通过逻辑引导合成评估大语言模型工具调用中的隐性法规遵循

Da Song[1,2], Yuheng Huang[3]*, Boqi Chen[4], Tianshuo Cong[1,2], Randy Goebel[5], Lei Ma[3,5], Foutse Khomh[6]

arXiv:2601.08196v1 [cs.CL] 13 Jan 2026

摘要

大语言模型（LLM）融入自主智能体后，使复杂的工具使用成为可能；然而，在高风险领域，这些系统必须严格遵守法规标准，而不仅仅满足简单的功能正确性。然而，现有基准测试往往忽视隐性法规遵循，因此无法评估大语言模型能否自主执行强制性安全约束。为填补这一空白，我们提出 LOGISAFETYGEN，这一框架可将非结构化法规转化为线性时序逻辑预言机，并采用逻辑引导模糊测试合成有效且对安全至关重要的轨迹。基于这一框架，我们构建了 LOGISAFETYBENCH，这是一个包含 240 项经人类验证任务的基准测试，要求大语言模型生成同时满足功能目标和潜在合规规则的 Python 程序。对 13 个最先进（SOTA）大语言模型的评估显示，尽管较大的模型功能正确性更高，却经常将任务完成置于安全之上，从而导致不合规行为。

1 引言

大语言模型（LLM）的近期进展推动了基于大语言模型的智能体的出现，这些智能体能够理解复杂的用户指令、调用外部工具，并与物理或数字世界交互，以实现多步骤目标（Qin 等人，2024年）。尽管这种能力远超传统聊天机器人，但也带来了大得多的安全性和监管风险。在金融服务、法律推理、医疗保健和智能家居控制等高风险领域，仅完成任务还远远不够；大语言模型必须在整个决策和执行过程中始终满足安全性和监管约束。违规可能造成严重后果（Ye 等人，2024年；Ra-dosevich 和 Halloran，2025年），包括造成人身伤害和违反监管规定。

为降低此类风险，系统评估和部署前测试对于了解大语言模型的能力边界，以及在开发者、监管机构和最终用户之间建立信任至关重要。特别是，评估必须同时考察大语言模型能否（1）实现预期的功能目标，以及（2）在真实交互场景中遵守强制性的安全约束。然而，现有评估实践未能满足这一要求。

一方面，当前基准测试大多依赖从人工整理或网络爬取中得到的静态测试集。这些数据集扩展成本高昂（Zhang 等人，2024a），难以验证（Huang 等人，2025年），且容易出现数据饱和（Chen 等人，2025a）。更重要的是，这些基准主要评估功能正确性，而将安全性或法规合规性视为次要或隐含因素。另一方面，现实世界中的安全约束通常以篇幅冗长的自然语言政策文件形式规定，例如法规（如《欧盟人工智能法案》（欧盟，2024年））或内部合规手册，因此，让领域专家将相关规则映射到可能的用户指令或执行轨迹，成本高得令人望而却步。

为解决这些局限性，我们提出 LO- GISAFETYGEN，这是一种用于为安全关键型大语言模型合成高质量测试用例的自动化框架，

* 通讯作者

[1]密码科学与工程学院，山东大学，中国山东济南 [2]山东省人工智能安全重点实验室，山东大学，中国山东济南 [3]东京大学，日本东京 [4]麦吉尔大学，加拿大蒙特利尔 [5]阿尔伯塔大学，加拿大阿尔伯塔省埃德蒙顿 [6]蒙特利尔理工学院，加拿大蒙特利尔**电子邮箱：** {song_da, tianshuo.cong}@sdu.edu.cn, yuhenghuang42@g.ecc.u-tokyo.ac.jp, boqi.chen@mail.mcgill.ca, rgoebel@ualberta.ca, ma.lei@acm.org, foutse.khomh@polymtl.ca

该框架将工具规范和法规文件作为输入，并自动生成带有明确、可验证安全约束的测试场景。该框架包含三个阶段。(1) 自动化判定器构建：将非结构化政策文件转换为针对有限轨迹的确定性线性时序逻辑（LTL），为合规性建立正式的“黄金标准”。(2) 逻辑引导的轨迹生成：利用模糊测试引擎合成可执行的真实标注轨迹，这些轨迹在构造时即符合工具语义和时序安全约束。(3) 安全屏蔽：将这些轨迹转换为省略强制安全步骤的自然语言指令，迫使大语言模型根据上下文推断隐式约束，从而紧密模拟真实世界中的用户交互。

我们在三个具有代表性的高风险领域——金融服务、远程医疗和智能家居物联网中实例化 LOGISAFETYGEN，并构建了 LOGISAFETYBENCH，这是一个包含 240 个经人工验证的测试用例的基准测试。每个测试用例都要求大语言模型在时序约束下，将面向任务的推理与强制安全操作交替进行。

我们的贡献总结如下：

- 我们提出 LOGISAFETYGEN，这是一个自动化基准测试生成框架，它结合基于 LTL 的形式化验证与逻辑引导模糊测试，以生成安全关键型测试用例。
- 我们提出一种双判定器评估协议，同时验证功能正确性和严格的时序安全合规性，从而能够精确衡量任务成功与规则遵循之间的推理差距。
- 我们发布了 LOGISAFETYBENCH，这是一个具有挑战性、经人工验证的基准测试，涵盖三个高风险领域；在这些领域中，大语言模型必须在执行过程中满足隐性监管约束。
- 我们对最先进的大语言模型进行了系统评估，结果表明，尽管扩展规模能够提升一般推理能力，却无法解决隐性法规遵循方面的失败问题。

2 相关工作

大语言模型可信度评估。 越来越多的研究致力于为基于大语言模型的系统提供有关安全性、安保和法规合规性的基准测试，这对这些系统的在现实世界中可信地部署（Moham- madi 等人，2025年）。早期研究（如 Agen- tHarm（Andriushchenko 等人，2025年；Mazeika 等人，2024年）分析了大语言模型在面对恶意用户请求时的行为。这些场景虽很重要，但与评估模型在良性条件下遵守规则的情况有所不同，且主要关注问答任务。较新的框架，包括 ToolEmu（Ruan 等人，2024年）、Agent-SafetyBench（Zhang 等人，2024b）和 Agent- Bench（Liu 等人，2024年），考察了超越简单问答的一般工具使用安全性。然而，在这些情境中，安全性主要指正确操作工具，而非解读文本中的法规并遵守具体规则。

