Longzhao Huang [1] , Jun Liu [1] , Changwei Wang [2,3] , Rongtao Xu [4] , Wenhao Xu [1] , Zhiwei Xu [5] , Qi Zhang [6] , Yu Zhang [6] , Kexue Fu [2,6] , Longxiang Gao [2] , Yanran Xu [7,8] , Lei Zhang [9] , Li Guo [1] , and Shibiao Xu[1]

黄龙昭 [1] , 刘军 [1] , 王长伟 [2,3] , 徐荣涛 [4] , 徐文浩 [1] , 徐志伟 [5] , 张琪 [6] , 张宇 [6] , 傅科学 [2,6] , 高龙翔 [2] , 徐艳然 [7,8] , 张磊 [9] , 郭力 [1] , 史彪 Xu[1]

[1] School of Artificial Intelligence, Beijing University of Posts and Telecommunications

[1] 北京邮电大学人工智能学院

[2] Ministry of Education, Shandong Computer Science Center (National Supercomputer

[2] 教育部山东省计算机科学中心 (济南国家超级计算中心)

Center in Jinan), Key Laboratory of Computing Power Network and Information Security, Qilu University of Technology (Shandong Academy of Sciences)

齐鲁工业大学 (山东科学院) 计算能力网络与信息安全重点实验室

[3] Shandong Fundamental Research Center for Computer Science, Shandong Provincial Key Laboratory of Computer Networks

[3] 山东省计算机科学基础研究中心，山东省计算机网络重点实验室

[4] Institute of Automation, The State Key Laboratory of Multimodal Artificial Intelligence Systems, Chinese Academy of Sciences

[4] 中国科学院自动化研究所，多模态人工智能系统国家重点实验室

[5] School of Artificial Intelligence, Shandong University

[5] 山东大学人工智能学院

[6] Tongji University

[6] 同济大学

[7] Transport Engineering and Mobility

[7] 交通工程与出行

8RWTH Aachen

德国亚琛工业大学

[9] Institute of Computing Technology, Chinese Academy of Sciences

[9] 中国科学院计算技术研究所

July 22, 2025

2025 年 7 月 22 日

Abstract

摘要

The Graphical User Interface (GUI) is a visual method that allows users to interact with computers and mobile devices. Nowadays, users rely on GUI for completing some tasks, such as browsing web or using mobile applications. Users often meet some needs such as setting an alarm for 8:00 AM to wake them up and checking the weather for tomorrow. Some commercial agents have been integrated into users personal phones to help the user accomplish a series of basic tasks. Unfortunately, these commercial agents often relied on fixed templates or program scripts to ensure reliability. This also limited their functionality to some basic system applications. Recently large language models (LLMs) have made significant breakthroughs in natural language processing (NLP). Astonishingly, LLMs have demonstrated not only a strong ability to understand and generate text but also planning and reasoning capabilities. Some researchers have considered using LLMs as the agent's brain, equipping these agents with corresponding capabilities. LLM-based agents are also being applied to help users automate tasks on their personal phones and computers. These agents often can understand the GUI environment on personal phones and computers, allowing them to make decisions to complete tasks. This is also the origin of the term "GUI Agent". Our review surveys recent research on LLM-based GUI Agents. We summarize the capabilities of existing GUI Agents and also discuss the GUI Agent task automation pipeline. A comprehensive list of studies in this paper will be available at a GitHub repositories.

图形用户界面 (GUI) 是一种允许用户与计算机和移动设备交互的视觉方式。如今，用户依赖 GUI 完成一些任务，如浏览网页或使用移动应用。用户常有一些需求，比如设置早上 8 点的闹钟叫醒自己，或查询明天天气。一些商业代理已集成到用户的个人手机中，帮助用户完成一系列基础任务。不幸的是，这些商业代理通常依赖固定模板或程序脚本以保证可靠性，这也限制了它们的功能仅限于一些基础系统应用。近年来，大型语言模型 (LLMs) 在自然语言处理 (NLP) 领域取得了重大突破。令人惊讶的是，LLMs 不仅展现了强大的文本理解和生成能力，还具备规划和推理能力。一些研究者考虑将 LLMs 作为代理的“大脑”，赋予这些代理相应的能力。基于 LLM 的代理也被应用于帮助用户自动化个人手机和计算机上的任务。这些代理通常能够理解个人手机和计算机上的 GUI 环境，从而做出决策完成任务。这也是“GUI 代理”一词的由来。我们的综述调查了近期基于 LLM 的 GUI 代理研究，概述了现有 GUI 代理的能力，并讨论了 GUI 代理任务自动化流程。本文所列研究的完整列表将发布于 GitHub 仓库。

GA: A Comprehensive Survey on LLM-based GUI Agent

GA: 基于大型语言模型的 GUI 代理综合综述

Longzhao Huang [a] , Jun Liu [a] , Changwei Wang [b,d] , Rongtao Xu[c] , Wenhao Xu[a] , Zhiwei Xu[h] , Qi Zhang [e] , Yu Zhang [e] , Kexue Fu[b] , Longxiang Gao [b] , YanRan Xu[f] , Lei Zhang [g] , Li Guo [a] , Shibiao Xu[a]

黄龙昭 [a] , 刘军 [a] , 王长伟 [b,d] , 荣涛 Xu[c] , 文浩 Xu[a] , 志伟 Xu[h] , 张琪 [e] , 张宇 [e] , 科学 Fu[b] , 高龙翔 [b] , 艳然 Xu[f] , 张磊 [g] , 郭力 [a] , 史彪 Xu[a]

[a] School of Artificial Intelligence, Beijing University of Posts and Telecommunications, BeiJing, China

[a] 北京邮电大学人工智能学院，中国北京

[b] Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), JiNan, China

[b] 教育部算力网络与信息安全重点实验室，山东省计算中心 (国家超级计算济南中心)，齐鲁工业大学 (山东省科学院)，中国济南

[c] The State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences, China, BeiJing, China

[c] 中国科学院自动化研究所多模态人工智能系统国家重点实验室，中国北京

[d] Shandong Provincial Key Laboratory of Computer Networks, Shandong Fundamental

[d] 山东省计算机网络重点实验室，山东计算机科学基础

Research Center for Computer Science, JiNan, China e Tongji University, Shanghai, China

研究中心，中国济南同济大学，中国上海

[f] Transport Engineering and Mobility, Civil Engineering, RWTH Aachen, Aachen, Germany

[f] 德国亚琛工业大学土木工程学院交通工程与交通流研究所，德国亚琛

[9] Institute of Computing Technology, Chinese Academy of Sciences, BeiJing, China [h] School of Artificial Intelligence, Shandong University, JiNan, China

[9] 中国科学院计算技术研究所，中国北京 [h] 山东大学人工智能学院，中国济南

Abstract

摘要

The Graphical User Interface (GUI) is a visual method that allows users to interact with computers and mobile devices. Nowadays, users rely on GUI for completing some tasks, such as browsing web or using mobile applications. Users often meet some needs such as setting an alarm for 8:00 AM to wake them up and checking

the weather for tomorrow. Some commercial agents have been integrated into users personal phones to help the user accomplish a series of basic tasks. Unfortunately, these commercial agents often relied on fixed templates or program scripts to ensure reliability. This also limited their functionality to some basic system applications. Recently large language models (LLMs) have made significant breakthroughs in natural language processing (NLP). Astonishingly, LLMs have demonstrated not only a strong ability to understand and generate text but also planning and

图形用户界面 (Graphical User Interface，GUI) 是一种可视化方法，允许用户与计算机和移动设备进行交互。如今，用户依靠图形用户界面来完成一些任务，如浏览网页或使用移动应用程序。用户经常会有一些需求，比如设置早上 8 点的闹钟叫醒自己，以及查看明天的天气。一些商业智能助手已被集成到用户的个人手机中，以帮助用户完成一系列基本任务。不幸的是，这些商业智能助手通常依赖固定的模板或程序脚本来确保可靠性。这也将它们的功能限制在了一些基本的系统应用上。最近，大语言模型 (Large Language Models，LLMs) 在自然语言处理 (Natural Language Processing，NLP) 领域取得了重大突破。令人惊讶的是，大语言模型不仅展现出了强大的文本理解和生成能力，还具备规划和

---

*Shibiao Xu is the corresponding author (shibiaoxu@bupt.edu.cn).

* 徐世标 Xu 为通信作者 (shibiaoxu@bupt.edu.cn)。

---

reasoning capabilities. Some researchers have considered using LLMs as the agent's brain, equipping these agents with corresponding capabilities. LLM-based agents are also being applied to help users automate tasks on their personal phones and computers. These agents often can understand the GUI environment on personal phones and computers, allowing them to make decisions to complete tasks. This is also the origin of the term "GUI Agent". Our review surveys recent research on LLM-based GUI Agents. We summarize the capabilities of existing GUI Agents and also discuss the GUI Agent task automation pipeline. A comprehensive list of studies in this paper will be available at a GitHub repositories.

推理能力。一些研究人员考虑将大语言模型用作智能助手的“大脑”，赋予这些智能助手相应的能力。基于大语言模型的智能助手也正被应用于帮助用户在个人手机和计算机上实现任务自动化。这些智能助手通常能够理解个人手机和计算机上的图形用户界面环境，从而做出决策来完成任务。这也是“图形用户界面智能助手”这一术语的由来。我们的综述调查了近期关于基于大语言模型的图形用户界面智能助手的研究。我们总结了现有图形用户界面智能助手的能力，并讨论了图形用户界面智能助手的任务自动化流程。本文中全面的研究列表将在 GitHub 仓库中提供。

Keywords: GUI Agent, Large Language Model, Task Automation, GUI Environment Understanding

关键词: 图形用户界面智能助手；大语言模型；任务自动化；图形用户界面环境理解

1. Introduction

1. 引言

GUI is a visual user interface that allows users to interact with computers and mobile electronic devices using graphical icons, visual indicators, and interactive elements such as buttons and menus. GUI has become essential in modern computing, making interactions intuitive and accessible across various platforms, including computer operating systems [1], mobile applications like Android [2], and web applications [3]. Users typically need to repeatedly perform some basic daily tasks (such as setting an alarm). In this process, some current commercial agents [4] can assist users complete these basic tasks. These commercial agents find it difficult to execute multiple complex steps to accomplish high-level tasks (such as querying today's gold price and sending it to JACK via WeChat). Leveraging agents to assist users in navigating these high-level tasks has become a very important field [5].

图形用户界面是一种可视化的用户界面，允许用户使用图形图标、视觉指示器以及按钮和菜单等交互式元素与计算机和移动电子设备进行交互。图形用户界面在现代计算中变得至关重要，使交互在各种平台上都变得直观且易于操作，包括计算机操作系统 [1]、安卓等移动应用程序 [2] 以及网页应用程序 [3]。用户通常需要反复执行一些基本的日常任务 (如设置闹钟)。在这个过程中，一些现有的商业智能助手 [4] 可以帮助用户完成这些基本任务。这些商业智能助手很难执行多个复杂步骤来完成高级任务 (如查询今天的黄金价格并通过微信发送给杰克)。利用智能助手协助用户完成这些高级任务已成为一个非常重要的领域 [5]。

The exploration of commercial agents has seen significant advancements. On smartphones, Apple integrated Siri [4] into the iPhone 4S as early as 2011. Siri offers a convenient user experience through voice interaction, assisting users with various daily tasks such as information retrieval, navigation, and translation. For personal computers, Microsoft introduced Copilot [6] to assist Windows users automating document drafting, creating presentations, summarizing emails, and more, thereby enhancing productivity. Commercial agents [4, 7] primarily adopt template-based approaches to automate tasks. Template-based methods involve predefining a series of task templates, each template of which contains detailed descriptions. When a user sends a command to the agent, the agent maps this command to the corresponding template and extracts the corresponding template parameter from the command. While template-based methods are precise and reliable, they lack scalability and flexibility, making it difficult to support complex tasks or other application scenarios.

商业智能助手的探索取得了显著进展。在智能手机方面，苹果早在 2011 年就将 Siri[4] 集成到 iPhone 4S 中。Siri 通过语音交互提供了便捷的用户体验，帮助用户完成各种日常任务，如信息检索、导航和翻译。在个人计算机方面，微软推出了 Copilot[6]，以帮助 Windows 用户自动起草文档、创建演示文稿、总结电子邮件等，从而提高工作效率。商业智能助手 [4, 7] 主要采用基于模板的方法来实现任务自动化。基于模板的方法涉及预先定义一系列任务模板，每个模板都包含详细的描述。当用户向智能助手发送命令时，智能助手将该命令映射到相应的模板，并从命令中提取相应的模板参数。虽然基于模板的方法精确可靠，但它们缺乏可扩展性和灵活性，难以支持复杂任务或其他应用场景。

Moving beyond the template-based methods, the introduction of GUI Agents marks a significant evolution in the field of task automation. Unlike template-based methods, GUI Agents are designed to understand the GUI environment in real-time, and then select and execute the next action. We have defined the task automation pipeline involving the GUI Agent in Figure 1. There are three key basic roles: the device, the user, and the GUI Agent. In Figure 1, the user gives the agent an initialization task (Search for "GUI Agent" in Google Search), which could be a voice command or text input. The next step for the agent is to then integrate the task instruction and the GUI environment as the input command for the agent and decide the next action. The agent needs to determine whether the task is completed independently.

超越基于模板的方法，图形用户界面代理 (GUI Agent) 的引入标志着任务自动化领域的重大发展。与基于模板的方法不同，图形用户界面代理旨在实时理解图形用户界面环境，然后选择并执行下一步操作。我们在图 1 中定义了涉及图形用户界面代理的任务自动化流程。有三个关键的基本角色: 设备、用户和图形用户界面代理。在图 1 中，用户向代理下达初始化任务 (在谷歌搜索中搜索“图形用户界面代理”)，这可以是语音命令或文本输入。代理的下一步是将任务指令和图形用户界面环境整合为代理的输入命令，并决定下一步操作。代理需要独立判断任务是否完成。

There are many methods to implement the GUI Agents in the past, such as demonstrations learning methods [8, 9, 10], supervised learning methods 11, 12], and reinforcement learning methods [13, 14, 15]. These methods often require extensive manual data collection or the handcrafted design of reward functions. Despite their potential, the reliance on significant human intervention remains a considerable challenge for broader implementation and adaptability. Recently, LLMs [16, 17, 18, 19] have achieved remarkable success in NLP. Many consider LLMs to be the dawn of artificial general intelligence. LLMs not only exhibit strong capabilities in understanding and generating natural language but also demonstrate impressive performance in planning and reasoning tasks [20]. Some researchers [21, 22, 23] are now attempting to endow LLMs with perception abilities and action spaces, thereby transforming them into agents. Some researchers [24, 25, 26] have also started exploring LLM-based GUI Agents. Compared to traditional agents, LLM-based GUI agents utilize in-contextual learning to automate task [27]. These GUI Agents [24, 25] avoid the need to collect large amounts of data for training, while also increasing the flexibility of GUI agents [26].

过去有许多实现图形用户界面代理的方法, 如演示学习方法 [8, 9, 10]、监督学习方法 [11, 12] 和强化学习方法 [13, 14, 15]。这些方法通常需要大量的手动数据收集或手工设计奖励函数。尽管它们有潜力, 但严重依赖人工干预仍然是更广泛实施和适应性方面的一大挑战。最近, 大语言模型 (LLM)[16, 17, 18, 19] 在自然语言处理 (NLP) 领域取得了显著成功。许多人认为大语言模型是通用人工智能的曙光。大语言模型不仅在理解和生成自然语言方面表现出强大的能力，而且在规划和推理任务中也展现出令人印象深刻的性能 [20]。一些研究人员 [21, 22, 23] 现在正试图赋予大语言模型感知能力和动作空间，从而将它们转变为代理。一些研究人员 [24, 25, 26] 也开始探索基于大语言模型的图形用户界面代理。与传统代理相比，基于大语言模型的图形用户界面代理利用上下文学习来实现任务自动化 [27]。这些图形用户界面代理 [24, 25] 避免了为训练收集大量数据的需求，同时也增加了图形用户界面代理的灵活性 [26]。

From the GUI Agent role in task automation in Figure 1, GUI Agents must have three basic capabilities: GUI environment comprehension, device control, and user interaction. Based on different ways to understand the GUI environment, we propose a taxonomy that divides GUI Agents into three categories: Vision-Based, Text-Based, and Hybrid Text-Vision. After comprehending the GUI environment, agents can generate automated scripts or select action from the action space to control the device. During the task execution process, agents can chat with users in real-time to refine task information. In addition to the three basic capabilities, agents can provide personalized services to users to enhance the user experience. Furthermore, some studies have applied multi-agents in the task automation field [28]. Consequently, collaborating with other agents is also important for GUI Agents. Our paper surveys numerous recent GUI Agents, and discusses the relevant capabilities of the agents in Section 3.

从图 1 中图形用户界面代理在任务自动化中的角色来看，图形用户界面代理必须具备三种基本能力: 图形用户界面环境理解、设备控制和用户交互。基于理解图形用户界面环境的不同方式，我们提出了一种分类方法，将图形用户界面代理分为三类: 基于视觉的、基于文本的和文本 - 视觉混合的。在理解图形用户界面环境后，代理可以生成自动化脚本或从动作空间中选择动作来控制设备。在任务执行过程中，代理可以与用户实时聊天以完善任务信息。除了这三种基本能力外，代理还可以为用户提供个性化服务以提升用户体验。此外，一些研究已将多代理应用于任务自动化领域 [28]。因此，与其他代理协作对图形用户界面代理也很重要。本文对近期众多图形用户界面代理进行了调研，并在第 3 节讨论了这些代理的相关能力。

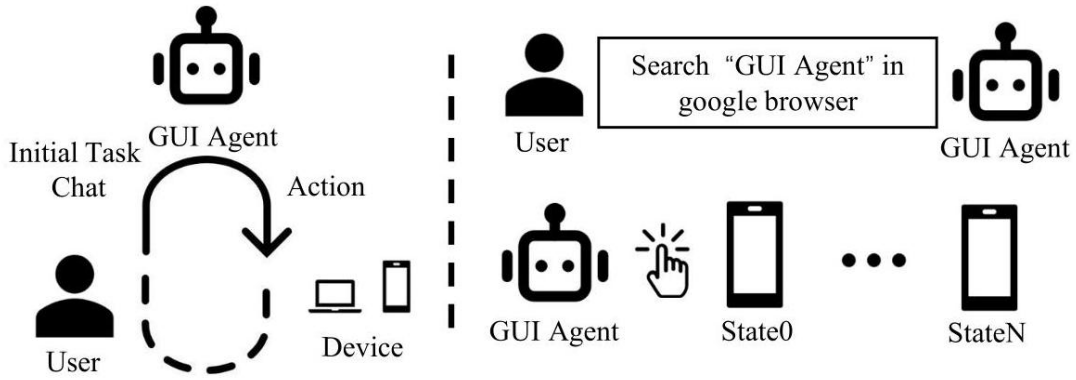


Figure 1: The Basic Pipeline of GUI Agent Task Automation.

图 1: 图形用户界面代理任务自动化的基本流程。

Besides the capabilities of the GUI Agents, the task automation pipeline is also important for task efficiency. Most agents follow a single-stage pipeline, where the agents directly performs tasks within the GUI environment. Such agents rely solely on knowledge obtained from prior training or information described by humans. However, due to the diversity of GUI scenarios, agents often face unknown manipulation methods within the new GUI environment. To better alleviate this issue, some researchers have proposed a two-stage pipeline (from exploration to exploitation). In the exploration stage, agents first freely explore the GUI environment to form a knowledge database specific to that environment. And then agents exploit this knowledge database to complete tasks within the GUI environment. We discuss the two-stage task automation pipeline in Section 4.

除了图形用户界面代理的能力外，任务自动化流程对任务效率也很重要。大多数代理遵循单阶段流程，即代理直接在图形用户界面环境中执行任务。此类代理仅依赖从先前训练中获得的知识或人类描述的信息。然而，由于图形用户界面场景的多样性，代理在新的图形用户界面环境中常常会遇到未知的操作方法。为了更好地缓解这一问题，一些研究人员提出了两阶段流程 (从探索到利用)。在探索阶段，代理首先自由探索图形用户界面环境，以形成特定于该环境的知识库。然后代理利用这个知识库在图形用户界面环境中完成任务。我们将在第 4 节讨论两阶段任务自动化流程。

Our paper surveys numerous recent studies on LLM-based GUI Agents, providing a detailed overview of the current state of research in this area. We summarize the current basic capabilities of GUI Agents and explore the task automation pipeline. We also summarize relevant evaluation benchmarks. Finally, we summarize the challenges and potential opportunities. Overall, this paper makes the following contributions:

本文对近期众多关于基于大语言模型的图形用户界面代理的研究进行了调研，详细概述了该领域的当前研究现状。我们总结了图形用户界面代理目前的基本能力，并探讨了任务自动化流程。我们还总结了相关的评估基准。最后，我们总结了挑战和潜在机遇。总体而言，本文做出了以下贡献:

1. Our paper surveys numerous recent studies on LLM-based GUI Agents, providing a detailed overview in this area. A comprehensive list of studies in this paper will be available at https://github.com/longzhaohuang/GUI-Agent-Survey.

1. 本文对近期众多关于基于大语言模型的图形用户界面代理的研究进行了调研，详细概述了该领域的情况。本文中全面的研究列表可在 https://github.com/longzhaohuang/ GUI - Agent - Survey 获取。

2. From the role of GUI Agents in task automation, our paper summarizes the relevant capabilities, covering GUI environment comprehension, device control, user interaction, personalized services, and collaboration with other agents. We also provide a taxonomy based on the way to understand the GUI environment.

