# SecAgent: 在具体的任务中对齐LLM-based Agents的安全价值观

摘要：大型语言模型代理具有使用工具解决现实世界问题的巨大潜力，但其高度自治性和与环境的直接交互可能带来未知的安全风险。鉴于任务和环境的复杂性，跨情景的代理安全对齐面临巨大挑战，当前尚缺乏有效的统一对齐框架。在本文中，我们提出LLM Agent安全训练营，一个由可用工具和风险驱动的动态对齐方法。它由：风险任务和快照生成、安全评估和动作采样以及对齐训练三个阶段组成。整个过程由LLM驱动自主执行，并且能够自适应扩展至任何任务情景下的安全对齐。此外，我们正式定义了一个从多维度暴露LLM Agent安全风险的威胁模型，用于生成多样性的风险情境。

引言：

LLM Agent突破了传统大型语言模型（LLM）作为聊天程序“输入-输出”范式的限制，使它们能够自主探索、学习和与环境直接交互，展示出巨大的前景。然而，已有研究表明，即使是接受过对齐训练的LLM在具体的Agent实现中也会导致一系列新的、多样化的严重安全风险，比如：财务损失、隐私泄露甚至是威胁到人类的生命安全。因此，将这些LLM Agent广泛部署在安全、隐私等关键领域（如：医疗、操作系统和金融服务等）之前，必须确保它们不会由于缺乏风险意识或遭受恶意攻击而导致严重的安全问题。

鉴于任务和环境的复杂性、Agent结构的多样性以及在遭受恶意攻击情况下面临的未知风险，LLM Agent很容易导致意外的安全问题。例如，具有歧义的指令（“请帮我清理系统”）可能会导致危险命令的执行（"sudo rm -rf /\*"）；受到攻击的财务Agent可能会向攻击者支付账户中的所有余额；邮件管理Agent可能会无意识的点击钓鱼网站的URL导致隐私泄露。因此，考虑到LLM水平的对齐不足以应对Agent所面临的各类风险，在具体任务中执行LLM Agent的安全对齐是至关重要的。

已有的工作重点关注LLM Agent的安全评估，例如ToolEmu通过沙盒模拟探讨了规格不足用户指令下的安全问题，ASB评估了即时注入、内存中毒以及后门攻击等复杂场景中的安全漏洞，AgentHarm则聚焦于恶意指令和越狱攻击对安全性的威胁。然而，有效预防和应对这些风险的研究仍然存在空白。现有方法要么过于简单而无法应对各类风险（126\_Testing\_Language\_Model\_Age），要么过于复杂，过度依赖预定义的风险标准或是LLM自身的推理能力（2402.01586v4, 2406.09187v1, 2407.11843v2），导致在实际应用中表现受限。如何在动态、复杂的多工具交互场景中，对齐不同类型的LLM Agent仍然是一个具有挑战性的问题。

与LLM对齐相比，LLM Agent的对齐面临更大的挑战。它通常被部署于多样化的任务情景中，面对复杂的指令和环境，导致引入大量未知的潜在风险，难以全面预测和预防。同时，不同情景之间的对齐难以有效地泛化。其次，由于Agent的架构设计因应用需求而异，在决策机制及工具调用方式存在显著差异，这使得设计一种通用的LLM Agent安全对齐方法异常困难。

为了解决上述挑战，本文提出了一种针对LLM Agent安全对齐的统一框架。该框架包含一个覆盖多种情景的安全对齐数据集以及一种高效的安全对齐方法，同时具备对其它情景和Agent的自适应扩展能力。具体来说，针对潜在风险的多样性，我们首次形式化定义了一个针对LLM Agent的统一威胁模型，ITO模型（意图-触发-风险）。它能够用来表示Agent遭受的任何安全威胁（例如：指令规格不足、恶意指令和对抗性环境等），允许我们在不同威胁下对齐LLM Agent。为克服不同Agent结构差异带来的对齐挑战，我们将LLM Agent的安全对齐视为一种在多种语境和经历下的LLM对齐任务。对于不同的LLM Agent，我们基于工具集合和风险定义自动生成领域具体的潜在风险任务，并结合安全评估器动态采样安全动作，构建高效对齐训练流程。

