

WebLeaper

*Empowering Efficiency and Efficacy in WebAgent
via Enabling Info-Rich Seeking*

Zhengwei Tao*, Haiyang Shen*, Baixuan Li*, Wenbiao Yin✉, Jialong Wu,
Kuan Li, Zhongwang Zhang, Huifeng Yin, Rui Ye, Liwen Zhang,
Xinyu Wang, Pengjun Xie, Jingren Zhou, Yong Jiang✉

Tongyi Lab, Alibaba Group

研究背景：LLM 代理与信息寻求的崛起

核心地位：LLM 代理是 AI 发展的下一个里程碑，**信息寻求 (IS)** 是实现自主智能的核心能力

大型语言模型代理的变革性意义

- 标志着 AI 从被动响应到主动解决问题的范式转变
- 在各领域提供前所未有的解决方案

信息寻求 (IS) 的核心作用

- IS 能力驱动代理在开放式任务中的适应性
- 是认知自主性的基础

业界前沿布局

OpenAI Deep Research

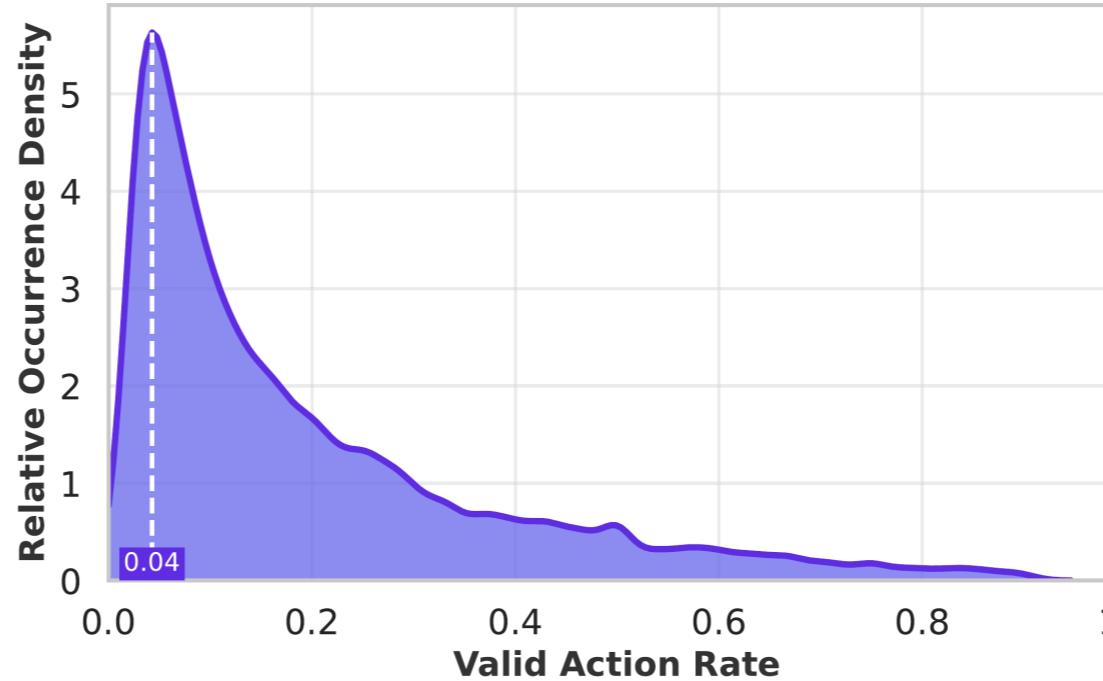
Perplexity AI

Google Gemini

Kimi-Researcher

核心发现：强大能力背后的效率瓶颈

当前 IS 代理虽然能力强大，但信息获取过程 普遍低效



有效动作的概率分布 (峰值仅 0.04)

一项关键发现：峰值仅在 0.04 附近！这意味着在一次信息寻求任务中，代理超过 95% 的动作都是在“兜圈子”——进行冗余查询、访问无关网页、或进行不必要的搜索链。

低效的三大表现

- **冗余查询 (Redundant Queries):** 反复搜索相似内容
- **无效访问 (Irrelevant Visits):** 大量访问与答案无关的网页
- **搜索路径过长 (Long Search Chains):** 为简单事实进行多轮不必要交互

探究根源：为何代理会"兜圈子"？

效率低下的根本原因在于训练数据的 "实体稀疏性" (Entity Sparsity)

实体稀疏性是什么？

- 传统数据合成：一个问题只提供极少数目标实体（1-3个）
- 例：问“谁获得了2020年诺贝尔文学奖？”答案只有1个实体
- 导致模型在有限上下文中缺乏学习机会

稀疏性的两大恶果

实践层面：一次任务只能学到1-2个“知识点”，无法学会高效“捞针”

理论层面：评估指标失效（见右侧）

理论证明（核心洞见）

$$\text{ISE} = \frac{n}{T}$$

$$\text{Var}(\text{ISE}) = \mathcal{O}\left(\frac{1}{n}\right)$$

命题 1 解读：ISE（信息寻求效率）的方差与实体数量 n 成反比。

含义：实体稀疏时，效率测量极不稳定。就像 **用晃动的尺子量长度**。

结论：必须 **构建实体密集 (n 大) 的任务**。

本文方案：WebLeaper - 为高效寻求而生

设计哲学：“既然问题出在数据，那就从数据入手。”