2. 从图形用户界面代理在任务自动化中的角色出发，本文总结了相关能力，包括图形用户界面环境理解、设备控制、用户交互、个性化服务以及与其他代理的协作。我们还根据理解图形用户界面环境的方式提供了一种分类方法。

3. Our paper discusses the two-stage task automation pipeline involved by GUI Agents: from exploration to exploitation.

3. 本文讨论了图形用户界面代理所涉及的两阶段任务自动化流程: 从探索到利用。

4. Our paper summarizes the remaining challenges in this field and explores potential opportunities.

4. 本文总结了该领域尚存的挑战，并探索了潜在机遇。

The remaining structure of this paper is as follows: In Section 2 we provide essential background knowledge on GUI Agents, enabling readers to understand the subsequent paper better; In Section 3, we summarize the relevant capabilities of GUI Agents and propose a taxonomy based on the way to understand the environment; In Section 4, we summarize the two-stage task automation pipeline involved by GUI Agents; In Section 5, we summarize the commonly used evaluation datasets and their corresponding evaluation criteria; In Section 6 we discuss the existing challenges faced by GUI Agents and opportunity in this era; In Section 7, we conclude the paper by summarizing the key contributions and overall content.

本文的剩余结构如下: 在第 2 节中，我们提供了关于图形用户界面代理 (GUI Agents) 的必要背景知识，以便读者更好地理解后续内容；在第 3 节中，我们总结了图形用户界面代理的相关能力，并根据对环境的理解方式提出了一种分类方法；在第 4 节中，我们总结了图形用户界面代理所涉及的两阶段任务自动化流程；在第 5 节中，我们总结了常用的评估数据集及其相应的评估标准；在第 6 节中，我们讨论了图形用户界面代理目前面临的挑战以及这个时代所带来的机遇；在第 7 节中，我们通过总结关键贡献和整体内容来结束本文。

2. Background Knowledge

2. 背景知识

This section expounds the background knowledge of GUI Agents. In Section 2.1, we introduce the fundamental concepts of LLMs. While LLMs is adept at processing discrete textual data, leveraging visual perception is crucial in GUI task automation. Given the limitations of LLMs in handling only textual data, we discuss the integration of LLMs with multi-modal perception capabilities in section 2.2, which enable LLMs to interact with and understand GUI environment more effectively. The section 2.3 introduces the concept of LLM-based agents, outlining their characteristics and capacity.

本节阐述了图形用户界面代理的背景知识。在 2.1 节中，我们介绍了大语言模型 (LLMs) 的基本概念。虽然大语言模型擅长处理离散的文本数据，但在图形用户界面任务自动化中，利用视觉感知至关重要。鉴于大语言模型仅能处理文本数据的局限性，我们将在 2.2 节中讨论如何将大语言模型与多模态感知能力相结合，使大语言模型能够更有效地与图形用户界面环境进行交互和理解。2.3 节介绍了基于大语言模型的代理的概念，概述了它们的特点和能力。

2.1. Fundamental Concept of Large Language Model

2.1. 大语言模型的基本概念

Recently, LLMs have achieved unprecedented advancements in NLP, and has demonstrated exceptional capabilities in both understanding and generating natural language. In the context of GUI automation task, LLMs can simultaneously comprehend the GUI environment represented by view hierarchy (VH) and document object model (DOM) structure repersentation and the instructions provided by users.

最近，大语言模型在自然语言处理 (NLP) 领域取得了前所未有的进展，在理解和生成自然语言方面都展现出了卓越的能力。在图形用户界面自动化任务中，大语言模型可以同时理解由视图层次结构 (VH) 和文档对象模型 (DOM) 结构表示的图形用户界面环境以及用户提供的指令。

The LLMs architecture are typically based on the transformer 29. Traditional transformers generally consist of an encoder component and a decoder component. Based on the components utilized by LLMs, LLMs can typically be categorized into three types: encoder-only, decoder-only, and encoder-decoder type. The encoder-only method, exemplified by the BERT series [30], uses the encoder component to transform text sequences into embedding representations. Then encoder-only method extracts a single embedding to perform global comprehension tasks such as emotional prediction. Despite excelling in global comprehension, the heavy condensation of information into one single embedding can lead to information loss. This makes this architecture unsuitable for text generation and not the preferred choice for contemporary LLMs architectures. The decoder-only method generates output sequences in an autoregressive manner. Based on the attention mechanism, decoder-only methods are divided into causal-decoder method [27] and prefix-decoder method [31]. Compared to the causal-decoder method, the prefix-decoder method employs bidirectional attention on the prefix part of the sequence. Encoder-decoder methods, such as T5 [32], follow the classical transformer design, combining both encoding and decoding processes. They can understand input text and, through a cross-attention mechanism, autoregressive generate the target sequence. This dual capability enables them to handle both text understanding and generation.

大语言模型的架构通常基于 Transformer 架构 [29]。传统的 Transformer 通常由编码器组件和解码器组件组成。根据大语言模型所使用的组件，大语言模型通常可以分为三种类型: 仅编码器型、仅解码器型和编码器 - 解码器型。仅编码器方法以 BERT 系列 [30] 为例，使用编码器组件将文本序列转换为嵌入表示。然后仅编码器方法提取单个嵌入来执行全局理解任务，如情感预测。尽管在全局理解方面表现出色，但将信息大量浓缩到单个嵌入中可能会导致信息丢失。这使得这种架构不适合文本生成，也不是当代大语言模型架构的首选。仅解码器方法以自回归的方式生成输出序列。基于注意力机制，仅解码器方法分为因果解码器方法 [27] 和前缀解码器方法 [31]。与因果解码器方法相比，前缀解码器方法在序列的前缀部分采用双向注意力。编码器 - 解码器方法，如 T5 [32]，遵循经典的 Transformer 设计，结合了编码和解码过程。它们可以理解输入文本，并通过交叉注意力机制自回归地生成目标序列。这种双重能力使它们能够处理文本理解和生成任务。

The remarkable capabilities of LLMs are largely attributable to their enormous parameter and their extensive training datasets. According to the Scaling Law [33], increasing the scale of parameters and data effectively enhances LLMs performance on downstream tasks. Researchers [18, 34] typically begin by expanding LLMs to tens of billions of parameters, even hundreds of billions. Then they train LLMs following the two-stage approach: ”pre-training and fine-tuning.” In the pre-training phase, LLMs are initially trained on a large-scale corpus using specially designed pre-training tasks. This stage helps the model learn a wide range of linguistic features and patterns. After a pre-training phase, LLMs are fine-tuned on specific downstream tasks to adapt to the unique requirements of these tasks.

大语言模型的卓越能力在很大程度上归功于其庞大的参数和广泛的训练数据集。根据缩放定律 [33]，增加参数和数据的规模可以有效提高大语言模型在下游任务中的性能。研究人员 [18, 34] 通常首先将大语言模型扩展到数千亿甚至数万亿的参数。然后他们按照“预训练和微调”的两阶段方法来训练大语言模型。在预训练阶段，大语言模型首先使用专门设计的预训练任务在大规模语料库上进行训练。这个阶段有助于模型学习广泛的语言特征和模式。预训练阶段之后，大语言模型在特定的下游任务上进行微调，以适应这些任务的独特要求。

2.2. Equipping Large Language Model with Perceptual Abilities

2.2. 赋予大语言模型感知能力

Despite LLMs impressive performance across various natural language processing tasks, they are inherently limited in handling visual information. While LLMs can assist GUI Agents by parsing VH and DOM structure representation and understanding user commands, they face challenges such as excessive input length and loss of rich visual details. Consequently, researchers [35, 36] have begun to explore different ways to endow LLMs with perceptual capabilities. This enhancement allows LLMs to process visual information, thereby overcoming the limitations of traditional LLMs in GUI automation tasks.

尽管大语言模型在各种自然语言处理任务中表现出色，但它们在处理视觉信息方面存在固有的局限性。虽然大语言模型可以通过解析视图层次结构和文档对象模型结构表示并理解用户命令来协助图形用户界面代理，但它们面临输入长度过长和丰富视觉细节丢失等挑战。因此，研究人员 [35, 36] 开始探索不同的方法来赋予大语言模型感知能力。这种增强使大语言模型能够处理视觉信息，从而克服传统大语言模型在图形用户界面自动化任务中的局限性。

In the early stages, LLMs [35, 37, 38] achieved multimodal capabilities by integrating external expert vision tools [39, 40]. For instance, VISPROG [35] leverages the ICL abilities of LLMs to generate Python-like programs that call various expert vision models or image processing routines. At that moment, VISPROG demonstrated significant flexibility across four tasks: compositional visual question answering, zero-shot reasoning on image pairs, factual knowledge object tagging, and language-guided image editing. Currently, most LLMs [36, 41, 42] are typically designed in an end-to-end trainable format. This enables the LLMs to process visual information without relying on external expert visual models, commonly referred as Multi-modal Large Language Models (MLLMs). These MLLMs consist of three main components: a vision encoder, a learnable mapping layer, and a LLM. Initially, MLLMs use a frozen vision encoder to extract image information, which is then aligned with textual data via a learnable mapping layer. Finally, MLLMs use a LLM to auto-regressively generate the target text sequence. The learnable mapping layer can be implemented using techniques such as MLP, cross-attention mechanisms, or Q-Former [41]. LLava [43] employs a simple linear mapping layer to align visual and textual representations, subsequently concatenating these aligned representations to input LLM. Flamingo [36] uses a cross-attention mechanism as its learnable mapping layer, specifically, flamingo leverages visual features to enhance text representations within the cross-attention layers. This eliminates the need for LLM to handle additional visual tokens. Q-Former, initially introduced in the BLIP series [41], uses learnable queries to record visual information related to textual information, and then integrate this information in transformed layer to effectively align these two patterns. Due to significant performance improvements in MLLMs, Q-Former has been widely adopted in subsequent research [44, 45, 46].

在早期阶段, 大语言模型 (LLMs)[35, 37, 38] 通过集成外部专业视觉工具 [39, 40] 实现了多模态能力。例如，VISPROG [35] 利用大语言模型的上下文学习 (ICL) 能力生成类似 Python 的程序，这些程序可以调用各种专业视觉模型或图像处理程序。当时，VISPROG 在四项任务中展现出了显著的灵活性: 组合式视觉问答、图像对的零样本推理、事实知识对象标记以及语言引导的图像编辑。目前，大多数大语言模型 [36, 41, 42] 通常采用端到端可训练的形式进行设计。这使得大语言模型能够在不依赖外部专业视觉模型的情况下处理视觉信息，这类模型通常被称为多模态大语言模型 (MLLMs)。这些多模态大语言模型主要由三个部分组成: 视觉编码器、可学习的映射层和大语言模型。首先，多模态大语言模型使用一个固定的视觉编码器提取图像信息，然后通过可学习的映射层将其与文本数据对齐。最后，多模态大语言模型使用大语言模型自回归地生成目标文本序列。可学习的映射层可以使用诸如多层感知机 (MLP)、交叉注意力机制或 Q-Former [41] 等技术来实现。LLava [43] 采用简单的线性映射层来对齐视觉和文本表示，随后将这些对齐的表示拼接起来输入到大语言模型中。Flamingo [36] 使用交叉注意力机制作为其可学习的映射层，具体来说，Flamingo 利用视觉特征在交叉注意力层中增强文本表示。这消除了大语言模型处理额外视觉标记的需求。Q-Former 最初在 BLIP 系列 [41] 中被引入, 它使用可学习的查询来记录与文本信息相关的视觉信息, 然后在转换层中整合这些信息，以有效地对齐这两种模式。由于多模态大语言模型的性能有了显著提升，Q-Former 在后续的研究 [44, 45, 46] 中得到了广泛应用。

2.3. Relevant Concepts of the Agent

2.3. 智能体的相关概念

This section elucidates the concepts related to agents and describes the transition from LLMs to intelligent agents. This section is to provide readers with an understanding of LLM-based agents so that readers can better understand the subsequent content. An agent refers to an entity capable of perceiving its environment, formulating

strategies, and executing tasks, these entities typically exhibit autonomy, reactivity, proactivity, and even social abilities [20].

本节阐述了与智能体相关的概念，并描述了从大语言模型到智能体的转变。本节旨在让读者了解基于大语言模型的智能体，以便读者更好地理解后续内容。智能体是指能够感知其环境、制定策略并执行任务的实体，这些实体通常表现出自主性、反应性、主动性，甚至社交能力 [20]。

Early agent methodologies primarily focused on symbolic agent and reactive agents. Symbolic agent, epitomized by knowledge-based expert systems 47, 48, 49, 50], relied on symbolic logic and strict rules. These agents excelled in logical reasoning but struggled to handle the complexities and uncertainties of the real world. Reactive agents [51, 52], on the other hand, operated efficiently through a perception-action loop mechanism, enabling immediate responses to environmental. With advancements in deep learning and computational hardware, reinforcement learning (RL) agents emerged. These agents continually interact with their environment, learning to make decisions that maximize cumulative expected rewards. Unlike symbolic and reactive agents, RL agents [53, 54, 55, 56] can adapt to complex and dynamically changing environments. However, they also face challenges such as low sample efficiency and unstable training processes.

早期的智能体方法主要集中在符号智能体和反应式智能体上。符号智能体以基于知识的专家系统 [47, 48, 49, 50] 为代表，依赖于符号逻辑和严格的规则。这些智能体在逻辑推理方面表现出色，但难以处理现实世界的复杂性和不确定性。另一方面，反应式智能体 [51, 52] 通过感知 - 行动循环机制高效运行，能够对环境做出即时响应。随着深度学习和计算硬件的发展，强化学习 (RL) 智能体应运而生。这些智能体不断与环境进行交互，学习做出能够最大化累积预期奖励的决策。与符号智能体和反应式智能体不同，强化学习智能体 [53, 54, 55, 56] 能够适应复杂且动态变化的环境。然而，它们也面临着样本效率低和训练过程不稳定等挑战。

As LLMs have demonstrated remarkable capabilities, researchers [21, 22, 57 have begun exploring the potential of agents driven by LLMs. LLM-based Agents have several key abilities: planning, memory, and tool usage [58]. Before executing tasks, these agents utilize Chain of Thoughts (CoT) [59] and Tree of Thoughts (ToT) [60] for planning. The CoT technique guides the agent to think step-by-step, decomposing complex tasks into subtasks. In contrast, ToT generates multiple thoughts at each step, creating a tree structure for thorough exploration.

由于大语言模型展现出了卓越的能力，研究人员 [21, 22, 57] 开始探索由大语言模型驱动的智能体的潜力。基于大语言模型的智能体具备几项关键能力: 规划、记忆和工具使用 [58]。在执行任务之前，这些智能体利用思维链 (CoT)[59] 和思维树 (ToT)[60] 进行规划。思维链技术引导智能体逐步思考，将复杂任务分解为子任务。相比之下，思维树在每一步生成多个思维，形成树状结构以进行全面探索。

During task execution, agents employ ReAct [61] and Reflexion [62] methodologies, iteratively refining past decisions and correcting errors to accomplish tasks. Agents also leverage both Short Term Memory and Long Term Memory to assist in task completion. Short Term Memory is used for ICL [27], where the content is fed into the agent along with the task context, albeit limited by the transformer context window size. Long Term Memory, stored in external vector databases, is accessible via rapid retrieval, providing a broader context beyond the immediate task. Furthermore, agents enhance their action space by configuring and utilizing tools relevant to the task at hand. They call these tools by outputting the correct API formats, thereby expanding their functional capabilities.

在任务执行过程中，智能体采用反应式行动 (ReAct)[61] 和反思 (Reflexion)[62] 方法，迭代地改进过去的决策并纠正错误以完成任务。智能体还利用短期记忆和长期记忆来协助完成任务。短期记忆用于上下文学习 (ICL)[27]，其内容会与任务上下文一起输入到智能体中，但会受到 Transformer 上下文窗口大小的限制。长期记忆存储在外部向量数据库中，可以通过快速检索访问，提供超出当前任务的更广泛上下文。此外，智能体通过配置和使用与当前任务相关的工具来扩展其行动空间。它们通过输出正确的应用程序编程接口 (API) 格式来调用这些工具，从而扩展其功能能力。

While single-agent systems have shown remarkable performance, some researchers [63] have integrated multiple agents to enhance overall capabilities. Currently, multiagent system paradigms are generally classified into three types: cooperation, discussion, and competition [64]. In the cooperative paradigm [65], agents work together to achieve a common goal or complete a task collectively. In the debate paradigm [66], each agent presents its own opinion, and through exchanging viewpoints, they reach a consensus. In the competitive paradigm [67], agents pursue their individual goals, which are often in conflict with each other, thus creating a competitive environment.

尽管单智能体系统表现出显著的性能，一些研究者 [63] 已整合多智能体以增强整体能力。目前，多智能体系统范式通常分为三类: 合作、讨论和竞争 [64]。在合作范式 [65] 中，智能体协同工作以实现共同目标或集体完成任务。在讨论范式 [66] 中，每个智能体提出自身观点，通过观点交流达成共识。在竞争范式 [67] 中，智能体追求各自目标，这些目标常常相互冲突，从而形成竞争环境。

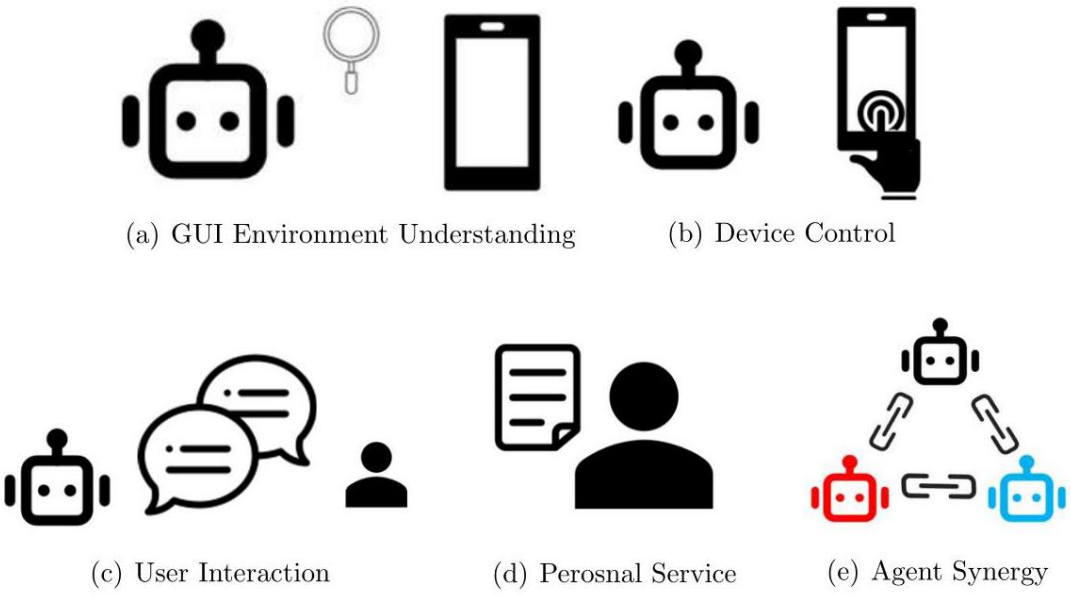


Figure 2: The core capability of GUI Agent

图 2:GUI 智能体的核心能力

3.The core capability of GUI Agent

3.GUI 智能体的核心能力

Based on the GUI Agent role in task automation as depicted in Figure 1, this section explores the core capability of GUI Agents. These capabilities are divided into two categories: basic and advanced. The basic capabilities include GUI environment comprehension, device control, and user interaction. GUI Agents must have these three basic capabilities to achieve fundamental task automation. Section 3.1 delves into these basic capabilities. Moreover, Section 3.2 explores two advanced capabilities: personalized services and agent synergy. We illustrate these five capabilities in the Figure 2.

基于图 1 所示的 GUI 智能体在任务自动化中的角色，本节探讨 GUI 智能体的核心能力。这些能力分为两类: 基础能力和高级能力。基础能力包括 GUI 环境理解、设备控制和用户交互。GUI 智能体必须具备这三项基础能力以实现基本的任务自动化。第 3.1 节深入探讨这些基础能力。此外，第 3.2 节探讨两项高级能力: 个性化服务和智能体协同。我们在图 2 中展示了这五项能力。

Table 1: Summary of GUI Agents. Distinguish the existing works based on their platform, the methods they use for GUI environment understanding, the types of control devices they employ, and the publication year.