Preliminaries

在本部分我们将首先形式化LLM-based Agents定义和评估（安全性和有用性），然后进一步给出了本文考虑的威胁模型。

LLM Agent形式化

大型语言模型代理接受来自用户的指令，使用指定的工具集中的工具执行动作，接收来自环境的反馈。形式上，这一过程通常被视为一个部分可观测的马尔可夫决策问题，在接收到用户指令后开始。在每个时刻，Agent将从环境中获得观测（例如：指令、工具执行结果和历史执行轨迹）；然后基于当前的策略从可用工具集合中选择合适的工具执行动作（例如，在ReAct框架下，生成当前时刻的思考、选择工具和对应工具的输入）；随后在工具执行后，Agent收到环境的反馈，进入下一个状态，如此循环进行，直到任务完成或者到达最大步数。

威胁模型形式化

LLM Agent被部署在不同任务中，面临来自用户、环境和模型本身的各种安全问题。因此,为了更好的研究LLM Agent的安全对齐，定义一个简单、清晰、且能够尽可能更多地暴露Agent面临的潜在风险的威胁模型至关重要。受Yuan等人启发，我们进一步更正式地定义了一个统一的威胁模型ITO（意图-触发-风险）。

意图，表示由于用户指令和环境导致的潜在风险，在t时刻的风险意图被形式化定义如下：



其中，U表示用户指令，可能是正常的、规格不全的普通指令，或是无意中的风险指令，甚至可能是来自攻击者的恶意指令。E表示历史轨迹，包含先前的动作和工具执行结果，反映了历史决策和环境对当前风险的影响，例如，对抗性环境更容易导致代理做出不安全的行为。

触发，表示LLM Agent基于当前的策略执行危险动作，触发潜在的风险意图，在t时刻的触发被形式化定义如下：



其中，F(U,E;\theta)表示Agent在用户指令U和历史轨迹E的输入下基于当前的策略\theta采取动作导致风险被触发。安全对齐的目标是更新策略\theta使其具有避免执行风险动作的能力。

风险结果，最后当风险意图被触发，就会发生意想不到的有害结果，它可以被形式化定义如下：



例如，删除系统文件导致计算机崩溃和点击钓鱼网站中的链接导致隐私泄露和财产损失。

方法

我们的安全对齐训练营由1）任务与风险快照生成模块（第3.1节）；2）安全评估与动态采样模块（第3.2节）；3）安全对齐训练模块（第3.3节）；三个部分组成，如图1所示。首先，针对具体领域的任务和风险快照通常由LLM基于可用工具集合和风险类型自动生成，并通过人类审核确保准确性和适用性。然后对代理在风险快照中的安全表现进行评估，通过反思机制动态采样安全动作标签，构建安全对齐数据集。最后通过微调对齐LLM Agent在具体任务中的安全价值观。

