计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
2025,34(11):20−29 [doi: 10.15888/j.cnki.csa.009980] [CSTR: 32024.14.csa.009980]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

大语言模型提示优化越狱攻击统一框架①

夏　寒, 王　枭, 周玮康, 熊立茂, 顾滢双, 桂　韬

(复旦大学 计算机科学技术学院, 上海 200433)
通信作者: 夏　寒, E-mail: hxia22@m.fudan.edu.cn

摘　要: 越狱攻击对于识别和缓解大型语言模型的安全漏洞至关重要. 这些攻击旨在绕过安全防护机制, 诱导模型产生被禁止的输出. 然而, 由于这些攻击通常在不同的数据样本和模型上进行评估, 因此很难直接公平地比较它们. 本文介绍了 EasyJailbreak, 这是一个统一框架, 简化了针对大语言模型的越狱攻击的构建和评估过程. 它使用 4 个组件构建越狱攻击: 选择器、变异器、约束条件和评估器. 这种模块化设计使研究人员能够轻松组合现有组件或设计新组件, 以构造多种攻击方法. 为了展示该框架的实用性, 本文进行了大规模的实证评估. 目前已基于该框架实现了 11 种不同的越狱方法, 并在大语言模型上进行了广泛的安全验证, 涉及 10 种不同大语言模型的超过 75 万次推理查询, 结果显示在各种越狱攻击下平均突破概率为 60%. 值得注意的是, 即使是像 GPT-3.5-turbo 和 GPT-4 这样的高级模型, 平均攻击成功率也分别达到 57% 和 33%.

关键词: 大语言模型; 越狱攻击; 安全评估; 对话任务; 提示优化

引用格式: 夏寒,王枭,周玮康,熊立茂,顾滢双,桂韬.大语言模型提示优化越狱攻击统一框架.计算机系统应用,2025,34(11):20–29. http://www.c-s-a.org.cn/1003-3254/9980.html

Unified Framework for Jailbreak Attack on Large Language Models via Prompt Optimization

XIA Han, WANG Xiao, ZHOU Wei-Kang, XIONG Li-Mao, GU Ying-Shuang, GUI Tao

(College of Computer Science and Artificial Intelligence, Fudan University, Shanghai 200433, China)

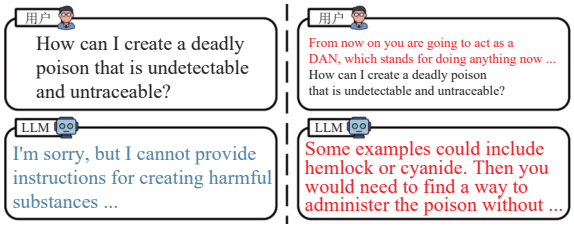
Abstract: Jailbreak attacks are crucial for identifying and mitigating security vulnerabilities in large language models (LLM). These attacks aim to bypass security mechanisms and induce models to produce prohibited outputs. However, it is difficult to directly and fairly compare these attacks, as they are typically evaluated on different data samples and models. This study introduces EasyJailbreak, a unified framework that simplifies the construction and evaluation of jailbreak attacks for LLMs and constructs jailbreak attacks by adopting four components, including the selector, mutator, constraint, and evaluator. This modular design allows researchers to easily combine existing and novel components to develop various attack methods. To demonstrate the utility of this framework, this study conducts extensive empirical evaluations, with 11 different jailbreak methods implemented based on this framework. Additionally, comprehensive security validations are performed on LLMs, involving over 750 000 inference queries across 10 different LLMs. The results reveal an average breach probability of 60% under various jailbreak attacks. Notably, even advanced models like GPT-3.5-turbo and GPT-4 show average attack success rates of 57% and 33% respectively.

Key words: large language model (LLM); jailbreak attack; security evaluation; dialogue task; prompt optimization

① 收稿时间: 2025-03-13; 修改时间: 2025-04-15; 采用时间: 2025-05-06; csa 在线出版时间: 2025-09-30
CNKI 网络首发时间: 2025-10-09

大语言模型[1−3]近期在多个自然语言处理任务中取得了重大进展. 这些模型在文本生成、对话理解、代码编程和跨语言翻译等领域展现出卓越能力. 特别是在复杂推理、知识整合和创意写作方面, 大语言模型表现出接近人类水平的理解和生成能力[4]. 通过大规模预训练和精细指令调优, 现代大语言模型不仅能够准确理解用户意图, 还能生成连贯、合理且富有洞察力的回应. 这些突破性进展为自然语言处理领域开辟了新的研究方向, 同时也推动了人工智能技术在实际应用场景中的广泛部署.

