Agent-as-a-Judge 综述分享

LiuKai

2026.1.28

目录

本节内容

LLM-as-a-judge

LLM-as-a-judge 简介

LLM-as-a-judge 是我们今天的主题的背景，他们的功能是类似的，都是充当一个 Graders,Assessors，例如评价一段文本写的怎么样，通不通顺，用词是否优美或者正确度如何。只是评价者从单个 LLM 变成了多个 Agent 协作。

LLM-as-a-judge

LLM 评价主要是两个式子:

$$\mathcal{E} \leftarrow \mathcal{P}_{\mathcal{LLM}}(\boldsymbol{x} \oplus \boldsymbol{C})$$

- $\mathcal{E}$: 最终的得分，它可能只是一个分数，一个 Yes/No 的选择，或者一个评价的句子。
- $\mathcal{P}_{\mathcal{LLM}}$: 对应 LLM 定义的概率函数，生成是一个自回归过程。
- $\boldsymbol{x}$: 任何可用类型的输入数据（文本、图像、视频），等待被评估。
- $\boldsymbol{C}$: 输入 $\boldsymbol{x}$ 的上下文，通常是提示模板或结合对话中的历史信息。
- $\oplus$: 组合操作符，将输入 $\boldsymbol{x}$ 与上下文 $\boldsymbol{C}$ 结合，这个操作根据上下文的不同可以放在开头、中间或结尾。

$$\mathcal{R} \leftarrow \boldsymbol{f}_{\mathrm{R}}(\mathcal{P}_{\mathcal{LLM}}, \boldsymbol{x}, \boldsymbol{C})$$

- $\mathcal{R}$: 这个检测指标设计是用来确保一致性，鲁棒性，和人类评价的一致性。这种可靠性将会通过其他的手段/校准得到验证。
- $\boldsymbol{f}_{\mathrm{R}}$: 一系列的限制和验证措施，用于保证评价的的可靠性。其中包括减少偏差，控制变量，确认在对抗输入下的鲁棒性

LLM-as-a-judge

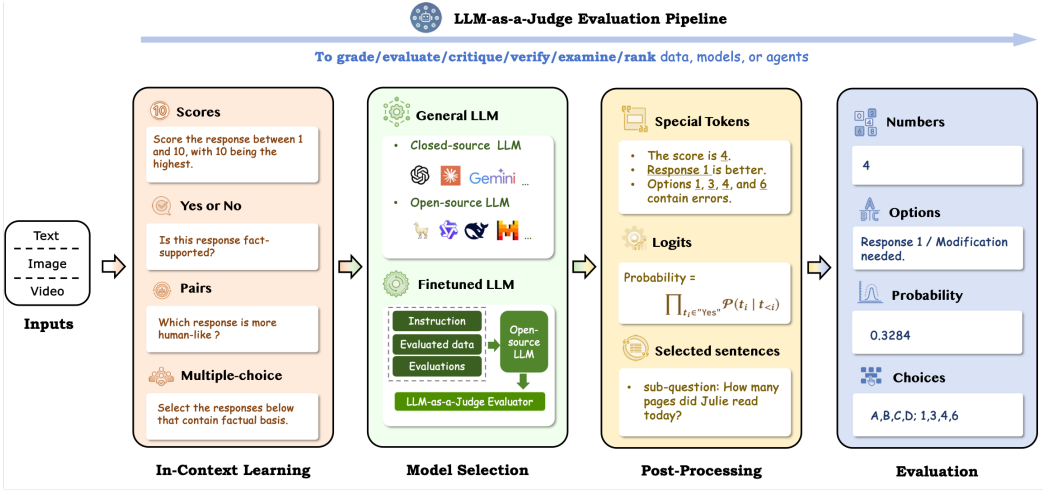

图: 流程图

本节内容

Agent 的三个优势：鲁棒性

。

首先, 在 LLM-as-a-judge 综述 [1] 第 43 页 6.1 节中就提到，单一的 LLM 在面对某些输入时会存在 *bias* , 例如自己生成的文本或者符合自己生成风格的文本，同时在 6.2 节中在面对 *AdversarialAttacks* , 即通过某种输入方式操纵的得分结果，而且单个 LLM（尤其是开源模型），有可能面对 *Jailbreaking* 的攻击。而通过 Agent 的方式，因为 Agnet 在将任务分解成多个子任务，同时在子任务中间检测中间插入搜索，工具调用等方式，引用各种专业知识进行解答，这样就可以有效的避免这些问题，从而提升鲁棒性。同时对于单个 LLM 存在的 *bias* 问题，因为 Agent 的法庭辩论机制，可以让多个 Agent 互相辩论，从而减少单个 LLM 的 *bias* 影响。例如，对于一篇论文的评价，可能 LLM 只观察其句子是否通顺从而给出一个打分，但是 Agent 可以让一个 Agent 负责检查论文的创新性，另一个 Agent 负责检查论文的实验代码的运行，最后再让一个 Agent 负责综合打分。