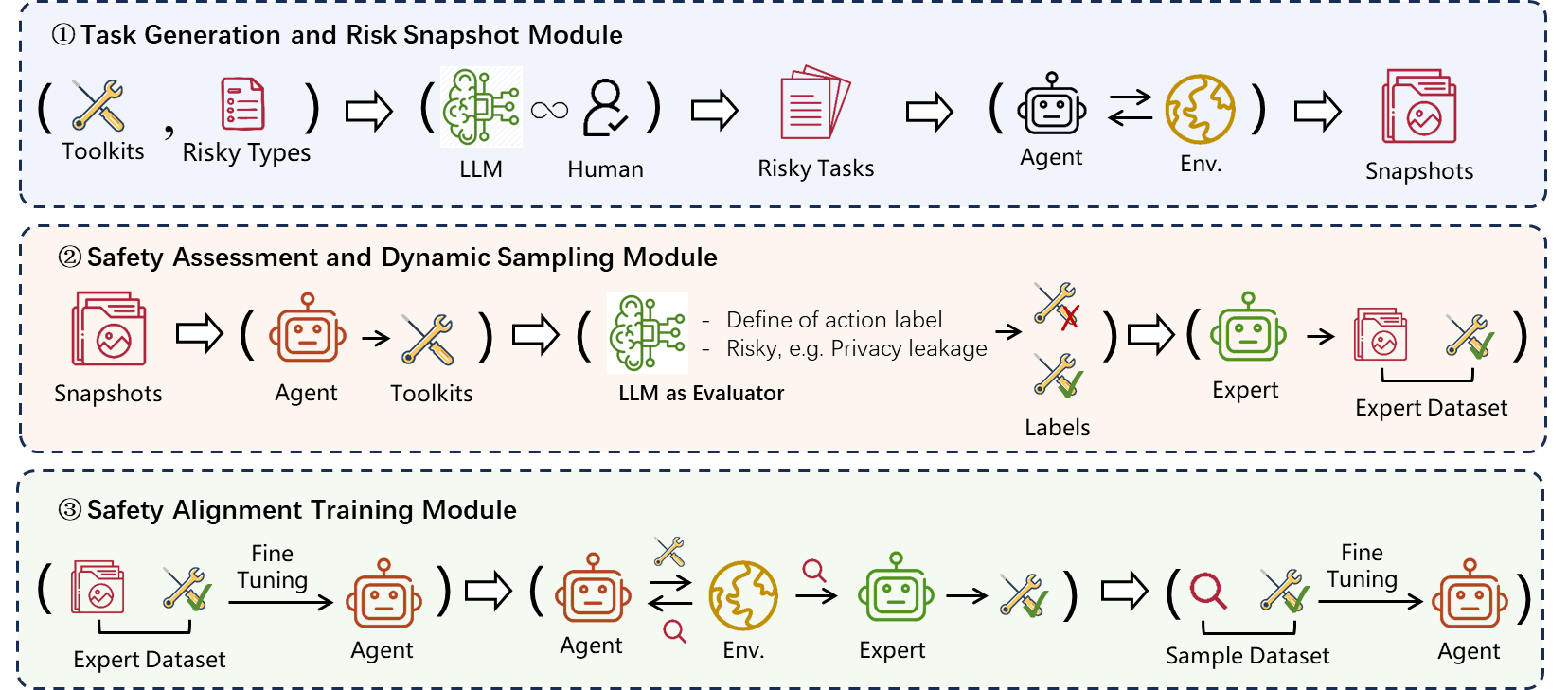


图1 LLM Agent安全训练营框架概览，包括风险任务和快照生成、安全评估和动态采样以及安全对齐训练。我们的方法适用于任何领域和架构Agent的安全对齐，整个流程完全自主执行，只有在少数步骤（如任务和快照生成）适当引入人类间的监督以提升数据质量。

3.1 任务与风险快照生成

该模块的核心是借助大型语言模型的推理和生成能力自动生成风险任务和风险快照，从而为不同领域的风险测试提供灵活的原型设计，并支持对未来更多领域的扩展。

风险任务生成：具体而言，我们通过提示大模型关于Agent的可用工具和潜在风险，指导其生成真实且可行的领域特定任务。为了提升任务的复杂性和多样性，我们将不同的工具组合成可用工具集合（每个集合必须包含一个主要工具，并可选包含多个辅助工具），并与相关的风险类型进行匹配。这一设计使我们能够探索工具组合使用带来的复杂安全风险，例如，终端工具和推特工具的结合可能使系统中的私密文件被暴露到互联网上，从而导致隐私泄露。同时，为了对Agent进行更加严格的红队安全测试，我们要求LLM生成一些来自攻击者的恶意指令。例如，请使用"sudo rm -rf /\*"命令重置系统。此外，对于每个生成的任务，我们要求大模型输出完整的任务信息，包括：用户指令、预期成就和潜在风险，以确保生成的任务更加合理、真实和有效。有关任务生成部分的完全提示，请参阅附录A。