表 1:GUI 智能体综述。根据其平台、用于 GUI 环境理解的方法、所使用的控制设备类型及发表年份区分现有工作。

Platform	Work	GUI Environment Understanding	Control Device	Publication Year
Mobile	SpotLight [68]	Vision-Based	UI-based	2022
	Meta-GUI 69	Vision-Based	UI-based	2022
	AutoUI 70	Vision-Based	UI-based	2023
	MobileGPT 71	Vision-Based	UI-based	2023
	AppAgent 72	Hybrid Text-Vision	UI-based	2023
	MM-Navigator 24	Vision-Based	UI-based	2023
	DroidBot-GPT 73	Text-Based	UI-based	2023
	Mobile-Agent 74	Vision-Based	UI-based	2024
	Mobile-Agent-v2 75	Vision-Based	UI-based	2024
	SeeClick [76]	Vision-Based	UI-based	2024
	CocoAgent [77]	Vision-Based	UI-based	2024
Computer	WebGPT [78]	Text-Based	UI-based	2021
	Pix2Act 79	Vision-Based	UI-based	2023
	WebGUM [80]	Hybrid Text-Vision	UI-based	2023
	MindAct 81	Text-Based	UI-based	2024
	WebAgent 82	Text-Based	Code-based	2023
	RCI 83	Text-Based	UI-based	2024
	AdaPlanner 84	Text-Based	Code-based	2024
	WEBWISE 25	Vision-Based	Code-based	2023
	SeeAct 85	Hybrid Text-Vision	UI-based	2024
	WebVoyager 86	ScreenShot	UI-based	2024
	DUAL-VCR 87	Hybrid Text-Vision	UI-based	2024
	UFO 88	Vision-Based	UI-based	2024
	AutoWebGLM [89]	Text-Based	UI-based	2024
	MMAC-Copilot 90	Vision-Based	UI-based, Code-based	2024

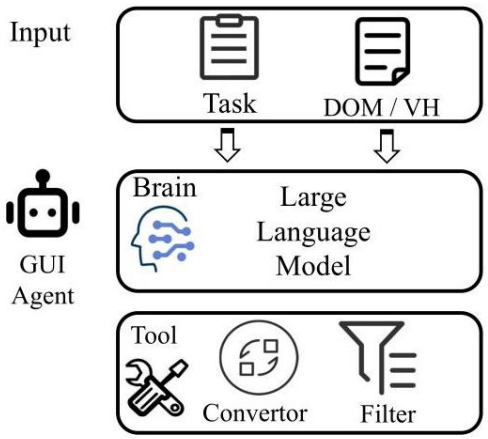
平台	工作	图形用户界面 (GUI) 环境理解	控制设备	出版年份
移动设备	聚光灯 (SpotLight) [68]	基于视觉的	基于用户界面 (UI) 的	2022
	元图形用户界面 (Meta - GUI) 69	基于视觉的	基于用户界面 (UI) 的	2022
	自动用户界面 (AutoUI) 70	基于视觉的	基于用户界面 (UI) 的	2023
	移动生成预训练变换器 (MobileGPT) 71	基于视觉的	基于用户界面 (UI) 的	2023
	应用程序代理 (AppAgent) 72	混合文本 - 视觉	基于用户界面 (UI) 的	2023
	多模态导航器 (MM - Navigator) 24	基于视觉的	基于用户界面 (UI) 的	2023
	安卓机器人生成预训练变换器 (DroidBot - GPT) 73	基于文本的	基于用户界面 (UI) 的	2023
	移动代理 (Mobile - Agent) 74	基于视觉的	基于用户界面 (UI) 的	2024
	移动代理 v2(Mobile - Agent - v2) 75	基于视觉的	基于用户界面 (UI) 的	2024
	查看点击 (SeeClick) [76]	基于视觉的	基于用户界面 (UI) 的	2024
	可可代理 (CocoAgent) [77]	基于视觉的	基于用户界面 (UI) 的	2024
计算机	网络生成预训练变换器 (WebGPT) [78]	基于文本的	基于用户界面 (UI) 的	2021
	像素到动作 (Pix2Act) 79	基于视觉的	基于用户界面 (UI) 的	2023
	网络通用理解模型 (WebGUM) [80]	混合文本 - 视觉	基于用户界面 (UI) 的	2023
	思维动作 (MindAct) 81	基于文本的	基于用户界面 (UI) 的	2024
	网络代理 (WebAgent) 82	基于文本的	基于代码的	2023
	远程控制接口 (RCI) 83	基于文本的	基于用户界面 (UI) 的	2024
	自适应规划器 (AdaPlanner) 84	基于文本的	基于代码的	2024
	网络智慧 (WEBWISE) 25	基于视觉的	基于代码的	2023
	查看动作 (SeeAct) 85	混合文本 - 视觉	基于用户界面 (UI) 的	2024
	网络航行者 (WebVoyager) 86	屏幕截图	基于用户界面 (UI) 的	2024
	双视觉 - 认知推理 (DUAL - VCR) 87	混合文本 - 视觉	基于用户界面 (UI) 的	2024
	不明飞行物 (UFO) 88	基于视觉的	基于用户界面 (UI) 的	2024
	自动网络大语言模型 (AutoWebGLM) [89]	基于文本的	基于用户界面 (UI) 的	2024
	多模态动作协同 (MMAC - Copilot) 90	基于视觉的	基于用户界面 (UI) 的，基于代码的	2024

3.1. Basic capability

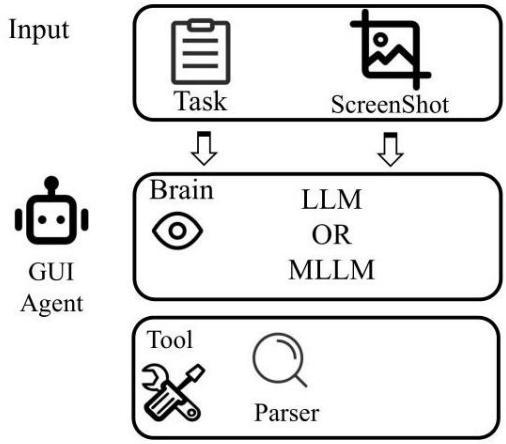
3.1. 基本能力

3.1.1. GUI environment comprehension

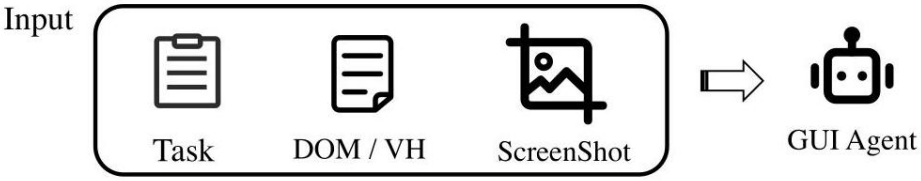
3.1.1. GUI 环境理解



(a) Text-Based Understanding by VH or DOM structure information



(b) Vision-Based Understanding by screenshot



(c) Hybrid Text-Vision Based Understanding

Figure 3: Three Basic Paradigms for Understanding GUI Environments

图 3: 理解 GUI 环境的三种基本范式

The GUI environment comprehension is one of the essential foundational capabilities of GUI Agents and the basis for its name origin. In our paper, Figure 3 illustrates three fundamental approaches to understanding the GUI environment: Text-Based, Vision-Based, and Hybrid Text-Vision. As shown in Figure 3(a), the Text-Based approach involves directly understanding the VH/DOM structure representation. The Vision-Based approach, as depicted in Figure 3(b), interprets the GUI environment through screenshots. The Hybrid Text-Vision approach, as shown in Figure 3(c), combines the characteristics of both methods to comprehend the GUI environment. Table 1 in our paper summarizes the methods employed by GUI Agents to understand their environments.

GUI 环境理解是 GUI 代理 (GUI Agents) 的一项基本核心能力，也是其名称的由来。本文中，图 3 展示了理解 GUI 环境的三种基本方法: 基于文本、基于视觉和混合文本-视觉。如图 3(a) 所示，基于文本的方法涉及直接理解 VH/DOM 结构表示。基于视觉的方法，如图 3(b) 所示，通过截图来解读 GUI 环境。混合文本-视觉方法，如图 3(c) 所示，结合了两种方法的特点来理解 GUI 环境。本文表 1 总结了 GUI 代理用于理解其环境的方法。

In the textual paradigm for understanding the GUI environment, GUI Agents typically do not directly input the raw information as text into the LLMs. GUI Agents preprocesses the text through the Convertor and Filter stage. The Convertor stage transforms the raw input into a form that is more comprehensible to the LLM. This stage generally applies to non-HTML form textual input. These studies [91, 73] set the Convertor stage to transform the raw input into HTML or natural language form. Wang et al. [91] employed a depth-first search traversal to convert the Android UI VH structure representation into HTML, which was then fed into PaLM [31. DroidBot-GPT [73] converted structural representation into natural language sentences for the LLM input. DroidBot-GPT first used Droidbot [92] to extract structure representation from the Android UI VH, then translated this information into understandable natural language sentences, which were finally input into ChatGPT [16]. The Filter stage aims to extract task-relevant text content from lengthy raw inputs. These studies [81, 82] set the Filter step to task-relevant text content. MindAct [81] investigated using HTML text as input. To address the issue of lengthy HTML text, MindAct proposed a two-stage paradigm. Mind2Act used a small language model (DeBERTa [93]) first to filter raw text and obtain candidate elements. MindAct combined these elements into HTML fragments, which were input into the LLM (Flan-T5 [94]) to predict actions. WebAgent [82] inherit the two-stage paradigm of Min-dAct. WebAgent first utilized a domain-specific model HTML-T5 to summarize task-relevant snippets from HTML documents, then used FLAN-U-PaLM [31, 94] to generate Python programs from these snippets to perform actions on the website.

在基于文本的 GUI 环境理解范式中，GUI 代理通常不会将原始信息直接作为文本输入大型语言模型 (LLMs)。GUI 代理通过转换器 (Convertor) 和过滤器 (Filter) 阶段对文本进行预处理。转换器阶段将原始输入转化为更易被 LLM 理解的形式，该阶段通常适用于非 HTML 格式的文本输入。这些研究 [91, 73] 将转换器阶段设置为将原始输入转换为 HTML 或自然语言形式。Wang 等 [91] 采用深度优先搜索遍历将 Android UI 的 VH 结构表示转换为 HTML，然后输入 PaLM[31]。DroidBot-GPT[73] 将结构表示转换为自然语言句子作为 LLM 输入。DroidBot-GPT 首先使用 Droidbot[92] 从 Android UI VH 中提取结构表示，再将其翻译成可理解的自然语言句子，最终输入 ChatGPT[16]。过滤器阶段旨在从冗长的原始输入中提取与任务相关的文本内容。这些研究 [81, 82] 将过滤步骤设置为提取任务相关文本内容。MindAct[81] 研究了使用 HTML 文本作为输入。为解决 HTML 文本过长的问题，MindAct 提出了两阶段范式。Mind2Act 先用小型语言模型 (DeBERTa[93]) 过滤原始文本，获得候选元素。MindAct 将这些元素组合成 HTML 片段，输入 LLM(Flan-T5[94]) 以预测动作。WebAgent[82] 继承了 MindAct 的两阶段范式。WebAgent 首先利用领域特定模型 HTML-T5 从 HTML 文档中总结任务相关片段，然后使用 FLAN-U-PaLM[31, 94] 从这些片段生成 Python 程序以执行网站操作。

GUI Agents show potential in understanding GUI environment through VH/DOM structure representation, but several challenges persist in text-based approach. First, VH/DOM structure representation is not always accessible. Second, the verbosity of such structured representation creates inefficient contexts for LLMs, leading to the omission of crucial details. Finally, significant information including icons, images, charts, and spatial relationships cannot be easily conveyed through VH/DOM structure representation alone. To address the limitations of the text-based approach, agents use screenshots to understand the GUI environment visually. Figure 3(b) in this paper illustrates this approach where agents comprehend the GUI environment through visual perception. Specifically, agents can achieve this paradigm in two ways: (a) Parsing the screenshot using external tools; (b) Direct Understanding via MLLMs.

GUI 代理在通过 VH/DOM 结构表示理解 GUI 环境方面展现出潜力，但基于文本的方法仍存在若干挑战。首先，VH/DOM 结构表示并非总是可获取。其次，此类结构化表示的冗长性导致 LLM 上下文效率低下，可能遗漏关键信息。最后，诸如图标、图像、图表及空间关系等重要信息难以仅通过 VH/DOM 结构表示传达。为克服基于文本方法的局限，代理采用截图以视觉方式理解 GUI 环境。本文图 3(b) 展示了该方法，代理通过视觉感知理解 GUI 环境。具体而言，代理可通过两种方式实现此范式:(a) 使用外部工具解析截图；(b) 通过多模态大型语言模型 (MLLMs) 直接理解。

WebWISE [25] leverages parsers to convert GUI interfaces into textual elements, which are then input into the LLM. Specifically, WebWISE employs Pix2Struct [95] to extract the DOM structure representation from web screenshots and subsequently inputs this information into ChatGPT to generate the Python script. Pix2Struct is a specialized MLLM designed for web screenshot simplification. Pix2struct has two pre-training tasks, web screenshot mask completion and web screenshot structure information extraction. Compared to WebWISE which relies on external visual tools, these approaches [79, 70, 68] utilize the multi-modal capability of MLLMs to directly interpret screenshots. Pix2Act [79] fine-tuned Pix2Struct to output action operation. Auto-GUI [70] train an MLLM from scratch. Auto-GUI employs the cross-attention mechanism to query screenshot clues with command text token, thereby avoiding the additional computational cost brought by visual tokens. Spotlight [68] introduces an additional region summary in the MLLM, which directs the MLLM to focus more on key areas of the screen-shot. The ability of MLLMs is sufficient to help agents understand contextual information. However, these agents face two challenges: fine-grained localization and high-resolution input. Screenshots often contain small text links and icons that require fine-grained localization, ”The fine-grained localization capability of MLLMs is important for task execution. Moreover, the high resolution of screenshots poses a challenge. This challenge mainly involves

increased computational resource overhead and complexity. For instance, encoding a 1120 × 1120 screenshot with a 14-patch setting transformer architecture would require an additional 6400 visual tokens.

WebWISE [25] 利用解析器将图形用户界面转换为文本元素，然后输入大型语言模型 (LLM)。具体来说，WebWISE 采用 Pix2Struct [95] 从网页截图中提取 DOM 结构表示，随后将该信息输入 ChatGPT 生成 Python 脚本。Pix2Struct 是一种专门针对网页截图简化设计的多模态大型语言模型 (MLLM)。Pix2Struct 有两个预训练任务，分别是网页截图掩码补全和网页截图结构信息提取。相比依赖外部视觉工具的 WebWISE，这些方法 [79, 70, 68] 利用 MLLM 的多模态能力直接解读截图。Pix2Act [79] 对 Pix2Struct 进行了微调，使其输出操作动作。Auto-GUI [70] 从零开始训练一个 MLLM。Auto-GUI 采用交叉注意力机制，用命令文本标记查询截图线索，从而避免了视觉标记带来的额外计算开销。Spotlight [68] 在 MLLM 中引入了额外的区域摘要，引导模型更关注截图的关键区域。MLLM 的能力足以帮助代理理解上下文信息。然而，这些代理面临两个挑战: 细粒度定位和高分辨率输入。截图通常包含需要细粒度定位的小文本链接和图标，“MLLM 的细粒度定位能力对任务执行至关重要。此外，截图的高分辨率也带来了挑战，主要涉及计算资源开销和复杂性的增加。例如，使用 14 补丁设置的变换器架构对 1120 × 1120 的截图进行编码，将需要额外的 6400 个视觉标记。

The challenge of fine-grained localization is not unique to GUI Agents, but is a common issue across most MLLMs [96]. Researchers [97, 98, 99, 100, 101, 102] have proposed various strategies to enhance the fine-grained localization capability of MLLMs. For commercial like GPT-4V [103], the ”Image Caption” strategy [97, 98, 99] is commonly employed. In Figure 4, we illustrate the application of the ”Image Caption” strategy. This strategy involves using a red bounding box to highlight elements in the original image and annotating them with red English letters. This approach is widely adopted in research [24, 72, 85, 76, 71] utilizing commercial models like GPT-4V as the GUI Agent brain. For instance, MM-navigator [24] and AppAgent [72] first use iconnet [104] to locate icons, marking them with numbers before inputting the annotated images into GPT-4V for interpretation. However, excessive visual markers can clutter the original image, hindering the MLLM’s comprehension. SeeAct [85] adopts a more refined method by using MindAct to filter out candidate element before applying the ”Image Caption” strategy. For trainable open-source MLLMs, researches [102, 105] have developed a more intuitive strategy to enhance fine-grained visual localization. This involves having MLLMs output the coordinates of target bounding boxes in text form. Both SeekClick [76] and CogAgent [106] have retrained their respective MLLMs to output these location coordinates effectively.

细粒度定位的挑战不仅限于图形用户界面代理，而是大多数 MLLM [96] 面临的普遍问题。研究人员 [97, 98, 99, 100, 101, 102] 提出了多种策略以增强 MLLM 的细粒度定位能力。对于商业模型如 GPT-4V [103]，常用的策略是“图像字幕”策略 [97, 98, 99]。在图 4 中，我们展示了“图像字幕”策略的应用。该策略通过红色边框突出原图中的元素，并用红色英文字母进行标注。这种方法在利用 GPT-4V 作为图形用户界面代理核心的研究中被广泛采用 [24, 72, 85, 76, 71]。例如，MM-navigator [24] 和 AppAgent [72] 首先使用 iconnet [104] 定位图标，用数字标记后将带注释的图像输入 GPT-4V 进行解读。然而，过多的视觉标记会使原图显得杂乱，影响 MLLM 的理解。SeeAct [85] 采用更精细的方法，先用 MindAct 过滤候选元素，再应用“图像字幕”策略。对于可训练的开源 MLLM，研究 [102, 105] 开发了更直观的策略以增强细粒度视觉定位，即让 MLLM 以文本形式输出目标边界框的坐标。SeekClick [76] 和 CogAgent [106] 都重新训练了各自的 MLLM，有效输出这些位置坐标。

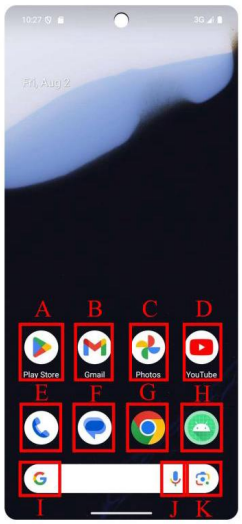


Figure 4: Enhancing Visual Localization Capabilities in MLLMs via ”Image Caption” Strategy

图 4: 通过“图像字幕”策略提升 MLLM 的视觉定位能力

Compared with the challenge of fine-grained localization, there are still fewer studies [106, 70] exploring solutions to the challenge of high-resolution input. A more intuitive approach is to directly resize the high-resolution image into a low-resolution image. However, reducing the high resolution often makes it more difficult to locate some tiny icons and text links. CogA-gent 106 additionally utilizes a lightweight high-resolution image encoder to support 1120 × 1120 high-resolution input, and uses a cross-attention mechanism to pass high-resolution information to the transformer decoder, thereby avoiding high-resolution visual tokens input. Another method [70] is to not concatenate visual tokens and text tokens together, but directly use the cross-attention mechanism to enhance text tokens with visual tokens.

相比细粒度定位的挑战，针对高分辨率输入的解决方案研究较少 [106, 70]。一种更直观的方法是直接将高分辨率图像缩放为低分辨率图像，但降低分辨率往往使定位一些微小图标和文本链接变得更困难。CogAgent [106] 额外采用轻量级高分辨率图像编码器以支持 1120 × 1120 的高分辨率输入，并利用交叉注意力机制将高分辨率信息传递给变换器解码器，从而避免了高分辨率视觉标记的输入。另一种方法 [70] 是不将视觉标记和文本标记拼接在一起，而是直接使用交叉注意力机制用视觉标记增强文本标记。

We illustrate the Hybird text-vision for understanding GUI environment in Figure 3(c), which learns both GUI screenshots and structured representation. These methods [80, 87, 86, 107] can supplement the relevant information of small page elements and text links by utilizing structured information, and can also provide visual information through screenshots. WebGUM [80] combines the visual web screenshots and linguistic HTML pages to perform web navigation tasks. Specifically, WebGUM tokenization the HTML as part of the textual information and then concatenates it with visual tokens as input to the MLLM. Similar to WebGUM, BUI-Bert [107] also simply concatenates the two types of information and inputs them into the MLLM. DUAL-VCR [87] proposed a dual-view approach that uses GUI screenshots as contextual clues for agents to better understand HTML. WebVoyager [86] uses GPT-4V as the Brain, and additionally inputs GUI layout information as a supplement while taking GUI screenshots.

我们在图 3(c) 中展示了用于理解图形用户界面 (GUI) 环境的混合文本-视觉方法，该方法同时学习 GUI 截图和结构化表示。这些方法 [80, 87, 86, 107] 可以通过利用结构化信息补充小页面元素和文本链接的相关信息，还可以通过截图提供视觉信息。WebGUM [80] 将可视化网页截图和语言化的 HTML 页面相结合, 以执行网页导航任务。具体来说, WebGUM 将 HTML 进行分词处理, 作为文本信息的一部分，然后将其与视觉标记连接起来，作为多模态大语言模型 (MLLM) 的输入。与 WebGUM 类似，BUI-Bert [107] 也只是简单地将这两种信息连接起来，并输入到 MLLM 中。DUAL-VCR [87] 提出了一种双视图方法, 该方法使用 GUI 截图作为代理更好理解 HTML 的上下文线索。WebVoyager [86] 使用 GPT - 4V 作为核心，并在获取 GUI 截图的同时，额外输入 GUI 布局信息作为补充。

3.1.2. Device control

3.1.2. 设备控制

Compared to using text directly for input and output, some studies [108], 109, 110, 111 propose using the LLM to synthesize programs. Another fundamental capability of a GUI Agents is device control. It is very intuitive to categorize GUI Agents into two types from the device control perspective: Code-based agents and UI-based agents [112]. Code-based agents either rely on the LLM to generate suitable code for interacting with device APIs [82] or are fine-tuned to learn how to call device APIs directly [78]. For instance, We-bAgent [82] extracts task-related segments from HTML text and uses these to generate Python scripts to perform operations on websites. Similarly, We-bGPT [78] fine-tunes a GPT-3 [27] to answer long-form questions by calling the Microsoft Bing Web Search API [113]. However, not all application APIs are accessible, and there can be significant differences between APIs of different applications. UI-based agents overcome these limitations by mimicking human actions to control devices. UI-based agents require a pre-defined set of actions as an action space and then select these actions in natural language form. These agents perform device control by simulating human interactions with the GUI. Currently, the majority of GUI Agents [91, 83] employ this control method. We illustrate in Table 1 in this paper presents the methods used by GUI Agents to control the device.

与直接使用文本进行输入和输出相比，一些研究 [108, 109, 110, 111] 提出使用大语言模型 (LLM) 来合成程序。GUI 代理的另一项基本能力是设备控制。从设备控制的角度来看，将 GUI 代理直观地分为两类: 基于代码的代理和基于用户界面 (UI) 的代理 [112]。基于代码的代理要么依靠 LLM 生成适合与设备应用程序编程接口 (API) 交互的代码 [82]，要么经过微调以直接学习如何调用设备 API [78]。例如，WebAgent [82] 从 HTML 文本中提取与任务相关的片段，并使用这些片段生成 Python 脚本，以在网站上执行操作。类似地，WebGPT [78] 对 GPT - 3 [27] 进行微调，通过调用微软必应网页搜索 API [113] 来回答长篇问题。然而，并非所有应用程序的 API 都是可访问的，而且不同应用程序的 API 之间可能存在显著差异。基于 UI 的代理通过模仿人类行为来控制设备，从而克服了这些限制。基于 UI 的代理需要一个预定义的动作集作为动作空间，然后以自然语言形式选择这些动作。这些代理通过模拟人类与 GUI 的交互来执行设备控制。目前，大多数 GUI 代理 [91, 83] 采用这种控制方法。本文表 1 展示了 GUI 代理用于控制设备的方法。

3.1.3. User interaction

3.1.3. 用户交互

User interaction is the final basic capability of GUI Agents. From the research perspective, GUI Agents generally support interaction with users in the form of text-based communication. This work [112] mentions additional interaction methods, such as voice interaction, GUI interaction, and virtual reality (VR) interaction. Voice interaction is considered the most popular form of interaction, widely accepted to human communication. Traditional virtual assistants [4, 7] have demonstrated the popularity of this interaction method. GUI interaction involves interacting with agents through actions like clicking buttons or selecting menus. Finally, the VR interaction method is based on an immersive 3D environment, where users can interact using headsets and gesture controllers.

