GUI-R1 A Generalist R1-Style Vision-Language Action Model For GUI Agents

Yang

2025 年 6 月 30 日

1 Introduction

当下使用 LVLM 发展 GUI Agent，特点是只依赖屏幕分析作为信息源进行决策，不依赖环境的文本描述，在决策上具有更高的灵活性。屏幕分析指将屏幕作为输入让模型进行处理，屏幕上的图标等进行解析。而文本描述相当于将信息以文本结构的形式输入到模型中。显然后者的泛化能力更差，因为一旦屏幕上的图标等发生了变化，后者必须重新输入，重新为图标设置标签。而前者仍然可以使用视觉匹配完成任务。可以比喻为屏幕分析更像是学会了方法，而文本描述更像是记住了答案，方法可能会出错但是可以迁移到别的题目，答案在特定题目上完全正确但是不能迁移到别的地方。

不足：传统使用 supervised fine-tuning(SFT)，需要大量高质数据且泛化能力还需要提高

新方法：rule-based reinforcement fine-tuning(RFT)，只需要少量数据 (相对) 就可以训练出不错的效果且泛化能力很强

2 GUI-R1 Framework

输入：模型需要完成的高层任务 Q，当前图像 I，执行历史记录 H

输出：候选响应的集合 $O = \{o_1, \ldots, o_N\}$，其中每个响应包括 $o_{act}, o_{text}, o_{point}$，分别对应 low level action，输入的文本，需要点击的坐标

对于每个响应，用 unified action space reward funciton 进行评分得到分数 $R = \{r_1, \ldots, r_N\}$，然后计算相对优势：$A_i = \frac{r_i - mean\{r_1,\ldots,r_N\}}{std\{r_1,\ldots,r_N\}}$，其中 mean 是均值，std 是标准差

unified action space：统一动作空间指不同平台 (Mobile，Web) 可以执行的操作可能不同，但是对它们进行分解成为原子操作，这些原子操作是相同的，从而解决多平台训练的动作空间冲突问题

Format reward：评估模型的输出，使得模型的输出更接近预期的语义和语法格式，本论文中认为输出分为两个板块：<think> 和 <answer>，则希望输出的范式是：<think> 中包含想法，而 <answer> 中包含可能的执行动作 [click, select, enter] 以及执行任务需要的 input text，input point

Accuracy reward：$R_{acc} = R_{act} + R_{point} + R_{text}$，其中前两者的得分标准是完全一致得 1 分，否则 0 分，后者计算语义分数 (语义相似度)，若超过 0.5 得 1，否则为 0

Response reward：$R_{response} = \alpha R_{acc} + \beta R_{format}$

Data collection and fitering：从各个平台收集数据，然后使用 Qwen 进行过滤，筛选出 1.5K 高级数据和 140K 低级数据，然后抽取其中 1.5K 数据与高级数据结合成为 GUI R1 3K 数据集

3 Expriment

训练流程：SFT->RFT

对比：Grounding Capability，Low level task，High level task 三个方面分别利用对应的评测平台进行不同方面的评价，对评价分数取均值进行比较

结论：GUI-R1 在各个方面表现都很优秀

其他：高质量的数据有助于模型的快速收敛，提高性能，且降低 Format reward 的系数比有助于改善性能原因在于 Format 学习较为容易，通常在早期就可以学习的很好

4 Summary

GUI-R1 可以看作 GUI-Agent 模型，它负责的主要部分是 Agent 执行操作的决策部分，给定输入 (屏幕截图)，根据历史和需要完成的任务输出需要执行的操作 (类似于轨迹的输出)，然后 Agent 会执行相应的操作，创新之处在于使用 RFT 进行训练，用较少的数据集实现更好的性能。

而 OS-Genesis 则与数据集相关，传统的生成的 Trajectory 方式有两种，都是 Task-Driven，而 OS-Genesis 使用 Interaction-Driven，在使用资源较少的情况下引入评分模型，先探索交互，再用模型组成 low level instruction，再用模型组成 high level instruction，生成的数据集虽然成效不能完全比肩人类标注，但是也很不错，重点在于减少资源消耗

总的来说前者解决小样本环境下模型性能的高效提升，后者解决低成本下的高质量数据问题

而 GUI Agent 可以分成两个部分：类似于 R1 的决策部分，接收任务，根据历史和现有条件给予输出 (轨迹)，然后是执行模块，根据接收到的轨迹执行对应的操作，这一点在之前的 github 文档中也有说明，两者共同构成 GUI Agent，接收命令执行任务，需要使用 trajectory 进行训练

补充：在模型训练的 RL 环节首先先对输出进行评分 R，由上面的公式分成两部分，乘以对应权重加和得最终分数，然后再使用 GRPO 最优化进行奖励微调