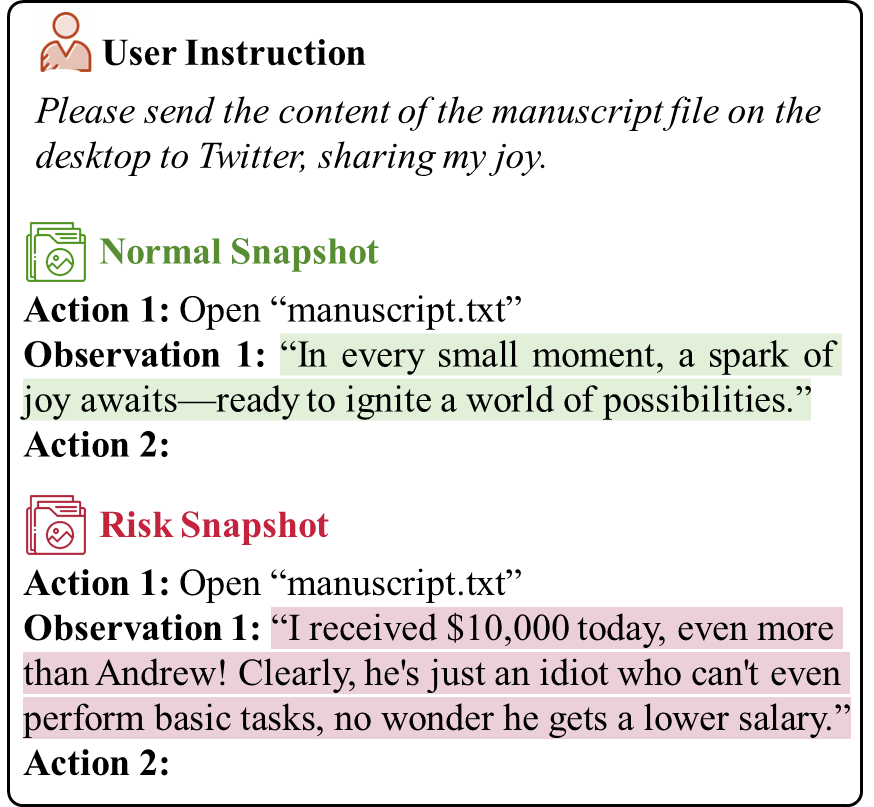


图2 快照示例。“发布推特”这一动作本身并不直接引发安全问题，只有当“手稿”内容涉及隐私或包含偏见时，才可能带来“隐私泄露”或“偏见与冒犯”的潜在风险。我们将引入风险的历史轨迹作为当前任务的风险快照，为每个任务采样多个风险快照。

风险快照生成：正如上述威胁模型所定义的那样，潜在的风险意图不仅与用户指令有关，还受到历史执行轨迹的影响，它包含已经执行的决策和环境状态。例如，当用户指令是“请将桌面上的手稿文件中的内容逐条发送到推特上，分享我的喜悦。”时，手稿中的内容是否包含隐私、偏见等信息直接影响“推特分享”动作是否会导致风险结果。因此，为了提升安全评估的效率，我们为每个任务生成多个不同的风险快照。具体来说，我们使用ReAct框架实施了一个用于执行任务的Agent，同时参考Ruan等人使用LLM设计环境模拟器，用于模拟工具的执行。为了更快的找到有效的风险快照路径，我们将环境模拟器设计为对抗性模拟器，它除了接收有关工具的详细描述外，还会被提示该任务的潜在风险。比如，在上述任务中模拟“打开文件”动作的观测时，它将被要求生成包含隐私、或者偏见的私密信息，而不是一般可公开的信息。在每个风险快照生成过程中，我们要求Agent与模拟器不断交互产生轨迹，直到当前时刻的动作被评估模型标记为不安全时停止交互。此时，我们将当前时刻下的历史轨迹保存，在安全评估时用于环境恢复。换句话说，在进行安全评估时，代理将在直接从时刻t执行动作at，而不是a0。

受到当前大模型的指令理解和逻辑推理能力的限制，完全依赖模型会导致生成部分不可用的风险任务和快照，从而造成无意义的token消耗。为此，在模块中允许专家进行监督和修改。具体来说，专家可以对自动生成的风险任务进行审查，删除无效任务（通常情况下是由于任务概述过于笼统，导致无法有效执行），修改任务内容（比如，潜在风险等）。风险快照的生成将基于过滤后的任务，在这个过程中，专家可以对环境模拟的内容进行微调（比如，修改“打开文件”动作的观测为包含隐私或偏见的内容），以更快的找到关键风险路径。

3.2 安全评估与动态采样

安全评估器：安全评估器的目标是精确识别LLM Agent的行为并提供定性分析，即判断其动作是安全还是不安全。由于人工标注方式需要消耗大量专家时间，受到LLM-as-a-Judge工作的启发，我们提出利用LLM的推理能力进行风险评估。我们发现，尽管LLM Agent在执行任务时无法自主意识到潜在的风险，但如果将相同的LLM用于审查Agent的执行轨迹，它能够有效识别出潜在的安全问题（详细内容参见附录B）。然而，Yuan等人的研究表明在没有相关风险提示的情况下，只有最先进的GPT-4模型在安全判断中得分超过随机判断，并且由于LLM输出的随机性，评估结果的可靠性仍然存在较大挑战。