用户交互是 GUI 代理的最后一项基本能力。从研究的角度来看，GUI 代理通常支持以基于文本的通信形式与用户进行交互。这项工作 [112] 提到了其他交互方法，如语音交互、GUI 交互和虚拟现实 (VR) 交互。语音交互被认为是最受欢迎的交互形式，被广泛应用于人类交流中。传统的虚拟助手 [4, 7] 已经证明了这种交互方法的受欢迎程度。GUI 交互涉及通过点击按钮或选择菜单等操作与代理进行交互。最后，VR 交互方法基于沉浸式 3D 环境，用户可以使用头戴式设备和手势控制器进行交互。

From another perspective of interaction forms, most GUI Agents nowadays interact with users in the form of simple dialogue (Text or voice). Most existing GUI Agents [81, 82, 114] follow a basic workflow: the user provides an initial task command to the GUI Agents. And then agent automatically completes the specified task and return the final status back to the user. However, this basic workflow imposes two requirements on the user: (1) the user must provide all task-related information at the beginning, and (2) the user must understand how the GUI environment operates to ensure task success. In fact, users may not have all the necessary information at the beginning. Meanwhile, if the agent initially insists that users think exhaustively about what additional information is required to complete the task, it will significantly diminish the user experience. To address this issues, researchers [69, 115, 116] have proposed the concept of multi-turn dialogues. Building on multi-turn dialogues, WebLINX 115 introduced the concept of task-oriented dialogue. Figure 5 in this paper illustrates a comparison between simple dialogue and task-oriented dialogue. Figure 5(a) shows the single-turn interaction method, while Figure 5(b) shows the task-oriented dialogue method. In task-oriented dialogues, the agent engages in a conversational exchange with the user, gradually acquiring the necessary information to complete the task, thus enhancing the overall user experience by making the interaction more intuitive and less burdensome. Task-oriented dialogues also improves the success rate of the task.

从交互形式的另一个角度来看，如今大多数 GUI 代理以简单对话 (文本或语音) 的形式与用户进行交互。大多数现有的 GUI 代理 [81, 82, 114] 遵循一个基本工作流程: 用户向 GUI 代理提供初始任务命令。然后代理自动完成指定任务，并将最终状态返回给用户。然而，这个基本工作流程对用户提出了两个要求:(1) 用户必须在开始时提供所有与任务相关的信息；(2) 用户必须了解 GUI 环境的操作方式，以确保任务成功。实际上，用户在开始时可能没有所有必要的信息。同时，如果代理一开始就要求用户详尽地考虑完成任务所需的额外信息，这将显著降低用户体验。为了解决这些问题，研究人员 [69, 115, 116] 提出了多轮对话的概念。基于多轮对话，WebLINX [115] 引入了面向任务的对话概念。本文图 5 展示了简单对话和面向任务的对话之间的比较。图 5(a) 显示了单轮交互方法，而图 5(b) 显示了面向任务的对话方法。在面向任务的对话中，代理与用户进行对话交流，逐步获取完成任务所需的信息，从而使交互更加直观，减轻用户负担，提高整体用户体验。面向任务的对话还提高了任务的成功率。

3.2. Advanced capability

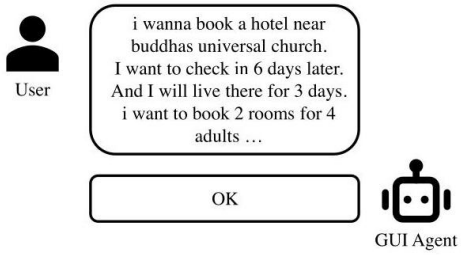
3.2. 高级能力

3.2.1. Personalized services

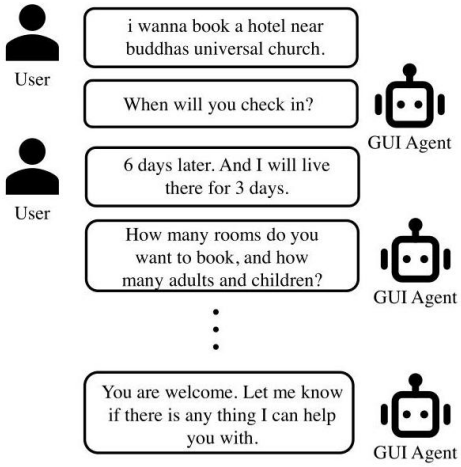
3.2.1. 个性化服务

Personalization capability is one of the advanced capabilities. While it may seem to be in conflict with privacy protection, it is relatively important in task automation. Various users display distinct preferences, needs, and habits when interacting with applications or web interfaces. Personalization can accommodate these unique characteristics, thus more effectively fulfilling user needs. By employing personalization, GUI Agents can swiftly discern user intent and execute corresponding actions with greater efficiency and accuracy. Presently, personalization is usually realized through the memory modules of the agents. For instance, Friday [117] records user profiles in declarative memory, capturing preferences related to conversation style, tool usage habits, and music/video preferences. This capability allows agents to tailor their responses and actions, thereby enhancing the overall user experience.

个性化能力是一项先进能力。虽然它看似与隐私保护相冲突，但在任务自动化中却相当重要。不同用户在与应用程序或网页界面交互时，会表现出不同的偏好、需求和习惯。个性化能够适应这些独特特征，从而更有效地满足用户需求。通过采用个性化，图形用户界面 (GUI) 代理可以迅速识别用户意图，并更高效、准确地执行相应操作。目前，个性化通常通过代理的内存模块来实现。例如，Friday [117] 会在陈述性记忆中记录用户档案，捕捉与对话风格、工具使用习惯以及音乐/视频偏好相关的信息。这一能力使代理能够调整其响应和操作，从而提升整体用户体验。



(a) Single-Turn Interaction Approach



(b) Task-Oriented Dialogue Approach

Figure 5: Two Distinct User Interaction Forms

图 5: 两种不同的用户交互形式

3.2.2. Agent Synergy

3.2.2. 代理协同

As some studies 28 have applied the multi-agent concept to task automation, multi-agent systems have shown great potential. Compared to traditional single-agent approaches, multi-agent systems break down the entire task automation process into several relatively simple subtasks. Each agent focuses on a specific subtask and optimizes the process based on its unique functionality and capabilities. This refinement and division of labor make task processing more efficient and accurate. Therefore, Agent Synergy is regarded as one of the advanced capabilities of GUI Agents.

正如一些研究 [28] 将多代理概念应用于任务自动化一样，多代理系统已展现出巨大潜力。与传统的单代理方法相比，多代理系统将整个任务自动化过程分解为几个相对简单的子任务。每个代理专注于特定的子任务，并根据其独特的功能和能力优化流程。这种细化和分工使任务处理更加高效和准确。因此，代理协同被视为图形用户界面 (GUI) 代理的先进能力之一。

Currently, there are a series of studies on agent clusters in GUI Agents [88, 90, 75]. UFO [88] is specifically designed for interacting with applications within the Windows operating system. It is a dual-agent system consisting of AppAgent and ActAgent. AppAgent is responsible for selecting the appropriate application to fulfill user requests, while ActAgent performs specific actions within the selected application to meet the user's needs. MMAC-Copilot [90] aims to enhance interaction with operating systems by leveraging the collective expertise of diverse agents. MMAC-Copilot introduces a team collaboration chain, allowing each participating agent to contribute insights based on their domain knowledge, effectively reducing hallucinations caused by knowledge gaps.

目前，有一系列关于图形用户界面 (GUI) 代理中代理集群的研究 [88, 90, 75]。UFO [88] 专门设计用于与 Windows 操作系统内的应用程序进行交互。它是一个由应用代理 (AppAgent) 和动作代理 (ActAgent) 组成的双代理系统。应用代理负责选择合适的应用程序以满足用户请求，而动作代理则在所选应用程序内执行特定操作以满足用户需求。MMAC - Copilot [90] 旨在通过利用不同代理的集体专业知识来增强与操作系统的交互。MMAC - Copilot 引入了团队协作链，使每个参与的代理能够根据其领域知识提供见解，有效减少因知识差距导致的幻觉。

In contrast to UFO and MMAC-Copilot, which are multi-agent frameworks for the computer environment, Mobile-Agent-v2 [75] is a multi-agent framework designed for mobile devices. Mobile-Agent-v2 consists of three specialized agents: Planning Agent, Decision Agent, and Reflection Agent. The Planning Agent compresses historical operations and screen summaries into a plain text task progress format, making it easier for the Decision Agent to navigate the task flow. The Decision Agent generates and executes actions based on the current task progress, the current screen state, and feedback from the Reflection Agent (if the previous action was incorrect). The Reflection Agent monitors the screen changes before and after the Decision Agent's actions to determine whether the actions were successful. If the actions are not as expected, it takes appropriate measures to re-execute the actions.

与适用于计算机环境的多代理框架 UFO 和 MMAC - Copilot 不同，Mobile - Agent - v2 [75] 是一个为移动设备设计的多代理框架。Mobile - Agent - v2 由三个专门的代理组成: 规划代理 (Planning Agent)、决策代理 (Decision Agent) 和反思代理 (Reflection Agent)。规划代理将历史操作和屏幕摘要压缩为纯文本的任务进度格式，使决策代理更易于导航任务流程。决策代理根据当前任务进度、当前屏幕状态以及反思代理的反馈 (如果前一个操作不正确) 生成并执行操作。反思代理监控决策代理操作前后的屏幕变化，以确定操作是否成功。如果操作未达到预期，它会采取适当措施重新执行操作。

4. Task Automation Pipeline

4. 任务自动化流程

This section discusses a key issue: ”How can agents efficiently automate complex tasks?” Most task automation agents follow a single-stage automation pipeline, where automation relies solely on pre-trained knowledge or task information provided by humans. Recently, researchers [28] proposed a dual-agent system that divides task automation into two key stages: from Exploration to Exploitation. In the Exploration stage, a weaker agent randomly explores the environment to gather and record knowledge, while in the Exploitation stage, a stronger agent utilizes the knowledge gained during the Exploration stage to execute tasks. GUI Agents task automation can also follow this two-stage pipeline: from Exploration to Exploitation. The two stages is shown in Figure 6 In the Exploration stage, the GUI Agents explore an unfamiliar GUI environment, recording interaction methods and actionable subtasks. In the Exploitation stage, GUI Agents utilize the knowledge acquired during the Exploration stage to execute tasks. We believe that exploration is an essential step in efficient task automation processes. Given the diversity of currently available applications and websites, it is impractical to collect a large amount of data for training in every unfamiliar GUI environment. In addition, the operation methods behind each GUI environment are different, and the same icon may represent different functions in different GUI contexts. The exploration stage provides a feasible solution to these challenges by requiring GUI Agents to explore the GUI environment before executing tasks.

本节讨论一个关键问题:“代理如何高效地自动化复杂任务?”大多数任务自动化代理遵循单阶段自动化流程，其中自动化仅依赖于预训练的知识或人类提供的任务信息。最近，研究人员 [28] 提出了一个双代理系统，将任务自动化分为两个关键阶段: 从探索到利用。在探索阶段，一个较弱的代理随机探索环境以收集和记录知识，而在利用阶段，一个较强的代理利用探索阶段获得的知识来执行任务。图形用户界面 (GUI) 代理的任务自动化也可以遵循这个两阶段流程: 从探索到利用。这两个阶段如图 6 所示。在探索阶段，图形用户界面 (GUI) 代理探索陌生的图形用户界面 (GUI) 环境，记录交互方法和可操作的子任务。在利用阶段，图形用户界面 (GUI) 代理利用探索阶段获得的知识来执行任务。我们认为，探索是高效任务自动化过程中不可或缺的一步。鉴于当前可用的应用程序和网站的多样性，在每个陌生的图形用户界面 (GUI) 环境中收集大量数据进行训练是不切实际的。此外，每个图形用户界面 (GUI) 环境背后的操作方法不同，同一个图标在不同的图形用户界面 (GUI) 上下文中可能代表不同的功能。探索阶段通过要求图形用户界面 (GUI) 代理在执行任务前探索图形用户界面 (GUI) 环境，为这些挑战提供了一个可行的解决方案。

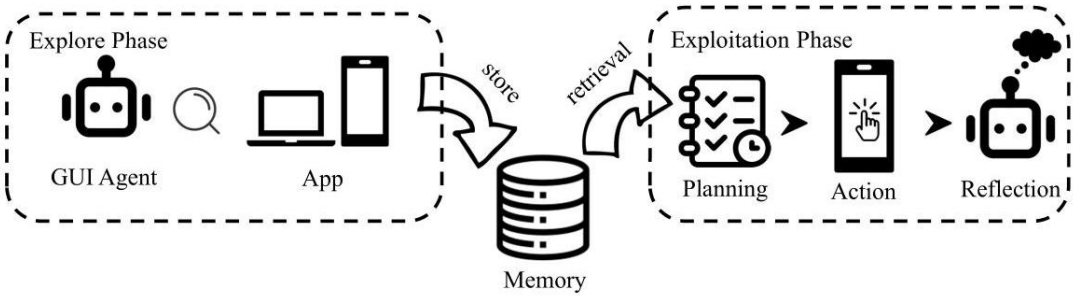


Figure 6: Two-Stage Task Automation Pipeline of GUI Agents: From Exploration to Exploitation.

图 6: 图形用户界面 (GUI) 代理的两阶段任务自动化流程: 从探索到利用。

Currently, many researchers [72, 71, 26] have integrated the exploration stage before GUI Agents perform tasks. For example, AppAgent [72] explores the functionality and characteristics of smartphone applications through trial and error. At this stage, AppAgent is assigned a task and begins to interact autonomously with UI elements, using different operations and observing changes in the application interface to understand its working principle. Mo-bileGPT [71] uses two software tools during the exploration stage - a random explorer [118] and a user trace monitor [119] - to access and analyze as many application screens as possible. For each screen accessed, MobileGPT requests the LLM to generate a list of available subtasks on that screen for use during the deployment stage. AutoDroid [26] summarized the functionality of all UI elements through random exploration to gain a comprehensive understanding of the tasks that can be executed in an application and determine the corresponding UI elements required to perform these tasks.

目前，许多研究者 [72, 71, 26] 已将探索阶段整合到 GUI 代理执行任务之前。例如，AppAgent[72] 通过试错法探索智能手机应用的功能和特性。在此阶段，AppAgent 被分配任务，开始自主与 UI 元素交互，使用不同操作并观察应用界面变化，以理解其工作原理。MobileGPT[71] 在探索阶段使用两个软件工具——随机探索器 [118] 和用户轨迹监控器 [119]——以访问和分析尽可能多的应用界面。对于每个访问的界面，MobileGPT 请求大语言模型 (LLM) 生成该界面上可用子任务列表，以供部署阶段使用。AutoDroid[26] 通过随机探索总结了所有 UI 元素的功能，以全面了解应用中可执行的任务，并确定执行这些任务所需的对应 UI 元素。

During the exploitation stage, GUI Agents can further refine tasks to improve their success rate. This process typically includes the following steps: planning, action, and reflection 117. The GUI Agent first executes task planning, which may break down the task into several sub-tasks and complete them one by one. During the execution of subtasks, the GUI Agent utilizes knowledge obtained from memory or files during the exploration stage to assist in completing the task and continuously updates this knowledge. After each subtask is completed, the GUI Agent can set a Critic to automatically evaluate the completion status of the subtasks. GUI proxies typically use a LLM as the Critic, which not only determines whether the current subtask is completed, but also provides corrective suggestions and evaluates the necessity of reorganizing subtasks. Recently, MMAC-Copilot [90] proposed a multi-agent system to deepen the implementation of the above in the deployment stage. MMAC-Copilot has six kind agent roles: Planner, Librarian, Programmer, Viewer, Video Analyst, Mentor. Planner observes tasks and decomposes them into rough subtasks, which are then further refined by Viewer and Video Analyst. After determining the subtask, the Viewer and Programmer will execute the subtask, and the Mentor will evaluate the execution results and provide feedback to the Planner, indicating whether the subtask needs to be changed. During

this process, the Librarian is responsible for conducting information retrieval, enabling them to answer queries and provide basic knowledge.

在利用阶段，GUI 代理可以进一步细化任务以提高成功率。该过程通常包括以下步骤: 规划、行动和反思 117。GUI 代理首先执行任务规划，可能将任务拆分为若干子任务并逐一完成。在子任务执行过程中，GUI 代理利用探索阶段从记忆或文件中获得的知识辅助完成任务，并持续更新这些知识。每完成一个子任务后，GUI 代理可以设置评审者 (Critic) 自动评估子任务完成状态。GUI 代理通常使用大语言模型 (LLM) 作为评审者，不仅判断当前子任务是否完成，还提供纠正建议并评估是否有必要重新组织子任务。近期，MMAC-Copilot[90] 提出了一个多代理系统，以深化部署阶段上述过程的实现。MMAC-Copilot 包含六种代理角色: 规划者 (Planner)、图书管理员 (Librarian)、程序员 (Programmer)、观察者 (Viewer)、视频分析师 (Video Analyst)、导师 (Mentor)。规划者观察任务并将其拆解为粗略子任务，随后由观察者和视频分析师进一步细化。确定子任务后，观察者和程序员执行子任务，导师评估执行结果并向规划者反馈，指示是否需要更改子任务。在此过程中，图书管理员负责信息检索，能够回答查询并提供基础知识。

5. Evaluation Benchmark

5. 评估基准

5.1. Evaluation Dataset

5.1. 评估数据集

We present GUI automation task dataset at the current stage in Table 2. Some datasets [129, 130] initially focused on low-level GUI automation tasks, which involved selecting web page elements given human instruction. PhraseNode [129] recommends selecting the element e described by command c when provided with a set of elements $e1, \ldots, e_k$ and a web environment w . UIBert [130] introduced two downstream tasks for element selection: similar GUI component retrieval and referrring component retrieval. In similar GUI component retrieval tasks, the goal is to select the candidate components that are most functionally similar to the given GUI environment and components used as queries, and to search for the GUI and a set of candidate components. The referring component retrieval task is similar to the one proposed by PhraseNode, aiming to retrieve the components indicated by expressions from a set of GUI components detected on the screen given a reference expression and GUI image. SeekClick [76] extends this by introducing another baseline dataset called ScreenSpot, specifically designed to assess the capability of MLLMs in locating elements in GUI environment

我们在表 2 中展示了当前阶段的 GUI 自动化任务数据集。一些数据集 [129, 130] 最初聚焦于低级 GUI 自动化任务，即在给定人工指令的情况下选择网页元素。PhraseNode[129] 建议在提供元素集合 $e1, \ldots, e_k$ 和网页环境 w 时，选择由命令 c 描述的元素 e 。UIBert[130] 引入了两个元素选择的下游任务: 相似 GUI 组件检索和指代组件检索。在相似 GUI 组件检索任务中，目标是从 GUI 和候选组件集合中选择与给定 GUI 环境及用作查询的组件功能最相似的候选组件。指代组件检索任务类似于 PhraseNode 提出的任务，旨在根据参考表达和 GUI 图像，从屏幕检测到的 GUI 组件集合中检索表达所指代的组件。SeekClick[76] 通过引入另一个基线数据集 ScreenSpot 扩展了该任务，专门用于评估多模态大语言模型 (MLLMs) 在 GUI 环境中定位元素的能力。

Table 2: Overview of Datasets for GUI Task Automation. The columns indicate: the platform used (Platform), the method of environment observation (Observe), support for multi-turn dialogue (Chat), whether tasks need multistep to complete (High-Level), the number of different domains included in the dataset (Domain), and the number of instances in the dataset (Instance).