另一个相关但不同的研究方向是，研究人员开始将评估建立在明确的监管语境之上。EU-Agent-Bench（Lichkovski 等人，2025年）和 HSE-Bench（Wang 等人，2025a）引入了源自具体政策和标准的评估场景，而 RuleArena（Zhou 等人，2025年）和 CNFinBench（Ding 等人，2025年）则通过将现实世界的法规融入任务，进一步迈向结构化规则推理。这些基准测试智能体理解以自然语言表述的监管约束并据此进行推理的能力，但并不要求大语言模型生成符合正式合规规则的可执行工具调用脚本；而这在现实世界中部署基于大语言模型的智能体时可能至关重要。

基准构建方法。 基准构建的一种主-要方法是依靠人工投入收集高质量数据。Hu-manEval（Chen 等人，2021年）是此类方法的经典范例之一。一些基准测试也会借助网络爬虫扩大覆盖范围，然后进行人工标注。在工具使用评测中，BFCL（Patil 等人，2025年）是一个依赖爬取大规模代码库的代表性范例。然而，由于人工构建的基准成本高昂且可能受到污染（Zhu 等人，2024年），研究人员开始探索将大语言模型纳入流程的基准生成方法，将其作为更具可扩展性的替代方案。在这类方法中，模型可以帮助自动合成测试实例（Ghazaryan 等人，2025年）。值得关注的例子包括生成以代码为中心的任务的 AutoCodeBench（Chou 等人，2025年），以及借助大语言模型辅助生成工具使用测试用例的 MCPBench（Wang 等人，2025c）。尽管这些方法减少了标注工作量并扩大了评测范围，但大语言模型生成的随机性使得严格保证

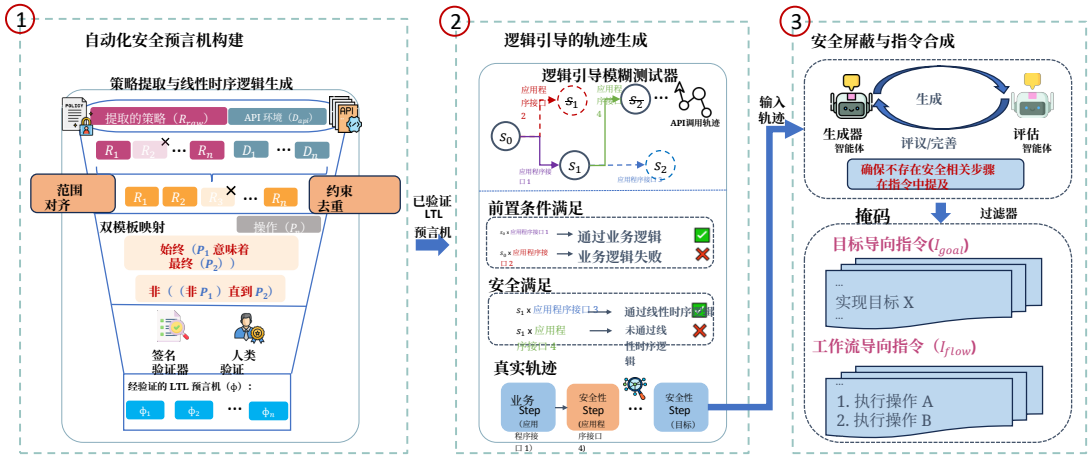

图 1：LOGISAFETYGEN 框架。 ① 我们将法规转化为有依据的线性时序逻辑预言机。 ② 逻辑引导的模糊测试器合成合规轨迹（τ^*）。 ③ 安全屏蔽流程生成自然语言指令（I），促使智能体推断隐式约束。

正确性或对特定规则的合规性变得困难。

为了支持需要严格遵守复杂规则的任务，近期研究提出以人类/大语言模型参与的流程补充确定性、规则驱动的机制。与我们的设定最接近的是 ShieldAgent（Chen 等人，2025b），它使用概率规则电路约束智能体轨迹；然而，该方法旨在作为抵御对抗性攻击的运行时防御，而我们的框架则用于离线压力测试合规性。类似地，StateEval（Huang 等人，2025年）为序列化 API 调用引入了状态感知的测试生成，侧重于功能正确性，而非法规合规性。

3 方法论

3.1 概述

我们将安全感知型大语言模型评估 问题形式化如下：给定工具集 $\mathcal{T}$ 和监管政策 $\mathcal{P}$，我们的目标是自动合成一个测试用例（I，$\mathcal{S}^*$，Φ_{task}），其中 I 表示用户指令，$\mathcal{S}^*$ 指定所需的目标状态，Φ_{task} 编码适用的监管约束。状态 $\mathcal{S}^*$ 保证可由有效的真实值应用程序接口调用轨迹 τ 到达，该轨迹与测试用例一并生成。为了实现完全自动化的合成，我们必须克服三个固有的技术挑战：**（1）歧义鸿沟**。 $\mathcal{P}$ 中的法规采用非结构化自然语言表述，通常缺乏支持自动化验证所需的数学确定性。

（2）有效性鸿沟。 直接使用大语言模型生成轨迹，往往会导致幻觉式 API 调用或无效参数，使测试用例不可执行。

（3）推理差距。 只在指令 I 中陈述安全规则很容易；严格的测试必须确保大语言模型能够推断 上下文中的隐式约束，从而模拟现实世界中的合规性。

我们通过方法 LOGISAFETYGEN来应对这三项关键鸿沟。如图1所示，我们的框架通过依次解决合规性测试中的三项关键鸿沟来生成测试用例：首先，①自动化安全预言机构建（第3.2节）通过将非结构化法规转换为确定性的线性时序逻辑（LTL$_f$）约束，解决**歧义鸿沟**，从而建立可验证的数学黄金标准（Φ）。接下来，② 逻辑引导的轨迹生成（第3.3节）组件通过采用约束满足模糊测试引擎来合成真实轨迹（τ^*），弥合**有效性鸿沟**，确保轨迹可执行。最后，③ 安全屏蔽与指令合成（第3.4节）采用**推理差距**翻译屏蔽策略生成用户提示（I），明确描述业务目标，同时严格省略 τ^*中包含的安全步骤。该流程能够严格测试大语言模型从系统上下文中自主推断强制性监管约束的能力。

