挑战对于开发真正可扩展且适应性强的GUI代理至关重要。

Potential Solutions: To enhance scalability and generalization, one solution from the dataset perspective is to create comprehensive GUI agent datasets that cover a wide range of environments, user requests, GUI designs, platforms, and interaction patterns. By exposing the LLM to diverse data sources during training, the model can learn common patterns and develop a more generalized understanding, enabling it to adapt to infer the functionality of new interfaces based on learned similarities [630].
潜在解决方案：为提升可扩展性和泛化能力，从数据集角度的一个方案是创建涵盖广泛环境、用户请求、GUI设计、平台和交互模式的综合GUI代理数据集。通过在训练中让大型语言模型接触多样化数据源，模型可以学习共性模式，形成更通用的理解，从而基于已学相似性推断新界面的功能[630]。

To further enhance adaptability, research can focus on techniques such as transfer learning [631] and meta-learning 632]. Transfer learning involves pre-training a model on a large, diverse dataset and then fine-tuning it on a smaller, task-specific dataset. In the context of GUI agents, this means training the LLM on a wide array of GUI interactions before customizing it for a particular application or domain. Meta-learning, enables the model to rapidly adapt to new tasks with minimal data by identifying underlying structures and patterns across different tasks. These approaches enable agents to generalize from limited data and adapt to new environments with minimal retraining.
为进一步增强适应性，研究可聚焦于迁移学习（transfer learning）[631]和元学习（meta-learning）[632]等技术。迁移学习指先在大规模多样化数据集上预训练模型，再在较小的特定任务数据集上微调。在GUI代理背景下，即先训练LLM掌握广泛的GUI交互，再针对特定应用或领域进行定制。元学习使模型能够通过识别不同任务间的底层结构和模式，快速适应新任务且所需数据极少。这些方法使代理能够从有限数据中泛化，并以最小的再训练适应新环境。

However, even with these measures, the agent may still encounter difficulties in unfamiliar environments. To address this, we advocate for developers to provide helpful knowledge bases, such as guidance documents, application documentation, searchable FAQs, and even human demonstrations on how to use the application [633- 635]. Techniques like RAG [189] can be employed, where the agent retrieves relevant information from a knowledge base at runtime to inform its decisions [636]. For instance, if the agent encounters an unknown interface element, it can query the documentation to understand its purpose and how to interact with it. This approach enhances the agent's capabilities without requiring extensive retraining. Implementing these solutions requires collaborative efforts not only from agent developers but also from application or environment providers.
然而，即使采取了这些措施，代理在不熟悉的环境中仍可能遇到困难。为此，我们建议开发者提供有用的知识库，如指导文档、应用文档、可搜索的常见问题解答，甚至是关于如何使用应用的人类示范[633-635]。可以采用如RAG（检索增强生成）[189]等技术，代理在运行时从知识库中检索相关信息以辅助决策[636]。例如，如果代理遇到未知的界面元素，它可以查询文档以了解其用途及交互方式。这种方法提升了代理的能力，而无需大量重新训练。实现这些解决方案不仅需要代理开发者的协作，还需应用或环境提供者的共同努力。

22.15 11.8 Summary

22.16 11.8 总结

LLM-brained GUI agents hold significant promise for automating complex tasks and enhancing user productivity across various applications. However, realizing this potential requires addressing the outlined limitations through dedicated research and development efforts. By addressing these challenges, the community can develop more robust and widely adopted GUI agents.
基于大型语言模型（LLM）的图形用户界面（GUI）代理在自动化复杂任务和提升用户生产力方面展现出巨大潜力，适用于多种应用场景。然而，要实现这一潜力，需通过专门的研发工作解决上述限制。通过克服这些挑战，社区能够开发出更稳健且被广泛采用的GUI代理。

Collaboration among researchers, industry practitioners, policymakers, and users is essential to navigate these challenges successfully. Establishing interdisciplinary teams can foster innovation and ensure that GUI agents are developed responsibly, with a clear understanding of technical, ethical, and societal implications. As the field progresses, continuous evaluation and adaptation will be crucial to align technological advancements with user needs and expectations, ultimately leading to more intelligent, safe, and user-friendly GUI agents.
研究人员、行业从业者、政策制定者和用户之间的协作对于成功应对这些挑战至关重要。组建跨学科团队能够促进创新，确保GUI代理的负责任开发，并清晰理解其技术、伦理及社会影响。随着领域的发展，持续的评估与调整对于使技术进步与用户需求和期望保持一致至关重要，最终实现更智能、安全且用户友好的GUI代理。

23 12 CONCLUSION

24 12 结论

The combination of LLMs and GUI automation marks a transformative moment in human-computer interaction. LLMs provide the "brain" for natural language processing, comprehension, and GUI understanding, while GUI automation tools serve as the "hands", translating the agent's cognitive abilities into actionable commands within software environments. Together, they form LLM-powered GUI agents that introduce a new paradigm in user interaction, allowing users to control applications through straightforward natural language commands instead of complex, platform-specific UI operations. This synergy has shown remarkable potential, with applications flourishing in both research and industry.

大型语言模型（LLM）与GUI自动化的结合标志着人机交互的变革时刻。LLM提供了自然语言处理、理解和GUI认知的“大脑”，而GUI自动化工具则作为“手”，将代理的认知能力转化为软件环境中的可执行命令。两者结合形成了基于LLM的GUI代理，开创了用户交互的新范式，使用户能够通过简单的自然语言命令控制应用程序，而无需复杂且平台特定的界面操作。这种协同展现出显著潜力，应用在科研和工业领域均蓬勃发展。

In this survey, we provide a comprehensive, systematic, and timely overview of the field of LLM-powered GUI agents. Our work introduces the core components and advanced techniques that underpin these agents, while also examining critical elements such as data collection, model development, frameworks, evaluation methodologies, and real-world applications. Additionally, we explore the current limitations and challenges faced by these agents and outline a roadmap for future research directions. We hope this survey serves as a valuable handbook for those learning about LLM-powered GUI agents and as a reference point for researchers aiming to stay at the forefront of developments in this field.
本综述系统、全面且及时地回顾了基于LLM的GUI代理领域。我们介绍了支撑这些代理的核心组件和先进技术，同时考察了数据收集、模型开发、框架、评估方法及实际应用等关键要素。此外，我们探讨了当前代理面临的限制与挑战，并规划了未来研究方向。希望本综述能成为学习基于LLM的GUI代理的宝贵手册，并为研究者提供前沿发展的参考。

As we look to the future, the concept of LLM-brained GUI agents promises to become increasingly tangible, fundamentally enhancing productivity and accessibility in daily life. With ongoing research and development, this technology stands poised to reshape how we interact with digital systems, transforming complex workflows into seamless, natural interactions.
展望未来，基于LLM的GUI代理概念有望日益具体化，根本性地提升日常生活中的生产力和可及性。随着持续的研发，这项技术将重塑我们与数字系统的交互方式，将复杂的工作流程转化为流畅自然的交互体验。

25 REFERENCES

26 参考文献

[1] B. J. Jansen, "The graphical user interface," ACM SIGCHI Bull., vol. 30, pp. 22-26, 1998. [Online]. Available: https://api.semanticscholar.org/CorpusID:18416305
[1] B. J. Jansen, "图形用户界面," ACM SIGCHI通报, 第30卷, 第22-26页, 1998年. [在线]. 可获取: https://api.semanticscholar.org/CorpusID:18416305

[2] H. Sampath, A. Merrick, and A. P. Macvean, "Accessibility of command line interfaces," Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID: 233987139
[2] H. Sampath, A. Merrick, 和 A. P. Macvean, "命令行界面的可访问性," 2021年CHI人因计算系统会议论文集, 2021年. [在线]. 可获取: https://api.semanticscholar.org/CorpusID:233987139

[3] R. Michalski, J. Grobelny, and W. Karwowski, "The effects of graphical interface design characteristics on human-computer interaction task efficiency," ArXiv, vol. abs/1211.6712, 2006. [Online]. Available: https://api.semanticscholar.org/CorpusID: 14695409
[3] R. Michalski, J. Grobelny, 和 W. Karwowski, "图形界面设计特征对人机交互任务效率的影响," ArXiv, 第abs/1211.6712卷, 2006年. [在线]. 可获取: https://api.semanticscholar.org/CorpusID:14695409

[4] T. D. Hellmann and F. Maurer, "Rule-based exploratory testing of graphical user interfaces," in 2011 Agile Conference. IEEE, 2011, pp. 107-116.
[4] T. D. Hellmann 和 F. Maurer, "基于规则的图形用户界面探索性测试," 2011年敏捷会议. IEEE, 2011年, 第107-116页.

[5] J. Steven, P. Chandra, B. Fleck, and A. Podgurski, "jrapture: A capture/replay tool for observation-based testing," SIGSOFT Softw. Eng. Notes, vol. 25, no. 5, p. 158-167, Aug. 2000. [Online]. Available: https://doi.org/10.1145/347636.348993
[5] J. Steven, P. Chandra, B. Fleck, 和 A. Podgurski, "jrapture: 一种基于观察的捕获/重放测试工具," SIGSOFT软件工程笔记, 第25卷第5期, 第158-167页, 2000年8月. [在线]. 可获取: https://doi.org/10.1145/347636.348993

[6] L. Ivančić, D. Suša Vugec, and V. Bosilj Vukšic, "Robotic process automation: systematic literature review," in Business Process Management: Blockchain and Central and Eastern Europe Forum: BPM 2019 Blockchain and CEE Forum, Vienna, Austria, September 1-6, 2019, Proceedings 17. Springer, 2019, pp. 280-295.
[6] L. Ivančić, D. Suša Vugec, 和 V. Bosilj Vukšic, “机器人流程自动化：系统文献综述，”发表于《业务流程管理：区块链与中东欧论坛：BPM 2019 区块链与中东欧论坛》，奥地利维也纳，2019年9月1-6日，论文集17。施普林格，2019年，第280-295页。

[7] W. contributors, "Large language model - wikipedia, the free encyclopedia," 2024, accessed: 2024-11-25. [Online]. Available: https://en.wikipedia.org/wiki/Large_language_model
[7] W. contributors, “大型语言模型 - 维基百科，自由的百科全书，”2024年，访问时间：2024-11-25。[在线]. 可用地址：https://en.wikipedia.org/wiki/Large_language_model

[8] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong et al., "A survey of large language models," arXiv preprint arXiv:2303.18223, 2023.
[8] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong 等，“大型语言模型综述，”arXiv 预印本 arXiv:2303.18223，2023年。

[9] H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, and A. Mian, "A comprehensive overview of large language models," arXiv preprint arXiv:2307.06435, 2023.
[9] H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, 和 A. Mian，“大型语言模型的全面概述，”arXiv 预印本 arXiv:2307.06435，2023年。

[10] S. Yin, C. Fu, S. Zhao, K. Li, X. Sun, T. Xu, and E. Chen, "A survey on multimodal large language models," arXiv preprint arXiv:2306.13549, 2023.
[10] S. Yin, C. Fu, S. Zhao, K. Li, X. Sun, T. Xu, 和 E. Chen，“多模态大型语言模型综述，”arXiv 预印本 arXiv:2306.13549，2023年。

[11] T. Wu, S. He, J. Liu, S. Sun, K. Liu, Q.-L. Han, and Y. Tang, "A brief overview of chatgpt: The history, status quo and potential future development," IEEE/CAA Journal of Automatica Sinica, vol. 10, no. 5, pp. 1122-1136, 2023.
[11] T. Wu, S. He, J. Liu, S. Sun, K. Liu, Q.-L. Han, 和 Y. Tang，“ChatGPT 简要概述：历史、现状及潜在未来发展，”《IEEE/CAA 自动化学报》，第10卷，第5期，2023年，第1122-1136页。

[12] J. Liu, K. Wang, Y. Chen, X. Peng, Z. Chen, L. Zhang, and Y. Lou, "Large language model-based agents for software engineering: A survey," arXiv preprint arXiv:2409.02977, 2024.
[12] J. Liu, K. Wang, Y. Chen, X. Peng, Z. Chen, L. Zhang, 和 Y. Lou，“基于大型语言模型的软件工程代理综述，”arXiv 预印本 arXiv:2409.02977，2024年。

[13] Z. Shen, "LIm with tools: A survey," arXiv preprint arXiv:2409.18807, 2024.
[13] Z. Shen，“带工具的语言模型（LIm）综述，”arXiv 预印本 arXiv:2409.18807，2024年。

[14] T. Feng, C. Jin, J. Liu, K. Zhu, H. Tu, Z. Cheng, G. Lin, and J. You, "How far are we from agi: Are Ilms all we need?" Transactions on Machine Learning Research.
[14] T. Feng, C. Jin, J. Liu, K. Zhu, H. Tu, Z. Cheng, G. Lin, 和 J. You，“我们距离通用人工智能（AGI）还有多远：语言模型（ILMs）是否足够？”《机器学习研究汇刊》。

[15] W. Hong, W. Wang, Q. Lv, J. Xu, W. Yu, J. Ji, Y. Wang, Z. Wang, Y. Zhang, J. Li, B. Xu, Y. Dong, M. Ding, and J. Tang, "Cogagent: A visual language model for gui agents," 2023. [Online]. Available: https://arxiv.org/abs/2312.08914
[15] W. Hong, W. Wang, Q. Lv, J. Xu, W. Yu, J. Ji, Y. Wang, 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。[在线]. 可用地址：https://arxiv.org/abs/2312.08914

[16] M. Xu, "Every software as an agent: Blueprint and case study," arXiv preprint arXiv:2502.04747, 2025.
[16] M. Xu，“每个软件皆为代理：蓝图与案例研究，”arXiv 预印本 arXiv:2502.04747，2025年。

[17] B. Zheng, B. Gou, J. Kil, H. Sun, and Y. Su, "Gpt-4v(ision) is a generalist web agent, if grounded," 2024. [Online]. Available: https://arxiv.org/abs/2401.01614
[17] B. Zheng, B. Gou, J. Kil, H. Sun, 和 Y. Su，“GPT-4V(ision) 是通用网络代理，前提是有基础支持，”2024年。[在线]. 可用地址：https://arxiv.org/abs/2401.01614

[18] C. Zhang, Z. Yang, J. Liu, Y. Han, X. Chen, Z. Huang, B. Fu, and G. Yu, "Appagent: Multimodal agents as smartphone users," 2023. [Online]. Available: https://arxiv.org/abs/2312.13771
[18] C. Zhang, Z. Yang, J. Liu, Y. Han, X. Chen, Z. Huang, B. Fu, 和 G. Yu，“Appagent：作为智能手机用户的多模态代理，”2023年。[在线]. 可用地址：https://arxiv.org/abs/2312.13771

[19] 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," arXiv preprint arXiv:2402.07939, 2024.
[19] 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：面向 Windows 操作系统交互的界面聚焦代理，”arXiv 预印本 arXiv:2402.07939，2024年。

[20] Y. Guan, D. Wang, Z. Chu, S. Wang, F. Ni, R. Song, L. Li, J. Gu, and C. Zhuang, "Intelligent virtual assistants with Ilm-based process automation," ArXiv, vol. abs/2312.06677, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:266174422
[20] Y. Guan, D. Wang, Z. Chu, S. Wang, F. Ni, R. Song, L. Li, J. Gu, 和 C. Zhuang，“基于Ilm的流程自动化的智能虚拟助手，” ArXiv, 卷. abs/2312.06677, 2023. [在线]. 可用: https://api.semanticscholar.org/CorpusID:266174422

[21] Y. Zhang, X. Zhao, J. Yin, L. Zhang, and Z. Chen, "Operating system and artificial intelligence: A systematic review," arXiv preprint arXiv:2407.14567, 2024.
[21] Y. Zhang, X. Zhao, J. Yin, L. Zhang, 和 Z. Chen，“操作系统与人工智能：系统综述，” arXiv预印本 arXiv:2407.14567, 2024.

[22] K. Mei, Z. Li, S. Xu, R. Ye, Y. Ge, and Y. Zhang, "Aios: LIm agent operating system," arXiv e-prints, pp. arXiv-2403, 2024.
[22] K. Mei, Z. Li, S. Xu, R. Ye, Y. Ge, 和 Y. Zhang, “Aios：LIm代理操作系统,” arXiv电子预印本, 页码 arXiv-2403, 2024.

[23] W. Aljedaani, A. Habib, A. Aljohani, M. M. Eler, and Y. Feng, "Does chatgpt generate accessible code? investigating accessibility challenges in Ilm-generated source code," in International Cross-Disciplinary Conference on Web Accessibility, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:273550267
[23] W. Aljedaani, A. Habib, A. Aljohani, M. M. Eler, 和 Y. Feng, “ChatGPT生成的代码是否具备无障碍性？探讨Ilm生成源代码中的无障碍挑战,” 载于国际跨学科网络无障碍会议, 2024. [在线]. 可用: https://api.semanticscholar.org/CorpusID:273550267

[24] D. Chin, Y. Wang, and G. G. Xia, "Human-centered Ilm-agent user interface: A position paper," ArXiv, vol. abs/2405.13050, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID: 269982753
[24] D. Chin, Y. Wang, 和 G. G. Xia, “以人为中心的Ilm代理用户界面：立场论文,” ArXiv, 卷. abs/2405.13050, 2024. [在线]. 可用: https://api.semanticscholar.org/CorpusID: 269982753

[25] 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
[25] K. Cheng, Q. Sun, Y. Chu, F. Xu, Y. Li, J. Zhang, 和 Z. Wu, “Seeclick：利用GUI定位实现高级视觉GUI代理,” 2024. [在线]. 可用: https://arxiv.org/abs/2401.10935

[26] M. Zhuge, C. Zhao, D. R. Ashley, W. Wang, D. Khizbullin, Y. Xiong, Z. Liu, E. Chang, R. Krishnamoorthi, Y. Tian, Y. Shi, V. Chandra, and J. Schmidhuber, "Agent-as-a-judge: Evaluate agents with agents," 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:273350802
[26] M. Zhuge, C. Zhao, D. R. Ashley, W. Wang, D. Khizbullin, Y. Xiong, Z. Liu, E. Chang, R. Krishnamoorthi, Y. Tian, Y. Shi, V. Chandra, 和 J. Schmidhuber, “代理即评审：用代理评估代理,” 2024. [在线]. 可用: https://api.semanticscholar.org/CorpusID:273350802

[27] K. Li and M. Wu, Effective GUI testing automation: Developing an automated GUI testing tool. John Wiley & Sons, 2006.
[27] K. Li 和 M. Wu, 有效的GUI测试自动化：开发自动化GUI测试工具. John Wiley & Sons, 2006.

[28] O. Rodríguez-Valdés, T. E. Vos, P. Aho, and B. Marín, "30 years of automated gui testing: a bibliometric analysis," in Quality of Information and Communications Technology: 14th International Conference, QUATIC 2021, Algarve, Portugal, September 8-11, 2021, Proceedings 14. Springer, 2021, pp. 473-488.
[28] O. Rodríguez-Valdés, T. E. Vos, P. Aho, 和 B. Marín, “30年自动化GUI测试：文献计量分析,” 载于信息与通信技术质量：第14届国际会议QUATIC 2021，葡萄牙阿尔加维，2021年9月8-11日，会议论文集14. Springer, 2021, 页473-488.

[29] Y. L. Arnatovich and L. Wang, "A systematic literature review of automated techniques for functional gui testing of mobile applications," arXiv preprint arXiv:1812.11470, 2018.
[29] Y. L. Arnatovich 和 L. Wang, “移动应用功能性GUI自动化测试技术的系统文献综述,” arXiv预印本 arXiv:1812.11470, 2018.

[30] K. S. Said, L. Nie, A. A. Ajibode, and X. Zhou, "Gui testing for mobile applications: objectives, approaches and challenges," in Proceedings of the 12th Asia-Pacific Symposium on Internetware, 2020, pp. 51-60.
[30] K. S. Said, L. Nie, A. A. Ajibode, 和 X. Zhou, “移动应用的GUI测试：目标、方法与挑战，”载于第12届亚太互联网软件研讨会论文集，2020年，第51-60页。

[31] X. Li, "Gui testing for android applications: a survey," in 2023 7th International Conference on Computer, Software and Modeling (ICCSM). IEEE, 2023, pp. 6-10.
[31] X. Li, “Android应用的GUI测试：综述，”载于2023年第七届计算机、软件与建模国际会议（ICCSM）。IEEE，2023年，第6-10页。

[32] J.-J. Oksanen, "Test automation for windows gui application," 2023.
[32] J.-J. Oksanen, “Windows GUI应用的测试自动化，”2023年。

[33] P. S. Deshmukh, S. S. Date, P. N. Mahalle, and J. Barot, "Automated gui testing for enhancing user experience (ux): A survey of the state of the art," in International Conference on ICT for Sustainable Development. Springer, 2023, pp. 619-628.
[33] P. S. Deshmukh, S. S. Date, P. N. Mahalle, 和 J. Barot, “提升用户体验（UX）的自动化GUI测试：现状综述，”载于国际可持续发展信息通信技术会议。Springer，2023年，第619-628页。

[34] M. Bajammal, A. Stocco, D. Mazinanian, and A. Mesbah, "A survey on the use of computer vision to improve software engineering tasks," IEEE Transactions on Software Engineering, vol. 48, no. 5, pp. 1722-1742, 2020.
[34] M. Bajammal, A. Stocco, D. Mazinanian, 和 A. Mesbah, “利用计算机视觉提升软件工程任务的综述，”IEEE软件工程汇刊，卷48，第5期，第1722-1742页，2020年。

[35] S. Yu, C. Fang, Z. Tuo, Q. Zhang, C. Chen, Z. Chen, and Z. Su, "Vision-based mobile app gui testing: A survey," arXiv preprint arXiv:2310.13518, 2023.
[35] S. Yu, C. Fang, Z. Tuo, Q. Zhang, C. Chen, Z. Chen, 和 Z. Su, “基于视觉的移动应用GUI测试：综述，”arXiv预印本 arXiv:2310.13518，2023年。

[36] R. Syed, S. Suriadi, M. Adams, W. Bandara, S. J. Leemans, C. Ouyang, A. H. Ter Hofstede, I. Van De Weerd, M. T. Wynn, and H. A. Reijers, "Robotic process automation: contemporary themes and challenges," Computers in Industry, vol. 115, p. 103162, 2020.
[36] R. Syed, S. Suriadi, M. Adams, W. Bandara, S. J. Leemans, C. Ouyang, A. H. Ter Hofstede, I. Van De Weerd, M. T. Wynn, 和 H. A. Reijers, “机器人流程自动化：当代主题与挑战，”工业计算机，卷115，文章号103162，2020年。

[37] T. Chakraborti, V. Isahagian, R. Khalaf, Y. Khazaeni, V. Muthusamy, Y. Rizk, and M. Unuvar, "From robotic process automation to intelligent process automation: -emerging trends-," in Business Process Management: Blockchain and Robotic Process Automation Forum: BPM 2020 Blockchain and RPA Forum, Seville, Spain, September 13-18, 2020, Proceedings 18. Springer, 2020, pp. 215-228.
[37] T. Chakraborti, V. Isahagian, R. Khalaf, Y. Khazaeni, V. Muthusamy, Y. Rizk, 和 M. Unuvar, “从机器人流程自动化到智能流程自动化：新兴趋势，”载于业务流程管理：区块链与机器人流程自动化论坛：BPM 2020区块链与RPA论坛，西班牙塞维利亚，2020年9月13-18日，论文集18。Springer，2020年，第215-228页。

[38] J. G. Enríquez, A. Jiménez-Ramírez, F. J. Domínguez-Mayo, and J. A. García-García, "Robotic process automation: a scientific and industrial systematic mapping study," IEEE Access, vol. 8, pp. 39113-39129, 2020.
[38] J. G. Enríquez, A. Jiménez-Ramírez, F. J. Domínguez-Mayo, 和 J. A. García-García, “机器人流程自动化：科学与工业系统映射研究，”IEEE Access，卷8，第39113-39129页，2020年。

[39] J. Ribeiro, R. Lima, T. Eckhardt, and S. Paiva, "Robotic process automation and artificial intelligence in industry 4.0-a literature review," Procedia Computer Science, vol. 181, pp. 51-58, 2021.
[39] J. Ribeiro, R. Lima, T. Eckhardt, 和 S. Paiva, “工业4.0中的机器人流程自动化与人工智能——文献综述，”计算机科学程序，卷181，第51-58页，2021年。

[40] M. Nass, E. Alégroth, and R. Feldt, "Why many challenges with gui test automation (will) remain," Information and Software Technology, vol. 138, p. 106625, 2021.
[40] M. Nass, E. Alégroth, 和 R. Feldt, “为何GUI测试自动化的诸多挑战依然存在（或将持续存在），”信息与软件技术，卷138，文章号106625，2021年。

[41] S. Agostinelli, A. Marrella, and M. Mecella, "Research challenges for intelligent robotic process automation," in Business Process Management Workshops: BPM 2019 International Workshops, Vienna, Austria, September 1-6, 2019, Revised Selected Papers 17. Springer, 2019, pp. 12-18.
[41] S. Agostinelli, A. Marrella, 和 M. Mecella, “智能机器人流程自动化的研究挑战，”载于业务流程管理研讨会：BPM 2019国际研讨会，奥地利维也纳，2019年9月1-6日，修订精选论文17。Springer，2019年，第12-18页。

[42] A. Wali, S. Mahamad, and S. Sulaiman, "Task automation intelligent agents: A review," Future Internet, vol. 15, no. 6, p. 196, 2023.
[42] A. Wali, S. Mahamad, 和 S. Sulaiman, “任务自动化智能代理：综述，”未来互联网，卷15，第6期，文章196，2023年。

[43] P. Zhao, Z. Jin, and N. Cheng, "An in-depth survey of large language model-based artificial intelligence agents," arXiv preprint arXiv:2309.14365, 2023.
[43] P. Zhao, Z. Jin, 和 N. Cheng, “基于大型语言模型的人工智能代理深入综述，”arXiv预印本 arXiv:2309.14365，2023年。

[44] Y. Cheng, C. Zhang, Z. Zhang, X. Meng, S. Hong, W. Li, Z. Wang, Z. Wang, F. Yin, J. Zhao et al., "Exploring large language model based intelligent agents: Definitions, methods, and prospects," arXiv preprint arXiv:2401.03428, 2024.
[44] Y. Cheng, C. Zhang, Z. Zhang, X. Meng, S. Hong, W. Li, Z. Wang, Z. Wang, F. Yin, J. Zhao 等, “探索基于大型语言模型的智能代理：定义、方法与前景，”arXiv预印本 arXiv:2401.03428，2024年。

[45] Y. Li, H. Wen, W. Wang, X. Li, Y. Yuan, G. Liu, J. Liu, W. Xu, X. Wang, Y. Sun et al., "Personal llm agents: Insights and survey about the capability, efficiency and security," arXiv preprint arXiv:2401.05459, 2024.
[45] Y. Li, H. Wen, W. Wang, X. Li, Y. Yuan, G. Liu, J. Liu, W. Xu, X. Wang, Y. Sun 等，“个人 llm 代理：关于能力、效率和安全性的见解与综述，”arXiv 预印本 arXiv:2401.05459，2024。

[46] Z. Xi, W. Chen, X. Guo, W. He, Y. Ding, B. Hong, M. Zhang, J. Wang, S. Jin, E. Zhou et al., "The rise and potential of large language model based agents: A survey," arXiv preprint arXiv:2309.07864, 2023.
[46] Z. Xi, W. Chen, X. Guo, W. He, Y. Ding, B. Hong, M. Zhang, J. Wang, S. Jin, E. Zhou 等，“基于大型语言模型的代理的兴起与潜力：综述，”arXiv 预印本 arXiv:2309.07864，2023。

[47] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin et al., "A survey on large language model based autonomous agents," Frontiers of Computer Science, vol. 18, no. 6, p. 186345, 2024.
[47] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin 等，“基于大型语言模型的自主代理综述，”《计算机科学前沿》，第18卷，第6期，页186345，2024。

[48] T. Guo, X. Chen, Y. Wang, R. Chang, S. Pei, N. V. Chawla, O. Wiest, and X. Zhang, "Large language model based multi-agents: A survey of progress and challenges," arXiv preprint arXiv:2402.01680, 2024.
[48] T. Guo, X. Chen, Y. Wang, R. Chang, S. Pei, N. V. Chawla, O. Wiest, 和 X. Zhang，“基于大型语言模型的多代理：进展与挑战综述，”arXiv 预印本 arXiv:2402.01680，2024。

[49] S. Han, Q. Zhang, Y. Yao, W. Jin, Z. Xu, and C. He, "LIm multi-agent systems: Challenges and open problems," arXiv preprint arXiv:2402.03578, 2024.
[49] S. Han, Q. Zhang, Y. Yao, W. Jin, Z. Xu, 和 C. He，“LIm 多代理系统：挑战与未解问题，”arXiv 预印本 arXiv:2402.03578，2024。

[50] C. Sun, S. Huang, and D. Pompili, "Llm-based multi-agent reinforcement learning: Current and future directions," arXiv preprint arXiv:2405.11106, 2024.
[50] C. Sun, S. Huang, 和 D. Pompili，“基于 Llm 的多代理强化学习：现状与未来方向，”arXiv 预印本 arXiv:2405.11106，2024。

[51] X. Huang, W. Liu, X. Chen, X. Wang, H. Wang, D. Lian, Y. Wang, R. Tang, and E. Chen, "Understanding the planning of Ilm agents: A survey," arXiv preprint arXiv:2402.02716, 2024.
[51] X. Huang, W. Liu, X. Chen, X. Wang, H. Wang, D. Lian, Y. Wang, R. Tang, 和 E. Chen，“理解 Ilm 代理的规划：综述，”arXiv 预印本 arXiv:2402.02716，2024。

[52] M. Aghzal, E. Plaku, G. J. Stein, and Z. Yao, "A survey on large language models for automated planning," arXiv preprint arXiv:2502.12435, 2025.
[52] M. Aghzal, E. Plaku, G. J. Stein, 和 Z. Yao，“大型语言模型在自动规划中的应用综述，”arXiv 预印本 arXiv:2502.12435，2025。

[53] J. Zheng, C. Shi, X. Cai, Q. Li, D. Zhang, C. Li, D. Yu, and Q. Ma, "Lifelong learning of large language model based agents: A roadmap," arXiv preprint arXiv:2501.07278, 2025.
[53] J. Zheng, C. Shi, X. Cai, Q. Li, D. Zhang, C. Li, D. Yu, 和 Q. Ma，“基于大型语言模型代理的终身学习：路线图，”arXiv 预印本 arXiv:2501.07278，2025。

[54] Z. Zhang, X. Bo, C. Ma, R. Li, X. Chen, Q. Dai, J. Zhu, Z. Dong, and J.-R. Wen, "A survey on the memory mechanism of large language model based agents," arXiv preprint arXiv:2404.13501, 2024.
[54] Z. Zhang, X. Bo, C. Ma, R. Li, X. Chen, Q. Dai, J. Zhu, Z. Dong, 和 J.-R. Wen，“基于大型语言模型代理的记忆机制综述，”arXiv 预印本 arXiv:2404.13501，2024。

[55] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. Yi, C. Wang, Y. Wang et al., "A survey on evaluation of large language models," ACM Transactions on Intelligent Systems and Technology, vol. 15, no. 3, pp. 1-45, 2024.
[55] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. Yi, C. Wang, Y. Wang 等，“大型语言模型评估综述，”《ACM 智能系统与技术汇刊》，第15卷，第3期，页1-45，2024。

[56] L. Li, G. Chen, H. Shi, J. Xiao, and L. Chen, "A survey on multimodal benchmarks: In the era of large ai models," arXiv preprint arXiv:2409.18142, 2024.
[56] L. Li, G. Chen, H. Shi, J. Xiao, 和 L. Chen，“多模态基准综述：大型人工智能模型时代，”arXiv 预印本 arXiv:2409.18142，2024。

[57] Z. Li, X. Wu, H. Du, H. Nghiem, and G. Shi, "Benchmark evaluations, applications, and challenges of large vision language models: A survey," arXiv preprint arXiv:2501.02189, 2025.
[57] Z. Li, X. Wu, H. Du, H. Nghiem, 和 G. Shi，“大型视觉语言模型的基准评估、应用与挑战综述，”arXiv 预印本 arXiv:2501.02189，2025。

[58] J. Huang and J. Zhang, "A survey on evaluation of multimodal large language models," arXiv preprint arXiv:2408.15769, 2024.
[58] J. Huang 和 J. Zhang，“多模态大型语言模型评估综述，”arXiv 预印本 arXiv:2408.15769，2024。

[59] J. Xie, Z. Chen, R. Zhang, X. Wan, and G. Li, "Large multimodal agents: A survey," arXiv preprint arXiv:2402.15116, 2024.
[59] J. Xie, Z. Chen, R. Zhang, X. Wan, 和 G. Li，“大型多模态代理综述，”arXiv 预印本 arXiv:2402.15116，2024。

[60] Z. Durante, Q. Huang, N. Wake, R. Gong, J. S. Park, B. Sarkar, R. Taori, Y. Noda, D. Terzopoulos, Y. Choi et al., "Agent ai: Surveying the horizons of multimodal interaction," arXiv preprint arXiv:2401.03568, 2024.
[60] Z. Durante, Q. Huang, N. Wake, R. Gong, J. S. Park, B. Sarkar, R. Taori, Y. Noda, D. Terzopoulos, Y. Choi 等, "Agent ai: 多模态交互的前沿综述," arXiv 预印本 arXiv:2401.03568, 2024.

[61] B. Wu, Y. Li, M. Fang, Z. Song, Z. Zhang, Y. Wei, and L. Chen, "Foundations and recent trends in multimodal mobile agents: A survey," arXiv preprint arXiv:2411.02006, 2024.
[61] B. Wu, Y. Li, M. Fang, Z. Song, Z. Zhang, Y. Wei, 和 L. Chen, "多模态移动代理的基础与最新趋势综述," arXiv 预印本 arXiv:2411.02006, 2024.

[62] S. Wang, W. Liu, J. Chen, W. Gan, X. Zeng, S. Yu, X. Hao, K. Shao, Y. Wang, and R. Tang, "Gui agents with foundation models: A comprehensive survey," 2024. [Online]. Available: https://arxiv.org/abs/2411.04890
[62] S. Wang, W. Liu, J. Chen, W. Gan, X. Zeng, S. Yu, X. Hao, K. Shao, Y. Wang, 和 R. Tang, "基于基础模型的图形用户界面代理：全面综述," 2024. [在线]. 可获取: https://arxiv.org/abs/2411.04890

[63] M. Gao, W. Bu, B. Miao, Y. Wu, Y. Li, J. Li, S. Tang, Q. Wu, Y. Zhuang, and M. Wang, "Generalist virtual agents: A survey on autonomous agents across digital platforms," arXiv preprint arXiv:2411.10943, 2024.
[63] M. Gao, W. Bu, B. Miao, Y. Wu, Y. Li, J. Li, S. Tang, Q. Wu, Y. Zhuang, 和 M. Wang, "通用虚拟代理：跨数字平台自主代理综述," arXiv 预印本 arXiv:2411.10943, 2024.

[64] 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
[64] 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, "图形用户界面代理综述," 2024. [在线]. 可获取: https://arxiv.org/abs/2412.13501

[65] G. Liu, P. Zhao, L. Liu, Y. Guo, H. Xiao, W. Lin, Y. Chai, Y. Han, S. Ren, H. Wang et al., "Llm-powered gui agents in phone automation: Surveying progress and prospects," arXiv preprint arXiv:2504.19838, 2025.
[65] G. Liu, P. Zhao, L. Liu, Y. Guo, H. Xiao, W. Lin, Y. Chai, Y. Han, S. Ren, H. Wang 等, "基于大型语言模型的手机自动化图形用户界面代理：进展与前景综述," arXiv 预印本 arXiv:2504.19838, 2025.

[66] X. Hu, T. Xiong, B. Yi, Z. Wei, R. Xiao, Y. Chen, J. Ye, M. Tao, X. Zhou, Z. Zhao et al., "Os agents: A survey on mllm-based agents for general computing devices use," 2024.
[66] X. Hu, T. Xiong, B. Yi, Z. Wei, R. Xiao, Y. Chen, J. Ye, M. Tao, X. Zhou, Z. Zhao 等, "操作系统代理：基于多模态大型语言模型的通用计算设备代理综述," 2024.

[67] Y. Shi, W. Yu, W. Yao, W. Chen, and N. Liu, "Towards trustworthy gui agents: A survey," arXiv preprint arXiv:2503.23434, 2025.
[67] Y. Shi, W. Yu, W. Yao, W. Chen, 和 N. Liu, "迈向可信赖的图形用户界面代理：综述," arXiv 预印本 arXiv:2503.23434, 2025.

[68] L. Ning, Z. Liang, Z. Jiang, H. Qu, Y. Ding, W. Fan, X.-y. Wei, S. Lin, H. Liu, P. S. Yu et al., "A survey of webagents: Towards next-generation ai agents for web automation with large foundation models," arXiv preprint arXiv:2503.23350, 2025.
[68] L. Ning, Z. Liang, Z. Jiang, H. Qu, Y. Ding, W. Fan, X.-y. Wei, S. Lin, H. Liu, P. S. Yu 等, "网络代理综述：面向基于大型基础模型的下一代网页自动化人工智能代理," arXiv 预印本 arXiv:2503.23350, 2025.