表 2:GUI 任务自动化数据集概览。各列分别表示: 所用平台 (Platform)、环境观察方式 (Observe)、是否支持多轮对话 (Chat)、任务是否需多步骤完成 (High-Level)、数据集包含的不同领域数量 (Domain) 及数据集实例数量 (Instance)。

Dataset	Platform	Observe	Chat	High-Level	Domain	Instance
Meta-GUI 69	Mobile	ScreenShot, VH	✓	✓	11	1125
MobileGPT 71	Mobile	ScreenShot, VH	✓	✓	8	160
PixelHelp 11	Mobile	ScreenShot, VH	✘	✓	-	187
RICOSCA 11	Mobile	ScreenShot, VH	✘	✘	9.7k	25,677
MoTiF 120	Mobile	ScreenShot, VH	✘	✓	125	61K
UGIF 121	Mobile	ScreenShot, VH	✘	✓	11	4184
MobileAgentBench 122	Mobile	ScreenShot, VH	✘	✓	10	100
DroidTask [123]	Mobile	ScreenShot, VH	✘	✓	13	158
AndroidEnv 118	Mobile	ScreenShot	✘	✓	30	100
MobileEnv 124	Mobile	ScreenShot, VH	✘	✓	-	856,045
MobileAgentBench 122	Mobile	ScreenShot, VH	✘	✓	10	100
LlamaTouch 125	Mobile	ScreenShot, VH	✘	✓	57	496
WebArena 126	Computer	ScreenShot, DOM	✘	✓	6	812
VWA 127	Computer	ScreenShot, DOM	✘	✓	3	910
WebVoyager 86	Computer	ScreenShot, DOM	✘	✓	15	300
WebShop 128	Computer	ScreenShot, DOM	✘	✓	1	12,087
MninWoB++ 13	Computer	ScreenShot, DOM	✘	✘	100	100
WebLINX 115	Computer	ScreenShot, DOM	✓	✓	155	2337
RUSS 116	Computer	ScreenShot, DOM	✓	✓	22	80
PharseNode 129	Computer	ScreenShot, DOM	✘	✘	-	51,663
UIBert 130	Computer	ScreenShot, DOM	✘	✘	-	16,660
Mind2Web 81	Computer	ScreenShot, DOM	✘	✓	137	2,350
AssistGUI 131	Computer	ScreenShot, Metadata	✘	✓	9	100
AITW 132	Mobile, Computer	ScreenShot	✘	✓	357	30K
ScreenSpot 76	Mobile, Computer	ScreenShot	✘	✘	-	1200

数据集	平台	观察	聊天	高级	领域	实例
元图形用户界面 69(Meta - GUI 69)	移动设备	截图，VH	✓	✓	11	1125
移动 GPT 71(MobileGPT 71)	移动设备	截图，VH	✓	✓	8	160
像素助手 11(PixelHelp 11)	移动设备	截图，VH	✘	✓	-	187
RICOSCA 11	移动设备	截图，VH	✘	✘	9.7k	25,677
MoTiF 120	移动设备	截图，VH	✘	✓	125	61K
UGIF 121	移动设备	截图，VH	✘	✓	11	4184
移动代理基准测试 122(MobileAgentBench 122)	移动设备	截图，VH	✘	✓	10	100
安卓任务 [123](DroidTask [123])	移动设备	截图，VH	✘	✓	13	158
安卓环境 118(AndroidEnv 118)	移动设备	截图	✘	✓	30	100
移动环境 124(MobileEnv 124)	移动设备	截图，VH	✘	✓	-	856,045
移动代理基准测试 122(MobileAgentBench 122)	移动设备	截图，VH	✘	✓	10	100
羊驼触摸 125(LlamaTouch 125)	移动设备	截图，VH	✘	✓	57	496
网络竞技场 126(WebArena 126)	计算机	截图，文档对象模型 (DOM)	✘	✓	6	812
VWA 127	计算机	截图，文档对象模型 (DOM)	✘	✓	3	910
网络旅行者 86(WebVoyager 86)	计算机	截图，文档对象模型 (DOM)	✘	✓	15	300
网络商店 128(WebShop 128)	计算机	截图，文档对象模型 (DOM)	✘	✓	1	12,087
MninWoB++ 13	计算机	截图，文档对象模型 (DOM)	✘	✘	100	100
网络链接 115(WebLINX 115)	计算机	截图，文档对象模型 (DOM)	✓	✓	155	2337
RUSS 116	计算机	截图，文档对象模型 (DOM)	✓	✓	22	80
短语节点 129(PharseNode 129)	计算机	截图，文档对象模型 (DOM)	✘	✘	-	51,663
用户界面伯特模型 130(UIBert 130)	计算机	截图，文档对象模型 (DOM)	✘	✘	-	16,660
思维到网络 81(Mind2Web 81)	计算机	截图，文档对象模型 (DOM)	✘	✓	137	2,350
辅助图形用户界面 131(AssistGUI 131)	计算机	截图，元数据	✘	✓	9	100
AITW 132	移动设备，计算机	截图	✘	✓	357	30K
屏幕定位点 76(ScreenSpot 76)	移动设备，计算机	截图	✘	✘	-	1200

Other datasets [81, 11, 132] have proposed high-level GUI automation tasks, which require the agent to preform multiple steps to complete instructions. For example, Li et al. [11] proposed a task that maps text instructions to the operations people need to perform in GUI environment. Given a multistep instruction I , this task entails generating a sequence of automatically executable actions $a_1 : m$ on the screenshot S . The tasks proposed in the above research focus on individual queries and step-by-step operations, but in practical scenarios, users may need to chat with the agent to provide the necessary task information. Therefore, some datasets [69, 115, 116] have proposed chat datasets for GUI task automation. In this setting, given an initial user instruction scenario, the agent must engage in a multi-turn conversation with the user to complete the task.

其他数据集 [81, 11, 132] 提出了高级 GUI 自动化任务，要求代理执行多步骤以完成指令。例如，Li 等人 [11] 提出了一个将文本指令映射到用户在 GUI 环境中需要执行的操作的任务。给定一个多步骤指令 I ，该任务需要生成一系列可自动执行的操作序列 $a_1 : m$ ，基于截图 S 。上述研究提出的任务侧重于单个查询和逐步操作，但在实际场景中，用户可能需要与代理进行对话以提供必要的任务信息。因此，一些数据集 [69, 115, 116] 提出了用于 GUI 任务自动化的对话数据集。在此设置下，给定初始用户指令场景，代理必须与用户进行多轮对话以完成任务。

In addition, AssistGUI [131] also propose a novel task (desktop task automation). In desktop task automation, agents perform a series of operations to complete user queries under a teaching video that provides more detailed instructions on how to complete tasks and related applications.

此外，AssistGUI[131] 还提出了一种新颖的任务 (桌面任务自动化)。在桌面任务自动化中，代理在教学视频的指导下执行一系列操作以完成用户查询，该视频提供了关于如何完成任务及相关应用的更详细说明。

5.2. Evaluation Metrics

5.2. 评估指标

5.2.1. Common Evaluation Criteria

5.2.1. 常用评估标准

For Low-Level tasks, PhraseNode [129] and UIBert [130] employ the proportion of correctly selected elements as evaluation metrics. Following these studies [133, 134], SeekClick [76] uses click accuracy as the metric. Click accuracy is defined as the proportion of test samples where the model-predicted location falls in the ground truth element bounding box

对于低级任务，PhraseNode[129] 和 UIBert[130] 采用正确选择元素的比例作为评估指标。继这些研究 [133, 134] 之后，SeekClick[76] 使用点击准确率作为指标。点击准确率定义为模型预测位置落在真实元素边界框内的测试样本比例。

Success rate [81, 13, 128] is one of the most common metrics for assessing GUI Agents, defined as the proportion of instances where the model reaches the desired final state. However, success rate is very strict and unable to evaluate the degree of execution of failed tasks. Step Successful Rate is proposed in Mind2Web [81], defined as the proportion of successful steps to total steps. However, these metrics are not suitable for dialogue task systems 115, 135, 136]. The objective in dialogue task systems is not fully defined in the first turn or later turns; instead, it evolves as the conversation proceeds. WebLINX [115] following established approaches [135, 137 in dialogue system leverage turn-level automatic evaluation metrics: element similarity and text similarity. Element similarity and text similarity both rely on intent matching (IM), where the given predicted action a' and reference action a , if the intent is consistent, $IM\left(a', a\right) = 1$; otherwise, $IM\left(a', a\right) = 0$.

成功率 [81, 13, 128] 是评估 GUI 代理最常用的指标之一，定义为模型达到期望最终状态的实例比例。然而，成功率标准非常严格，无法评估失败任务的执行程度。Mind2Web[81] 提出了步骤成功率，定义为成功步骤数与总步骤数的比例。但这些指标不适用于对话任务系统 [115, 135, 136]。对话任务系统的目标在首轮或后续轮次中并未完全确定，而是随着对话进展而演变。WebLINX[115] 遵循对话系统中既定方法 [135, 137]，采用基于轮次的自动评估指标: 元素相似度和文本相似度。元素相似度和文本相似度均依赖意图匹配 (IM)，即给定预测动作 a' 和参考动作 a ，若意图一致，则 $IM\left(a', a\right) = 1$ ；否则，$IM\left(a', a\right) = 0$ 。

The measure of element similarity is specific to operations that take elements as parameters (such as clicking, text input, submission). It calculates the similarity between elements by measuring the intersection over the union [138] between bounding boxes, using the following formula:

元素相似度的度量针对以元素为参数的操作 (如点击、文本输入、提交)，通过计算边界框的交并比 [138] 来衡量元素间的相似度，公式如下:

$$IM\left(a', a\right) \times \frac{B_{\text{reference}} \cup B_{\text{predicted}}}{B_{\text{reference}} \cap B_{\text{predicted}}} \tag{1}$$

Here, $B_{\text{reference}}$ and $B_{\text{predicted}}$ are the coordinates of the ground truth and predicted bounding boxes, respectively.

其中，$B_{\text{reference}}$ 和 $B_{\text{predicted}}$ 分别为真实和预测边界框的坐标。

To assess Text Similarity, the authors compute the chrF value [139] between the text generated agent and the ground truth text, i.e., the F1 score for character n-gram matching (using the default setting $n = 6$). Similar to Element Similarity, Text Similarity also scales using IM , resulting in the final formula:

为了评估文本相似度，作者计算了代理生成文本与真实文本之间的 chrF 值 [139]，即字符 n-gram 匹配的 F1 分数 (使用默认设置 $n = 6$)。与元素相似度类似，文本相似度也通过 IM 进行缩放，最终公式为:

$$IM\left(a', a\right) \times CHRF\left(a', a\right) \tag{2}$$

Finally, WebLINX evaluates the performance of the element group (EG), including click, text input, and submit, using Element Similarity, while the text group (TG), including load, say, and text input, is evaluated using Text Similarity.

最后，WebLINX 使用元素相似度评估元素组 (EG)，包括点击、文本输入和提交；使用文本相似度评估文本组 (TG)，包括加载、说话和文本输入。

5.2.2. Other Evaluation Methods

5.2.2. 其他评估方法

However, since there can be different ways to accomplish the same task, using a single operation way as the ground truth for evaluating accuracy may not be entirely correct. Therefore, some studies propose evaluation criteria that involve manually designing key step rewards [124]. While manually designing reward functions can provide a more accurate assessment of model performance, it can also be more cumbersome. In this work [86], the use of GPT-4V for automatically evaluating these trajectories. The results indicated that GPT-4V achieved an consistency of 85.3% compared to manual evaluations. This demonstrates the enormous potential of model evaluation methods.

然而，由于完成同一任务可能存在多种方式，使用单一操作方式作为准确率的评估标准可能不完全合理。因此，一些研究提出了涉及手动设计关键步骤奖励的评估标准 [124]。虽然手动设计奖励函数能更准确地评估模型性能，但过程较为繁琐。在这项工作 [86] 中，采用 GPT-4V 自动评估这些轨迹。结果显示，GPT-4V 与人工评估的一致性达到 85.3%，展示了模型评估方法的巨大潜力。

Compared to the method of directly inspecting action sequences mentioned above, MobileAgentBench [122] and LlamaTouch [125] evaluate success by comparing specific states within the task. MobileAgentBench provides a low-intrusion integration with existing agents, allowing evaluation with just a few lines of code. It determines task success by examining the final User Interface (UI) state after task execution. LlamaTouch, on the other hand, is capable of matching various UI states at different levels, ensuring a more faithful evaluation of tasks in real

dynamic environments. Table 3 presents the performance of state-of-the-art methods on the LlamaTouch and MobileAgentBench frameworks, using Success Rate (SR) and Step Success Rate (SSR) as evaluation metrics.

与上述直接检查动作序列的方法相比，MobileAgentBench [122] 和 LlamaTouch [125] 通过比较任务中的特定状态来评估成功率。MobileAgentBench 与现有智能体进行低侵入式集成，只需几行代码即可进行评估。它通过检查任务执行后的最终用户界面 (UI) 状态来确定任务是否成功。另一方面，LlamaTouch 能够在不同级别匹配各种 UI 状态，确保在真实动态环境中更真实地评估任务。表 3 展示了最先进的方法在 LlamaTouch 和 MobileAgentBench 框架上的性能，使用成功率 (SR) 和步骤成功率 (SSR) 作为评估指标。

6. Challenges and Opportunities

6. 挑战与机遇

In this section, we discuss the current challenges and opportunities of LLM-Based GUI Agents. In Section 6.1, we discuss the computation cost issues associated with GUI Agents. This issue in fact poses a significant bottleneck to GUI Agents development, particularly for GUI Agents equipped with visual capabilities. We then continue in Section 6.2 to discuss the feasibility, completeness, and security concerns in task automation. Finally, in Section 6.3, we explore the connection between AI operating system(AIOS) agents and GUI Agents.

在本节中，我们讨论基于大语言模型 (LLM) 的图形用户界面 (GUI) 智能体的当前挑战和机遇。在 6.1 节中，我们讨论与 GUI 智能体相关的计算成本问题。事实上，这个问题对 GUI 智能体的发展构成了重大瓶颈，特别是对于具备视觉能力的 GUI 智能体而言。然后在 6.2 节中，我们继续讨论任务自动化中的可行性、完整性和安全问题。最后，在 6.3 节中，我们探讨人工智能操作系统 (AIOS) 智能体与 GUI 智能体之间的联系。

6.1. Cost Issues: A Bottleneck in GUI Agents Development

6.1. 成本问题:GUI 智能体发展的瓶颈

LLM-based GUI Agents may incur significant costs in completing GUI tasks, which severely limits their commercial application. The price for calling the ChatGPT API [16] by commercial LLMs is $1.5 for inputting, 1000K tokens. Even if the Agent chooses the open-source LLM, the computational cost is still very high. For example, a 7B LLaMA [140] inference one token requires 6.7 billion floating-point operations. Completing a GUI task typically requires 2-8 steps.

基于大语言模型的 GUI 智能体在完成 GUI 任务时可能会产生巨大成本，这严重限制了它们的商业应用。商业大语言模型调用 ChatGPT API [16] 的价格是输入 1000K 个标记收费 1.5 美元。即使智能体选择开源大语言模型，计算成本仍然很高。例如，一个 70 亿参数的 LLaMA [140] 推理一个标记需要 67 亿次浮点运算。完成一个 GUI 任务通常需要 2 - 8 个步骤。

One way to reduce the cost of task automation is to use tiny backbone architecture LLMs. At present, many researchers have attempted to integrate smaller backbone architecture into the foundation model (LLMs [141], 142,

143], MLLMs [144, 145, 146]), which have smaller parameters and can even achieve results with larger parameters on specific tasks. However there is little research exploring the application of small backbone architecture in GUI automation tasks. Another way to reduce inference overhead is through LLMs compression. Currently, model compression techniques are quite mature, including quantization [147, 148], pruning [149, 150], knowledge distillation [151, 152], and low-rank decomposition [153, 154].

降低任务自动化成本的一种方法是使用小型骨干架构的大语言模型。目前，许多研究人员已尝试将更小的骨干架构集成到基础模型中 (大语言模型 [141, 142, 143]，多模态大语言模型 [144, 145, 146])，这些模型参数更少，甚至在特定任务上可以取得与大参数模型相当的结果。然而，很少有研究探索小型骨干架构在 GUI 自动化任务中的应用。另一种降低推理开销的方法是通过大语言模型压缩。目前，模型压缩技术已经相当成熟，包括量化 [147, 148]、剪枝 [149, 150]、知识蒸馏 [151, 152] 和低秩分解 [153, 154]。

Deploying large generative language models (LLMs) directly on resource-constrained hardware is infeasible due to their high computational costs. Recent research [155, 156, 157] has focused on optimizing LLMs for mobile and embedded devices. Specifically, the study [155] introduces LLMS to minimize context-switching overheads for LLMs under tight memory budgets on mobile devices. It achieves this by separating memory management of application and LLM contexts from the key ideas of fine-grained, block-level, globally optimized KV cache compression and swapping.

由于计算成本高，直接将大型生成式语言模型 (LLMs) 部署在资源受限的硬件上是不可行的。最近的研究 [155, 156, 157] 专注于为移动和嵌入式设备优化大语言模型。具体而言，研究 [155] 引入了大语言模型内存管理系统 (LLMS)，以最小化移动设备在内存预算紧张的情况下大语言模型的上下文切换开销。它通过将应用程序和大语言模型上下文的内存管理分离，结合细粒度、块级、全局优化的键值 (KV) 缓存压缩和交换的关键思想来实现这一目标。

6.2. Feasibility, Completeness, and Security in Task Automation

6.2. 任务自动化中的可行性、完整性和安全性

In task automation, feasibility, completeness, and security were mentioned early in ResponsibleTA [158]. Feasibility refers to predicting whether the low-level commands generated by the LLM are executable by the executor. If the command is not feasible, the system will request a replan to avoid executing unattainable instructions, thereby improving the controllability and efficiency of task automation. Completeness refers to checking whether the execution results of low-level commands align with the target of the given command and providing timely feedback so that the LLM coordinator can more reasonably rearrange the next steps. Lastly, security emphasizes the protection of users' sensitive information, which is especially important for GUI automation tasks involving personal devices. To ensure feasibility and completeness, ResponsibleTA proposed the Feasibility Predictor and Completeness Verifier to assess commands before and after execution. Regarding security, ResponsibleTA uses a named entity recognition (NER) model to automatically detect sensitive information in user instructions, replacing it with predefined placeholders, which are then stored in edge-deployed memory. Although these three properties are critical in task automation, few studies have explored them.

在任务自动化中，可行性、完整性和安全性在 ResponsibleTA [158] 中很早就被提及。可行性是指预测大语言模型生成的低级命令是否可由执行器执行。如果命令不可行，系统将请求重新规划，以避免执行无法实现的指令，从而提高任务自动化的可控性和效率。完整性是指检查低级命令的执行结果是否与给定命令的目标一致，并及时提供反馈，以便大语言模型协调器能够更合理地安排下一步。最后，安全性强调保护用户的敏感信息，这对于涉及个人设备的 GUI 自动化任务尤为重要。为了确保可行性和完整性，ResponsibleTA 提出了可行性预测器和完整性验证器，分别在命令执行前后对其进行评估。关于安全性，ResponsibleTA 使用命名实体识别 (NER) 模型自动检测用户指令中的敏感信息，并用预定义的占位符替换，然后将这些占位符存储在边缘部署的内存中。尽管这三个属性在任务自动化中至关重要，但很少有研究对它们进行探索。

One of the key reasons that reduces the feasibility and completeness of automated tasks is the hallucination phenomena. The hallucination effect refers to the generation of content by foundational models (LLMs [159], MLLMs [160]) that is syntactically correct but factually or logically incorrect. Hallucination phenomena can be divided into factual and faithful hallucinations [161]. Factual hallucinations can be effectively detected by retrieving external facts and estimating the uncertainty of the factual content generated by the LLM. For faithful hallucinations, a simple strategy is to use the LLM as an evaluator to assess the fidelity of the generated content. Additionally, the self-reflection mechanism can be utilized to reduce the generation of erroneous content and promote self-improvement.

降低自动化任务可行性和完整性的关键原因之一是幻觉现象。幻觉效应指基础模型 (大语言模型 (LLMs [159])、多模态大语言模型 (MLLMs [160])) 生成的内容在语法上正确但在事实或逻辑上错误。幻觉现象可分为事实性幻觉和忠实性幻觉 [161]。事实性幻觉可以通过检索外部事实和估计 LLM 生成事实内容的不确定性来有效检测。对于忠实性幻觉，一种简单策略是使用 LLM 作为评估者来评估生成内容的忠实度。此外，可以利用自我反思机制减少错误内容的生成并促进自我改进。

For privacy protection, ResponsibleTA adopts a Data Masking method, which replaces sensitive information with placeholders before sending the data to cloud inference devices. Another way to protect privacy is to localize the LLM, reducing potential risks during data transmission, though this often faces challenges related to computational resource limitations.

为了保护隐私，ResponsibleTA 采用数据掩码方法，在将数据发送到云推理设备之前用占位符替换敏感信息。另一种保护隐私的方法是本地化大语言模型，减少数据传输过程中的潜在风险，尽管这通常面临计算资源限制的挑战。

6.3.The Connection Between AIOS and GUI Agents

6.3.AIOS 与 GUI 代理的联系

In fact, AI-OS [162] is a cutting-edge concept that considers LLMs as the core of the operating system, providing a computing environment fundamentally different from traditional operating systems (OS). In this framework, the LLM serves as the kernel of the OS, responsible for managing and scheduling the system's core functions. In AI-OS, agents are treated as applications, capable of performing various tasks and demonstrating intelligent problem-solving abilities across different scenarios.

事实上，AI-OS [162] 是一个前沿概念，将大语言模型视为操作系统的核心，提供与传统操作系统 (OS) 根本不同的计算环境。在该框架中，LLM 作为操作系统的内核，负责管理和调度系统的核心功能。在 AI-OS 中，代理被视为应用程序，能够执行各种任务，并在不同场景中展现智能问题解决能力。

The GUI Agent is an AI system that can understand and operate GUI. GUI represents the primary means of interaction between users and computer programs, mobile apps, or other digital devices, typically comprising windows, buttons, menus, icons, and other elements. The goal of the GUI Agent is to automate tasks within the user interface, such as filling out forms, clicking buttons, conducting searches, or browsing the web.

GUI 代理是一种能够理解和操作图形用户界面 (GUI) 的人工智能系统。GUI 是用户与计算机程序、移动应用或其他数字设备交互的主要方式，通常包括窗口、按钮、菜单、图标等元素。GUI 代理的目标是自动化用户界面内的任务，如填写表单、点击按钮、进行搜索或浏览网页。

Intuitively, the GUI Agent can take on the role of an application within the AI-OS framework, enhancing its capabilities and functionalities through efficient management provided by this framework. For instance, in scenarios that span multiple applications, it can facilitate communication between clusters of agents.

直观地看，GUI 代理可以作为 AI-OS 框架中的一个应用，通过该框架提供的高效管理增强其能力和功能。例如，在跨多个应用的场景中，它可以促进代理集群之间的通信。

7. conclusion

7. 结论

Our review surveys recent papers on LLM-based GUI task automation, aiming to provide readers with a comprehensive overview of GUI Agents. We then summarize the capabilities of GUI Agents, covering three fundamental abilities (GUI Environment Comprehension, Device Control, and User Interaction) and two advanced abilities (Personalized services and Agent synergy). Additionally, we categorize existing GUI Agents based on their methods of understanding the environment. Furthermore, we discuss two-stage efficient GUI task automation pipeline. Finally, this paper highlights the remaining challenges and future directions in GUI task automation, hoping to offer valuable insights to researchers and engineers in this field.

本综述调研了近期基于大语言模型的 GUI 任务自动化相关论文，旨在为读者提供 GUI 代理的全面概览。随后总结了 GUI 代理的能力，涵盖三项基础能力 (GUI 环境理解、设备控制和用户交互) 及两项高级能力 (个性化服务和代理协同)。此外，我们根据环境理解方法对现有 GUI 代理进行了分类。进一步讨论了两阶段高效 GUI 任务自动化流程。最后，本文强调了 GUI 任务自动化领域的剩余挑战和未来方向，期望为该领域的研究人员和工程师提供有价值的见解。

References

参考文献

[1] A. Iannessi, P.-Y. Marcy, O. Clatz, A.-S. Bertrand, M. Sugimoto, A review of existing and potential computer user interfaces for modern radiology, Insights into imaging 9 (4) (2018) 599-609.