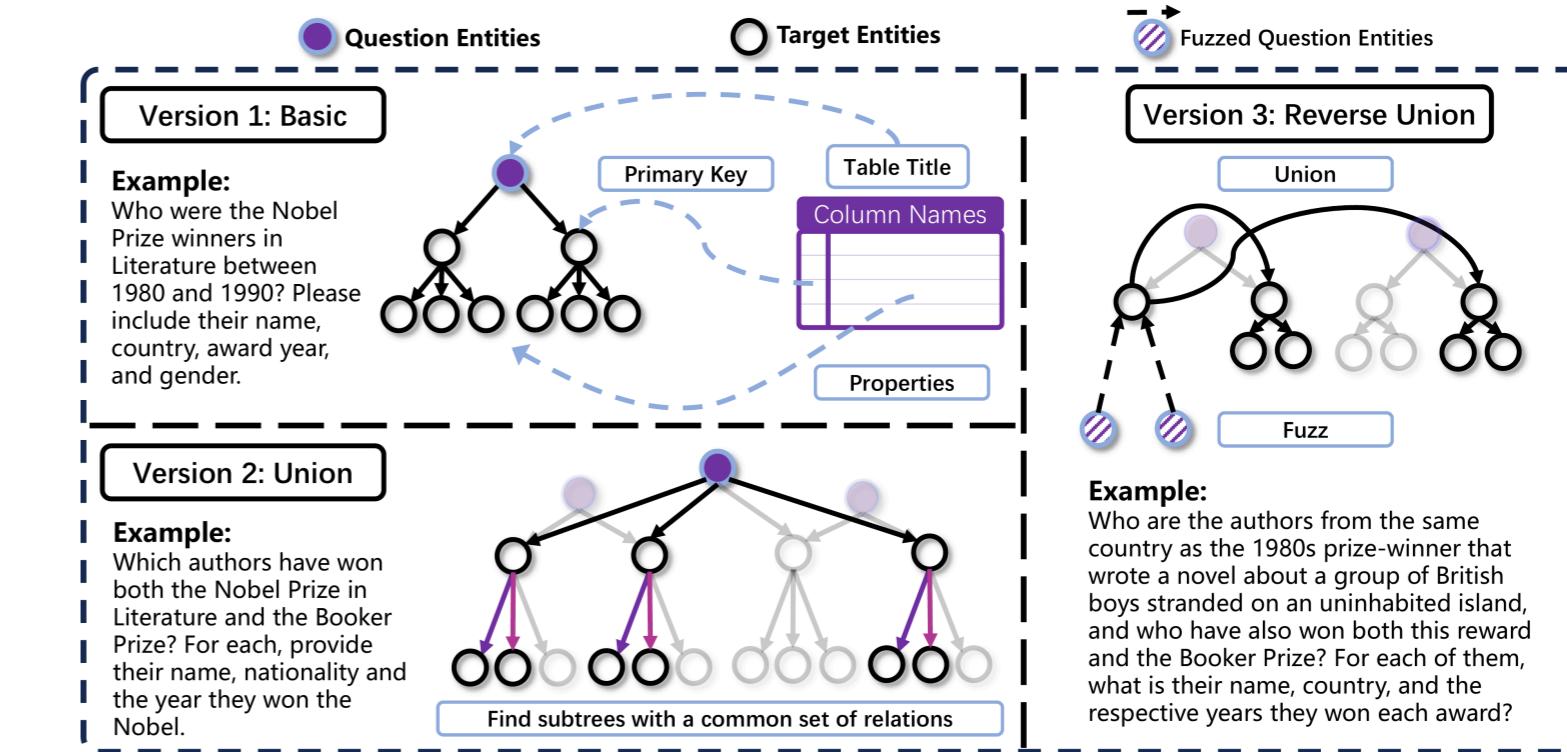
框架两大支柱

支柱一：实体密集型任务合成

- 不再满足于“一问一答”
- 设计“一问多答”甚至“一问百答”的复杂任务
- 三个递进版本：
Basic → Union → Reverse-Union

支柱二：效率感知的轨迹构建与学习

- 生成并筛选“最优解法”（又快又准的轨迹）
- 设计相应奖励机制引导模型学习
- ISR & ISE 双指标过滤 + 混合奖励系统



WebLeaper 框架：树状结构 + 三个递进版本

第二部分 方法深度剖析

WebLeaper 方法论详解 - 路线图

Part 1

实体密集型任务合成

- Version-I: **Basic**
单源高密度
- Version-II: **Union**
多源整合
- Version-III: **Reverse-Union**
逆向推理

Part 2

效率感知的轨迹构建

- 数据源选择
维基百科表格清洗
- ISR & ISE 指标定义
准确性 + 效率
- 高质量轨迹筛选
双重过滤标准

Part 3

混合奖励的强化学习

- 奖励设计挑战
稀疏 + 评估困难
- 软 F-Score 奖励
细粒度 + 语义感知
- GRPO 优化算法
相对优势估计

任务合成的基石：树状结构 + 维基百科表格

为何选择树状结构？

- 层次清晰、结构紧凑：一个任务中自然容纳大量从属实体
- 符合人类思维：从主题发散到子主题，再到具体属性

- 根节点 (Root): Question Entity
- 第二层 (Layer 2): Target Entities (Intermediate)