[69] F. Tang, H. Xu, H. Zhang, S. Chen, X. Wu, Y. Shen, W. Zhang, G. Hou, Z. Tan, Y. Yan, K. Song, J. Shao, W. Lu, J. Xiao, and Y. Zhuang, "A survey on (m)llm-based gui agents," 2025. [Online]. Available: https://arxiv.org/abs/2504.13865
[69] F. Tang, H. Xu, H. Zhang, S. Chen, X. Wu, Y. Shen, W. Zhang, G. Hou, Z. Tan, Y. Yan, K. Song, J. Shao, W. Lu, J. Xiao, 和 Y. Zhuang, "基于(多模态)大型语言模型的图形用户界面代理综述," 2025. [在线]. 可获取: https://arxiv.org/abs/2504.13865

[70] J. Li and K. Huang, "A summary on gui agents with foundation models enhanced by reinforcement learning," 2025. [Online]. Available: https://arxiv.org/abs/2504.20464
[70] J. Li 和 K. Huang, "基于强化学习增强的基础模型图形用户界面代理综述," 2025. [在线]. 可获取: https://arxiv.org/abs/2504.20464

[71] P. J. Sager, B. Meyer, P. Yan, R. von Wartburg-Kottler, L. Etaiwi, A. Enayati, G. Nobel, A. Abdulkadir, B. F. Grewe, and T. Stadelmann, "Ai agents for computer use: A review of instruction-based computer control, gui automation, and operator assistants," arXiv preprint arXiv:2501.16150, 2025.
[71] P. J. Sager, B. Meyer, P. Yan, R. von Wartburg-Kottler, L. Etaiwi, A. Enayati, G. Nobel, A. Abdulkadir, B. F. Grewe, 和 T. Stadelmann, “用于计算机使用的AI代理：基于指令的计算机控制、GUI自动化和操作员助手的综述，” arXiv预印本 arXiv:2501.16150, 2025。

[72] T. S. d. Moura, E. L. Alves, H. F. d. Figueirèdo, and C. d. S. Baptista, "Cytestion: Automated gui testing for web applications," in Proceedings of the XXXVII Brazilian Symposium on Software Engineering, 2023, pp. 388-397.
[72] T. S. d. Moura, E. L. Alves, H. F. d. Figueirèdo, 和 C. d. S. Baptista, “Cytestion：面向Web应用的自动化GUI测试，” 载于第37届巴西软件工程研讨会论文集，2023，页388-397。

[73] T. Yeh, T.-H. Chang, and R. C. Miller, "Sikuli: using gui screenshots for search and automation," in Proceedings of the 22nd annual ACM symposium on User interface software and technology, 2009, pp. 183-192.
[73] T. Yeh, T.-H. Chang, 和 R. C. Miller, “Sikuli：利用GUI截图进行搜索与自动化，” 载于第22届ACM用户界面软件与技术年会论文集，2009，页183-192。

[74] C. E. Shannon, "Prediction and entropy of printed english," Bell system technical journal, vol. 30, no. 1, pp. 50-64, 1951.
[74] C. E. Shannon, “印刷英语的预测与熵，” 贝尔系统技术期刊，卷30，第1期，页50-64，1951。

[75] W. B. Cavnar, J. M. Trenkle et al., "N-gram-based text categorization," in Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval, vol. 161175. Ann Arbor, Michigan, 1994, p. 14.
[75] W. B. Cavnar, J. M. Trenkle 等，“基于N-gram的文本分类，” 载于SDAIR-94第三届文档分析与信息检索年会论文集，卷161175。密歇根州安娜堡，1994，页14。

[76] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," arXiv preprint arXiv:1412.3555, 2014.
[76] J. Chung, C. Gulcehre, K. Cho, 和 Y. Bengio, “门控循环神经网络在序列建模中的实证评估，” arXiv预印本 arXiv:1412.3555, 2014。

[77] B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal et al., "Language models are few-shot learners," arXiv preprint arXiv:2005.14165, vol. 1, 2020.
[77] B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal 等，“语言模型是少样本学习者，” arXiv预印本 arXiv:2005.14165, 卷1, 2020。

[78] J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le, "Finetuned language models are zero-shot learners," arXiv preprint arXiv:2109.01652, 2021.
[78] J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, 和 Q. V. Le, “微调语言模型是零样本学习者，” arXiv预印本 arXiv:2109.01652, 2021。

[79] L. R. Medsker, L. Jain et al., "Recurrent neural networks," Design and Applications, vol. 5, no. 64-67, p. 2, 2001.
[79] L. R. Medsker, L. Jain 等，“循环神经网络，” 设计与应用，卷5，第64-67期，页2，2001。

[80] S. Hochreiter, "Long short-term memory," Neural Computation MIT-Press, 1997.
[80] S. Hochreiter, “长短期记忆（LSTM），” 神经计算，MIT出版社，1997。

[81] A. Vaswani, "Attention is all you need," Advances in Neural Information Processing Systems, 2017.
[81] A. Vaswani, “注意力机制即一切，” 神经信息处理系统进展，2017。

[82] J. Devlin, "Bert: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805, 2018.
[82] J. Devlin, “BERT：用于语言理解的深度双向变换器预训练，” arXiv预印本 arXiv:1810.04805, 2018。

[83] Y. Liu, "Roberta: A robustly optimized bert pretraining approach," arXiv preprint arXiv:1907.11692, vol. 364, 2019.
[83] Y. Liu, “RoBERTa：一种鲁棒优化的BERT预训练方法，” arXiv预印本 arXiv:1907.11692, 卷364, 2019。

[84] Z. Lan, "Albert: A lite bert for self-supervised learning of language representations," arXiv preprint arXiv:1909.11942, 2019.
[84] Z. Lan, “ALBERT：一种轻量级BERT用于语言表示的自监督学习，” arXiv预印本 arXiv:1909.11942, 2019。

[85] A. Radford, "Improving language understanding by generative pre-training," 2018.
[85] A. Radford, “通过生成式预训练提升语言理解，” 2018。

[86] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever et al., "Language models are unsupervised multitask learners," OpenAI blog, vol. 1, no. 8, p. 9, 2019.
[86] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever 等，“语言模型是无监督多任务学习者，” OpenAI博客，卷1，第8期，页9，2019。

[87] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," Journal of machine learning research, vol. 21, no. 140, pp. 1-67, 2020.
[87] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, 和 P. J. Liu, “探索统一文本到文本转换器（text-to-text transformer）迁移学习的极限,” 机器学习研究杂志, 第21卷, 第140期, 页1-67, 2020年。

[88] M. Lewis, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," arXiv preprint arXiv:1910.13461, 2019.
[88] M. Lewis, “Bart：用于自然语言生成、翻译和理解的去噪序列到序列预训练,” arXiv预印本 arXiv:1910.13461, 2019年。

[89] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray et al., "Training language models to follow instructions with human feedback," Advances in neural information processing systems, vol. 35, pp. 27730-27744, 2022.
[89] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray 等, “通过人类反馈训练语言模型以遵循指令,” 神经信息处理系统进展, 第35卷, 页27730-27744, 2022年。

[90] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat et al., "Gpt-4 technical report," arXiv preprint arXiv:2303.08774, 2023.
[90] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat 等, “GPT-4技术报告,” arXiv预印本 arXiv:2303.08774, 2023年。

[91] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan et al., "The llama 3 herd of models," arXiv preprint arXiv:2407.21783, 2024.
[91] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan 等, “Ilama 3模型群,” arXiv预印本 arXiv:2407.21783, 2024年。

[92] G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican et al., "Gemini: a family of highly capable multimodal models," arXiv preprint arXiv:2312.11805, 2023.
[92] G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican 等, “Gemini：一系列高性能多模态模型,” arXiv预印本 arXiv:2312.11805, 2023年。

[93] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford et al., "Gpt-40 system card," arXiv preprint arXiv:2410.21276, 2024.
[93] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford 等, “GPT-40系统说明,” arXiv预印本 arXiv:2410.21276, 2024年。

[94] Y. Jiang, C. Zhang, S. He, Z. Yang, M. Ma, S. Qin, Y. Kang, Y. Dang, S. Rajmohan, Q. Lin et al., "Xpert: Empowering incident management with query recommendations via large language models," in Proceedings of the IEEE/ACM 46th International Conference on Software Engineering, 2024, pp. 1-13.
[94] Y. Jiang, C. Zhang, S. He, Z. Yang, M. Ma, S. Qin, Y. Kang, Y. Dang, S. Rajmohan, Q. Lin 等, “Xpert：通过大型语言模型利用查询推荐增强事件管理,” IEEE/ACM第46届国际软件工程会议论文集, 2024年, 页1-13。

[95] C. Zhang, Z. Ma, Y. Wu, S. He, S. Qin, M. Ma, X. Qin, Y. Kang, Y. Liang, X. Gou et al., "Allhands: Ask me anything on large-scale verbatim feedback via large language models," arXiv preprint arXiv:2403.15157, 2024.
[95] C. Zhang, Z. Ma, Y. Wu, S. He, S. Qin, M. Ma, X. Qin, Y. Kang, Y. Liang, X. Gou 等, “Allhands：基于大型语言模型的大规模逐字反馈问答,” arXiv预印本 arXiv:2403.15157, 2024年。

[96] J. Liu, C. Zhang, J. Qian, M. Ma, S. Qin, C. Bansal, Q. Lin, S. Rajmohan, and D. Zhang, "Large language models can deliver accurate and interpretable time series anomaly detection," arXiv preprint arXiv:2405.15370, 2024.
[96] J. Liu, C. Zhang, J. Qian, M. Ma, S. Qin, C. Bansal, Q. Lin, S. Rajmohan, 和 D. Zhang, “大型语言模型能够实现准确且可解释的时间序列异常检测,” arXiv预印本 arXiv:2405.15370, 2024年。

[97] Q. Dong, L. Li, D. Dai, C. Zheng, J. Ma, R. Li, H. Xia, J. Xu, Z. Wu, T. Liu et al., "A survey on in-context learning," arXiv preprint arXiv:2301.00234, 2022.
[97] Q. Dong, L. Li, D. Dai, C. Zheng, J. Ma, R. Li, H. Xia, J. Xu, Z. Wu, T. Liu 等, “上下文学习综述,” arXiv预印本 arXiv:2301.00234, 2022年。

[98] S. Zhang, L. Dong, X. Li, S. Zhang, X. Sun, S. Wang, J. Li, R. Hu, T. Zhang, F. Wu et al., "Instruction tuning for large language models: A survey," arXiv preprint arXiv:2308.10792, 2023.
[98] S. Zhang, L. Dong, X. Li, S. Zhang, X. Sun, S. Wang, J. Li, R. Hu, T. Zhang, F. Wu 等, “大型语言模型的指令调优综述,” arXiv预印本 arXiv:2308.10792, 2023年。

[99] J. Huang and K. C.-C. Chang, "Towards reasoning in large language models: A survey," arXiv preprint arXiv:2212.10403, 2022.
[99] J. Huang 和 K. C.-C. Chang, “面向大规模语言模型推理的综述,” arXiv 预印本 arXiv:2212.10403, 2022.

[100] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou et al., "Chain-of-thought prompting elicits reasoning in large language models," Advances in neural information processing systems, vol. 35, pp. 24824-24837, 2022.
[100] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou 等, “链式思维提示激发大规模语言模型的推理能力,” 神经信息处理系统进展, 第35卷, 页24824-24837, 2022.

[101] 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," arXiv preprint arXiv:2311.04254, 2023.
[101] R. Ding, C. Zhang, L. Wang, Y. Xu, M. Ma, W. Zhang, S. Qin, S. Rajmohan, Q. Lin, 和 D. Zhang, “思维的全貌：挑战彭罗斯三角定律的思维生成,” arXiv 预印本 arXiv:2311.04254, 2023.

[102] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. D. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman et al., "Evaluating large language models trained on code," arXiv preprint arXiv:2107.03374, 2021.
[102] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. D. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman 等, “评估基于代码训练的大规模语言模型,” arXiv 预印本 arXiv:2107.03374, 2021.

[103] T. D. White, G. Fraser, and G. J. Brown, "Improving random gui testing with image-based widget detection," in Proceedings of the 28th ACM SIGSOFT international symposium on software testing and analysis, 2019, pp. 307-317.
[103] T. D. White, G. Fraser, 和 G. J. Brown, “通过基于图像的小部件检测改进随机GUI测试,” 载于第28届ACM SIGSOFT国际软件测试与分析研讨会论文集, 2019, 页307-317.

[104] G. Kim, P. Baldi, and S. McAleer, "Language models can solve computer tasks," 2023. [Online]. Available: https://arxiv.org/abs/2303.17491
[104] G. Kim, P. Baldi, 和 S. McAleer, “语言模型能够解决计算机任务,” 2023. [在线]. 可获取: https://arxiv.org/abs/2303.17491

[105] B. Qiao, L. Li, X. Zhang, S. He, Y. Kang, C. Zhang, F. Yang, H. Dong, J. Zhang, L. Wang et al., "Taskweaver: A code-first agent framework," arXiv preprint arXiv:2311.17541, 2023.
[105] B. Qiao, L. Li, X. Zhang, S. He, Y. Kang, C. Zhang, F. Yang, H. Dong, J. Zhang, L. Wang 等, “Taskweaver：一个以代码为先的代理框架,” arXiv 预印本 arXiv:2311.17541, 2023.

[106] M. A. Boshart and M. J. Kosa, "Growing a gui from an xml tree," ACM SIGCSE Bulletin, vol. 35, no. 3, pp. 223-223, 2003.
[106] M. A. Boshart 和 M. J. Kosa, “从XML树生成GUI,” ACM SIGCSE通报, 第35卷, 第3期, 页223-223, 2003.

[107] Y. Li and O. Hilliges, Artificial intelligence for human computer interaction: a modern approach. Springer, 2021.
[107] Y. Li 和 O. Hilliges, 人机交互的人工智能：现代方法. Springer, 2021.

[108] H. Y. Abuaddous, A. M. Saleh, O. Enaizan, F. Ghabban, and A. B. Al-Badareen, "Automated user experience (ux) testing for mobile application: Strengths and limitations." International Journal of Interactive Mobile Technologies, vol. 16, no. 4, 2022.
[108] H. Y. Abuaddous, A. M. Saleh, O. Enaizan, F. Ghabban, 和 A. B. Al-Badareen, “移动应用的自动化用户体验（UX）测试：优势与局限,” 国际交互式移动技术期刊, 第16卷, 第4期, 2022.

[109] D. Gao, L. Ji, Z. Bai, M. Ouyang, P. Li, D. Mao, Q. Wu, W. Zhang, P. Wang, X. Guo et al., "Assistgui: Task-oriented pc graphical user interface automation," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 13289-13298.
[109] D. Gao, L. Ji, Z. Bai, M. Ouyang, P. Li, D. Mao, Q. Wu, W. Zhang, P. Wang, X. Guo 等, “AssistGUI：面向任务的PC图形用户界面自动化,” 载于IEEE/CVF计算机视觉与模式识别会议论文集, 2024, 页13289-13298.

[110] J. Qian, Z. Shang, S. Yan, Y. Wang, and L. Chen, "Roscript: A visual script driven truly non-intrusive robotic testing system for touch screen applications," in 2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE), 2020, pp. 297-308.
[110] J. Qian, Z. Shang, S. Yan, Y. Wang, 和 L. Chen, “Roscript：一种视觉脚本驱动的真正非侵入式触摸屏应用机器人测试系统,” 载于2020年IEEE/ACM第42届国际软件工程会议（ICSE）, 2020, 页297-308.

[111] A. Bruns, A. Kornstadt, and D. Wichmann, "Web application tests with selenium," IEEE software, vol. 26, no. 5, pp. 88-91, 2009.
[111] A. Bruns, A. Kornstadt, 和 D. Wichmann, “使用Selenium进行Web应用测试,” IEEE软件, 第26卷, 第5期, 页88-91, 2009.

[112] N. Rupp, K. Peschke, M. Köppl, D. Drissner, and T. Zuchner, "Establishment of low-cost laboratory automation processes using autoit and 4-axis robots," SLAS technology, vol. 27, no. 5, pp. 312-318, 2022.
[112] N. Rupp, K. Peschke, M. Köppl, D. Drissner, 和 T. Zuchner, “利用AutoIt和四轴机器人建立低成本实验室自动化流程,” SLAS技术, 第27卷, 第5期, 页312-318, 2022.

[113] M. F. Granda, O. Parra, and B. Alba-Sarango, "Towards a model-driven testing framework for gui test cases generation from user stories." in ENASE, 2021, pp. 453-460.
[113] M. F. Granda, O. Parra, 和 B. Alba-Sarango, “面向从用户故事生成GUI测试用例的模型驱动测试框架探索,” 载于ENASE, 2021, 页453-460.

[114] J. Xu, W. Du, X. Liu, and X. Li, "Llm4workflow: An llm-based automated workflow model generation tool," Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:273465368
[114] J. Xu, W. Du, X. Liu, 和 X. Li, “Llm4workflow：基于LLM的自动化工作流模型生成工具，”发表于第39届IEEE/ACM自动化软件工程国际会议论文集，2024年。[在线]. 可获取：https://api.semanticscholar.org/CorpusID:273465368

[115] R. Gove and J. Faytong, "Machine learning and event-based software testing: classifiers for identifying infeasible gui event sequences," in Advances in computers. Elsevier, 2012, vol. 86, pp. 109-135.
[115] R. Gove 和 J. Faytong, “机器学习与基于事件的软件测试：用于识别不可行GUI事件序列的分类器，”载于《计算机进展》，Elsevier出版社，2012年，第86卷，第109-135页。

[116] T. J.-J. Li, L. Popowski, T. Mitchell, and B. A. Myers, "Screen2vec: Semantic embedding of gui screens and gui components," in Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems, 2021, pp. 1-15.
[116] T. J.-J. Li, L. Popowski, T. Mitchell, 和 B. A. Myers, “Screen2vec：GUI屏幕及组件的语义嵌入，”发表于2021年CHI人机交互大会论文集，2021年，第1-15页。

[117] T.-H. Chang, T. Yeh, and R. C. Miller, "Gui testing using computer vision," in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2010, pp. 1535-1544.
[117] T.-H. Chang, T. Yeh, 和 R. C. Miller, “基于计算机视觉的GUI测试，”发表于SIGCHI人机交互大会论文集，2010年，第1535-1544页。

[118] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," Proceedings of the IEEE, vol. 111, no. 3, pp. 257-276, 2023.
[118] Z. Zou, K. Chen, Z. Shi, Y. Guo, 和 J. Ye, “目标检测二十年综述，”《IEEE汇刊》，第111卷，第3期，2023年，第257-276页。

[119] J. Ye, K. Chen, X. Xie, L. Ma, R. Huang, Y. Chen, Y. Xue, and J. Zhao, "An empirical study of gui widget detection for industrial mobile games," in Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2021, pp. 1427-1437.
[119] J. Ye, K. Chen, X. Xie, L. Ma, R. Huang, Y. Chen, Y. Xue, 和 J. Zhao, “工业移动游戏中GUI控件检测的实证研究，”发表于第29届ACM欧洲软件工程会议与软件工程基础研讨会联合会议论文集，2021年，第1427-1437页。

[120] J. Chen, M. Xie, Z. Xing, C. Chen, X. Xu, L. Zhu, and G. Li, "Object detection for graphical user interface: Old fashioned or deep learning or a combination?" in proceedings of the 28th ACM joint meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2020, pp. 1202-1214.
[120] J. Chen, M. Xie, Z. Xing, C. Chen, X. Xu, L. Zhu, 和 G. Li, “图形用户界面目标检测：传统方法、深度学习还是两者结合？”发表于第28届ACM欧洲软件工程会议与软件工程基础研讨会联合会议论文集，2020年，第1202-1214页。

[121] J. Qian, Y. Ma, C. Lin, and L. Chen, "Accelerating ocr-based widget localization for test automation of gui applications," in Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering, 2022, pp. 1-13.
[121] J. Qian, Y. Ma, C. Lin, 和 L. Chen, “加速基于OCR的控件定位以实现GUI应用的测试自动化，”发表于第37届IEEE/ACM自动化软件工程国际会议论文集，2022年，第1-13页。

[122] O. Gambino, L. Rundo, V. Cannella, S. Vitabile, and R. Pirrone, "A framework for data-driven adaptive gui generation based on dicom," Journal of biomedical informatics, vol. 88, pp. 37-52, 2018.
[122] O. Gambino, L. Rundo, V. Cannella, S. Vitabile, 和 R. Pirrone, “基于DICOM的数据驱动自适应GUI生成框架，”《生物医学信息学杂志》，第88卷，2018年，第37-52页。

[123] J. He, I.-L. Yen, T. Peng, J. Dong, and F. Bastani, "An adaptive user interface generation framework for web services," in 2008 IEEE Congress on Services Part II (services-2 2008). IEEE, 2008, pp. 175-182.
[123] J. He, I.-L. Yen, T. Peng, J. Dong, 和 F. Bastani, “面向Web服务的自适应用户界面生成框架，”发表于2008年IEEE服务大会第二部分（services-2 2008），IEEE，2008年，第175-182页。

[124] Z. Stefanidi, G. Margetis, S. Ntoa, and G. Papagiannakis, "Real-time adaptation of context-aware intelligent user interfaces, for enhanced situational awareness," IEEE Access, vol. 10, pp. 23367- 23393, 2022.
[124] Z. Stefanidi, G. Margetis, S. Ntoa, 和 G. Papagiannakis, “上下文感知智能用户界面的实时自适应，以增强情境感知，”《IEEE Access》，第10卷，2022年，第23367-23393页。

[125] Z. Liu, C. Chen, J. Wang, M. Chen, B. Wu, X. Che, D. Wang, and Q. Wang, "Make Ilm a testing expert: Bringing human-like interaction to mobile gui testing via functionality-aware decisions," in Proceedings of the IEEE/ACM 46th International Conference on Software Engineering, 2024, pp. 1-13.
[125] Z. Liu, C. Chen, J. Wang, M. Chen, B. Wu, X. Che, D. Wang, 和 Q. Wang, “让ILM成为测试专家：通过功能感知决策为移动GUI测试带来类人交互，”发表于第46届IEEE/ACM国际软件工程会议论文集，2024年，第1-13页。

[126] P. Brie, N. Burny, A. Sluỳters, and J. Vanderdonckt, "Evaluating a large language model on searching for gui layouts," Proceedings of the ACM on Human-Computer Interaction, vol. 7, no. EICS, pp. 1-37, 2023.
[126] P. Brie, N. Burny, A. Sluỳters, 和 J. Vanderdonckt, “评估大型语言模型在搜索GUI布局中的表现,” 《ACM人机交互学报》, 第7卷, EICS期, 第1-37页, 2023年。

[127] T. Wetzlmaier, R. Ramler, and W. Putschögl, "A framework for monkey gui testing," in 2016 IEEE international conference on software testing, verification and validation (ICST). IEEE, 2016, pp. 416-423.
[127] T. Wetzlmaier, R. Ramler, 和 W. Putschögl, “猴子GUI测试框架,” 载于2016年IEEE国际软件测试、验证与确认会议（ICST），IEEE, 2016年, 第416-423页。

[128] X. Zeng, D. Li, W. Zheng, F. Xia, Y. Deng, W. Lam, W. Yang, and T. Xie, "Automated test input generation for android: are we really there yet in an industrial case?" in Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, ser. FSE 2016. New York, NY, USA: Association for Computing Machinery, 2016, p. 987-992. [Online]. Available: https://doi.org/10.1145/2950290.2983958
[128] X. Zeng, D. Li, W. Zheng, F. Xia, Y. Deng, W. Lam, W. Yang, 和 T. Xie, “安卓自动测试输入生成：工业案例中我们真的达到目标了吗？” 载于2016年第24届ACM SIGSOFT软件工程基础国际研讨会论文集（FSE 2016），纽约，美国：计算机协会，2016年，第987-992页。[在线]. 可访问：https://doi.org/10.1145/2950290.2983958

[129] A. M. Memon, M. E. Pollack, and M. L. Soffa, "Hierarchical gui test case generation using automated planning," IEEE transactions on software engineering, vol. 27, no. 2, pp. 144-155, 2001.
[129] A. M. Memon, M. E. Pollack, 和 M. L. Soffa, “基于自动规划的分层GUI测试用例生成,” 《IEEE软件工程汇刊》, 第27卷, 第2期, 第144-155页, 2001年。

[130] S. Agostinelli, M. Lupia, A. Marrella, and M. Mecella, "Automated generation of executable rpa scripts from user interface logs," in Business Process Management: Blockchain and Robotic Process Automation Forum: BPM 2020 Blockchain and RPA Forum, Seville, Spain, September 13-18, 2020, Proceedings 18. Springer, 2020, pp. 116-131.
[130] S. Agostinelli, M. Lupia, A. Marrella, 和 M. Mecella, “基于用户界面日志的可执行机器人流程自动化（RPA）脚本自动生成,” 载于《业务流程管理：区块链与机器人流程自动化论坛：BPM 2020区块链与RPA论坛》，西班牙塞维利亚，2020年9月13-18日，论文集18，施普林格，2020年，第116-131页。

[131] A. Memon, I. Banerjee, N. Hashmi, and A. Nagarajan, "Dart: a framework for regression testing "nightly/daily builds" of gui applications," in International Conference on Software Maintenance, 2003. ICSM 2003. Proceedings., 2003, pp. 410-419.
[131] A. Memon, I. Banerjee, N. Hashmi, 和 A. Nagarajan, “Dart：用于GUI应用程序“夜间/每日构建”回归测试的框架,” 载于国际软件维护会议，2003年，ICSM 2003论文集，第410-419页。

[132] Microsoft, "Create desktop flows using record with copilot (preview)," 2024, accessed: 2024-11-16. [Online]. Available: https://learn.microsoft.com/en-us/power-automate/ desktop-flows/create-flow-using-ai-recorder
[132] Microsoft, “使用Copilot录制创建桌面流程（预览）,” 2024年，访问时间：2024-11-16。[在线]. 可访问：https://learn.microsoft.com/en-us/power-automate/desktop-flows/create-flow-using-ai-recorder

[133] selenium. (2024) Selenium: Browser automation. Accessed: 2024-11-05. [Online]. Available: https://www.selenium.dev/
[133] selenium. (2024) Selenium：浏览器自动化。访问时间：2024-11-05。[在线]. 可访问：https://www.selenium.dev/

[134] appium. (2024) Appium: Cross-platform automation framework for all kinds of apps. Accessed: 2024-11-05. [Online]. Available: https://appium.io/docs/en/latest/
[134] appium. (2024) Appium：跨平台自动化框架，适用于各种应用。访问时间：2024-11-05。[在线]. 可访问：https://appium.io/docs/en/latest/

[135] smartbear. (2024) Testcomplete: Automated ui testing tool. Accessed: 2024-11-05. [Online]. Available: https://smartbear.com/ product/testcomplete/
[135] smartbear. (2024) Testcomplete：自动化UI测试工具。访问时间：2024-11-05。[在线]. 可访问：https://smartbear.com/product/testcomplete/

[136] katalon. (2024) Katalon studio: Easy test automation for web, api, mobile, and desktop. Accessed: 2024-11-05. [Online]. Available: https://katalon.com/katalon-studio
[136] katalon. (2024) Katalon Studio：简易的网页、API、移动和桌面测试自动化工具。访问时间：2024-11-05。[在线]. 可访问：https://katalon.com/katalon-studio

[137] ranorex. (2024) Ranorex studio: Test automation for gui testing. Accessed: 2024-11-05. [Online]. Available: https: //www.ranorex.com/
[137] ranorex. (2024) Ranorex Studio：GUI测试自动化工具。访问时间：2024-11-05。[在线]. 可访问：https://www.ranorex.com/

[138] G. Hu, L. Zhu, and J. Yang, "Appflow: using machine learning to synthesize robust, reusable ui tests," in Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ser. ESEC/FSE 2018. New York, NY, USA: Association for Computing Machinery, 2018, p. 269-282. [Online]. Available: https://doi.org/10.1145/3236024.3236055
[138] G. Hu, L. Zhu, 和 J. Yang, “Appflow：利用机器学习合成稳健且可复用的UI测试,” 载于2018年第26届ACM欧洲软件工程会议与软件工程基础研讨会联合会议论文集（ESEC/FSE 2018），纽约，美国：计算机协会，2018年，第269-282页。[在线]. 可访问：https://doi.org/10.1145/3236024.3236055

[139] Y. Li, Z. Yang, Y. Guo, and X. Chen, "Humanoid: A deep learning-based approach to automated black-box android app testing," in 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, 2019, pp. 1070-1073.
[139] Y. Li, Z. Yang, Y. Guo, 和 X. Chen, “Humanoid：基于深度学习的安卓应用黑盒自动测试方法,” 载于2019年第34届IEEE/ACM自动化软件工程国际会议（ASE），IEEE, 2019年，第1070-1073页。

[140] F. YazdaniBanafsheDaragh and S. Malek, "Deep gui: Black-box gui input generation with deep learning," in 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, 2021, pp. 905-916.
[140] F. YazdaniBanafsheDaragh 和 S. Malek，“Deep gui：基于深度学习的黑盒GUI输入生成”，发表于2021年第36届IEEE/ACM自动化软件工程国际会议（ASE），IEEE，2021年，第905-916页。

[141] M. Xie, S. Feng, Z. Xing, J. Chen, and C. Chen, "Uied: a hybrid tool for gui element detection," in Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2020, pp. 1655-1659.
[141] M. Xie, S. Feng, Z. Xing, J. Chen 和 C. Chen，“Uied：一种混合工具用于GUI元素检测”，发表于第28届ACM欧洲软件工程会议与软件工程基础研讨会联合会议论文集，2020年，第1655-1659页。

[142] N. Xu, S. Masling, M. Du, G. Campagna, L. Heck, J. Landay, and M. S. Lam, "Grounding open-domain instructions to automate web support tasks," 2021. [Online]. Available: https://arxiv.org/abs/2103.16057
[142] N. Xu, S. Masling, M. Du, G. Campagna, L. Heck, J. Landay 和 M. S. Lam，“将开放域指令落地以自动化网页支持任务”，2021年。[在线]。可访问：https://arxiv.org/abs/2103.16057

[143] S. Mazumder and O. Riva, "Flin: A flexible natural language interface for web navigation," arXiv preprint arXiv:2010.12844, 2020.
[143] S. Mazumder 和 O. Riva，“Flin：一种灵活的网页导航自然语言接口”，arXiv预印本 arXiv:2010.12844，2020年。

[144] Y. Li, J. He, X. Zhou, Y. Zhang, and J. Baldridge, "Mapping natural language instructions to mobile ui action sequences," in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 8198-8210.
[144] Y. Li, J. He, X. Zhou, Y. Zhang 和 J. Baldridge，“将自然语言指令映射到移动UI操作序列”，发表于第58届计算语言学协会年会论文集，2020年，第8198-8210页。

[145] T. Shi, A. Karpathy, L. Fan, J. Hernandez, and P. Liang, "World of bits: An open-domain platform for web-based agents," in Proceedings of the 34th International Conference on Machine Learning, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 06-11 Aug 2017, pp. 3135-3144. [Online]. Available: https://proceedings.mlr.press/v70/shi17a.html
[145] T. Shi, A. Karpathy, L. Fan, J. Hernandez 和 P. Liang，“World of bits：一个面向网页代理的开放域平台”，发表于第34届国际机器学习会议，机器学习研究进展系列，D. Precup 和 Y. W. Teh 编，卷70，PMLR，2017年8月6-11日，第3135-3144页。[在线]。可访问：https://proceedings.mlr.press/v70/shi17a.html

[146] E. Z. Liu, K. Guu, P. Pasupat, T. Shi, and P. Liang, "Reinforcement learning on web interfaces using workflow-guided exploration," 2018. [Online]. Available: https://arxiv.org/abs/1802.08802
[146] 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”，2018年。[在线]。可访问：https://arxiv.org/abs/1802.08802

[147] Y. Lan, Y. Lu, Z. Li, M. Pan, W. Yang, T. Zhang, and X. Li, "Deeply reinforcing android gui testing with deep reinforcement learning," in Proceedings of the 46th IEEE/ACM International Conference on Software Engineering, 2024, pp. 1-13.
[147] Y. Lan, Y. Lu, Z. Li, M. Pan, W. Yang, T. Zhang 和 X. Li，“利用深度强化学习深入强化安卓GUI测试”，发表于第46届IEEE/ACM国际软件工程会议，2024年，第1-13页。

[148] D. Toyama, P. Hamel, A. Gergely, G. Comanici, A. Glaese, Z. Ahmed, T. Jackson, S. Mourad, and D. Precup, "Androidenv: A reinforcement learning platform for android," arXiv preprint arXiv:2105.13231, 2021.
[148] D. Toyama, P. Hamel, A. Gergely, G. Comanici, A. Glaese, Z. Ahmed, T. Jackson, S. Mourad 和 D. Precup，“Androidenv：一个面向安卓的强化学习平台”，arXiv预印本 arXiv:2105.13231，2021年。

[149] S. Yao, H. Chen, J. Yang, and K. Narasimhan, "Webshop: Towards scalable real-world web interaction with grounded language agents," Advances in Neural Information Processing Systems, vol. 35, pp. 20744-20757, 2022.
[149] S. Yao, H. Chen, J. Yang 和 K. Narasimhan，“Webshop：面向可扩展真实网页交互的基于语言的代理”，《神经信息处理系统进展》，第35卷，第20744-20757页，2022年。

[150] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," IEEE Communications surveys & tutorials, vol. 21, no. 3, pp. 2224-2287, 2019.
[150] C. Zhang, P. Patras 和 H. Haddadi，“移动和无线网络中的深度学习综述”，《IEEE通信综述与教程》，第21卷，第3期，第2224-2287页，2019年。

[151] P. Martins, F. Sá, F. Morgado, and C. Cunha, "Using machine learning for cognitive robotic process automation (rpa)," in 2020 15th Iberian Conference on Information Systems and Technologies (CISTI). IEEE, 2020, pp. 1-6.
[151] P. Martins, F. Sá, F. Morgado 和 C. Cunha，“利用机器学习实现认知机器人流程自动化（RPA）”，发表于2020年第15届伊比利亚信息系统与技术会议（CISTI），IEEE，2020年，第1-6页。

[152] I. Gur, H. Furuta, A. Huang, M. Safdari, Y. Matsuo, D. Eck, and A. Faust, "A real-world webagent with planning, long context understanding, and program synthesis," arXiv preprint arXiv:2307.12856, 2023.
[152] I. Gur, H. Furuta, A. Huang, M. Safdari, Y. Matsuo, D. Eck 和 A. Faust，“具备规划、长上下文理解及程序合成能力的真实网页代理”，arXiv预印本 arXiv:2307.12856，2023年。

[153] H. Furuta, K.-H. Lee, O. Nachum, Y. Matsuo, A. Faust, S. S. Gu, and I. Gur, "Multimodal web navigation with instruction-finetuned foundation models," arXiv preprint arXiv:2305.11854, 2023.
[153] H. Furuta, K.-H. Lee, O. Nachum, Y. Matsuo, A. Faust, S. S. Gu 和 I. Gur，“基于指令微调基础模型的多模态网页导航”，arXiv预印本 arXiv:2305.11854，2023年。

[154] K. Ma, H. Zhang, H. Wang, X. Pan, W. Yu, and D. Yu, "Laser: LIm agent with state-space exploration for web navigation," arXiv preprint arXiv:2309.08172, 2023.
[154] K. Ma, H. Zhang, H. Wang, X. Pan, W. Yu, 和 D. Yu, “Laser：用于网页导航的状态空间探索大语言模型代理（LIm agent），” arXiv预印本 arXiv:2309.08172, 2023.

[155] 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.
[155] Y. Deng, X. Zhang, W. Zhang, Y. Yuan, S.-K. Ng, 和 T.-S. Chua, “关于对话式网页代理的多轮指令跟随,” arXiv预印本 arXiv:2402.15057, 2024.

[156] H. Wen, Y. Li, G. Liu, S. Zhao, T. Yu, T. J.-J. Li, S. Jiang, Y. Liu, Y. Zhang, and Y. Liu, "Autodroid: LIm-powered task automation in android," in Proceedings of the 30th Annual International Conference on Mobile Computing and Networking, 2024, pp. 543- 557.
[156] H. Wen, Y. Li, G. Liu, S. Zhao, T. Yu, T. J.-J. Li, S. Jiang, Y. Liu, Y. Zhang, 和 Y. Liu, “Autodroid：基于大语言模型（LIm）的安卓任务自动化,” 载于第30届国际移动计算与网络会议论文集, 2024, 页543-557.

[157] A. Yan, Z. Yang, W. Zhu, K. Lin, L. Li, J. Wang, J. Yang, Y. Zhong, J. McAuley, J. Gao et al., "Gpt-4v in wonderland: Large multimodal models for zero-shot smartphone gui navigation," arXiv preprint arXiv:2311.07562, 2023.
[157] A. Yan, Z. Yang, W. Zhu, K. Lin, L. Li, J. Wang, J. Yang, Y. Zhong, J. McAuley, J. Gao 等, “奇境中的GPT-4V：用于零样本智能手机图形用户界面导航的大型多模态模型,” arXiv预印本 arXiv:2311.07562, 2023.