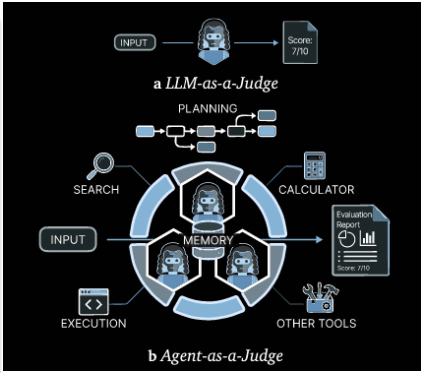

图: 对比图

Agent 的三个优势：可验证

原先的 LLM-as-a-judge 依靠直觉，即通过语言的流畅度，似真性，来评判“看起来有多真”。现在 Agent 更多是从实践的角度出发，代码使用解释器去运行，内容运用搜索引擎，阅读对应的文档进行查证，还可以查询系统状态。这些工具的集成都是之前 LLM 说不具备的能力。同时，Agent 可以将评估的过程拆解成多个步骤，每个步骤都可以被单独验证和检查，从而确保每个部分的正确性。例如，在评估一个复杂的任务时，Agent 可以先进行数据预处理，然后进行模型训练，最后进行结果评估。每个步骤的输出都可以被单独验证，从而确保整个评估过程的可靠性。

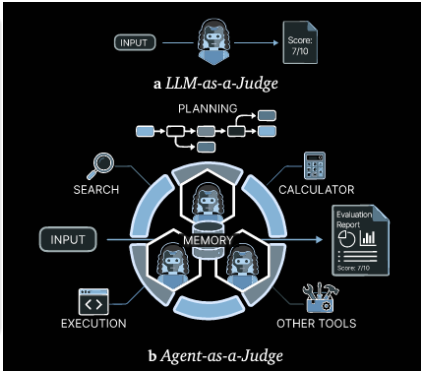

图: 对比图

Agent 的三个优势：细粒度

在对比图和上述的介绍中我们知道，LLM-as-a-judge 最后的 output 可能只是一个分数或者一段话。这些评价是比较模糊的。在 LLM-as-a-judge 综述 [1] 的 6.4 节也提到了判决书的可解释性与透明性。评判的黑盒过程与最后输出的模糊性导致评价的可信度下降。而在 Agent-as-a-judge 中，首先 LLM 可以动态制定裁判规则，然后将一个大任务分解成若干个子任务，每个子任务之间因为 LLM 具有 Memory 的特性，从而可以根据上一个阶段的发现决定下一个阶段，最后由一个 LLM 负责综合裁决。这样每个子任务的输出都是细粒度的，从而提升了整体评价的细粒度。

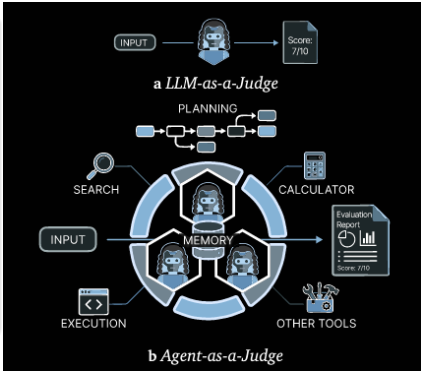

图: 对比图

本节内容

Multi-Agent Collaboration

Collective Consensus

这种利用多个 LLM 的集体意识，可以有效避免 *bias* 和幻觉问题。不过这个避免 *bias* 的原理并不简单的，一个 LLM 与另一个 LLM 之间偏见的抵消。而是通过一种法庭辩论的机制，让多个 Agent 相互辩论，从而得出结果。在 Multi-Agent-as-Judge[2] 3.2 节中提出的 MAJ-Eval 方法中，第一阶段会有一个 LLM 识别输入的利益相关者，然后根据每个利益相关者构建一个相对应的人设，人设被实例化后，先进行自我裁判，独立评估，然后多个智能体组内自由辩论，然后由一个” 法官” 智能体进行最终裁决。

Tasks Decomposition

Tasks Decomposition 就是指一个任务采取“分而治之” 的策略，将复杂任务拆解成多个子任务，而每个子任务被委派不同的 LLM 去完成，而且还可以根据任务的难度动态调整每个任务委派的 LLM 的能力。这同时也对应了 Collective Consensus 中的法庭辩论机制。将每个分派到不同任务的 LLM 作为一个辩论者，然后最后由一个 LLM 作为法官进行裁决。

