KG-RAG: Enhancing GUI Agent Decision-Making via Knowledge Graph-Driven Retrieval-Augmented Generation

KG-RAG: 通过知识图驱动的检索增强生成提升 GUI 代理决策

Ziyi Guan [1,2*†] , Jason Chun Lok Li [1*] , Zhijian Hou [1*] , Pingping Zhang [3] , Donglai Xu [5] , Yuzhi Zhao [1‡] , Mengyang Wu [1] , Jinpeng Chen [3] , Nguyen Thanh Toan [1] , Pengfei Xian [1] , Wenao Ma [1] , Shengchao Qin [3,4] , Graziano Chesi [2] , Ngai Wong [2]

关子怡 [1,2*†] ， 李俊落 [1*] ， 侯志坚 [1*] ， 张萍萍 [3] ， 徐东来 [5] ， 赵玉芝 [1‡] ， 吴梦阳 [1] ， 陈金鹏 [3] ， 阮清全 [1] ， 冼鹏飞 [1] ， 马文奥 [1] ， 秦胜超 [3,4] ， Graziano Chesi [2] ， 黄毅 [2]

[1] Huawei Hong Kong Research Center [2] The University of Hong Kong [3] City University of Hong Kong [3] Guangzhou Institute of Technology, Xidian University [4] ICTT and ISN Laboratory, Xidian University [5] Independent Researcher zyguan@eee.hku.hk; yzzhao2-c@my.cityu.edu.hk

[1] 华为香港研究中心 [2] 香港大学 [3] 香港城市大学 [3] 广州科技学院，西安电子科技大学 [4] 西电 ICTT 与 ISN 实验室 [5] 独立研究员 zyguan@eee.hku.hk; yzzhao2-c@my.cityu.edu.hk

Abstract

Despite recent progress, Graphic User Interface (GUI) agents powered by Large Language Models (LLMs) struggle with complex mobile tasks due to limited app-specific knowledge. While UI Transition Graphs (UTGs) offer structured navigation representations, they are underuti-lized due to poor extraction and inefficient integration. We introduce KG-RAG, a Knowledge Graph-driven Retrieval-Augmented Generation framework that transforms fragmented UTGs into structured vector databases for efficient real-time retrieval. By leveraging an intent-guided LLM search method, KG-RAG generates actionable navigation paths, enhancing agent decision-making. Experiments across diverse mobile apps show that KG-RAG outperforms existing methods, achieving a 75.8% success rate (8.9% improvement over AutoDroid), 84.6% decision accuracy (8.1% improvement), and reducing average task steps from 4.5 to 4.1. Additionally, we present KG-Android-Bench and KG-Harmony-Bench, two benchmarks tailored to the Chinese mobile ecosystem for future research. Finally, KG-RAG transfers to web/desktop (+40% SR on Weibo-web; +20% on QQ Music-desktop), and a UTG cost ablation shows accuracy saturates at $\sim$ 4h per complex app, enabling practical deployment tradeoffs.

摘要

尽管近年来取得进展，基于大模型的图形用户界面 (GUI) 代理在处理复杂移动任务时仍受限于应用特定知识匮乏。尽管 UI 转移图 (UTG) 提供了结构化的导航表示，但因提取质量差和整合低效而未被充分利用。我们提出 KG-RAG，一种知识图驱动的检索增强生成框架，将零碎的 UTG 转换为结构化向量数据库以实现高效的实时检索。通过意图引导的 LLM 搜索方法，KG-RAG 生成可执行的导航路径，提升代理决策能力。跨多款移动应用的实验显示，KG-RAG 优于现有方法，成功率达 75.8%(较 AutoDroid 提升 8.9%)、决策准确率达 84.6%(提升 8.1%)，平均任务步骤从 4.5 降至 4.1。此外，我们提供了面向中文移动生态的两个基准 KG-Android-Bench 和 KG-Harmony-Bench，供后续研究使用。最后，KG-RAG 可迁移至 web/桌面端 (微博 web 提升 +40% 成功率；QQ 音乐桌面端 +20%)，且 UTG 成本消融表明复杂应用的准确率在每个应用 ∼ 4h 时趋于饱和，支持实用部署权衡。

1 Introduction

1 引言

In recent years, LLM-based GUI agents (Zhang et al., 2023; Lee et al., 2023; Yoon et al., 2023; Hong et al., 2024; You et al., 2024; Wen et al., 2024; Wang et al., 2025; Qin et al., 2025) have advanced in interacting with and navigating mobile applications. However, efficiently completing complex tasks remains challenging due to their limited app-specific knowledge, particularly when faced with unfamiliar user interfaces (UIs) and unconventional navigation logic. For example, as shown in Figure 1, the ”Privacy Policy” page is deeply embedded within the ”About Pomodoro” page, making it difficult to locate without a clear user menu, even for human users encountering the app for the first time. App UI Transition Graphs (UTGs), structured representations of app navigational flows, have emerged as promising solutions to enhance agents' navigational capabilities. Despite their potential, UTGs face significant barriers, including low-quality graph extraction from apps and inefficient integration into real-time decision-making processes.

近年来，基于 LLM 的 GUI 代理 (Zhang 等，2023；Lee 等，2023；Yoon 等，2023；Hong 等，2024；You 等，2024；Wen 等，2024；Wang 等，2025；Qin 等，2025) 在与移动应用交互与导航方面取得进展。但由于缺乏应用特定知识，尤其面对陌生界面和非常规导航逻辑时，高效完成复杂任务仍具挑战。例如如图 1 所示，“隐私政策”页面深藏在“关于番茄钟”页面内，即便是首次使用该应用的人也难以在没有清晰用户菜单的情况下找到。应用 UI 转移图 (UTG) 作为应用导航流程的结构化表示，被视为增强代理导航能力的有前途方案。尽管如此，UTG 仍面临显著障碍，包括从应用中提取的图质量低下以及在实时决策中整合效率低。

To address the limitations of existing GUI agents, we propose KG-RAG, a Knowledge Graph-driven Retrieval-Augmented Generation framework designed to transform incomplete UTGs into structured vector databases, enabling rapid and precise information retrieval during online execution. Our approach employs an LLM-powered offline graph-search algorithm to systematically preprocess low-quality UTGs, converting incomplete or fragmented graphs into structured, vector-based knowledge repositories optimized for retrieval-augmented generation. During online execution, the agent dynamically queries this vector-based repository using embedding-based similarity search, rapidly retrieving relevant navigational paths and app-specific information tailored precisely to the user's intent. This retrieval-augmented approach significantly enhances the agent's decision-making, reducing reliance on extensive real-time exploration and enabling swift, adaptive responses to complex, dynamic app scenarios. Extensive experiments across diverse mobile apps demonstrate that agents equipped with KG-RAG achieve notably higher task completion rates and require fewer steps

per task, highlighting KG-RAG's effectiveness in supplementing online agents with critical, domain-specific knowledge. The key contributions of our work are:

为解决现有 GUI 代理的局限，我们提出 KG-RAG，一种知识图驱动的检索增强生成框架，旨在将不完整的 UTG 转换为结构化向量数据库，从而在在线执行时实现快速精确的信息检索。我们的方法采用由 LLM 驱动的离线图搜索算法系统地预处理低质量 UTG，将不完整或碎片化的图转换为针对检索增强生成优化的结构化向量知识库。在在线执行阶段，代理使用基于嵌入的相似度搜索动态查询该向量知识库，快速检索与用户意图精确匹配的相关导航路径和应用特定信息。该检索增强方法显著提升代理决策能力，减少对大规模实时探索的依赖，使其能够在复杂动态的应用场景中快速、灵活响应。大量跨应用实验表明，配备 KG-RAG 的代理在任务完成率上有显著提升且平均步骤更少，凸显 KG-RAG 在为在线代理补充关键领域知识方面的有效性。我们的主要贡献包括:

---

*These authors contributed equally.

* 这些作者贡献相同。

† Internship with Huawei Hong Kong Research Center.

† 在华为香港研究中心实习。

*Corresponding Author and Project Lead.

* 通讯作者与项目负责人。

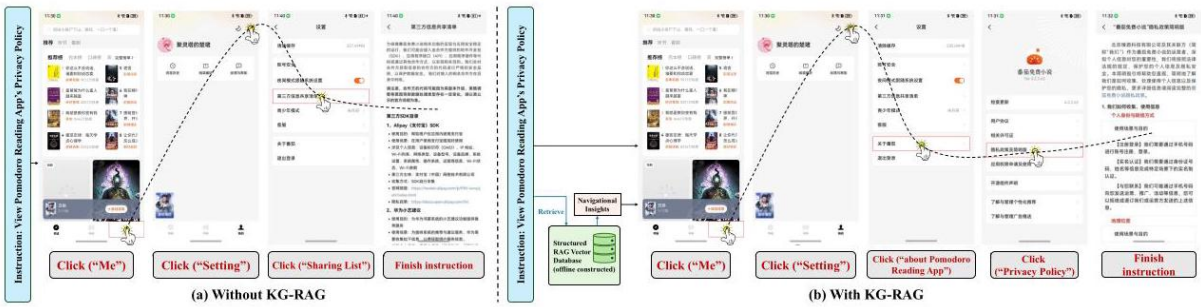

Figure 1: Improved Task Execution for "View Privacy Policy" in Tomato Novel App Using Graph-based RAG. (a) Without KG-RAG: Fails to identify the correct navigation path to access the "View Privacy Policy" page. (b) With KG-RAG: Successfully generates the correct path by leveraging knowledge graph-based information.

图 1: 在番茄小说应用中使用基于图的 RAG 改进"查看隐私政策"任务执行。(a) 无 KG-RAG: 未能识别到达"查看隐私政策"页面的正确导航路径。(b) 有 KG-RAG: 通过利用基于知识图的信息成功生成了正确路径。

- Proposing KG-RAG, a novel pipeline that transforms incomplete or fragmented UI Transition Graphs (UTGs) into a structured, vector-based knowledge database, optimized for rapid and precise real-time retrieval during online execution.

  - 提出 KG-RAG，一种新颖流水线，将不完整或碎片化的 UI 转移图 (UTG) 转换为结构化的向量化知识库，优化以支持在线执行期间的快速且精确的实时检索。

- Introducing KG-Android-Bench and KG-Harmony-Bench, two comprehensive, cross-platform benchmarks tailored for evaluating GUI agents in the diverse Chinese mobile ecosystem.