数学定义：

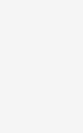
- 第三层 (Layer 3): Attributes/Properties

为何选择维基百科表格？

- 天然的结构化数据：表格本身就是实体和关系的集合
- 数据丰富：覆盖领域广泛，实体数量庞大
- 高质量：经过社区编辑和审核

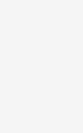
数据清洗严谨性

起点：2,000,000 张维基百科表格



多阶段清洗流程：

- 大小过滤 (Size Filtering)
- 语义过滤 (Semantic Filtering)
- 同构筛选 (Homogeneity Screening)



结果：高质量、结构化的表格数据集

Version-I: Basic - 单源高密度

核心观点：从单个表格出发，构建三层推理树，直接解决实体稀疏性问题

构建过程

第一层 (根/问题)

- 来源：表格标题
- 示例："Nobel Prize in Literature laureates"

第二层 (子树根)

- 来源：LLM 自动识别的"主键"列
- 示例：获奖作家姓名 (Czesław Miłosz, Mo Yan, ...)
- 每个姓名都是一个独立子树根

第三层 (叶/属性)

- 来源：表格的其余列
- 示例：Country (Poland), Year (1980), Gender (Male)

任务示例

"请列出 1980-1990 年间诺贝尔文学奖得主及其国籍和获奖年份。"

贡献与意义

- `Basic` 版本首次将任务实体密度 提升了数个数量级
- 为代理提供了在单次任务中学习处理和组织大量信息的机会
- 解决了传统方法的稀疏性问题

统计数据

指标	Basic	传统方法
平均实体数	20-50	1-3
密度提升	10-50 倍	

Version-II: Union - 跨越多源信息整合

核心：通过“最大二分团枚举”智能融合多个 Basic 任务，创造需要跨源信息整合的复杂任务

动机：真实世界问题往往需要整合多个来源的信息

技术核心 - 最大二分团枚举

“如何自动发现哪些表格可以被‘有意义’地合并？”

1. 构建二分图

- 左：Basic任务树；右：关系集合
- 树包含关系则连边

2. 寻找最大二分团

- 找共享最多共同关系的任务树
- 高效智能的搜索策略

任务生成示例

发现：'Nobel Prize' 和 'Booker Prize' 共享：

has_author, has_nationality, has_award_year

生成问题：“Which authors have won both the Nobel Prize in Literature and the Booker Prize? List their nationalities.”

贡献

- 迫使代理学会 跨源信息整合
- 需要关系运算（如求交集）
- 显著提升任务认知复杂度

Version-III: Reverse-Union - 逆向推理对抗捷径

核心观点：通过颠倒推理顺序，强制代理进行"线索演绎 → 扩展搜索"的深度推理，防止依赖关键词匹配走捷径

动机

作者指出，即使是 Union 任务，代理也可能通过分别搜索再取交集的方式"走捷径"。`Reverse-Union` 的设计正是为了防止这种情况。

两阶段设计

第一阶段：演绎模糊化 (DEDUCTIVE FUZZ)

- 起点：不是明确的实体，而是一组描述性"线索"（第三层实体）
- 示例：

"那位在 1980 年代获奖、写了一本关于一群英国男孩被困荒岛的小说的作家..."

- 要求：代理必须先 **推理** 出这是 "William Golding"
- 目的：防止直接关键词搜索

第二阶段：基于联合的扩展搜索

- 推理后：任务并未结束
- 扩展：利用推理出的实体的某个属性（如国籍"英国"）作为"枢轴"
- 最终问题：

"...找出所有和他一样是英国国籍，并且同时获得了诺贝尔奖和布克奖的作家。"

贡献

轨迹构建：如何定义“好”的解法？ (1/2)

核心：好的解法兼具 **准确性** 和 **效率**。定义 ISR 和 ISE 两个核心指标。

信息寻求率 (ISR)

$$\text{ISR} = \frac{|R \cap O|}{|R|}$$

- **衡量：**找得全不全（召回率/覆盖率）
- **R：**目标实体集 (Ground Truth)
- **O：**代理找到的实体集
- **范围：** [0, 1]，越高越好

示例：如果目标有10个实体，代理找到了8个，则 $\text{ISR} = 8/10 = 0.8$

信息寻求效率 (ISE)

$$\text{ISE} = \frac{n}{T}$$

- **衡量：**找得快不快（单位步数的实体数）
- **n：**目标实体总数
- **T：**总工具调用步数
- **含义：**每一步平均找到多少目标实体

示例：如果10个实体用了5步找到，则 $\text{ISE} = 10/5 = 2.0$ (每步找到2个实体)

轨迹构建：如何定义"好"的解法？ (2/2)

轨迹筛选策略 - 双重过滤标准

标准： $\text{ISR} > \alpha$ (足够准确) 且 $\text{ISE} > \beta$ (足够高效)

