arxiv.org/abs/2507.02652

Decoupled Planning and Execution: A Hierarchical Reasoning Framework for Deep Search

Jiajie Jin¹, Xiaoxi Li¹, Guanting Dong¹, Yuyao Zhang¹, Yutao Zhu¹, Yang Zhao¹, Hongjin Qian², Zhicheng Dou^{1*}

Gaoling School of Artificial Intelligence, Renmin University of China BAAI {jinjiajie, dou}@ruc.edu.cn

开源代码: https://github.com/ignorejjj/HiRA

简介

目前TIR依赖一个reasoning IIm做完整的工具推理,本文提出HiRA (Hierarchical Reasoning Architecture),将工具推理过程拆分为层级结构,包含三部分模块: HiRA = Planner(分解任务+生成最终的answer) + Coordinator (分配任务 + 蒸馏回流+memory机制) + Executors (执行任务)。注意: 1) HiRA依赖prompt engineering实现,不需要tuning IIm; 2) 作者将HiRA称为层级推理框架,我个人认为这属于multi-agent了吧。

背景

本文属于TIR领域的工作,前面读过很多,相关背景就不 多说了,说下作者的insight:

- 目前TIR依赖一个推理模型做工具推理,即thinking又 规划调用tool还负责理解tool结果,太累
- 目前TIR中对于tool的返回结果基本上直接append到 reasoning trajectory中,然后reasoning IIm继续推理,返回结果中的噪声数据/无关信息可能污染推理链
- TIR对tool的扩展能力差,现在基本RLVR训练,添加 新工具需要重新tuning

实验设置

- 完全基于prompt engineering, 不tuning Ilm
- 实验对象: QwQ作为planner和executor,
 Qwen2.5-Instruct作为coordinator
- tool: Bing Web Search API, Qwen2.5-Omni-7B作为多模态tool, Python interpreter

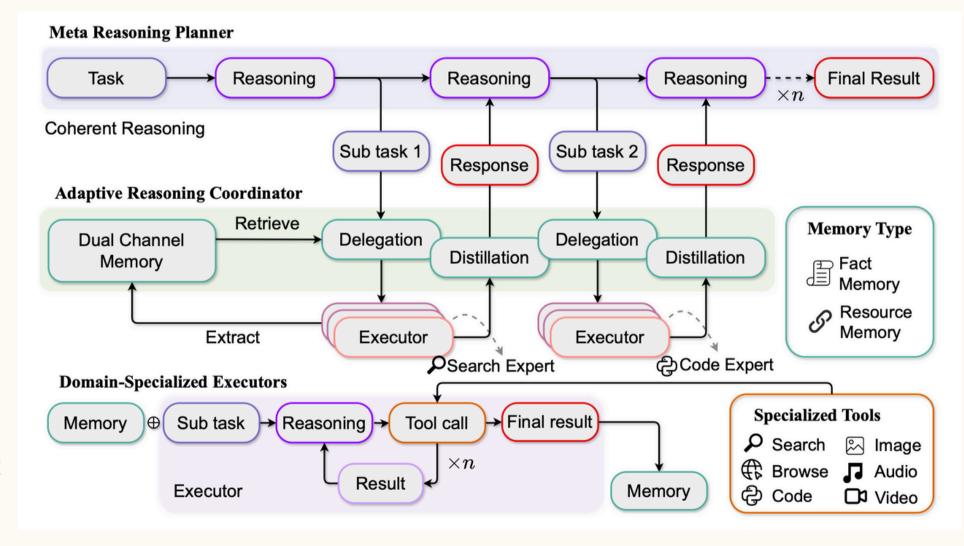
HiRA描述

右边的框架图乍一看一脸懵,其实 没那么复杂。

1.可以把planner看作TIR的 reasoning model,它们的解码过程基本一致

 $P_M(a) = P_M(a \mid q, \mathcal{O}_{<t}, \mathcal{A}_j(s_j)_{j \leq K}),$ 注意子任务描述s_j不参与解码。 2. coordinator: 1) 要根据planner 提供的子任务描述选择合适的 executor(分类问题); 2)负责将子任务执行结果先蒸馏(总结)再 refine; 3) 包含memory机制,实现跨executor的信息共享与知识迁移

3. executor: 底层干活的,很多类型的agent,每个都可以借助tool完成推理,和之前见过的TIR一样



思考

本文对Search baesd TIR进行了multi-agent改进,具体使用prompt engineering无需tuning参数,当然仔细考虑下,本文的工作可以不局限tool类型,能推广到广义的TIR。其实,planning和execution分离在TIR之前的tool-augmented LLM已经有类似工作了,比如我们之前读过的HuggingGPT,但是在以reasoning model为核心的TIR还没见过,到底是一个model还是system(比如multi-agent)更合适?难说,反正回顾技术演化的历史,架构经常在All-in-One与分布式协作之间反复交替循环。当然本文和HuggingGPT的planning还是有区别的,HuggingGPT那时候没有reasoningIIm,是先全局拆解(一次性规划完整任务依赖图),再批量调度执行,而本文用的是reasoningmodel,是一边思考一边分解任务,然后等拿到子任务结果再继续思考,所以我说planner的解码和TIR的工具推理解码是一回事。

の机器爱学习