  - 引入 KG-Android-Bench 和 KG-Harmony-Bench，两个面向多样化中国移动生态的跨平台完整基准，专为评估 GUI 智能体而设。

- Demonstrating through rigorous evaluation across diverse mobile apps that KG-RAG serves as a plug-and-play module, achieving a 75.8% task success rate (8.9% improvement over prior methods) and reducing average task steps from 4.5 to 4.1, significantly enhancing the efficiency and effectiveness of existing GUI agents.

  - 通过对多款移动应用的严格评估证明，KG-RAG 可作为即插即用模块，使任务成功率达到 75.8%(较先前方法提升 8.9%)，并将平均任务步数从 4.5 降至 4.1，显著提升现有 GUI 智能体的效率与效果。

2 Related Work

2 相关工作

Recent LLM-driven agents have made significant progress in automating mobile UI tasks. Auto-Droid (Wen et al., 2024), the most relevant to our work, constructs an app's UTG offline and uses an LLM online to plan actions, significantly outperforming a GPT-4 baseline in task success rate. However, AutoDroid does not fully leverage the UTG during execution for rapid retrieval of knowledge. Other advanced methods, such as Mobile-Agent-v2 (Wang et al., 2025) and UI-TARS (Qin et al., 2025), focus on collaborative agents and end-to-end learning, respectively, achieving strong results on GUI benchmarks. Yet, none of these approaches explicitly leverage a structured knowledge graph of the app's UI for decision-making. Our proposed KG-RAG fills this gap by providing agents with a UTG-derived knowledge graph, enabling efficient retrieval of navigational knowledge. When integrated with agents like MobileAgent-v2 or UI-TARS, KG-RAG significantly boosts their task success rates, as shown in Section 5.3.

近期基于大模型的智能体在自动化移动 UI 任务上取得了重要进展。与我们工作最相关的 AutoDroid(Wen 等，2024) 离线构建应用的 UTG 并在线使用 LLM 进行行动规划，在任务成功率上大幅超越 GPT-4 基线。然而，AutoDroid 在执行过程中并未充分利用 UTG 以实现快速知识检索。其他先进方法，如 Mobile-Agent-v2(Wang 等，2025) 和 UI-TARS(Qin 等，2025)，分别侧重于协作智能体和端到端学习，在 GUI 基准上取得了良好成绩，但均未明确利用应用 UI 的结构化知识图进行决策。我们提出的 KG-RAG 填补了这一空白，为智能体提供由 UTG 派生的知识图，从而实现导航知识的高效检索。将 KG-RAG 与 MobileAgent-v2 或 UI-TARS 等智能体整合后，可显著提升其任务成功率，如 5.3 节所示。

Our proposed approach KG-RAG differs from previous GUI agents by integrating structured knowledge graphs derived from UTGs with LLM-based reasoning. While earlier agents either un-derutilized app-specific graphs or lacked any structured memory, KG-RAG employs a hierarchical knowledge graph, pre-processed for rapid retrieval, to enhance decision-making. This structured memory provides navigation insights that are challenging for LLMs alone to infer. Furthermore, KG-RAG's plug-and-play design allows it to be seamlessly integrated into various agent architectures, as demonstrated with MobileAgent-v2 and UI-TARS. This flexibility and the combination of offline UTG processing with online retrieval-augmented decision-making mark a significant advancement, leading to higher success rates and greater efficiency in GUI agent tasks.

我们提出的 KG-RAG 与以往 GUI 智能体的不同在于将由 UTG 派生的结构化知识图与基于 LLM 的推理相结合。早期智能体要么未充分利用特定应用的图，要么缺乏任何结构化记忆，而 KG-RAG 采用分层知识图并预处理以支持快速检索，从而增强决策能力。这种结构化记忆提供了仅靠 LLM 难以推断的导航洞见。此外，KG-RAG 的即插即用设计使其能够无缝集成到各种智能体架构中，如与 MobileAgent-v2 和 UI-TARS 的结合所示。这种灵活性及将离线 UTG 处理与在线检索增强决策相结合的做法标志着一项重要进步，带来更高的成功率和更强的 GUI 任务效率。

3 Method

3 方法

This section presents KG-RAG, a framework that enhances online agent decision-making by fully leveraging the rich information embedded within UTGs. An overview of KG-RAG is provided in Figure 2.

本节介绍 KG-RAG，该框架通过充分利用 UTG 中蕴含的丰富信息来增强在线智能体的决策能力。KG-RAG 的概述见图 2。

3.1 UTG Extraction

3.1 UTG 提取

To effectively extract UTGs from mobile applications, we build upon the methodology proposed by DroidBot (Li et al., 2017), adapting it significantly to meet our framework's requirements. Specifically, we introduce a dedicated extraction tool dubbed as xTester (App Test Executor Use Case Execution Framework) as shown in Figure 2(a). It is designed to systematically navigate mobile app interfaces, identify interactive UI components, and document their interactions. The outcome is a structured knowledge graph that encompasses the app's UI layouts, control structures, and potential user interactions.

为有效从移动应用中提取 UTG，我们基于 DroidBot(Li 等，2017) 提出的方法进行了改进，针对本框架需求做出重大调整。具体地，我们引入了名为 xTester(应用测试执行用例执行框架) 的专用提取工具，如图 2(a) 所示。它旨在系统性地遍历移动应用界面，识别可交互的 UI 组件并记录其交互，产出涵盖应用 UI 布局、控制结构和潜在用户交互的结构化知识图。

This knowledge graph is represented in a hierarchical JSON format, capturing essential information including app metadata (e.g., product ID, app name, and package name), UI components description, screen

description, and actionable interactions associated with specific widgets. Actions, such as swipe gestures, text inputs, and button clicks, are annotated and linked to corresponding UI elements, providing a comprehensive representation of each screen's functionality.

该知识图以层级 JSON 格式表示，捕获关键信息，包括应用元数据 (如产品 ID、应用名称和包名)、UI 组件描述、屏幕描述以及与特定控件相关的可执行交互。诸如滑动手势、文本输入和按钮点击等操作被注释并链接到对应的 UI 元素，为每个屏幕的功能提供了全面表示。

For simpler English-language apps sourced from DroidTask (Wen et al., 2024), we utilize a 1-hour automated exploration per app using xTester. This procedure efficiently captures nearly all relevant app content due to the straightforward UI structures involved. In contrast, for the 30 more complex Chinese-language applications we specifically constructed, we perform an extensive, in-depth exploration for 8 hours. This extended analysis ensures comprehensive coverage of intricate UI states and interactions, significantly surpassing typical extraction methods in both depth and breadth.

对于来自 DroidTask(Wen 等，2024) 的结构较为简单的英文应用，我们对每个应用使用 xTester 进行 1 小时的自动化探索。由于 UI 结构较为直接，此过程能高效捕获几乎所有相关内容。相比之下，对于我们专门构建的 30 款更复杂的中文应用，我们进行了长达 8 小时的深度探索。这一扩展分析确保全面覆盖复杂的 UI 状态和交互，在深度与广度上显著超越典型的提取方法。

Overall, our rigorous UTG extraction approach serves as a crucial foundation for accurately capturing UI widget descriptions and detailed app layout information, directly contributing to the effective construction of the KG-RAG vector database. By providing high-quality and structured vector em-beddings of navigational paths, our method enables rapid retrieval and significantly enhanced decision-making capabilities in automated mobile app interactions.

总体而言，我们严格的 UTG 提取方法为准确捕捉 UI 小部件描述和详细的应用布局信息提供了关键基础，直接促进了 KG-RAG 向量数据库的有效构建。通过为导航路径提供高质量且结构化的向量嵌入，我们的方法实现了快速检索并显著增强了自动化移动应用交互的决策能力。

3.2 Offline Pathfinding via Intent-Guided LLM Search

3.2 通过意图引导的 LLM 离线路径查找

The offline pathfinding component is central to KG-RAG and consists of two key parts: (1) the intent generation module and (2) the LLM search module. Together, they enable effective navigation through imperfect UTGs, even with incomplete or missing paths. This component significantly enhances the efficiency and robustness of offline trajectory discovery.

离线路径查找组件是 KG-RAG 的核心，包含两个关键部分:(1) 意图生成模块和 (2) LLM 搜索模块。二者协同，使得即使在不完美的 UTG 中、路径不完整或缺失时也能有效地导航。该组件显著提高了离线轨迹发现的效率与鲁棒性。

The intent generation module, illustrated in Figure 2(b), begins by analyzing screenshots (nodes) extracted from UTGs to identify plausible user intents for each app screen. In practice, we utilize a vision-

language model (VLM) to automatically infer these user intents from visual context. Next, a powerful instruction-tuned LLM decomposes each high-level intent into a structured sequence of intermediate milestones. These milestones represent incremental, clearly defined subgoals guiding the agent through complex UI interactions. By breaking down user intents into manageable intermediate steps, our approach reduces ambiguity and prevents premature task termination.

意图生成模块 (见图 2(b)) 首先分析从 UTG 提取的截图 (节点)，为每个应用界面识别可能的用户意图。实际中，我们利用视觉-语言模型 (VLM) 从视觉上下文自动推断这些用户意图。随后，强大的指令微调 LLM 将每个高层意图分解为结构化的中间里程碑序列。这些里程碑代表逐步、明确的子目标，引导代理完成复杂的 UI 交互。通过将用户意图拆解为可管理的中间步骤，我们的方法降低了歧义并防止过早终止任务。

The generated intents and milestones are then utilized by the LLM trajectory scoring module, as depicted in Figure 2(c). Given a set of candidate trajectories extracted from the UTG, this module assigns each trajectory both a coarse and a fine-grained score. Specifically, the LLM is used to predict the likelihood of each trajectory successfully achieving the given milestones. The scoring is performed by tokenizing the LLM output, extracting logits corresponding to milestone completion predictions, and computing probability distributions via a softmax operation over the relevant logits indices. For a given user intent (with m milestones), we prompt the LLM to predict the likelihood of completing 0 through m milestones along a candidate trajectory. We then apply a softmax over the relevant output logits (Algorithm 1) to obtain a probability distribution over milestone completion counts. The highest probability in this distribution (corresponding to reaching a certain milestone) is taken as the trajectory's progress score. Next, we compute a proximity score for the trajectory (Algorithm 2) by measuring how closely the LLM's probability distribution aligns with an ideal monotonically decreasing sequence (favoring trajectories that complete milestones in order). Each trajectory is primarily ranked by its progress score, with the proximity score used to break ties among similar candidates.