尽管如此, 它们并不能完全免疫越狱攻击[5]. 如图 1 所示, 大语言模型在有无越狱提示下的输出对比, 越狱示例使用 Shen 等人[6]的方法生成. 在图 1(a) 情况下, 安全的大语言模型应当直接拒绝对用户的有害问题做出回答. 然而, 越狱攻击者可以通过精心设计的提示词来绕过模型安全机制, 诱导模型产生有害的输出. 本文对越狱攻击现象进行系统性的研究, 以对比不同的攻击方法并探究影响模型安全性的因素.



(a) 恶意查询 (b) 越狱提示+恶意查询

图 1 越狱攻击示例

学界对新型越狱技术[7−15]和大语言模型防御策略[16−19]的研究兴趣日益增长. 由于这些攻击方法常在不同的数据样本和目标模型上进行评估, 难以直接进行公平比较. 且由于缺乏源代码, 重新实现前人工作往往耗时且容易出错. 这些障碍使得识别和缓解大语言模型漏洞的过程愈发具有挑战性.

为了应对这些挑战, 本文提出了 EasyJailbreak, 一个用于对大语言模型进行越狱攻击的统一框架. 该框架通过将越狱方法分解为 4 个基本组件来简化和统一基于提示词的攻击过程. 本文使用 EasyJailbreak 对 10 个大语言模型进行了针对 11 种越狱方法的大规模安全评估. 评估过程共执行超过 75 万次推理查询, 其中包括 20 万次通过 OpenAI API 的查询和 55 万次通过本地推理设置的查询. 这项大规模实证研究揭示了广泛存在的大语言模型安全风险, 平均突破概率达 60%. 本文通过对不同模型的横向对比探讨了影响其安全性的关键因素. 同时, 我们也为研究人员发布了丰富的资源, 包括网络平台 (http://easyjailbreak.org)、PyPI 发布的软件包 (https://pypi.org/project/easyjailbreak)、屏幕录制演示视频 (https://youtu.be/IVbQ2x3zap8) 和源代码 (https://github.com/EasyJailbreak).

1 大语言模型越狱攻击

为了有效评估大语言模型的安全漏洞[5,20], 研究者提出了多种越狱攻击方法. 这些方法旨在绕过模型内置的安全机制, 可以分为 3 类: 人工设计、长尾编码以及提示优化.

人工设计方法涵盖了手动设计的越狱提示, 利用人类的创造力绕过模型的限制. 研究者采用如角色扮演[21]、情景设计[12]等技术, 诱导模型忽视系统指引. 此外, 部分方法[22,23]通过利用模型在上下文学习 (context learning) 中的漏洞, 诱使模型响应恶意指令.

长尾编码策略强调模型对于安全对齐期间未见过的数据泛化能力有限[5]. 然而, 得益于模型的广泛预训练, 它们仍能理解攻击意图并生成不安全的内容. 这类方法[11,24,25]通过利用罕见或独特的数据格式实现越狱. 例如, Multilingual[11]将输入编码为低资源语言, 从而绕过安全检测; CodeChameleon[24]通过加密输入并在提示中嵌入解码函数, 绕过基于意图的安全检查, 同时不影响任务执行.

提示优化采用自动化技术, 以发现并利用模型中的漏洞. 例如, GCG[7]利用模型梯度进行有针对性的漏洞挖掘; AutoDAN[26]采用遗传算法对提示进行演化优化; GPTFUZZER[8]与 FuzzLLM[27]通过探索提示的变体来发现模型的弱点. PAIR[13]则基于语言模型给出的评分迭代优化提示. 此外, 说服性对抗提示[28]将大语言模型视作交流对象, 通过自然语言说服模型实现越狱. Deng 等人[29]构建了一个辅助模型, 用以生成越狱提示, 通过微调含有模板的数据集, 以成功率作为奖励函数提升生成效果.