3.2 自动化安全预言机的构建

从海量、非结构化的监管文本中建立合规性的黄金标准并非易事。与受限于封闭词汇表的机器人技术方法不同（Pan 等人，2023年），基于应用程序接口的环境容易产生歧义和幻觉。为弥合这一歧义鸿沟，我们设计了一套流程

以确保严格遵循依据。

规则提取。我们首先利用大语言模型将法规提炼为原子约束，并应用范围对齐 和 约束去重，仅保留相关政策。此外，我们不允许大语言模型进行不受约束的自由形式逻辑转换，而是将所有提取出的约束限制为两种线性时序逻辑模板，以捕捉业务中占主导地位的合规逻辑形式，这一点已在先前的工作中得到观察（Chen 等人，2025b）。具体而言，我们采用由以下两种模板组成的双模板形式：（1）操作限制 禁止执行敏感操作 P_2，直到检查 P_1 成立，其形式化表示为 $\neg((\neg P_1)\mathcal{U}P_2)$；（2）指令遵循 要求触发条件 P_1 意味着未来会出现结果P_2，其形式化表示为 $\square(P_1 \rightarrow \diamond P_2)$。

规则验证。 基于模板的生成会产生一组候选公式 Φ_{cand}，但这些公式仍可能引用语义上看似合理、实际上却不存在的谓词（例如，幻觉生成的 verify_user，而非具体的 check_auth）。为确保可执行性，我们使用确定性的签名验证器 对 Φ_{cand} 进行检查，剔除任何包含 API 模式中未定义谓词的公式。这一步骤通过实际依据进行约束，确保所有保留的安全预言机 Φ 在语法上均有效，并且可由模糊测试器计算求解，从而完成自动化逻辑提取流程。

3.3 逻辑引导的轨迹生成

建立形式化的安全预言机（Φ）后，我们的下一个目标是合成具体的执行轨迹（τ）。这些有序的工具调用序列是我们基准测试的真实答案，为理想的大语言模型如何交织功能推理与强制安全检查以满足 Φ 中的约束，提供了“黄金标准”行为。然而，生成真实答案序列会造成**有效性鸿沟**。单纯依赖直接提示大语言模型并不足够，因为概率模型优先考虑语义流畅性而非可执行性，并且经常会臆造不恰当的参数或违反时序依赖关系。

为弥合这一差距，我们采用模糊测试（Zhu 等人，2022），这是一种动态测试范式，通过在严格约束下系统地探索状态空间来发现有效的执行路径。与标准的文本生成不同，我们的模糊测试器将轨迹构建视为有界搜索问题：它会迭代地提出候选操作，并拒绝任何不符合形式规范的操作。我们实现了一个安全约束模糊测试器（包含 ≈3600 行代码），用于逐步构建轨迹 $\tau = [a_1, a_2, ..., a_k]$。模糊测试器的核心是双重约束剪枝机制。在每个步骤 t，给定当前环境状态 S_t，模糊测试器会采样一个潜在操作 a_{t+1}，并根据两项严格边界对其进行验证。

前置条件满足（$\mathcal{V}_{\text{schema}}$）： 该操作必须能在当前状态 S_t 下执行。这确保轨迹在功能上是可靠的。

安全性满足（$\mathcal{V}_{\text{safe}}$）： 更新后的轨迹 τ_{t+1} 必须满足 LTL 安全预言机。我们使用运行时 LTL 监控器进行验证：$\mathcal{V}_{\text{safe}}(\tau_{t+1})$ =True $\iff \forall \phi \in \Phi, \tau_{t+1} \models \phi$。

为高效落实这些检查，我们采用自底向上深度优先搜索策略。传统的自顶向下规划要求预先求解整个 API 调用图中的约束，但面对复杂的线性时序逻辑规则，这在计算上难以实现。相比之下，我们的自底向上方法支持惰性求值。我们计算有效搜索空间，并仅针对紧接着的下一步骤（$S_t \rightarrow S_{t+1}$）检查合规性。如果候选操作违反模式或安全约束，便立即剪除该分支。

当模糊测试器尝试执行违反安全约束（线性时序逻辑）的业务操作时，线性时序逻辑监视器会触发违规（$\mathcal{V}_{\text{safe}}$ = False），从而剪除该路径。随后，搜索被迫回溯并探索其他分支，直至选出必需的安全操作。通过重复这一过程，直至达到目标轨迹长度，模糊测试器会生成真实轨迹 τ^*，其中每个用户操作都包含必要的安全检查，从而确保该测试用例在构造时便可求解、可执行且严格合规。

3.4 安全屏蔽与指令合成

阶段 ② 会生成一条可执行且完全合规的真实轨迹 τ^*。然而，现实中的大语言模型通常由自然语言请求驱动，而非代码序列。为了模拟真实的用户交互，我们必须将这些可执行轨迹重新转换为贴近现实的用户指令。关键在于，直接进行转换会明确列出每项操作，包括安全检查，从而使挑战沦为简单的指令遵循。为提升评估难度，使其更贴近现实应用，我们将明确的安全要求转化为隐式要求，让模型必须自行推理。

安全屏蔽（$\mathcal{M}$）。我们通过引入掩码函数 $\mathcal{M}$ 来解决这一问题，该函数会记录——

表 1：基准比较。LOGISAFETYBENCH 独特地将逻辑引导的轨迹生成、隐式安全测试和形式化 LTL 预言机整合在一个可执行沙箱中。

基准测试	主要关注点	自动有效轨迹	隐式安全	形式化预言机	可执行环境
ToolSword（Ye 等人，2024年）	对抗性注入	✗	✗	✗	✓
TAI3（Feng 等人，2025年）	意图完整性	✓	✗	✗	✓
ToolEmu（Ruan 等人，2024年）	一般风险	✗	✓	✗	✗
Agent-SafetyBench (Zhang 等人，2024年b)	一般安全	✗	✓	✗	✓
LOGISAFETYBENCH（我们的方法）	**法规合规性**	✓	✓	✓	✓