生成的意图和里程碑随后被 LLM 轨迹评分模块使用，如图 2(c) 所示。针对从 UTG 提取的一组候选轨迹，该模块为每条轨迹分配粗粒度和细粒度评分。具体而言，LLM 被用于预测每条轨迹实现给定里程碑的可能性。评分通过对 LLM 输出进行分词，提取对应里程碑完成预测的 logits，并对相关 logits 索引应用 softmax 运算来计算概率分布来完成。对于一个具有 m 个里程碑的用户意图，我们提示 LLM 预测在一条候选轨迹上完成 0 到 m 个里程碑的可能性。然后我们对相关输出 logits 应用 softmax(算法 1)，以获得里程碑完成计数的概率分布。该分布中概率最高者 (对应达到某个里程碑) 被视为轨迹的进展得分。接着，我们通过测量 LLM 的概率分布与理想单调递减序列的吻合程度 (偏好按顺序完成里程碑的轨迹) 来计算轨迹的接近度得分 (算法 2)。每条轨迹主要按其进展得分排序，接近度得分用于在相似候选中破除平局。

Guided by these scoring metrics, we perform a breadth-first search (BFS) to explore the UTG for high-quality trajectories (Algorithm 3). The BFS expands possible paths in increments of the given step size and scores new candidates in batches using the LLM. Any trajectory that fails to achieve a minimum progress (milestone completion) threshold is pruned early, which focuses computation on promising paths. This batched BFS strategy balances thorough exploration with efficiency: it allows parallel LLM evaluations of multiple paths and limits search depth to max_depth. The outcome is a set of top-ranked valid trajectories for each intent. Finally, these trajectories are passed through a summarization module that condenses each sequence of UI actions into a concise description, ready to be stored in the knowledge base for online use.

在这些评分度量的引导下, 我们执行广度优先搜索 (BFS) 以在 UTG 中探索高质量轨迹 (算法 3)。BFS 按给定步长增量扩展可能路径，并使用 LLM 对新候选按批评分。任何未达到最小进展 (里程碑完成) 阈值的轨迹会被提前剪枝，从而将计算集中在有前景的路径上。该分批 BFS 策略在全面探索与效率之间取得平衡: 允许对多条路径并行进行 LLM 评估，并将搜索深度限制为 max_depth。最终得到的是针对每个意图的若干顶级有效轨迹。最后，这些轨迹通过摘要模块将每个 UI 操作序列压缩为简洁描述，准备存储到知识库以供在线使用。

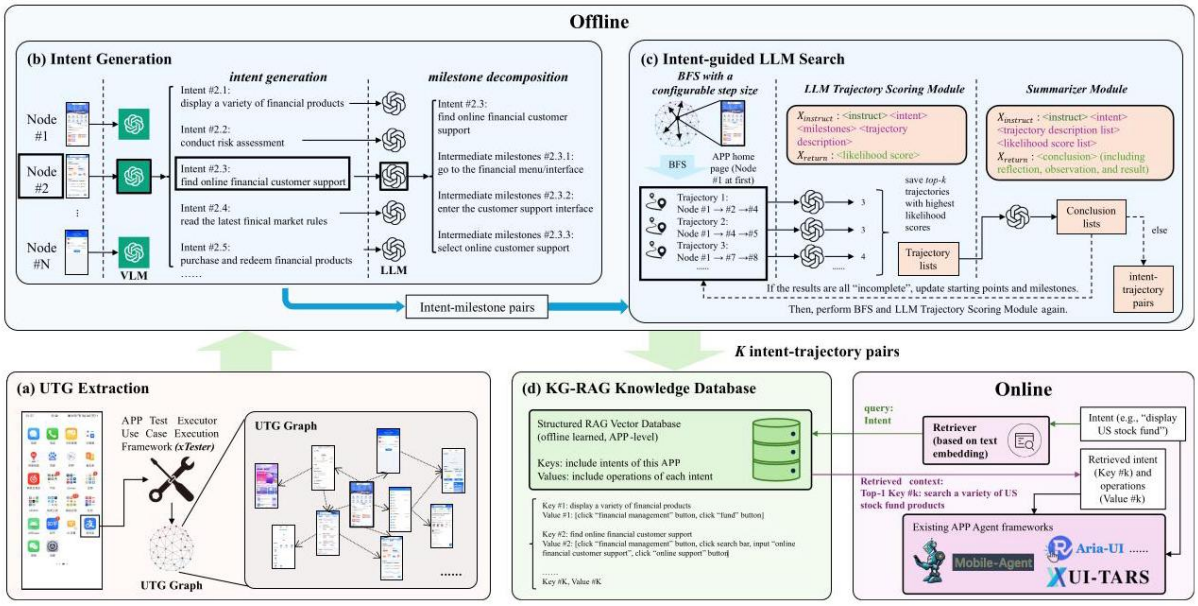

Figure 2: Overview of KG-RAG architecture: (a) UTG extraction capturing app UI navigation structures; (b) Intent generation suggesting plausible user intents and decomposing them into intermediate milestones; (c) Intent-guided LLM search efficiently identifying candidate trajectories aligned with user intents; and (d) KG-RAG knowledge database supporting effective online mobile app interactions.

图 2:KG-RAG 架构概览:(a) 捕捉应用 UI 导航结构的 UTG 提取；(b) 意图生成，建议合理用户意图并将其分解为中间里程碑；(c) 意图引导的 LLM 搜索，高效识别与用户意图一致的候选轨迹；(d) 支持高效在线移动应用交互的 KG-RAG 知识数据库。

Algorithm 1: Softmax Computation

算法 1:Softmax 计算

Input: x : Logits array

输入: x :Logits 数组

start_ind: Start index for slicing the logits

start_ind: 用于切片 logits 的起始索引

end_ind: End index for slicing the logits

end_ind: 用于切片 logits 的结束索引

temperature: Softmax temperature (default = 1)

temperature:Softmax 温度 (默认 = 1)

Output: softmax_probs: Probability distribution

输出:softmax_probs: 概率分布

over indices [start_ind, end_ind]

在索引 [start_ind, end_ind] 上

Procedure:

过程:

$$x_{\text{slice}} \leftarrow x[\text{ start\_ind} : \text{end\_ind }]$$
$$z \leftarrow \exp\left(\frac{x_{\text{slice}}}{\text{temperature}}\right)$$
$$\text{softmax\_probs} \leftarrow \frac{z}{\sum z}$$

$$\text{softmax\_probs} \leftarrow \frac{z}{\sum z}$$

return softmax_probs

return softmax_probs

3.3 Knowledge Graph-Augmented Online Execution

3.3 知识图增强的在线执行

During online execution, the KG-RAG agent leverages a vector database of offline-generated intent-trajectory pairs to rapidly retrieve relevant navigation knowledge. Each entry in our structured vector database is a key-value pair: the key is a high-dimensional embedding of an intent discovered offline, and the value is the corresponding trajectory (sequence of UI actions to fulfill that intent). These intent-trajectory pairs are obtained from the offline phase: the VLM infers plausible user intents from app screens (Figure 2(b)), and the LLM-based search finds successful trajectories for those intents (Figure 2(c)). We encode each intent (along with its trajectory) into a vector using a text embedding model, creating a repository of navigational knowledge optimized for fast similarity search.

在在线执行期间，KG-RAG 代理利用离线生成的意图-轨迹对向量数据库来快速检索相关的导航知识。我们结构化向量数据库中的每一项都是一个键值对: 键是离线发现的意图的高维嵌入，值是对应的轨迹 (为实现该意图的一系列 UI 操作)。这些意图-轨迹对来自离线阶段:VLM 从应用界面推断出合理的用户意图 (图 2(b))，基于 LLM 的搜索为这些意图找到成功的轨迹 (图 2(c))。我们使用文本嵌入模型将每个意图 (及其轨迹) 编码为向量，创建一个为快速相似性搜索优化的导航知识库。

Algorithm 2: Proximity Score Calculation

算法 2: 接近度得分计算

Input: pdf : Probability distribution function as a list

输入: pdf : 以列表形式的概率分布函数

of probabilities

概率

Output: proximity_score: Scalar value indicating

输出: proximity_score: 标量，指示

alignment with ideal descending order

与理想降序排列的一致性

Procedure:

过程:

ranked_indices ← indices of pdf sorted in

ranked_indices ← pdf 的索引按

descending order

降序排列

ideal_order ← list from $n-1$ down to 0, where n

ideal_order ← 列表从 $n-1$ 到 0，其中 n

is length of pdf

是 pdf 的长度

proximity_score ←

proximity_score ←

$$-\sum_{i=0}^{n-1}(\text{ ideal\_order }[i]-\text{ ranked\_indices }[i])^2$$

return proximity_score

返回 proximity_score

During online execution, the agent leverages this repository to retrieve relevant guidance in real time. The user's current task instruction is encoded into a query vector (using the same text embedding model) and compared against the stored intent vectors via cosine similarity. The agent then retrieves the top- K most similar intent-trajectory entries (Figure 2(d)). Each retrieved trajectory serves as contextual knowledge, suggesting a proven navigation path for the agent to follow. For example, as illustrated in Figure 2(d), if the user needs to "display US stock fund" information, the agent will retrieve the stored trajectory that accomplishes this task, providing the sequence of UI actions required to reach the stock fund content. By following such retrieved trajectories, the KG-RAG-enhanced agent can quickly navigate to the target state instead of relying on trial-and-error exploration. This retrieval-augmented execution allows the agent to respond adaptively to new tasks using the collective knowledge encoded in the UTG-derived graph memory.