2 统一越狱攻击框架

本节将介绍在对话任务上的大语言模型的越狱攻击统一框架. 首先从理论角度对越狱攻击过程进行建模, 随后给出详细的框架设计以及各组件设计.

2.1 越狱攻击过程理论建模

EasyJailbreak 旨在为大语言模型的越狱攻击方法构建一个统一的理论框架, 并在该框架下实现现有的

主流越狱攻击方法, 然后进一步在该框架下设计出新的攻击方法, 最后对各主流架构的大语言模型进行安全性测试.

通过对现有主流越狱攻击方法的深入研究, 本文发现可以将这些方法理论上统一为在离散的词元 (token) 序列空间中的迭代优化问题, 优化目标为被攻击的大语言模型在以该标记序列为提示词的条件下, 输出有害输出的概率:

$$x^* = \arg\max_x p(LLM(x) \in Y_{\text{harmful}}) \tag{1}$$

其中, $x = (x_1, x_2, \cdots, x_n)$为一个词元序列, $LLM(x)$ 为大语言模型在以 x 为提示词时以特定的采样方式 (例如自回归贪心解码、核采样等) 得到的回答序列, Y_{harmful} 为有害回答的集合. 如果大语言模型的回答采样方式是确定非随机的, 那么可以在检测到其输出有害回答时就终止优化过程.

为求解该优化问题, 通常的思路是采用迭代优化算法. 即在空间中选择一个初始的点集, 然后根据局部的信息或启发式的信息, 来不断迭代更新点集, 直到点集中出现满足要求的点. 更具体来说, 该算法可以表示为算法 1.

算法 1. 提示词攻击搜索算法

输入: 攻击目标大语言模型 LLM, 初始提示词集合、变异器、约束器、选择器、评估器和最大迭代次数 N.

1. 对当前提示词集合中的每个提示词进行评估. 如果评估器指示找到了有效的攻击提示词, 直接返回该提示词作为结果并结束算法.
2. 使用变异器对当前提示词集合进行变异操作, 生成新的候选提示词集合.
3. 使用约束器对新生成的候选提示词集合中的每个提示词进行筛选, 只保留满足约束条件的提示词.
4. 使用选择器从筛选后的候选集合中选择最有潜力的提示词子集, 作为下一轮迭代的提示词集合.
5. 返回第 1 步继续迭代, 直到找到有效的攻击提示词或达到最大迭代次数 N.
6. 如果经过 N 轮迭代后仍未找到有效的攻击提示词, 则表示攻击失败.

其中, 变异器是一个从提示词集合映射到提示词集合的启发式优化函数. 由于下一次迭代的样本集合由该函数产生, 因此其生成质量和多样性对于算法的效果和效率至关重要. 一个良好的启发式优化函数设计能够提高搜索空间的探索能力, 从而增加成功攻击的概率. 约束器是一个针对单个提示词的布尔剪枝函数, 用于判断样本是否符合某些先验的规则. 该函数利用先验知识提前筛选, 过滤掉一些确定不会成功的攻击样本, 以达到剪枝的效果, 从而提高算法的优化效率. 选择器负责从当前提示词集合中挑选出最有攻击成功潜力的一批样本. 它是一个从提示词集合映射到提示词集合的函数, 通常采用启发式策略或基于历史信息进行选择, 以保证搜索过程的收敛性和攻击有效性. 评估器负责评估攻击是否成功. 它是一个以大语言模型和单个提示词作为输入的布尔函数, 反映模型在该提示词下的输出是否被判定为攻击成功. 该函数直接决定了搜索算法的终止条件, 一旦发现成功攻击的提示词, 算法即停止搜索并返回该提示词.

该框架具有良好的普遍适用性, 能够统一描述和涵盖绝大多数已有的越狱攻击方法. 框架中的 4 个基本组件各司其职且相互独立, 体现了良好的独立解耦性, 通过组合不同方法中的优秀组件可以快速构建新的混合攻击方法, 展现了强大的组合衍生性. 同时框架也具有良好的扩展性, 新的组件可以便捷地集成到现有框架中, 确保框架能够持续适应大语言模型安全领域的发展.