[158] 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," 2024. [Online]. Available: https://arxiv.org/abs/2401.16158
[158] J. Wang, H. Xu, J. Ye, M. Yan, W. Shen, J. Zhang, F. Huang, 和 J. Sang, “Mobile-agent：具备视觉感知的自主多模态移动设备代理,” 2024. [在线]. 可获取：https://arxiv.org/abs/2401.16158

[159] S. Nong, J. Zhu, R. Wu, J. Jin, S. Shan, X. Huang, and W. Xu, "Mobileflow: A multimodal Ilm for mobile gui agent," 2024. [Online]. Available: https://arxiv.org/abs/2407.04346
[159] S. Nong, J. Zhu, R. Wu, J. Jin, S. Shan, X. Huang, 和 W. Xu, “Mobileflow：用于移动图形用户界面代理的多模态大语言模型,” 2024. [在线]. 可获取：https://arxiv.org/abs/2407.04346

[160] 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 preprint arXiv:2403.02713, 2024.
[160] J. Zhang, J. Wu, Y. Teng, M. Liao, N. Xu, X. Xiao, Z. Wei, 和 D. Tang, “Android动物园：图形用户界面代理的动作链思维,” arXiv预印本 arXiv:2403.02713, 2024.

[161] 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," 2024. [Online]. Available: https://arxiv.org/abs/2403.03186
[161] 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. [在线]. 可获取：https://arxiv.org/abs/2403.03186

[162] 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," 2024. [Online]. Available: https://arxiv.org/abs/2402.07456
[162] 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. [在线]. 可获取：https://arxiv.org/abs/2402.07456

[163] Anthropic. (2024) Introducing computer use, a new claude 3.5 sonnet, and claude 3.5 haiku. Accessed: 2024- 10-26. [Online]. Available: https://www.anthropic.com/news/ 3-5-models-and-computer-use
[163] Anthropic. (2024) 介绍computer use、新的Claude 3.5十四行诗和Claude 3.5俳句。访问时间：2024-10-26. [在线]. 可获取：https://www.anthropic.com/news/3-5-models-and-computer-use

[164] S. Hu, M. Ouyang, D. Gao, and M. Z. Shou, "The dawn of gui agent: A preliminary case study with claude 3.5 computer use," 2024. [Online]. Available: https://arxiv.org/abs/2411.10323
[164] S. Hu, M. Ouyang, D. Gao, 和 M. Z. Shou, “图形用户界面代理的曙光：基于Claude 3.5 computer use的初步案例研究,” 2024. [在线]. 可获取：https://arxiv.org/abs/2411.10323

[165] OpenAI, "Computer-using agent: Introducing a universal interface for ai to interact with the digital world," 2025. [Online]. Available: https://openai.com/index/computer-using-agent
[165] OpenAI, “计算机使用代理：引入AI与数字世界交互的通用接口,” 2025. [在线]. 可获取：https://openai.com/index/computer-using-agent

[166] C. Zhang, S. He, L. Li, S. Qin, Y. Kang, Q. Lin, and D. Zhang, "Api agents vs. gui agents: Divergence and convergence," arXiv preprint arXiv:2503.11069, 2025.
[166] C. Zhang, S. He, L. Li, S. Qin, Y. Kang, Q. Lin, 和 D. Zhang, “API代理与图形用户界面代理：分歧与融合,” arXiv预印本 arXiv:2503.11069, 2025.

[167] A. M. Memon, I. Banerjee, and A. Nagarajan, "Gui ripping: reverse engineering of graphical user interfaces for testing." in WCRE, vol. 3, 2003, p. 260.
[167] A. M. Memon, I. Banerjee, 和 A. Nagarajan, “GUI剥离：用于测试的图形用户界面逆向工程。”发表于WCRE，卷3，2003年，第260页。

[168] J. Wang, Z. Liu, L. Zhao, Z. Wu, C. Ma, S. Yu, H. Dai, Q. Yang, Y. Liu, S. Zhang et al., "Review of large vision models and visual prompt engineering," Meta-Radiology, p. 100047, 2023.
[168] J. Wang, Z. Liu, L. Zhao, Z. Wu, C. Ma, S. Yu, H. Dai, Q. Yang, Y. Liu, S. Zhang 等，“大型视觉模型与视觉提示工程综述，”Meta-Radiology，2023年，第100047页。

[169] S. Mitra and T. Acharya, "Gesture recognition: A survey," IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 37, no. 3, pp. 311-324, 2007.
[169] S. Mitra 和 T. Acharya，“手势识别：综述，”IEEE系统、人类与控制论学报C辑（应用与评论），第37卷，第3期，2007年，第311-324页。

[170] R. Hardy and E. Rukzio, "Touch & interact: touch-based interaction of mobile phones with displays," in Proceedings of the 10th international conference on Human computer interaction with mobile devices and services, 2008, pp. 245-254.
[170] R. Hardy 和 E. Rukzio，“触摸与交互：基于触摸的手机与显示屏交互，”发表于第10届国际移动设备与服务人机交互会议论文集，2008年，第245-254页。

[171] H. Lee, J. Park, and U. Lee, "A systematic survey on android api usage for data-driven analytics with smartphones," ACM Computing Surveys, vol. 55, no. 5, pp. 1-38, 2022.
[171] H. Lee, J. Park 和 U. Lee，“基于数据驱动分析的安卓API使用系统综述，”ACM计算机综述，第55卷，第5期，2022年，第1-38页。

[172] K. Jokinen, "User interaction in mobile navigation applications," in Map-based Mobile Services: Design, Interaction and Usability. Springer, 2008, pp. 168-197.
[172] K. Jokinen，“移动导航应用中的用户交互，”收录于《基于地图的移动服务：设计、交互与可用性》，施普林格，2008年，第168-197页。

[173] W. Enck, D. Octeau, P. D. McDaniel, and S. Chaudhuri, "A study of android application security." in USENIX security symposium, vol. 2, no. 2, 2011.
[173] W. Enck, D. Octeau, P. D. McDaniel 和 S. Chaudhuri，“安卓应用安全研究。”发表于USENIX安全研讨会，卷2，第2期，2011年。

[174] M. Egele, C. Kruegel, E. Kirda, and G. Vigna, "Pios: Detecting privacy leaks in ios applications." in NDSS, vol. 2011, 2011, p. 18th.
[174] M. Egele, C. Kruegel, E. Kirda 和 G. Vigna，“PIOS：检测iOS应用中的隐私泄露。”发表于NDSS，2011年，第18届。

[175] B. Sierkowski, "Achieving web accessibility," in Proceedings of the 30th annual ACM SIGUCCS conference on User services, 2002, pp. 288-291.
[175] B. Sierkowski，“实现网页无障碍，”发表于第30届ACM SIGUCCS用户服务年会论文集，2002年，第288-291页。

[176] N. Fernandes, R. Lopes, and L. Carriço, "On web accessibility evaluation environments," in Proceedings of the International Cross-Disciplinary Conference on Web Accessibility, 2011, pp. 1-10.
[176] N. Fernandes, R. Lopes 和 L. Carriço，“关于网页无障碍评估环境，”发表于国际跨学科网页无障碍会议论文集，2011年，第1-10页。

[177] J. J. Garrett et al., "Ajax: A new approach to web applications," 2005.
[177] J. J. Garrett 等，“Ajax：一种新的网页应用方法，”2005年。

[178] J. Yang, H. Zhang, F. Li, X. Zou, C. Li, and J. Gao, "Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v," arXiv preprint arXiv:2310.11441, 2023.
[178] J. Yang, H. Zhang, F. Li, X. Zou, C. Li 和 J. Gao，“标记集提示释放GPT-4V中卓越的视觉定位能力，”arXiv预印本 arXiv:2310.11441，2023年。

[179] X. Wu, J. Ye, K. Chen, X. Xie, Y. Hu, R. Huang, L. Ma, and J. Zhao, "Widget detection-based testing for industrial mobile games," in 2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP). IEEE, 2023, pp. 173-184.
[179] X. Wu, J. Ye, K. Chen, X. Xie, Y. Hu, R. Huang, L. Ma 和 J. Zhao，“基于控件检测的工业移动游戏测试，”发表于2023年 IEEE/ACM第45届国际软件工程大会：软件工程实践（ICSE-SEIP），IEEE，2023年，第173-184页。

[180] E. Gamma, "Design patterns: elements of reusable object-oriented software," Person Education Inc, 1995.
[180] E. Gamma，“设计模式：可复用面向对象软件的元素，”Person Education Inc，1995年。

[181] F. Wang, Z. Zhang, X. Zhang, Z. Wu, T. Mo, Q. Lu, W. Wang, R. Li, J. Xu, X. Tang, Q. He, Y. Ma, M. Huang, and S. Wang, "A comprehensive survey of small language models in the era of large language models: Techniques, enhancements, applications, collaboration with llms, and trustworthiness," 2024. [Online]. Available: https://arxiv.org/abs/2411.03350
[181] F. Wang, Z. Zhang, X. Zhang, Z. Wu, T. Mo, Q. Lu, W. Wang, R. Li, J. Xu, X. Tang, Q. He, Y. Ma, M. Huang 和 S. Wang，“大语

言模型时代小型语言模型的综合综述：技术、增强、应用、与大语言模型的协作及可信性，”2024年。[在线]。可访问：https://arxiv.org/abs/2411.03350

[182] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo et al., "Segment anything," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 4015-4026.
[182] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo 等，“Segment anything，”发表于2023年IEEE/CVF国际计算机视觉会议论文集，第4015-4026页。

[183] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su et al., "Grounding dino: Marrying dino with grounded pre-training for open-set object detection," arXiv preprint arXiv:2303.05499, 2023.
[183] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su 等，“Grounding dino：将dino与基于定位的预训练结合用于开放集目标检测”，arXiv预印本 arXiv:2303.05499，2023年。

[184] Y. Lu, J. Yang, Y. Shen, and A. Awadallah, "Omniparser for pure vision based gui agent," arXiv preprint arXiv:2408.00203, 2024.
[184] Y. Lu, J. Yang, Y. Shen, 和 A. Awadallah，“基于纯视觉的GUI代理的Omniparser”，arXiv预印本 arXiv:2408.00203，2024年。

[185] K. Moran, C. Watson, J. Hoskins, G. Purnell, and D. Poshyvanyk, "Detecting and summarizing gui changes in evolving mobile apps," in Proceedings of the 33rd ACM/IEEE international conference on automated software engineering, 2018, pp. 543-553.
[185] K. Moran, C. Watson, J. Hoskins, G. Purnell, 和 D. Poshyvanyk，“检测和总结不断演进的移动应用中的GUI变化”，发表于第33届ACM/IEEE自动化软件工程国际会议论文集，2018年，第543-553页。

[186] F. P. Ricós, R. Neeft, B. Marín, T. E. Vos, and P. Aho, "Using gui change detection for delta testing," in International Conference on Research Challenges in Information Science. Springer, 2023, pp. 509-517.
[186] F. P. Ricós, R. Neeft, B. Marín, T. E. Vos, 和 P. Aho，“利用GUI变化检测进行增量测试”，发表于信息科学研究挑战国际会议，Springer出版社，2023年，第509-517页。

[187] Y. Du, F. Wei, and H. Zhang, "Anytool: Self-reflective, hierarchical agents for large-scale api calls," arXiv preprint arXiv:2402.04253, 2024.
[187] Y. Du, F. Wei, 和 H. Zhang，“Anytool：用于大规模API调用的自我反思分层代理”，arXiv预印本 arXiv:2402.04253，2024年。

[188] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel et al., "Retrieval-augmented generation for knowledge-intensive nlp tasks," Advances in Neural Information Processing Systems, vol. 33, pp. 9459-9474, 2020.
[188] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel 等，“面向知识密集型自然语言处理任务的检索增强生成”，《神经信息处理系统进展》(NeurIPS)，第33卷，第9459-9474页，2020年。

[189] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, and H. Wang, "Retrieval-augmented generation for large language models: A survey," arXiv preprint arXiv:2312.10997, 2023.
[189] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, 和 H. Wang，“面向大型语言模型的检索增强生成综述”，arXiv预印本 arXiv:2312.10997，2023年。

[190] S. Zhang, Z. Zhang, K. Chen, X. Ma, M. Yang, T. Zhao, and M. Zhang, "Dynamic planning for llm-based graphical user interface automation," arXiv preprint arXiv:2410.00467, 2024.
[190] S. Zhang, Z. Zhang, K. Chen, X. Ma, M. Yang, T. Zhao, 和 M. Zhang，“基于LLM的图形用户界面自动化动态规划”，arXiv预印本 arXiv:2410.00467，2024年。

[191] J. Cho, J. Kim, D. Bae, J. Choo, Y. Gwon, and Y.-D. Kwon, "Caap: Context-aware action planning prompting to solve computer tasks with front-end ui only," arXiv preprint arXiv:2406.06947, 2024.
[191] J. Cho, J. Kim, D. Bae, J. Choo, Y. Gwon, 和 Y.-D. Kwon，“CAAP：仅使用前端UI解决计算机任务的上下文感知动作规划提示”，arXiv预印本 arXiv:2406.06947，2024年。

[192] G. Dagan, F. Keller, and A. Lascarides, "Dynamic planning with a llm," arXiv preprint arXiv:2308.06391, 2023.
[192] G. Dagan, F. Keller, 和 A. Lascarides，“基于LLM的动态规划”，arXiv预印本 arXiv:2308.06391，2023年。

[193] T. Khot, H. Trivedi, M. Finlayson, Y. Fu, K. Richardson, P. Clark, and A. Sabharwal, "Decomposed prompting: A modular approach for solving complex tasks," arXiv preprint arXiv:2210.02406, 2022.
[193] T. Khot, H. Trivedi, M. Finlayson, Y. Fu, K. Richardson, P. Clark, 和 A. Sabharwal，“分解提示：解决复杂任务的模块化方法”，arXiv预印本 arXiv:2210.02406，2022年。

[194] Y. Chen, A. Pesaranghader, T. Sadhu, and D. H. Yi, "Can we rely on llm agents to draft long-horizon plans? let's take travelplanner as an example," arXiv preprint arXiv:2408.06318, 2024.
[194] Y. Chen, A. Pesaranghader, T. Sadhu, 和 D. H. Yi，“我们能否依赖LLM代理制定长远计划？以TravelPlanner为例”，arXiv预印本 arXiv:2408.06318，2024年。

[195] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, "Zero-shot text-to-image generation," in International conference on machine learning. Pmlr, 2021, pp. 8821-8831.
[195] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, 和 I. Sutskever，“零样本文本到图像生成”，发表于国际机器学习会议，PMLR出版社，2021年，第8821-8831页。

[196] X. Gu, H. Zhang, D. Zhang, and S. Kim, "Deep api learning," in Proceedings of the 2016 24th ACM SIGSOFT international symposium on foundations of software engineering, 2016, pp. 631-642.
[196] X. Gu, H. Zhang, D. Zhang, 和 S. Kim，“深度API学习”，发表于2016年第24届ACM SIGSOFT软件工程基础国际研讨会论文集，2016年，第631-642页。

[197] T. Masterman, S. Besen, M. Sawtell, and A. Chao, "The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey," arXiv preprint arXiv:2404.11584, 2024.
[197] T. Masterman, S. Besen, M. Sawtell, 和 A. Chao，“新兴AI代理架构在推理、规划和工具调用中的全景图：一项综述，” arXiv预印本 arXiv:2404.11584, 2024。

[198] J. Lu, Z. Zhang, F. Yang, J. Zhang, L. Wang, C. Du, Q. Lin, S. Rajmohan, D. Zhang, and Q. Zhang, "Turn every application into an agent: Towards efficient human-agent-computer interaction with api-first Ilm-based agents," arXiv preprint arXiv:2409.17140, 2024.
[198] J. Lu, Z. Zhang, F. Yang, J. Zhang, L. Wang, C. Du, Q. Lin, S. Rajmohan, D. Zhang, 和 Q. Zhang，“将每个应用转变为代理：面向高效人-代理-计算机交互的基于API优先的Ilm代理，” arXiv预印本 arXiv:2409.17140, 2024。

[199] Y. Song, F. Xu, S. Zhou, and G. Neubig, "Beyond browsing: Api-based web agents," arXiv preprint arXiv:2410.16464, 2024.
[199] Y. Song, F. Xu, S. Zhou, 和 G. Neubig，“超越浏览：基于API的网页代理，” arXiv预印本 arXiv:2410.16464, 2024。

[200] S. Lee, J. Choi, J. Lee, M. H. Wasi, H. Choi, S. Y. Ko, S. Oh, and I. Shin, "Explore, select, derive, and recall: Augmenting IIm with human-like memory for mobile task automation," arXiv preprint arXiv:2312.03003, 2023.
[200] S. Lee, J. Choi, J. Lee, M. H. Wasi, H. Choi, S. Y. Ko, S. Oh, 和 I. Shin，“探索、选择、推导与回忆：通过类人记忆增强IIm以实现移动任务自动化，” arXiv预印本 arXiv:2312.03003, 2023。

[201] J. Lu, S. An, M. Lin, G. Pergola, Y. He, D. Yin, X. Sun, and Y. Wu, "Memochat: Tuning Ilms to use memos for consistent long-range open-domain conversation," arXiv preprint arXiv:2308.08239, 2023.
[201] J. Lu, S. An, M. Lin, G. Pergola, Y. He, D. Yin, X. Sun, 和 Y. Wu，“Memochat：调优Ilms以使用备忘录实现一致的长距离开放域对话，” arXiv预印本 arXiv:2308.08239, 2023。

[202] W. Wang, L. Dong, H. Cheng, X. Liu, X. Yan, J. Gao, and F. Wei, "Augmenting language models with long-term memory," Advances in Neural Information Processing Systems, vol. 36, 2024.
[202] W. Wang, L. Dong, H. Cheng, X. Liu, X. Yan, J. Gao, 和 F. Wei，“通过长期记忆增强语言模型，” 神经信息处理系统进展，卷36，2024。

[203] J. Tack, J. Kim, E. Mitchell, J. Shin, Y. W. Teh, and J. R. Schwarz, "Online adaptation of language models with a memory of amortized contexts," arXiv preprint arXiv:2403.04317, 2024.
[203] J. Tack, J. Kim, E. Mitchell, J. Shin, Y. W. Teh, 和 J. R. Schwarz，“带有摊销上下文记忆的语言模型在线适应，” arXiv预印本 arXiv:2403.04317, 2024。

[204] X. Zhu, Y. Chen, H. Tian, C. Tao, W. Su, C. Yang, G. Huang, B. Li, L. Lu, X. Wang et al., "Ghost in the minecraft: Generally capable agents for open-world environments via large language models with text-based knowledge and memory," arXiv preprint arXiv:2305.17144, 2023.
[204] X. Zhu, Y. Chen, H. Tian, C. Tao, W. Su, C. Yang, G. Huang, B. Li, L. Lu, X. Wang 等，“我的世界中的幽灵：通过基于文本知识和记忆的大型语言模型实现开放世界环境中的通用能力代理，” arXiv预印本 arXiv:2305.17144, 2023。

[205] 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
[205] L. Zheng, R. Wang, X. Wang, 和 B. An，“Synapse：基于记忆的轨迹示例提示用于计算机控制，” 2024。[在线]. 可获取：https://arxiv.org/abs/2306.07863

[206] J. Li, D. Li, S. Savarese, and S. Hoi, "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," in International conference on machine learning. PMLR, 2023, pp. 19730-19742.
[206] J. Li, D. Li, S. Savarese, 和 S. Hoi，“Blip-2：利用冻结的图像编码器和大型语言模型引导语言-图像预训练，” 载于国际机器学习会议。PMLR, 2023, 页19730-19742。

[207] D. Reis, J. Kupec, J. Hong, and A. Daoudi, "Real-time flying object detection with yolov8," arXiv preprint arXiv:2305.09972, 2023.
[207] D. Reis, J. Kupec, J. Hong, 和 A. Daoudi，“基于yolov8的实时飞行物体检测，” arXiv预印本 arXiv:2305.09972, 2023。

[208] A. Nguyen, "Improved gui grounding via iterative narrowing," 2024. [Online]. Available: https://arxiv.org/abs/2411.13591
[208] A. Nguyen，“通过迭代缩小改进GUI定位，” 2024。[在线]. 可获取：https://arxiv.org/abs/2411.13591

[209] Z. Ge, J. Li, X. Pang, M. Gao, K. Pan, W. Lin, H. Fei, W. Zhang, S. Tang, and Y. Zhuang, "Iris: Breaking gui complexity with adaptive focus and self-refining," 2024. [Online]. Available: https://arxiv.org/abs/2412.10342
[209] Z. Ge, J. Li, X. Pang, M. Gao, K. Pan, W. Lin, H. Fei, W. Zhang, S. Tang, 和 Y. Zhuang, “Iris：通过自适应聚焦和自我优化破解GUI复杂性，” 2024。[在线]. 可获取：https://arxiv.org/abs/2412.10342

[210] J. Bai, S. Bai, S. Yang, S. Wang, S. Tan, P. Wang, J. Lin, C. Zhou, and J. Zhou, "Qwen-vl: A frontier large vision-language model with versatile abilities," arXiv preprint arXiv:2308.12966, 2023.
[210] J. Bai, S. Bai, S. Yang, S. Wang, S. Tan, P. Wang, J. Lin, C. Zhou, 和 J. Zhou, “Qwen-vl：具备多功能能力的前沿大型视觉-语言模型，” arXiv预印本 arXiv:2308.12966, 2023。

[211] H.-M. Xu, Q. Chen, L. Wang, and L. Liu, "Attention-driven gui grounding: Leveraging pretrained multimodal large language models without fine-tuning," 2024. [Online]. Available: https://arxiv.org/abs/2412.10840
[211] H.-M. Xu, Q. Chen, L. Wang, 和 L. Liu, “基于注意力驱动的GUI定位：利用预训练多模态大型语言模型无需微调，” 2024。[在线]. 可获取：https://arxiv.org/abs/2412.10840

[212] 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, pp. 28091-28114, 2023.
[212] X. Deng, Y. Gu, B. Zheng, S. Chen, S. Stevens, B. Wang, H. Sun, 和 Y. Su, “Mind2web：迈向通用网页智能体,” 神经信息处理系统进展(Advances in Neural Information Processing Systems), 第36卷, 页28091-28114, 2023年。

[213] 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?" 2024. [Online]. Available: https://arxiv.org/abs/2404.05955
[213] J. Liu, Y. Song, B. Y. Lin, W. Lam, G. Neubig, Y. Li, 和 X. Yue, “Visualwebbench：多模态大语言模型在网页理解与定位方面的发展现状,” 2024年。[在线]. 可获取：https://arxiv.org/abs/2404.05955

[214] 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
[214] Y. Yang, Y. Wang, D. Li, Z. Luo, B. Chen, C. Huang, 和 J. Li, “Aria-ui：图形用户界面(GUI)指令的视觉定位,” 2024年。[在线]. 可获取：https://arxiv.org/abs/2412.16256

[215] D. Li, Y. Liu, H. Wu, Y. Wang, Z. Shen, B. Qu, X. Niu, G. Wang, B. Chen, and J. Li, "Aria: An open multimodal native mixture-of-experts model," arXiv preprint arXiv:2410.05993, 2024.
[215] D. Li, Y. Liu, H. Wu, Y. Wang, Z. Shen, B. Qu, X. Niu, G. Wang, B. Chen, 和 J. Li, “Aria：一个开放的多模态原生专家混合模型,” arXiv预印本 arXiv:2410.05993, 2024年。

[216] B. Gou, R. Wang, B. Zheng, Y. Xie, C. Chang, Y. Shu, H. Sun, and Y. Su, "Navigating the digital world as humans do: Universal visual grounding for gui agents," 2024. [Online]. Available: https://arxiv.org/abs/2410.05243
[216] B. Gou, R. Wang, B. Zheng, Y. Xie, C. Chang, Y. Shu, H. Sun, 和 Y. Su, “像人类一样导航数字世界：面向GUI智能体的通用视觉定位,” 2024年。[在线]. 可获取：https://arxiv.org/abs/2410.05243

[217] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," Advances in neural information processing systems, vol. 36, 2024.
[217] 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(Advances in Neural Information Processing Systems), 第36卷, 2024年。

[218] Y. Fan, H. Zhao, R. Zhang, Y. Shen, X. E. Wang, and G. Wu, "Gui-bee: Align gui action grounding to novel environments via autonomous exploration," 2025. [Online]. Available: https://arxiv.org/abs/2501.13896
[218] Y. Fan, H. Zhao, R. Zhang, Y. Shen, X. E. Wang, 和 G. Wu, “Gui-bee：通过自主探索将GUI动作定位对齐到新环境,” 2025年。[在线]. 可获取：https://arxiv.org/abs/2501.13896

[219] W. Chen, J. Cui, J. Hu, Y. Qin, J. Fang, Y. Zhao, C. Wang, J. Liu, G. Chen, Y. Huo, Y. Yao, Y. Lin, Z. Liu, and M. Sun, "Guicourse: From general vision language models to versatile gui agents," 2024. [Online]. Available: https://arxiv.org/abs/2406.11317
[219] W. Chen, J. Cui, J. Hu, Y. Qin, J. Fang, Y. Zhao, C. Wang, J. Liu, G. Chen, Y. Huo, Y. Yao, Y. Lin, Z. Liu, 和 M. Sun, “Guicourse：从通用视觉语言模型到多功能GUI智能体,” 2024年。[在线]. 可获取：https://arxiv.org/abs/2406.11317

[220] J. Liu, T. Ou, Y. Song, Y. Qu, W. Lam, C. Xiong, W. Chen, G. Neubig, and X. Yue, "Harnessing webpage uis for text-rich visual understanding," arXiv preprint arXiv:2410.13824, 2024.
[220] J. Liu, T. Ou, Y. Song, Y. Qu, W. Lam, C. Xiong, W. Chen, G. Neubig, 和 X. Yue, “利用网页用户界面进行文本丰富的视觉理解,” arXiv预印本 arXiv:2410.13824, 2024年。

[221] J. Yang and H. Hou, "Rwkv-ui: Ui understanding with enhanced perception and reasoning," arXiv preprint arXiv:2502.03971, 2025.
[221] J. Yang 和 H. Hou, “Rwkv-ui：增强感知与推理的用户界面理解,” arXiv预印本 arXiv:2502.03971, 2025年。

[222] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer, "Sigmoid loss for language image pre-training," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 11 975- 11986.
[222] X. Zhai, B. Mustafa, A. Kolesnikov, 和 L. Beyer, “用于语言图像预训练的Sigmoid损失,” 载于IEEE/CVF国际计算机视觉会议论文集, 2023, 页码 11975-11986.

[223] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby et al., "Dinov2: Learning robust visual features without supervision," arXiv preprint arXiv:2304.07193, 2023.
[223] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby 等, “Dinov2：无监督学习鲁棒视觉特征,” arXiv预印本 arXiv:2304.07193, 2023.

[224] H. Laurengon, L. Tronchon, and V. Sanh, "Unlocking the conversion of web screenshots into html code with the websight dataset," arXiv preprint arXiv:2403.09029, 2024.
[224] H. Laurengon, L. Tronchon, 和 V. Sanh, “利用Websight数据集解锁网页截图转HTML代码的转换,” arXiv预印本 arXiv:2403.09029, 2024.

[225] J. Wu, S. Wang, S. Shen, Y.-H. Peng, J. Nichols, and J. P. Bigham, "Webui: A dataset for enhancing visual ui understanding with web semantics," in Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, 2023, pp. 1-14.
[225] J. Wu, S. Wang, S. Shen, Y.-H. Peng, J. Nichols, 和 J. P. Bigham, “Webui：用于增强视觉用户界面理解与网页语义的数据集,” 载于2023年CHI人机交互大会论文集, 2023, 页码 1-14.

[226] S. Yun, H. Lin, R. Thushara, M. Q. Bhat, Y. Wang, Z. Jiang, M. Deng, J. Wang, T. Tao, J. Li et al., "Web2code: A large-scale webpage-to-code dataset and evaluation framework for multimodal llms," arXiv preprint arXiv:2406.20098, 2024.
[226] S. Yun, H. Lin, R. Thushara, M. Q. Bhat, Y. Wang, Z. Jiang, M. Deng, J. Wang, T. Tao, J. Li 等, “Web2code：大规模网页到代码数据集及多模态ILMs评估框架,” arXiv预印本 arXiv:2406.20098, 2024.

[227] K. Singh, S. Singh, and M. Khanna, "Trishul: Towards region identification and screen hierarchy understanding for large vlm based gui agents," 2025. [Online]. Available: https://arxiv.org/abs/2502.08226
[227] K. Singh, S. Singh, 和 M. Khanna, “Trishul：面向基于大型视觉语言模型的GUI代理的区域识别与屏幕层级理解,” 2025年. [在线]. 可获取：https://arxiv.org/abs/2502.08226

[228] H. Li, J. Chen, J. Su, Y. Chen, Q. Li, and Z. Zhang, "Autogui: Scaling gui grounding with automatic functionality annotations from llms," arXiv preprint arXiv:2502.01977, 2025.
[228] H. Li, J. Chen, J. Su, Y. Chen, Q. Li, 和 Z. Zhang, “Autogui：通过ILMs自动功能注释扩展GUI定位,” arXiv预印本 arXiv:2502.01977, 2025.

[229] Y.-F. Zhang, Q. Wen, C. Fu, X. Wang, Z. Zhang, L. Wang, and R. Jin, "Beyond llava-hd: Diving into high-resolution large multimodal models," arXiv preprint arXiv:2406.08487, 2024.
[229] Y.-F. Zhang, Q. Wen, C. Fu, X. Wang, Z. Zhang, L. Wang, 和 R. Jin, “超越llava-hd：深入高分辨率大型多模态模型,” arXiv预印本 arXiv:2406.08487, 2024.

[230] Z. Wu, P. Cheng, Z. Wu, T. Ju, Z. Zhang, and G. Liu, "Smoothing grounding and reasoning for mllm-powered gui agents with query-oriented pivot tasks," arXiv preprint arXiv:2503.00401, 2025.
[230] Z. Wu, P. Cheng, Z. Wu, T. Ju, Z. Zhang, 和 G. Liu, “针对多模态大型语言模型驱动的GUI代理的平滑定位与推理，基于查询导向的枢纽任务,” arXiv预印本 arXiv:2503.00401, 2025.

[231] 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
[231] 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, 和 J. Lin, “Qwen2-vl：提升视觉语言模型对任意分辨率世界的感知能力,” 2024年. [在线]. 可获取：https://arxiv.org/abs/2409.12191

[232] Z. Wu, Z. Wu, F. Xu, Y. Wang, Q. Sun, C. Jia, K. Cheng, Z. Ding, L. Chen, P. P. Liang et al., "Os-atlas: A foundation action model for generalist gui agents," arXiv preprint arXiv:2410.23218, 2024.
[232] Z. Wu, Z. Wu, F. Xu, Y. Wang, Q. Sun, C. Jia, K. Cheng, Z. Ding, L. Chen, P. P. Liang 等, “Os-atlas：面向通用GUI代理的基础动作模型,” arXiv预印本 arXiv:2410.23218, 2024.

[233] Z. Hui, Y. Li, D. zhao, T. Chen, C. Banbury, and K. Koishida, "Winclick: Gui grounding with multimodal large language models," 2025. [Online]. Available: https://arxiv.org/abs/2503.04730
[233] Z. Hui, Y. Li, D. Zhao, T. Chen, C. Banbury, 和 K. Koishida, “Winclick：基于多模态大型语言模型的GUI定位,” 2025年. [在线]. 可获取：https://arxiv.org/abs/2503.04730

[234] M. Abdin, J. Aneja, H. Awadalla, A. Awadallah, A. A. Awan, N. Bach, A. Bahree, A. Bakhtiari, J. Bao, H. Behl et al., "Phi-3 technical report: A highly capable language model locally on your phone," arXiv preprint arXiv:2404.14219, 2024.
[234] M. Abdin, J. Aneja, H. Awadalla, A. Awadallah, A. A. Awan, N. Bach, A. Bahree, A. Bakhtiari, J. Bao, H. Behl 等, “Phi-3技术报告：一款可在手机本地运行的高性能语言模型,” arXiv预印本 arXiv:2404.14219, 2024.

[235] 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," arXiv preprint arXiv:2503.06470, 2025.
[235] F. Tang, Y. Shen, H. Zhang, S. Chen, G. Hou, W. Zhang, W. Zhang, K. Song, W. Lu, 和 Y. Zhuang, “三思而后行：通过快慢系统增强GUI定位，”arXiv预印本 arXiv:2503.06470, 2025。

[236] 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
[236] Y. Xu, Z. Wang, J. Wang, D. Lu, T. Xie, A. Saha, D. Sahoo, T. Yu, 和 C. Xiong, “Aguvis：用于自主GUI交互的统一纯视觉代理，”2024。[在线]. 可获取：https://arxiv.org/abs/2412.04454

[237] L. Zheng, Z. Huang, Z. Xue, X. Wang, B. An, and S. Yan, "Agentstudio: A toolkit for building general virtual agents," 2024. [Online]. Available: https://arxiv.org/abs/2403.17918
[237] 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，”2024。[在线]. 可获取：https://arxiv.org/abs/2403.17918

[238] 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
[238] K. Q. Lin, L. Li, D. Gao, Z. Yang, S. Wu, Z. Bai, W. Lei, L. Wang, 和 M. Z. Shou, “Showui：用于GUI视觉代理的单一视觉-语言-动作模型，”2024。[在线]. 可获取：https://arxiv.org/abs/2411.17465

[239] X. Liu, X. Zhang, Z. Zhang, and Y. Lu, "Ui-e2i-synth: Advancing gui grounding with large-scale instruction synthesis," arXiv preprint arXiv:2504.11257, 2025.
[239] X. Liu, X. Zhang, Z. Zhang, 和 Y. Lu, “Ui-e2i-synth：通过大规模指令合成推进GUI定位，”arXiv预印本 arXiv:2504.11257, 2025。

[240] T. Luo, L. Logeswaran, J. Johnson, and H. Lee, "Visual test-time scaling for gui agent grounding," 2025. [Online]. Available: https://arxiv.org/abs/2505.00684
[240] T. Luo, L. Logeswaran, J. Johnson, 和 H. Lee, “用于GUI代理定位的视觉测试时缩放，”2025。[在线]. 可获取：https://arxiv.org/abs/2505.00684

[241] 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.
[241] 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。

[242] Q. Yang, W. Bi, H. Shen, Y. Guo, and Y. Ma, "Pixelweb: The first web gui dataset with pixel-wise labels," 2025. [Online]. Available: https://arxiv.org/abs/2504.16419
[242] Q. Yang, W. Bi, H. Shen, Y. Guo, 和 Y. Ma, “Pixelweb：首个带像素级标签的网页GUI数据集，”2025。[在线]. 可获取：https://arxiv.org/abs/2504.16419

[243] X. Zhan, T. Liu, L. Fan, L. Li, S. Chen, X. Luo, and Y. Liu, "Research on third-party libraries in android apps: A taxonomy and systematic literature review," IEEE Transactions on Software Engineering, vol. 48, no. 10, pp. 4181-4213, 2021.
[243] X. Zhan, T. Liu, L. Fan, L. Li, S. Chen, X. Luo, 和 Y. Liu, “安卓应用中第三方库的研究：分类法与系统文献综述，”IEEE软件工程汇刊, 第48卷，第10期，页4181-4213, 2021。

[244] Y. Li, G. Li, L. He, J. Zheng, H. Li, and Z. Guan, "Widget captioning: Generating natural language description for mobile user interface elements," arXiv preprint arXiv:2010.04295, 2020.
[244] Y. Li, G. Li, L. He, J. Zheng, H. Li, 和 Z. Guan, “控件描述：为移动用户界面元素生成自然语言描述，”arXiv预印本 arXiv:2010.04295, 2020。

[245] B. Wang, G. Li, X. Zhou, Z. Chen, T. Grossman, and Y. Li, "Screen2words: Automatic mobile ui summarization with multimodal learning," in The 34th Annual ACM Symposium on User Interface Software and Technology, 2021, pp. 498-510.
[245] B. Wang, G. Li, X. Zhou, Z. Chen, T. Grossman, 和 Y. Li, “Screen2words：基于多模态学习的自动移动UI摘要，”发表于第34届ACM用户界面软件与技术年会, 2021, 页498-510。

[246] C. Bai, X. Zang, Y. Xu, S. Sunkara, A. Rastogi, J. Chen et al., "Uibert: Learning generic multimodal representations for ui understanding," arXiv preprint arXiv:2107.13731, 2021.
[246] C. Bai, X. Zang, Y. Xu, S. Sunkara, A. Rastogi, J. Chen 等, “Uibert：学习通用多模态表示以理解UI，”arXiv预印本 arXiv:2107.13731, 2021。

[247] G. Li, H. A. A. K. Hammoud, H. Itani, D. Khizbullin, and B. Ghanem, "Camel: Communicative agents for "mind" exploration of large language model society," in Thirty-seventh Conference on Neural Information Processing Systems, 2023.
[247] G. Li, H. A. A. K. Hammoud, H. Itani, D. Khizbullin, 和 B. Ghanem, “Camel：用于大型语言模型社会“思维”探索的交流代理，”发表于第三十七届神经信息处理系统会议, 2023。

[248] W. Chen, Z. You, R. Li, Y. Guan, C. Qian, C. Zhao, C. Yang, R. Xie, Z. Liu, and M. Sun, "Internet of agents: Weaving a web of heterogeneous agents for collaborative intelligence," 2024. [Online]. Available: https://arxiv.org/abs/2407.07061
[248] W. Chen, Z. You, R. Li, Y. Guan, C. Qian, C. Zhao, C. Yang, R. Xie, Z. Liu, 和 M. Sun, “代理互联网：构建异构代理协同智能的网络，”2024。[在线]. 可获取：https://arxiv.org/abs/2407.07061

[249] Z. Song, Y. Li, M. Fang, Z. Chen, Z. Shi, Y. Huang, and L. Chen, "Mmac-copilot: Multi-modal agent collaboration operating system copilot," arXiv preprint arXiv:2404.18074, 2024.
[249] 宋志强, 李勇, 方明, 陈志, 史志, 黄勇, 陈磊, "Mmac-copilot: 多模态代理协作操作系统助手," arXiv预印本 arXiv:2404.18074, 2024.