在在线执行期间，代理利用该知识库实时检索相关指导。用户当前的任务指令被编码为查询向量 (使用相同的文本嵌入模型)，并通过余弦相似度与存储的意图向量进行比较。然后代理检索最相似的前 K 个意图-轨迹条目 (图 2(d))。每个检索到的轨迹作为上下文知识，提供一个已验证的导航路径供代理参考。例如，如图 2(d) 所示，如果用户需要“显示美国股票基金”信息，代理会检索到完成此任务的存储轨迹，提供到达股票基金内容所需的 UI 操作序列。通过遵循这些检索到的轨迹，KG-RAG 增强的代理可以快速导航到目标状态，而无需依赖反复试错探索。这种检索增强的执行使代理能够利用 UTG 派生的图状记忆中编码的集体知识，自适应地响应新任务。

Algorithm 3: LLM BFS Search

算法 3:LLM 广度优先搜索

Input: User intent, UTG graph, start_node, LLM

输入: 用户意图，UTG 图，start_node，LLM

model, threshold, step_size, max_depth,

模型，阈值，步长，最大深度，

top-K

top-K

Output: valid_trajectories achieving the intent

输出: 达到意图的 valid_trajectories

Initialize: $Q \leftarrow$ [(start_node , [start_node])] ,

初始化: $Q \leftarrow$ [(start_node , [start_node])] ,

valid_trajectories $\leftarrow \varnothing$

valid_trajectories $\leftarrow \varnothing$

while $Q \neq \varnothing$ and depth < max_depth do

当 $Q \neq \varnothing$ 且 depth < max_depth 时

depth $\leftarrow$ depth + step_size

depth $\leftarrow$ depth + step_size

Expand Q to generate candidates; remove

扩展 Q 以生成候选；移除

duplicates and loops

重复项和环路

foreach batch in candidates do

对于候选中的每个批次

Compute LLM scores for batch

为批次计算 LLM 分数

foreach trajectory do

对于每条轨迹

if score $\geq$ threshold then

如果 score $\geq$ threshold 则

Add to valid_trajectories

加入 valid_trajectories

end

结束

end

结束

end

结束

Keep top- K candidates (by score, prefer shorter

保留前 K 名候选 (按分数，偏好更短的路径

paths) and update Q

) 并更新 Q

end

结束

return valid_trajectories

返回 valid_trajectories

Table 1: Comparison of KG-Android-Bench and KG-Harmony-Bench with existing GUI agent benchmarks

表 1:KG-Android-Bench 与 KG-Harmony-Bench 与现有 GUI 代理基准的比较

Benchmark	No. of Tasks	No. of Apps
AndroidLab	138	9
DroidTask	162	12
KG-Android-Bench (Ours)	300	30
KG-Harmony-Bench (Ours)	150	15

基准测试	任务数	应用数
AndroidLab	138	9
DroidTask	162	12
KG-Android-Bench(我们的)	300	30
KG-Harmony-Bench(我们的)	150	15

4 Benchmark Construction

4 基准构建

We present KG-Android-Bench and KG-Harmony-Bench, two comprehensive cross-platform benchmarks for evaluating GUI agents in Chinese mobile ecosystems. As shown in Table 1, KG-Android-Bench significantly outperforms existing benchmarks with 300 tasks across 30 mainstream Chinese applications, compared to DroidTask's 162 tasks and AndroidLab's 138 tasks. This expanded scale, coupled with support for 10 functional categories (as shown in Table 2), offers a more diverse and thorough evaluation of agent capabilities across varied mobile environments.

我们提出了 KG-Android-Bench 与 KG-Harmony-Bench 两个面向中文移动生态的跨平台图谱基准，用于评估 GUI 代理。如表 1 所示，KG-Android-Bench 在 30 个主流中文应用中包含 300 个任务，显著优于 DroidTask 的 162 个任务和 AndroidLab 的 138 个任务。此规模扩展以及对 10 个功能类别的支持 (见表 2)，为在多样化移动环境中更全面地评估代理能力提供了更丰富的测试。

Table 2: Applications in KG-Android-Bench covering 10 functional categories.

表 2:KG-Android-Bench 中覆盖 10 个功能类别的应用。

Category	Applications
Music & Audio	QQ Music, NetEase Cloud Music, Himalaya FM
Video & Entertainment	Douyin(Chinese Tiktok), Youku, Douyu, WeSing
Social & Communication	Weibo, Zhihu, Baihe
Navigation & Travel	Amap, Ctrip, Zhixing Train Tickets
E-commerce & Retail	Taobao, Vipshop, Dianping
Food Services	Meituan Takeaway, Pupu Supermarket
Health & Lifestyle	Keep, Moji Weather, Daily Alarm Clock
News & Reading	Tomato Novel, Jinri Toutiao, Hupu, Dongchedi
Productivity	Baidu Browser, NetEase Mail, Youdao Dictionary
Photo & Video Editing	Xingtu, CapCut

类别	应用
音乐与音频	QQ 音乐、网易云音乐、喜马拉雅 FM
视频与娱乐	抖音、优酷、斗鱼、唱吧
社交与通讯	微博、知乎、百合
导航与出行	高德地图、携程、智行火车票
电商与零售	淘宝、唯品会、大众点评
餐饮服务	美团外卖、朴朴超市
健康与生活方式	Keep、墨迹天气、每日闹钟
新闻与阅读	番茄小说、今日头条、虎扑、懂车帝
办公与效率	百度浏览器、网易邮箱、有道词典
照片与视频编辑	醒图、剪映

A key innovation of KG-Android-Bench is its use of structured knowledge graphs and intent-action mappings, which provide a detailed map of app interfaces. These elements enable more realistic assessments of

agents' performance on high-frequency mobile interactions, such as those found in social media, e-commerce, navigation, and fitness tracking. Unlike existing benchmarks that rely solely on basic UI traces, KG-Android-Bench encodes the full navigation flow and task execution sequences within each app.

KG-Android-Bench 的一个关键创新是其使用结构化知识图谱和意图-动作映射，这些提供了应用界面的详细地图。这些元素使得对代理在高频移动交互 (如社交媒体、电子商务、导航和健身追踪) 中的表现进行更真实的评估成为可能。不同于仅依赖基础界面轨迹的现有基准，KG-Android-Bench 对每个应用内的完整导航流程和任务执行序列进行了编码。

For instance, to complete the task "View App Privacy Policy" in the Tomato Novel app shown in Figure 1, KG-Android-Bench defines a sequence of the following actions: Step 1: Open Profile by tapping the "My Profile" button (the user's profile/account section). Step 2: Open Settings by tapping the "Settings" gear icon. Step 3: About Section by tapping "About Tomato" (the about page of the app). Step 4: Privacy Policy by tapping "Privacy Policy" to view the content of the policy.

例如，为了完成图 1 中番茄小说应用的“查看应用隐私政策”任务，KG-Android-Bench 定义了如下动作序列: 步骤 1: 通过点击“我的资料”按钮 (用户的资料/账户部分) 打开个人资料。步骤 2: 通过点击“设置”齿轮图标打开设置。步骤 3: 通过点击“关于番茄”进入关于页面。步骤 4: 通过点击“隐私政策”查看政策内容。

These steps are generated by KG-RAG, as demonstrated in Figure 1, where the agent successfully identifies the correct navigation path. Such sequences capture multi-step relationships and guides the agent's decision-making process in real-time. By leveraging this structured data, agents can efficiently navigate deeply nested app structures and complete multi-step tasks without the need of extensive random exploration.

这些步骤由 KG-RAG 生成，如图 1 所示，代理成功识别了正确的导航路径。此类序列捕捉了多步关系并在实时中引导代理的决策过程。通过利用这些结构化数据，代理可以高效地导航深层嵌套的应用结构并完成多步任务，而无需大量随机探索。

A knowledge graph enhances a GUI agent's ability to navigate deep UI structures by capturing multi-step relationships. In KG-Android-Bench, each app's actions are linked in a logical sequence, allowing the agent to retrieve the correct next step for a given intent. This knowledge-driven approach enables effective multi-step reasoning for complex interactions that simpler benchmarks might fail to capture.

知识图谱通过捕捉多步关系增强了 GUI 代理导航深层界面结构的能力。在 KG-Android-Bench 中，每个应用的动作按逻辑序列连接，允许代理为给定意图检索正确的下一步。这种以知识为驱动的方法使得在复杂交互中进行有效的多步推理成为可能，而简单的基准可能无法捕捉这些交互。

Table 3: Per-application comparison of AutoDroid vs. KG-RAG using GPT-4 model. SR and DA are in %. AS is the average number of steps taken to complete a task.

表 3: 使用 GPT-4 模型对每个应用进行 AutoDroid 与 KG-RAG 的比较。SR 和 DA 以% 表示。AS 为完成任务所需步骤的平均数量。

App	AutoDroid SR↑	KG-RAG SR↑	AutoDroid DA↑	KG-RAG DA↑	AutoDroid AS↓	KG-RAG AS↓
Gallery	40.00	60.00	40.00	60.00	3.75	3.25
Music Player	80.00	80.00	80.00	90.00	3.50	3.25
Voice Recorder	80.00	90.00	80.00	100.00	4.13	3.88
Dialer	80.00	86.67	93.33	100.00	5.50	4.92
Contacts	73.33	93.33	80.00	93.33	4.64	4.45
Calendar	50.00	56.25	75.00	81.25	4.25	4.00
Notes	60.00	73.33	73.33	80.00	5.44	5.11
SMS Messenger	80.00	80.00	86.67	86.67	4.45	4.18
File Manager	60.00	66.67	80.00	73.33	3.44	3.22
Clock	73.33	80.00	80.00	93.33	4.18	3.36
App Launcher	80.00	90.00	90.00	90.00	7.00	6.25
Camera	46.67	53.33	60.00	66.67	3.57	3.29
Average	66.94	75.80	76.53	84.55	4.49	4.10

应用	AutoDroid SR↑	KG-RAG SR↑	AutoDroid DA↑	KG-RAG DA↑	AutoDroid AS↓	KG-RAG AS↓
图库	40.00	60.00	40.00	60.00	3.75	3.25
音乐播放器	80.00	80.00	80.00	90.00	3.50	3.25
录音机	80.00	90.00	80.00	100.00	4.13	3.88
拨号器	80.00	86.67	93.33	100.00	5.50	4.92
联系人	73.33	93.33	80.00	93.33	4.64	4.45
日历	50.00	56.25	75.00	81.25	4.25	4.00
记事本	60.00	73.33	73.33	80.00	5.44	5.11
短信	80.00	80.00	86.67	86.67	4.45	4.18
文件管理器	60.00	66.67	80.00	73.33	3.44	3.22
时钟	73.33	80.00	80.00	93.33	4.18	3.36
应用启动器	80.00	90.00	90.00	90.00	7.00	6.25
相机	46.67	53.33	60.00	66.67	3.57	3.29
平均值	66.94	75.80	76.53	84.55	4.49	4.10

Table 4: Performance comparison of AutoDroid vs. KG-RAG using Qwen2-VL-72B model.