轨迹过滤流程

步骤 1：生成大量轨迹（使用基础模型）



步骤 2：计算每条轨迹的 ISR 和 ISE



强化学习：混合奖励系统 (1/2)

核心观点：针对实体密集型任务的奖励稀疏和评估困难问题，设计混合奖励系统

RL 面临的困境

1. 奖励稀疏 (Reward Sparsity)

- 问题：几十个实体全对才给奖励
- 后果：模型永远学不会（正反馈太少）

2. 评估困难 (Evaluation Challenge)

- Exact Match: 太死板，无法处理同义词 ("USA" vs "United States")
- LLM-as-a-Judge: 又贵又不稳定，难以大规模应用

解决方案：设计软 F-Score (Soft F-Score) 混合奖励，兼顾细粒度反馈和智能评估

强化学习：混合奖励系统 (2/2)

解决方案：软 F-Score (Soft F-Score)

1. 语义相似度函数

$$s(e_o, e_r) \in [0, 1]$$

理解同义词和语义等价

2. 软召回率

$$\mathcal{R}_c = \frac{1}{|R|} \sum_{e_r \in R} \max_{e_o \in O} s(e_o, e_r)$$

找到了多少目标实体

3. 软精确率

$$\mathcal{P} = \frac{1}{|O|} \sum_{e_o \in O} \max_{e_r \in R} s(e_o, e_r)$$

找到的实体有多准确

4. WebLeaper 最终奖励

$$\begin{aligned} \mathcal{R}_w &= (1 + \omega^2) \\ &\frac{\mathcal{P}}{\omega^2 \mathcal{P} + \mathcal{R}_c} \end{aligned}$$

平衡精确率和召回率

优势：细粒度反馈 + 智能评估 + 兼容性（保留传统数据集的评估方式）

强化学习：GRPO 优化算法 (1/2)

核心观点：采用 GRPO 算法，通过比较一组轨迹的相对好坏进行优化，比 PPO 更适合奖励复杂场景

为什么不用传统 PPO？

- PPO 依赖学习出的**价值函数 (Value Function)**
- 当奖励函数复杂多变时，价值函数很难学准
- 导致训练不稳定

问题：该研究提出的混合奖励系统包含多个复杂组件，传统 PPO 难以应对

GRPO 的核心思想

Group Relative Policy Optimization

(组相对策略优化)

核心：一次生成 k 条轨迹，在组内进行“横向比较”，不依赖不稳定的价值函数，而是用 **相对优势**

类比：不需要知道每个学生的绝对分数，只需要知道谁比班级平均水平好多少

强化学习：GRPO 优化算法 (2/2)

优势估计公式

$$\hat{A}_i = \frac{R_i - \text{mean}(\{R_j\})}{\text{std}(\{R_j\}) + \epsilon}$$

- R_i : 轨迹 i 的奖励
- 相对于同组其他轨迹的标准化优势

GRPO 损失函数

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\mathbb{E} \left[\sum_i \sum_t \min \left(r_{i,t}(\theta) \hat{A}_i, \text{clip}(r_{i,t}(\theta), 1 - \varepsilon, 1 + \varepsilon) \hat{A}_i \right) \right]$$

使用clipping确保策略更新稳定

GRPO 的三大优势

1. 相对信号更稳定
 - 不需要绝对价值估计
 - 只需知道“轨迹 i 比同组其他轨迹好多少”
2. 适合复杂奖励
 - 面对混合奖励系统依然稳健
 - 不受单个奖励函数波动的影响
3. 训练效率高
 - 无需训练单独的价值网络
 - 减少计算开销

工作流程：采样 k 条轨迹 → 计算奖励 → 组内标准化得到相对优势 → 更新策略 → 重复

第三部分

实验结果与分析

实验设置：在严苛环境中验证实力

基准 (Benchmarks)

基准	描述	指标
BrowserComp	复杂浏览任务	Pass@1
GAIA	通用 AI 助手	Pass@1
xbench-DS	深度搜索评估	Pass@1
Seal-0	复杂问答任务	Pass@1
WideSearch	宽度搜察能力	SR, F1

基线模型 (Baselines)

闭源/专有代理:

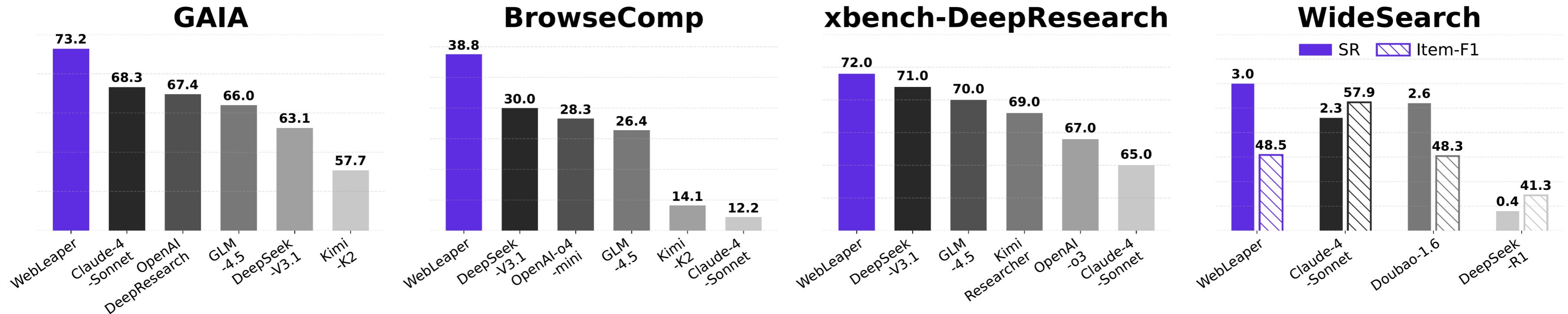
- Claude-4-Sonnet
- OpenAI-o3
- OpenAI DeepResearch