Planning

Workflow Orchestration

Workflow Orchestration 主要分为两个阶段。第一个阶段是 Agent 根据预先制定好的机制，将任务分解成固定的序列。第二个阶段事 Agent 可以根据任务类型的不同随时随地调整策略，同时还可以根据上一步即时的反馈，可以根据证据及时暂停。在 Evaluation Agent[3] 中的 3.2 节中，提到的 Plan Agent 可以模拟评估过程中的人类行为，包括括规划和调整评估方向、观察中间结果、总结最终结果。

Rubric Discovery

Rubric Discovery 主要是指 Agent 可以根据不同的任务动态制定评估标准，例如对两篇不同的文章小说和医学报告，前者的文采和情节设计可能更加重要而后者的正确性和专业程度更加重要。在 EvalAgents[4] 中第 3 节中提到 EvalAgent 方法会从教学文档中提取评判标准。还有一个 *QueryGenerator* 模块，给定指令 x，它会提示 LLM 生成概念性查询 Q，Q 的每个元素都是有助于 x 的教学建议。*ExpertRetriever* 则会过滤掉一些不重要的概念。然后 *CriteriaGenerator* 将所有查询进行综合，最后根据原始的输入把每个指标进行排序。

Tools Integration

Evidence Collection

Evidence Collection 主要是 Agent 可以调用外部工具收集证据，进行验证。其中包括搜索引擎，数据库，代码解释器，还有多模态工具进行图像视频的处理等。这些工具将抽象的视频图像，代码等整合成与任务相关的证据上，让“法官”获得更多信息从而会有更可靠的评估。

Correctness Verification

Correctness Verification 中，Agent 就像一个评委一样去查验内容的真伪，对某些关键的部分（断言或步骤）调用 LLM 或工具去验证。例如 HERMES[5] 第三节提出的 HERMES 框架，这个框架分成四个模块，第一部分推理模块，生成自然语言的解题步骤；第二部分翻译模块，将自然语言步骤翻译成 Lean4 的形式化代码；第三部分验证模块，使用 Lean4 的证明引擎去验证代码的正确性；第四部分反馈模块，将验证的结果反馈给推理模块，如果出现错误，就会把 Lean4 的报错信息翻译成自然语言返回给推理模块。HERMES 的可验证性大大提升了数学问题求解的可靠性。

Memory and Personalization

Intermediate State

在任务分解成多个子任务中，评估也被分解成多步，在多步评估的设置中，Intermediate State 主要是指在每个子任务之间保存中间状态，从而让后续的子任务可以根据前面的结果进行调整。比如前面提到 HERMES 框架中就会保留前面步骤的证明结果，从而让后续的步骤可以根据前面的结果进行调整。

Personalized Context

Personalized Context 是 Agent 可以通过之前的回答调整评判的标准，类似于 PersRMR1,FSPO[6] 会存储偏好标签和历史案例，RLPA[7],SYnthesizeMe[8] 会生成用户画像。例如 RLPA[7] 在第三节提出的方法，通过强化学习进行个性化对齐，生成用户画像。首先 RLPA 在每一轮的对话中，不仅生成回复，还会更新它心中对于用户的印象，而为了训练 Agent，使用模拟用户与 Agent 进行对话，看 Agent 生成的用户画像是否与模拟用户的真实画像一致，将回答的质量和与真实画像的对应程度作为奖励训练 Agent, 从而提升 Agent 猜测用户画像的能力。

Optimization Paradigms

Training-Time Optimization

Training-Time Optimization 是指在训练时，更新模型参数，利用监督微调和强化学习去实现更可靠的评估。这一点主要是规范 Agent 判断行为，让模型遵循明确的标准。例如 ARM-Thinker[9] 3.2 节中所提到的 Supervised Fine-Tuning & Cold Start Generation，ARM-Thinker 是一个多模态集成的工具，在实验中会创建专门的数据集，专门模型在什么情况下该用什么工具

Inference-Time Optimization

现有的 Inference-Time Optimization 分为两种，第一种是固定推理步骤，验证程序和提示词，确保一致性和效率。第二种是允许评估行为在推理过程中自适应，修改评判过程。前面提到的 Evaluation Agent[3] 和 HERMES[5] 都是属于第一种。总的来说，Inference-Time Optimization 可以实现对评估行为的灵活控制, 从固有的程序设定到 Agent 自适应调整的判断

本节内容

General domains

- Math and Code 例如前面提到 HERMES[5] 就是使用 Agent 集成证明数学定理和解答数学问题
- Fact-Checking 事实检查。例如论文 Fact-AUDIT[10] 就是通过多智能体训练一个事实检查系统，一个 Generator 生成伪造事实，一个 Retriever 检索相关的证据与事实，一个 Evaluator 负责判断 Retriever 的回答是否正确，理由是否充分。
- Conversation and Interaction Agent 从对单个对话孤立的恢复转变为多轮交流，从而可以在不断变化的目标和用户反应下进行评估。
- Multimodal and Vision 例如前面提到的 ARM-Thinker[9] 就是一个多模态集成的工具，可以处理图像，文本等多种模态的数据。