[250] M. Renze and E. Guven, "Self-reflection in Ilm agents: Effects on problem-solving performance," arXiv preprint arXiv:2405.06682, 2024.
[250] M. Renze 和 E. Guven, "Ilm代理中的自我反思：对问题解决性能的影响," arXiv预印本 arXiv:2405.06682, 2024.

[251] J. Pan, Y. Zhang, N. Tomlin, Y. Zhou, S. Levine, and A. Suhr, "Autonomous evaluation and refinement of digital agents," in First Conference on Language Modeling, 2024.
[251] 潘杰, 张勇, N. Tomlin, 周颖, S. Levine, 和 A. Suhr, "数字代理的自主评估与优化," 语言建模首届会议, 2024.

[252] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," arXiv preprint arXiv:2210.03629, 2022.
[252] 姚爽, 赵军, 于丹, 杜楠, I. Shafran, K. Narasimhan, 和 曹阳, "React: 语言模型中推理与行动的协同," arXiv预印本 arXiv:2210.03629, 2022.

[253] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning," Advances in Neural Information Processing Systems, vol. 36, 2024.
[253] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, 和 S. Yao, "Reflexion: 具备语言强化学习的语言代理," 神经信息处理系统进展, 第36卷, 2024.

[254] Z. Tao, T.-E. Lin, X. Chen, H. Li, Y. Wu, Y. Li, Z. Jin, F. Huang, D. Tao, and J. Zhou, "A survey on self-evolution of large language models," arXiv preprint arXiv:2404.14387, 2024.
[254] 陶志, 林天恩, 陈晓, 李浩, 吴洋, 李勇, 金志, 黄峰, 陶东, 和 周军, "大型语言模型自我进化综述," arXiv预印本 arXiv:2404.14387, 2024.

[255] A. Zhao, D. Huang, Q. Xu, M. Lin, Y.-J. Liu, and G. Huang, "Expel: LIm agents are experiential learners," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, no. 17, 2024, pp. 19632-19642.
[255] 赵安, 黄丹, 徐强, 林明, 刘一杰, 和 黄刚, "Expel: LIm代理作为经验学习者," AAAI人工智能会议论文集, 第38卷第17期, 2024, 页19632-19642.

[256] Z. Zhu, Y. Xue, X. Chen, D. Zhou, J. Tang, D. Schuurmans, and H. Dai, "Large language models can learn rules," arXiv preprint arXiv:2310.07064, 2023.
[256] 朱志, 薛阳, 陈晓, 周东, 唐杰, D. Schuurmans, 和 戴浩, "大型语言模型能够学习规则," arXiv预印本 arXiv:2310.07064, 2023.

[257] Y. Zhang, P. Xiao, L. Wang, C. Zhang, M. Fang, Y. Du, Y. Puzyrev, R. Yao, S. Qin, Q. Lin, M. Pechenizkiy, D. Zhang, S. Rajmohan, and Q. Zhang, "Ruag: Learned-rule-augmented generation for large language models," 2024. [Online]. Available: https://arxiv.org/abs/2411.03349
[257] 张勇, 肖鹏, 王磊, 张超, 方明, 杜勇, Puzyrev Y., 姚锐, 秦松, 林强, Pechenizkiy M., 张东, Rajmohan S., 和 张强, "Ruag: 用于大型语言模型的学习规则增强生成," 2024. [在线]. 可访问: https://arxiv.org/abs/2411.03349

[258] W. Jiang, Y. Zhuang, C. Song, X. Yang, and C. Zhang, "Appagentx: Evolving gui agents as proficient smartphone users," arXiv preprint arXiv:2503.02268, 2025.
[258] 蒋伟, 庄勇, 宋超, 杨翔, 和 张超, "Appagentx: 进化的图形界面代理作为熟练的智能手机用户," arXiv预印本 arXiv:2503.02268, 2025.

[259] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," Journal of artificial intelligence research, vol. 4, pp. 237-285, 1996.
[259] L. P. Kaelbling, M. L. Littman, 和 A. W. Moore, "强化学习综述," 人工智能研究杂志, 第4卷, 页237-285, 1996.

[260] Y. Wang, W. Zhong, L. Li, F. Mi, X. Zeng, W. Huang, L. Shang, X. Jiang, and Q. Liu, "Aligning large language models with human: A survey," arXiv preprint arXiv:2307.12966, 2023.
[260] 王勇, 钟伟, 李磊, 米飞, 曾翔, 黄伟, 商磊, 姜晓, 和 刘强, "大型语言模型与人类的对齐综述," arXiv预印本 arXiv:2307.12966, 2023.

[261] Y. Zhai, H. Bai, Z. Lin, J. Pan, S. Tong, Y. Zhou, A. Suhr, S. Xie, Y. LeCun, Y. Ma et al., "Fine-tuning large vision-language models as decision-making agents via reinforcement learning," arXiv preprint arXiv:2405.10292, 2024.
[261] 翟阳, 白浩, 林志, 潘杰, 童松, 周颖, A. Suhr, 谢松, Y. LeCun, 马勇 等, "通过强化学习微调大型视觉语言模型作为决策代理," arXiv预印本 arXiv:2405.10292, 2024.

[262] M. L. Puterman, "Markov decision processes," Handbooks in operations research and management science, vol. 2, pp. 331-434, 1990.
[262] M. L. Puterman, "马尔可夫决策过程," 运筹学与管理科学手册, 第2卷, 页331-434, 1990.

[263] D. Toyama, P. Hamel, A. Gergely, G. Comanici, A. Glaese, Z. Ahmed, T. Jackson, S. Mourad, and D. Precup, "Androidenv: A reinforcement learning platform for android," 2021. [Online]. Available: https://arxiv.org/abs/2105.13231
[263] D. Toyama, P. Hamel, A. Gergely, G. Comanici, A. Glaese, Z. Ahmed, T. Jackson, S. Mourad, 和 D. Precup, "Androidenv: 一个面向安卓的强化学习平台," 2021. [在线]. 可访问: https://arxiv.org/abs/2105.13231

[264] H. Bai, Y. Zhou, M. Cemri, J. Pan, A. Suhr, S. Levine, and A. Kumar, "Digirl: Training in-the-wild device-control agents with autonomous reinforcement learning," 2024. [Online]. Available: https://arxiv.org/abs/2406.11896
[264] H. Bai, Y. Zhou, M. Cemri, J. Pan, A. Suhr, S. Levine, 和 A. Kumar，“Digirl：利用自主强化学习训练野外设备控制代理”，2024年。[在线]. 可获取：https://arxiv.org/abs/2406.11896

[265] T. Wang, Z. Wu, J. Liu, J. Hao, J. Wang, and K. Shao, "Distrl: An asynchronous distributed reinforcement learning framework for on-device control agents," arXiv preprint arXiv:2410.14803, 2024.
[265] T. Wang, Z. Wu, J. Liu, J. Hao, J. Wang, 和 K. Shao，“Distrl：一种用于设备端控制代理的异步分布式强化学习框架”，arXiv预印本 arXiv:2410.14803，2024年。

[266] H. Chae, N. Kim, K. T. iunn Ong, M. Gwak, G. Song, J. Kim, S. Kim, D. Lee, and J. Yeo, "Web agents with world models: Learning and leveraging environment dynamics in web navigation," 2024. [Online]. Available: https://arxiv.org/abs/2410.13232
[266] H. Chae, N. Kim, K. T. iunn Ong, M. Gwak, G. Song, J. Kim, S. Kim, D. Lee, 和 J. Yeo，“具备世界模型的网页代理：学习并利用环境动态进行网页导航”，2024年。[在线]. 可获取：https://arxiv.org/abs/2410.13232

[267] I. Gur, H. Furuta, A. Huang, M. Safdari, Y. Matsuo, D. Eck, and A. Faust, "A real-world webagent with planning, long context understanding, and program synthesis," 2024. [Online]. Available: https://arxiv.org/abs/2307.12856
[267] I. Gur, H. Furuta, A. Huang, M. Safdari, Y. Matsuo, D. Eck, 和 A. Faust，“具备规划、长上下文理解及程序合成能力的真实世界网页代理”，2024年。[在线]. 可获取：https://arxiv.org/abs/2307.12856

[268] K. Ma, H. Zhang, H. Wang, X. Pan, W. Yu, and D. Yu, "Laser: LIm agent with state-space exploration for web navigation," 2024. [Online]. Available: https://arxiv.org/abs/2309.08172
[268] K. Ma, H. Zhang, H. Wang, X. Pan, W. Yu, 和 D. Yu，“Laser：基于状态空间探索的网页导航LIm代理”，2024年。[在线]. 可获取：https://arxiv.org/abs/2309.08172

[269] 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," 2024. [Online]. Available: https://arxiv.org/abs/2401.13919
[269] H. He, W. Yao, K. Ma, W. Yu, Y. Dai, H. Zhang, Z. Lan, 和 D. Yu，“Webvoyager：构建基于大型多模态模型的端到端网页代理”，2024年。[在线]. 可获取：https://arxiv.org/abs/2401.13919

[270] H. Lai, X. Liu, I. L. long, S. Yao, Y. Chen, P. Shen, H. Yu, H. Zhang, X. Zhang, Y. Dong, and J. Tang, "Autowebglm: Bootstrap and reinforce a large language model-based web navigating agent," 2024. [Online]. Available: https://arxiv.org/abs/2404.03648
[270] H. Lai, X. Liu, I. L. long, 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的自举与强化”，2024年。[在线]. 可获取：https://arxiv.org/abs/2404.03648

[271] T. Xie, F. Zhou, Z. Cheng, P. Shi, L. Weng, Y. Liu, T. J. Hua, J. Zhao, Q. Liu, C. Liu, L. Z. Liu, Y. Xu, H. Su, D. Shin, C. Xiong, and T. Yu, "Openagents: An open platform for language agents in the wild," 2023. [Online]. Available: https://arxiv.org/abs/2310.10634
[271] T. Xie, F. Zhou, Z. Cheng, P. Shi, L. Weng, Y. Liu, T. J. Hua, J. Zhao, Q. Liu, C. Liu, L. Z. Liu, Y. Xu, H. Su, D. Shin, C. Xiong, 和 T. Yu，“Openagents：面向真实环境语言代理的开放平台”，2023年。[在线]. 可获取：https://arxiv.org/abs/2310.10634

[272] J. Kil, C. H. Song, B. Zheng, X. Deng, Y. Su, and W.-L. Chao, "Dual-view visual contextualization for web navigation," 2024. [Online]. Available: https://arxiv.org/abs/2402.04476
[272] J. Kil, C. H. Song, B. Zheng, X. Deng, Y. Su, 和 W.-L. Chao，“用于网页导航的双视角视觉语境化”，2024年。[在线]. 可获取：https://arxiv.org/abs/2402.04476

[273] T. Abuelsaad, D. Akkil, P. Dey, A. Jagmohan, A. Vempaty, and R. Kokku, "Agent-e: From autonomous web navigation to foundational design principles in agentic systems," 2024. [Online]. Available: https://arxiv.org/abs/2407.13032
[273] T. Abuelsaad, D. Akkil, P. Dey, A. Jagmohan, A. Vempaty, 和 R. Kokku，“Agent-e：从自主网页导航到代理系统的基础设计原则”，2024年。[在线]. 可获取：https://arxiv.org/abs/2407.13032

[274] J. Y. Koh, S. McAleer, D. Fried, and R. Salakhutdinov, "Tree search for language model agents," arXiv preprint arXiv:2407.01476, 2024.
[274] J. Y. Koh, S. McAleer, D. Fried, 和 R. Salakhutdinov，“语言模型代理的树搜索”，arXiv预印本 arXiv:2407.01476，2024年。

[275] Y. Zhang, Z. Ma, Y. Ma, Z. Han, Y. Wu, and V. Tresp, "Webpilot: A versatile and autonomous multi-agent system for web task execution with strategic exploration," arXiv preprint arXiv:2408.15978, 2024.
[275] Y. Zhang, Z. Ma, Y. Ma, Z. Han, Y. Wu, 和 V. Tresp，“Webpilot：用于网页任务执行的多功能自主多代理系统，具备策略性探索能力”，arXiv预印本 arXiv:2408.15978，2024年。

[276] K. Yang, Y. Liu, S. Chaudhary, R. Fakoor, P. Chaudhari, G. Karypis, and H. Rangwala, "Agentoccam: A simple yet strong baseline for Ilm-based web agents," 2024. [Online]. Available: https://arxiv.org/abs/2410.13825
[276] K. Yang, Y. Liu, S. Chaudhary, R. Fakoor, P. Chaudhari, G. Karypis, 和 H. Rangwala，“Agentoccam：基于Ilm的网页代理的简单而强大的基线”，2024年。[在线]. 可获取：https://arxiv.org/abs/2410.13825

[277] S. Murty, D. Bahdanau, and C. D. Manning, "Nnetscape navigator: Complex demonstrations for web agents without a demonstrator," arXiv preprint arXiv:2410.02907, 2024.
[277] S. Murty, D. Bahdanau, 和 C. D. Manning，“Nnetscape navigator：无需示范者的复杂网页代理演示，” arXiv预印本 arXiv:2410.02907, 2024.

[278] M. Shahbandeh, P. Alian, N. Nashid, and A. Mesbah, "Navigate: Functionality-guided web application navigation," arXiv preprint arXiv:2409.10741, 2024.
[278] M. Shahbandeh, P. Alian, N. Nashid, 和 A. Mesbah，“Navigate：基于功能引导的网页应用导航，” arXiv预印本 arXiv:2409.10741, 2024.

[279] I. L. long, X. Liu, Y. Chen, H. Lai, S. Yao, P. Shen, H. Yu, Y. Dong, and J. Tang, "Openwebagent: An open toolkit to enable web agents on large language models," in Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), 2024, pp. 72-81.
[279] I. L. long, X. Liu, Y. Chen, H. Lai, S. Yao, P. Shen, H. Yu, Y. Dong, 和 J. Tang，“Openwebagent：支持大型语言模型（large language models）网页代理的开放工具包，” 载于第62届计算语言学协会年会论文集（第3卷：系统演示），2024，页72-81.

[280] B. Tang and K. G. Shin, "Steward: Natural language web automation," arXiv preprint arXiv:2409.15441, 2024.
[280] B. Tang 和 K. G. Shin，“Steward：自然语言网页自动化，” arXiv预印本 arXiv:2409.15441, 2024.

[281] P. Putta, E. Mills, N. Garg, S. Motwani, C. Finn, D. Garg, and R. Rafailov, "Agent q: Advanced reasoning and learning for autonomous ai agents," arXiv preprint arXiv:2408.07199, 2024.
[281] P. Putta, E. Mills, N. Garg, S. Motwani, C. Finn, D. Garg, 和 R. Rafailov，“Agent q：自主人工智能代理的高级推理与学习，” arXiv预印本 arXiv:2408.07199, 2024.

[282] Y. Gu, B. Zheng, B. Gou, K. Zhang, C. Chang, S. Srivastava, Y. Xie, P. Qi, H. Sun, and Y. Su, "Is your Ilm secretly a world model of the internet? model-based planning for web agents," arXiv preprint arXiv:2411.06559, 2024.
[282] Y. Gu, B. Zheng, B. Gou, K. Zhang, C. Chang, S. Srivastava, Y. Xie, P. Qi, H. Sun, 和 Y. Su，“你的因果语言模型（ILM）是否暗藏互联网的世界模型？基于模型的网页代理规划，” arXiv预印本 arXiv:2411.06559, 2024.

[283] G. Verma, R. Kaur, N. Srishankar, Z. Zeng, T. Balch, and M. Veloso, "Adaptagent: Adapting multimodal web agents with few-shot learning from human demonstrations," arXiv preprint arXiv:2411.13451, 2024.
[283] G. Verma, R. Kaur, N. Srishankar, Z. Zeng, T. Balch, 和 M. Veloso，“Adaptagent：通过少量人类示范的少样本学习适应多模态网页代理，” arXiv预印本 arXiv:2411.13451, 2024.

[284] J. Kim, D.-K. Kim, L. Logeswaran, S. Sohn, and H. Lee, "Auto-intent: Automated intent discovery and self-exploration for large language model web agents," arXiv preprint arXiv:2410.22552, 2024.
[284] J. Kim, D.-K. Kim, L. Logeswaran, S. Sohn, 和 H. Lee，“Auto-intent：大型语言模型网页代理的自动意图发现与自我探索，” arXiv预印本 arXiv:2410.22552, 2024.

[285] J. Shen, A. Jain, Z. Xiao, I. Amlekar, M. Hadji, A. Podolny, and A. Talwalkar, "Scribeagent: Towards specialized web agents using production-scale workflow data," 2024. [Online]. Available: https://arxiv.org/abs/2411.15004
[285] J. Shen, A. Jain, Z. Xiao, I. Amlekar, M. Hadji, A. Podolny, 和 A. Talwalkar，“Scribeagent：基于生产级工作流数据的专业化网页代理探索，” 2024. [在线]. 可获取：https://arxiv.org/abs/2411.15004

[286] Y. Zhou, Q. Yang, K. Lin, M. Bai, X. Zhou, Y.-X. Wang, S. Levine, and E. Li, "Proposer-agent-evaluator (pae): Autonomous skill discovery for foundation model internet agents," 2024. [Online]. Available: https://arxiv.org/abs/2412.13194
[286] Y. Zhou, Q. Yang, K. Lin, M. Bai, X. Zhou, Y.-X. Wang, S. Levine, 和 E. Li，“Proposer-agent-evaluator (PAE)：基础模型互联网代理的自主技能发现，” 2024. [在线]. 可获取：https://arxiv.org/abs/2412.13194

[287] J. Liu, J. Hao, C. Zhang, and Z. Hu, "Wepo: Web element preference optimization for llm-based web navigation," 2024. [Online]. Available: https://arxiv.org/abs/2412.10742
[287] J. Liu, J. Hao, C. Zhang, 和 Z. Hu, “Wepo：基于因果语言模型（ILM）的网页导航元素偏好优化,” 2024. [在线]. 可获取：https://arxiv.org/abs/2412.10742

[288] T. Huang, K. Basu, I. Abdelaziz, P. Kapanipathi, J. May, and M. Chen, "R2d2: Remembering, reflecting and dynamic decision making for web agents," arXiv preprint arXiv:2501.12485, 2025.
[288] T. Huang, K. Basu, I. Abdelaziz, P. Kapanipathi, J. May, 和 M. Chen, “R2D2：网页代理的记忆、反思与动态决策,” arXiv预印本 arXiv:2501.12485, 2025.

[289] R. Zhang, M. Qiu, Z. Tan, M. Zhang, V. Lu, J. Peng, K. Xu, L. Z. Agudelo, P. Qian, and T. Chen, "Symbiotic cooperation for web agents: Harnessing complementary strengths of large and small llms," arXiv preprint arXiv:2502.07942, 2025.
[289] R. Zhang, M. Qiu, Z. Tan, M. Zhang, V. Lu, J. Peng, K. Xu, L. Z. Agudelo, P. Qian, 和 T. Chen, “网页代理的共生合作：利用大型与小型因果语言模型（ILMs）的互补优势,” arXiv预印本 arXiv:2502.07942, 2025.

[290] V. Pahuja, Y. Lu, C. Rosset, B. Gou, A. Mitra, S. Whitehead, Y. Su, and A. Awadallah, "Explorer: Scaling exploration-driven web trajectory synthesis for multimodal web agents," arXiv preprint arXiv:2502.11357, 2025.
[290] V. Pahuja, Y. Lu, C. Rosset, B. Gou, A. Mitra, S. Whitehead, Y. Su, 和 A. Awadallah, “Explorer：面向多模态网页代理的探索驱动网页轨迹合成扩展,” arXiv预印本 arXiv:2502.11357, 2025.

[291] M. Wornow, A. Narayan, K. Opsahl-Ong, Q. McIntyre, N. Shah, and C. Re, "Automating the enterprise with foundation models," Proceedings of the VLDB Endowment, vol. 17, no. 11, pp. 2805- 2812, 2024.
[291] M. Wornow, A. Narayan, K. Opsahl-Ong, Q. McIntyre, N. Shah, 和 C. Re, “利用基础模型自动化企业,” VLDB Endowment 会议录, 第17卷, 第11期, 页2805-2812, 2024年。

[292] D. Zhang, B. Rama, J. Ni, S. He, F. Zhao, K. Chen, A. Chen, and J. Cao, "Litewebagent: The open-source suite for vlm-based web-agent applications," arXiv preprint arXiv:2503.02950, 2025.
[292] D. Zhang, B. Rama, J. Ni, S. He, F. Zhao, K. Chen, A. Chen, 和 J. Cao, “Litewebagent：基于视觉语言模型（VLM）的开源网页代理应用套件,” arXiv 预印本 arXiv:2503.02950, 2025年。

[293] P. P. S. Dammu, "Towards ethical and personalized web navigation agents: A framework for user-aligned task execution," in Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining, 2025, pp. 1074-1076.
[293] P. P. S. Dammu, “迈向伦理且个性化的网页导航代理：用户对齐任务执行框架,” 第十八届ACM国际网页搜索与数据挖掘会议论文集, 2025年, 页1074-1076。

[294] L. E. Erdogan, N. Lee, S. Kim, S. Moon, H. Furuta, G. Anu-manchipalli, K. Keutzer, and A. Gholami, "Plan-and-act: Improving planning of agents for long-horizon tasks," arXiv preprint arXiv:2503.09572, 2025.
[294] L. E. Erdogan, N. Lee, S. Kim, S. Moon, H. Furuta, G. Anumanchipalli, K. Keutzer, 和 A. Gholami, “计划与执行：提升代理在长时任务中的规划能力,” arXiv 预印本 arXiv:2503.09572, 2025年。

[295] B. Zheng, M. Y. Fatemi, X. Jin, Z. Z. Wang, A. Gandhi, Y. Song, Y. Gu, J. Srinivasa, G. Liu, G. Neubig et al., "Skillweaver: Web agents can self-improve by discovering and honing skills," arXiv preprint arXiv:2504.07079, 2025.
[295] B. Zheng, M. Y. Fatemi, X. Jin, Z. Z. Wang, A. Gandhi, Y. Song, Y. Gu, J. Srinivasa, G. Liu, G. Neubig 等, “Skillweaver：网页代理通过发现与磨炼技能实现自我提升,” arXiv 预印本 arXiv:2504.07079, 2025年。

[296] Z. Z. Wang, A. Gandhi, G. Neubig, and D. Fried, "Inducing programmatic skills for agentic tasks," arXiv preprint arXiv:2504.06821, 2025.
[296] Z. Z. Wang, A. Gandhi, G. Neubig, 和 D. Fried, “为代理任务诱导程序化技能,” arXiv 预印本 arXiv:2504.06821, 2025年。

[297] Z. Zhang, T. Fang, K. Ma, W. Yu, H. Zhang, H. Mi, and D. Yu, "Enhancing web agents with explicit rollback mechanisms," arXiv preprint arXiv:2504.11788, 2025.
[297] Z. Zhang, T. Fang, K. Ma, W. Yu, H. Zhang, H. Mi, 和 D. Yu, “通过显式回滚机制增强网页代理,” arXiv 预印本 arXiv:2504.11788, 2025年。

[298] 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," 2024. [Online]. Available: https://arxiv.org/abs/2403.02713
[298] J. Zhang, J. Wu, Y. Teng, M. Liao, N. Xu, X. Xiao, Z. Wei, 和 D. Tang, “Android 动物园：面向GUI代理的行动链思维,” 2024年。[在线]. 可获取：https://arxiv.org/abs/2403.02713

[299] Y. Song, Y. Bian, Y. Tang, G. Ma, and Z. Cai, "Visiontasker: Mobile task automation using vision based ui understanding and llm task planning," in Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology, ser. UIST '24. ACM, Oct. 2024, p. 1-17. [Online]. Available: http://dx.doi.org/10.1145/3654777.3676386
[299] Y. Song, Y. Bian, Y. Tang, G. Ma, 和 Z. Cai, “Visiontasker：基于视觉的用户界面理解与ILM任务规划的移动任务自动化,” 第37届

ACM用户界面软件与技术年会论文集，UIST '24系列，ACM，2024年10月，页1-17。[在线]. 可获取：http://dx.doi.org/10.1145/3654777.3676386

[300] H. Wen, H. Wang, J. Liu, and Y. Li, "Droidbot-gpt: Gpt-powered ui automation for android," 2024. [Online]. Available: https://arxiv.org/abs/2304.07061
[300] H. Wen, H. Wang, J. Liu, 和 Y. Li, “Droidbot-gpt：基于GPT的安卓UI自动化,” 2024年。[在线]. 可获取：https://arxiv.org/abs/2304.07061

[301] X. Ma, Z. Zhang, and H. Zhao, "Coco-agent: A comprehensive cognitive mllm agent for smartphone gui automation," 2024. [Online]. Available: https://arxiv.org/abs/2402.11941
[301] X. Ma, Z. Zhang, 和 H. Zhao, “Coco-agent：面向智能手机GUI自动化的综合认知多模态大语言模型（MLLM）代理,” 2024年。[在线]. 可获取：https://arxiv.org/abs/2402.11941

[302] Z. Zhang and A. Zhang, "You only look at screens: Multimodal chain-of-action agents," 2024. [Online]. Available: https://arxiv.org/abs/2309.11436
[302] Z. Zhang 和 A. Zhang, “你只需看屏幕：多模态行动链代理,” 2024年。[在线]. 可获取：https://arxiv.org/abs/2309.11436

[303] 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," 2023. [Online]. Available: https://arxiv.org/abs/2311.07562
[303] 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奇境：面向零样本智能手机GUI导航的大型多模态模型,” 2023年。[在线]. 可获取：https://arxiv.org/abs/2311.07562

[304] 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," 2024. [Online]. Available: https://arxiv.org/abs/2408.11824
[304] Y. Li, C. Zhang, W. Yang, B. Fu, P. Cheng, X. Chen, L. Chen, 和 Y. Wei, “Appagent v2：灵活移动交互的高级代理,” 2024年。[在线]. 可获取：https://arxiv.org/abs/2408.11824

[305] H. Wen, S. Tian, B. Pavlov, W. Du, Y. Li, G. Chang, S. Zhao, J. Liu, Y. Liu, Y.-Q. Zhang, and Y. Li, "Autodroid-v2: Boosting slm-based gui agents via code generation," 2024. [Online]. Available: https://arxiv.org/abs/2412.18116
[305] H. Wen, S. Tian, B. Pavlov, W. Du, Y. Li, G. Chang, S. Zhao, J. Liu, Y. Liu, Y.-Q. Zhang, 和 Y. Li, “Autodroid-v2：通过代码生成提升基于SLM的GUI代理性能，”2024年。[在线]. 可获取：https://arxiv.org/abs/2412.18116

[306] 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," 2024. [Online]. Available: https://arxiv.org/abs/2406.01014
[306] 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。[在线]. 可获取：https://arxiv.org/abs/2406.01014

[307] J. Zhang, C. Zhao, Y. Zhao, Z. Yu, M. He, and J. Fan, "Mobileexperts: A dynamic tool-enabled agent team in mobile devices," 2024. [Online]. Available: https://arxiv.org/abs/2407.03913
[307] J. Zhang, C. Zhao, Y. Zhao, Z. Yu, M. He, 和 J. Fan, “Mobileexperts：移动设备中动态工具支持的代理团队，”2024年。[在线]. 可获取：https://arxiv.org/abs/2407.03913

[308] F. Christianos, G. Papoudakis, T. Coste, J. Hao, J. Wang, and K. Shao, "Lightweight neural app control," 2024. [Online]. Available: https://arxiv.org/abs/2410.17883
[308] F. Christianos, G. Papoudakis, T. Coste, J. Hao, J. Wang, 和 K. Shao, “轻量级神经应用控制，”2024年。[在线]. 可获取：https://arxiv.org/abs/2410.17883

[309] Z. Zhu, H. Tang, Y. Li, K. Lan, Y. Jiang, H. Zhou, Y. Wang, S. Zhang, L. Sun, L. Chen et al., "Moba: A two-level agent system for efficient mobile task automation," arXiv preprint arXiv:2410.13757, 2024.
[309] Z. Zhu, H. Tang, Y. Li, K. Lan, Y. Jiang, H. Zhou, Y. Wang, S. Zhang, L. Sun, L. Chen 等, “Moba：一种高效移动任务自动化的两级代理系统，”arXiv预印本 arXiv:2410.13757, 2024年。

[310] S. Lee, J. Choi, J. Lee, M. H. Wasi, H. Choi, S. Ko, S. Oh, and I. Shin, "Mobilegpt: Augmenting llm with human-like app memory for mobile task automation," in Proceedings of the 30th Annual International Conference on Mobile Computing and Networking, 2024, pp. 1119-1133.
[310] S. Lee, J. Choi, J. Lee, M. H. Wasi, H. Choi, S. Ko, S. Oh, 和 I. Shin, “Mobilegpt：通过类人应用记忆增强大型语言模型以实现移动任务自动化，”发表于第30届国际移动计算与网络年会论文集，2024年，第1119-1133页。

[311] 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
[311] Z. Wang, H. Xu, J. Wang, X. Zhang, M. Yan, J. Zhang, F. Huang, 和 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

[312] J. Hoscilowicz, B. Maj, B. Kozakiewicz, O. Tymoshchuk, and A. Janicki, "Clickagent: Enhancing ui location capabilities of autonomous agents," arXiv preprint arXiv:2410.11872, 2024.
[312] J. Hoscilowicz, B. Maj, B. Kozakiewicz, O. Tymoshchuk, 和 A. Janicki, “Clickagent：增强自主代理的用户界面定位能力，”arXiv预印本 arXiv:2410.11872, 2024年。

[313] Q. Wu, W. Liu, J. Luan, and B. Wang, "Reachagent: Enhancing mobile agent via page reaching and operation," arXiv preprint arXiv:2502.02955, 2025.
[313] Q. Wu, W. Liu, J. Luan, 和 B. Wang, “Reachagent：通过页面到达与操作增强移动代理，”arXiv预印本 arXiv:2502.02955, 2025年。

[314] W. Wang, Z. Yu, W. Liu, R. Ye, T. Jin, S. Chen, and Y. Wang, "Fedmobileagent: Training mobile agents using decentralized self-sourced data from diverse users," arXiv preprint arXiv:2502.02982, 2025.
[314] W. Wang, Z. Yu, W. Liu, R. Ye, T. Jin, S. Chen, 和 Y. Wang, “Fedmobileagent：利用来自多样用户的去中心化自源数据训练移动代理，”arXiv预印本 arXiv:2502.02982, 2025年。

[315] T. Huang, C. Yu, W. Shi, Z. Peng, D. Yang, W. Sun, and Y. Shi, "Prompt2task: Automating ui tasks on smartphones from textual prompts," ACM Transactions on Computer-Human Interaction.
[315] T. Huang, C. Yu, W. Shi, Z. Peng, D. Yang, W. Sun, 和 Y. Shi, “Prompt2task：基于文本提示自动化智能手机上的用户界面任务，”ACM人机交互汇刊。

[316] J. Wang, H. Xu, X. Zhang, M. Yan, J. Zhang, F. Huang, and J. Sang, "Mobile-agent-v: Learning mobile device operation through video-guided multi-agent collaboration," arXiv preprint arXiv:2502.17110, 2025.
[316] J. Wang, H. Xu, X. Zhang, M. Yan, J. Zhang, F. Huang, 和 J. Sang, “Mobile-agent-v：通过视频引导的多代理协作学习移动设备操作，”arXiv预印本 arXiv:2502.17110, 2025年。

[317] Y. Liu, H. Sun, W. Liu, J. Luan, B. Du, and R. Yan, "Mobilesteward: Integrating multiple app-oriented agents with self-evolution to automate cross-app instructions," arXiv preprint arXiv:2502.16796, 2025.
[317] Y. Liu, H. Sun, W. Liu, J. Luan, B. Du, 和 R. Yan, “Mobilesteward：集成多应用导向代理并具备自我进化能力以实现跨应用指令自动化，”arXiv预印本 arXiv:2502.16796, 2025年。

[318] Y. Zhou, S. Wang, S. Dai, Q. Jia, Z. Du, Z. Dong, and J. Xu, "Chop: Mobile operating assistant with constrained high-frequency optimized subtask planning," arXiv preprint arXiv:2503.03743, 2025.
[318] 周毅, 王帅, 戴帅, 贾强, 杜志, 董志, 许杰, "Chop: 受限高频优化子任务规划的移动操作助手," arXiv预印本 arXiv:2503.03743, 2025.

[319] P. Cheng, Z. Wu, Z. Wu, A. Zhang, Z. Zhang, and G. Liu, "Os-kairos: Adaptive interaction for mllm-powered gui agents," arXiv preprint arXiv:2503.16465, 2025.
[319] 程鹏, 吴志, 吴志, 张安, 张志, 刘刚, "Os-kairos: 基于多模态大语言模型(mllm)的自适应交互图形用户界面代理," arXiv预印本 arXiv:2503.16465, 2025.

[320] G. Dai, S. Jiang, T. Cao, Y. Li, Y. Yang, R. Tan, M. Li, and L. Qiu, "Advancing mobile gui agents: A verifier-driven approach to practical deployment," arXiv preprint arXiv:2503.15937, 2025.
[320] 戴刚, 蒋帅, 曹涛, 李阳, 杨洋, 谭锐, 李明, 邱磊, "推进移动图形用户界面代理：一种基于验证器驱动的实用部署方法," arXiv预印本 arXiv:2503.15937, 2025.

[321] G. Liu, P. Zhao, L. Liu, Z. Chen, Y. Chai, S. Ren, H. Wang, S. He, and W. Meng, "Learnact: Few-shot mobile gui agent with a unified demonstration benchmark," 2025. [Online]. Available: https://arxiv.org/abs/2504.13805
[321] 刘刚, 赵鹏, 刘磊, 陈志, 柴阳, 任帅, 王浩, 何帅, 孟伟, "Learnact: 具有统一示范基准的少样本移动图形用户界面代理," 2025. [在线]. 可获取：https://arxiv.org/abs/2504.13805

[322] H. Lai, J. Gao, X. Liu, Y. Xu, S. Zhang, Y. Dong, and J. Tang, "Androidgen: Building an android language agent under data scarcity," arXiv preprint arXiv:2504.19298, 2025.
[322] 赖浩, 高杰, 刘翔, 徐阳, 张帅, 董阳, 唐杰, "Androidgen: 在数据稀缺条件下构建安卓语言代理," arXiv预印本 arXiv:2504.19298, 2025.

[323] B. Wang, G. Li, and Y. Li, "Enabling conversational interaction with mobile ui using large language models," in Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, 2023, pp. 1-17.
[323] 王博, 李刚, 李阳, "利用大型语言模型实现移动用户界面的对话交互," 载于2023年CHI人机交互大会论文集, 2023, 页1-17.

[324] N. Kahlon, G. Rom, A. Efros, F. Galgani, O. Berkovitch, S. Caduri, W. E. Bishop, O. Riva, and I. Dagan, "Agent-initiated interaction in phone ui automation," arXiv preprint arXiv:2503.19537, 2025.
[324] 卡隆 · 纳赫隆, 罗姆 · 格雷格, 埃弗罗斯 · 阿里, 加尔加尼 · 弗朗西斯科, 贝尔科维奇 · 奥利, 卡杜里 · 萨姆, 主教 · 威廉 · E, 里瓦 · 奥利弗, 达甘 · 伊扎克, "电话用户界面自动化中的代理发起交互," arXiv预印本 arXiv:2503.19537, 2025.

[325] W. E. Bishop, A. Li, C. Rawles, and O. Riva, "Latent state estimation helps ui agents to reason," arXiv preprint arXiv:2405.11120, 2024.
[325] 主教 · 威廉 · E, 李安, 罗尔斯 · 克里斯, 里瓦 · 奥利弗, "潜在状态估计助力用户界面代理推理," arXiv预印本 arXiv:2405.11120, 2024.

[326] 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," 2024. [Online]. Available: https://arxiv.org/abs/2410.08164
[326] 阿加谢 · 萨姆, 韩杰, 甘帅, 杨杰, 李安, 王晓东, "Agent S: 一个开放的代理框架，像人类一样使用计算机," 2024. [在线]. 可获取：https://arxiv.org/abs/2410.08164

[327] Q. Wu, D. Gao, K. Q. Lin, Z. Wu, X. Guo, P. Li, W. Zhang, H. Wang, and M. Z. Shou, "Gui action narrator: Where and when did that action take place?" 2024. [Online]. Available: https://arxiv.org/abs/2406.13719
[327] 吴强, 高东, 林克强, 吴志, 郭翔, 李鹏, 张伟, 王浩, 寿明哲, "图形用户界面动作叙述者：那个动作发生的时间和地点？ " 2024. [在线]. 可获取：https://arxiv.org/abs/2406.13719

[328] T. Li, G. Li, Z. Deng, B. Wang, and Y. Li, "A zero-shot language agent for computer control with structured reflection," arXiv preprint arXiv:2310.08740, 2023.
[328] 李涛, 李刚, 邓志, 王博, 李阳, "一种基于结构化反思的零样本语言代理用于计算机控制," arXiv预印本 arXiv:2310.08740, 2023.