开源代理:

- ASearcher-Web-32B
- DeepDive-32B
- DeepDiver-V2-38B
- MiroThinker-32B
- Kimi-K2-1T
- WebExplorer-8B
- WebDancer-32B
- WebSailor-32B
- WebShaper-32B

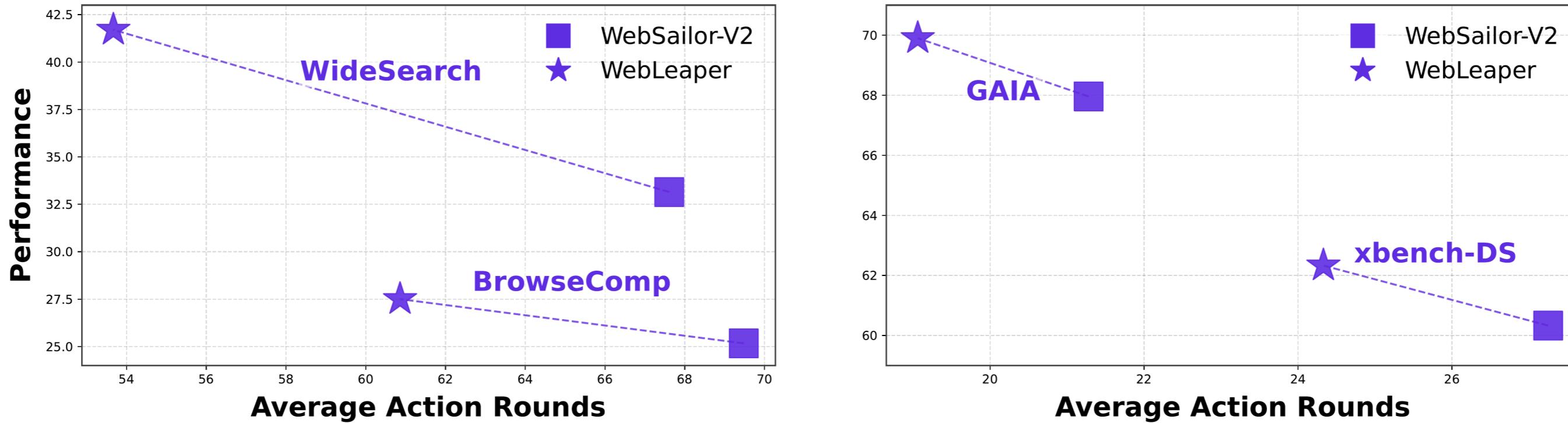
训练配置

核心结果 (1): 全面超越，刷新开源模型记录



在所有基准上的性能对比 (Comprehensive Setting)

核心结果 (2): 不仅做得更好，而且用时更少 (1/2)



效率 (工具调用次数) vs 效果 (任务得分) 对比 - 四个基准测试

"左上角"的含义:

- 左 (Less Tool Calls) → 更高效
- 上 (Higher Score) → 更有效

核心发现: 在所有四个基准上, 蓝点 (WebLeaper) 都稳定地出现在橙点 (基线) 的左上角。

核心结果 (2): 不仅做得更好，而且用时更少 (2/2)

具体数据对比

GAIA 数据集

模型	分数	步数
WebLeaper	73.2	12
WebSailor	53.2	18
提升	+20	-6
改进率	+38%	-33%

BrowseComp 数据集

模型	分数	步数
WebLeaper	38.8	8
最强基线	15.7	14
提升	+23.1	-6
改进率	+147%	-43%

结论：这印证了作者们的核心假设：**效率和效果不是零和博弈**。通过优化效率，可以同时提升效果。该研究的代理学会了如何“聪明地”工作，而不是“费力地”工作。

深入分析(1): 为何 Reverse-Union 效果最好? (1/2)

不同数据源版本的性能对比

Data Source	BrowseComp	WideSearch	GAIA	Seal-0	xbench-DS	Avg.
WebSailor-V2-5k	25.17	33.15	67.69	34.23	60.00	44.05
Basic-5k	20.67 (-4.50)	32.26 (-0.89)	40.78 (-26.91)	30.03 (-4.20)	58.33 (-1.67)	36.41 (-7.64)
Union-5k	27.50 (+2.33)	41.70 (+8.55)	69.90 (+2.21)	35.14 (+0.82)	62.33 (+2.33)	47.31 (+3.26)
Reverse-Union-10k	27.67 (+2.50)	44.07 (+10.92)	66.99 (-0.70)	37.24 (+3.01)	66.00 (+6.00)	48.39 (+4.34)