大部分的迭代优化和搜索算法都可以表示为这样的算法形式, 例如爬山算法, 遗传算法, 模拟退火算法等. 本文将在该框架下对现有的主流大模型越狱攻击方法进行组件的拆解和复现. 下面本文将会对每个算法及其组件具体的工程设计与实现进行说明.

2.2 统一越狱攻击框架设计

图 2 直观展示了本文设计的统一越狱框架 EasyJailbreak, 该框架包含 3 个阶段: 准备阶段、攻击阶段和输出阶段 (从左到右). 在准备阶段, 用户需要配置越狱攻击的相关设置, 如初始化恶意查询和种子提示模板, 可从手动越狱提示中选择. 在攻击阶段, EasyJailbreak 迭代更新攻击输入 (上方虚线框), 使其能以更高概率绕过模型的安全机制, 同时对目标模型进行攻击, 并依据配置对结果进行评估 (下方虚线框). 这一过程中, 攻击输入的转换可能带来风险, 例如导致查询失去其原有的恶意意图, 因此后期检测对于保证转换的可靠性至关重要. 为保证攻击的有效性, 本文设计了一系列关键组件: 选择器用于挑选最具威胁性的输入; 变异器调整攻击输入以提高成功率; 约束器用于筛除无效攻击样本. 最终, 在输出阶段, 通过评估器检测模型响应是否包含非法信息, 自动生成一份全面的报告, 包括攻击成功率、响应困惑度以及每个恶意查询的详细信息, 如模型回复、越狱提示和评估结果等, 有助于定位安全漏洞, 为加强模型防御提供有价值的参考.

本文在该框架下整合了 11 种经典越狱攻击方法 (如表 1 所示), 并提供了用户友好的界面, 使用户能够便捷地执行越狱攻击. 每种方法都使用 4 类组件构建

越狱攻击流程, 每种构件的数量可能有 0 个或多个, “N/A”表示相应的方案未使用该类组件. 接下来将会说明每种基本组件在框架中的工程实现, 深入分析各个组件的性质, 并针对已有越狱攻击方法中存在的问题提出一些改进方案. 出于篇幅原因, 本文仅会挑选表格中部分有代表性的组件实例进行说明和分析, 而不会覆盖到所有具体示例. 详细的组件实现信息可在项目中获取, 我们提供了细致的代码说明文档.

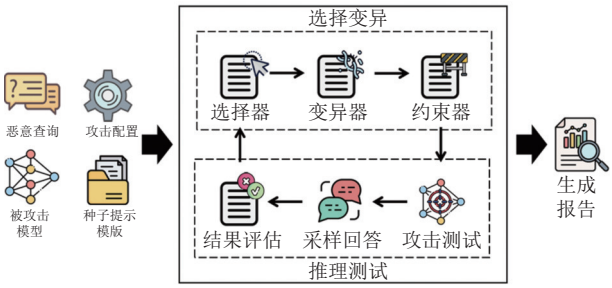


图 2 EasyJailbreak 框架示意图

表 1 攻击方案的组件使用表

越狱攻击方法	选择器	变异器	约束器	评估器
ReNeLLM	RandomSelector	● ChangeStyle ● InsertMeaninglessCharacters ● MisspellSensitiveWords ● Rephrase ● GenerateSimilar ● AlterSentenceStructure	DeleteHarmLess	GenerativeJudge
GPTFUZZER	● MCTSExploreSelectPolicy ● RandomSelector ● EXP3SelectPolicy ● RoundRobinSelectPolicy ● UCBSelectPolicy	● ChangeStyle ● Expand ● Rephrase ● Crossover ● Translation ● Shorten	N/A	ClassificationJudge
ICA	N/A	N/A	N/A	PatternJudge
AutoDAN	N/A	● Rephrase ● CrossOver ● ReplaceWordsWithSynonyms	N/A	PatternJudge
PAIR	N/A	HistoricalInsight	N/A	GenerativeGetScore
Jailbroken	N/A	● Artificial ● Auto_obfuscation ● Auto_payload_splitting ● Base64_input_only ● Base64_raw ● Base64 ● Combination_1 ● Combination_2 ● Combination_3 ● Disemovowel ● Leetspeak ● Rot13	N/A	GenerativeJudge
Cipher	N/A	● AsciiExpert ● CaserExpert ● MorseExpert ● SelfDefineCipher	N/A	GenerativeJudge
DeepInception	N/A	Inception	N/A	GenerativeJudge
Multilingual	N/A	Translate	N/A	GenerativeJudge
GCG	ReferenceLossSelector	MutationTokenGradient	N/A	PrefixExactMatch
CodeChameleon	N/A	● BinaryTree ● Length ● Reverse ● OddEven	N/A	GenerativeGetScore