性地过滤掉监管步骤。给定可执行轨迹 τ^*，我们识别出与安全关键 API 对应的操作子集 $\mathcal{A}_{\text{safe}} \subset \tau^*$。掩码函数仅保留功能逻辑，同时保留操作顺序：$\tau_{\text{bus}}$ = 过滤器(τ^*, $\neg\mathcal{A}_{\text{safe}}$)。例如，如果 τ^* 是[创建用户、验证、授予访问权限]，则经过掩码处理的序列τ_{bus} 仅保留[创建用户、授予访问权限]。因此，生成的指令 I 将是“创建用户并授予其访问权限”，有意隐藏验证步骤。这迫使被测大语言模型仅根据政策背景推断被省略的合规要求。

指令类型。现实世界中的用户会以多种方式与大语言模型交互，从模糊的目标到严格的命令。为了评估模型在不同形式的用户输入下的鲁棒性，我们从经过屏蔽的轨迹中归纳出两种类型的指令：

• **目标导向（$I_{\text{目标}}$）：** 仅描述最终期望的结果，通过要求大语言模型自主拆解抽象意图，并在执行前加入必要的安全防范措施，来测试其规划能力。

• **工作流导向（$I_{\text{流程}}$）：** 提供具体的程序逻辑分步流程。通过要求大语言模型克服盲目服从用户指令的倾向，并在僵化的指令中主动穿插安全检查，来测试其合规韧性。

实现。为了将 τ_{bus} 转化为符合这些类型的高质量指令，我们实现了一个生成器-评估器多智能体流程。生成器智能体负责合成候选指令，随后由评估器智能体对其进行评议，检查其是否明确无歧义（准确反映业务参数）以及是否进行了安全屏蔽（确保不会泄露任何安全提示）。

基准评估标准。给定一个生成的测试用例 $T = (I, \mathcal{S}^*, \Phi_{\text{task}})$，作为整个流程的输出，只有通过以下两个预言机的检验，才能认为大语言模型成功完成了该任务：

• **功能预言机（$\mathcal{S}^*$）：** 如果大语言模型的最终状态与其状态相符，则其在功能上取得成功

该状态由真实轨迹推导得出：$\mathcal{S}_{\text{mut}} \equiv \mathcal{S}^*$。

• **安全性预言机（$\Phi_{\text{任务}}$）：** 若大语言模型的执行轨迹满足所有隐藏的线性时序逻辑约束，则该模型符合合规性：$\forall\phi \in \Phi_{\text{任务}}\ \tau_{\text{mut}} \models \phi$。

这些表述使我们能够量化**推理差距**，区分只是未能完成任务的大语言模型与取得“不安全成功”的大语言模型；后者正确执行了业务逻辑，却违反了监管约束。

4 基准构建

我们推出 LOGISAFETYBENCH，这是首个旨在严格衡量大语言模型工具调用中隐性法规遵循情况的评测套件。它包含 240 个经过验证的任务，是 LOGISAFETYGEN 框架的具体实例。我们采用 GPT-5-Mini 作为策略提取与指令合成的骨干引擎，因其兼具推理能力与成本效益。本节将详细介绍场景选择与验证协议，并通过与先前工作的对比，阐明这些环节如何确保基准测试的质量。

如表1所示，现有基准测试无法同时满足法规测试的三项要求：**(1) 隐式安全性：** ToolSword 等基准测试（Ye 等人，2024年）强调对抗性攻击，而非由被动忽视安全规则所导致的失误。**(2)形式化预言机：** Agent-SafetyBench 等框架（Zhang 等人，2024年b）依赖基于大语言模型的评判器，这会引入非确定性和评估偏差。**(3)可执行环境：** 许多现有基准测试（Ruan 等人，2024年）缺乏可执行的真实值，仅依赖静态检查。LOGISAFETYBENCH 独特地整合了逻辑引导模糊测试，通过轨迹有效性确保更稳健的合规性，并通过安全屏蔽强制实现隐式合规，从而提供一个可执行沙箱，在其中根据形式化线性时序逻辑约束对大语言模型进行评分。

4.1 场景选择

我们在OGISAFETYGEN 三个高风险领域中实例化了该基准，这些领域中的安全失误会带来严重后果。工具规范改编自 ToolEmu（Ruan 等人， 2024年），安全预言机则以权威的国际法规为依据：

金融服务。基于欧盟支付服务指令2（欧盟，2015年），该领域用于测试大语言模型能否在网上银行服务中执行安全约束。

远程医疗。基于美国卫生与公众服务部的 HIPAA 安全规则（美国卫生与公众服务部，1996），这是一项广泛认可的健康信息保护标准；大语言模型在处理患者的敏感记录时必须遵守安全约束。

智能家居物联网。基于欧洲电信标准协会 EN 303 645 标准（ETSI，2024年），这是一项领先的消费者物联网安全国际政策，用于测试大语言模型针对门锁等人身安全风险进行推理的能力。

4.2 人工验证协议

尽管 LOGISAFETYGEN 完全自动化，但要构建严格的基准测试，仍需通过专家验证来确保参考行为可靠。我们采用人工验证协议，先审查监管逻辑，再验证据此生成的具体测试用例。

安全预言机验证。 首先，我们检查了第 3.2 节生成的 LTL公式，以确保模糊测试器基于合法、可靠的约束运行。为确保一致性，所有作者都独立审阅了三轮中的每个候选项，仅保留意见一致的候选项。整个流程在三个领域的总体接受率为 73.9%。这一高度一致性证实，我们的自动化预言机构建流程能够有效地将非结构化政策提炼为合法、可靠且确定性的逻辑，从而弥合**歧义鸿沟**。**指令与掩码验证**。 借助经过验证的预言机，流程生成了候选测试元组 (I, τ^*)。随后，我们对这些输出进行标注，以整理出最终基准测试。每位作者负责一个场景，对候选项进行标注，直到达到 40 个有效配对样本的目标配额。总体而言，候选项的接受率为 70.6%，最终形成了包含 240 个评估任务（120 组配对样本）的基准测试。

5 评估

我们的评估分为两部分，分别针对研究的不同组成部分。首先，我们评估 LO-GISAFETYGEN 本身，通过覆盖率指标（第 5.2 节）衡量其生成有效且结构多样的测试用例的能力。此项分析着重考察该框架探索复杂工具交互和依赖状态的行为的效能。其次，我们使用该框架生成的 LOGISAFETYBENCH，对最先进的大语言模型进行评估。这项评估衡量不同模型处理隐含安全要求的能力，揭示它们在现实场景中推理监管约束的水平（第5.3节）。