表 4: 使用 Qwen2-VL-72B 模型的 AutoDroid 与 KG-RAG 性能对比。

App	AutoDroid SR↑	KG-RAG SR↑	AutoDroid DA↑	KG-RAG DA↑	AutoDroid AS ↓	KG-RAG AS↓
Average	62.8	70.5	71.6	80.2	8.7	7.9

应用	AutoDroid SR↑	KG-RAG SR↑	AutoDroid DA↑	KG-RAG DA↑	AutoDroid AS ↓	KG-RAG AS↓
平均	62.8	70.5	71.6	80.2	8.7	7.9

By leveraging the knowledge graph as context, a GUI agent can navigate deep or hidden UI elements through graph-based retrieval. The combination of detailed intent-action mapping and structured knowledge allows the agent to anticipate how to achieve high-level goals, such as locating a privacy policy or completing a purchase, by following a sequence of UI actions. This significantly improves the agent's ability to handle multi-step tasks and adapt to app-specific workflows.

通过将知识图作为上下文，GUI 代理可以通过基于图的检索导航到深层或隐藏的 UI 元素。详细的意图—动作映射与结构化知识相结合，使代理能够预判实现高层目标的方式，例如定位隐私政策或完成购买，方法是按序执行一系列 UI 操作。这显著提升了代理处理多步任务并适应应用特定工作流的能力。

Furthermore, KG-Android-Bench and KG-Harmony-Bench are designed to support both Android and HarmonyOS, respectively, ensuring consistent evaluation across different mobile operating systems. This cross-platform capability is especially important given the rising adoption of Har-monyOS in China, now accounting for approximately 17% of the domestic mobile OS market. By evaluating agents on the same tasks in both environments, KG-RAG ensures robust generalization across platform-specific UI differences.

此外，KG-Android-Bench 和 KG-Harmony-Bench 分别为 Android 与 HarmonyOS 设计，确保在不同移动操作系统上的评估一致。鉴于 HarmonyOS 在中国采纳率上升、目前约占国内移动操作系统市场的 17%，这种跨平台能力尤为重要。通过在两种环境中对相同任务进行评估，KG-RAG 确保了对平台特定 UI 差异的稳健泛化。

In summary, we set new benchmarks called KG-Android-Bench and KG-Harmony-Bench for evaluating autonomous mobile agents by integrating structured knowledge graphs and detailed intent-action mappings. Its cross-platform support makes it an essential tool for assessing agents in diverse, real-world app environments, driving further advancements in the field. Compared with prior Android datasets that mainly provide UI traces, our KG-Android-Bench and KG-Harmony-Bench couple intent-action mappings with UTG-derived graph memory, enabling evaluation of multi-step, deeply nested goals common in Chinese mobile apps (e.g., finance, e-commerce, travel). The paired Android/HarmonyOS suites also stress cross-platform robustness under heterogeneous UI paradigms- a setting underrepresented in existing resources-and therefore serve as a testbed for graph-augmented agents rather than a near-duplicate of earlier benchmarks.

总之，我们提出了名为 KG-Android-Bench 与 KG-Harmony-Bench 的新基准，通过整合结构化知识图与详细的意图—动作映射来评估自主移动代理。其跨平台支持使其成为评估代理在多样化、真实应用环境中的重要工具，推动该领域进一步发展。与主要提供 UI 跟踪的以往 Android 数据集相比，我们的 KG-Android-Bench 与 KG-Harmony-Bench 将意图—动作映射与基于 UTG 的图记忆相结合，能够评估在中国移动应用中常见的多步、深层嵌套目标 (如金融、电子商务、出行)。配套的 Android/HarmonyOS 套件也强调在异构 UI 范式下的跨平台鲁棒性——这是现有资源中代表性不足的场景——因此它们更适合作为面向图增强代理的测试床，而非早期基准的近似重复。

5 Experiments

5 实验

5.1 Experimental Setup

5.1 实验设置

Evaluation Benchmarks. We conduct comprehensive evaluations on the following three benchmarks: (1) DroidTask (EN) (Wen et al., 2024): 162 tasks across 12 English Android apps (e.g., Gallery, Camera, and File Manager.) (2) KG-Android-Bench (CN) (Ours): A new benchmark of over 200 real-world tasks spanning 30 Chinese apps. These tasks cover diverse domains such as music, social media, navigation, e-commerce, and more (in Chinese UI environments). (3) KG-Harmony-Bench (CN) (Ours): A set of 150 cross-platform tasks on HarmonyOS devices, covering 15 app categories (e.g., mapping apps like Amap, travel apps like Ctrip). This benchmark evaluates an agent's ability to generalize across platforms.

评估基准。我们在以下三个基准上进行全面评估: (1) DroidTask (EN) (Wen et al., 2024): 涵盖 12 个英文 Android 应用 (如图库、相机、文件管理器等) 的 162 个任务。(2) KG-Android-Bench (CN) (本工作): 包含 30 个中文应用、超过 200 个真实任务的新基准，覆盖音乐、社交、导航、电子商务等多领域 (中文 UI 环境)。(3) KG-Harmony-Bench (CN) (本工作): 在 HarmonyOS 设备上的 150 个跨平台任务，涵盖 15 类应用 (如高德这类地图、携程这类出行应用)。该基准评估代理跨平台泛化能力。

To evaluate agent performance, we use three key metrics: Success Rate (SR), which measures the percentage of user instructions successfully completed; Decision Accuracy (DA), reflecting the correctness of the agent's action decisions; and Average Steps (AS), indicating task efficiency through the average number of steps taken to finish a task.

为评估代理性能，我们使用三项关键指标: 成功率 (SR)，衡量用户指令成功完成的比例；决策准确率 (DA)，反映代理动作决策的正确性；以及平均步骤数 (AS)，通过完成任务所需平均步骤数来表示任务效率。

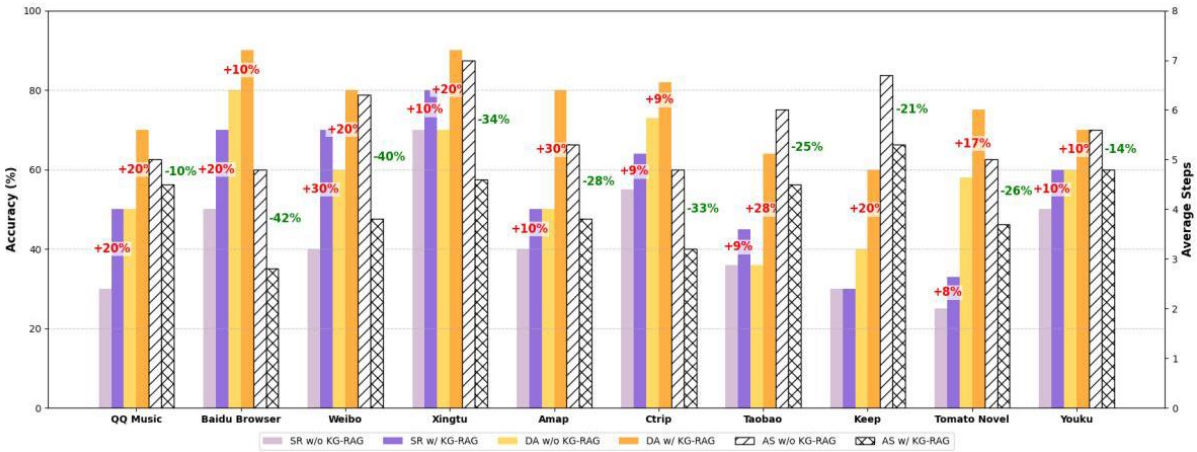

Figure 3: Performance comparison across 10 representative Chinese mobile applications, showing improvements in Success Rate (SR) and Decision Accuracy (DA) with KG-RAG, while reducing task average steps (AS) (indicated by striped bars). Red numbers reflect accuracy gains, and green numbers indicate step reductions.

图 3: 在 10 个具有代表性的中文移动应用上的性能比较，显示 KG-RAG 提升了成功率 (SR) 和决策准确率 (DA)，同时减少了任务平均步骤数 (AS)(以条纹柱表示)。红色数字表示准确率提升，绿色数字表示步骤减少。

Table 5: Performance comparison of MobileAgent-v2 and UI-TARS with and without the integration of KG-RAG on the ”QQ Music” application.