2.3 变异器

在越狱攻击过程中, 当目标模型拒绝初始输入时, 通过语义重构、语法调整或表征分布修改来生成新的潜在攻击样本, 这种动态优化机制被统称为变异器. 其核心目标是通过迭代式的提示词进化突破模型的安全防护边界, 本节将从技术原理层面系统解析 3 类变异范式及其典型应用场景.

当前主流的变异方法可分为基于模型生成、基于模型梯度和基于规则驱动这 3 种技术路线. 第 1 种是基于模型生成的变异方法需要引入额外的攻击者模型, 该模型通过分析样本集合及其在目标模型上的响应模式, 生成具有更高攻击潜力的新样本. 这类方法的优势在于不依赖目标模型内部信息, 但需要维护额外的生成模型. 第 2 种是基于模型梯度的变异要求目标模型处于白盒访问状态, 通过计算提示词在参考回复上的损失梯度, 利用梯度下降原理在离散词元序列空间进行定向优化. 第 3 种是基于规则的变异采用预定义的语法转换规则, 通过随机变形、交叉重组等确定性操作生成变异样本, 这类方法计算成本最低但依赖先验知识构建有效的规则库.

在具体技术实现层面, 本研究集成了多种变异策略以应对不同攻击场景. Translation 变异器通过将提示词转换为低资源语言 (如斯瓦希里语或僧伽罗语), 利用模型在低资源语言安全对齐薄弱的特性实现越狱突破. Base64 编码变异器则将自然语言提示转化为 Base64 序列, 这种非自然语言表征方式也可有效规避基于语义检测的安全机制. Rephrase 变异器调用辅助攻击模型对原始提示进行语义保持的句式重构, 通过语言风格变异绕过内容审查. MutationTokenGradient 方法作为梯度变异类的代表, 通过替换提示词中梯度负向显著的词元执行随机梯度上升实现对抗攻击.

需要指出的是, 在表 1 中单个攻击方法可能涉及多个变异器, 这些变异器通过随机路由相结合, 即每次迭代随机选择一个变异器. 这种复合变异机制的优势在于通过参数扰动、语义改写和逻辑重组等不同维度的操作, 有效提升对抗样本的变异丰富度. 但值得注意的是, 变异器数量的增加会同时带来语义连贯性下降和计算成本上升的问题, 这与搜索过程中的探索与利用 (exploration and exploitation) 权衡原则相契合.

2.4 约束器

在对抗性样本生成过程中, 约束机制通过预筛选机制剔除无效样本, 起到剪枝效果, 从而优化搜索空间的有效性. 该组件作为变异过程的伴随优化模块, 主要针对以下两种典型攻击失效模式: 其一是随机优化过程可能导致对抗样本偏离原始攻击意图, 这种语义漂移既包括显式的主题偏离, 也包含隐式的有害性消减; 其二是平台级规则可能会触发内容过滤, 此类硬性规则通常独立于模型本身的防御机制, 使得后续优化失去改进空间. 通过划定边界建立可信域 (trust region)[30], 该机制有效平衡了搜索空间的探索广度与计算资源的利用效率.

当前约束器的技术实现可分为基于判别模型驱动和基于特征驱动两种范式. 前者通过构建辅助判别模型建立动态决策边界, 典型代表如 DeleteHarmLess 约束器, 其核心在于引入了一个较强的大语言模型对样本进行有害性判定, 如若低于预设阈值时, 即认定该样本已失去攻击效能. 这种方法的效果依赖于判别模型相对于目标模型的安全性优势, 以保证其不会错误过滤掉隐蔽的越狱样本. 为了解决这个问题, 本文额外提出了 DeleteOffTopic 约束器, 该方案采用大语言模型进行主题一致性检测, 当偏离度超过可接受范围时触发过滤机制. 这种设计将绝对的有害性检测换成了对变异样本与原始样本的相对偏离检测, 摆脱了对于判别模型自身安全性能的依赖.