[329] 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
[329] 何阳, 金杰, 夏帅, 苏杰, 范锐, 邹浩, 胡翔, 刘鹏, "PC代理：当你沉睡时，人工智能在工作——数字世界的认知之旅," 2024. [在线]. 可获取：https://arxiv.org/abs/2412.17589

[330] H. Liu, X. Zhang, H. Xu, Y. Wanyan, J. Wang, M. Yan, J. Zhang, C. Yuan, C. Xu, W. Hu et al., "Pc-agent: A hierarchical multi-agent collaboration framework for complex task automation on pc," arXiv preprint arXiv:2502.14282, 2025.
[330] 刘浩, 张翔, 许浩, 万岩, 王杰, 颜明, 张杰, 袁超, 徐超, 胡伟 等, "PC-agent: 用于PC复杂任务自动化的分层多代理协作框架," arXiv预印本 arXiv:2502.14282, 2025.

[331] P. Aggarwal and S. Welleck, "Programming with pixels: Computer-use meets software engineering," arXiv preprint arXiv:2502.18525, 2025.
[331] 阿加瓦尔 · 普拉提克, 韦莱克 · 斯蒂芬, "像素编程：计算机使用遇上软件工程," arXiv预印本 arXiv:2502.18525, 2025.

[332] D. Zhao, L. Ma, S. Wang, M. Wang, and Z. Lv, "Cola: A scalable multi-agent framework for windows ui task automation," arXiv preprint arXiv:2503.09263, 2025.
[332] 赵东, 马磊, 王帅, 王明, 吕志, "Cola: 一个可扩展的多代理框架用于Windows用户界面任务自动化," arXiv预印本 arXiv:2503.09263, 2025.

[333] F. Lu, Z. Zhong, Z. Wei, S. Liu, C.-W. Fu, and J. Jia, "Steve: Astep verification pipeline for computer-use agent training," arXiv preprint arXiv:2503.12532, 2025.
[333] F. Lu, Z. Zhong, Z. Wei, S. Liu, C.-W. Fu, 和 J. Jia, "Steve：用于计算机使用代理训练的步骤验证流程," arXiv预印本 arXiv:2503.12532, 2025.

[334] C. Zhang, H. Huang, C. Ni, J. Mu, S. Qin, S. He, L. Wang, F. Yang, P. Zhao, C. Du et al., "Ufo2: The desktop agentos," arXiv preprint arXiv:2504.14603, 2025.
[334] C. Zhang, H. Huang, C. Ni, J. Mu, S. Qin, S. He, L. Wang, F. Yang, P. Zhao, C. Du 等, "Ufo2：桌面代理操作系统," arXiv预印本 arXiv:2504.14603, 2025.

[335] Y. Yin, Y. Mei, C. Yu, T. J.-J. Li, A. K. Jadoon, S. Cheng, W. Shi, M. Chen, and Y. Shi, "From operation to cognition: Automatic modeling cognitive dependencies from user demonstrations for gui task automation," in Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems, 2025, pp. 1-24.
[335] Y. Yin, Y. Mei, C. Yu, T. J.-J. Li, A. K. Jadoon, S. Cheng, W. Shi, M. Chen, 和 Y. Shi, "从操作到认知：基于用户演示自动建模GUI任务自动化中的认知依赖," 载于2025年CHI人机交互大会论文集, 2025, 页码1-24.

[336] X. Liu, B. Qin, D. Liang, G. Dong, H. Lai, H. Zhang, H. Zhao, I. L. long, J. Sun, J. Wang et al., "Autoglm: Autonomous foundation agents for guis," arXiv preprint arXiv:2411.00820, 2024.
[336] X. Liu, B. Qin, D. Liang, G. Dong, H. Lai, H. Zhang, H. Zhao, I. L. long, J. Sun, J. Wang 等, "AutoGLM：面向GUI的自主基础代理," arXiv预印本 arXiv:2411.00820, 2024.

[337] P. Pawlowski, K. Zawistowski, W. Lapacz, M. Skorupa, A. Wiacek, S. Postansque, and J. Hoscilowicz, "Tinyclick: Single-turn agent for empowering gui automation," arXiv preprint arXiv:2410.11871, 2024.
[337] P. Pawlowski, K. Zawistowski, W. Lapacz, M. Skorupa, A. Wiacek, S. Postansque, 和 J. Hoscilowicz, "TinyClick：用于增强GUI自动化的单轮代理," arXiv预印本 arXiv:2410.11871, 2024.

[338] H. Su, R. Sun, J. Yoon, P. Yin, T. Yu, and S. O. Arik, "Learn-by-interact: A data-centric framework for self-adaptive agents in realistic environments," 2025. [Online]. Available: https://arxiv.org/abs/2501.10893
[338] H. Su, R. Sun, J. Yoon, P. Yin, T. Yu, 和 S. O. Arik, "Learn-by-Interact：面向现实环境中自适应代理的数据中心框架," 2025. [在线]. 可获取：https://arxiv.org/abs/2501.10893

[339] Z. He, Z. Liu, P. Li, M. Fung, M. Yan, J. Zhang, F. Huang, and Y. Liu, "Enhancing language multi-agent learning with multi-agent credit re-assignment for interactive environment generalization," arXiv preprint arXiv:2502.14496, 2025.
[339] Z. He, Z. Liu, P. Li, M. Fung, M. Yan, J. Zhang, F. Huang, 和 Y. Liu, "通过多代理信用重新分配提升语言多代理学习以实现交互环境泛化," arXiv预印本 arXiv:2502.14496, 2025.

[340] X. Wang and B. Liu, "Oscar: Operating system control via state-aware reasoning and re-planning," arXiv preprint arXiv:2410.18963, 2024.
[340] X. Wang 和 B. Liu, "Oscar：基于状态感知推理与重新规划的操作系统控制," arXiv预印本 arXiv:2410.18963, 2024.

[341] C. Jia, M. Luo, Z. Dang, Q. Sun, F. Xu, J. Hu, T. Xie, and Z. Wu, "Agentstore: Scalable integration of heterogeneous agents as specialized generalist computer assistant," arXiv preprint arXiv:2410.18603, 2024.
[341] C. Jia, M. Luo, Z. Dang, Q. Sun, F. Xu, J. Hu, T. Xie, 和 Z. Wu, "AgentStore：作为专业通用计算机助手的异构代理可扩展集成," arXiv预印本 arXiv:2410.18603, 2024.

[342] Y. Wang, H. Zhang, J. Tian, and Y. Tang, "Ponder & press: Advancing visual gui agent towards general computer control," 2024. [Online]. Available: https://arxiv.org/abs/2412.01268
[342] Y. Wang, H. Zhang, J. Tian, 和 Y. Tang, "Ponder & Press：推动视觉GUI代理迈向通用计算机控制," 2024. [在线]. 可获取：https://arxiv.org/abs/2412.01268

[343] Y. Liu, P. Li, Z. Wei, C. Xie, X. Hu, X. Xu, S. Zhang, X. Han, H. Yang, and F. Wu, "Infiguiagent: A multimodal generalist gui agent with native reasoning and reflection," 2025. [Online]. Available: https://arxiv.org/abs/2501.04575
[343] Y. Liu, P. Li, Z. Wei, C. Xie, X. Hu, X. Xu, S. Zhang, X. Han, H. Yang, 和 F. Wu, "InfiguiAgent：具备本地推理与反思能力的多模态通用GUI代理," 2025. [在线]. 可获取：https://arxiv.org/abs/2501.04575

[344] S. Agashe, K. Wong, V. Tu, J. Yang, A. Li, and X. E. Wang, "Agent s2: A compositional generalist-specialist framework for computer use agents," arXiv preprint arXiv:2504.00906, 2025.
[344] S. Agashe, K. Wong, V. Tu, J. Yang, A. Li, 和 X. E. Wang, "Agent S2：面向计算机使用代理的组合通用-专家框架," arXiv预印本 arXiv:2504.00906, 2025.

[345] Z. Hu, S. Xiong, Y. Zhang, S.-K. Ng, A. T. Luu, B. An, S. Yan, and B. Hooi, "Guiding vlm agents with process rewards at inference time for gui navigation," 2025. [Online]. Available: https://arxiv.org/abs/2504.16073
[345] Z. Hu, S. Xiong, Y. Zhang, S.-K. Ng, A. T. Luu, B. An, S. Yan, 和 B. Hooi, "在推理阶段通过过程奖励引导视觉语言模型代理进行GUI导航," 2025. [在线]. 可获取：https://arxiv.org/abs/2504.16073

[346] J. Huang, Z. Zeng, W. Han, Y. Zhong, L. Zheng, S. Fu, J. Chen, and L. Ma, "Scaletrack: Scaling and back-tracking automated gui agents," 2025. [Online]. Available: https://arxiv.org/abs/2505.00416
[346] J. Huang, Z. Zeng, W. Han, Y. Zhong, L. Zheng, S. Fu, J. Chen, 和 L. Ma, “Scaletrack：自动化GUI代理的扩展与回溯,” 2025年。[在线]. 可获取：https://arxiv.org/abs/2505.00416

[347] Q. Chen, D. Pitawela, C. Zhao, G. Zhou, H.-T. Chen, and Q. Wu, "Webvln: Vision-and-language navigation on websites," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, no. 2, pp. 1165-1173, Mar. 2024. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/27878
[347] Q. Chen, D. Pitawela, C. Zhao, G. Zhou, H.-T. Chen, 和 Q. Wu, “Webvln：基于视觉与语言的网站导航,” AAAI人工智能会议论文集, 第38卷，第2期，页1165-1173，2024年3月。[在线]. 可获取：https://ojs.aaai.org/index.php/AAAI/article/view/27878

[348] X. H. Lu, Z. Kasner, and S. Reddy, "Weblinx: Real-world website navigation with multi-turn dialogue," in International Conference on Machine Learning. PMLR, 2024, pp. 33 007-33 056.
[348] X. H. Lu, Z. Kasner, 和 S. Reddy, “Weblinx：基于多轮对话的真实网站导航,” 机器学习国际会议论文集，PMLR，2024年，页33007-33056。

[349] Y. Pan, D. Kong, S. Zhou, C. Cui, Y. Leng, B. Jiang, H. Liu, Y. Shang, S. Zhou, T. Wu, and Z. Wu, "Webcanvas: Benchmarking web agents in online environments," 2024. [Online]. Available: https://arxiv.org/abs/2406.12373
[349] Y. Pan, D. Kong, S. Zhou, C. Cui, Y. Leng, B. Jiang, H. Liu, Y. Shang, S. Zhou, T. Wu, 和 Z. Wu, “Webcanvas：在线环境中网页代理的基准测试,” 2024年。[在线]. 可获取：https://arxiv.org/abs/2406.12373

[350] Y. Xu, D. Lu, Z. Shen, J. Wang, Z. Wang, Y. Mao, C. Xiong, and T. Yu, "Agenttrek: Agent trajectory synthesis via guiding replay with web tutorials," 2024. [Online]. Available: https://arxiv.org/abs/2412.09605
[350] Y. Xu, D. Lu, Z. Shen, J. Wang, Z. Wang, Y. Mao, C. Xiong, 和 T. Yu, “Agenttrek：通过网络教程引导回放的代理轨迹合成,” 2024年。[在线]. 可获取：https://arxiv.org/abs/2412.09605

[351] B. Trabucco, G. Sigurdsson, R. Piramuthu, and R. Salakhutdi-nov, "Towards internet-scale training for agents," arXiv preprint arXiv:2502.06776, 2025.
[351] B. Trabucco, G. Sigurdsson, R. Piramuthu, 和 R. Salakhutdi-nov, “迈向互联网规模的代理训练,” arXiv预印本 arXiv:2502.06776, 2025年。

[352] K. You, H. Zhang, E. Schoop, F. Weers, A. Swearngin, J. Nichols, Y. Yang, and Z. Gan, "Ferret-ui: Grounded mobile ui understanding with multimodal llms," in European Conference on Computer Vision. Springer, 2025, pp. 240-255.
[352] K. You, H. Zhang, E. Schoop, F. Weers, A. Swearngin, J. Nichols, Y. Yang, 和 Z. Gan, “Ferret-ui：基于多模态大语言模型的移动界面理解,” 欧洲计算机视觉会议论文集，Springer, 2025年，页240-255。

[353] Q. Wu, W. Xu, W. Liu, T. Tan, J. Liu, A. Li, J. Luan, B. Wang, and S. Shang, "Mobilevlm: A vision-language model for better intra- and inter-ui understanding," 2024. [Online]. Available: https://arxiv.org/abs/2409.14818
[353] Q. Wu, W. Xu, W. Liu, T. Tan, J. Liu, A. Li, J. Luan, B. Wang, 和 S. Shang, “Mobilevlm：提升界面内外理解的视觉语言模型,” 2024年。[在线]. 可获取：https://arxiv.org/abs/2409.14818

[354] Z. Meng, Y. Dai, Z. Gong, S. Guo, M. Tang, and T. Wei, "Vga: Vision gui assistant - minimizing hallucinations through image-centric fine-tuning," 2024. [Online]. Available: https://arxiv.org/abs/2406.14056
[354] Z. Meng, Y. Dai, Z. Gong, S. Guo, M. Tang, 和 T. Wei, “Vga：视觉GUI助手——通过以图像为中心的微调减少幻觉,” 2024年。[在线]. 可获取：https://arxiv.org/abs/2406.14056

[355] B. Deka, Z. Huang, C. Franzen, J. Hibschman, D. Afergan, 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, ser. UIST '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 845-854. [Online]. Available: https://doi.org/10.1145/3126594.3126651
[355] 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,” 第30届ACM用户界面软件与技术年会论文集，UIST '17系列。纽约，纽约，美国：计算机协会，2017年，页845-854。[在线]. 可获取：https://doi.org/10.1145/3126594.3126651

[356] 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," 2022. [Online]. Available: https://arxiv.org/abs/2202.02312
[356] A. Burns, D. Arsan, S. Agrawal, R. Kumar, K. Saenko, 和 B. A. Plummer, “用于交互式视觉语言导航的未知命令可行性数据集,” 2022年。[在线]. 可获取：https://arxiv.org/abs/2202.02312

[357] L. Sun, X. Chen, L. Chen, T. Dai, Z. Zhu, and K. Yu, "Meta-gui: Towards multi-modal conversational agents on mobile gui," 2022. [Online]. Available: https://arxiv.org/abs/2205.11029
[357] 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,” 2022年。[在线]. 可获取：https://arxiv.org/abs/2205.11029

[358] C. Rawles, A. Li, D. Rodriguez, O. Riva, and T. Lillicrap, "An-droidinthewild: A large-scale dataset for android device control," Advances in Neural Information Processing Systems, vol. 36, pp. 59708-59728, 2023.
[358] C. Rawles, A. Li, D. Rodriguez, O. Riva, 和 T. Lillicrap, “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, 第36卷, 页59708-59728, 2023年。

[359] 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
[359] 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

[360] 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," 2024. [Online]. Available: https://arxiv.org/abs/2407.17490
[360] 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Expo数据集，”2024年。[在线]. 可获取：https://arxiv.org/abs/2407.17490

[361] W. Chen, Z. Li, Z. Guo, and Y. Shen, "Octo-planner: On-device language model for planner-action agents," 2024. [Online]. Available: https://arxiv.org/abs/2406.18082
[361] W. Chen, Z. Li, Z. Guo, 和 Y. Shen, “Octo-planner：面向规划动作代理的设备端语言模型，”2024年。[在线]. 可获取：https://arxiv.org/abs/2406.18082

[362] K. Wang, T. Xia, Z. Gu, Y. Zhao, S. Shen, C. Meng, W. Wang, and K. Xu, "E-ant: A large-scale dataset for efficient automatic gui navigation," 2024. [Online]. Available: https://arxiv.org/abs/2406.14250
[362] K. Wang, T. Xia, Z. Gu, Y. Zhao, S. Shen, C. Meng, W. Wang, 和 K. Xu, “E-ant：高效自动GUI导航的大规模数据集，”2024年。[在线]. 可获取：https://arxiv.org/abs/2406.14250

[363] 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," 2024. [Online]. Available: https://arxiv.org/abs/2410.24024
[363] Y. Xu, X. Liu, X. Sun, S. Cheng, H. Yu, H. Lai, S. Zhang, D. Zhang, J. Tang, 和 Y. Dong, “Androidlab：Android自主代理的训练与系统性基准测试，”2024年。[在线]. 可获取：https://arxiv.org/abs/2410.24024

[364] L. Gao, L. Zhang, S. Wang, S. Wang, Y. Li, and M. Xu, "Mobileviews: A large-scale mobile gui dataset," arXiv preprint arXiv:2409.14337, 2024.
[364] L. Gao, L. Zhang, S. Wang, S. Wang, Y. Li, 和 M. Xu, “Mobileviews：大规模移动GUI数据集，”arXiv预印本 arXiv:2409.14337，2024年。

[365] Y. Sun, S. Zhao, T. Yu, H. Wen, S. Va, M. Xu, Y. Li, and C. Zhang, "Gui-xplore: Empowering generalizable gui agents with one exploration," arXiv preprint arXiv:2503.17709, 2025.
[365] Y. Sun, S. Zhao, T. Yu, H. Wen, S. Va, M. Xu, Y. Li, 和 C. Zhang, “Gui-xplore：通过一次探索赋能通用GUI代理，”arXiv预印本 arXiv:2503.17709，2025年。

[366] 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," 2024. [Online]. Available: https://arxiv.org/abs/2402.07945
[366] R. Niu, J. Li, S. Wang, Y. Fu, X. Hu, X. Leng, H. Kong, Y. Chang, 和 Q. Wang, “Screenagent：基于视觉语言模型的计算机控制代理，”2024年。[在线]. 可获取：https://arxiv.org/abs/2402.07945

[367] L. Wang, F. Yang, C. Zhang, J. Lu, J. Qian, S. He, P. Zhao, B. Qiao, R. Huang, S. Qin, Q. Su, J. Ye, Y. Zhang, J.-G. Lou, Q. Lin, S. Rajmohan, D. Zhang, and Q. Zhang, "Large action models: From inception to implementation," 2024. [Online]. Available: https://arxiv.org/abs/2412.10047
[367] L. Wang, F. Yang, C. Zhang, J. Lu, J. Qian, S. He, P. Zhao, B. Qiao, R. Huang, S. Qin, Q. Su, J. Ye, Y. Zhang, J.-G. Lou, Q. Lin, S. Rajmohan, D. Zhang, 和 Q. Zhang, “大型动作模型：从构想到实现，”2024年。[在线]. 可获取：https://arxiv.org/abs/2412.10047

[368] Y. Xu, L. Yang, H. Chen, H. Wang, Z. Chen, and Y. Tang, "Deskvision: Large scale desktop region captioning for advanced gui agents," arXiv preprint arXiv:2503.11170, 2025.
[368] Y. Xu, L. Yang, H. Chen, H. Wang, Z. Chen, 和 Y. Tang, “Deskvision：面向高级GUI代理的大规模桌面区域描述，”arXiv预印本 arXiv:2503.11170，2025年。

[369] Q. Sun, K. Cheng, Z. Ding, C. Jin, Y. Wang, F. Xu, Z. Wu, C. Jia, L. Chen, Z. Liu et al., "Os-genesis: Automating gui agent trajectory construction via reverse task synthesis," arXiv preprint arXiv:2412.19723, 2024.
[369] Q. Sun, K. Cheng, Z. Ding, C. Jin, Y. Wang, F. Xu, Z. Wu, C. Jia, L. Chen, Z. Liu 等, “Os-genesis：通过逆向任务合成自动构建GUI代理轨迹，”arXiv预印本 arXiv:2412.19723，2024年。

[370] R. Chawla, A. Jha, M. Kumar, M. NS, and I. Bhola, "Guide: Graphical user interface data for execution," arXiv preprint arXiv:2404.16048, 2024.
[370] R. Chawla, A. Jha, M. Kumar, M. NS, 和 I. Bhola, “Guide：用于执行的图形用户界面数据，”arXiv预印本 arXiv:2404.16048，2024年。

[371] D. Chen, Y. Huang, S. Wu, J. Tang, L. Chen, Y. Bai, Z. He, C. Wang, H. Zhou, Y. Li, T. Zhou, Y. Yu, C. Gao, Q. Zhang, Y. Gui, Z. Li, Y. Wan, P. Zhou, J. Gao, and L. Sun, "Gui-world: A dataset for gui-oriented multimodal Ilm-based agents," 2024. [Online]. Available: https://arxiv.org/abs/2406.10819
[371] D. Chen, Y. Huang, S. Wu, J. Tang, L. Chen, Y. Bai, Z. He, C. Wang, H. Zhou, Y. Li, T. Zhou, Y. Yu, C. Gao, Q. Zhang, Y. Gui, Z. Li, Y. Wan, P. Zhou, J. Gao, 和 L. Sun, “Gui-world：面向GUI的多模态ILM（交互语言模型）代理数据集，”2024年。[在线]. 可获取：https://arxiv.org/abs/2406.10819

[372] J. Zhang, T. Lan, M. Zhu, Z. Liu, T. Hoang, S. Kokane, W. Yao, J. Tan, A. Prabhakar, H. Chen et al., "xlam: A family of large action models to empower ai agent systems," arXiv preprint arXiv:2409.03215, 2024.
[372] 张杰, 兰涛, 朱明, 刘志, 黄涛, 科坎尼, 姚伟, 谭军, 普拉巴卡尔, 陈浩等, "xlam: 一系列大型动作模型以增强AI代理系统," arXiv预印本 arXiv:2409.03215, 2024.

[373] H. Shen, C. Liu, G. Li, X. Wang, Y. Zhou, C. Ma, and X. Ji, "Falcon-ui: Understanding gui before following user instructions," arXiv preprint arXiv:2412.09362, 2024.
[373] 沈浩, 刘晨, 李刚, 王晓, 周颖, 马超, 纪翔, "Falcon-ui: 理解图形用户界面以遵循用户指令," arXiv预印本 arXiv:2412.09362, 2024.

[374] X. Liu, T. Zhang, Y. Gu, I. L. long, Y. Xu, X. Song, S. Zhang, H. Lai, X. Liu, H. Zhao, J. Sun, X. Yang, Y. Yang, Z. Qi, S. Yao, X. Sun, S. Cheng, Q. Zheng, H. Yu, H. Zhang, W. Hong, M. Ding, L. Pan, X. Gu, A. Zeng, Z. Du, C. H. Song, Y. Su, Y. Dong, and J. Tang, "Visualagentbench: Towards large multimodal models as visual foundation agents," 2024. [Online]. Available: https://arxiv.org/abs/2408.06327
[374] 刘翔, 张涛, 顾阳, I. L. Long, 徐阳, 宋翔, 张帅, 赖浩, 刘翔, 赵浩, 孙杰, 杨翔, 杨洋, 齐志, 姚松, 孙翔, 程松, 郑强, 余浩, 张浩, 洪伟, 丁明, 潘磊, 顾翔, 曾安, 杜志, 宋春华, 苏阳, 董阳, 唐军, "VisualAgentBench: 面向大型多模态模型作为视觉基础代理," 2024. [在线]. 可用: https://arxiv.org/abs/2408.06327

[375] 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
[375] Baechler G., Sunkara S., Wang M., Zubach F., Mansoor H., Etter V., Cărbune V., Lin J., Chen J., Sharma A., "ScreenAI: 用于界面和信息图理解的视觉-语言模型," 2024. [在线]. 可用: https://arxiv.org/abs/2402.04615

[376] I. Chaimalas, A. VyLAniauskas, and G. Brostow, "Explorer: Robust collection of interactable gui elements," arXiv preprint arXiv:2504.09352, 2025.
[376] Chaimalas I., VyLAniauskas A., Brostow G., "Explorer: 交互式图形用户界面元素的鲁棒收集," arXiv预印本 arXiv:2504.09352, 2025.

[377] Z. Cheng, Z. Huang, J. Pan, Z. Hou, and M. Zhan, "Navi-plus: Managing ambiguous gui navigation tasks with follow-up," arXiv preprint arXiv:2503.24180, 2025.
[377] 程志, 黄志, 潘军, 侯志, 詹明, "Navi-plus: 通过后续管理模糊的图形用户界面导航任务," arXiv预印本 arXiv:2503.24180, 2025.

[378] OpenAI, "Gpt-4v(ision) system card," OpenAI, Tech. Rep., September 2023. [Online]. Available: https://cdn.openai.com/papers/GPTV_System_Card.pdf
[378] OpenAI, "GPT-4V(ision) 系统说明," OpenAI, 技术报告, 2023年9月. [在线]. 可用: https://cdn.openai.com/papers/GPTV_System_Card.pdf

[379] Z. Chen, J. Wu, W. Wang, W. Su, G. Chen, S. Xing, M. Zhong, Q. Zhang, X. Zhu, L. Lu et al., "Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 24 185-24 198.
[379] 陈志, 吴军, 王伟, 苏伟, 陈刚, 邢松, 钟明, 张强, 朱晓, 陆磊等, "InternVL: 扩展视觉基础模型并对齐通用视觉语言任务," 载于IEEE/CVF计算机视觉与模式识别会议论文集, 2024, 页码24185-24198.

[380] Z. Chen, W. Wang, H. Tian, S. Ye, Z. Gao, E. Cui, W. Tong, K. Hu, J. Luo, Z. Ma et al., "How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites," arXiv preprint arXiv:2404.16821, 2024.
[380] 陈志, 王伟, 田浩, 叶松, 高志, 崔恩, 童伟, 胡凯, 罗军, 马志等, "距离GPT-4V还有多远？利用开源套件缩小与商业多模态模型的差距," arXiv预印本 arXiv:2404.16821, 2024.

[381] W. Wang, Q. Lv, W. Yu, W. Hong, J. Qi, Y. Wang, J. Ji, Z. Yang, L. Zhao, X. Song, J. Xu, B. Xu, J. Li, Y. Dong, M. Ding, and J. Tang, "Cogvlm: Visual expert for pretrained language models," 2024. [Online]. Available: https://arxiv.org/abs/2311.03079
[381] 王伟, 吕强, 余伟, 洪伟, 齐军, 王颖, 纪军, 杨志, 赵磊, 宋翔, 徐军, 徐波, 李军, 董阳, 丁明, 唐军, "CogVLM: 预训练语言模型的视觉专家," 2024. [在线]. 可用: https://arxiv.org/abs/2311.03079

[382] H. You, H. Zhang, Z. Gan, X. Du, B. Zhang, Z. Wang, L. Cao, S.-F. Chang, and Y. Yang, "Ferret: Refer and ground anything anywhere at any granularity," arXiv preprint arXiv:2310.07704, 2023.
[382] 游浩, 张浩, 甘志, 杜翔, 张斌, 王志, 曹磊, Chang S.-F., 杨洋, "Ferret: 在任何位置以任意粒度引用和定位任何事物," arXiv预印本 arXiv:2310.07704, 2023.

[383] H. Liu, C. Li, Y. Li, and Y. J. Lee, "Improved baselines with visual instruction tuning," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 26 296- 26306.
[383] 刘浩, 李晨, 李阳, Lee Y. J., "通过视觉指令调优改进基线," 载于IEEE/CVF计算机视觉与模式识别会议论文集, 2024, 页码26296-26306.

[384] Z. Huang, Z. Cheng, J. Pan, Z. Hou, and M. Zhan, "Spirit-sight agent: Advanced gui agent with one look," arXiv preprint arXiv:2503.03196, 2025.
[384] Z. Huang, Z. Cheng, J. Pan, Z. Hou, 和 M. Zhan, “Spirit-sight agent：一款一眼识别的高级图形用户界面代理”，arXiv预印本 arXiv:2503.03196, 2025。

[385] M. Fereidouni and A. B. Siddique, "Search beyond queries: Training smaller language models for web interactions via reinforcement learning," 2024. [Online]. Available: https: //arxiv.org/abs/2404.10887
[385] M. Fereidouni 和 A. B. Siddique, “超越查询的搜索：通过强化学习训练更小的语言模型以实现网页交互”，2024。[在线]. 可获取：https://arxiv.org/abs/2404.10887

[386] L.-A. Thil, M. Popa, and G. Spanakis, "Navigating webai: Training agents to complete web tasks with large language models and reinforcement learning," in Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing, ser. SAC '24, vol. 30. ACM, Apr. 2024, p. 866-874. [Online]. Available: http://dx.doi.org/10.1145/3605098.3635903
[386] L.-A. Thil, M. Popa, 和 G. Spanakis, “导航WebAI：利用大型语言模型和强化学习训练代理完成网页任务”，载于第39届ACM/SIGAPP应用计算研讨会论文集，系列 SAC '24，第30卷。ACM，2024年4月，第866-874页。[在线]. 可获取：http://dx.doi.org/10.1145/3605098.3635903

[387] H. He, W. Yao, K. Ma, W. Yu, H. Zhang, T. Fang, Z. Lan, and D. Yu, "Openwebvoyager: Building multimodal web agents via iterative real-world exploration, feedback and optimization," 2024. [Online]. Available: https://arxiv.org/abs/2410.19609
[387] H. He, W. Yao, K. Ma, W. Yu, H. Zhang, T. Fang, Z. Lan, 和 D. Yu, “OpenWebVoyager：通过迭代的真实世界探索、反馈与优化构建多模态网页代理”，2024。[在线]. 可获取：https://arxiv.org/abs/2410.19609

[388] Z. Qi, X. Liu, I. L. long, H. Lai, X. Sun, X. Yang, J. Sun, Y. Yang, S. Yao, T. Zhang, W. Xu, J. Tang, and Y. Dong, "Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning," 2024. [Online]. Available: https://arxiv.org/abs/2411.02337
[388] Z. Qi, X. Liu, I. L. Long, H. Lai, X. Sun, X. Yang, J. Sun, Y. Yang, S. Yao, T. Zhang, W. Xu, J. Tang, 和 Y. Dong, “WebRL：通过自我进化的在线课程强化学习训练LLM网页代理”，2024。[在线]. 可获取：https://arxiv.org/abs/2411.02337

[389] J. Zhang, Z. Ding, C. Ma, Z. Chen, Q. Sun, Z. Lan, and J. He, "Breaking the data barrier-building gui agents through task generalization," arXiv preprint arXiv:2504.10127, 2025.
[389] J. Zhang, Z. Ding, C. Ma, Z. Chen, Q. Sun, Z. Lan, 和 J. He, “打破数据壁垒——通过任务泛化构建图形用户界面代理”，arXiv预印本 arXiv:2504.10127, 2025。

[390] Y. Qian, Y. Lu, A. Hauptmann, and O. Riva, "Visual grounding for user interfaces," in Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track), Y. Yang, A. Davani, A. Sil, and A. Kumar, Eds. Mexico City, Mexico: Association for Computational Linguistics, Jun. 2024, pp. 97-107. [Online]. Available: https://aclanthology.org/2024.naacl-industry.9
[390] Y. Qian, Y. Lu, A. Hauptmann, 和 O. Riva, “用户界面的视觉定位”，载于2024年北美计算语言学协会人类语言技术会议论文集（第6卷：产业轨迹），Y. Yang, A. Davani, A. Sil, 和 A. Kumar编。墨西哥城，墨西哥：计算语言学协会，2024年6月，第97-107页。[在线]. 可获取：https://aclanthology.org/2024.naacl-industry.9

[391] W. Chen, Z. Li, and M. Ma, "Octopus: On-device language model for function calling of software apis," 2024. [Online]. Available: https://arxiv.org/abs/2404.01549
[391] W. Chen, Z. Li, 和 M. Ma, “Octopus：面向软件API函数调用的设备端语言模型”，2024。[在线]. 可获取：https://arxiv.org/abs/2404.01549

[392] W. Chen and Z. Li, "Octopus v2: On-device language model for super agent," 2024. [Online]. Available: https://arxiv.org/abs/2404.01744
[392] W. Chen 和 Z. Li, “Octopus v2：面向超级代理的设备端语言模型”，2024。[在线]. 可获取：https://arxiv.org/abs/2404.01744

[393] —, "Octopus v3: Technical report for on-device sub-billion multimodal ai agent," 2024. [Online]. Available: https://arxiv.org/abs/2404.11459
[393] —, “Octopus v3：面向设备端亚十亿参数多模态人工智能代理的技术报告”，2024。[在线]. 可获取：https://arxiv.org/abs/2404.11459

[394] —, "Octopus v4: Graph of language models," 2024. [Online]. Available: https://arxiv.org/abs/2404.19296
[394] —, “Octopus v4：语言模型图谱”，2024。[在线]. 可获取：https://arxiv.org/abs/2404.19296

[395] W. Li, F.-L. Hsu, W. Bishop, F. Campbell-Ajala, M. Lin, and O. Riva, "Uinav: A practical approach to train on-device automation agents," in Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track), 2024, pp. 36- 51.
[395] W. Li, F.-L. Hsu, W. Bishop, F. Campbell-Ajala, M. Lin, 和 O. Riva, “UINav：一种实用的设备端自动化代理训练方法”，载于2024年北美计算语言学协会人类语言技术会议论文集（第6卷：产业轨迹），2024，第36-51页。

[396] Q. Wu, J. Liu, J. Hao, J. Wang, and K. Shao, "Vsc-rl: Advancing autonomous vision-language agents with variational subgoal-conditioned reinforcement learning," arXiv preprint arXiv:2502.07949, 2025.
[396] Q. Wu, J. Liu, J. Hao, J. Wang, 和 K. Shao, “VSC-RL：通过变分子目标条件强化学习推进自主视觉语言代理”，arXiv预印本 arXiv:2502.07949, 2025。

[397] G. Papoudakis, T. Coste, Z. Wu, J. Hao, J. Wang, and K. Shao, "Appvlm: A lightweight vision language model for online app control," arXiv preprint arXiv:2502.06395, 2025.
[397] G. Papoudakis, T. Coste, Z. Wu, J. Hao, J. Wang, 和 K. Shao, “AppVLM：一款轻量级视觉语言模型用于在线应用控制”，arXiv预印本 arXiv:2502.06395, 2025。

[398] H. Bai, Y. Zhou, L. E. Li, S. Levine, and A. Kumar, "Digi-q: Learning q-value functions for training device-control agents," arXiv preprint arXiv:2502.15760, 2025.
[398] H. Bai, Y. Zhou, L. E. Li, S. Levine, 和 A. Kumar, “Digi-q：用于训练设备控制代理的Q值函数学习,” arXiv预印本 arXiv:2502.15760, 2025.

[399] J. Zheng, L. Wang, F. Yang, C. Zhang, L. Mei, W. Yin, Q. Lin, D. Zhang, S. Rajmohan, and Q. Zhang, "Vem: Environment-free exploration for training gui agent with value environment model," arXiv preprint arXiv:2502.18906, 2025.
[399] J. Zheng, L. Wang, F. Yang, C. Zhang, L. Mei, W. Yin, Q. Lin, D. Zhang, S. Rajmohan, 和 Q. Zhang, “Vem：基于价值环境模型的无环境探索用于训练GUI代理,” arXiv预印本 arXiv:2502.18906, 2025.

[400] Z. Wang, W. Chen, L. Yang, S. Zhou, S. Zhao, H. Zhan, J. Jin, L. Li, Z. Shao, and J. Bu, "Mp-gui: Modality perception with mllms for gui understanding," arXiv preprint arXiv:2503.14021, 2025.
[400] Z. Wang, W. Chen, L. Yang, S. Zhou, S. Zhao, H. Zhan, J. Jin, L. Li, Z. Shao, 和 J. Bu, “Mp-gui：结合多模态大语言模型（MLLMs）的GUI理解感知,” arXiv预印本 arXiv:2503.14021, 2025.

[401] Z. Lu, Y. Chai, Y. Guo, X. Yin, L. Liu, H. Wang, G. Xiong, and H. Li, "Ui-r1: Enhancing action prediction of gui agents by reinforcement learning," arXiv preprint arXiv:2503.21620, 2025.
[401] Z. Lu, Y. Chai, Y. Guo, X. Yin, L. Liu, H. Wang, G. Xiong, 和 H. Li, “Ui-r1：通过强化学习提升GUI代理的动作预测,” arXiv预印本 arXiv:2503.21620, 2025.

[402] D. Luo, B. Tang, K. Li, G. Papoudakis, J. Song, S. Gong, J. Hao, J. Wang, and K. Shao, "Vimo: A generative visual gui world model for app agent," 2025. [Online]. Available: https://arxiv.org/abs/2504.13936
[402] D. Luo, B. Tang, K. Li, G. Papoudakis, J. Song, S. Gong, J. Hao, J. Wang, 和 K. Shao, “Vimo：用于应用代理的生成式视觉GUI世界模型,” 2025. [在线]. 可获取：https://arxiv.org/abs/2504.13936

[403] J. Yang, Y. Dong, S. Liu, B. Li, Z. Wang, H. Tan, C. Jiang, J. Kang, Y. Zhang, K. Zhou et al., "Octopus: Embodied vision-language programmer from environmental feedback," in European Conference on Computer Vision. Springer, 2025, pp. 20-38.
[403] J. Yang, Y. Dong, S. Liu, B. Li, Z. Wang, H. Tan, C. Jiang, J. Kang, Y. Zhang, K. Zhou 等, “Octopus：基于环境反馈的具身视觉-语言程序员,” 载于欧洲计算机视觉会议（ECCV），施普林格，2025，页20-38.