表 5: 在“QQ 音乐”应用上，比较 MobileAgent-v2 与 UI-TARS 在有无集成 KG-RAG 情况下的性能。

Method	Perception Model	Decision Model	SR (%)↑	DA (%)↑	AS↓
MobileAgent-v2	GroundingDINO	Qwen2-VL	20.0	40.0	5.5
MobileAgent-v2+KG-RAG	GroundingDINO	Qwen2-VL	30.0	60.0	5.0
MobileAgent-v2	VUT	Qwen2-VL	30.0	50.0	5.0
MobileAgent-v2+KG-RAG	VUT	Qwen2-VL	50.0	70.0	4.5
MobileAgent-v2	GroundingDINO	GPT-40	30.0	50.0	5.3
MobileAgent-v2+KG-RAG	GroundingDINO	GPT-40	40.0	60.0	5.0
MobileAgent-v2	VUT	GPT-40	50.0	60.0	4.8
MobileAgent-v2+KG-RAG	VUT	GPT-40	60.0	70.0	3.2
UI-TARS	UI-TARS-7B-SFT	UI-TARS-7B-SFT	90.0	90.0	5.9
UI-TARS+KG-RAG	UI-TARS-7B-SFT	UI-TARS-7B-SFT	90.0	100.0	5.2

方法	感知模型	决策模型	成功率 (SR)↑	到达率 (DA)↑	平均步骤数 (AS)↓
MobileAgent-v2	GroundingDINO	Qwen2-VL	20.0	40.0	5.5
MobileAgent-v2+KG-RAG	GroundingDINO	Qwen2-VL	30.0	60.0	5.0
MobileAgent-v2	VUT	Qwen2-VL	30.0	50.0	5.0
MobileAgent-v2+KG-RAG	VUT	Qwen2-VL	50.0	70.0	4.5
MobileAgent-v2	GroundingDINO	GPT-40	30.0	50.0	5.3
MobileAgent-v2+KG-RAG	GroundingDINO	GPT-40	40.0	60.0	5.0
MobileAgent-v2	VUT	GPT-40	50.0	60.0	4.8
MobileAgent-v2+KG-RAG	VUT	GPT-40	60.0	70.0	3.2
UI-TARS	UI-TARS-7B-SFT	UI-TARS-7B-SFT	90.0	90.0	5.9
UI-TARS+KG-RAG	UI-TARS-7B-SFT	UI-TARS-7B-SFT	90.0	100.0	5.2

5.2 Comparison with AutoDroid

5.2 与 AutoDroid 的比较

We first compare our KG-RAG approach with similar UTG-based AutoDroid (Wen et al., 2024) on the DroidTask benchmark using two LLM back-ends: Qwen2-VL (Wang et al., 2024c) and GPT- 4 (Achiam et al., 2023).

我们首先在 DroidTask 基准上使用两种 LLM 后端:Qwen2-VL (Wang et al., 2024c) 和 GPT-4 (Achiam et al., 2023)，将我们的 KG-RAG 方法与基于 UTG 的 AutoDroid (Wen et al., 2024) 进行比较。

First, Table 3 provides a detailed per-application performance breakdown for AutoDroid and KG-RAG on the DroidTask benchmark (using a GPT- 4 backend). We report Success Rate (SR), Decision Accuracy (DA), and Average Steps (AS) for each app. KG-RAG achieves consistent improvements on almost all individual apps, underlining the robustness of our approach. For example, on the Gallery app, KG-RAG raises the success rate from 40.0% to 60.0% and the decision accuracy from 40.0% to 60.0%, while also reducing the average steps

from 3.75 to 3.25. Similar gains are observed in most cases (e.g., Voice Recorder SR 80% → 90% , Dialer DA 93. 3% → 100%), indicating that our method's advantages hold at the per-app level, not just in aggregate results.

首先，表 3 给出在 DroidTask 基准上 (使用 GPT-4 后端)AutoDroid 与 KG-RAG 的按应用性能细分。我们对每个应用报告成功率 (SR)、决策准确率 (DA) 和平均步骤数 (AS)。KG-RAG 在几乎所有单个应用上都取得了持续提升，突显了我们方法的鲁棒性。例如，在 Gallery 应用上，KG-RAG 将成功率从 40.0% 提升到 60.0%，将决策准确率从 40.0% 提升到 60.0%，同时将平均步骤数从 3.75 降至 3.25。大多数情况下也观察到类似的提升 (例如，Voice Recorder SR 80% → 90% , Dialer DA 93. 3% → 100%)，表明我们方法的优势不仅体现在总体结果上，也在单应用层面成立。

Additionally, Tables 3 and 4 summarize the overall performance of AutoDroid vs. KG-RAG across key metrics and across two different LLM backends (GPT-4 and Qwen2-VL). KG-RAG consistently outperforms AutoDroid on every metric under both LLM settings. In particular, KG-RAG yields higher average success rates and decision accuracy, while requiring fewer steps on average. These results demonstrate that our knowledge-augmented approach provides clear benefits over the prior UTG-based method in both accuracy and efficiency.

此外，表 3 和表 4 汇总了在两种不同 LLM 后端 (GPT-4 和 Qwen2-VL) 下 AutoDroid 与 KG-RAG 在关键指标上的总体表现。KG-RAG 在两种 LLM 设置下的每个指标上均持续优于 AutoDroid。特别是，KG-RAG 在平均成功率和决策准确率上更高，同时平均所需步骤更少。这些结果证明，我们的知识增强方法在准确性和效率方面均比先前的基于 UTG 的方法具有明显优势。

5.3 Evaluating KG-RAG as a Plug-and-Play Module in GUI Agents

5.3 将 KG-RAG 作为 GUI 代理的即插即用模块评估

To demonstrate the versatility of our approach, we integrate KG-RAG as a plug-and-play module into two state-of-the-art GUI agent frameworks: MobileAgent-v2 (Wang et al., 2025) and UI-TARS (Qin et al., 2025) and we choose different perception models (VUT (Li et al., 2021) and GroundingDINO (Liu et al., 2024)) and decision models (GPT4-o (Hurst et al., 2024) and Qwen2-VL (Wang et al., 2024c)). This integration is seamless, requiring no architectural changes, highlighting that KG-RAG can serve as a general enhancement layer for different agents.

为展示我们方法的通用性，我们将 KG-RAG 作为即插即用模块集成到两种最先进的 GUI 代理框架:MobileAgent-v2 (Wang et al., 2025) 和 UI-TARS (Qin et al., 2025)，并选择了不同的感知模型 (VUT (Li et al., 2021) 和 GroundingDINO (Liu et al., 2024)) 与决策模型 (GPT4-o (Hurst et al., 2024) 和 Qwen2-VL (Wang et al., 2024c))。该集成无缝进行，不需要架构更改，凸显了 KG-RAG 可作为不同代理的一般增强层。

Table 6: Performance comparison of UI-TARS and MobileAgent-v2 using GPT-40 on HarmonyOS.

表 6: 在 HarmonyOS 上使用 GPT-40 的 UI-TARS 与 MobileAgent-v2 性能比较。

Method	SR (%)↑	DA (%)↑	AS ↓
MobileAgent-v2	41.9	64.2	4.6
MobileAgent-v2 + KG-RAG	55.4	77.4	4.2
UI-TARS	61.4	71.0	4.4
UI-TARS + KG-RAG	70.9	85.5	4.0

方法	成功率 (%)↑	域适应 (%)↑	AS ↓
MobileAgent-v2	41.9	64.2	4.6
MobileAgent-v2 + KG-RAG	55.4	77.4	4.2
UI-TARS	61.4	71.0	4.4
UI-TARS + KG-RAG	70.9	85.5	4.0

Table 5 summarizes the performance of MobileAgent-v2 and UI-TARS with and without KG-RAG. For MobileAgent-v2, KG-RAG boosts the success rate (SR) and the decision accuracy (DA), while also reducing the number of steps needed for task completion. Similarly, UI-TARS benefits from the KG-RAG integration, showing notable improvements in both SR and DA.

表 5 总结了 MobileAgent-v2 和 UI-TARS 在有无 KG-RAG 情况下的性能。对 MobileAgent-v2，KG-RAG 提高了成功率 (SR) 和决策准确率 (DA)，同时减少了完成任务所需的步骤。同样，UI-TARS 在集成 KG-RAG 后也受益，SR 和 DA 均显著提升。

Figure 3 illustrates these improvements, showing that agents equipped with KG-RAG not only achieve higher success and accuracy, but also require fewer interactions to complete tasks compared to their base versions. The consistent performance boost across both MobileAgent-v2 and UI-TARS emphasizes KG-RAG's effectiveness as a drop-in module. By providing relevant contextual knowledge on the fly, KG-RAG helps guide the agent's decisions, leading to more successful outcomes and streamlined action sequences.

图 3 展示了这些改进，表明配备 KG-RAG 的智能体不仅在成功率和准确率上更高，而且完成任务所需的交互更少，相较于其基础版本更为高效。KG-RAG 作为即插即用模块在 MobileAgent-v2 和 UI-TARS 上均带来了稳定的性能提升。通过即时提供相关的上下文知识，KG-RAG 引导智能体决策，从而带来更高的成功率和更简洁的行动序列。

5.4 Evaluation on HarmonyOS

5.4 在 HarmonyOS 上的评估

Table 6 reports similar success rates and decision accuracy for KG-RAG on HarmonyOS as the results on Android in Table 5, showing consistent performance across both platforms. This highlights KG-RAG's ability to generalize across different UI paradigms without requiring platform-specific modifications, making it a versatile solution for mobile app interactions.

表 6 报告了 KG-RAG 在 HarmonyOS 上的成功率和决策准确率与表 5 中 Android 结果相似，显示出跨两个平台的一致性能。这突显了 KG-RAG 无需特定平台修改即可在不同 UI 范式间泛化的能力，使其成为移动应用交互的通用解决方案。

5.5 Ablation Study of RAG Construction

5.5 关于 RAG 构建的消融研究

To evaluate the impact of different knowledge construction components, we conduct an ablation study varying the choice of VLM used in intent generation module in Figure 2(b) and LLM used in the LLM search module in Figure 2(c) to construct the KG-RAG Knowledge Database.

为评估不同知识构建组件的影响，我们在图 2(b) 中改变用于意图生成模块的 VLM 选择，以及在图 2(c) 中改变用于 LLM 检索模块的 LLM 选择，以构建 KG-RAG 知识库并进行消融研究。

Table 7: Effect of VLM and LLM choices on KG-RAG database construction performance on KG-Android-Bench.

表 7:VLM 和 LLM 选择对 KG-Android-Bench 上 KG-RAG 数据库构建性能的影响。

VLM + LLM Combination	SR (%)?	DA (%)?	AS?
InternVL2-76B+Qwen2-72B	67.00	70.70	5.00
InternVL2-76B+Deepseek-14B	70.96	74.66	5.01
Qwen2-VL-72B+Qwen2-72B	72.59	78.07	5.16
Qwen2-VL-72B+Deepseek-14B	78.33	82.03	4.92

VLM + LLM 组合	成功率 (SR)?	判定准确率 (DA)?	平均步数 (AS)?
InternVL2-76B+Qwen2-72B	67.00	70.70	5.00
InternVL2-76B+Deepseek-14B	70.96	74.66	5.01
Qwen2-VL-72B+Qwen2-72B	72.59	78.07	5.16
Qwen2-VL-72B+Deepseek-14B	78.33	82.03	4.92

Table 7 compares four configurations of VLM and LLM used to construct the KG-RAG knowledge database on KG-Android-Bench (using UI-TARS as the agent). For the VLM+LLM configurations, we experiment with InternVL2 (Chen et al., 2024), Qwen2-VL (Wang et al., 2024b), and DeepSeek-14B (DeepSeek-AI, 2025). The results show that leveraging a stronger vision-language model (VLM) and a specialized reasoning LLM yields the best performance. In particular, Qwen2- VL-72B + DeepSeek-14B achieves the highest success rate (SR 78.33%) and decision accuracy (DA 82.03%), along with the lowest average steps (AS 4.92), outperforming all other combinations. Overall, the Qwen2-VL + DeepSeek-14B pairing produces the largest gains in both success and accuracy, and also finds shorter navigation paths (lowest AS), confirming that richer visual semantics combined with focused reasoning yields the most effective knowledge graph construction for KG-RAG.