- Basic 版本在 GAIA 上 **大幅下降 -26.91**, 说明任务过于简单
- Union 版本 **平均提升 +3.26**, 证明结构复杂度重要

关键观察: • Reverse-Union **平均提升 +4.34**, 效果最佳

深入分析 (1): 为何 Reverse-Union 效果最好? (2/2)

1. Basic 版本的问题

- ✗ 性能反而下降，尤其在 GAIA 上 (-26.91)
- 原因：任务过于简单，模型学会了"抄近道"
- 后果：过拟合简单模式，泛化能力变差
- 教训：仅仅增加实体密度还不够，**任务复杂度同样重要**

2. Union 版本的提升

- ✓ 平均提升 +3.26，大部分基准都有提升
- 原因：多源信息整合迫使模型进行复杂推理
- 意义：证明了结构复杂度的重要性

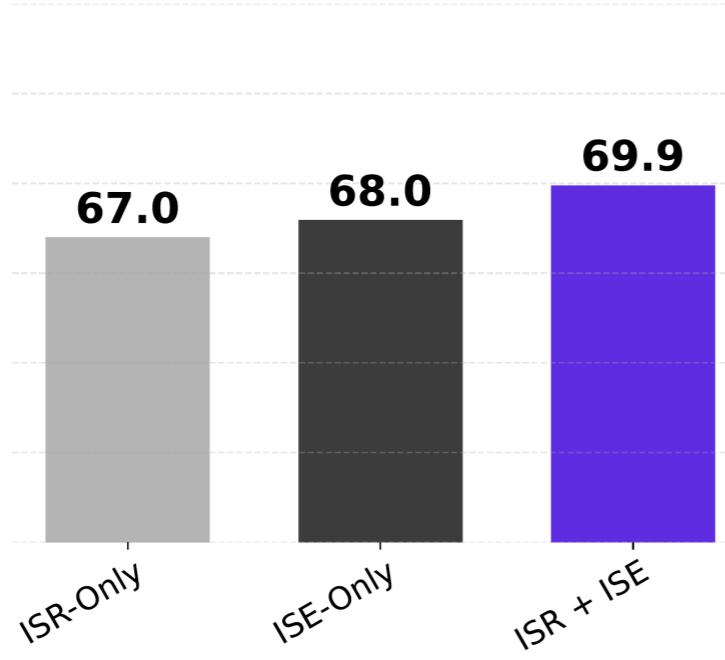
3. Reverse-Union 版本的最佳效果

- ✓ 平均提升 +4.34，在大多数基准上最好
- 原因：
 - 不仅结构复杂，还有逆向推理设计
 - 杜绝了关键词搜索捷径
 - 先演绎 → 再规划 → 再搜索（锻炼高级认知能力）
- 特别优势：在 WideSearch 上提升最大 (+10.92)

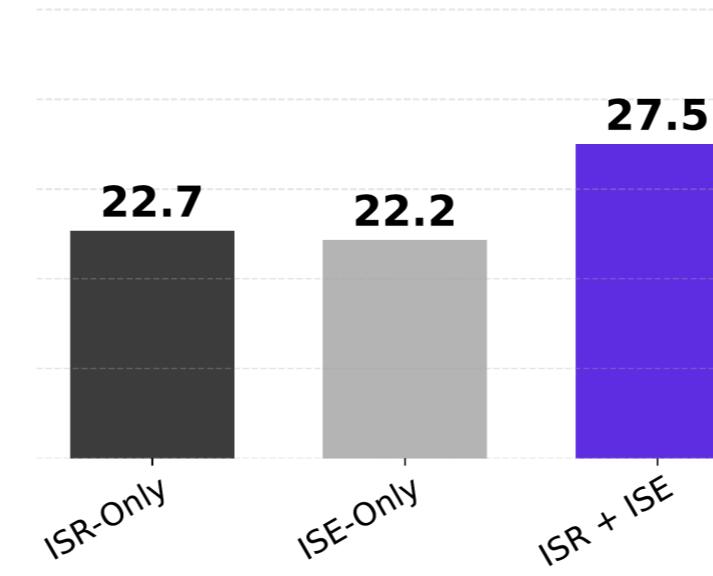
结论：任务复杂度的递进设计至关重要。Reverse-Union 通过逆向推理，从根本上提升了代理的**规划和推理能力**。

深入分析(2): 如何筛选出"黄金轨迹"?