[404] Y. Jin, S. Petrangeli, Y. Shen, and G. Wu, "Screenllm: Stateful screen schema for efficient action understanding and prediction," 2025.
[404] Y. Jin, S. Petrangeli, Y. Shen, 和 G. Wu, “Screenllm：用于高效动作理解与预测的有状态屏幕模式,” 2025.

[405] Z. Zhang, W. Xie, X. Zhang, and Y. Lu, "Reinforced ui instruction grounding: Towards a generic ui task automation api," 2023. [Online]. Available: https://arxiv.org/abs/2310.04716
[405] Z. Zhang, W. Xie, X. Zhang, 和 Y. Lu, “强化的UI指令定位：迈向通用UI任务自动化API,” 2023. [在线]. 可获取：https://arxiv.org/abs/2310.04716

[406] 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," arXiv preprint arXiv:2410.18967, 2024.
[406] 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,” arXiv预印本 arXiv:2410.18967, 2024.

[407] 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
[407] 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, 和 G. Shi, “Ui-tars：开创性自动化GUI交互的本地代理,” 2025. [在线]. 可获取：https://arxiv.org/abs/2501.12326

[408] A. Rahman, R. Chawla, M. Kumar, A. Datta, A. Jha, M. NS, and I. Bhola, "V-zen: Efficient gui understanding and precise grounding with a novel multimodal llm," arXiv preprint arXiv:2405.15341, 2024.
[408] A. Rahman, R. Chawla, M. Kumar, A. Datta, A. Jha, M. NS, 和 I. Bhola, “V-zen：基于新型多模态语言模型（LLM）的高效GUI理解与精准定位,” arXiv预印本 arXiv:2405.15341, 2024.

[409] J. Yang, R. Tan, Q. Wu, R. Zheng, B. Peng, Y. Liang, Y. Gu, M. Cai, S. Ye, J. Jang et al., "Magma: A foundation model for multimodal ai agents," arXiv preprint arXiv:2502.13130, 2025.
[409] J. Yang, R. Tan, Q. Wu, R. Zheng, B. Peng, Y. Liang, Y. Gu, M. Cai, S. Ye, J. Jang 等, “Magma：多模态AI代理的基础模型,” arXiv预印本 arXiv:2502.13130, 2025.

[410] X. Xia and R. Luo, "Gui-r1: A generalist r1-style vision-language action model for gui agents," arXiv preprint arXiv:2504.10458, 2025.
[410] 夏晓(X. Xia)和罗锐(R. Luo)，“Gui-r1：一种面向GUI代理的通用r1风格视觉-语言动作模型”，arXiv预印本 arXiv:2504.10458，2025年。

[411] Y. Liu, P. Li, C. Xie, X. Hu, X. Han, S. Zhang, H. Yang, and F. Wu, "Infigui-r1: Advancing multimodal gui agents from reactive actors to deliberative reasoners," 2025. [Online]. Available: https://arxiv.org/abs/2504.14239
[411] 刘洋(Y. Liu)、李鹏(P. Li)、谢晨(C. Xie)、胡晓(X. Hu)、韩旭(X. Han)、张帅(S. Zhang)、杨辉(H. Yang)和吴飞(F. Wu)，“Infigui-r1：推动多模态GUI代理从反应型执行者向深思熟虑的推理者转变”，2025年。[在线]。可获取：https://arxiv.org/abs/2504.14239

[412] 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," in The Twelfth International Conference on Learning Representations.
[412] 周松(S. Zhou)、徐飞飞(F. F. Xu)、朱浩(H. Zhu)、周翔(X. Zhou)、罗锐(R. Lo)、斯里达尔(A. Sridhar)、程翔(X. Cheng)、欧涛(T. Ou)、比斯克(Y. Bisk)、弗里德(D. Fried)等，“Webarena：用于构建自主代理的真实网页环境”，发表于第十二届国际学习表征会议。

[413] 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," 2024. [Online]. Available: https://arxiv.org/abs/2401.13649
[413] 柯俊逸(J. Y. Koh)、罗锐(R. Lo)、张磊(L. Jang)、杜弗(V. Duvvur)、林明昌(M. C. Lim)、黄鹏宇(P.- Y. Huang)、纽比格(G. Neubig)、周松(S. Zhou)、萨拉胡丁诺夫(R. Salakhutdinov)和弗里德(D. Fried)，“Visualwebarena：在真实视觉网页任务上评估多模态代理”，2024年。[在线]。可获取：https://arxiv.org/abs/2401.13649

[414] 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," 2024. [Online]. Available: https://arxiv.org/abs/2402.15057
[414] 邓毅，张晓，张伟，袁洋，吴思凯，蔡天石，“关于对话式网络代理的多轮指令跟随”，2024年。[在线]. 可获取：https://arxiv.org/abs/2402.15057

[415] Z. Zhang, S. Tian, L. Chen, and Z. Liu, "Mmina: Benchmarking multihop multimodal internet agents," 2024. [Online]. Available: https://arxiv.org/abs/2404.09992
[415] 张志，田帅，陈磊，刘志，“Mmina：多跳多模态互联网代理的基准测试”，2024年。[在线]. 可获取：https://arxiv.org/abs/2404.09992

[416] I. Levy, B. Wiesel, S. Marreed, A. Oved, A. Yaeli, and S. Shlo-mov, "St-webagentbench: A benchmark for evaluating safety and trustworthiness in web agents," arXiv preprint arXiv:2410.06703, 2024.
[416] I. Levy, B. Wiesel, S. Marreed, A. Oved, A. Yaeli, 和 S. Shlomov，“St-webagentbench：用于评估网络代理安全性和可信度的基准”，arXiv预印本 arXiv:2410.06703，2024年。

[417] H. Furuta, Y. Matsuo, A. Faust, and I. Gur, "Exposing limitations of language model agents in sequential-task compositions on the web," in ICLR 2024 Workshop on Large Language Model (LLM) Agents, 2024.
[417] 古田浩，松尾洋，A. Faust，I. Gur，“揭示语言模型代理在网络上顺序任务组合中的局限性”，发表于ICLR 2024大型语言模型（LLM）代理研讨会，2024年。

[418] K. Xu, Y. Kordi, T. Nayak, A. Asija, Y. Wang, K. Sanders, A. Byerly, J. Zhang, B. Van Durme, and D. Khashabi, "Tur [k] ingbench: A challenge benchmark for web agents," arXiv preprint arXiv:2403.11905, 2024.
[418] 徐凯，Y. Kordi，T. Nayak，A. Asija，王洋，K. Sanders，A. Byerly，张杰，B. Van Durme，和D. Khashabi，“Tur [k] ingbench：网络代理的挑战基准”，arXiv预印本 arXiv:2403.11905，2024年。

[419] T. Xie, D. Zhang, J. Chen, X. Li, S. Zhao, R. Cao, T. J. Hua, Z. Cheng, D. Shin, F. Lei, Y. Liu, Y. Xu, S. Zhou, S. Savarese, C. Xiong, V. Zhong, and T. Yu, "Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments," 2024. [Online]. Available: https://arxiv.org/abs/2404.07972
[419] 谢涛，张东，陈军，李翔，赵帅，曹锐，华天杰，程志，申东，雷飞，刘洋，徐阳，周松，S. Savarese，熊超，V. Zhong，和余涛，“Osworld：真实计算机环境中开放式任务的多模态代理基准”，2024年。[在线]. 可获取：https://arxiv.org/abs/2404.07972

[420] A. Drouin, M. Gasse, M. Caccia, I. H. Laradji, M. Del Verme, T. Marty, L. Boisvert, M. Thakkar, Q. Cappart, D. Vazquez et al., "Workarena: How capable are web agents at solving common knowledge work tasks?" arXiv preprint arXiv:2403.07718, 2024.
[420] A. Drouin，M. Gasse，M. Caccia，I. H. Laradji，M. Del Verme，T. Marty，L. Boisvert，M. Thakkar，Q. Cappart，D. Vazquez 等，“Workarena：网络代理解决常见知识工作任务的能力如何？”arXiv预印本 arXiv:2403.07718，2024年。

[421] L. Jang, Y. Li, C. Ding, J. Lin, P. P. Liang, D. Zhao, R. Bonatti, and K. Koishida, "Videowebarena: Evaluating long context multimodal agents with video understanding web tasks," arXiv preprint arXiv:2410.19100, 2024.
[421] L. Jang，李阳，丁晨，林军，P. P. Liang，赵东，R. Bonatti，和K. Koishida，“Videowebarena：通过视频理解网络任务评估长上下文多模态代理”，arXiv预印本 arXiv:2410.19100，2024年。

[422] X. Ma, Y. Wang, Y. Yao, T. Yuan, A. Zhang, Z. Zhang, and H. Zhao, "Caution for the environment: Multimodal agents are susceptible to environmental distractions," 2024. [Online]. Available: https://arxiv.org/abs/2408.02544
[422] 马翔，王洋，姚阳，袁涛，A. Zhang，张志，赵宏，“环境警示：多模态代理易受环境干扰”，2024年。[在线]. 可获取：https://arxiv.org/abs/2408.02544

[423] M. Wornow, A. Narayan, B. Viggiano, I. S. Khare, T. Verma, T. Thompson, M. A. F. Hernandez, S. Sundar, C. Trujillo, K. Chawla et al., "Wonderbread: A benchmark for evaluating multimodal foundation models on business process management tasks," in The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track.
[423] M. Wornow，A. Narayan，B. Viggiano，I. S. Khare，T. Verma，T. Thompson，M. A. F. Hernandez，S. Sundar，C. Trujillo，K. Chawla 等，“Wonderbread：评估多模态基础模型在业务流程管理任务中的基准”，发表于第三十八届神经信息处理系统大会数据集与基准赛道。

[424] B. Zheng, B. Gou, S. Salisbury, Z. Du, H. Sun, and Y. Su, "Webolympus: An open platform for web agents on live websites," in Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, 2024, pp. 187-197.
[424] 郑博，苟斌，S. Salisbury，杜志，孙浩，苏阳，“Webolympus：面向实时网站的网络代理开放平台”，发表于2024年自然语言处理实证方法会议系统演示，2024年，第187-197页。

[425] S. Yao, H. Chen, J. Yang, and K. Narasimhan, "Webshop: Towards scalable real-world web interaction with grounded language agents," 2023. [Online]. Available: https://arxiv.org/abs/2207.01206
[425] 姚森，陈浩，杨军，K. Narasimhan，“Webshop：面向可扩展真实网络交互的有根语言代理”，2023年。[在线]. 可获取：https://arxiv.org/abs/2207.01206

[426] D. Chezelles, T. Le Sellier, M. Gasse, A. Lacoste, A. Drouin, M. Caccia, L. Boisvert, M. Thakkar, T. Marty, R. Assouel et al., "The browsergym ecosystem for web agent research," arXiv preprint arXiv:2412.05467, 2024.
[426] D. Chezelles, T. Le Sellier, M. Gasse, A. Lacoste, A. Drouin, M. Caccia, L. Boisvert, M. Thakkar, T. Marty, R. Assouel 等，“BrowserGym生态系统用于网页代理研究，” arXiv预印本 arXiv:2412.05467, 2024.

[427] J. Wu, W. Yin, Y. Jiang, Z. Wang, Z. Xi, R. Fang, D. Zhou, P. Xie, and F. Huang, "Webwalker: Benchmarking llms in web traversal," 2025. [Online]. Available: https://arxiv.org/abs/2501.07572
[427] J. Wu, W. Yin, Y. Jiang, Z. Wang, Z. Xi, R. Fang, D. Zhou, P. Xie, 和 F. Huang，“Webwalker：网页遍历中大语言模型（LLMs）的基准测试，” 2025. [在线]. 可用: https://arxiv.org/abs/2501.07572

[428] G. Thomas, A. J. Chan, J. Kang, W. Wu, F. Christianos, F. Greenlee, A. Toulis, and M. Purtorab, "Webgames: Challenging general-purpose web-browsing ai agents," arXiv preprint arXiv:2502.18356, 2025.
[428] G. Thomas, A. J. Chan, J. Kang, W. Wu, F. Christianos, F. Greenlee, A. Toulis, 和 M. Purtorab，“Webgames：挑战通用网页浏览人工智能代理，” arXiv预印本 arXiv:2502.18356, 2025.

[429] A. D. Tur, N. Meade, X. H. Lù, A. Zambrano, A. Patel, E. Durmus, S. Gella, K. Stańczak, and S. Reddy, "Safearena: Evaluating the safety of autonomous web agents," 2025. [Online]. Available: https://arxiv.org/abs/2503.04957
[429] A. D. Tur, N. Meade, X. H. Lù, A. Zambrano, A. Patel, E. Durmus, S. Gella, K. Stańczak, 和 S. Reddy，“SafeArena：评估自主网页代理的安全性，” 2025. [在线]. 可用: https://arxiv.org/abs/2503.04957

[430] S. Kara, F. Faisal, and S. Nath, "Waber: Web agent benchmarking for efficiency and reliability," in ICLR 2025 Workshop on Foundation Models in the Wild.
[430] S. Kara, F. Faisal, 和 S. Nath，“Waber：网页代理效率与可靠性基准测试，” 载于 ICLR 2025 野外基础模型研讨会.

[431] T. Xue, W. Qi, T. Shi, C. H. Song, B. Gou, D. Song, H. Sun, and Y. Su, "An illusion of progress? assessing the current state of web agents," arXiv preprint arXiv:2504.01382, 2025.
[431] T. Xue, W. Qi, T. Shi, C. H. Song, B. Gou, D. Song, H. Sun, 和 Y. Su，“进展的错觉？评估当前网页代理的状态，” arXiv预印本 arXiv:2504.01382, 2025.

[432] A. Zharmagambetov, C. Guo, I. Evtimov, M. Pavlova, R. Salakhutdi-nov, and K. Chaudhuri, "Agentdam: Privacy leakage evaluation for autonomous web agents," arXiv preprint arXiv:2503.09780, 2025.
[432] A. Zharmagambetov, C. Guo, I. Evtimov, M. Pavlova, R. Salakhutdi-nov, 和 K. Chaudhuri，“AgentDAM：自主网页代理的隐私泄露评估，” arXiv预印本 arXiv:2503.09780, 2025.

[433] X. H. Lù, A. Kazemnejad, N. Meade, A. Patel, D. Shin, A. Zambrano, K. Stańczak, P. Shaw, C. J. Pal, and S. Reddy, "Agentrewardbench: Evaluating automatic evaluations of web agent trajectories," arXiv preprint arXiv:2504.08942, 2025.
[433] X. H. Lù, A. Kazemnejad, N. Meade, A. Patel, D. Shin, A. Zambrano, K. Stańczak, P. Shaw, C. J. Pal, 和 S. Reddy，“AgentRewardBench：评估网页代理轨迹自动评估方法，” arXiv预印本 arXiv:2504.08942, 2025.

[434] S. Ye, H. Shi, D. Shih, H. Yun, T. Roosta, and T. Shu, "Real-webassist: A benchmark for long-horizon web assistance with real-world users," arXiv preprint arXiv:2504.10445, 2025.
[434] S. Ye, H. Shi, D. Shih, H. Yun, T. Roosta, 和 T. Shu，“Real-WebAssist：面向真实用户的长时域网页辅助基准，” arXiv预印本 arXiv:2504.10445, 2025.

[435] D. Garg, S. VanWeelden, D. Caples, A. Draguns, N. Ravi, P. Putta, N. Garg, T. Abraham, M. Lara, F. Lopez et al., "Real: Benchmarking autonomous agents on deterministic simulations of real websites," arXiv preprint arXiv:2504.11543, 2025.
[435] D. Garg, S. VanWeelden, D. Caples, A. Draguns, N. Ravi, P. Putta, N. Garg, T. Abraham, M. Lara, F. Lopez 等，“REAL：基于真实网站确定性仿真的自主代理基准测试，” arXiv预印本 arXiv:2504.11543, 2025.

[436] Y. Song, K. Thai, C. M. Pham, Y. Chang, M. Nadaf, and M. Iyyer, "Bearcubs: A benchmark for computer-using web agents," arXiv preprint arXiv:2503.07919, 2025.
[436] Y. Song, K. Thai, C. M. Pham, Y. Chang, M. Nadaf, 和 M. Iyyer，“BearCUBS：面向使用计算机的网页代理的基准，” arXiv预印本 arXiv:2503.07919, 2025.

[437] I. Evtimov, A. Zharmagambetov, A. Grattafiori, C. Guo, and K. Chaudhuri, "Wasp: Benchmarking web agent security against prompt injection attacks," arXiv preprint arXiv:2504.18575, 2025.
[437] I. Evtimov, A. Zharmagambetov, A. Grattafiori, C. Guo, 和 K. Chaudhuri, “WASP：针对提示注入攻击的网页代理安全基准测试,” arXiv预印本 arXiv:2504.18575, 2025.

[438] D. Zhang, Z. Shen, R. Xie, S. Zhang, T. Xie, Z. Zhao, S. Chen, L. Chen, H. Xu, R. Cao, and K. Yu, "Mobile-env: Building qualified evaluation benchmarks for llm-gui interaction," 2024. [Online]. Available: https://arxiv.org/abs/2305.08144
[438] D. Zhang, Z. Shen, R. Xie, S. Zhang, T. Xie, Z. Zhao, S. Chen, L. Chen, H. Xu, R. Cao, 和 K. Yu, “Mobile-Env：构建合格的LLM-GUI交互评估基准,” 2024. [在线]. 可用: https://arxiv.org/abs/2305.08144

[439] J. Lee, T. Min, M. An, C. Kim, and K. Lee, "Benchmarking mobile device control agents across diverse configurations," 2024. [Online]. Available: https://arxiv.org/abs/2404.16660
[439] J. Lee, T. Min, M. An, C. Kim, 和 K. Lee, “跨多种配置的移动设备控制代理基准测试,” 2024. [在线]. 可获取: https://arxiv.org/abs/2404.16660

[440] 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," 2024. [Online]. Available: https://arxiv.org/abs/2405.14573
[440] 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: 一个用于自主代理的动态基准测试环境,” 2024. [在线]. 可获取: https://arxiv.org/abs/2405.14573

[441] 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.
[441] M. Xing, R. Zhang, H. Xue, Q. Chen, F. Yang, 和 Z. Xiao, “理解复杂Android环境中大型语言模型代理的弱点,” 载于第30届ACM SIGKDD知识发现与数据挖掘会议论文集, 2024, 页6061-6072.

[442] S. Deng, W. Xu, H. Sun, W. Liu, T. Tan, J. Liu, A. Li, J. Luan, B. Wang, R. Yan et al., "Mobile-bench: An evaluation benchmark for llm-based mobile agents," arXiv preprint arXiv:2407.00993, 2024.
[442] S. Deng, W. Xu, H. Sun, W. Liu, T. Tan, J. Liu, A. Li, J. Luan, B. Wang, R. Yan 等, “Mobile-bench: 基于Llm的移动代理评估基准,” arXiv预印本 arXiv:2407.00993, 2024.

[443] J. Lee, D. Hahm, J. S. Choi, W. B. Knox, and K. Lee, "Mobilesafe-tybench: Evaluating safety of autonomous agents in mobile device control," arXiv preprint arXiv:2410.17520, 2024.
[443] J. Lee, D. Hahm, J. S. Choi, W. B. Knox, 和 K. Lee, “Mobilesafe-tybench: 移动设备控制中自主代理安全性的评估,” arXiv预印本 arXiv:2410.17520, 2024.

[444] J. Chen, D. Yuen, B. Xie, Y. Yang, G. Chen, Z. Wu, L. Yixing, X. Zhou, W. Liu, S. Wang et al., "Spa-bench: A comprehensive benchmark for smartphone agent evaluation," in NeurIPS 2024 Workshop on Open-World Agents, 2024.
[444] J. Chen, D. Yuen, B. Xie, Y. Yang, G. Chen, Z. Wu, L. Yixing, X. Zhou, W. Liu, S. Wang 等, “Spa-bench: 智能手机代理评估的综合基准,” 载于NeurIPS 2024开放世界代理研讨会, 2024.

[445] 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 task automation," 2024. [Online]. Available: https://arxiv.org/abs/2404.16054
[445] L. Zhang, S. Wang, X. Jia, Z. Zheng, Y. Yan, L. Gao, Y. Li, 和 M. Xu, “Llamatouch: 一个真实且可扩展的移动UI任务自动化测试平台,” 2024. [在线]. 可获取: https://arxiv.org/abs/2404.16054

[446] 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," 2024. [Online]. Available: https://arxiv.org/abs/2406.08184
[446] 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Llm代理基准,” 2024. [在线]. 可获取: https://arxiv.org/abs/2406.08184

[447] 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," arXiv preprint arXiv:2412.18426, 2024.
[447] 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,” arXiv预印本 arXiv:2412.18426, 2024.

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

[449] D. Ran, M. Wu, H. Yu, Y. Li, J. Ren, Y. Cao, X. Zeng, H. Lu, Z. Xu, M. Xu et al., "Beyond pass or fail: A multi-dimensional benchmark for mobile ui navigation," arXiv preprint arXiv:2501.02863, 2025.
[449] D. Ran, M. Wu, H. Yu, Y. Li, J. Ren, Y. Cao, X. Zeng, H. Lu, Z. Xu, M. Xu 等, “超越通过或失败: 移动UI导航的多维基准,” arXiv预印本 arXiv:2501.02863, 2025.

[450] Y. Chen, X. Hu, K. Yin, J. Li, and S. Zhang, "Aeia-mn: Evaluating the robustness of multimodal llm-powered mobile agents against active environmental injection attacks," arXiv preprint arXiv:2502.13053, 2025.
[450] Y. Chen, X. Hu, K. Yin, J. Li, 和 S. Zhang, “Aeia-mn: 评估多模态llm驱动移动代理对主动环境注入攻击的鲁棒性,” arXiv预印本 arXiv:2502.13053, 2025.

[451] J. Sun, Z. Hua, and Y. Xia, "Autoeval: A practical framework for autonomous evaluation of mobile agents," arXiv preprint arXiv:2503.02403, 2025.
[451] J. Sun, Z. Hua, 和 Y. Xia, “Autoeval: 一个用于移动代理自主评估的实用框架,” arXiv预印本 arXiv:2503.02403, 2025.

[452] W. Wang, Z. Yu, R. Ye, J. Zhang, S. Chen, and Y. Wang, "Fedmabench: Benchmarking mobile agents on decentralized heterogeneous user data," arXiv preprint arXiv:2503.05143, 2025.
[452] W. Wang, Z. Yu, R. Ye, J. Zhang, S. Chen, 和 Y. Wang, “Fedmabench：基于去中心化异构用户数据的移动代理基准测试，”arXiv预印本 arXiv:2503.05143, 2025。

[453] R. Bonatti, D. Zhao, F. Bonacci, D. Dupont, S. Abdali, Y. Li, Y. Lu, J. Wagle, K. Koishida, A. Bucker, L. Jang, and Z. Hui, "Windows agent arena: Evaluating multi-modal os agents at scale," 2024. [Online]. Available: https://arxiv.org/abs/2409.08264
[453] R. Bonatti, D. Zhao, F. Bonacci, D. Dupont, S. Abdali, Y. Li, Y. Lu, J. Wagle, K. Koishida, A. Bucker, L. Jang, 和 Z. Hui, “Windows代理竞技场：大规模评估多模态操作系统代理，”2024。[在线]. 可获取：https://arxiv.org/abs/2409.08264

[454] R. Cao, F. Lei, H. Wu, J. Chen, Y. Fu, H. Gao, X. Xiong, H. Zhang, Y. Mao, W. Hu, T. Xie, H. Xu, D. Zhang, S. Wang, R. Sun, P. Yin, C. Xiong, A. Ni, Q. Liu, V. Zhong, L. Chen, K. Yu, and T. Yu, "Spider2-v: How far are multimodal agents from automating data science and engineering workflows?" 2024. [Online]. Available: https://arxiv.org/abs/2407.10956
[454] R. Cao, F. Lei, H. Wu, J. Chen, Y. Fu, H. Gao, X. Xiong, H. Zhang, Y. Mao, W. Hu, T. Xie, H. Xu, D. Zhang, S. Wang, R. Sun, P. Yin, C. Xiong, A. Ni, Q. Liu, V. Zhong, L. Chen, K. Yu, 和 T. Yu, “Spider2-v：多模态代理在自动化数据科学与工程工作流方面的距离有多远？”2024。[在线]. 可获取：https://arxiv.org/abs/2407.10956

[455] Z. Wang, Y. Cui, L. Zhong, Z. Zhang, D. Yin, B. Y. Lin, and J. Shang, "Officebench: Benchmarking language agents across multiple applications for office automation," 2024. [Online]. Available: https://arxiv.org/abs/2407.19056
[455] Z. Wang, Y. Cui, L. Zhong, Z. Zhang, D. Yin, B. Y. Lin, 和 J. Shang, “Officebench：跨多应用的办公自动化语言代理基准测试，”2024。[在线]. 可获取：https://arxiv.org/abs/2407.19056

[456] 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
[456] H. H. Zhao, D. Gao, 和 M. Z. Shou, “Worldgui：面向全面桌面GUI自动化的动态测试，”2025。[在线]. 可获取：https://arxiv.org/abs/2502.08047

[457] S. Nayak, X. Jian, K. Q. Lin, J. A. Rodriguez, M. Kalsi, R. Awal, N. Chapados, M. T. Özsu, A. Agrawal, D. Vazquez et al., "Ui-vision: A desktop-centric gui benchmark for visual perception and interaction," arXiv preprint arXiv:2503.15661, 2025.
[457] S. Nayak, X. Jian, K. Q. Lin, J. A. Rodriguez, M. Kalsi, R. Awal, N. Chapados, M. T. Özsu, A. Agrawal, D. Vazquez 等, “Ui-vision：面向视觉感知与交互的桌面中心GUI基准，”arXiv预印本 arXiv:2503.15661, 2025。

[458] B. Wang, X. Wang, J. Deng, T. Xie, R. Li, Y. Zhang, G. Li, T. J. Hua, I. Stoica, W.-L. Chiang, D. Yang, Y. Su, Y. Zhang, Z. Wang, V. Zhong, and T. Yu, "Computer agent arena: Compare & test computer use agents on crowdsourced real-world tasks," 2025.
[458] B. Wang, X. Wang, J. Deng, T. Xie, R. Li, Y. Zhang, G. Li, T. J. Hua, I. Stoica, W.-L. Chiang, D. Yang, Y. Su, Y. Zhang, Z. Wang, V. Zhong, 和 T. Yu, “计算机代理竞技场：基于众包真实任务的计算机使用代理比较与测试，”2025。

[459] R. Kapoor, Y. P. Butala, M. Russak, J. Y. Koh, K. Kamble, W. Alshikh, and R. Salakhutdinov, "Omniact: A dataset and benchmark for enabling multimodal generalist autonomous agents for desktop and web," 2024. [Online]. Available: https://arxiv.org/abs/2402.17553
[459] R. Kapoor, Y. P. Butala, M. Russak, J. Y. Koh, K. Kamble, W. Alshikh, 和 R. Salakhutdinov, “Omniact：支持桌面与网页多模态通用自主代理的数据集与基准，”2024。[在线]. 可获取：https://arxiv.org/abs/2402.17553

[460] 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," 2024. [Online]. Available: https://arxiv.org/abs/2406.10227
[460] K. Q. Lin, L. Li, D. Gao, Q. WU, M. Yan, Z. Yang, L. Wang, 和 M. Z. Shou, “Videogui：基于教学视频的GUI自动化基准，”2024。[在线]. 可获取：https://arxiv.org/abs/2406.10227

[461] T. Xu, L. Chen, D.-J. Wu, Y. Chen, Z. Zhang, X. Yao, Z. Xie, Y. Chen, S. Liu, B. Qian, P. Torr, B. Ghanem, and G. Li, "Crab: Cross-environment agent benchmark for multimodal language model agents," 2024. [Online]. Available: https://arxiv.org/abs/2407.01511
[461] T. Xu, L. Chen, D.-J. Wu, Y. Chen, Z. Zhang, X. Yao, Z. Xie, Y. Chen, S. Liu, B. Qian, P. Torr, B. Ghanem, 和 G. Li, “Crab：面向多模态语言模型代理的跨环境代理基准，”2024。[在线]. 可获取：https://arxiv.org/abs/2407.01511

[462] 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," 2024. [Online]. Available: https://arxiv.org/abs/2406.19263
[462] 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，”2024。[在线]. 可获取：https://arxiv.org/abs/2406.19263

[463] D. Zimmermann and A. Koziolek, "Gui-based software testing: An automated approach using gpt-4 and selenium webdriver," in 2023 38th IEEE/ACM International Conference on Automated Software Engineering Workshops (ASEW). IEEE, 2023, pp. 171-174.
[463] D. Zimmermann 和 A. Koziolek，“基于GUI的软件测试：一种使用GPT-4和Selenium WebDriver的自动化方法”，发表于2023年第38届IEEE/ACM自动化软件工程国际会议研讨会（ASEW）。IEEE，2023，页171-174。

[464] J. Yoon, R. Feldt, and S. Yoo, "Intent-driven mobile gui testing with autonomoufs large language model agents," in 2024 IEEE Conference on Software Testing, Verification and Validation (ICST). IEEE, 2024, pp. 129-139.
[464] J. Yoon、R. Feldt 和 S. Yoo，“基于意图驱动的移动GUI测试，采用自主大型语言模型代理”，发表于2024年IEEE软件测试、验证与确认会议（ICST）。IEEE，2024，页129-139。

[465] Y. Hu, X. Wang, Y. Wang, Y. Zhang, S. Guo, C. Chen, X. Wang, and Y. Zhou, "Auitestagent: Automatic requirements oriented gui function testing," 2024. [Online]. Available: https://arxiv.org/abs/2407.09018
[465] Y. Hu、X. Wang、Y. Wang、Y. Zhang、S. Guo、C. Chen、X. Wang 和 Y. Zhou，“Auitestagent：面向需求的自动化GUI功能测试”，2024年。[在线]。可访问：https://arxiv.org/abs/2407.09018

[466] Z. Liu, C. Li, C. Chen, J. Wang, B. Wu, Y. Wang, J. Hu, and Q. Wang, "Vision-driven automated mobile gui testing via multimodal large language model," 2024. [Online]. Available: https://arxiv.org/abs/2407.03037
[466] Z. Liu、C. Li、C. Chen、J. Wang、B. Wu、Y. Wang、J. Hu 和 Q. Wang，“基于视觉驱动的多模态大型语言模型自动化移动GUI测试”，2024年。[在线]。可访问：https://arxiv.org/abs/2407.03037

[467] M. Taeb, A. Swearngin, E. Schoop, R. Cheng, Y. Jiang, and J. Nichols, "Axnaux: Replaying accessibility tests from natural language," in Proceedings of the CHI Conference on Human Factors in Computing Systems, 2024, pp. 1-16.
[467] M. Taeb、A. Swearngin、E. Schoop、R. Cheng、Y. Jiang 和 J. Nichols，“Axnaux：从自然语言重放无障碍测试”，发表于2024年CHI人机交互大会论文集，页1-16。

[468] C. Cui, T. Li, J. Wang, C. Chen, D. Towey, and R. Huang, "Large language models for mobile gui text input generation: An empirical study," arXiv preprint arXiv:2404.08948, 2024.
[468] C. Cui、T. Li、J. Wang、C. Chen、D. Towey 和 R. Huang，“大型语言模型用于移动GUI文本输入生成：一项实证研究”，arXiv预印本 arXiv:2404.08948，2024年。

[469] Z. Liu, C. Chen, J. Wang, X. Che, Y. Huang, J. Hu, and Q. Wang, "Fill in the blank: Context-aware automated text input generation for mobile gui testing," in 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE). IEEE, 2023, pp. 1355-1367.
[469] Z. Liu、C. Chen、J. Wang、X. Che、Y. Huang、J. Hu 和 Q. Wang，“填空：面向上下文的自动化文本输入生成用于移动GUI测试”，发表于2023年第45届IEEE/ACM国际软件工程会议（ICSE）。IEEE，2023，页1355-1367。

[470] Y. Huang, J. Wang, Z. Liu, Y. Wang, S. Wang, C. Chen, Y. Hu, and Q. Wang, "Crashtranslator: Automatically reproducing mobile application crashes directly from stack trace," in Proceedings of the 46th IEEE/ACM International Conference on Software Engineering, 2024, pp. 1-13.
[470] Y. Huang、J. Wang、Z. Liu、Y. Wang、S. Wang、C. Chen、Y. Hu 和 Q. Wang，“Crashtranslator：基于堆栈跟踪自动重现移动应用崩溃”，发表于2024年第46届IEEE/ACM国际软件工程会议，页1-13。

[471] S. Feng and C. Chen, "Prompting is all you need: Automated android bug replay with large language models," in Proceedings of the 46th IEEE/ACM International Conference on Software Engineering, 2024, pp. 1-13.
[471] S. Feng 和 C. Chen，“提示即所需：利用大型语言模型自动重放Android缺陷”，发表于2024年第46届IEEE/ACM国际软件工程会议，页1-13。

[472] L. Ding, J. Bheemanpally, and Y. Zhang, "Improving technical" how-to" query accuracy with automated search results verification and reranking," arXiv preprint arXiv:2404.08860, 2024.
[472] L. Ding、J. Bheemanpally 和 Y. Zhang，“通过自动搜索结果验证与重排序提升技术性‘操作指南’查询准确性”，arXiv预印本 arXiv:2404.08860，2024年。

[473] B. Beyzaei, S. Talebipour, G. Rafiei, N. Medvidovic, and S. Malek, "Automated test transfer across android apps using large language models," arXiv preprint arXiv:2411.17933, 2024.
[473] B. Beyzaei、S. Talebipour、G. Rafiei、N. Medvidovic 和 S. Malek，“利用大型语言模型实现Android应用间的自动化测试迁移”，arXiv预印本 arXiv:2411.17933，2024年。

[474] Y. Lu, B. Yao, H. Gu, J. Huang, J. Wang, L. Li, J. Gesi, Q. He, T. J.-J. Li, and D. Wang, "Uxagent: An Ilm agent-based usability testing framework for web design," arXiv preprint arXiv:2502.12561, 2025.
[474] Y. Lu、B. Yao、H. Gu、J. Huang、J. Wang、L. Li、J. Gesi、Q. He、T. J.-J. Li 和 D. Wang，“Uxagent：基于ILM代理的网页设计可用性测试框架”，arXiv预印本 arXiv:2502.12561，2025年。

[475] D. Ran, H. Wang, Z. Song, M. Wu, Y. Cao, Y. Zhang, W. Yang, and T. Xie, "Guardian: A runtime framework for Ilm-based ui exploration," in Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis, 2024, pp. 958-970.
[475] D. Ran、H. Wang、Z. Song、M. Wu、Y. Cao、Y. Zhang、W. Yang 和 T. Xie，“Guardian：基于ILM的运行时UI探索框架”，发表于2024年第33届ACM SIGSOFT国际软件测试与分析研讨会，页958-970。

[476] Y. Li, Y. Li, and Y. Yang, "Test-agent: A multimodal app automation testing framework based on the large language model," in 2024 IEEE 4th International Conference on Digital Twins and Parallel Intelligence (DTPI). IEEE, 2024, pp. 609-614.
[476] Y. Li、Y. Li 和 Y. Yang，“Test-agent：基于大型语言模型的多模态应用自动化测试框架”，发表于2024年IEEE第四届数字孪生与并行智能国际会议（DTPI）。IEEE，2024，页609-614。

[477] B. F. Demissie, Y. N. Tun, L. K. Shar, and M. Ceccato, "Vlm-fuzz: Vision language model assisted recursive depth-first search exploration for effective ui testing of android apps," arXiv preprint arXiv:2504.11675, 2025.
[477] B. F. Demissie, Y. N. Tun, L. K. Shar, 和 M. Ceccato, "Vlm-fuzz：基于视觉语言模型辅助的递归深度优先搜索探索，用于安卓应用的高效UI测试," arXiv预印本 arXiv:2504.11675, 2025.

[478] E. Yapağci, Y. A. S. Öztürk, and E. Tüzün, "Bugcraft: End-to-end crash bug reproduction using Ilm agents in minecraft," arXiv preprint arXiv:2503.20036, 2025.
[478] E. Yapağci, Y. A. S. Öztürk, 和 E. Tüzün, "Bugcraft：使用Ilm代理在Minecraft中实现端到端崩溃缺陷复现," arXiv预印本 arXiv:2503.20036, 2025.

[479] X. Li, J. Cao, Y. Liu, S.-C. Cheung, and H. Wang, "Reusedroid: A vlm-empowered android ui test migrator boosted by active feedback," arXiv preprint arXiv:2504.02357, 2025.
[479] X. Li, J. Cao, Y. Liu, S.-C. Cheung, 和 H. Wang, "Reusedroid：一种由视觉语言模型驱动并通过主动反馈增强的安卓UI测试迁移工具," arXiv预印本 arXiv:2504.02357, 2025.

[480] A. Chevrot, A. Vernotte, J.-R. Falleri, X. Blanc, and B. Leg-eard, "Are autonomous web agents good testers?" arXiv preprint arXiv:2504.01495, 2025.
[480] A. Chevrot, A. Vernotte, J.-R. Falleri, X. Blanc, 和 B. Leg-eard, "自主网页代理是优秀的测试者吗？" arXiv预印本 arXiv:2504.01495, 2025.

[481] T. Rosenbach, D. Heidrich, and A. Weinert, "Automated testing of the gui of a real-life engineering software using large language models," in 2025 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW). IEEE, 2025, pp. 103-110.
[481] T. Rosenbach, D. Heidrich, 和 A. Weinert, "利用大型语言模型对真实工程软件的图形用户界面进行自动化测试," 载于2025年IEEE国际软件测试、验证与验证研讨会（ICSTW），IEEE, 2025, 页103-110.