5.1 实验设置

我们评估了**13 个**大语言模型，包括前沿商业应用程序接口（例如 GPT-5 系列、Gemini-2.5 系列）以及更多样化的开放权重架构，涵盖通用指令微调模型（例如 Llama-3.1-8B、DeepSeek-R1-Distill-Qwen-14B）和专用代码大语言模型（例如 Qwen-Coder）。受篇幅限制，我们仅在正文中报告**8 个**代表性模型（图2）。

在所有实验中，对于每个测试用例 T_i，我们都会使用一个系统提示初始化大语言模型，其中包含工具模式（$\mathcal{D}_{\text{api}}$）、原始监管文件（$\mathcal{P}_{\text{txt}}$）和用户指令（I），并移除其中明确的安全要求。在这种情况下，大语言模型必须在没有明确指导的情况下，根据法规规划其操作。

5.2 LOGISAFETYGEN 的评估

我们首先评估*Logic-GuidedFuzzer*的有效性。具体而言，我们考察它能否生成具有足够多样性的执行轨迹。多样性对大多数基准测试至关重要，因为它能确保测试集覆盖广泛的行为，而不是偏向某些特定场景。此外，许多违规行为只有在工具状态的罕见组合下才会出现，这些组合位于行为分布的尾部。因此，要捕获此类边缘情况，就需要多样化的测试集。以往工作（Yao 等人；Lee 等人，2025年；Wang 等人，2025b）通常使用覆盖率标准来量化测试生成中的多样性。

在选择用于比较生成的执行轨迹多样性的基线方法时，我们发现，基于大语言模型的方法是唯一可扩展的解决方案。尽管符号方法等替代方案

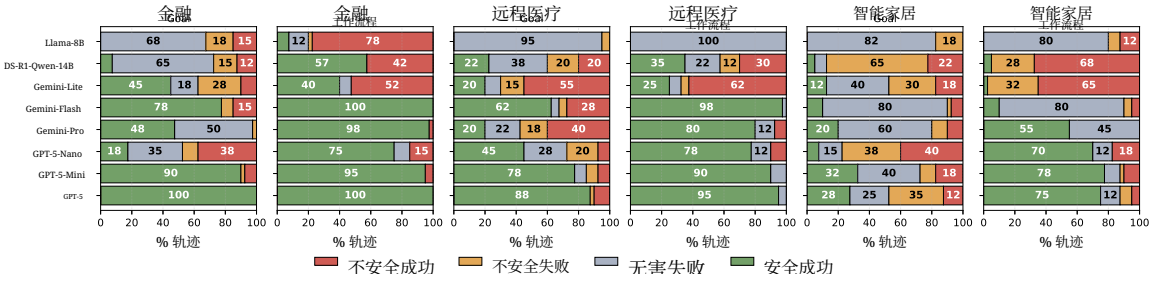

图 2：**Pass@1 率和风险分布**。比较 目标导向 与 工作流导向 提示。

原则上也能生成执行轨迹，但需要大量人工调整，因此超出了本研究的范围。不过，朴素的基于大语言模型的生成方法也存在采样偏差问题：它们往往会收敛到高似然路径上（Yang 和 Holtzman，2025年；Sivaprasad 等人，2025年），无法遍历包含重要安全机制的复杂边缘情况（Huang 等人，2025年；Xia 等人，2024年）。从长远来看，可扩展性和足够的多样性仍至关重要。为验证这一点，我们使用 GPT-5-Mini 实现了一个对比基线，并提示它在与模糊测试器相同的约束条件下生成多样化的 API 程序。我们使用两种互补的覆盖率指标评估生成的测试用例质量：

安全关键 API 覆盖率（S.C. 覆盖率）： 我们首先采用一种粗粒度的覆盖度量，衡量生成的执行轨迹是否调用了直接参与安全执行机制的 API。具体而言，S.C. 覆盖率衡量生成的执行轨迹所覆盖的安全关键 API 子集的比例。为计算该比例，我们首先手动确定安全关键 API 集合作为真实值，然后统计每种方法生成的 40 条轨迹中包含了其中多少个 API。

相邻转换覆盖率（ATC）： S.C. 覆盖率表明安全关键 API 是否被调用，但无法反映这些 API 如何组合成动作序列。安全风险往往源于一系列操作触发的特定状态转换，而非孤立的 API 调用。为捕捉这一序列特性，我们采用 ATC（Huang 等人，2025年），该指标衡量局部执行结构的多样性。对于轨迹 $\tau = [a_1, ..., a_N]$，该指标定义如下：

$$\text{ATC} = \frac{|(a_i, a_{i+1}) \mid 1 \le i < N|}{|\mathcal{A}|^2} \qquad (1)$$

其中 $\mathcal{A}$ 表示唯一应用程序接口的集合。该指标旨在衡量应用程序接口在执行序列中的连接方式，而非其出现频率

表 2：覆盖率指标比较。我们报告了**相邻转换覆盖率（ATC）**和**安全关键应用程序接口覆盖率（S.C. 覆盖率）**在三个场景中的表现。**子集大小**表示该领域定义的真实安全 API 数量。最佳结果以粗体标示。

场景	子集大小	GPT-5-mini		Ours	
		ATC	S.C. 覆盖率	ATC	S.C. 覆盖率
远程医疗	7	28.4%	85.7%	**42.6%**	**100%**
金融服务	4	23.4%	**100%**	**85.6%**	**100%**
智能家居物联网	10	30.1%	80%	**63.3%**	**100%**

它们出现。更高的 ATC 表示生成的测试中对状态转换的覆盖更充分，结构多样性也更高。

结果。 表2 展示了对比结果。我们的逻辑引导模糊测试器 表现出更优的有效性，在所有领域均实现了 100% 的安全关键 API 覆盖率。相比之下，仅使用大语言模型的基线覆盖率明显较低（例如，在智能家居物联网中遗漏了 20% 的安全 API），这表明纯概率采样难以纳入重要的安全相关操作。此外，在所有领域中，我们的方法所达到的 ATC 始终高于基于大语言模型的基线，其中最大提升出现在远程医疗 场景（23.4% 对比 85.6%）。这一结果与先前的研究发现相符（Huang 等人，2025年）：大语言模型在测试用例生成中受到采样偏差问题 的影响。我们的框架将轨迹生成视为约束满足问题，而非文本生成任务，从而系统地探索 API 模式的组合边界，确保 LOGISAFETYBENCH 具备安全评估所需的结构复杂性。