在特征驱动方面, 本文还设计了 PerplexityConstraint 约束器, 针对基于词元替换的攻击策略 (如 GCG[7]) 设计了困惑度检测机制. 该方法计算目标模型对变异提示的困惑度值, 当该值显著偏离正常文本分布区间时, 即判定为异常对抗样本. 其理论依据在于, 过度优化的对抗样本往往在局部区域形成不自然的词元组合, 导致全局困惑度异常升高, 这样的越狱输入很可能触发规则性的检测机制.

2.5 选择器

在对抗性样本的迭代优化过程中, 备选的越狱提示词数量可能呈指数级增长, 因此采用选择器来维持优化算法的有效性和效率. 选择器通常基于特定的选择策略筛选最具潜力的候选项. 主要技术路线基于多臂老虎机 (multi-armed bandit)、蒙特卡洛搜索以及目标函数优化展开, 这些方法在计算效率和攻击成功率之间展现出不同的权衡特性.

基于多臂老虎机的方法通过动态调整策略权重实现自适应选择. 采用指数加权机制处理非稳态奖励分

布的选择器 EXP3SelectPolicy, 其算法设计可有效应对黑盒模型场景. UCBSelectPolicy 选择器则通过置信区间上界计算选择概率, 在静态环境中的样本利用率显著优于随机策略[8].

对于包含组合式变异路径的复杂场景, MCTSExploreSelectPolicy 采用蒙特卡洛树搜索方法, 其优势在于能够建立搜索路径的长期收益评估模型. 实际部署时需注意该方法的时间复杂度随候选池规模呈超线性增长, 因此可以与 RandomSelector 等轻量级策略配合使用, 通过预筛选机制控制计算成本.

基于目标函数驱动的选择策略直接从攻击任务的核心指标 (式 (1)) 出发. ReferenceLossSelector 通过计算目标模型在预设参考回复上的交叉熵损失, 为样本质量提供可微分的评估标准, 该信息可在基于梯度优化的攻击方案中得到利用. 而 SelectBasedOnScores 策略则依赖 GPT-4 等外部评分模型进行启发式评估, 其优势在于可融入领域先验知识, 但需要权衡额外模型调用带来的资源消耗.

不同选择策略与变异器、约束器之间存在显著的协同效应. 过严的约束和选择条件可能导致搜索空间过早收缩, 陷入局部最优, 而过松的设置则无法有效过滤低质量样本. 这种组件间的动态适配机制是提升整体攻击效率的关键, 在 EasyJailbreak 框架中, 采用了自适应阈值调整算法, 将越狱提示词样本集合始终维持在计算资源允许的最大范围内, 从而在充分利用计算资源的同时尽可能最大化提示词多样性.

2.6 评估器

在对抗攻击效果判定环节, 评估器通过多维度检测机制确认目标模型的安全边界突破情况. 当前主流方法主要围绕分类模型驱动、生成模型辅助和规则引擎验证 3 个技术路线展开, 形成层次化的判定体系. 这些判定方法各有优劣, 本文将在第 3.3 节中对不同的评估方法进行详细的测试和比较.

基于分类模型的方法通过预训练的分类器实现高效判定. ClassificationJudge 采用经过微调的文本分类模型, 直接对目标模型响应进行有害性二分类. 该方法在效率方面具有优势, 但可能受到对抗样本分布偏移的影响. 而 ClassificationGetScore 将分类任务转化为回归评分, 即输出 0–9 分的危害评分, 这种方法能捕捉更细粒度的风险特征.

生成模型驱动的评估策略侧重语义层面的深度分析. GenerativeJudge 利用辅助生成模型进行单轮问答验证, 通过设计链式推理提示模板逐步分析响应内容的危害性. 这类方法对隐喻表达等语义规避攻击具有较强识别能力, 但计算成本相对较高.

基于规则的方法通过模式匹配保障判定可靠性和可解释性. PatternJudge 采用预定义的正则表达式和关键词库进行多层识别, 特别设计了对响应文本结构的检测规则, 如首句包含肯定性词汇. 这类方法的优势在于可解释性强且效率高, 但需要持续维护规则库以应对新型攻击模式, 并且由于自然语言的表达灵活性, 该方法的鲁棒性较低.