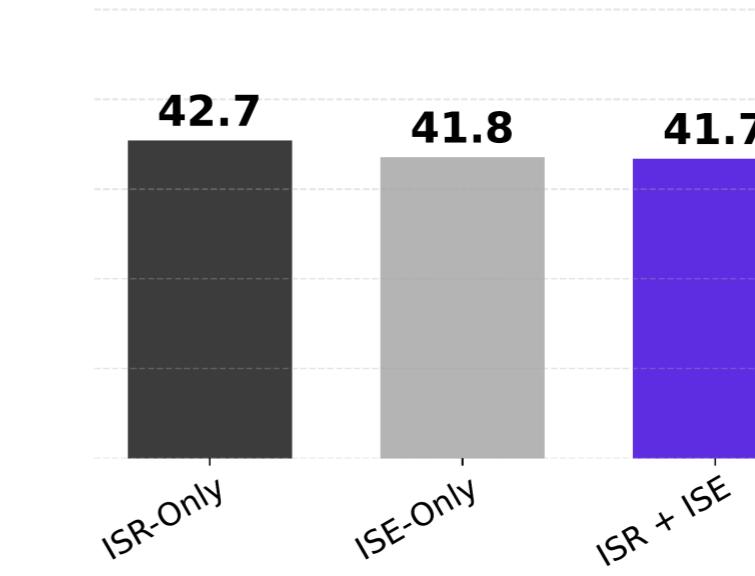
GAIA



BrowseComp



WideSearch



不同轨迹筛选策略的效果对比

深入分析 (3): 强化学习带来的"最后一跃" (1/2)

RL 性能提升

Stage	BrowseComp	GAIA	xbench-DS
SFT	37.80	69.9	69.0
SFT+RL	38.8 (+1.0)	73.2 (+3.3)	72.0 (+3.0)

全面性能提升

- 四个基准均有提升
- **最大增益:** WideSearch SR (+2.5) 与 Row F1 (+8.0)
- **稳定增益:** GAIA (+3.3)、 xbench-DS (+3.0)

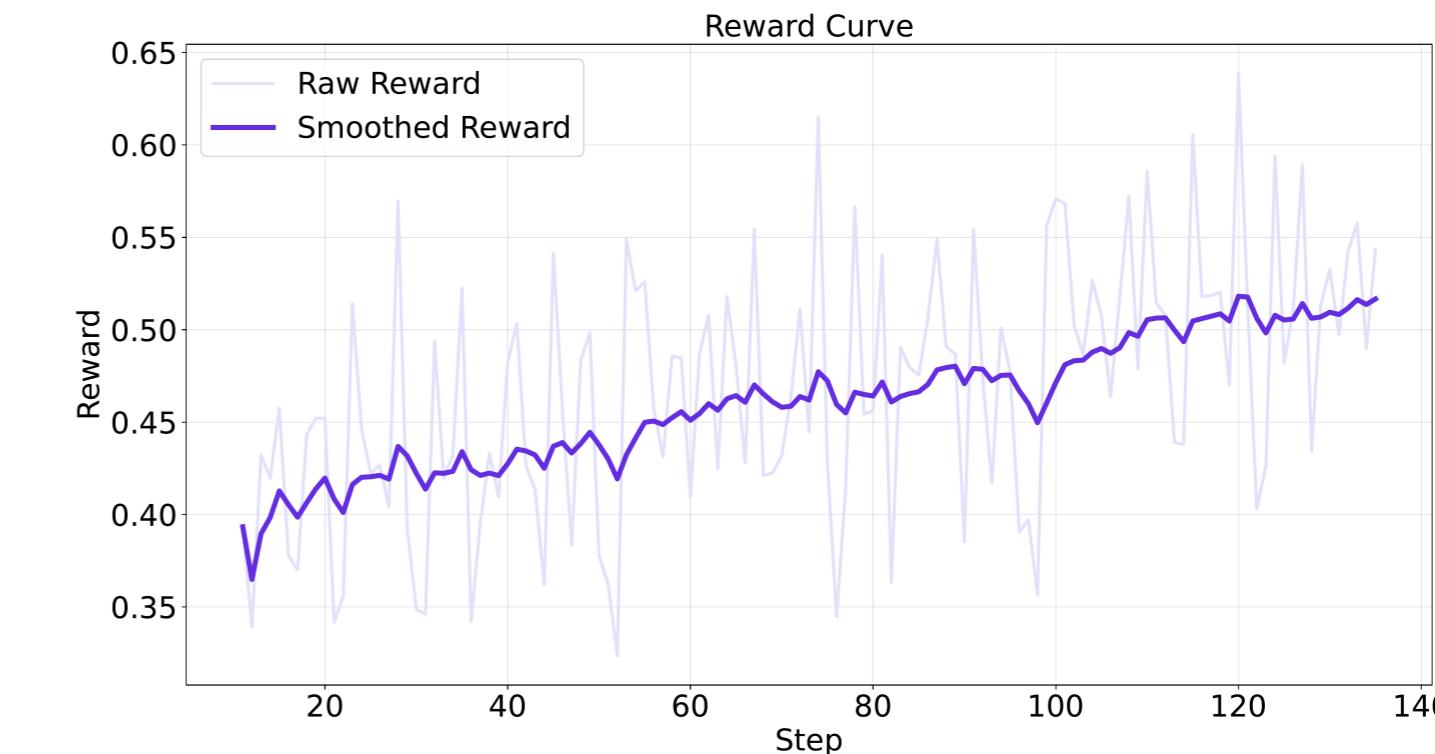
启示: RL 补上了 SFT 的“最后一跃”，将策略从“会做”推向“做得更优”。

深入分析(3): 强化学习带来的"最后一跃" (2/2)