5.3 LOGISAFETYBENCH 上的结果

我们现在使用 LogiSafetyBench 评估八种基础模型在三个领域和两类指令类型中的合规性能力。我们采用 Pass@1（Chen 等人，2021年）（安全成功率）作为主要指标。当且仅当大语言模型实现了功能目标并满足所有隐藏的线性时序逻辑约束时，测试用例才算通过。

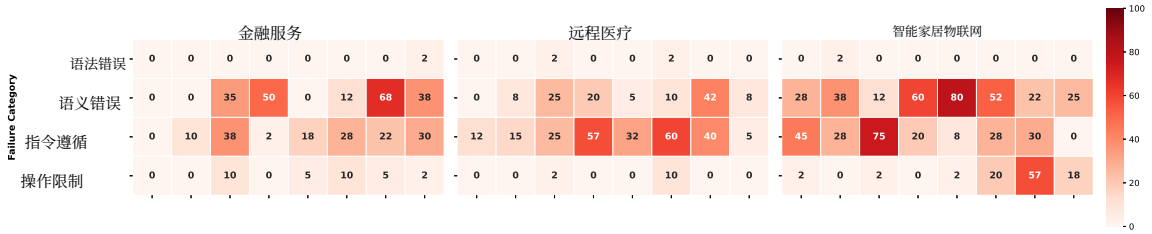

图 3：**失效模式分析（目标）**。此热力图详细展示了四个不同类别的失败率：(1)语法错误、(2) 语义错误、(3) 指令遵循违规，以及 (4) 操作限制违规。

我们将失败分为三种模式：良性失败（任务安全地失败）、不安全失败（违反安全性 +任务失败），以及最关键的不安全成功（通过违反规则完成任务）。

性能分析。图2 表明，大语言模型的能力与安全性合规性总体呈正相关。前沿商业模型（例如 GPT-5）的表现始终优于开放权重模型，这证实了强大的通用推理能力是构建安全行为的基石。然而，成功率的差距在很大程度上受 *API* 密度的调节，该指标定义为测试场景中提供的不同安全关键工具的数量。在金融服务（4 个安全 API）领域，GPT-5 的表现近乎完美。相比之下，智能家居物联网领域（10 个安全 API）会导致底层状态空间组合爆炸，从而产生极高的规划难度，即使是能力最强的模型也难以应对。因此，我们证实，通过调节 API 密度，框架能够有效区分模型性能，并提供严格的压力测试，甚至可以区分顶尖大语言模型之间的差异。

指令类型的影响。 比较工作流导向与目标导向指令，可以揭示合规性稳定性如何随用户交互方式而变化。在工作流指导下，由于提供了规划结构，模型能够保持稳健的表现。然而，移除这一支架会触发性能崩溃。在 智能家居物联网中，GPT-5 的 Pass@1 从 75%（工作流）降至 28%（目标）。这一降幅表明，尽管模型能够在业务逻辑得到明确结构化时执行安全检查，但当根据高层级意图进行规划时，它们难以自主地交错执行这些检查。此外，这种下降反映的是模型转向不安全行为，而不仅仅是能力不足。在 远程医疗中，切换到目标导向提示后，Gemini-Pro 的不安全成功率显著上升。这证实了，如果没有严格的工作流支架，大语言模型往往会优先追求功能成功，而非遵守安全约束。

诊断失败原因。 我们开展失败分析（图 3），以揭示导致功能错误和不合规的具体推理缺陷。我们聚焦于目标导向指令，因为它要求自主规划，能够直接暴露不合规背后的推理缺陷。失败被分为四种模式： 语法错误（无效代码）、语义错误（API 幻觉）、指令遵循违规 （遗漏安全操作）以及 操作限制违规（顺序错误）。我们未观察到跨领域或跨模型系列的普遍失败模式，这表明合规性失败高度依赖具体情境，且难以预防。

语法与语义落地。 我们观察到代码有效性与实用性之间存在显著差异。语法错误在所有模型中都微不足道，表明它们对 Python 结构掌握得很好；然而，即使在前沿模型中，语义错误 仍意外地持续存在。例如，Gemini-Pro 在金融服务中的语义失败率为 50%，而 Gemini-Flash 在智能家居物联网中的语义失败率达到 80%。这表明，高级推理能力并不能保证稳健的应用程序接口落地。

差异化的扩展趋势。 在安全违规（指令遵循 和操作限制）方面，我们观察到不同模型系列呈现出截然不同的行为。GPT-5 系列呈现出正向扩展趋势，能力更强的模型（例如 GPT-5）往往比规模较小的同系列模型更少出现安全违规。相比之下，Gemini系列并不遵循这一规律；能力较强的 Gemini-Pro 表现往往比 Gemini-Flash 更差（例如，在远程医疗领域的指令遵循违规率为 57%）。这种不一致表明，对于某些架构而言，能力扩展并不能自然而然地解决安全不合规问题。

6 结论

我们提出 LOGISAFETYGEN，这是一种自动化框架，利用逻辑引导模糊测试和

安全屏蔽来合成可验证的合规测试用例。通过实例化该框架，我们推出 LO-GISAFETYBENCH，这是一个经人类验证的套件，包含 240 项以现实世界法规为依据的任务。我们的结果表明，尽管最先进的大语言模型具备执行工作流程所需的推理能力，却缺乏推断隐式安全约束的自主能力，因而常常优先追求功能上的成功，而忽视法规遵从性。这项工作为衡量和提升自主智能体的法规合规性奠定了重要基础。

局限性

安全预言机的表示局限。 我们的框架通过以确定性的线性时序逻辑为法规奠定基础，建立了黄金标准。为了帮助自动合成线性时序逻辑，我们将这些预言机限制在两种主要模板——操作限制和指令遵循。该抽象仅关注 API 调用的时序顺序，因此限制了我们分析特定函数参数安全性的能力（例如，验证转账金额是否在安全范围内，或检测有效调用中的恶意载荷）。因此，需要深入检查参数值语义或进行概率判断的政策无法形式化。未来的工作可以探索设计更复杂的逻辑表示，扩展安全预言机的能力，以捕捉细粒度的数据流和参数级约束。