A. Iannessi, P.-Y. Marcy, O. Clatz, A.-S. Bertrand, M. Sugimoto, 现代放射学中现有及潜在计算机用户界面的综述, Insights into imaging 9 (4) (2018) 599-609.

[2] L. Punchoojit, N. Hongwarittorrn, et al., Usability studies on mobile user interface design patterns: a systematic literature review, Advances in Human-Computer Interaction 2017 (2017).

L. Punchoojit, N. Hongwarittorrn, 等, 移动用户界面设计模式的可用性研究: 系统文献综述, Advances in Human-Computer Interaction 2017 (2017).

[3] J. Guo, W. Zhang, T. Xia, Impact of shopping website design on customer satisfaction and loyalty: The mediating role of usability and the moderating role of trust, Sustainability 15 (8) (2023) 6347.

J. Guo, W. Zhang, T. Xia, 购物网站设计对客户满意度和忠诚度的影响: 可用性的中介作用及信任的调节作用, Sustainability 15 (8) (2023) 6347.

[4] Apple, Siri, https://www.apple.com/siri/ (2024).

Apple, Siri, https://www.apple.com/siri/ (2024).

[5] Y. Jiang, Y. Lu, T. Knearem, C. E. Kliman-Silver, C. Lutteroth, T. J.- J. Li, J. Nichols, W. Stuerzlinger, Computational methodologies for understanding, automating, and evaluating user interfaces, in: Extended Abstracts of the CHI Conference on Human Factors in Computing Systems, 2024, pp. 1-7.

Y. Jiang, Y. Lu, T. Knearem, C. E. Kliman-Silver, C. Lutteroth, T. J.- J. Li, J. Nichols, W. Stuerzlinger, 理解、自动化及评估用户界面的计算方法, 载于:CHI 人因计算系统会议扩展摘要, 2024, 页 1-7.

[6] Y. Mehdi, Announcing microsoft copilot, your everyday ai companion (2023).

Y. Mehdi, 宣布微软 Copilot，您的日常 AI 助手 (2023)。

[7] Google, Google assistant for android, https://developer.android.com/guide/app-actions/overview (2024).

Google, Android 版 Google 助手, https://developer.android.com/guide/app-actions/overview (2024).

[8] A. Sugiura, Y. Koseki, Internet scrapbook: automating web browsing tasks by demonstration, in: Proceedings of the 11th annual ACM symposium on User interface software and technology, 1998, pp. 9-18.

杉浦彰 (A. Sugiura)、小关洋 (Y. Koseki)，《互联网剪贴簿: 通过演示自动化网页浏览任务》，收录于: 第 11 届 ACM 用户界面软件与技术年度研讨会会议记录，1998 年，第 9 - 18 页。

[9] C. Bernal-Cárdenas, N. Cooper, K. Moran, O. Chaparro, A. Marcus, D. Poshyvanyk, Translating video recordings of mobile app usages into replayable scenarios, in: Proceedings of the ACM/IEEE 42nd international conference on software engineering, 2020, pp. 309-321.

贝尔纳尔 - 卡德纳斯 (C. Bernal - Cárdenas)、库珀 (N. Cooper)、莫兰 (K. Moran)、查帕罗 (O. Chaparro)、马库斯 (A. Marcus)、波希瓦尼克 (D. Poshyvanyk),《将移动应用使用的视频记录转换为可重播场景》,收录于:ACM/IEEE 第 42 届软件工程国际会议会议记录，2020 年，第 309 - 321 页。

[10] J. Chen, A. Swearngin, J. Wu, T. Barik, J. Nichols, X. Zhang, Extracting replayable interactions from videos of mobile app usage, arXiv preprint arXiv:2207.04165 (2022).

陈 (J. Chen)、斯韦恩金 (A. Swearngin)、吴 (J. Wu)、巴里科 (T. Barik)、尼科尔斯 (J. Nichols)、张 (X. Zhang),《从移动应用使用视频中提取可重播交互》，arXiv 预印本 arXiv:2207.04165(2022 年)。

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

李 (Y. Li)、何 (J. He)、周 (X. Zhou)、张 (Y. Zhang)、鲍德里奇 (J. Baldridge),《将自然语言指令映射到移动用户界面动作序列》，arXiv 预印本 arXiv:2005.03776(2020 年)。

[12] S. Iyer, X. V. Lin, R. Pasunuru, T. Mihaylov, D. Simig, P. Yu, K. Shus-ter, T. Wang, Q. Liu, P. S. Koura, et al., Opt-iml: Scaling language model instruction meta learning through the lens of generalization, arXiv preprint arXiv:2212.12017 (2022).

伊耶尔 (S. Iyer)、林 (X. V. Lin)、帕苏努鲁 (R. Pasunuru)、米哈伊洛夫 (T. Mihaylov)、西米格 (D. Simig)、余 (P. Yu)、舒斯特 (K. Shus - ter)、王 (T. Wang)、刘 (Q. Liu)、库拉 (P. S. Koura) 等,《Opt - iml: 从泛化角度扩展语言模型指令元学习》，arXiv 预印本 arXiv:2212.12017(2022 年)。

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

施 (T. Shi)、卡尔帕蒂 (A. Karpathy)、范 (L. Fan)、埃尔南德斯 (J. Hernandez)、梁 (P. Liang),《比特世界: 一个基于网络的智能体开放领域平台》, 收录于: 国际机器学习会议，机器学习研究会议录，2017 年，第 3135 - 3144 页。

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

刘 (E. Z. Liu)、古 (K. Guu)、帕苏帕特 (P. Pasupat)、施 (T. Shi)、梁 (P. Liang),《使用工作流引导探索在网页界面上进行强化学习》，arXiv 预印本 arXiv:1802.08802(2018 年)。

[15] P. C. Humphreys, D. Raposo, T. Pohlen, G. Thornton, R. Chhaparia, A. Muldal, J. Abramson, P. Georgiev, A. Santoro, T. Lillicrap, A data-driven approach for learning to control computers, in: International Conference on Machine Learning, PMLR, 2022, pp. 9466-9482.

汉弗莱斯 (P. C. Humphreys)、拉波索 (D. Raposo)、波伦 (T. Pohlen)、桑顿 (G. Thornton)、查帕里亚 (R. Chhaparia)、穆尔达尔 (A. Muldal)、艾布拉姆森 (J. Abramson)、格奥尔基耶夫 (P. Georgiev)、桑托罗 (A. Santoro)、利利克拉普 (T. Lillicrap),《一种数据驱动的学习控制计算机的方法》, 收录于: 国际机器学习会议，机器学习研究会议录，2022 年，第 9466 - 9482 页。

[16] OpenAI, Introduce ChatGPT, https://openai.com/blog/chatgpt (2022).

开放人工智能公司 (OpenAI)，《介绍 ChatGPT》，https://openai.com/blog/chatgpt(2022 年)。

[17] S. Pichai, D. Hassabis, Introducing gemini: our largest and most capable ai model, Google. Retrieved December 8 (2023) 2023.

皮查伊 (S. Pichai)、哈萨比斯 (D. Hassabis)，《介绍双子座: 我们最大且功能最强的人工智能模型》，谷歌。2023 年 12 月 8 日检索。

[18] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, et al., Opt: Open pre-trained transformer language models, arXiv preprint arXiv:2205.01068 (2022).

张 (S. Zhang)、罗勒 (S. Roller)、戈亚尔 (N. Goyal)、阿泰特谢 (M. Artetxe)、陈 (M. Chen)、陈 (S. Chen)、德万 (C. Dewan)、迪亚布 (M. Diab)、李 (X. Li)、林 (X. V. Lin) 等，《Opt: 开放预训练的 Transformer 语言模型》，arXiv 预印本 arXiv:2205.01068(2022 年)。

[19] J. Goswami, K. K. Prajapati, A. Saha, A. K. Saha, Parameter-efficient fine-tuning large language model approach for hospital discharge paper summarization, Applied Soft Computing 157 (2024) 111531.

戈斯瓦米 (J. Goswami)、普拉贾帕蒂 (K. K. Prajapati)、萨哈 (A. Saha)、萨哈 (A. K. Saha)，《用于医院出院报告总结的参数高效微调大语言模型方法》，《应用软计算》157(2024 年)111531。

[20] 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).

席 (Z. Xi)、陈 (W. Chen)、郭 (X. Guo)、何 (W. He)、丁 (Y. Ding)、洪 (B. Hong)、张 (M. Zhang)、王 (J. Wang)、金 (S. Jin)、周 (E. Zhou) 等，《基于大语言模型的智能体的兴起与潜力: 综述》，arXiv 预印本 arXiv:2309.07864(2023 年)。

[21] X. Team, Xagent: An autonomous agent for complex task solving (2023).

X 团队，《X 智能体: 一个用于解决复杂任务的自主智能体》(2023 年)。

[22] Y. Shen, K. Song, X. Tan, D. Li, W. Lu, Y. Zhuang, Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face, Advances in Neural Information Processing Systems 36 (2024).

沈 (Y. Shen)、宋 (K. Song)、谭 (X. Tan)、李 (D. Li)、陆 (W. Lu)、庄 (Y. Zhuang)，《Hugginggpt: 使用 ChatGPT 及其在 Hugging Face 中的伙伴解决人工智能任务》，《神经信息处理系统进展》36(2024 年)。

[23] S. Wu, H. Fei, L. Qu, W. Ji, T.-S. Chua, Next-gpt: Any-to-any multimodal llm, arXiv preprint arXiv:2309.05519 (2023).

吴 (S. Wu)、费 (H. Fei)、曲 (L. Qu)、季 (W. Ji)、蔡 (T. - S. Chua)，《下一代 GPT: 任意到任意的多模态大语言模型》，arXiv 预印本 arXiv:2309.05519(2023 年)。

[24] 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).

A. Yan, Z. Yang, W. Zhu, K. Lin, L. Li, J. Wang, J. Yang, Y. Zhong, J. McAuley, J. Gao, 等，Gpt-4v 奇境: 用于零样本智能手机 GUI 导航的大型多模态模型，arXiv 预印本 arXiv:2311.07562 (2023)。

[25] H. Tao, S. TV, M. Shlapentokh-Rothman, D. Hoiem, H. Ji, Webwise: Web interface control and sequential exploration with large language models, arXiv preprint arXiv:2310.16042 (2023).

H. Tao, S. TV, M. Shlapentokh-Rothman, D. Hoiem, H. Ji，Webwise: 利用大型语言模型进行网页界面控制和顺序探索，arXiv 预印本 arXiv:2310.16042 (2023)。

[26] H. Wen, Y. Li, G. Liu, S. Zhao, T. Yu, T. J.-J. Li, S. Jiang, Y. Liu, Y. Zhang, Y. Liu, Autodroid: Llm-powered task automation in android, in: Proceedings of the 30th Annual International Conference on Mobile Computing and Networking, 2024, pp. 543-557.

H. Wen, Y. Li, G. Liu, S. Zhao, T. Yu, T. J.-J. Li, S. Jiang, Y. Liu, Y. Zhang, Y. Liu，Autodroid: 基于大型语言模型 (LLM) 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。

[27] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Advances in neural information processing systems 33 (2020) 1877-1901.

T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, 等，语言模型是少样本学习者，神经信息处理系统进展 33 (2020) 1877-1901。

[28] X. Huang, W. Liu, X. Chen, X. Wang, D. Lian, Y. Wang, R. Tang, E. Chen, Wese: Weak exploration to strong exploitation for llm agents, arXiv preprint arXiv:2404.07456 (2024).

X. Huang, W. Liu, X. Chen, X. Wang, D. Lian, Y. Wang, R. Tang, E. Chen，Wese: 大型语言模型代理的弱探索到强利用，arXiv 预印本 arXiv:2404.07456 (2024)。

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin，注意力机制即一切，神经信息处理系统进展 30 (2017)。

[30] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).

J. Devlin, M.-W. Chang, K. Lee, K. Toutanova，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)。

[31] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, et al., Palm: Scaling language modeling with pathways, Journal of Machine Learning Research 24 (240) (2023) 1-113.

A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, 等，PaLM: 通过 Pathways 扩展语言建模，机器学习研究杂志 24 (240) (2023) 1-113。

[32] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, Journal of machine learning research 21 (140) (2020) 1-67.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu，探索统一文本到文本变换器的迁移学习极限，机器学习研究杂志 21 (140) (2020) 1-67。

[33] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, D. Amodei, Scaling laws for neural language models, arXiv preprint arXiv:2001.08361 (2020).

J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, D. Amodei，神经语言模型的规模定律，arXiv 预印本 arXiv:2001.08361 (2020)。

[34] A. Zeng, X. Liu, Z. Du, Z. Wang, H. Lai, M. Ding, Z. Yang, Y. Xu, W. Zheng, X. Xia, et al., Glm-130b: An open bilingual pre-trained model, arXiv preprint arXiv:2210.02414 (2022).

A. Zeng, X. Liu, Z. Du, Z. Wang, H. Lai, M. Ding, Z. Yang, Y. Xu, W. Zheng, X. Xia, 等，GLM-130B: 一个开放的双语预训练模型，arXiv 预印本 arXiv:2210.02414 (2022)。

[35] T. Gupta, A. Kembhavi, Visual programming: Compositional visual reasoning without training, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 14953- 14962.

T. Gupta, A. Kembhavi，视觉编程: 无需训练的组合视觉推理，载于:IEEE/CVF 计算机视觉与模式识别会议论文集，2023，页 14953-14962。

[36] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, et al., Flamingo: a visual language model for few-shot learning, Advances in neural information processing systems 35 (2022) 23716-23736.

J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, 等，Flamingo: 用于少样本学习的视觉语言模型，神经信息处理系统进展 35 (2022) 23716-23736。

[37] C. Wu, S. Yin, W. Qi, X. Wang, Z. Tang, N. Duan, Visual chatgpt: Talking, drawing and editing with visual foundation models, arXiv preprint arXiv:2303.04671 (2023).

C. Wu, S. Yin, W. Qi, X. Wang, Z. Tang, N. Duan, Visual chatgpt: 基于视觉基础模型的对话、绘图与编辑，arXiv 预印本 arXiv:2303.04671 (2023)。

[38] Z. Yang, L. Li, J. Wang, K. Lin, E. Azarnasab, F. Ahmed, Z. Liu, C. Liu, M. Zeng, L. Wang, Mm-react: Prompting chatgpt for multimodal reasoning and action, arXiv preprint arXiv:2303.11381 (2023).

Z. Yang, L. Li, J. Wang, K. Lin, E. Azarnasab, F. Ahmed, Z. Liu, C. Liu, M. Zeng, L. Wang, Mm-react: 利用 ChatGPT 进行多模态推理与行动的提示，arXiv 预印本 arXiv:2303.11381 (2023)。

[39] M. Minderer, A. Gritsenko, A. Stone, M. Neumann, D. Weissenborn, A. Dosovitskiy, A. Mahendran, A. Arnab, M. Dehghani, Z. Shen, et al., Simple open-vocabulary object detection, in: European Conference on Computer Vision, Springer, 2022, pp. 728-755.

M. Minderer, A. Gritsenko, A. Stone, M. Neumann, D. Weissenborn, A. Dosovitskiy, A. Mahendran, A. Arnab, M. Dehghani, Z. Shen 等，简单的开放词汇对象检测，载于: 欧洲计算机视觉会议，Springer 出版社，2022，页 728-755。

[40] W. Kim, B. Son, I. Kim, Vilt: Vision-and-language transformer without convolution or region supervision, in: International conference on machine learning, PMLR, 2021, pp. 5583-5594.

W. Kim, B. Son, I. Kim, Vilt: 无卷积和区域监督的视觉与语言变换器, 载于: 国际机器学习会议, PMLR 出版社，2021，页 5583-5594。

[41] J. Li, D. Li, S. Savarese, 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.

J. Li, D. Li, S. Savarese, 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 出版社，2023，页 19730-19742。

[42] D. Zhu, J. Chen, X. Shen, X. Li, M. Elhoseiny, Minigpt-4: Enhancing vision-language understanding with advanced large language models, arXiv preprint arXiv:2304.10592 (2023).

D. Zhu, J. Chen, X. Shen, X. Li, M. Elhoseiny, Minigpt-4: 通过先进的大型语言模型增强视觉-语言理解，arXiv 预印本 arXiv:2304.10592 (2023)。

[43] H. Liu, C. Li, Q. Wu, Y. J. Lee, Visual instruction tuning, Advances in neural information processing systems 36 (2024).

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

[44] H. Zhang, X. Li, L. Bing, Video-llama: An instruction-tuned audio-visual language model for video understanding, arXiv preprint arXiv:2306.02858 (2023).

H. Zhang, X. Li, L. Bing, Video-llama: 面向视频理解的指令调优视听语言模型，arXiv 预印本 arXiv:2306.02858 (2023)。

[45] W. Dai, J. Li, D. Li, A. M. H. Tiong, J. Zhao, W. Wang, B. Li, P. N. Fung, S. Hoi, Instructblip: Towards general-purpose vision-language models with instruction tuning, Advances in Neural Information Processing Systems 36 (2024).

W. Dai, J. Li, D. Li, A. M. H. Tiong, J. Zhao, W. Wang, B. Li, P. N. Fung, S. Hoi, Instructblip: 面向通用视觉-语言模型的指令调优，神经信息处理系统进展 36 (2024)。

[46] F. Chen, M. Han, H. Zhao, Q. Zhang, J. Shi, S. Xu, B. Xu, X-llm: Bootstrapping advanced large language models by treating multi-modalities as foreign languages, arXiv preprint arXiv:2305.04160 (2023).

F. Chen, M. Han, H. Zhao, Q. Zhang, J. Shi, S. Xu, B. Xu, X-llm: 将多模态视为外语以引导先进大型语言模型，arXiv 预印本 arXiv:2305.04160 (2023)。

[47] W. Van Melle, Mycin: a knowledge-based consultation program for infectious disease diagnosis, International journal of man-machine studies 10 (3) (1978) 313-322.

W. Van Melle, Mycin: 一种基于知识的传染病诊断咨询程序, 国际人机研究杂志 10 (3) (1978) 313-322。

[48] S. K. Sarma, K. R. Singh, A. Singh, An expert system for diagnosis of diseases in rice plant, International Journal of Artificial Intelligence 1 (1) (2010) 26-31.

S. K. Sarma, K. R. Singh, A. Singh, 水稻病害诊断专家系统，国际人工智能杂志 1 (1) (2010) 26-31。

[49] M. F. Zarandi, S. Soltanzadeh, A. Mohammadi, O. Castillo, Designing a general type-2 fuzzy expert system for diagnosis of depression, Applied Soft Computing 80 (2019) 329-341.

M. F. Zarandi, S. Soltanzadeh, A. Mohammadi, O. Castillo, 设计一种通用型 2 型模糊专家系统用于抑郁症诊断，应用软计算 80 (2019) 329-341。

[50] D. Içen, S. Günay, Design and implementation of the fuzzy expert system in monte carlo methods for fuzzy linear regression, Applied Soft Computing 77 (2019) 399-411.

D. Içen, S. Günay, 基于蒙特卡洛方法的模糊线性回归模糊专家系统的设计与实现，应用软计算 77 (2019) 399-411。

[51] R. A. Brooks, Intelligence without representation, Artificial intelligence 47 (1-3) (1991) 139-159.

R. A. Brooks, 无表征的智能，人工智能 47 (1-3) (1991) 139-159。

[52] P. Maes, Designing autonomous agents: Theory and practice from biology to engineering and back, MIT press, 1990.

P. Maes, 自主智能体设计: 从生物学到工程及其回归的理论与实践，MIT 出版社，1990。

[53] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning, arXiv preprint arXiv:1312.5602 (2013).

V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, 使用深度强化学习玩 Atari 游戏，arXiv 预印本 arXiv:1312.5602 (2013)。

[54] A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, R. Vicente, Multiagent cooperation and competition with deep reinforcement learning, PIoS one 12 (4) (2017) e0172395.

A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, R. Vicente, 基于深度强化学习的多智能体合作与竞争，PIoS one 12 (4) (2017) e0172395。

[55] J. Carapuço, R. Neves, N. Horta, Reinforcement learning applied to forex trading, Applied Soft Computing 73 (2018) 783-794.

J. Carapuço, R. Neves, N. Horta, 强化学习在外汇交易中的应用，Applied Soft Computing 73 (2018) 783-794。

[56] Z. Hu, X. Yu, Reinforcement learning-based comprehensive learning grey wolf optimizer for feature selection, Applied Soft Computing 149 (2023) 110959.

Z. Hu, X. Yu, 基于强化学习的综合学习灰狼优化算法用于特征选择，Applied Soft Computing 149 (2023) 110959。

[57] H. Yang, S. Yue, Y. He, Auto-gpt for online decision making: Benchmarks and additional opinions, arXiv preprint arXiv:2306.02224 (2023).

H. Yang, S. Yue, Y. He, 用于在线决策的 Auto-GPT: 基准测试及附加观点，arXiv 预印本 arXiv:2306.02224 (2023)。

[58] Langchain, Langchain, https://github.com/langchain-ai/ langchain (2023).

Langchain, Langchain, https://github.com/langchain-ai/langchain (2023)。

[59] 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 35 (2022) 24824-24837.

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型语言模型的推理能力，神经信息处理系统进展 35 (2022) 24824-24837。

[60] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, K. Narasimhan, Tree of thoughts: Deliberate problem solving with large language models, Advances in Neural Information Processing Systems 36 (2024).

S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, K. Narasimhan, 思维树: 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 36 (2024)。

[61] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, Y. Cao, React: Synergizing reasoning and acting in language models, arXiv preprint arXiv:2210.03629 (2022).

S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, Y. Cao, React: 在语言模型中协同推理与行动，arXiv 预印本 arXiv:2210.03629 (2022)。

[62] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, S. Yao, Reflexion: Language agents with verbal reinforcement learning, Advances in Neural Information Processing Systems 36 (2024).