表 7 比较了用于在 KG-Android-Bench 上构建 KG-RAG 知识库的四种 VLM 与 LLM 配置 (使用 UI-TARS 作为代理)。在 VLM+LLM 配置中, 我们尝试了 InternVL2 (Chen et al., 2024)、Qwen2-VL (Wang et al., 2024b) 与 DeepSeek-14B (DeepSeek-AI, 2025)。结果表明，结合更强的视觉-语言模型 (VLM) 与专门的推理型 LLM 可获得最佳性能。尤其是 Qwen2-VL-72B + DeepSeek-14B 达到最高成功率 (SR 78.33%) 和决策准确率 (DA 82.03%)，且平均步数最少 (AS 4.92)，优于其他组合。总体上，Qwen2-VL + DeepSeek-14B 在成功率与准确率上带来最大提升，并能找到更短的导航路径 (最低 AS)，证实更丰富的视觉语义与聚焦推理相结合能为 KG-RAG 的知识图谱构建带来最有效的结果。

We further investigate different text embedding models for retrieval, using doubao-embedding-text (Seed et al., 2025), gte-multilingual-base (Zhang et al., 2024), and multilingual-e5- large-instruct (Wang et al., 2024a). Table 8 examines the impact of different text embedding models on retrieval performance. Here, doubao-embedding-text (Seed et al., 2025) emerges as the top performer, achieving the highest SR (73.90%) and a joint-highest DA (80.00%), while also requiring the fewest steps (AS 4.73). Based on these results, we select doubao-embedding-text for the final KG-RAG system due to its stronger retrieval alignment and lower step count, which together contribute to higher overall efficiency.

我们进一步考察了用于检索的不同文本嵌入模型，使用 doubao-embedding-text (Seed et al., 2025)、gte-multilingual-base (Zhang et al., 2024) 与 multilingual-e5-large-instruct (Wang et al., 2024a)。表 8 检验了不同文本嵌入模型对检索性能的影响。结果显示 doubao-embedding-text (Seed et al., 2025) 表现最好，取得最高 SR(73.90%) 和并列最高的 DA(80.00%)，同时所需步数最少 (AS 4.73)。基于这些结果，我们为最终的 KG-RAG 系统选择 doubao-embedding-text，因其检索对齐更强且步数更少，从而提升整体效率。

5.6 Generalization Across GUIs and UTG Cost Trade-offs

5.6 跨 GUI 的泛化与 UTG 成本权衡

We further report results that (1) transfer KG-RAG to non-mobile GUIs without retraining, (2) test cross-device/OS robustness, and (3) quantify the UTG construction cost-quality trade-off. First, we transfer KG-RAG to non-mobile GUIs without any retraining; Table 9 shows large gains on Weibo (web) and QQ Music (desktop). Second, we evaluate cross-device/OS robustness on Weibo; Table 10 demonstrates consistent improvements from low-end to flagship devices as well as on HarmonyOS. Finally, we quantify the UTG construction cost-quality trade-off; Table 11 indicates accuracy saturates at about 4 hours per complex app, enabling practical deployment with bounded offline cost.

我们还报告了以下结果:(1) 在不重训练的情况下将 KG-RAG 转移到非移动端 GUI，(2) 测试跨设备/操作系统的鲁棒性，以及 (3) 量化 UTG 构建的成本-质量权衡。首先，我们在不重训练的情况下将 KG-RAG 转移到非移动端 GUI；表 9 显示在微博 (网页) 和 QQ 音乐 (桌面) 上有大幅提升。其次，我们在微博上评估跨设备/操作系统的鲁棒性；表 10 展示了从低端到旗舰设备以及在 HarmonyOS 上的一致改进。最后，我们量化了 UTG 构建的成本-质量权衡；表 11 表明准确率在每个复杂应用约 4 小时时趋于饱和，从而支持可控离线成本下的实际部署。

Table 8: Comparison of retrieval performance using different text embedding models on KG-Android-Bench.

表 8: 在 KG-Android-Bench 上使用不同文本嵌入模型的检索性能比较。

Embedding Model	SR (%)↑	DA (%)↑	AS↓
multilingual-e5-large-instruct	70.60	76.60	5.08
gte-multilingual-base	70.60	80.00	5.19
doubao-embedding-text	73.90	80.00	4.73

嵌入模型	成功率 (%)?	DA (%)?	平均步数 ?
multilingual-e5-large-instruct	70.60	76.60	5.08
gte-multilingual-base	70.60	80.00	5.19
doubao-embedding-text	73.90	80.00	4.73

Table 9: Non-mobile GUIs via training-free transfer of a mobile-built KG-RAG database.

表 9: 通过对用移动端构建的 KG-RAG 数据库进行免训练迁移得到的非移动 GUI。

Domain	Method	SR(%)?	DA(%)?	AS?
Weibo (web)	UI-TARS-web	50.0	70.0	7.6
	+ KG-RAG	90.0	100.0	5.2
QQ Music (desktop)	UI-TARS-desktop	60.0	60.0	5.9
	+ KG-RAG	80.0	90.0	4.3

领域	方法	SR(%)?	DA(%)?	AS?
微博 (网页版)	UI-TARS-网页	50.0	70.0	7.6
	+ KG-RAG	90.0	100.0	5.2
QQ 音乐 (桌面版)	UI-TARS-桌面	60.0	60.0	5.9
	+ KG-RAG	80.0	90.0	4.3

As seen in Table 9, transferring a mobile-built KG-RAG to web/desktop yields +40% SR on Weibo-web and +20% SR on QQ Music-desktop, without any platform-specific retraining. Table 10 further shows that the gains hold across heterogeneous chipsets and on HarmonyOS, while steps (AS) are consistently reduced. Finally, Table 11 reveals that performance saturates at about 4 hours of UTG extraction, indicating a practical operating point for large-scale deployments.

如表 9 所示，将在移动端构建的 KG-RAG 迁移到网页/桌面端，Weibo-web 的成功率提高了 +40%，QQ Music-desktop 提高了 +20%，且无需任何针对平台的重新训练。表 10 进一步表明，这些收益在不同芯片组和 HarmonyOS 上仍然成立，同时步骤数 (AS) 持续减少。最后，表 11 显示性能在约 4 小时 UTG 提取时趋于饱和，提示大规模部署的一个实际运行点。

6 Conclusion

6 结论

The paper presents KG-RAG, a framework that enhances GUI agents by leveraging structured knowledge from UTGs. KG-RAG addresses key limitations of existing agents, such as their inability to fully utilize app-specific knowledge from incomplete UTGs. Our intent-guided offline pathfinding algorithm transforms UTGs into structured vector embeddings, creating a robust knowledge database that improves real-time decision-making with precomputed navigation paths for specific user intents.

本文提出了 KG-RAG，一种通过利用来自 UTG 的结构化知识来增强 GUI 代理的框架。KG-RAG 解决了现有代理的关键局限，例如无法充分利用来自不完整 UTG 的应用特定知识。我们的意图引导离线路径搜索算法将 UTG 转换为结构化向量嵌入，构建了一个健壮的知识库，通过为特定用户意图预计算导航路径改善实时决策。

We also introduce KG-Android-Bench, a comprehensive benchmark for GUI agents, supporting cross-platform (Android, HarmonyOS) with detailed intent-action mappings in knowledge graphs. Experimental results show that KG-RAG significantly improves agent efficiency, reducing task completion steps and increasing success rates. Our findings highlight the effectiveness of knowledge-driven retrieval augmentation in overcoming practical challenges for mobile GUI agents, setting new performance benchmarks.

我们还引入了 KG-Android-Bench，一个面向 GUI 代理的综合基准，支持跨平台 (Android、HarmonyOS)，在知识图中提供详细的意图-动作映射。实验结果表明，KG-RAG 显著提高了代理效率，减少了完成任务所需步骤并提升了成功率。我们的研究强调了知识驱动检索增强在克服移动 GUI 代理实际挑战方面的有效性，树立了新的性能基准。

Table 10: Cross-device/OS performance on Weibo.

表 10:Weibo 上的跨设备/操作系统性能。

OS	Device (Chip)	SR(%)↑	DA(%)↑	AS↓
Baseline (no KG-RAG)	HUAWEI P40 (Kirin 990)	40.0	60.0	6.25
Android	HUAWEI Y9s (Kirin 710F)	60.0	70.0	5.75
Android	OPPO K9s (Snapdragon 778G)	60.0	80.0	5.00
Android	Vivo iQOO 8 (Snapdragon 888)	70.0	70.0	4.50
Android	HUAWEI P40 (Kirin 990)	70.0	80.0	3.75
HarmonyOS	HUAWEI Mate 60 (Kirin 9000S)	80.0	80.0	3.25

操作系统	设备 (芯片)	成功率 (%)↑	DA(%)↑	AS↓
基线 (无 KG-RAG)	华为 P40(麒麟 990)	40.0	60.0	6.25
安卓	华为 Y9s(麒麟 710F)	60.0	70.0	5.75
安卓	OPPO K9s(骁龙 778G)	60.0	80.0	5.00
安卓	Vivo iQOO 8(骁龙 888)	70.0	70.0	4.50
安卓	华为 P40(麒麟 990)	70.0	80.0	3.75
HarmonyOS	华为 Mate 60(麒麟 9000S)	80.0	80.0	3.25

Table 11: UTG extraction time vs. performance on Weibo. Accuracy saturates at ~ 4 h .