RL 特别有效的场景

- **WideSearch:** 需要精细策略的任务, RL 提升巨大
- **GAIA:** 需要准确判断的任务, RL 带来 3.3 个点提升

小结: RL 通过稳定、细粒度的奖励信号, 促使代理形成更高效的检索-推理循环。



混合奖励系统的训练曲线

奖励曲线解读

- ✓ 稳步上扬: 学习有效
- ✓ 无剧烈震荡: 训练稳定
- ✓ 持续增长: 尚有潜力 (约 135 步停止)

结论: WebLeaper 数据不仅是优秀的 SFT 教材, 更是高质量的 RL 训练场, 成功引导代理学会更高级、更优化的信息寻求策略。

案例分析：WebLeaper vs. 普通代理（1/2）

示例任务：“找出与《蝇王》作者同一国籍，且获得过诺贝尔文学奖的另一位作家。”

✗ 普通代理的思考路径

Step 1: Search("蝇王 作者")

→ 找到 "William Golding"

Step 2: Search("William Golding 国籍")

→ 找到 "英国"

Step 3: Search("诺贝尔奖作家名单")

→ 得到一个长列表（上百名）

Step 4: Search("A 的国籍")

Step 5: Search("B 的国籍")

...

Step N: 逐一排查（非常低效）

总步数：15-20 步

特点：机械、繁琐、缺乏规划

✓ WebLeaper 的思考路径

Step 1: Search("《蝇王》作者 国籍")

→ 一步到位：

William Golding, 英国 ✓

Step 2: Search("获得诺贝尔奖的英国作家")

→ 直接将多个条件组合成高效查询

→ 返回：

Winston Churchill,

Kazuo Ishiguro, ...

Step 3: Visit(相关页面)

→ 找到答案 ✓

总步数：3-4 步

特点：智能、高效、有规划

案例分析：WebLeaper vs. 普通代理 (2/2)

详细对比分析

维度	普通代理	WebLeaper
查询策略	单一、线性	组合、并行
信息整合	逐步累积	一次到位
步数	15-20	3-4
效率	低 (20%)	高 (75%)

• 普通代理：像“盲目的探险家”，每次只走一小步，看到什么就记录什么

核心差异：• WebLeaper：像“经验丰富的猎手”，会先规划路线，将多个条件组合成精准查询

关键启示：该研究表明，高效寻求的本质不是“快速移动”，而是“智能规划”。WebLeaper 学会了如何将复杂问题分解为最优查询序列。

第四部分 结论与展望

相关工作：站在巨人的肩膀上

信息寻求代理

现有工作的三大流派：

- **模型微调**

WebSailor, WebDancer, WebShaper,
DeepDive

- **架构改进**

多阶段推理、分层规划

- **多代理协作**

WebResearcher、协作式搜索

WebLeaper 的定位：

属于模型微调流派，独特贡献是引入 "实体丰富度" 这一新维度

代理数据合成

现有关注点：

- **多步推理**

推理链的长度和复杂度

- **长视界规划**

跨多个子任务的规划

WebLeaper 的不同之处：

关注数据的 "语义丰富度" 和 "实体密度" ——被忽视但至关重要的方向

强化学习奖励设计

现有方法：

- 二元奖励 (Binary Reward)

- LLM-as-a-Judge

- 过程奖励模型 (PRM)

WebLeaper 的贡献：

- **混合奖励系统**：软 F-Score + 传统奖励
- **GRPO**：相对优势 vs. 绝对价值

总结：WebLeaper 并非要取代现有工作，而是从一个 **新的、互补的角度** ——即提升数据的信息密度和训练效率——来推动整个领域的发展。

核心贡献总结

诊断了核心病症：现有 IS 代理的效率瓶颈源于训练数据的 实体稀疏性

贡献一

实体密集型任务合成

- **Basic Version**
 - 解决实体稀疏性
 - 单源高密度（20-50 个实体/任务）
- **Union Version**
 - 跨源信息整合
 - 关系运算（交集、并集）
- **Reverse-Union**
 - 逆向推理对抗捷径
 - 演绎 → 规划 → 搜索

贡献二

效率感知的轨迹构建

- **ISR (Information-Seeking Rate)**
 - 衡量准确性（召回率）
 - $ISR = |R \cap O| / |R|$
- **ISE (Information-Seeking Efficiency)**
 - 衡量效率（单位步数收益）
 - $ISE = n / T$
- 双重过滤
 - 筛选“又准又快”的黄金轨迹

贡献三

混合奖励与 GRPO 优化

- **软 F-Score 奖励**
 - 细粒度反馈
 - 语义智能评估
- **GRPO 算法**
 - 相对优势估计
 - 稳定训练信号
- **混合策略**
 - 兼容多种数据源

Thanks!

Q&A

核心贡献回顾

- 实体密集型任务合成 (Basic, Union, Reverse-Union)
- 效率感知轨迹构建 (ISR & ISE)
- 混合奖励强化学习 (Soft F-Score + GRPO)

关键数字

73.2

GAIA (开源最佳)

38.8

BrowseComp

72.0

xbench-DS

-33%

步数减少

代码、数据、模型全部开源，欢迎使用和贡献！