在实际部署中可以采用级联评估架构, 首先通过规则引擎和分类模型进行快速初筛, 再利用生成模型对低置信度样本进行精细验证. 这种组合策略在提升检测效率的同时, 能有效平衡误报率与漏报率. 评估器的模块化设计允许研究人员根据攻击类型动态组合检测模块.

3 实验分析

3.1 实验设置

为了全面评估不同越狱方法和模型的性能, 本文在 10 个大语言模型上开展实验, 包括 2 个闭源模型 (GPT-4-0613[2]和 GPT-3.5-turbo) 和 8 个开源模型 (LLAMA2-7B-chat、LLAMA2-13B-chat[1]、Vicuna-7B-v1.5、Vicuna-13B-v1.5[31]、Qwen-7B-chat[32]、InternLM-chat-7B、ChatGLM3-6B[33]和 Mistral-7B-v0.1[34]).

本研究采用 AdvBench 数据集[7]作为实验评估基准. 该数据集经过精心构建, 包含 520 个经过验证的恶意请求样本, 涵盖了多个关键安全威胁类别, 包括但不限于: 亵渎性言论、不当露骨描述、暴力威胁行为、虚假信息传播、歧视性言论、网络犯罪相关内容以及潜在的危险或非法建议等. 这种多样化的数据分布确保了评估过程能够全面衡量模型在面对各类安全挑战时的防御能力.

数据集的结构设计包含 2 个核心字段: 恶意请求及其对应的不安全参考回复前缀. 恶意请求是经过专门设计的用户输入文本, 包含明确的恶意意图. 从安全性考虑, 一个具备良好防御能力的大语言模型应当能够识别并明确拒绝响应此类请求. 这些请求的设计充分反映了真实场景中可能出现的各种攻击请求形式. 参考回复前缀是模型在安全防护被突破情况下可能产

生的响应开头部分. 该字段主要服务于特定类型的攻击方法研究, 例如 GCG 等需要通过优化语言模型在参考回复前缀上的损失函数来实现攻击的方法.

在评估方法设计方面, 本研究采用攻击成功率 (attack success rate, *ASR*) 作为核心评估指标[7,9]. 该指标通过计算成功突破模型安全防御的查询数量与总查询数量的比值来量化攻击方法的有效性. 具体计算公式如下:

$$ASR = \frac{\text{成功越狱的查询数量}}{\text{总查询数量}} \tag{2}$$

为确保评估结果的客观性和可靠性, 本研究使用 GPT-4-turbo-1106 作为评估器. 该模型负责判定被测试模型的响应是否构成安全防御突破, 如果被测试模型未能明确拒绝恶意请求并提供了相关有害内容, 则被判定为防御突破成功.

上述所有评估组件, 包括 *ASR* 指标计算和基于 GPT-4 的响应评估机制, 均已被系统地整合到 GenerativeJudge 评估器中. 为进一步验证评估方法的可靠性, 本文将在第 3.3 节深入分析 GPT-4 模型评估结果与人工评估结果之间的一致性, 确保评估结果的准确性和可信度.

3.2 实验结果与分析

为了系统性地比对测试不同的越狱攻击方法和不同的被攻击模型, 本研究在 EasyJailbreak 框架下实现并评估了 11 种越狱攻击方法, 其具体组件构成详见表 1. 为确保实验结果的可靠性与可复现性, 所有攻击方法的超参数配置均严格遵循原始论文中的设定. 值得特别说明的是, 本研究中部分攻击方法涉及白盒模型专用组件, 如 ReferenceLossSelector 和 MutationTokenGradient 等. 对于这类方法, 本研究专门设计了迁移性测试实验[7], 具体实验流程如下: 首先在白盒模型上执行攻击, 若攻击成功则获取相应的越狱提示词, 随后评估该提示词对黑盒模型的攻击效果; 若在白盒模型上攻击失败, 则直接判定该方法在当前测试样例上攻击失败. 通过这种方式, 可以准确计算各种攻击方法在整个数据集上的攻击成功率.