[482] Q. Kong, Z. Lv, Y. Xiong, J. Sun, T. Su, D. Wang, L. Li, X. Yang, and G. Huo, "Prophetagent: Automatically synthesizing gui tests from test cases in natural language for mobile apps."
[482] Q. Kong, Z. Lv, Y. Xiong, J. Sun, T. Su, D. Wang, L. Li, X. Yang, 和 G. Huo, "Prophetagent：自动从自然语言测试用例合成移动应用的GUI测试."

[483] S. Feng, C. Du, H. Liu, Q. Wang, Z. Lv, G. Huo, X. Yang, and C. Chen, "Agent for user: Testing multi-user interactive features in tiktok," arXiv preprint arXiv:2504.15474, 2025.
[483] S. Feng, C. Du, H. Liu, Q. Wang, Z. Lv, G. Huo, X. Yang, 和 C. Chen, "Agent for user：测试抖音中的多用户交互功能," arXiv预印本 arXiv:2504.15474, 2025.

[484] J. Gorniak, Y. Kim, D. Wei, and N. W. Kim, "Vizability: Enhancing chart accessibility with Ilm-based conversational interaction," in Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology, 2024, pp. 1-19.
[484] J. Gorniak, Y. Kim, D. Wei, 和 N. W. Kim, "Vizability：基于Ilm的对话交互提升图表无障碍性," 载于第37届ACM用户界面软件与技术年会论文集, 2024, 页1-19.

[485] Y. Ye, X. Cong, S. Tian, J. Cao, H. Wang, Y. Qin, Y. Lu, H. Yu, H. Wang, Y. Lin et al., "Proagent: From robotic process automation to agentic process automation," arXiv preprint arXiv:2311.10751, 2023.
[485] Y. Ye, X. Cong, S. Tian, J. Cao, H. Wang, Y. Qin, Y. Lu, H. Yu, H. Wang, Y. Lin 等, "Proagent：从机器人流程自动化到智能流程自动化," arXiv预印本 arXiv:2311.10751, 2023.

[486] Y. Guan, D. Wang, Z. Chu, S. Wang, F. Ni, R. Song, and C. Zhuang, "Intelligent agents with llm-based process automation," in Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2024, pp. 5018-5027.
[486] Y. Guan, D. Wang, Z. Chu, S. Wang, F. Ni, R. Song, 和 C. Zhuang, "基于llm的智能代理流程自动化," 载于第30届ACM知识发现与数据挖掘会议论文集, 2024, 页5018-5027.

[487] M. D. Vu, H. Wang, Z. Li, J. Chen, S. Zhao, Z. Xing, and C. Chen, "Gptvoicetasker: Llm-powered virtual assistant for smartphone," arXiv preprint arXiv:2401.14268, 2024.
[487] M. D. Vu, H. Wang, Z. Li, J. Chen, S. Zhao, Z. Xing, 和 C. Chen, "Gptvoicetasker：基于LIm的智能手机虚拟助手," arXiv预印本 arXiv:2401.14268, 2024.

[488] L. Pan, B. Wang, C. Yu, Y. Chen, X. Zhang, and Y. Shi, "Autotask: Executing arbitrary voice commands by exploring and learning from mobile gui," arXiv preprint arXiv:2312.16062, 2023.
[488] L. Pan, B. Wang, C. Yu, Y. Chen, X. Zhang, 和 Y. Shi, "Autotask：通过探索和学习移动GUI执行任意语音命令," arXiv预印本 arXiv:2312.16062, 2023.

[489] D. Gao, S. Hu, Z. Bai, Q. Lin, and M. Z. Shou, "Assisteditor: Multi-agent collaboration for gui workflow automation in video creation," in Proceedings of the 32nd ACM International Conference on Multimedia, 2024, pp. 11 255-11 257.
[489] D. Gao, S. Hu, Z. Bai, Q. Lin, 和 M. Z. Shou, "Assisteditor：用于视频创作中GUI工作流自动化的多代理协作," 载于第32届ACM国际多媒体会议论文集, 2024, 页11255-11257.

[490] T. Huang, C. Yu, W. Shi, Z. Peng, D. Yang, W. Sun, and Y. Shi, "Promptrpa: Generating robotic process automation on smartphones from textual prompts," arXiv preprint arXiv:2404.02475, 2024.
[490] T. Huang, C. Yu, W. Shi, Z. Peng, D. Yang, W. Sun, 和 Y. Shi, "Promptrpa：基于文本提示生成智能手机上的机器人流程自动化," arXiv预印本 arXiv:2404.02475, 2024.

[491] W. Gao, K. Du, Y. Luo, W. Shi, C. Yu, and Y. Shi, "Easyask: An in-app contextual tutorial search assistant for older adults with voice and touch inputs," Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, vol. 8, no. 3, pp. 1-27, 2024.
[491] W. Gao, K. Du, Y. Luo, W. Shi, C. Yu, 和 Y. Shi, “Easyask：一款面向老年人的应用内上下文教程搜索助手，支持语音和触控输入，”《ACM交互式、移动、可穿戴及普适技术会议录》，第8卷，第3期，页码1-27，2024年。

[492] OpenAdapt AI, "OpenAdapt: Open Source Generative Process Automation," 2024, accessed: 2024-10-26. [Online]. Available: https://github.com/OpenAdaptAI/OpenAdapt
[492] OpenAdapt AI，“OpenAdapt：开源生成式流程自动化，”2024年，访问时间：2024-10-26。[在线]。可用地址：https://github.com/OpenAdaptAI/OpenAdapt

[493] AgentSeaf AI. (2024) Introduction to agentsea platform. Accessed: 2024-10-26. [Online]. Available: https://www.agentsea.ai/
[493] AgentSeaf AI。（2024）Agentsea平台介绍。访问时间：2024-10-26。[在线]。可用地址：https://www.agentsea.ai/

[494] O. Interpreter, "Open interpreter: A natural language interface for computers," GitHub repository, 2024, accessed: 2024- 10-27. [Online]. Available: https://github.com/OpenInterpreter/ open-interpreter
[494] O. Interpreter，“Open interpreter：计算机的自然语言接口，”GitHub代码库，2024年，访问时间：2024-10-27。[在线]。可用地址：https://github.com/OpenInterpreter/open-interpreter

[495] MultiOn AI. (2024) Multion ai: Ai agents that act on your behalf. Accessed: 2024-10-26. [Online]. Available: https://www.multion.ai/
[495] MultiOn AI。（2024）Multion AI：代表您行动的AI代理。访问时间：2024-10-26。[在线]。可用地址：https://www.multion.ai/

[496] HONOR, "Honor introduces magicos 9.0," 2024, accessed: 2024-11-16. [Online]. Available: https://www.fonearena.com/blog/438680/honor-magicos-9-0-features.html
[496] HONOR，“荣耀发布MagicOS 9.0，”2024年，访问时间：2024-11-16。[在线]。可用地址：https://www.fonearena.com/blog/438680/honor-magicos-9-0-features.html

[497] T. Srinivasan and S. Patapati, "Webnav: An intelligent agent for voice-controlled web navigation," arXiv preprint arXiv:2503.13843, 2025.
[497] T. Srinivasan 和 S. Patapati，“Webnav：一款智能语音控制网页导航代理，”arXiv预印本 arXiv:2503.13843，2025年。

[498] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma et al., "Scaling instruction-finetuned language models," Journal of Machine Learning Research, vol. 25, no. 70, pp. 1-53, 2024.
[498] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma 等，“指令微调语言模型的扩展，”《机器学习研究杂志》，第25卷，第70期，页码1-53，2024年。

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

[500] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," IEEE Transactions on Computational Intelligence and AI in games, vol. 4, no. 1, pp. 1-43, 2012.
[500] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis 和 S. Colton，“蒙特卡洛树搜索方法综述，”《IEEE计算智能与人工智能游戏汇刊》，第4卷，第1期，页码1-43，2012年。

[501] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. I. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier et al., "Mistral 7b," arXiv preprint arXiv:2310.06825, 2023.
[501] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. I. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier 等，“Mistral 7B，”arXiv预印本 arXiv:2310.06825，2023年。

[502] G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, J. Love et al., "Gemma: Open models based on gemini research and technology," arXiv preprint arXiv:2403.08295, 2024.
[502] G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, J. Love 等，“Gemma：基于Gemini研究与技术的开放模型，”arXiv预印本 arXiv:2403.08295，2024年。

[503] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," Advances in Neural Information Processing Systems, vol. 36, 2024.
[503] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon 和 C. Finn，“直接偏好优化：你的语言模型实际上是一个奖励模型，”《神经信息处理系统进展》，第36卷，2024年。

[504] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi et al., "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," arXiv preprint arXiv:2501.12948, 2025.
[504] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi 等，“DeepSeek-R1：通过强化学习激励大规模语言模型的推理能力，”arXiv预印本 arXiv:2501.12948，2025年。

[505] CogAgent Team, "Cogagent: Cognitive ai agent platform," https: //cogagent.aminer.cn/home| 2024, accessed: 2024-12-17.
[505] CogAgent团队，“Cogagent：认知人工智能代理平台，”https://cogagent.aminer.cn/home| 2024，访问时间：2024-12-17。

[506] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," Advances in Neural Information Processing Systems, vol. 36, 2024.
[506] S. Yao, D. Yu, J. Zhao, I. Shafran, T. 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，”《神经信息处理系统进展》（Advances in Neural Information Processing Systems），第36卷，2024年。

[507] R. Anil, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen, E. Chu, J. H. Clark, L. E. Shafey, Y. Huang, K. Meier-Hellstern, G. Mishra, E. Moreira, M. Omernick, K. Robinson, S. Ruder, Y. Tay, K. Xiao, Y. Xu, Y. Zhang, G. H. Abrego, J. Ahn, J. Austin, P. Barham, J. Botha, J. Bradbury, S. Brahma, K. Brooks, M. Catasta, Y. Cheng, C. Cherry, C. A. Choquette-Choo, A. Chowdhery, C. Crepy, S. Dave, M. Dehghani, S. Dev, J. Devlin, M. Díaz, N. Du, E. Dyer, V. Feinberg, F. Feng, V. Fienber, M. Freitag, X. Garcia, S. Gehrmann, L. Gonzalez, G. Gur-Ari, S. Hand, H. Hashemi, L. Hou, J. Howland, A. Hu, J. Hui, J. Hurwitz, M. Isard, A. Ittycheriah, M. Jagielski, W. Jia, K. Kenealy, M. Krikun, S. Kudugunta, C. Lan, K. Lee, B. Lee, E. Li, M. Li, W. Li, Y. Li, J. Li, H. Lim, H. Lin, Z. Liu, F. Liu, M. Maggioni, A. Mahendru, J. Maynez, V. Misra, M. Moussalem, Z. Nado, J. Nham, E. Ni, A. Nystrom, A. Parrish, M. Pellat, M. Polacek, A. Polozov, R. Pope, S. Qiao, E. Reif, B. Richter, P. Riley, A. C. Ros, A. Roy, B. Saeta, R. Samuel, R. Shelby, A. Slone, D. Smilkov, D. R. So, D. Sohn, S. Tokumine, D. Valter, V. Vasudevan, K. Vodrahalli, X. Wang, P. Wang, Z. Wang, T. Wang, J. Wieting, Y. Wu, K. Xu, Y. Xu, L. Xue, P. Yin, J. Yu, Q. Zhang, S. Zheng, C. Zheng, W. Zhou, D. Zhou, S. Petrov, and Y. Wu, "Palm 2 technical report," 2023. [Online]. Available: https://arxiv.org/abs/2305.10403
[507] R. Anil, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen, E. Chu, J. H. Clark, L. E. Shafey, Y. Huang, K. Meier-Hellstern, G. Mishra, E. Moreira, M. Omernick, K. Robinson, S. Ruder, Y. Tay, K. Xiao, Y. Xu, Y. Zhang, G. H. Abrego, J. Ahn, J. Austin, P. Barham, J. Botha, J. Bradbury, S. Brahma, K. Brooks, M. Catasta, Y. Cheng, C. Cherry, C. A. Choquette-Choo, A. Chowdhery, C. Crepy, S. Dave, M. Dehghani, S. Dev, J. Devlin, M. Díaz, N. Du, E. Dyer, V. Feinberg, F. Feng, V. Fienber, M. Freitag, X. Garcia, S. Gehrmann, L. Gonzalez, G. Gur-Ari, S. Hand, H. Hashemi, L. Hou, J. Howland, A. Hu, J. Hui, J. Hurwitz, M. Isard, A. Ittycheriah, M. Jagielski, W. Jia, K. Kenealy, M. Krikun, S. Kudugunta, C. Lan, K. Lee, B. Lee, E. Li, M. Li, W. Li, Y. Li, J. Li, H. Lim, H. Lin, Z. Liu, F. Liu, M. Maggioni, A. Mahendru, J. Maynez, V. Misra, M. Moussalem, Z. Nado, J. Nham, E. Ni, A. Nystrom, A. Parrish, M. Pellat, M. Polacek, A. Polozov, R. Pope, S. Qiao, E. Reif, B. Richter, P. Riley, A. C. Ros, A. Roy, B. Saeta, R.

Samuel, R. Shelby, A. Slone, D. Smilkov, D. R. So, D. Sohn, S. Tokumine, D. Valter, V. Vasudevan, K. Vodrahalli, X. Wang, P. Wang, Z. Wang, T. Wang, J. Wieting, Y. Wu, K. Xu, Y. Xu, L. Xue, P. Yin, J. Yu, Q. Zhang, S. Zheng, C. Zheng, W. Zhou, D. Zhou, S. Petrov, 和 Y. Wu，“PaLM 2技术报告，”2023年。[在线]。可获取：https://arxiv.org/abs/2305.10403

[508] Baidu Research, "ERNIE Bot: Baidu's Knowledge-Enhanced Large Language Model Built on Full AI Stack Technology," 2024, [Online; accessed 9-November-2024]. [Online]. Available: https://research.baidu.com/Blog/index-view?id=183
[508] 百度研究院，“ERNIE Bot：百度基于全栈AI技术构建的知识增强大型语言模型，”2024年，[在线；访问时间2024年11月9日]。[在线]。可获取：https://research.baidu.com/Blog/index-view?id=183

[509] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark et al., "Learning transferable visual models from natural language supervision," in International conference on machine learning. PMLR, 2021, pp. 8748-8763.
[509] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark 等，“从自然语言监督中学习可迁移的视觉模型，”载于国际机器学习大会（International Conference on Machine Learning），PMLR，2021年，第8748-8763页。

[510] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, I. Stoica, and E. P. Xing, "Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality," March 2023. [Online]. Available: https://lmsys.org/blog/2023-03-30-vicuna/
[510] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, I. Stoica, 和 E. P. Xing，“Vicuna：一个开源聊天机器人，达到90%* ChatGPT质量，令GPT-4印象深刻，”2023年3月。[在线]. 可用: https://lmsys.org/blog/2023-03-30-vicuna/

[511] 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
[511] 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, 和 T. Zhu，“Qwen技术报告，”2023年。[在线]. 可用: https://arxiv.org/abs/2309.16609

[512] 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
[512] 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, 和 A. Kembhavi，“Molmo和Pixmo：面向最先进视觉-语言模型的开放权重和开放数据，”2024年。[在线]. 可用: https://arxiv.org/abs/2409.17146

[513] OpenAI, "Operator system card," Jan. 2025, released on January 23, 2025.
[513] OpenAI，“操作系统卡，”2025年1月，2025年1月23日发布。

[514] B. Xiao, H. Wu, W. Xu, X. Dai, H. Hu, Y. Lu, M. Zeng, C. Liu, and L. Yuan, "Florence-2: Advancing a unified representation for a variety of vision tasks," 2023. [Online]. Available: https://arxiv.org/abs/2311.06242
[514] B. Xiao, H. Wu, W. Xu, X. Dai, H. Hu, Y. Lu, M. Zeng, C. Liu, 和 L. Yuan，“Florence-2：推进多种视觉任务的统一表示，”2023年。[在线]. 可用: https://arxiv.org/abs/2311.06242

[515] W. Li, W. Bishop, A. Li, C. Rawles, F. Campbell-Ajala, D. Tyama-gundlu, and O. Riva, "On the effects of data scale on computer control agents," arXiv preprint arXiv:2406.03679, 2024.
[515] W. Li, W. Bishop, A. Li, C. Rawles, F. Campbell-Ajala, D. Tyama-gundlu, 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年。

[516] D. Chen, Y. Huang, Z. Ma, H. Chen, X. Pan, C. Ge, D. Gao, Y. Xie, Z. Liu, J. Gao et al., "Data-juicer: A one-stop data processing system for large language models," in Companion of the 2024 International Conference on Management of Data, 2024, pp. 120- 134.
[516] D. Chen, Y. Huang, Z. Ma, H. Chen, X. Pan, C. Ge, D. Gao, Y. Xie, Z. Liu, J. Gao 等，“Data-juicer：面向大型语言模型的一站式数据处理系统，”载于2024年国际数据管理会议伴刊, 2024年, 第120-134页。

[517] B. Ding, C. Qin, R. Zhao, T. Luo, X. Li, G. Chen, W. Xia, J. Hu, L. A. Tuan, and S. Joty, "Data augmentation using llms: Data perspectives, learning paradigms and challenges," in Findings of the Association for Computational Linguistics ACL 2024, 2024, pp. 1679-1705.
[517] B. Ding, C. Qin, R. Zhao, T. Luo, X. Li, G. Chen, W. Xia, J. Hu, L. A. Tuan, 和 S. Joty, “使用llms进行数据增强：数据视角、学习范式与挑战，”发表于2024年计算语言学协会（ACL）研究成果，2024年，第1679-1705页。

[518] Z. Tan, D. Li, S. Wang, A. Beigi, B. Jiang, A. Bhattacharjee, M. Karami, J. Li, L. Cheng, and H. Liu, "Large language models for data annotation: A survey," arXiv preprint arXiv:2402.13446, 2024.
[518] Z. Tan, D. Li, S. Wang, A. Beigi, B. Jiang, A. Bhattacharjee, M. Karami, J. Li, L. Cheng, 和 H. Liu, “用于数据标注的大型语言模型综述，”arXiv预印本 arXiv:2402.13446，2024年。

[519] J. Andreas, J. Bufe, D. Burkett, C. Chen, J. Clausman, J. Crawford, K. Crim, J. DeLoach, L. Dorner, J. Eisner et al., "Task-oriented dialogue as dataflow synthesis," Transactions of the Association for Computational Linguistics, vol. 8, pp. 556-571, 2020.
[519] J. Andreas, J. Bufe, D. Burkett, C. Chen, J. Clausman, J. Crawford, K. Crim, J. DeLoach, L. Dorner, J. Eisner 等，“面向任务的对话作为数据流合成，”计算语言学协会汇刊，卷8，第556-571页，2020年。

[520] Z. Guo, S. Cheng, H. Wang, S. Liang, Y. Qin, P. Li, Z. Liu, M. Sun, and Y. Liu, "Stabletoolbench: Towards stable large-scale benchmarking on tool learning of large language models," 2024.
[520] Z. Guo, S. Cheng, H. Wang, S. Liang, Y. Qin, P. Li, Z. Liu, M. Sun, 和 Y. Liu, “Stabletoolbench：迈向大型语言模型工具学习的稳定大规模基准测试，”2024年。

[521] C. Ma, J. Zhang, Z. Zhu, C. Yang, Y. Yang, Y. Jin, Z. Lan, L. Kong, and J. He, "Agentboard: An analytical evaluation board of multi-turn llm agents," arXiv preprint arXiv:2401.13178, 2024.
[521] C. Ma, J. Zhang, Z. Zhu, C. Yang, Y. Yang, Y. Jin, Z. Lan, L. Kong, 和 J. He, “Agentboard：多轮llm代理的分析评估板，”arXiv预印本 arXiv:2401.13178，2024年。

[522] 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
[522] 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, “一张图像相当于16x16个词：大规模图像识别的Transformer，”2021年。[在线]。可访问：https://arxiv.org/abs/2010.11929

[523] H. Laurengon, L. Tronchon, M. Cord, and V. Sanh, "What matters when building vision-language models?" arXiv preprint arXiv:2405.02246, 2024.
[523] H. Laurengon, L. Tronchon, M. Cord, 和 V. Sanh, “构建视觉-语言模型时的关键因素？”arXiv预印本 arXiv:2405.02246，2024年。

[524] Z. Du, Y. Qian, X. Liu, M. Ding, J. Qiu, Z. Yang, and J. Tang, "Glm: General language model pretraining with autoregressive blank infilling," arXiv preprint arXiv:2103.10360, 2021.
[524] Z. Du, Y. Qian, X. Liu, M. Ding, J. Qiu, Z. Yang, 和 J. Tang, “Glm：基于自回归空白填充的通用语言模型预训练，”arXiv预印本 arXiv:2103.10360，2021年。

[525] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in Proceedings of the IEEE/CVF international conference on computer vision, 2021, pp. 10012-10022.
[525] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, 和 B. Guo, “Swin Transformer：基于移位窗口的分层视觉Transformer，”发表于IEEE/CVF国际计算机视觉会议，2021年，第10012-10022页。

[526] B. Rozière, J. Gehring, F. Gloeckle, S. Sootla, I. Gat, X. E. Tan, Y. Adi, J. Liu, R. Sauvestre, T. Remez, J. Rapin, A. Kozhevnikov, I. Evtimov, J. Bitton, M. Bhatt, C. C. Ferrer, A. Grattafiori, W. Xiong, A. Défossez, J. Copet, F. Azhar, H. Touvron, L. Martin, N. Usunier, T. Scialom, and G. Synnaeve, "Code llama: Open foundation models for code," 2024. [Online]. Available: https://arxiv.org/abs/2308.12950
[526] B. Rozière, J. Gehring, F. Gloeckle, S. Sootla, I. Gat, X. E. Tan, Y. Adi, J. Liu, R. Sauvestre, T. Remez, J. Rapin, A. Kozhevnikov, I. Evtimov, J. Bitton, M. Bhatt, C. C. Ferrer, A. Grattafiori, W. Xiong, A. Défossez, J. Copet, F. Azhar, H. Touvron, L. Martin, N. Usunier, T. Scialom, 和 G. Synnaeve, “Code llama：开源代码基础模型，”2024年。[在线]。可访问：https://arxiv.org/abs/2308.12950

[527] W. Cai, J. Jiang, F. Wang, J. Tang, S. Kim, and J. Huang, "A survey on mixture of experts," arXiv preprint arXiv:2407.06204, 2024.
[527] W. Cai, J. Jiang, F. Wang, J. Tang, S. Kim, 和 J. Huang, “专家混合模型综述，”arXiv预印本 arXiv:2407.06204，2024年。

[528] I. Turc, M.-W. Chang, K. Lee, and K. Toutanova, "Well-read students learn better: On the importance of pre-training compact models," arXiv preprint arXiv:1908.08962, 2019.
[528] I. Turc, M.-W. Chang, K. Lee, 和 K. Toutanova, “博学的学生学得更好：关于紧凑模型预训练重要性的研究，”arXiv预印本 arXiv:1908.08962，2019年。

[529] L. Beyer, A. Steiner, A. S. Pinto, A. Kolesnikov, X. Wang, D. Salz, M. Neumann, I. Alabdulmohsin, M. Tschannen, E. Bugliarello et al., "Paligemma: A versatile 3b vlm for transfer," arXiv preprint arXiv:2407.07726, 2024.
[529] L. Beyer, A. Steiner, A. S. Pinto, A. Kolesnikov, X. Wang, D. Salz, M. Neumann, I. Alabdulmohsin, M. Tschannen, E. Bugliarello 等, "Paligemma: 一种多功能的3b视觉语言模型（vlm）用于迁移," arXiv预印本 arXiv:2407.07726, 2024.

[530] Z. Cai, M. Cao, H. Chen, K. Chen, K. Chen, X. Chen, X. Chen, Z. Chen, Z. Chen, P. Chu et al., "Internlm2 technical report," arXiv preprint arXiv:2403.17297, 2024.
[530] Z. Cai, M. Cao, H. Chen, K. Chen, K. Chen, X. Chen, X. Chen, Z. Chen, Z. Chen, P. Chu 等, "Internlm2技术报告," arXiv预印本 arXiv:2403.17297, 2024.

[531] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, pp. 10684-10695.
[531] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, 和 B. Ommer, "基于潜在扩散模型的高分辨率图像合成," 载于IEEE/CVF计算机视觉与模式识别会议论文集, 2022, 页10684-10695.

[532] Y.-C. Hsiao, F. Zubach, G. Baechler, V. Carbune, J. Lin, M. Wang, S. Sunkara, Y. Zhu, and J. Chen, "Screenga: Large-scale question-answer pairs over mobile app screenshots," 2024. [Online]. Available: https://arxiv.org/abs/2209.08199
[532] Y.-C. Hsiao, F. Zubach, G. Baechler, V. Carbune, J. Lin, M. Wang, S. Sunkara, Y. Zhu, 和 J. Chen, "Screenga: 大规模移动应用截图问答对," 2024. [在线]. 可获取：https://arxiv.org/abs/2209.08199

[533] A. Jaech, A. Kalai, A. Lerer, A. Richardson, A. El-Kishky, A. Low, A. Helyar, A. Madry, A. Beutel, A. Carney et al., "Openai o1 system card," arXiv preprint arXiv:2412.16720, 2024.
[533] A. Jaech, A. Kalai, A. Lerer, A. Richardson, A. El-Kishky, A. Low, A. Helyar, A. Madry, A. Beutel, A. Carney 等, "Openai o1系统卡," arXiv预印本 arXiv:2412.16720, 2024.

[534] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu et al., "Deepseekmath: Pushing the limits of mathematical reasoning in open language models," arXiv preprint arXiv:2402.03300, 2024.
[534] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu 等, "Deepseekmath: 推动开放语言模型中数学推理的极限," arXiv预印本 arXiv:2402.03300, 2024.

[535] MosaicML, "Mosaicml: Mpt-7b," 2023, accessed: 2024-11-19. [Online]. Available: https://www.mosaicml.com/blog/mpt-7b
[535] MosaicML, "Mosaicml: Mpt-7b," 2023, 访问时间: 2024-11-19. [在线]. 可获取：https://www.mosaicml.com/blog/mpt-7b

[536] X. Chen, X. Wang, L. Beyer, A. Kolesnikov, J. Wu, P. Voigtlaender, B. Mustafa, S. Goodman, I. Alabdulmohsin, P. Padlewski et al., "Pali-3 vision language models: Smaller, faster, stronger," arXiv preprint arXiv:2310.09199, 2023.
[536] X. Chen, X. Wang, L. Beyer, A. Kolesnikov, J. Wu, P. Voigtlaender, B. Mustafa, S. Goodman, I. Alabdulmohsin, P. Padlewski 等, "Pali-3视觉语言模型：更小、更快、更强," arXiv预印本 arXiv:2310.09199, 2023.

[537] Q. Sun, Y. Fang, L. Wu, X. Wang, and Y. Cao, "Eva-clip: Improved training techniques for clip at scale," arXiv preprint arXiv:2303.15389, 2023.
[537] Q. Sun, Y. Fang, L. Wu, X. Wang, 和 Y. Cao, "Eva-clip: 大规模clip改进训练技术," arXiv预印本 arXiv:2303.15389, 2023.

[538] D. Guo, Q. Zhu, D. Yang, Z. Xie, K. Dong, W. Zhang, G. Chen, X. Bi, Y. Wu, Y. Li et al., "Deepseek-coder: When the large language model meets programming-the rise of code intelligence," arXiv preprint arXiv:2401.14196, 2024.
[538] D. Guo, Q. Zhu, D. Yang, Z. Xie, K. Dong, W. Zhang, G. Chen, X. Bi, Y. Wu, Y. Li 等, "Deepseek-coder: 大型语言模型遇上编程——代码智能的崛起," arXiv预印本 arXiv:2401.14196, 2024.

[539] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, pp. 11976- 11986.
[539] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, 和 S. Xie, "面向2020年代的卷积网络," 载于IEEE/CVF计算机视觉与模式识别会议论文集, 2022, 页11976-11986.

[540] S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang et al., "Qwen2. 5-vl technical report," arXiv preprint arXiv:2502.13923, 2025.
[540] S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang 等, "Qwen2.5-vl技术报告," arXiv预印本 arXiv:2502.13923, 2025.

[541] X. Liu, H. Yu, H. Zhang, Y. Xu, X. Lei, H. Lai, Y. Gu, H. Ding, K. Men, K. Yang et al., "Agentbench: Evaluating llms as agents," arXiv preprint arXiv:2308.03688, 2023.
[541] X. Liu, H. Yu, H. Zhang, Y. Xu, X. Lei, H. Lai, Y. Gu, H. Ding, K. Men, K. Yang 等, "Agentbench: 评估大规模语言模型（llms）作为代理," arXiv预印本 arXiv:2308.03688, 2023.

[542] D. Zimmermann and A. Koziolek, "Automating gui-based software testing with gpt-3," in 2023 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), 2023, pp. 62-65.
[542] D. Zimmermann 和 A. Koziolek, "基于 GPT-3 的图形用户界面软件测试自动化," 载于 2023 年 IEEE 软件测试、验证与确认国际会议研讨会 (ICSTW), 2023, 页 62-65.

[543] J. Wang, Y. Huang, C. Chen, Z. Liu, S. Wang, and Q. Wang, "Software testing with large language models: Survey, landscape, and vision," IEEE Transactions on Software Engineering, 2024.
[543] J. Wang, Y. Huang, C. Chen, Z. Liu, S. Wang 和 Q. Wang, "利用大型语言模型进行软件测试：综述、现状与展望," IEEE 软件工程汇刊, 2024.

[544] K. Q. Lin, P. Zhang, J. Chen, S. Pramanick, D. Gao, A. J. Wang, R. Yan, and M. Z. Shou, "Univtg: Towards unified video-language temporal grounding," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 2794-2804.
[544] K. Q. Lin, P. Zhang, J. Chen, S. Pramanick, D. Gao, A. J. Wang, R. Yan 和 M. Z. Shou, "Univtg：迈向统一的视频-语言时间定位," 载于 IEEE/CVF 国际计算机视觉会议论文集, 2023, 页 2794-2804.

[545] F. Al, "Eko - build production-ready agentic workflow with natural language," https://eko.fellou.ai/ 2025, accessed: 2025-01-15.
[545] F. Al, "Eko - 使用自然语言构建生产就绪的自主工作流," https://eko.fellou.ai/ 2025, 访问时间：2025-01-15.

[546] Z. Liao, L. Mo, C. Xu, M. Kang, J. Zhang, C. Xiao, Y. Tian, B. Li, and H. Sun, "Eia: Environmental injection attack on generalist web agents for privacy leakage," arXiv preprint arXiv:2409.11295, 2024.
[546] Z. Liao, L. Mo, C. Xu, M. Kang, J. Zhang, C. Xiao, Y. Tian, B. Li 和 H. Sun, "Eia：针对通用网络代理的环境注入攻击导致隐私泄露," arXiv 预印本 arXiv:2409.11295, 2024.

[547] F. He, T. Zhu, D. Ye, B. Liu, W. Zhou, and P. S. Yu, "The emerged security and privacy of llm agent: A survey with case studies," arXiv preprint arXiv:2407.19354, 2024.
[547] F. He, T. Zhu, D. Ye, B. Liu, W. Zhou 和 P. S. Yu, "llm 代理的安全与隐私新兴问题：带案例研究的综述," arXiv 预印本 arXiv:2407.19354, 2024.

[548] Y. Gan, Y. Yang, Z. Ma, P. He, R. Zeng, Y. Wang, Q. Li, C. Zhou, S. Li, T. Wang, Y. Gao, Y. Wu, and S. Ji, "Navigating the risks: A survey of security, privacy, and ethics threats in llm-based agents," 2024. [Online]. Available: https://arxiv.org/abs/2411.09523
[548] Y. Gan, Y. Yang, Z. Ma, P. He, R. Zeng, Y. Wang, Q. Li, C. Zhou, S. Li, T. Wang, Y. Gao, Y. Wu 和 S. Ji, "风险导航：llm 基代理中的安全、隐私与伦理威胁综述," 2024. [在线]. 可访问：https://arxiv.org/abs/2411.09523

[549] Y. Yang, X. Yang, S. Li, C. Lin, Z. Zhao, C. Shen, and T. Zhang, "Security matrix for multimodal agents on mobile devices: A systematic and proof of concept study," arXiv preprint arXiv:2407.09295, 2024.
[549] Y. Yang, X. Yang, S. Li, C. Lin, Z. Zhao, C. Shen 和 T. Zhang, "移动设备多模态代理的安全矩阵：系统性研究与概念验证," arXiv 预印本 arXiv:2407.09295, 2024.

[550] X. Zhang, H. Xu, Z. Ba, Z. Wang, Y. Hong, J. Liu, Z. Qin, and K. Ren, "Privacyasst: Safeguarding user privacy in tool-using large language model agents," IEEE Transactions on Dependable and Secure Computing, 2024.
[550] X. Zhang, H. Xu, Z. Ba, Z. Wang, Y. Hong, J. Liu, Z. Qin 和 K. Ren, "Privacyasst：保护使用工具的大型语言模型代理中的用户隐私," IEEE 可靠与安全计算汇刊, 2024.

[551] J. Xu, Z. Li, W. Chen, Q. Wang, X. Gao, Q. Cai, and Z. Ling, "On-device language models: A comprehensive review," arXiv preprint arXiv:2409.00088, 2024.
[551] J. Xu, Z. Li, W. Chen, Q. Wang, X. Gao, Q. Cai 和 Z. Ling, "设备端语言模型：全面综述," arXiv 预印本 arXiv:2409.00088, 2024.

[552] G. Qu, Q. Chen, W. Wei, Z. Lin, X. Chen, and K. Huang, "Mobile edge intelligence for large language models: A contemporary survey," arXiv preprint arXiv:2407.18921, 2024.
[552] G. Qu, Q. Chen, W. Wei, Z. Lin, X. Chen 和 K. Huang, "面向大型语言模型的移动边缘智能：当代综述," arXiv 预印本 arXiv:2407.18921, 2024.

[553] J. Lin, J. Tang, H. Tang, S. Yang, W.-M. Chen, W.-C. Wang, G. Xiao, X. Dang, C. Gan, and S. Han, "Awq: Activation-aware weight quantization for on-device llm compression and acceleration," Proceedings of Machine Learning and Systems, vol. 6, pp. 87-100, 2024.
[553] J. Lin, J. Tang, H. Tang, S. Yang, W.-M. Chen, W.-C. Wang, G. Xiao, X. Dang, C. Gan 和 S. Han, "Awq：面向设备端 llm 压缩与加速的激活感知权重量化," 机器学习与系统会议论文集, 卷 6, 页 87-100, 2024.

[554] Z. Liu, C. Zhao, F. Iandola, C. Lai, Y. Tian, I. Fedorov, Y. Xiong, E. Chang, Y. Shi, R. Krishnamoorthi et al., "Mobilellm: Optimizing sub-billion parameter language models for on-device use cases," arXiv preprint arXiv:2402.14905, 2024.
[554] Z. Liu, C. Zhao, F. Iandola, C. Lai, Y. Tian, I. Fedorov, Y. Xiong, E. Chang, Y. Shi, R. Krishnamoorthi 等, "Mobilellm：优化亚十亿参数语言模型以适应设备端应用," arXiv 预印本 arXiv:2402.14905, 2024.

[555] Z. Zhou, X. Ning, K. Hong, T. Fu, J. Xu, S. Li, Y. Lou, L. Wang, Z. Yuan, X. Li et al., "A survey on efficient inference for large language models," arXiv preprint arXiv:2404.14294, 2024.
[555] Z. Zhou, X. Ning, K. Hong, T. Fu, J. Xu, S. Li, Y. Lou, L. Wang, Z. Yuan, X. Li 等, "大型语言模型高效推理综述," arXiv 预印本 arXiv:2404.14294, 2024.