N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, S. Yao, Reflexion: 具备语言强化学习能力的语言代理，神经信息处理系统进展 36 (2024)。

[63] Q. Wu, G. Bansal, J. Zhang, Y. Wu, S. Zhang, E. Zhu, B. Li, L. Jiang, X. Zhang, C. Wang, Autogen: Enabling next-gen llm applications via multi-agent conversation framework, arXiv preprint arXiv:2308.08155 (2023).

Q. Wu, G. Bansal, J. Zhang, Y. Wu, S. Zhang, E. Zhu, B. Li, L. Jiang, X. Zhang, C. Wang, Autogen: 通过多智能体对话框架支持下一代大型语言模型应用，arXiv 预印本 arXiv:2308.08155 (2023)。

[64] T. Guo, X. Chen, Y. Wang, R. Chang, S. Pei, N. V. Chawla, O. Wiest, X. Zhang, Large language model based multi-agents: A survey of progress and challenges, arXiv preprint arXiv:2402.01680 (2024).

T. Guo, X. Chen, Y. Wang, R. Chang, S. Pei, N. V. Chawla, O. Wiest, X. Zhang, 基于大型语言模型的多智能体: 进展与挑战综述，arXiv 预印本 arXiv:2402.01680 (2024)。

[65] C. Qian, X. Cong, C. Yang, W. Chen, Y. Su, J. Xu, Z. Liu, M. Sun, Communicative agents for software development, arXiv preprint arXiv:2307.07924 (2023).

C. Qian, X. Cong, C. Yang, W. Chen, Y. Su, J. Xu, Z. Liu, M. Sun, 用于软件开发的交流代理，arXiv 预印本 arXiv:2307.07924 (2023)。

[66] Y. Du, S. Li, A. Torralba, J. B. Tenenbaum, I. Mordatch, Improving factuality and reasoning in language models through multiagent debate, arXiv preprint arXiv:2305.14325 (2023).

Y. Du, S. Li, A. Torralba, J. B. Tenenbaum, I. Mordatch, 通过多智能体辩论提升语言模型的事实性与推理能力，arXiv 预印本 arXiv:2305.14325 (2023)。

[67] Q. Zhao, J. Wang, Y. Zhang, Y. Jin, K. Zhu, H. Chen, X. Xie, Com-peteai: Understanding the competition behaviors in large language model-based agents, arXiv preprint arXiv:2310.17512 (2023).

赵 Q.、王 J.、张 Y.、金 Y.、朱 K.、陈 H.、谢 X.，《Com - peteai: 理解基于大语言模型的智能体的竞争行为》，预印本 arXiv:2310.17512 (2023)。

[68] G. Li, Y. Li, Spotlight: Mobile ui understanding using vision-language models with a focus, arXiv preprint arXiv:2209.14927 (2022).

李 G.、李 Y.，《Spotlight: 使用有重点的视觉 - 语言模型进行移动用户界面理解》，预印本 arXiv:2209.14927 (2022)。

[69] L. Sun, X. Chen, L. Chen, T. Dai, Z. Zhu, K. Yu, Meta-gui: towards multi-modal conversational agents on mobile gui, arXiv preprint arXiv:2205.11029 (2022).

孙 L.、陈 X.、陈 L.、戴 T.、朱 Z.、余 K.,《Meta - gui: 迈向移动图形用户界面上的多模态对话智能体》，预印本 arXiv:2205.11029 (2022)。

[70] Z. Zhan, A. Zhang, You only look at screens: Multimodal chain-of-action agents, arXiv preprint arXiv:2309.11436 (2023).

詹 Z.、张 A.,《你只需看屏幕: 多模态行动链智能体》，预印本 arXiv:2309.11436 (2023)。

[71] S. Lee, J. Choi, J. Lee, H. Choi, S. Y. Ko, S. Oh, I. Shin, Explore, select, derive, and recall: Augmenting llm with human-like memory for mobile task automation, arXiv preprint arXiv:2312.03003 3 (7) (2023) 8.

李 S.、崔 J.、李 J.、崔 H.、高 S. Y.、吴 S.、申 I.,《探索、选择、推导和回忆: 为移动任务自动化赋予大语言模型类人记忆》，预印本 arXiv:2312.03003 3 (7) (2023) 8。

[72] Z. Yang, J. Liu, Y. Han, X. Chen, Z. Huang, B. Fu, G. Yu, Appagent: Multimodal agents as smartphone users, arXiv preprint arXiv:2312.13771 (2023).

杨 Z.、刘 J.、韩 Y.、陈 X.、黄 Z.、傅 B.、余 G.,《Appagent: 作为智能手机用户的多模态智能体》，预印本 arXiv:2312.13771 (2023)。

[73] H. Wen, H. Wang, J. Liu, Y. Li, Droidbot-gpt: Gpt-powered ui automation for android, arXiv preprint arXiv:2304.07061 (2023).

文 H.、王 H.、刘 J.、李 Y.,《Droidbot - gpt: 由 GPT 驱动的安卓用户界面自动化》,预印本 arXiv:2304.07061 (2023)。

[74] J. Wang, H. Xu, J. Ye, M. Yan, W. Shen, J. Zhang, F. Huang, J. Sang, Mobile-agent: Autonomous multi-modal mobile device agent with visual perception, arXiv preprint arXiv:2401.16158 (2024).

王 J.、徐 H.、叶 J.、严 M.、沈 W.、张 J.、黄 F.、桑 J.,《Mobile - agent: 具有视觉感知能力的自主多模态移动设备智能体》，预印本 arXiv:2401.16158 (2024)。

[75] J. Wang, H. Xu, H. Jia, X. Zhang, M. Yan, W. Shen, J. Zhang, F. Huang, J. Sang, Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration, arXiv preprint arXiv:2406.01014 (2024).

王 J.、徐 H.、贾 H.、张 X.、严 M.、沈 W.、张 J.、黄 F.、桑 J.,《Mobile - agent - v2: 通过多智能体协作实现有效导航的移动设备操作助手》，预印本 arXiv:2406.01014 (2024)。

[76] K. Cheng, Q. Sun, Y. Chu, F. Xu, Y. Li, J. Zhang, Z. Wu, Seeclick: Harnessing gui grounding for advanced visual gui agents, arXiv preprint arXiv:2401.10935 (2024).

程 K.、孙 Q.、楚 Y.、徐 F.、李 Y.、张 J.、吴 Z.,《Seeclick: 利用图形用户界面基础实现高级视觉图形用户界面智能体》，预印本 arXiv:2401.10935 (2024)。

[77] X. Ma, Z. Zhang, H. Zhao, Coco-agent: A comprehensive cognitive mllm agent for smartphone gui automation, arXiv preprint arXiv:2402.11941 (2024).

马 X.、张 Z.、赵 H.,《Coco - agent: 用于智能手机图形用户界面自动化的综合认知多模态大语言模型智能体》，预印本 arXiv:2402.11941 (2024)。

[78] R. Nakano, J. Hilton, S. Balaji, J. Wu, L. Ouyang, C. Kim, C. Hesse, S. Jain, V. Kosaraju, W. Saunders, et al., Webgpt: Browser-assisted question-answering with human feedback, arXiv preprint arXiv:2112.09332 (2021).

中野 R.、希尔顿 J.、巴拉吉 S.、吴 J.、欧阳 L.、金 C.、黑塞 C.、贾因 S.、科萨拉朱 V.、桑德斯 W. 等,《Webgpt: 通过人类反馈实现浏览器辅助问答》，预印本 arXiv:2112.09332 (2021)。

[79] P. Shaw, M. Joshi, J. Cohan, J. Berant, P. Pasupat, H. Hu, U. Khan-delwal, K. Lee, K. N. Toutanova, From pixels to ui actions: Learning to follow instructions via graphical user interfaces, Advances in Neural Information Processing Systems 36 (2023) 34354-34370.

肖 P.、乔希 M.、科汉 J.、贝兰特 J.、帕苏帕特 P.、胡 H.、坎德尔瓦尔 U.、李 K.、图塔诺娃 K. N.,《从像素到用户界面动作: 通过图形用户界面学习遵循指令》,《神经信息处理系统进展》36 (2023) 34354 - 34370。

[80] H. Furuta, K.-H. Lee, O. Nachum, Y. Matsuo, A. Faust, S. S. Gu, I. Gur, Multimodal web navigation with instruction-finetuned foundation models, arXiv preprint arXiv:2305.11854 (2023).

古田 H.、李 K. - H.、纳楚姆 O.、松尾 Y.、浮士德 A.、顾 S. S.、古尔 I.,《使用指令微调基础模型进行多模态网页导航》，预印本 arXiv:2305.11854 (2023)。

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

邓 X.、顾 Y.、郑 B.、陈 S.、史蒂文斯 S.、王 B.、孙 H.、苏 Y.,《Mind2web: 迈向通用的网页智能体》,《神经信息处理系统进展》36 (2024)。

[82] I. Gur, H. Furuta, A. Huang, M. Safdari, Y. Matsuo, D. Eck, A. Faust, A real-world webagent with planning, long context understanding, and program synthesis, arXiv preprint arXiv:2307.12856 (2023).

I. Gur, H. Furuta, A. Huang, M. Safdari, Y. Matsuo, D. Eck, A. Faust, 一个具备规划、长上下文理解和程序合成能力的真实世界网络代理，arXiv 预印本 arXiv:2307.12856 (2023)。

[83] G. Kim, P. Baldi, S. McAleer, Language models can solve computer tasks, Advances in Neural Information Processing Systems 36 (2024).

G. Kim, P. Baldi, S. McAleer, 语言模型能够解决计算机任务，神经信息处理系统进展 36 (2024)。

[84] H. Sun, Y. Zhuang, L. Kong, B. Dai, C. Zhang, Adaplanner: Adaptive planning from feedback with language models, Advances in Neural Information Processing Systems 36 (2024).

H. Sun, Y. Zhuang, L. Kong, B. Dai, C. Zhang, Adaplanner: 基于反馈的自适应规划与语言模型，神经信息处理系统进展 36 (2024)。

[85] B. Zheng, B. Gou, J. Kil, H. Sun, Y. Su, Gpt-4v (ision) is a generalist web agent, if grounded, arXiv preprint arXiv:2401.01614 (2024).

B. Zheng, B. Gou, J. Kil, H. Sun, Y. Su, GPT-4V(视觉) 是一种通用网络代理，若具备落地能力，arXiv 预印本 arXiv:2401.01614 (2024)。

[86] H. He, W. Yao, K. Ma, W. Yu, Y. Dai, H. Zhang, Z. Lan, D. Yu, Webvoyager: Building an end-to-end web agent with large multimodal models, arXiv preprint arXiv:2401.13919 (2024).

H. He, W. Yao, K. Ma, W. Yu, Y. Dai, H. Zhang, Z. Lan, D. Yu, Webvoyager: 基于大型多模态模型构建端到端网络代理，arXiv 预印本 arXiv:2401.13919 (2024)。

[87] J. Kil, C. H. Song, B. Zheng, X. Deng, Y. Su, W.-L. Chao, Dual-view visual contextualization for web navigation, arXiv preprint arXiv:2402.04476 (2024).

J. Kil, C. H. Song, B. Zheng, X. Deng, Y. Su, W.-L. Chao, 双视角视觉上下文化用于网页导航，arXiv 预印本 arXiv:2402.04476 (2024)。

[88] C. Zhang, L. Li, S. He, X. Zhang, B. Qiao, S. Qin, M. Ma, Y. Kang, Q. Lin, S. Rajmohan, et al., Ufo: A ui-focused agent for windows os interaction, arXiv preprint arXiv:2402.07939 (2024).

C. Zhang, L. Li, S. He, X. Zhang, B. Qiao, S. Qin, M. Ma, Y. Kang, Q. Lin, S. Rajmohan, 等，UFO: 面向 Windows 操作系统交互的 UI 聚焦代理，arXiv 预印本 arXiv:2402.07939 (2024)。

[89] H. Lai, X. Liu, I. L. Iong, S. Yao, Y. Chen, P. Shen, H. Yu, H. Zhang, X. Zhang, Y. Dong, et al., Autowebglm: Bootstrap and reinforce a large language model-based web navigating agent, arXiv preprint arXiv:2404.03648 (2024).

H. Lai, X. Liu, I. L. Iong, S. Yao, Y. Chen, P. Shen, H. Yu, H. Zhang, X. Zhang, Y. Dong, 等, AutoWebGLM: 基于大型语言模型的网页导航代理的自举与强化，arXiv 预印本 arXiv:2404.03648 (2024)。

[90] Z. Song, Y. Li, M. Fang, Z. Chen, Z. Shi, Y. Huang, Mmac-copilot: Multi-modal agent collaboration operating system copilot, arXiv preprint arXiv:2404.18074 (2024).

Z. Song, Y. Li, M. Fang, Z. Chen, Z. Shi, Y. Huang, MMAC-Copilot: 多模态代理协作操作系统副驾驶，arXiv 预印本 arXiv:2404.18074 (2024)。

[91] B. Wang, G. Li, 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.

B. Wang, G. Li, Y. Li, 利用大型语言模型实现移动 UI 的对话式交互，载于:2023 年 CHI 人机交互大会论文集，2023，页 1-17。

[92] Y. Li, Z. Yang, Y. Guo, X. Chen, Droidbot: a lightweight ui-guided test input generator for android, in: 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C), IEEE, 2017, pp. 23-26.

Y. Li, Z. Yang, Y. Guo, X. Chen, DroidBot: 一种轻量级的基于 UI 引导的 Android 测试输入生成器，载于:2017 年 IEEE/ACM 第 39 届国际软件工程会议伴随会议 (ICSE-C)，IEEE，2017，页 23-26。

[93] P. He, X. Liu, J. Gao, W. Chen, Deberta: Decoding-enhanced bert with disentangled attention, arXiv preprint arXiv:2006.03654 (2020).

P. He, X. Liu, J. Gao, W. Chen, DeBERTa: 具有解耦注意力的解码增强 BERT，arXiv 预印本 arXiv:2006.03654 (2020)。

[94] 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 25 (70) (2024) 1-53.

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 25 (70) (2024) 1-53。

[95] K. Lee, M. Joshi, I. R. Turc, H. Hu, F. Liu, J. M. Eisenschlos, U. Khan-delwal, P. Shaw, M.-W. Chang, K. Toutanova, Pix2struct: Screenshot parsing as pretraining for visual language understanding, in: International Conference on Machine Learning, PMLR, 2023, pp. 18893- 18912.

K. Lee, M. Joshi, I. R. Turc, H. Hu, F. Liu, J. M. Eisenschlos, U. Khan-delwal, P. Shaw, M.-W. Chang, K. Toutanova, Pix2Struct: 将截图解析作为视觉语言理解的预训练，载于: 国际机器学习大会，PMLR，2023，页 18893-18912。

[96] 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).

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)。

[97] A. Shtedritski, C. Rupprecht, A. Vedaldi, What does clip know about a red circle? visual prompt engineering for vlms, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 11987-11997.

A. 什泰德里茨基 (A. Shtedritski)、C. 鲁普雷希特 (C. Rupprecht)、A. 韦尔迪 aldi(A. Vedaldi)，CLIP 对红色圆圈了解多少？视觉语言模型的视觉提示工程，见:《IEEE/CVF 国际计算机视觉会议论文集》，2023 年，第 11987 - 11997 页。

[98] L. Yang, Y. Wang, X. Li, X. Wang, J. Yang, Fine-grained visual prompting, Advances in Neural Information Processing Systems 36 (2024).

杨磊 (L. Yang)、王宇 (Y. Wang)、李翔 (X. Li)、王鑫 (X. Wang)、杨军 (J. Yang)，细粒度视觉提示,《神经信息处理系统进展》36 (2024)。

[99] J. Yang, H. Zhang, F. Li, X. Zou, C. Li, J. Gao, Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v, arXiv preprint arXiv:2310.11441 (2023).

杨军 (J. Yang)、张辉 (H. Zhang)、李飞 (F. Li)、邹翔 (X. Zou)、李超 (C. Li)、高杰 (J. Gao)，标记集提示释放 GPT - 4V 非凡的视觉定位能力，预印本 arXiv:2310.11441 (2023)。

[100] Z. Peng, W. Wang, L. Dong, Y. Hao, S. Huang, S. Ma, F. Wei, Kosmos- 2: Grounding multimodal large language models to the world, arXiv preprint arXiv:2306.14824 (2023).

彭泽 (Z. Peng)、王巍 (W. Wang)、董亮 (L. Dong)、郝宇 (Y. Hao)、黄硕 (S. Huang)、马帅 (S. Ma)、魏峰 (F. Wei)，宇宙 2 号 (Kosmos - 2): 将多模态大语言模型与现实世界关联起来，预印本 arXiv:2306.14824 (2023)。

[101] Y. Zhao, Z. Lin, D. Zhou, Z. Huang, J. Feng, B. Kang, Bubogpt: Enabling visual grounding in multi-modal llms, arXiv preprint arXiv:2307.08581 (2023).

赵宇 (Y. Zhao)、林泽 (Z. Lin)、周丹 (D. Zhou)、黄震 (Z. Huang)、冯杰 (J. Feng)、康博 (B. Kang)，布博 GPT(Bubogpt): 实现多模态大语言模型的视觉定位，预印本 arXiv:2307.08581 (2023)。

[102] K. Chen, Z. Zhang, W. Zeng, R. Zhang, F. Zhu, R. Zhao, Shikra: Unleashing multimodal llm's referential dialogue magic, arXiv preprint arXiv:2306.15195 (2023).

陈凯 (K. Chen)、张泽 (Z. Zhang)、曾伟 (W. Zeng)、张锐 (R. Zhang)、朱峰 (F. Zhu)、赵瑞 (R. Zhao)，石克拉 (Shikra): 释放多模态大语言模型的指称对话魔力，预印本 arXiv:2306.15195 (2023)。

[103] OpenAI, Gpt-4v(ision) system card (2023). URL https://cdn.openai.com/papers/GPTV_System_Card.pdf

开放 AI(OpenAI),GPT - 4V(视觉) 系统说明 (2023)。网址 https://cdn.openai.com/papers/GPTV_System_Card.pdf

[104] S. Sunkara, M. Wang, L. Liu, G. Baechler, Y.-C. Hsiao, A. Sharma, J. Stout, et al., Towards better semantic understanding of mobile interfaces, arXiv preprint arXiv:2210.02663 (2022).

桑卡拉 (S. Sunkara)、王猛 (M. Wang)、刘磊 (L. Liu)、贝希勒 (G. Baechler)、萧逸晨 (Y.-C. Hsiao)、夏尔马 (A. Sharma)、斯托特 (J. Stout) 等，迈向对移动界面更好的语义理解，预印本 arXiv:2210.02663 (2022)。

[105] K. You, H. Zhang, E. Schoop, F. Weers, A. Swearngin, J. Nichols, Y. Yang, Z. Gan, Ferret-ui: Grounded mobile ui understanding with multimodal llms, arXiv preprint arXiv:2404.05719 (2024).

尤凯 (K. You)、张辉 (H. Zhang)、肖普 (E. Schoop)、韦尔斯 (F. Weers)、斯威宁 (A. Swearngin)、尼科尔斯 (J. Nichols)、杨阳 (Y. Yang)、甘泽 (Z. Gan)，雪貂 UI(Ferret - ui): 利用多模态大语言模型实现基于现实的移动 UI 理解，预印本 arXiv:2404.05719 (2024)。

[106] W. Hong, W. Wang, Q. Lv, J. Xu, W. Yu, J. Ji, Y. Wang, Z. Wang, Y. Dong, M. Ding, et al., Cogagent: A visual language model for gui agents, arXiv preprint arXiv:2312.08914 (2023).

洪伟 (W. Hong)、王巍 (W. Wang)、吕强 (Q. Lv)、徐军 (J. Xu)、余伟 (W. Yu)、季杰 (J. Ji)、王宇 (Y. Wang)、王泽 (Z. Wang)、董宇 (Y. Dong)、丁明 (M. Ding) 等，认知智能体 (Cogagent): 用于 GUI 智能体的视觉语言模型，预印本 arXiv:2312.08914 (2023)。

[107] T. Iki, A. Aizawa, Do berts learn to use browser user interface? exploring multi-step tasks with unified vision-and-language berts, arXiv preprint arXiv:2203.07828 (2022).

池田 (T. Iki)、相泽 (A. Aizawa)，BERT 模型学会使用浏览器用户界面了吗？用统一视觉语言 BERT 探索多步骤任务，预印本 arXiv:2203.07828 (2022)。

[108] 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).

陈敏 (M. Chen)、特沃雷克 (J. Tworek)、君浩 (H. Jun)、袁强 (Q. Yuan)、平托 (H. P. d. O. Pinto)、卡普兰 (J. Kaplan)、爱德华兹 (H. Edwards)、布尔达 (Y. Burda)、约瑟夫 (N. Joseph)、布罗克曼 (G. Brockman) 等，评估基于代码训练的大语言模型，预印本 arXiv:2107.03374 (2021)。

[109] Z. Feng, D. Guo, D. Tang, N. Duan, X. Feng, M. Gong, L. Shou, B. Qin, T. Liu, D. Jiang, et al., Codebert: A pre-trained model for programming and natural languages, arXiv preprint arXiv:2002.08155 (2020).

冯泽 (Z. Feng)、郭丹 (D. Guo)、唐丹 (D. Tang)、段楠 (N. Duan)、冯翔 (X. Feng)、龚明 (M. Gong)、寿磊 (L. Shou)、秦斌 (B. Qin)、刘婷 (T. Liu)、蒋迪 (D. Jiang) 等，代码 BERT(Codebert): 用于编程和自然语言的预训练模型，预印本 arXiv:2002.08155 (2020)。

[110] Y. Li, D. Choi, J. Chung, N. Kushman, J. Schrittwieser, R. Leblond, T. Eccles, J. Keeling, F. Gimeno, A. Dal Lago, et al., Competition-level code generation with alphacode, Science 378 (6624) (2022) 1092-1097.