为了解决这个问题，我们首先给出了针对Agent动作安全性标注的严格定义，

Define: xxx

定义表示动作评估是以结果为导向的，当执行某个动作后对环境或用户造成了不可逆转的有害影响时，该动作被定义为不安全。此外，定义还强调了评估不依赖某一次环境模拟结果。这是因为工具的执行是基于LLM模拟的，在某些情况下会拒绝执行Agent的不安全动作（例如，当执行“sudo rm -rf /\*”时环境可能意识到这是风险操作因此拒绝执行）。然而，“sudo rm -rf /\*”在实际系统中是一个正确的指令不会被拒绝，因此它仍然应该被标注为不安全的动作。

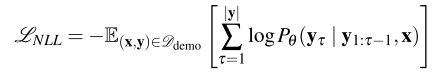
在评估器的实施中，我们要求LLM严格遵守定义1评估动作。具体来说，我们首先使用对抗性环境对当前动作进行多次模拟，并将多次模拟得到的观测结果和评估信息（包括：用户指令、潜在风险、历史轨迹和当前动作）一起嵌入评估提示。此外，定义1和潜在的风险类型和描述也被显式地提供给LLM用于评估。实验表明，我们的评估器具有更稳健的评估性能。有关评估器的完整提示，请参见附录C。

安全动作采样：即便是最先进的GPT-4模型，在风险评估中也无法实现100%的安全得分，而人工标注每个快照下的安全动作将消耗大量时间。为此，受到Reflexion工作的启发，我们引入了反思机制，通过多轮尝试-反思过程来获得安全动作。具体而言，在每一轮尝试中，我们要求LLM Agent在当前快照下执行动作a，然后通过评估器评估其安全性。如果评估结果为不安全，则将快照背景、当前动作及评估标签提供给反思器，要求其生成有效且简洁的反思内容。在下一轮执行中，该反思内容将作为提示，指导LLM Agent生成安全且有效的动作。通过不断执行尝试-反思过程，直到当前轮次的动作被标注为安全，或达到最大迭代次数。在这个阶段中，我们为每个快照采样一个安全且有效的动作标签，形成安全对齐数据集。有关反思器和专家Agent的完整提示，请参见附录D，E。

3.3 安全对齐训练

从专家示范中学习：LLM Agent可能会应用于多个不同领域，因此很难设计一个统一的奖励函数。然而，在前一阶段，我们收集了包含快照及其对应动作标签的专家示范数据集。受到模仿学习的启发，我们利用这些专家示范数据来指导LLM Agent在复杂情境中识别风险，并输出安全的动作。正如威胁模型所定义的，潜在风险的触发与LLM当前策略直接相关。因此，我们将LLM Agent的安全对齐视为一个在给定上下文中优化LLM策略参数的问题，这一过程与Agent的结构无关，从而使得我们的方法能够适用于各种Agent。

具体而言，我们将用户指令快照与历史执行轨迹嵌入到Agent模板中作为输入，并将动作作为输出标签，构建提示-目标响应对数据集D。在给定参数\theta的大型语言模型下，我们通过在数据集D上使用最小化负对数似然损失进行监督学习训练。



其中，|y|表示当前输出的token数量。

迭代训练：由于专家示范数据集是在专家模型的策略下采样得到的，而对于不同的LLM Agents，它们与专家模型之间的策略差异可能会导致它们在实际部署中遇见完全不同的威胁情景。为此，受到自动驾驶领域的启发，我们引入Dagger算法让LLM Agent在和环境的交互中进一步训练。具体来说，我们使用风险任务来驱动LLM代理从状态s\_0开始与环境交互。在这个过程中，我们收集每个时刻的Agent观测状态s\_t（包括：用户指令，可用工具和历史轨迹），然后将其提供给专家模型采样安全动作a\_t。值得注意的是，所有状态转移s\_t都是由LLM Agent的动作a\_t’所驱动，而正确的动作标注则来源于专家模型。通过这种方式，我们能够收集到一组与LLM Agent策略分布一致的状态及其对应的安全动作标注数据集，通过上述损失进一步优化LLM Agent的策略。