[556] W. Kuang, B. Qian, Z. Li, D. Chen, D. Gao, X. Pan, Y. Xie, Y. Li, B. Ding, and J. Zhou, "Federatedscope-llm: A comprehensive package for fine-tuning large language models in federated learning," in Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2024, pp. 5260-5271.
[556] W. Kuang, B. Qian, Z. Li, D. Chen, D. Gao, X. Pan, Y. Xie, Y. Li, B. Ding, 和 J. Zhou, “Federatedscope-llm：联邦学习中大规模语言模型微调的综合工具包，”发表于第30届ACM SIGKDD知识发现与数据挖掘大会论文集，2024年，第5260-5271页。

[557] P. Mai, R. Yan, Z. Huang, Y. Yang, and Y. Pang, "Split-and-denoise: Protect large language model inference with local differential privacy," arXiv preprint arXiv:2310.09130, 2023.
[557] P. Mai, R. Yan, Z. Huang, Y. Yang, 和 Y. Pang, “分割与去噪：利用本地差分隐私保护大规模语言模型推理，”arXiv预印本 arXiv:2310.09130，2023年。

[558] L. de Castro, A. Polychroniadou, and D. Escudero, "Privacy-preserving large language model inference via gpu-accelerated fully homomorphic encryption," in Neurips Safe Generative AI Workshop 2024.
[558] L. de Castro, A. Polychroniadou, 和 D. Escudero, “通过GPU加速的全同态加密实现隐私保护的大规模语言模型推理，”发表于Neurips安全生成式人工智能研讨会2024。

[559] J. Wolff, W. Lehr, and C. S. Yoo, "Lessons from gdpr for ai policymaking," Virginia Journal of Law & Technology, vol. 27, no. 4, p. 2, 2024.
[559] J. Wolff, W. Lehr, 和 C. S. Yoo, “从GDPR对人工智能政策制定的启示，”《弗吉尼亚法律与技术杂志》，第27卷，第4期，第2页，2024年。

[560] Z. Zhang, M. Jia, H.-P. Lee, B. Yao, S. Das, A. Lerner, D. Wang, and T. Li, ""it's a fair game", or is it? examining how users navigate disclosure risks and benefits when using llm-based conversational agents," in Proceedings of the CHI Conference on Human Factors in Computing Systems, 2024, pp. 1-26.
[560] Z. Zhang, M. Jia, H.-P. Lee, B. Yao, S. Das, A. Lerner, D. Wang, 和 T. Li, “‘这是公平的游戏’，真的是吗？探讨用户在使用基于llm的对话代理时如何权衡披露风险与收益，”发表于CHI人机交互大会论文集，2024年，第1-26页。

[561] B. Li, Y. Jiang, V. Gadepally, and D. Tiwari, "Llm inference serving: Survey of recent advances and opportunities," arXiv preprint arXiv:2407.12391, 2024.
[561] B. Li, Y. Jiang, V. Gadepally, 和 D. Tiwari, “大规模语言模型推理服务：近期进展与机遇综述，”arXiv预印本 arXiv:2407.12391，2024年。

[562] M. Xu, W. Yin, D. Cai, R. Yi, D. Xu, Q. Wang, B. Wu, Y. Zhao, C. Yang, S. Wang et al., "A survey of resource-efficient llm and multimodal foundation models," arXiv preprint arXiv:2401.08092, 2024.
[562] M. Xu, W. Yin, D. Cai, R. Yi, D. Xu, Q. Wang, B. Wu, Y. Zhao, C. Yang, S. Wang 等, “资源高效的大规模语言模型及多模态基础模型综述，”arXiv预印本 arXiv:2401.08092，2024年。

[563] D. Chen, Y. Liu, M. Zhou, Y. Zhao, H. Wang, S. Wang, X. Chen, T. F. Bissyandé, J. Klein, and L. Li, "Llm for mobile: An initial roadmap," arXiv preprint arXiv:2407.06573, 2024.
[563] D. Chen, Y. Liu, M. Zhou, Y. Zhao, H. Wang, S. Wang, X. Chen, T. F. Bissyandé, J. Klein, 和 L. Li, “面向移动端的LIm：初步路线图，”arXiv预印本 arXiv:2407.06573，2024年。

[564] L. Krupp, D. Geißler, P. Lukowicz, and J. Karolus, "Towards sustainable web agents: A plea for transparency and dedicated metrics for energy consumption," arXiv preprint arXiv:2502.17903, 2025.
[564] L. Krupp, D. Geißler, P. Lukowicz, 和 J. Karolus, “迈向可持续的网络代理：呼吁透明度与专门的能耗指标，”arXiv预印本 arXiv:2502.17903，2025年。

[565] Z. Wan, X. Wang, C. Liu, S. Alam, Y. Zheng, J. Liu, Z. Qu, S. Yan, Y. Zhu, Q. Zhang et al., "Efficient large language models: A survey," arXiv preprint arXiv:2312.03863, 2023.
[565] Z. Wan, X. Wang, C. Liu, S. Alam, Y. Zheng, J. Liu, Z. Qu, S. Yan, Y. Zhu, Q. Zhang 等, “高效大规模语言模型综述，”arXiv预印本 arXiv:2312.03863，2023年。

[566] X. Xu, M. Li, C. Tao, T. Shen, R. Cheng, J. Li, C. Xu, D. Tao, and T. Zhou, "A survey on knowledge distillation of large language models," arXiv preprint arXiv:2402.13116, 2024.
[566] X. Xu, M. Li, C. Tao, T. Shen, R. Cheng, J. Li, C. Xu, D. Tao, 和 T. Zhou, “大规模语言模型知识蒸馏综述，”arXiv预印本 arXiv:2402.13116，2024年。

[567] C. Kachris, "A survey on hardware accelerators for large language models," arXiv preprint arXiv:2401.09890, 2024.
[567] C. Kachris, “大规模语言模型硬件加速器综述，”arXiv预印本 arXiv:2401.09890，2024年。

[568] W. Lee, J. Lee, J. Seo, and J. Sim, "{InfiniGen}: Efficient generative inference of large language models with dynamic {KV} cache management," in 18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24), 2024, pp. 155-172.
[568] W. Lee, J. Lee, J. Seo, 和 J. Sim, “{InfiniGen}：基于动态{KV}缓存管理的大规模语言模型高效生成推理，”发表于第18届USENIX操作系统设计与实现研讨会（OSDI 24），2024年，第155-172页。

[569] Z. Wang, J. Wohlwend, and T. Lei, "Structured pruning of large language models," arXiv preprint arXiv:1910.04732, 2019.
[569] Z. Wang, J. Wohlwend, 和 T. Lei, “大规模语言模型的结构化剪枝，”arXiv预印本 arXiv:1910.04732，2019年。

[570] B. Wu, Y. Zhong, Z. Zhang, G. Huang, X. Liu, and X. Jin, "Fast distributed inference serving for large language models," arXiv preprint arXiv:2305.05920, 2023.
[570] B. Wu, Y. Zhong, Z. Zhang, G. Huang, X. Liu, 和 X. Jin, “大规模语言模型的快速分布式推理服务，”arXiv预印本 arXiv:2305.05920，2023年。

[571] U. Anwar, A. Saparov, J. Rando, D. Paleka, M. Turpin, P. Hase, E. S. Lubana, E. Jenner, S. Casper, O. Sourbut et al., "Foundational challenges in assuring alignment and safety of large language models," arXiv preprint arXiv:2404.09932, 2024.
[571] U. Anwar, A. Saparov, J. Rando, D. Paleka, M. Turpin, P. Hase, E. S. Lubana, E. Jenner, S. Casper, O. Sourbut 等，“确保大型语言模型对齐与安全的基础性挑战”，arXiv 预印本 arXiv:2404.09932，2024年。

[572] L. Zhong and Z. Wang, "A study on robustness and reliability of large language model code generation," arXiv preprint arXiv:2308.10335, 2023.
[572] L. Zhong 和 Z. Wang，“大型语言模型代码生成的鲁棒性与可靠性研究”，arXiv 预印本 arXiv:2308.10335，2023年。

[573] T. Yuan, Z. He, L. Dong, Y. Wang, R. Zhao, T. Xia, L. Xu, B. Zhou, F. Li, Z. Zhang et al., "R-judge: Benchmarking safety risk awareness for Ilm agents," arXiv preprint arXiv:2401.10019, 2024.
[573] T. Yuan, Z. He, L. Dong, Y. Wang, R. Zhao, T. Xia, L. Xu, B. Zhou, F. Li, Z. Zhang 等，“R-judge：用于Ilm代理安全风险意识的基准测试”，arXiv 预印本 arXiv:2401.10019，2024年。

[574] L. Zhang, Q. Jin, H. Huang, D. Zhang, and F. Wei, "Respond in my language: Mitigating language inconsistency in response generation based on large language models," in Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2024, pp. 4177-4192.
[574] L. Zhang, Q. Jin, H. Huang, D. Zhang 和 F. Wei，“用我的语言回应：基于大型语言模型缓解响应生成中的语言不一致性”，载于第62届计算语言学协会年会论文集（第一卷：长文），2024年，第4177-4192页。

[575] H. Zhao, T. Chen, and Z. Wang, "On the robustness of gui grounding models against image attacks," arXiv preprint arXiv:2504.04716, 2025.
[575] H. Zhao, T. Chen 和 Z. Wang，“GUI定位模型对图像攻击的鲁棒性研究”，arXiv 预印本 arXiv:2504.04716，2025年。

[576] Y. Zhang, Y. Li, L. Cui, D. Cai, L. Liu, T. Fu, X. Huang, E. Zhao, Y. Zhang, Y. Chen, L. Wang, A. T. Luu, W. Bi, F. Shi, and S. Shi, "Siren's song in the ai ocean: A survey on hallucination in large language models," 2023. [Online]. Available: https://arxiv.org/abs/2309.01219
[576] Y. Zhang, Y. Li, L. Cui, D. Cai, L. Liu, T. Fu, X. Huang, E. Zhao, Y. Zhang, Y. Chen, L. Wang, A. T. Luu, W. Bi, F. Shi 和 S. Shi，“AI海洋中的塞壬之歌：大型语言模型幻觉现象综述”，2023年。[在线]。可访问：https://arxiv.org/abs/2309.01219

[577] J. Y. F. Chiang, S. Lee, J.-B. Huang, F. Huang, and Y. Chen, "Why are web ai agents more vulnerable than standalone Ilms? a security analysis," arXiv preprint arXiv:2502.20383, 2025.
[577] J. Y. F. Chiang, S. Lee, J.-B. Huang, F. Huang 和 Y. Chen，“为何网络AI代理比独立Ilm更易受攻击？安全性分析”，arXiv 预印本 arXiv:2502.20383，2025年。

[578] C. Chen, Z. Zhang, B. Guo, S. Ma, I. Khalilov, S. A. Gebreegziabher, Y. Ye, Z. Xiao, Y. Yao, T. Li et al., "The obvious invisible threat: LIm-powered gui agents' vulnerability to fine-print injections," arXiv preprint arXiv:2504.11281, 2025.
[578] C. Chen, Z. Zhang, B. Guo, S. Ma, I. Khalilov, S. A. Gebreegziabher, Y. Ye, Z. Xiao, Y. Yao, T. Li 等，“显而易见的隐形威胁：LIm驱动GUI代理对细则注入的脆弱性”，arXiv 预印本 arXiv:2504.11281，2025年。

[579] C. Xu, M. Kang, J. Zhang, Z. Liao, L. Mo, M. Yuan, H. Sun, and B. Li, "Advweb: Controllable black-box attacks on vlm-powered web agents," arXiv preprint arXiv:2410.17401, 2024.
[579] C. Xu, M. Kang, J. Zhang, Z. Liao, L. Mo, M. Yuan, H. Sun 和 B. Li，“Advweb：对基于VLM的网络代理的可控黑盒攻击”，arXiv 预印本 arXiv:2410.17401，2024年。

[580] Y. Yang, X. Yang, S. Li, C. Lin, Z. Zhao, C. Shen, and T. Zhang, "Systematic categorization, construction and evaluation of new attacks against multi-modal mobile gui agents," 2025. [Online]. Available: https://arxiv.org/abs/2407.09295
[580] Y. Yang, X. Yang, S. Li, C. Lin, Z. Zhao, C. Shen 和 T. Zhang，“多模态移动GUI代理新型攻击的系统分类、构建与评估”，2025年。[在线]。可访问：https://arxiv.org/abs/2407.09295

[581] L. Aichberger, A. Paren, Y. Gal, P. Torr, and A. Bibi, "Attacking multimodal os agents with malicious image patches," in ICLR 2025 Workshop on Foundation Models in the Wild.
[581] L. Aichberger, A. Paren, Y. Gal, P. Torr 和 A. Bibi，“利用恶意图像补丁攻击多模态操作系统代理”，发表于ICLR 2025年“野外基础模型”研讨会。

[582] J. Lee, D. Lee, C. Choi, Y. Im, J. Wi, K. Heo, S. Oh, S. Lee, and I. Shin, "Safeguarding mobile gui agent via logic-based action verification," arXiv preprint arXiv:2503.18492, 2025.
[582] J. Lee, D. Lee, C. Choi, Y. Im, J. Wi, K. Heo, S. Oh, S. Lee 和 I. Shin，“通过基于逻辑的动作验证保障移动GUI代理安全”，arXiv 预印本 arXiv:2503.18492，2025年。

[583] L. Pan, M. S. Saxon, W. Xu, D. Nathani, X. Wang, and W. Y. Wang, "Automatically correcting large language models: Surveying the landscape of diverse self-correction strategies," ArXiv, vol. abs/2308.03188, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:260682695
[583] L. Pan, M. S. Saxon, W. Xu, D. Nathani, X. Wang 和 W. Y. Wang，“自动纠正大型语言模型：多样化自我纠正策略综述”，ArXiv，卷abs/2308.03188，2023年。[在线]。可访问：https://api.semanticscholar.org/CorpusID:260682695

[584] X. Huang, W. Ruan, W. Huang, G. Jin, Y. Dong, C. Wu, S. Bensalem, R. Mu, Y. Qi, X. Zhao, K. Cai, Y. Zhang, S. Wu, P. Xu, D. Wu, A. Freitas, and M. A. Mustafa, "A survey of safety and trustworthiness of large language models through the lens of verification and validation," Artif. Intell. Rev., vol. 57, p. 175, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:258823083
[584] X. Huang, W. Ruan, W. Huang, G. Jin, Y. Dong, C. Wu, S. Bensalem, R. Mu, Y. Qi, X. Zhao, K. Cai, Y. Zhang, S. Wu, P. Xu, D. Wu, A. Freitas, 和 M. A. Mustafa, “通过验证与确认视角对大型语言模型安全性与可信性的综述,” 人工智能评论, 第57卷, 第175页, 2023年. [在线]. 可用: https://api.semanticscholar.org/CorpusID:258823083

[585] S. Jha, S. K. Jha, P. Lincoln, N. D. Bastian, A. Velasquez, and S. Neema, "Dehallucinating large language models using formal methods guided iterative prompting," 2023 IEEE International Conference on Assured Autonomy (ICAA), pp. 149-152, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID: 260810131
[585] S. Jha, S. K. Jha, P. Lincoln, N. D. Bastian, A. Velasquez, 和 S. Neema, “利用形式方法引导的迭代提示消除大型语言模型幻觉,” 2023年IEEE保障自主国际会议(ICAA), 第149-152页, 2023年. [在线]. 可用: https://api.semanticscholar.org/CorpusID: 260810131

[586] Q. Zhang, T. Zhang, J. Zhai, C. Fang, B.-C. Yu, W. Sun, and Z. Chen, "A critical review of large language model on software engineering: An example from chatgpt and automated program repair," ArXiv, vol. abs/2310.08879, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:264127977
[586] Q. Zhang, T. Zhang, J. Zhai, C. Fang, B.-C. Yu, W. Sun, 和 Z. Chen, “大型语言模型在软件工程中的批判性综述：以ChatGPT和自动程序修复为例,” ArXiv, 第abs/2310.08879卷, 2023年. [在线]. 可用: https://api.semanticscholar.org/CorpusID:264127977

[587] R. Koo and S. Toueg, "Checkpointing and rollback-recovery for distributed systems," IEEE Transactions on Software Engineering, vol. SE-13, pp. 23-31, 1986. [Online]. Available: https://api.semanticscholar.org/CorpusID:206777989
[587] R. Koo 和 S. Toueg, “分布式系统的检查点与回滚恢复,” IEEE软件工程汇刊, 第SE-13卷, 第23-31页, 1986年. [在线]. 可用: https://api.semanticscholar.org/CorpusID:206777989

[588] Y. Luo, Q. Zhang, Q. Shen, H. Liu, and Z. Wu, "Android multi-level system permission management approach," ArXiv, vol. abs/1712.02217, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:20909985
[588] Y. Luo, Q. Zhang, Q. Shen, H. Liu, 和 Z. Wu, “Android多级系统权限管理方法,” ArXiv, 第abs/1712.02217卷, 2017年. [在线]. 可用: https://api.semanticscholar.org/CorpusID:20909985

[589] H. Hao, V. Singh, and W. Du, "On the effectiveness of api-level access control using bytecode rewriting in android," in Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security, 2013, pp. 25-36.
[589] H. Hao, V. Singh, 和 W. Du, “基于字节码重写的Android API级访问控制的有效性研究,” 发表在第8届ACM SIGSAC信息、计算机与通信安全研讨会, 2013年, 第25-36页.

[590] A. P. Felt, E. Chin, S. Hanna, D. Song, and D. Wagner, "Android permissions demystified," in Proceedings of the 18th ACM conference on Computer and communications security, 2011, pp. 627-638.
[590] A. P. Felt, E. Chin, S. Hanna, D. Song, 和 D. Wagner, “Android权限解析,” 发表在第18届ACM计算机与通信安全会议, 2011年, 第627-638页.

[591] M. Lutaaya, "Rethinking app permissions on ios," in Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems, 2018, pp. 1-6.
[591] M. Lutaaya, “重新思考iOS应用权限,” 发表在2018年CHI人机交互大会扩展摘要, 2018年, 第1-6页.

[592] Z. Xiang, L. Zheng, Y. Li, J. Hong, Q. Li, H. Xie, J. Zhang, Z. Xiong, C. Xie, C. Yang et al., "Guardagent: Safeguard Ilm agents by a guard agent via knowledge-enabled reasoning," arXiv preprint arXiv:2406.09187, 2024.
[592] Z. Xiang, L. Zheng, Y. Li, J. Hong, Q. Li, H. Xie, J. Zhang, Z. Xiong, C. Xie, C. Yang 等, “Guardagent：通过知识驱动推理的守护代理保护ILM代理,” arXiv预印本 arXiv:2406.09187, 2024年.

[593] S. Berkovits, J. D. Guttman, and V. Swarup, "Authentication for mobile agents," in Mobile Agents and Security, 1998. [Online]. Available: https://api.semanticscholar.org/CorpusID:13987376
[593] S. Berkovits, J. D. Guttman, 和 V. Swarup, “移动代理的认证,” 发表在移动代理与安全, 1998年. [在线]. 可用: https://api.semanticscholar.org/CorpusID:13987376

[594] J. Gao, S. A. Gebreegziabher, K. T. W. Choo, T. J.-J. Li, S. T. Perrault, and T. W. Malone, "A taxonomy for human-llm interaction modes: An initial exploration," in Extended Abstracts of the CHI Conference on Human Factors in Computing Systems, 2024, pp. 1-11.
[594] J. Gao, S. A. Gebreegziabher, K. T. W. Choo, T. J.-J. Li, S. T. Perrault, 和 T. W. Malone, “人机交互模式分类法：初步探索,” 发表在CHI人机交互大会扩展摘要, 2024年, 第1-11页.

[595] J. M. Bradshaw, P. J. Feltovich, and M. Johnson, "Human-agent interaction," in The handbook of human-machine interaction. CRC Press, 2017, pp. 283-300.
[595] J. M. Bradshaw, P. J. Feltovich, 和 M. Johnson, “人机代理交互,” 收录于《人机交互手册》, CRC出版社, 2017年, 第283-300页.

[596] X. Feng, Z.-Y. Chen, Y. Qin, Y. Lin, X. Chen, Z. Liu, and J.-R. Wen, "Large language model-based human-agent collaboration for complex task solving," arXiv preprint arXiv:2402.12914, 2024.
[596] X. Feng, Z.-Y. Chen, Y. Qin, Y. Lin, X. Chen, Z. Liu, 和 J.-R. Wen, “基于大型语言模型的人机协作解决复杂任务,” arXiv预印本 arXiv:2402.12914, 2024年.

[597] A. Amayuelas, L. Pan, W. Chen, and W. Wang, "Knowledge of knowledge: Exploring known-unknowns uncertainty with large language models," arXiv preprint arXiv:2305.13712, 2023.
[597] A. Amayuelas, L. Pan, W. Chen, 和 W. Wang, “知识的知识：利用大型语言模型探索已知-未知的不确定性,” arXiv预印本 arXiv:2305.13712, 2023.

[598] C. Chen, Z. Zhang, I. Khalilov, B. Guo, S. A. Gebreegziabher, Y. Ye, Z. Xiao, Y. Yao, T. Li, and T. J.-J. Li, "Toward a human-centered evaluation framework for trustworthy llm-powered gui agents," arXiv preprint arXiv:2504.17934, 2025.
[598] C. Chen, Z. Zhang, I. Khalilov, B. Guo, S. A. Gebreegziabher, Y. Ye, Z. Xiao, Y. Yao, T. Li, 和 T. J.-J. Li, “面向以人为中心的可信赖大型语言模型驱动图形用户界面代理的评估框架,” arXiv预印本 arXiv:2504.17934, 2025.

[599] C. Y. Kim, C. P. Lee, and B. Mutlu, "Understanding large-language model (llm)-powered human-robot interaction," in Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction, 2024, pp. 371-380.
[599] C. Y. Kim, C. P. Lee, 和 B. Mutlu, “理解大型语言模型（LLM）驱动的人机交互,” 载于2024年ACM/IEEE国际人机交互会议论文集, 2024, 页371-380.

[600] Y. Lu, Y. Yang, Q. Zhao, C. Zhang, and T. J.-J. Li, "Ai assistance for ux: A literature review through human-centered ai," arXiv preprint arXiv:2402.06089, 2024.
[600] Y. Lu, Y. Yang, Q. Zhao, C. Zhang, 和 T. J.-J. Li, “面向用户体验的人工智能辅助：通过以人为中心的人工智能的文献综述,” arXiv预印本 arXiv:2402.06089, 2024.

[601] J. Wester, T. Schrills, H. Pohl, and N. van Berkel, ""as an ai language model, i cannot": Investigating llm denials of user requests," in Proceedings of the CHI Conference on Human Factors in Computing Systems, 2024, pp. 1-14.
[601] J. Wester, T. Schrills, H. Pohl, 和 N. van Berkel, “‘作为一个人工智能语言模型，我不能’: 探讨大型语言模型拒绝用户请求的现象,” 载于CHI人因计算系统会议论文集, 2024, 页1-14.

[602] J. Wang, W. Ma, P. Sun, M. Zhang, and J.-Y. Nie, "Understanding user experience in large language model interactions," arXiv preprint arXiv:2401.08329, 2024.
[602] J. Wang, W. Ma, P. Sun, M. Zhang, 和 J.-Y. Nie, “理解大型语言模型交互中的用户体验,” arXiv预印本 arXiv:2401.08329, 2024.

[603] E. Cambria, L. Malandri, F. Mercorio, N. Nobani, and A. Seveso, "XAI meets Ilms: A survey of the relation between explainable ai and large language models," 2024. [Online]. Available: https://arxiv.org/abs/2407.15248
[603] E. Cambria, L. Malandri, F. Mercorio, N. Nobani, 和 A. Seveso, “可解释人工智能（XAI）与大型语言模型的交汇：一项综述,” 2024. [在线]. 可获取：https://arxiv.org/abs/2407.15248

[604] X. Wu, H. Zhao, Y. Zhu, Y. Shi, F. Yang, T. Liu, X. Zhai, W. Yao, J. Li, M. Du et al., "Usable xai: 10 strategies towards exploiting explainability in the Ilm era," arXiv preprint arXiv:2403.08946, 2024.
[604] X. Wu, H. Zhao, Y. Zhu, Y. Shi, F. Yang, T. Liu, X. Zhai, W. Yao, J. Li, M. Du 等, “可用的可解释人工智能：面向大型语言模型时代利用可解释性的10种策略,” arXiv预印本 arXiv:2403.08946, 2024.

[605] H. Cai, Y. Li, W. Wang, F. Zhu, X. Shen, W. Li, and T.-S. Chua, "Large language models empowered personalized web agents," arXiv preprint arXiv:2410.17236, 2024.
[605] H. Cai, Y. Li, W. Wang, F. Zhu, X. Shen, W. Li, 和 T.-S. Chua, “大型语言模型赋能的个性化网页代理,” arXiv预印本 arXiv:2410.17236, 2024.

[606] H. Li, C. Yang, A. Zhang, Y. Deng, X. Wang, and T.-S. Chua, "Hello again! Ilm-powered personalized agent for long-term dialogue," arXiv preprint arXiv:2406.05925, 2024.
[606] H. Li, C. Yang, A. Zhang, Y. Deng, X. Wang, 和 T.-S. Chua, “你好，再次见面！基于Ilm的个性化代理用于长期对话,” arXiv预印本 arXiv:2406.05925, 2024.

[607] H. Li, H. Jiang, T. Zhang, Z. Yu, A. Yin, H. Cheng, S. Fu, Y. Zhang, and W. He, "Traineragent: Customizable and efficient model training through llm-powered multi-agent system," arXiv preprint arXiv:2311.06622, 2023.
[607] H. Li, H. Jiang, T. Zhang, Z. Yu, A. Yin, H. Cheng, S. Fu, Y. Zhang, 和 W. He, “Traineragent：通过基于llm的大型语言模型多代理系统实现可定制且高效的模型训练,” arXiv预印本 arXiv:2311.06622, 2023.

[608] Z. Tan and M. Jiang, "User modeling in the era of large language models: Current research and future directions," arXiv preprint arXiv:2312.11518, 2023.
[608] Z. Tan 和 M. Jiang, “大语言模型时代的用户建模：当前研究与未来方向,” arXiv预印本 arXiv:2312.11518, 2023.

[609] G. Gao, A. Taymanov, E. Salinas, P. Mineiro, and D. Misra, "Aligning Ilm agents by learning latent preference from user edits," arXiv preprint arXiv:2404.15269, 2024.
[609] G. Gao, A. Taymanov, E. Salinas, P. Mineiro, 和 D. Misra, “通过学习用户编辑的潜在偏好对Ilm代理进行对齐,” arXiv预印本 arXiv:2404.15269, 2024.

[610] T. Kaufmann, P. Weng, V. Bengs, and E. Hüllermeier, "A survey of reinforcement learning from human feedback," arXiv preprint arXiv:2312.14925, 2023.
[610] T. Kaufmann, P. Weng, V. Bengs, 和 E. Hüllermeier, “基于人类反馈的强化学习综述,” arXiv预印本 arXiv:2312.14925, 2023.

[611] S. Kim, H. Kang, S. Choi, D. Kim, M. Yang, and C. Park, "Large language models meet collaborative filtering: An efficient all-round llm-based recommender system," in Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2024, pp. 1395-1406.
[611] S. Kim, H. Kang, S. Choi, D. Kim, M. Yang, 和 C. Park, “大型语言模型遇上协同过滤：一种高效的全方位基于Ilm的推荐系统,” 载于第30届ACM SIGKDD知识发现与数据挖掘会议论文集, 2024, 页1395-1406.

[612] W. Talukdar and A. Biswas, "Improving large language model (llm) fidelity through context-aware grounding: A systematic approach to reliability and veracity," arXiv preprint arXiv:2408.04023, 2024.
[612] W. Talukdar 和 A. Biswas, “通过上下文感知的基础增强大型语言模型(llm)的真实性：一种系统化的可靠性与真实性方法,” arXiv预印本 arXiv:2408.04023, 2024.

[613] X. Xiao and Y. Tao, "Personalized privacy preservation," in Proceedings of the 2006 ACM SIGMOD international conference on Management of data, 2006, pp. 229-240.
[613] X. Xiao 和 Y. Tao, “个性化隐私保护,” 载于2006年ACM SIGMOD国际数据管理会议论文集, 2006, 页229-240.

[614] N. Hojo, K. Shinoda, Y. Yamazaki, K. Suzuki, H. Sugiyama, K. Nishida, and K. Saito, "Generativegui: Dynamic gui generation leveraging llms for enhanced user interaction on chat interfaces," in Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems, 2025, pp. 1-9.
[614] N. Hojo, K. Shinoda, Y. Yamazaki, K. Suzuki, H. Sugiyama, K. Nishida, 和 K. Saito, “Generativegui：利用大型语言模型(llms)动态生成图形用户界面以增强聊天界面用户交互,” 载于CHI人机交互大会扩展摘要, 2025, 页1-9.

[615] I. H. Sarker, "LIm potentiality and awareness: a position paper from the perspective of trustworthy and responsible ai modeling," Discover Artificial Intelligence, vol. 4, no. 1, p. 40, 2024.
[615] I. H. Sarker, “LIm潜力与认知：从可信赖与负责任的人工智能建模视角出发的立场论文,” Discover Artificial Intelligence, 第4卷第1期, 页40, 2024.

[616] A. Biswas and W. Talukdar, "Guardrails for trust, safety, and ethical development and deployment of large language models (llm)," Journal of Science & Technology, vol. 4, no. 6, pp. 55-82, 2023.
[616] A. Biswas 和 W. Talukdar, “大型语言模型(llm)的信任、安全及伦理开发与部署的防护措施,” 科技期刊, 第4卷第6期, 页55-82, 2023.

[617] Y. Li, M. Du, R. Song, X. Wang, and Y. Wang, "A survey on fairness in large language models," arXiv preprint arXiv:2308.10149, 2023.
[617] Y. Li, M. Du, R. Song, X. Wang, 和 Y. Wang, “大型语言模型中的公平性综述,” arXiv预印本 arXiv:2308.10149, 2023.

[618] Z. Zhang, E. Schoop, J. Nichols, A. Mahajan, and A. Swearngin, "From interaction to impact: Towards safer ai agent through understanding and evaluating mobile ui operation impacts," in Proceedings of the 30th International Conference on Intelligent User Interfaces, 2025, pp. 727-744.
[618] Z. Zhang, E. Schoop, J. Nichols, A. Mahajan, 和 A. Swearngin, “从交互到影响：通过理解和评估移动用户界面操作影响迈向更安全的人工智能代理,” 载于第30届国际智能用户界面会议论文集, 2025, 页727-744.

[619] E. Ferrara, "Should chatgpt be biased? challenges and risks of bias in large language models," arXiv preprint arXiv:2304.03738, 2023.
[619] E. Ferrara, “ChatGPT应当存在偏见吗？大型语言模型偏见的挑战与风险,” arXiv预印本 arXiv:2304.03738, 2023.

[620] Y. Yu, Y. Zhuang, J. Zhang, Y. Meng, A. J. Ratner, R. Krishna, J. Shen, and C. Zhang, "Large language model as attributed training data generator: A tale of diversity and bias," Advances in Neural Information Processing Systems, vol. 36, 2024.
[620] Y. Yu, Y. Zhuang, J. Zhang, Y. Meng, A. J. Ratner, R. Krishna, J. Shen, 和 C. Zhang, “大型语言模型作为带属性训练数据生成器：多样性与偏见的故事,” 神经信息处理系统进展, 第36卷, 2024.

[621] A. Piñeiro-Martín, C. García-Mateo, L. Docío-Fernández, and M. D. C. Lopez-Perez, "Ethical challenges in the development of virtual assistants powered by large language models," Electronics, vol. 12, no. 14, p. 3170, 2023.
[621] A. Piñeiro-Martín, C. García-Mateo, L. Docío-Fernández, 和 M. D. C. Lopez-Perez, “基于大型语言模型的虚拟助手开发中的伦理挑战,” Electronics, 第12卷第14期, 页3170, 2023.

[622] B. Zheng, Z. Liu, S. Salisbury, Z. Du, X. Huang, Q. Zheng, L. Davis, M. Lin, X. Jin, H. Sun et al., "Agentmonitor: Towards a generalist guardrail for web agent."
[622] B. Zheng, Z. Liu, S. Salisbury, Z. Du, X. Huang, Q. Zheng, L. Davis, M. Lin, X. Jin, H. Sun 等, "Agentmonitor：面向通用型网络代理的安全护栏。"

[623] C.-M. Chan, J. Yu, W. Chen, C. Jiang, X. Liu, W. Shi, Z. Liu, W. Xue, and Y. Guo, "Agentmonitor: A plug-and-play framework for predictive and secure multi-agent systems," arXiv preprint arXiv:2408.14972, 2024.
[623] C.-M. Chan, J. Yu, W. Chen, C. Jiang, X. Liu, W. Shi, Z. Liu, W. Xue, 和 Y. Guo, "Agentmonitor：一个即插即用的预测与安全多代理系统框架," arXiv 预印本 arXiv:2408.14972, 2024。

[624] L. Lin, L. Wang, J. Guo, and K.-F. Wong, "Investigating bias in llm-based bias detection: Disparities between llms and human perception," arXiv preprint arXiv:2403.14896, 2024.
[624] L. Lin, L. Wang, J. Guo, 和 K.-F. Wong, "基于llm的偏见检测中的偏差研究：llm与人类感知之间的差异," arXiv 预印本 arXiv:2403.14896, 2024。

[625] Y. Zhang, T. Yu, and D. Yang, "Attacking vision-language computer agents via pop-ups," 2024. [Online]. Available: https://arxiv.org/abs/2411.02391
[625] Y. Zhang, T. Yu, 和 D. Yang, "通过弹窗攻击视觉-语言计算代理," 2024。[在线]. 可获取：https://arxiv.org/abs/2411.02391

[626] S. Shekkizhar and R. Cosentino, "Agi is coming... right after ai learns to play wordle," 2025. [Online]. Available: https://arxiv.org/abs/2504.15434
[626] S. Shekkizhar 和 R. Cosentino, "通用人工智能（AGI）即将到来……就在人工智能学会玩Wordle之后," 2025。[在线]. 可获取：https://arxiv.org/abs/2504.15434

[627] R. Grosse, J. Bae, C. Anil, N. Elhage, A. Tamkin, A. Tajdini, B. Steiner, D. Li, E. Durmus, E. Perez et al., "Studying large language model generalization with influence functions," arXiv preprint arXiv:2308.03296, 2023.
[627] R. Grosse, J. Bae, C. Anil, N. Elhage, A. Tamkin, A. Tajdini, B. Steiner, D. Li, E. Durmus, E. Perez 等, "利用影响函数研究大型语言模型的泛化能力," arXiv 预印本 arXiv:2308.03296, 2023。

[628] X. Zhang, J. Li, W. Chu, J. Hai, R. Xu, Y. Yang, S. Guan, J. Xu, and P. Cui, "On the out-of-distribution generalization of multimodal large language models," arXiv preprint arXiv:2402.06599, 2024.
[628] X. Zhang, J. Li, W. Chu, J. Hai, R. Xu, Y. Yang, S. Guan, J. Xu, 和 P. Cui, "多模态大型语言模型的分布外泛化研究," arXiv 预印本 arXiv:2402.06599, 2024。

[629] E. Li and J. Waldo, "Websuite: Systematically evaluating why web agents fail," arXiv preprint arXiv:2406.01623, 2024.
[629] E. Li 和 J. Waldo, "Websuite：系统性评估网络代理失败的原因," arXiv 预印本 arXiv:2406.01623, 2024。

[630] Y. Song, W. Xiong, X. Zhao, D. Zhu, W. Wu, K. Wang, C. Li, W. Peng, and S. Li, "Agentbank: Towards generalized llm agents via fine-tuning on 50000+ interaction trajectories," arXiv preprint arXiv:2410.07706, 2024.
[630] Y. Song, W. Xiong, X. Zhao, D. Zhu, W. Wu, K. Wang, C. Li, W. Peng, 和 S. Li, "Agentbank：通过对5万多条交互轨迹微调，迈向通用llm代理," arXiv 预印本 arXiv:2410.07706, 2024。

[631] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," Journal of Big data, vol. 3, pp. 1-40, 2016.
[631] K. Weiss, T. M. Khoshgoftaar, 和 D. Wang, "迁移学习综述," 大数据期刊, 第3卷, 第1-40页, 2016。

[632] Y. Chen, R. Zhong, S. Zha, G. Karypis, and H. He, "Meta-learning via language model in-context tuning," arXiv preprint arXiv:2110.07814, 2021.
[632] Y. Chen, R. Zhong, S. Zha, G. Karypis, 和 H. He, "通过语言模型上下文调优实现元学习," arXiv 预印本 arXiv:2110.07814, 2021。

[633] 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 Ilm-based agents," arXiv preprint arXiv:2403.03101, 2024.
[633] Y. Zhu, S. Qiao, Y. Ou, S. Deng, N. Zhang, S. Lyu, Y. Shen, L. Liang, J. Gu, 和 H. Chen, "Knowagent：基于知识增强的Ilm代理规划," arXiv 预印本 arXiv:2403.03101, 2024。

[634] Y. Guan, D. Wang, Y. Wang, H. Wang, R. Sun, C. Zhuang, J. Gu, and Z. Chu, "Explainable behavior cloning: Teaching large language model agents through learning by demonstration," arXiv preprint arXiv:2410.22916, 2024.
[634] Y. Guan, D. Wang, Y. Wang, H. Wang, R. Sun, C. Zhuang, J. Gu, 和 Z. Chu, "可解释行为克隆：通过示范学习教导大型语言模型代理," arXiv 预印本 arXiv:2410.22916, 2024。

[635] C.-Y. Hsieh, S.-A. Chen, C.-L. Li, Y. Fujii, A. Ratner, C.-Y. Lee, R. Krishna, and T. Pfister, "Tool documentation enables zero-shot tool-usage with large language models," arXiv preprint arXiv:2308.00675, 2023.
[635] C.-Y. Hsieh, S.-A. Chen, C.-L. Li, Y. Fujii, A. Ratner, C.-Y. Lee, R. Krishna, 和 T. Pfister, "工具文档支持大型语言模型的零样本工具使用," arXiv 预印本 arXiv:2308.00675, 2023。

[636] T. Kagaya, T. J. Yuan, Y. Lou, J. Karlekar, S. Pranata, A. Kinose, K. Oguri, F. Wick, and Y. You, "Rap: Retrieval-augmented planning with contextual memory for multimodal Ilm agents," arXiv preprint arXiv:2402.03610, 2024.
[636] T. Kagaya, T. J. Yuan, Y. Lou, J. Karlekar, S. Pranata, A. Kinose, K. Oguri, F. Wick, 和 Y. You, “Rap：基于上下文记忆的多模态ILM代理检索增强规划”，arXiv预印本 arXiv:2402.03610，2024年。