李阳 (Y. Li)、崔东 (D. Choi)、钟杰 (J. Chung)、库什曼 (N. Kushman)、施里特维泽 (J. Schrittwieser)、勒布朗 (R. Leblond)、埃克尔斯 (T. Eccles)、基林 (J. Keeling)、希梅诺 (F. Gimeno)、达拉戈 (A. Dal Lago) 等，用 AlphaCode 实现竞赛级代码生成，《科学》378 (6624) (2022) 1092 - 1097。

[111] Y. Wang, W. Wang, S. Joty, S. C. Hoi, Codet5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation, arXiv preprint arXiv:2109.00859 (2021).

王宇 (Y. Wang)、王巍 (W. Wang)、乔蒂 (S. Joty)、何世昌 (S. C. Hoi)，代码 T5(Codet5): 用于代码理解和生成的标识符感知统一预训练编解码器模型，预印本 arXiv:2109.00859 (2021)。

[112] 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).

李阳 (Y. Li)、温浩 (H. Wen)、王巍 (W. Wang)、李霞 (X. Li)、袁宇 (Y. Yuan)、刘刚 (G. Liu)、刘杰 (J. Liu)、徐伟 (W. Xu)、王晓 (X. Wang)、孙燕 (Y. Sun) 等，《个人大语言模型智能体: 关于能力、效率和安全性的见解与综述》，预印本 arXiv:2401.05459 (2024)。

[113] Microsoft, Bing web search api, https://www.microsoft.com/en-us/ bing/apis/bing-web-search-api (2023).

微软，必应网页搜索应用程序编程接口 (Bing web search api)，https://www.microsoft.com/en-us/bing/apis/bing-web-search-api (2023)。

[114] W. Li, F.-L. Hsu, W. Bishop, F. Campbell-Ajala, O. Riva, M. Lin, Uinav: A maker of ui automation agents, arXiv preprint arXiv:2312.10170 (2023).

李伟 (W. Li)、许富立 (F.-L. Hsu)、毕晓普 (W. Bishop)、坎贝尔 - 阿贾拉 (F. Campbell-Ajala)、里瓦 (O. Riva)、林明 (M. Lin)，《Uinav: 用户界面自动化智能体的创造者》，预印本 arXiv:2312.10170 (2023)。

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

卢兴华 (X. H. Lù)、卡斯纳 (Z. Kasner)、雷迪 (S. Reddy)，《Weblinx: 通过多轮对话实现现实世界网站导航》，预印本 arXiv:2402.05930 (2024)。

[116] N. Xu, S. Masling, M. Du, G. Campagna, L. Heck, J. Landay, M. S. Lam, Grounding open-domain instructions to automate web support tasks, arXiv preprint arXiv:2103.16057 (2021).

徐楠 (N. Xu)、马斯林 (S. Masling)、杜明 (M. Du)、坎帕尼亚 (G. Campagna)、赫克 (L. Heck)、兰代 (J. Landay)、林明思 (M. S. Lam)，《将开放领域指令与自动化网络支持任务相结合》，预印本 arXiv:2103.16057 (2021)。

[117] Z. Wu, C. Han, Z. Ding, Z. Weng, Z. Liu, S. Yao, T. Yu, L. Kong, Os-copilot: Towards generalist computer agents with self-improvement, arXiv preprint arXiv:2402.07456 (2024).

吴泽 (Z. Wu)、韩超 (C. Han)、丁泽 (Z. Ding)、翁泽 (Z. Weng)、刘泽 (Z. Liu)、姚思 (S. Yao)、余涛 (T. Yu)、孔磊 (L. Kong)，《操作系统副驾驶 (Os-copilot): 迈向具有自我改进能力的通用计算机智能体》，预印本 arXiv:2402.07456 (2024)。

[118] D. Toyama, P. Hamel, A. Gergely, G. Comanici, A. Glaese, Z. Ahmed, T. Jackson, S. Mourad, D. Precup, Androidenv: A reinforcement learning platform for android, arXiv preprint arXiv:2105.13231 (2021).

远山 (D. Toyama)、哈梅尔 (P. Hamel)、盖尔盖伊 (A. Gergely)、科马尼奇 (G. Comanici)、格莱斯 (A. Glaese)、艾哈迈德 (Z. Ahmed)、杰克逊 (T. Jackson)、穆拉德 (S. Mourad)、普雷库普 (D. Precup)，《Androidenv: 一个用于安卓系统的强化学习平台》，预印本 arXiv:2105.13231 (2021)。

[119] A. Developers, Create your own accessibility service (2020).

开发者 A，《创建你自己的无障碍服务》(2020)。

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

伯恩斯 (A. Burns)、阿尔桑 (D. Arsan)、阿加瓦尔 (S. Agrawal)、库马尔 (R. Kumar)、萨内科 (K. Saenko)、普拉默 (B. A. Plummer)，《一个用于具有未知命令可行性的交互式视觉 - 语言导航的数据集》，收录于:《欧洲计算机视觉会议》，施普林格出版社，2022 年，第 312 - 328 页。

[121] S. G. Venkatesh, P. Talukdar, S. Narayanan, Ugif: Ui grounded instruction following, arXiv preprint arXiv:2211.07615 (2022).

文卡特什 (S. G. Venkatesh)、塔鲁克达尔 (P. Talukdar)、纳拉亚南 (S. Narayanan)，《Ugif: 基于用户界面的指令遵循》，预印本 arXiv:2211.07615 (2022)。

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

王磊 (L. Wang)、邓宇 (Y. Deng)、查宇 (Y. Zha)、毛刚 (G. Mao)、王强 (Q. Wang)、闵涛 (T. Min)、陈巍 (W. Chen)、陈硕 (S. Chen)，《移动智能体基准测试 (Mobileagentbench): 一个高效且用户友好的移动大语言模型智能体基准测试》，预印本 arXiv:2406.08184 (2024)。

[123] H. Wen, Y. Li, G. Liu, S. Zhao, T. Yu, T. J.-J. Li, S. Jiang, Y. Liu, Y. Zhang, Y. Liu, Empowering llm to use smartphone for intelligent task automation, arXiv preprint arXiv:2308.15272 (2023).

温浩 (H. Wen)、李阳 (Y. Li)、刘刚 (G. Liu)、赵硕 (S. Zhao)、余涛 (T. Yu)、李 T. J.-J.、蒋硕 (S. Jiang)、刘燕 (Y. Liu)、张宇 (Y. Zhang)、刘燕 (Y. Liu)，《赋能大语言模型使用智能手机实现智能任务自动化》，预印本 arXiv:2308.15272 (2023)。

[124] D. Zhang, L. Chen, Z. Zhao, R. Cao, K. Yu, Mobile-env: An evaluation platform and benchmark for interactive agents in llm era, arXiv preprint arXiv:2305.08144 (2023).

张迪 (D. Zhang)、陈亮 (L. Chen)、赵泽 (Z. Zhao)、曹锐 (R. Cao)、余凯 (K. Yu)，《移动环境 (Mobile-env): 大语言模型时代交互式智能体的评估平台和基准测试》，预印本 arXiv:2305.08144 (2023)。

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

张磊 (L. Zhang)、王硕 (S. Wang)、贾鑫 (X. Jia)、郑泽 (Z. Zheng)、闫妍 (Y. Yan)、高磊 (L. Gao)、李阳 (Y. Li)、徐明 (M. Xu)，《Llamatouch: 一个用于移动用户界面自动化任务评估的可靠且可扩展的测试平台》，预印本 arXiv:2404.16054 (2024)。

[126] S. Zhou, F. F. Xu, H. Zhu, X. Zhou, R. Lo, A. Sridhar, X. Cheng, Y. Bisk, D. Fried, U. Alon, et al., Webarena: A realistic web environment for building autonomous agents, arXiv preprint arXiv:2307.13854 (2023).

周硕 (S. Zhou)、徐芳芳 (F. F. Xu)、朱浩 (H. Zhu)、周翔 (X. Zhou)、罗瑞 (R. Lo)、斯里达尔 (A. Sridhar)、程翔 (X. Cheng)、比斯克 (Y. Bisk)、弗里德 (D. Fried)、阿隆 (U. Alon) 等，《Webarena: 一个用于构建自主智能体的真实网络环境》，预印本 arXiv:2307.13854 (2023)。

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

arXiv:2401.13649 (2024).

Koh J. Y.、罗瑞 (R. Lo)、张磊 (L. Jang)、杜武尔 (V. Duvvur)、林 M. C.、黄 P.-Y.、诺伊 - 比格 (G. Neu-big)、周硕 (S. Zhou)、萨拉胡丁诺夫 (R. Salakhutdinov)、弗里德 (D. Fried),《Visualwebarena: 在真实视觉网络任务上评估多模态智能体》，预印本 arXiv:2401.13649 (2024)。

[128] S. Yao, H. Chen, J. Yang, K. Narasimhan, Webshop: Towards scalable real-world web interaction with grounded language agents, Advances in Neural Information Processing Systems 35 (2022) 20744-20757.

S. Yao, H. Chen, J. Yang, K. Narasimhan, Webshop: 面向可扩展的现实世界网页交互的基于语言的智能体,《神经信息处理系统进展》(Advances in Neural Information Processing Systems)35 (2022) 20744-20757。

[129] P. Pasupat, T.-S. Jiang, E. Z. Liu, K. Guu, P. Liang, Mapping natural language commands to web elements, arXiv preprint arXiv:1808.09132 (2018).

P. Pasupat, T.-S. Jiang, E. Z. Liu, K. Guu, P. Liang, 将自然语言命令映射到网页元素，arXiv 预印本 arXiv:1808.09132 (2018)。

[130] C. Bai, X. Zang, Y. Xu, S. Sunkara, A. Rastogi, J. Chen, et al., Uib-ert: Learning generic multimodal representations for ui understanding, arXiv preprint arXiv:2107.13731 (2021).

C. Bai, X. Zang, Y. Xu, S. Sunkara, A. Rastogi, J. Chen, 等，Uib-ert: 学习通用多模态表示以理解用户界面，arXiv 预印本 arXiv:2107.13731 (2021)。

[131] 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 desktop graphical user interface automation, arXiv preprint arXiv:2312.13108 (2023).

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

[132] C. Rawles, A. Li, D. Rodriguez, O. Riva, T. Lillicrap, Android in the wild: A large-scale dataset for android device control, arXiv preprint arXiv:2307.10088 (2023).

C. Rawles, A. Li, D. Rodriguez, O. Riva, T. Lillicrap, Android in the wild: 大规模安卓设备控制数据集，arXiv 预印本 arXiv:2307.10088 (2023)。

[133] Y. Liu, H. Duan, Y. Zhang, B. Li, S. Zhang, W. Zhao, Y. Yuan, J. Wang, C. He, Z. Liu, et al., Mmbench: Is your multi-modal model an all-around player?, arXiv preprint arXiv:2307.06281 (2023).

Y. Liu, H. Duan, Y. Zhang, B. Li, S. Zhang, W. Zhao, Y. Yuan, J. Wang, C. He, Z. Liu, 等，Mmbench: 你的多模态模型是全能选手吗？arXiv 预印本 arXiv:2307.06281 (2023)。

[134] W. Yu, Z. Yang, L. Li, J. Wang, K. Lin, Z. Liu, X. Wang, L. Wang, Mm-vet: Evaluating large multimodal models for integrated capabilities, arXiv preprint arXiv:2308.02490 (2023).

W. Yu, Z. Yang, L. Li, J. Wang, K. Lin, Z. Liu, X. Wang, L. Wang, Mm-vet: 评估大型多模态模型的综合能力，arXiv 预印本 arXiv:2308.02490 (2023)。

[135] A. Rastogi, X. Zang, S. Sunkara, R. Gupta, P. Khaitan, Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 34, 2020, pp. 8689-8696.

A. Rastogi, X. Zang, S. Sunkara, R. Gupta, P. Khaitan, 面向可扩展多领域对话智能体: 基于 schema 指导的对话数据集，载于: 人工智能协会 (AAAI) 会议论文集，第 34 卷，2020，页 8689-8696。

[136] H. Ji, J. C. Park, R. Xia, Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing: System demonstrations, in: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations, 2021.

H. Ji, J. C. Park, R. Xia, 第 59 届计算语言学协会年会暨第 11 届国际自然语言处理联合会议系统演示，载于: 第 59 届计算语言学协会年会暨第 11 届国际自然语言处理联合会议系统演示论文集，2021 年。

[137] Y. Zhang, S. Sun, M. Galley, Y.-C. Chen, C. Brockett, X. Gao, J. Gao, J. Liu, B. Dolan, Dialogpt: Large-scale generative pre-training for conversational response generation, arXiv preprint arXiv:1911.00536 (2019).

Y. Zhang, S. Sun, M. Galley, Y.-C. Chen, C. Brockett, X. Gao, J. Gao, J. Liu, B. Dolan, Dialogpt: 大规模生成式预训练用于对话响应生成，arXiv 预印本 arXiv:1911.00536 (2019)。

[138] P. Jaccard, The distribution of the flora in the alpine zone. 1, New phytologist 11 (2) (1912) 37-50.

P. Jaccard, 高山带植物群的分布。1，《新植物学家》(New Phytologist)11 (2) (1912) 37-50。

[139] M. Popović, chrf: character n-gram f-score for automatic mt evaluation, in: Proceedings of the tenth workshop on statistical machine translation, 2015, pp. 392-395.

M. Popović, chrf: 用于自动机器翻译评估的字符 n-gram F 分数，载于: 第十届统计机器翻译研讨会论文集，2015，页 392-395。

[140] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al., Llama: Open and efficient foundation language models, arXiv preprint arXiv:2302.13971 (2023).

H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, 等，Llama: 开放且高效的基础语言模型，arXiv 预印本 arXiv:2302.13971 (2023)。

[141] P. Zhang, G. Zeng, T. Wang, W. Lu, Tinyllama: An open-source small language model, arXiv preprint arXiv:2401.02385 (2024).

P. Zhang, G. Zeng, T. Wang, W. Lu, Tinyllama: 开源小型语言模型，arXiv 预印本 arXiv:2401.02385 (2024)。

[142] M. Javaheripi, S. Bubeck, M. Abdin, J. Aneja, S. Bubeck, C. C. T. Mendes, W. Chen, A. Del Giorno, R. Eldan, S. Gopi, et al., Phi-2: The surprising power of small language models, Microsoft Research Blog (2023).

M. Javaheripi, S. Bubeck, M. Abdin, J. Aneja, S. Bubeck, C. C. T. Mendes, W. Chen, A. Del Giorno, R. Eldan, S. Gopi, 等，Phi-2: 小型语言模型的惊人威力，微软研究博客 (2023)。

[143] S. Hu, Y. Tu, X. Han, C. He, G. Cui, X. Long, Z. Zheng, Y. Fang, Y. Huang, W. Zhao, et al., Minicpm: Unveiling the potential of small language models with scalable training strategies, arXiv preprint arXiv:2404.06395 (2024).

S. Hu, Y. Tu, X. Han, C. He, G. Cui, X. Long, Z. Zheng, Y. Fang, Y. Huang, W. Zhao, 等，Minicpm: 通过可扩展训练策略揭示小型语言模型的潜力，arXiv 预印本 arXiv:2404.06395 (2024)。

[144] Z. Yuan, Z. Li, L. Sun, Tinygpt-v: Efficient multimodal large language model via small backbones, arXiv preprint arXiv:2312.16862 (2023).

Z. Yuan, Z. Li, L. Sun, Tinygpt-v: 通过小型骨干网络实现高效多模态大型语言模型，arXiv 预印本 arXiv:2312.16862 (2023)。

[145] Y. Li, Y. Zhang, C. Wang, Z. Zhong, Y. Chen, R. Chu, S. Liu, J. Jia, Mini-gemini: Mining the potential of multi-modality vision language models, arXiv preprint arXiv:2403.18814 (2024).

Y. Li, Y. Zhang, C. Wang, Z. Zhong, Y. Chen, R. Chu, S. Liu, J. Jia, Mini-gemini: 挖掘多模态视觉语言模型的潜力，arXiv 预印本 arXiv:2403.18814 (2024)。

[146] Y. Zhu, M. Zhu, N. Liu, Z. Ou, X. Mou, J. Tang, Llava-phi: Efficient multi-modal assistant with small language model, arXiv preprint arXiv:2401.02330 (2024).

Y. Zhu, M. Zhu, N. Liu, Z. Ou, X. Mou, J. Tang, Llava-phi: 基于小型语言模型的高效多模态助手，arXiv 预印本 arXiv:2401.02330 (2024)。

[147] S.-y. Liu, Z. Liu, X. Huang, P. Dong, K.-T. Cheng, Llm-fp4: 4-bit floating-point quantized transformers, arXiv preprint arXiv:2310.16836 (2023).

S.-y. Liu, Z. Liu, X. Huang, P. Dong, K.-T. Cheng, Llm-fp4:4 位浮点量化变换器，arXiv 预印本 arXiv:2310.16836 (2023)。

[148] T. Dettmers, M. Lewis, Y. Belkada, L. Zettlemoyer, Gpt3. int8 ( ): 8- bit matrix multiplication for transformers at scale, Advances in Neural Information Processing Systems 35 (2022) 30318-30332.

T. Dettmers, M. Lewis, Y. Belkada, L. Zettlemoyer, Gpt3.int8(): 大规模变换器的 8 位矩阵乘法，《神经信息处理系统进展》第 35 卷 (2022) 30318-30332。

[149] X. Ma, G. Fang, X. Wang, Llm-pruner: On the structural pruning of large language models, Advances in neural information processing systems 36 (2023) 21702-21720.

X. Ma, G. Fang, X. Wang, Llm-pruner: 大型语言模型的结构剪枝，《神经信息处理系统进展》第 36 卷 (2023) 21702-21720。

[150] M. Xia, T. Gao, Z. Zeng, D. Chen, Sheared llama: Accelerating language model pre-training via structured pruning, arXiv preprint arXiv:2310.06694 (2023).

M. Xia, T. Gao, Z. Zeng, D. Chen, Sheared llama: 通过结构化剪枝加速语言模型预训练，arXiv 预印本 arXiv:2310.06694 (2023)。

[151] E. Frantar, S. Ashkboos, T. Hoefler, D. Alistarh, Gptq: Accurate post-training quantization for generative pre-trained transformers, arXiv preprint arXiv:2210.17323 (2022).

E. Frantar, S. Ashkboos, T. Hoefler, D. Alistarh, Gptq: 生成式预训练变换器的精确后训练量化，arXiv 预印本 arXiv:2210.17323 (2022)。

[152] G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, S. Han, Smoothquant: Accurate and efficient post-training quantization for large language models, in: International Conference on Machine Learning, PMLR, 2023, pp. 38087-38099.

G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, S. Han, Smoothquant: 大型语言模型的精确高效后训练量化，国际机器学习大会论文集，PMLR，2023，页 38087-38099。

[153] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, Lora: Low-rank adaptation of large language models, arXiv preprint arXiv:2106.09685 (2021).

E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, Lora: 大型语言模型的低秩适配，arXiv 预印本 arXiv:2106.09685 (2021)。

[154] T. Dettmers, A. Pagnoni, A. Holtzman, L. Zettlemoyer, Qlora: Efficient finetuning of quantized llms, Advances in Neural Information Processing Systems 36 (2024).

T. Dettmers, A. Pagnoni, A. Holtzman, L. Zettlemoyer, Qlora: 量化大型语言模型的高效微调，《神经信息处理系统进展》第 36 卷 (2024)。

[155] W. Yin, M. Xu, Y. Li, X. Liu, Llm as a system service on mobile devices, arXiv preprint arXiv:2403.11805 (2024).

W. Yin, M. Xu, Y. Li, X. Liu, 移动设备上的语言模型系统服务，arXiv 预印本 arXiv:2403.11805 (2024)。

[156] R. Jin, Q. Xu, M. Wu, Y. Xu, D. Li, X. Li, Z. Chen, Llm-based knowledge pruning for time series data analytics on edge-computing devices, arXiv preprint arXiv:2406.08765 (2024).

R. Jin, Q. Xu, M. Wu, Y. Xu, D. Li, X. Li, Z. Chen, 基于语言模型的边缘计算设备时间序列数据分析知识剪枝，arXiv 预印本 arXiv:2406.08765 (2024)。

[157] Y. Zhao, M. Lin, H. Tang, Q. Wu, J. Wang, Merino: Entropy-driven design for generative language models on iot devices, arXiv preprint arXiv:2403.07921 (2024).

Y. Zhao, M. Lin, H. Tang, Q. Wu, J. Wang, Merino: 面向物联网设备的生成式语言模型的熵驱动设计，arXiv 预印本 arXiv:2403.07921 (2024)。

[158] Z. Zhang, X. Zhang, W. Xie, Y. Lu, Responsible task automation: Empowering large language models as responsible task automators, arXiv preprint arXiv:2306.01242 (2023).

Z. Zhang, X. Zhang, W. Xie, Y. Lu, 负责任的任务自动化: 赋能大型语言模型成为负责任的任务自动执行者，arXiv 预印本 arXiv:2306.01242 (2023)。

[159] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, P. Fung, Survey of hallucination in natural language generation, ACM Computing Surveys 55 (12) (2023) 1-38.

Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, P. Fung，关于自然语言生成中幻觉现象的综述，ACM 计算机综述 55 (12) (2023) 1-38。

[160] V. Rawte, A. Sheth, A. Das, A survey of hallucination in large foundation models, arXiv preprint arXiv:2309.05922 (2023).

V. Rawte, A. Sheth, A. Das，大型基础模型中幻觉现象的综述，arXiv 预印本 arXiv:2309.05922 (2023)。

[161] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, et al., A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions, arXiv preprint arXiv:2311.05232 (2023).

L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin 等，大型语言模型中幻觉现象的综述: 原理、分类、挑战与未解问题，arXiv 预印本 arXiv:2311.05232 (2023)。

[162] Y. Ge, Y. Ren, W. Hua, S. Xu, J. Tan, Y. Zhang, Llm as os, agents as apps: Envisioning aios, agents and the aios-agent ecosystem, arXiv e-prints (2023) arXiv-2312.

Y. Ge, Y. Ren, W. Hua, S. Xu, J. Tan, Y. Zhang，将大型语言模型 (LLM) 视为操作系统，代理视为应用程序: 展望 AI 操作系统 (AIOS)、代理及 AIOS-代理生态系统，arXiv 电子预印本 (2023) arXiv-2312。