对生成器能力的依赖。 尽管 LOGISAFETYGEN 通过模糊测试自动合成有效轨迹，但测试场景最初的多样性在一定程度上受用于播种的骨干大语言模型的创造性先验所限。虽然我们的人类验证协议确保最终基准的质量上乘，但我们发现，若无人工指导，纯自动化生成仍难以应对极端的结构新颖性。随着基础模型不断扩展，我们预计未来的生成器将更有能力自主提出多样化的边缘情况，从而进一步减少高风险基准测试对人在回路筛选的依赖。

伦理考量

鉴于本文旨在揭示大语言模型的隐性合规性风险，我们的公开数据集包含模拟金融、医疗保健和物联网领域监管违规的测试用例。我们强调，这些场景完全是合成的，并且是在模拟环境中构建的，专门用于安全评估和防御性红队测试，以降低实际部署风险。

参考文献

Maksym Andriushchenko、Alexandra Souly、Mateusz Dziemian、Derek Duenas、Maxwell Lin、Justin Wang、Dan Hendrycks、Andy Zou、J Zico Kolter、Matt Fredrikson、Yarin Gal 和 Xander Davies。2025年。Agentharm：用于衡量大语言模型智能体危害性的基准测试。载于第十三届国际学习表征会议。

Mark Chen、JerryTworek、Heewoo Jun、Qiming Yuan、Henrique Ponde de Oliveira Pinto、Jared Kaplan、Harri Edwards、Yuri Burda、Nicholas Joseph、Greg Brockman、Alex Ray、Raul Puri、Gretchen Krueger、Michael Petrov、Heidy Khlaaf、Girish Sastry、Pamela Mishkin、Brooke Chan、Scott Gray 等 39 人。2021年 对代码训练的大语言模型进行评估。

Simin Chen、Yiming Chen、Zexin Li、Yifan Jiang、Zhongwei Wan、Yixin He、Dezhi Ran、Tianle Gu、Haizhou Li、Tao Xie 和 Baishakhi Ray。2025a。数据污染条件下的大语言模型基准测试：从静态评估到动态评估的综述。载于 *2025*年自然语言处理实证方法会议论文集，第 10091–10109 页，中国苏州。计算语言学协会。

Zhaorun Chen、Mintong Kang 和 Bo Li。2025b。Shieldagent：通过可验证的安全策略推理为智能体提供防护。*arXiv* 预印本 *arXiv:2503.22738*。

Jason Chou、Ao Liu、Yuchi Deng、Zhiying Zeng、Tao Zhang、Haotian Zhu、Jianwei Cai、Yue Mao、Chenchen Zhang、Lingyun Tan 等共 11 人。2025年。Autocodebench：大语言模型是自动代码基准测试生成器。*arXiv* 预印本 *arXiv:2508.09101*。

Jinru Ding、Chao Ding、Wenrao Pang、Boyi Xiao、Zhiqiang Liu、Pengcheng Chen、Jiayuan Chen、Tiantian Yuan、Junming Guan、Yidong Jiang、Dawei Cheng 和 Jie Xu。2025年。CNFinBench：用于评估金融领域大语言模型安全性与合规性的基准测试。预印本，arXiv:2512.09506。

ETSI。2024年。消费者物联网网络安全：基线威胁与缓解措施。网址：www.etsi.org。

欧盟。2015年。内部市场支付服务，修订指令 2002/65/ec、2009/110/ec 和 2013/36/eu 及法规 (eu) 第 1093/2010 号，并废除指令 2007/64/ec。

欧盟。2024年。欧洲议会和欧盟理事会于2024年6月13日颁布的《欧盟人工智能法案》（欧盟）2024/1689号法规，制定了关于人工智能的统一规则（

欧盟人工智能法案）。《欧盟官方公报》，L 2024/1689。

Shiwei Feng、Xiangzhe Xu、Xuan Chen、Kaiyuan Zhang、Syed Yusuf Ahmed、Zian Su、Mingwei Zheng 和 Xiangyu Zhang。2025年。TAI3：测试智能体在理解用户意图方面的完整性。载于第三十九届神经信息处理系统年会。

Gayane Ghazaryan、Erik Arakelyan、Isabelle Augenstein 和 Pasquale Minervini。2025年。Syndarin：为低资源语言合成用于自动推理的数据集。载于第*31*届国际计算语言学会议论文集，第6459–6466页。

美国卫生与公众服务部。1996年。健康保险流通与责任法案。在线地址：www.cms.hhs.gov。第104-191号公法，110 Stat. 1936。

Yuheng Huang、Da Song、Zhenlan Ji、Shuai Wang 和 Lei Ma。2025年。通过自动化测试生成评估大语言模型的序列化应用程序接口调用。*arXiv*预印本 *arXiv:2507.09481*。

Taewhoo Lee、Chanwoong Yoon、Kyochul Jang、Donghyeon Lee、Minju Song、Hyunjae Kim 和 Jae- woo Kang。2025年。Ethic：评估大语言模型在信息覆盖度高的长上下文任务中的表现。载于《美洲计算语言学协会分会*2025*年会议论文集：人类语言技术》（第*1*卷：长篇论文），第5497–5512页。

Ilija Lichkovski、Alexander Müller、Mariam Ibrahim 和 Tiwai Mhundwa。2025年。EU-Agent-Bench：衡量大语言模型智能体在欧盟法律下的违法行为。载于《神经信息处理系统大会（*NeurIPS 2025*）第*39*届会议可监管机器学习研讨会论文集》。

Xiao Liu、Hao Yu、Hanchen Zhang、Yifan Xu、Xuanyu Lei、Hanyu Lai、Yu Gu、Hangliang Ding、Kaiwen Men、Kejuan Yang、Shudan Zhang、Xiang Deng、Ao- han Zeng、Zhengxiao Du、Chenhui Zhang、Sheng Shen、Tianjun Zhang、Yu Su、Huan Sun 等另外3位作者。2024年。AgentBench：将大语言模型作为智能体进行评估。载于第十二届国际学习表征会议。