Professional Domains

- Medicine
- Law
- Finance
- Education

本节内容

计算成本与隐私

计算成本主要集中在多个智能体的训练费用和调用上，而且在检测时还要调用大量的工具和外部资源，这些都会增加计算成本。同时，Agent 需要存储大量的中间状态和用户画像，从而可能会引发隐私问题，尤其是在处理敏感数据时，需要确保数据的安全性和隐私保护。

Safety

而 Safety 主要分为三个方面:

- Agent 因为要检测，被允许执行代码，调用外部工具等，这些操作可能会引发安全问题，例如代码注入攻击，恶意工具调用等。例如攻击者可以在文本中隐藏恶意指令，可能会造成实质性的破坏
- 多智能体带来的错误传染，如果一个 LLM 出现了错误，那么这个错误可能会被传递给其他的 LLM，从而导致整个评估过程出现错误。更有甚者，出现对抗式交互，为了赢生成极端内容。
- 最后因为强化学习的关系，Judge Agent 的评分通常作为奖励信号来训练，但是如果 Judge Agent 本身存在偏见或者错误，那么最终训练出来的模型就会为了迎合 Judge Agent 而产生偏差。

本节内容

参考文献 I

Yeshuang Chang, Gorui Wang, Chuan Wang, Yuting Wu, Linyi Zhu, Xiaobi Hao, Kai Yi, Cunxiang Wang, Yidong Wang, Wei Ye, et al.
A survey on evaluation of large language models.
arXiv preprint arXiv:2307.03109, 2023.

Jiaju Chen, Yuxuan Lu, Xiaojie Wang, Huimin Zeng, Jing Huang, Jiri Gesi, Ying Xu, Bingsheng Yao, and Dakuo Wang.
Multi-agent-as-judge: Aligning llm-agent-based automated evaluation with multi-dimensional human evaluation.
arXiv preprint arXiv:2507.21028, 2025.

Mingchen Zhuge, Changsheng Zhao, Dylan Ashley, Wenyi Wang, Dmitrii Khizbullin, Yunyang Xiong, Zechun Liu, Ernie Chang, Raghuraman Krishnamoorthi, Yuandong Tian, et al.
Agent-as-a-judge: Evaluate agents with agents.
arXiv preprint arXiv:2410.10934, 2024.

Manya Wadhwa, Zayne Sprague, Chaitanya Malaviya, Philippe Laban, Junyi Jessy Li, and Greg Durrett.
Evalagent: Discovering implicit evaluation criteria from the web.
arXiv preprint arXiv:2504.15219, 2025.

Azim Ospanov, Zijin Feng, Jiacheng Sun, Haoli Bai, Xin Shen, and Farzan Farnia.
Hermes: Towards efficient and verifiable mathematical reasoning in llms.
arXiv preprint arXiv:2511.18760, 2025.

Anikait Singh, Sheryl Hsu, Kyle Hsu, Eric Mitchell, Stefano Ermon, Tatsunori Hashimoto, Archit Sharma, and Chelsea Finn.
Fspo: Few-shot preference optimization of synthetic preference data in llms elicits effective personalization to real users.
arXiv preprint arXiv:2502.19312, 2025.

Weixiang Zhao, Xingyu Sui, Yulin Hu, Jiahe Guo, Haixiao Liu, Biye Li, Yanyan Zhao, Bing Qin, and Ting Liu.
Teaching language models to evolve with users: Dynamic profile modeling for personalized alignment.
arXiv preprint arXiv:2505.15456, 2025.

参考文献 II

Michael J Ryan, Omar Shaikh, Aditri Bhagirath, Daniel Frees, William Held, and Diyi Yang.

Synthesizeme! inducing persona-guided prompts for personalized reward models in llms.

arXiv preprint arXiv:2506.05598, 2025.

Shengyuan Ding, Xinyu Fang, Ziyu Liu, Yuhang Zang, Yuhang Cao, Xiangyu Zhao, Haodong Duan, Xiaoyi Dong, Jianze Liang, Bin Wang, et al.

Arm-thinker: Reinforcing multimodal generative reward models with agentic tool use and visual reasoning.

arXiv preprint arXiv:2512.05111, 2025.

Hongzhan Lin, Yang Deng, Yuxuan Gu, Wenxuan Zhang, Jing Ma, See-Kiong Ng, and Tat-Seng Chua.

Fact-audit: An adaptive multi-agent framework for dynamic fact-checking evaluation of large language models.

arXiv preprint arXiv:2502.17924, 2025.