表 11:UTG 提取时间与微博上性能的关系。准确率在 ~ 4 h 时趋于饱和。

Extraction Time	SR(%)↑	DA(%)↑	AS↓	ΔSR over Baseline
Baseline (w/o RAG)	40.0	60.0	6.25	-
1 h	50.0	60.0	5.75	+25%
2 h	60.0	70.0	5.25	+50%
4 h	70.0	80.0	4.50	+75%
8 h	70.0	80.0	3.75	+75%

抽取时间	SR(%)?	DA(%)?	AS?	较基线的 ΔSR
基线 (无 RAG)	40.0	60.0	6.25	-
1 h	50.0	60.0	5.75	+25%
2 h	60.0	70.0	5.25	+50%
4 h	70.0	80.0	4.50	+75%
8 h	70.0	80.0	3.75	+75%

Limitations

限制

KG-RAG relies on app UI Transition Graphs (UTG) extracted by an automatic app testing system. Currently, it costs one or several hours to fully construct the UTG due to the ambitious goal of obtaining a high page coverage inside the app. In the future, it is worth exploring a more advanced automatic app testing system to mitigate resource cost. Moreover, it is interesting to adopt this KG-RAG framework in the domain of web and PC.

KG-RAG 依赖由自动化应用测试系统提取的应用界面转换图 (UTG)。目前，为了在应用内获得较高的页面覆盖率，完整构建 UTG 需要一到数小时。未来值得探索更先进的自动化应用测试系统以降低资源开销。此外，将该 KG-RAG 框架应用于网页和 PC 领域也很有意义。

Furthermore, KG-RAG designs a knowledge database for each app. Future work could explore the design of a vertical domain (e.g., shopping).

此外，KG-RAG 为每个应用设计单独的知识库。未来工作可探索面向垂直领域 (例如购物) 的设计。

Ethics Discussion

伦理讨论

In constructing the knowledge graph and developing the KG-RAG system, we took careful steps to protect user privacy and data confidentiality: (1) Anonymous Data Collection: All UTGs are fully anonymized and contain only abstract UI structures and navigation paths, without any PII or user content; (2) Automatic Privacy Masking: Potentially sensitive on-screen fields are automatically masked during data capture; (3) No Authentication Required: We evaluate only logged-out scenarios and never use accounts, passwords, or authenticated actions; (4) Data Security: Data are stored securely for research purposes only, and no information that can identify individuals is released.

在构建知识图谱和开发 KG-RAG 系统过程中，我们采取了谨慎措施以保护用户隐私和数据机密性: (1) 数据匿名化: 所有 UTG 均已完全匿名，仅包含抽象的界面结构和导航路径，不含任何 PII 或用户内容；(2) 自动隐私掩码: 在数据采集时会自动掩盖潜在敏感的屏幕字段；(3) 无需认证: 我们仅评估未登录场景，且从不使用账号、密码或需认证的操作；(4) 数据安全: 数据仅为研究目的安全存储，不会发布可识别个人的信息。

References

参考文献

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. arXiv preprint arXiv:2303.08774.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. arXiv preprint arXiv:2303.08774.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. 2024. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 24185-24198.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. 2024. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 24185-24198.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. Preprint, arXiv:2501.12948.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. Preprint, arXiv:2501.12948.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. 2024. Cogagent: A visual language model for gui agents. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 14281-14290.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. 2024. Cogagent: A visual language model for gui agents. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 14281-14290.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Os-trow, Akila Welhinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-40 system card. arXiv preprint arXiv:2410.21276.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welhinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-40 system card. arXiv preprint arXiv:2410.21276.

Sunjae Lee, Junyoung Choi, Jungjae Lee, Munim Hasan Wasi, Hojun Choi, Steven Y Ko, Sangeun Oh, and Insik Shin. 2023. Explore, select, derive, and recall: Augmenting llm with human-like memory for mobile task automation. arXiv preprint arXiv:2312.03003.

Sunjae Lee, Junyoung Choi, Jungjae Lee, Munim Hasan Wasi, Hojun Choi, Steven Y Ko, Sangeun Oh, and Insik Shin. 2023. Explore, select, derive, and recall: Augmenting llm with human-like memory for mobile task automation. arXiv preprint arXiv:2312.03003.

Yang Li, Gang Li, Xin Zhou, Mostafa Dehghani, and Alexey Gritsenko. 2021. Vut: Versatile ui transformer for multi-modal multi-task user interface modeling. arXiv preprint arXiv:2112.05692.

Yang Li, Gang Li, Xin Zhou, Mostafa Dehghani, and Alexey Gritsenko. 2021. Vut: Versatile ui transformer for multi-modal multi-task user interface modeling. arXiv preprint arXiv:2112.05692.

Yuanchun Li, Ziyue Yang, Yao Guo, and Xiangqun Chen. 2017. Droidbot: a lightweight ui-guided test input generator for android. In 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C), pages 23-26.

Yuanchun Li, Ziyue Yang, Yao Guo, and Xiangqun Chen. 2017. Droidbot: a lightweight ui-guided test input generator for android. In 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C), pages 23-26.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. 2024. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In European Conference on Computer Vision, pages 38-55. Springer.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. 2024. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In European Conference on Computer Vision, pages 38-55. Springer.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. 2025. UI-TARS: Pioneering automated gui interaction with native agents. arXiv preprint arXiv:2501.12326.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. 2025. UI-TARS: Pioneering automated gui interaction with native agents. arXiv preprint arXiv:2501.12326.

ByteDance Seed, Yufeng Yuan, Yu Yue, Mingxuan Wang, Xiaochen Zuo, Jiaze Chen, Lin Yan, Wenyuan Xu, Chi Zhang, Xin Liu, et al. 2025. Seed-thinking-v1. 5: Advancing superb reasoning models with reinforcement learning. arXiv preprint arXiv:2504.13914.

ByteDance Seed、袁宇峰、岳宇、王明轩、左晓臣、陈家泽、闫琳、许文远、张驰、刘鑫等. 2025. Seed-thinking-v1.5: 通过强化学习推进卓越推理模型. arXiv 预印本 arXiv:2504.13914.

Junyang Wang, Haiyang Xu, Haitao Jia, Xi Zhang, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. 2025. Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration. Advances in Neural Information Processing Systems, 37:2686-2710.

王俊阳、徐海洋、贾海涛、张曦、闫明、沈伟舟、张吉、黄飞、桑继涛. 2025. Mobile-agent-v2: 通过多智能体协作实现高效导航的移动设备操作助手. Advances in Neural Information Processing Systems, 37:2686-2710.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024a. Multilingual e5 text embeddings: A technical report. arXiv preprint arXiv:2402.05672.

王亮、杨楠、黄晓龙、杨林钧、Rangan Majumder、魏福如. 2024a. 多语种 e5 文本嵌入: 技术报告. arXiv 预印本 arXiv:2402.05672.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhi-hao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024b. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. arXiv preprint arXiv:2409.12191.

王鹏、白帅、谭思南、王世杰、范志豪、白金泽、陈克勤、刘雪晶、王佳麟、葛文彬、范洋、党凯、杜梦菲、任轩成、门睿、刘达恒、周畅、周靖人、林军阳. 2024b. Qwen2-vl: 提升视觉-语言模型在任意分辨率下对世界的感知. arXiv 预印本 arXiv:2409.12191.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhi-hao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. 2024c. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. arXiv preprint arXiv:2409.12191.

王鹏、白帅、谭思南、王世杰、范志豪、白金泽、陈克勤、刘雪晶、王佳麟、葛文彬等. 2024c. Qwen2-vl: 提升视觉-语言模型在任意分辨率下对世界的感知. arXiv 预印本 arXiv:2409.12191.

Hao Wen, Yuanchun Li, Guohong Liu, Shanhui Zhao, Tao Yu, Toby Jia-Jun Li, Shiqi Jiang, Yunhao Liu, Yaqin Zhang, and Yunxin Liu. 2024. Autodroid: Llm-powered task automation in android. In Proceedings of the 30th Annual International Conference on Mobile Computing and Networking, pages 543-557.

温昊、李远春、刘国宏、赵善辉、余涛、李家骏、蒋世祺、刘云浩、张雅琴、刘运鑫. 2024. Autodroid: 基于大模型的 Android 任务自动化. 载于第 30 届国际移动计算与网络会议论文集，页 543-557.

Juyeon Yoon, Robert Feldt, and Shin Yoo. 2023. Autonomous large language model agents enabling intent-driven mobile gui testing. arXiv preprint arXiv:2311.08649.

尹周延、Robert Feldt、Shin Yoo. 2023. 自主大语言模型代理实现意图驱动的移动 GUI 测试. arXiv 预印本 arXiv:2311.08649.

Keen You, Haotian Zhang, Eldon Schoop, Floris Weers, Amanda Swearngin, Jeffrey Nichols, Yinfei Yang, and Zhe Gan. 2024. Ferret-ui: Grounded mobile ui understanding with multimodal llms. In European Conference on Computer Vision, pages 240- 255. Springer.

尤勤、张浩天、Eldon Schoop、Floris Weers、Amanda Swearngin、Jeffrey Nichols、杨吟飞、甘喆. 2024. Ferret-ui: 基于多模态大模型的有根移动界面理解. 载于欧洲计算机视觉大会，页 240-255. Springer.

Chi Zhang, Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. 2023. Appagent: Multimodal agents as smartphone users. arXiv preprint arXiv:2312.13771.

张驰、杨钊、刘家宣、韩昱成、陈鑫、黄泽彪、付斌、于刚. 2023. Appagent: 作为智能手机用户的多模态代理. arXiv 预印本 arXiv:2312.13771.

Xin Zhang, Yanzhao Zhang, Dingkun Long, Wen Xie, Ziqi Dai, Jialong Tang, Huan Lin, Baosong Yang, Pengjun Xie, Fei Huang, et al. 2024. mgte: Generalized long-context text representation and reranking models for multilingual text retrieval. In Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 1393-1412.

张昕、张艳钊、龙定坤、谢文、戴子奇、唐嘉龙、林欢、杨保松、谢鹏军、黄飞等. 2024. mgte: 用于多语种文本检索的广义长上下文文本表示与重排序模型. 载于 2024 年实证方法自然语言处理大会: 产业专题论文集，页 1393-1412.