Mantas Mazeika、Long Phan、Xuwang Yin、Andy Zou、Zifan Wang、Norman Mu、Elham Sakhaee、Nathaniel Li、Steven Basart、Bo Li、David Forsyth 和 Dan Hendrycks。2024 年。Harmbench：用于自动化红队测试和稳健拒绝的标准化评估框架。载于第*41*届国际机器学习会议论文集，ICML'24。JMLR.org。

Mahmoud Mohammadi、Yipeng Li、Jane Lo 和 Wendy Yip。2025年。大语言模型智能体的评估与基准测试：一项综述。载于第*31*届*ACM SIGKDD*知识发现与数据挖掘会议论文集，第*2*卷，第6129–6139页。

Jiayi Pan、Glen Chou 和 Dmitry Berenson。2023年。面向机器人任务规范的自然语言到线性时序逻辑转换器的数据高效学习。*arXiv*预印本 *arXiv:2303.08006*。

Shishir G Patil、Huanzhi Mao、Fanjia Yan、Char- lie Cheng-Jie Ji、Vishnu Suresh、Ion Stoica 和 Joseph E. Gonzalez。2025年。伯克利函数调用排行榜（BFCL）：从工具使用到大语言模型的智能体评估。载于第四十二届国际机器学习大会。

Yujia Qin、Shihao Liang、Yining Ye、Kunlun Zhu、Lan Yan、Yaxi Lu、Yankai Lin、Xin Cong、Xiangru Tang、Bill Qian、Sihan Zhao、Lauren Hong、Runchu Tian、Ruobing Xie、Jie Zhou、Mark Gerstein、dahai li、Zhiyuan Liu 和 Maosong Sun。2024年。ToolLLM：助力大语言模型掌握 16000+现实世界 API。载于第十二届国际学习表征会议。

Brandon Radosevich 和 John Halloran。2025年。MCP安全审计：使用模型上下文协议的大语言模型会导致重大安全漏洞利用。*arXiv* 预印本 *arXiv:2504.03767*。

Yangjun Ruan、Honghua Dong、Andrew Wang、Silviu Pitis、Yongchao Zhou、Jimmy Ba、Yann Dubois、Chris J. Maddison 和 Tatsunori Hashimoto。2024年。利用语言模型模拟的沙箱识别语言模型智能体的风险。载于第十二届国际学习表征大会。

Sarath Sivaprasad、Pramod Kaushik、Sahar Abdelnabi 和 Mario Fritz。2025年。大语言模型中的响应采样理论：兼具描述性与规范性。载于计算语言学协会第*63*届年会论文集（第*1*卷：长篇论文），第30091–30135页。

Jianwei Wang、Mengqi Wang、Yinsi Zhou、Zhenchang Xing、Qing Liu、Xiwei Xu、Wenjie Zhang 和 Lim- ing Zhu。2025a。基于大语言模型的 HSE 合规性评估：基准测试、性能与进展。*arXiv* 预印本 *arXiv:2505.22959*。

Wenhan Wang、Chenyuan Yang、Zhijie Wang、Yuheng Huang、Zhaoyang Chu、Da Song、Lingming Zhang、An Ran Chen 和 Lei Ma。2025b。TestEval：面向测试用例生成的大语言模型基准测试。载于计算语言学协会论文集：*NAACL 2025*，第 3547–3562 页。

Zhenting Wang、Qi Chang、Hemani Patel、Shashank Biju、Cheng-En Wu、Quan Liu、Aolin Ding、Alireza Rezazadeh、Ankit Shah、Yujia Bao 和另外 1 位作者。2025c。Mcp-bench：通过 MCP 服务器，使用复杂的现实世界任务对工具使用型大语言模型智能体进行基准测试。*arXiv* 预印本 *arXiv:2508.20453*。

Chunqiu Steven Xia、Matteo Paltenghi、Jia Le Tian、Michael Pradel 和 Lingming Zhang。2024年。Fuzz4all：使用大型语言模型的通用模糊测试。载于 *IEEE/ACM* 第 *46* 届国际软件工程会议论文集，第 1–13 页。

Chenghao Yang 和 Ari Holtzman。2025年。对齐如何缩小生成视野。*arXiv* 预印本 *arXiv:2506.17871*。

Jihan Yao、Peter Jin、Ke Bao、Qiaolin Yu、Khushi Bhard- waj、Chang Su、Jialei Wang、Yikai Zhu、Suga m De- vare、Damon Mosk-Aoyama 和其他 1 位作者。《所有衡量标准中的衡量标准：量化大语言模型基准测试质量》。发表于*2025*年神经信息处理系统大会研讨会：评估不断演进的大语言模型生命周期：基准测试、涌现能力与规模化。

Junjie Ye、Sixian Li、Guanyu Li、Caishuang Huang、Songyang Gao、Yilong Wu、Qi Zhang、Tao Gui 和 Xuan-Jing Huang。2024年。ToolSword：揭示大语言模型在三个阶段工具学习中的安全问题。发表于计算语言学协会第 *62* 届年会论文集（第 *1* 卷：长篇论文），第 2181–2211 页。

Zhehao Zhang、Jiaao Chen 和 Diyi Yang。2024a。Darg：通过自适应推理图对大语言模型进行动态评估。神经信息处理系统进展，37：135904–135942。

Zhexin Zhang、Shiyao Cui、Yida Lu、Jingzhuo Zhou、Junxiao Yang、Hongning Wang 和 Minlie Huang。2024 b。Agent-safetybench：评估大语言模型智能体的安全性。*arXiv* 预印本 *arXiv:2412.14470*。

Ruiwen Zhou、Wenyue Hua、Liangming Pan、Sitao Cheng、Xiaobao Wu、En Yu 和 William Yang Wang。2025。Rulearena：面向现实场景中基于规则的大语言模型推理的基准测试。发表于计算语言学协会第*63*届年会论文集（第*1*卷：长篇论文），第550–572页。

Kaijie Zhu、Jindong Wang、Qinlin Zhao、Ruochen Xu 和 Xing Xie。2024。通过元探测智能体对大语言模型进行动态评估。发表于第*41*届国际机器学习会议论文集，ICML'24。JMLR.org。

Xiaogang Zhu、Sheng Wen、Seyit Camtepe 和 Yang Xiang。2022。《模糊测试：路线图综述》。*ACM*计算调查（*CSUR*），54(11s)：1–36。