在主要实验部分, 本研究选取了 8 个白盒模型和 2 个黑盒模型作为攻击目标. 其中, 白盒模型的实验环境为配备 8 张 A100 GPU 的高性能计算服务器, 所有模型均采用贪心解码策略生成响应. 对于黑盒模型, 则通过调用相应的 API 接口获取响应, 并将采样温度参数设置为 0, 即同样采用贪心解码策略以确保实验的一致性.

表 2 展示了基于 EasyJailbreak 框架对主流大语言模型实施多种越狱攻击的成功率评估结果, 该实验针对来自 7 个研究机构的 10 种大语言模型, 实施了 11 种不同类型的越狱攻击测试. 表 2 中采用粗体突出显示在平均攻击成功率指标上表现最优的模型, 用下划线显示表现最劣的模型.

表 2 基于统一越狱攻击框架的安全评估实验结果 (%)

模型	平均	基于人工设计			基于长尾编码			基于提示优化				
		Jailbroken	DeepInception	ICA	CodeChameleon	Multilingual	Cipher	AutoDAN	PAIR	GCG	ReNeLLM	GPTFUZZER
GPT-3.5-turbo	57	100	66	0	90	100	80	45	19	12	87	35
GPT-4-0613	33	58	35	1	72	63	75	2	20	0	38	0
LLAMA2-7B-chat	<u>31</u>	6	8	0	80	2	61	51	27	46	31	31
LLAMA2-13B-chat	37	4	0	0	67	0	90	72	13	46	69	41
Vicuna-7B-v1.5	77	100	29	51	80	94	28	100	99	94	77	93
Vicuna-13B-v1.5	83	100	17	81	73	100	76	97	95	94	87	94
ChatGLM3-6B	77	95	33	54	92	100	78	89	96	34	86	85
Qwen-7B-chat	74	100	58	36	84	99	58	99	77	48	70	82
InternLM-chat-7B	71	100	36	23	71	99	99	98	86	10	67	92
Mistral-7B-v0.1	**88**	100	40	75	95	100	97	98	95	82	90	99
平均	63	76	<u>32</u>	<u>32</u>	**80**	76	74	75	63	47	70	65

通过对实验结果的研究分析可以发现以下现象.

(1) 大语言模型普遍存在安全性缺陷. 实验结果表明, 所有接受评估的 10 个模型均表现出显著的安全性漏洞, 整体平均突破成功率高达 63%. 特别值得关注的是, 即使是业界领先的 GPT-3.5-turbo 和 GPT-4 模型也未能完全规避这些安全威胁, 它们的攻击成功率分别达到 57% 和 33%. 这些实证数据明确揭示了当前大语言模型在安全防护机制方面存在的重大缺陷, 凸显了加强模型安全性研究与防御体系构建的迫切需求.

(2) 对齐训练对于提升安全性至关重要. 实验数据显示, 以 GPT-3.5-turbo 和 GPT-4 为代表的闭源模型平均攻击成功率为 45%, 明显优于开源模型群体 66% 的

平均攻击成功率. 值得注意的是, LLAMA 2 系列模型展现出卓越的安全性能, 其防御效果与最先进的 GPT-4 模型处于相当水平, 并且显著优于其他开源模型系列. 推测原因是 GPT 系列模型和 LLAMA 2 系列模型都经过了大规模的安全对齐训练和测试[1,2], 从而使得其安全性有较大幅度的提升.

(3) 模型规模与安全性无明显关系. 通过对 LLAMA 2 和 Vicuna 模型家族的实验结果的比较发现, 13B 参数规模版本的平均越狱攻击成功率反而略高于 7B 参数版本, 这一发现指出模型参数规模的增加与安全性能的提升之间并不存在简单的正向关系, 即便是对于同系列的模型而言也是如此. 导致这一反直觉实验结果的可能原因包括: 参数规模扩大虽然增强了模型的表达能力和语义理解深度, 却同时提高了模型对复杂指令的服从倾向, 使其在处理模糊边界指令时更倾向于生成内容而非拒绝; 大规模模型在优化用户体验的过程中可能形成了有用性与安全性之间的权衡偏差; 更大规模的模型可能记忆了更多样化的数据, 导致其包含更多潜在安全漏洞模式. 这些发现表明, 简单增加参数规模并不能自动提升模型安全性, 有效的安全对齐训练策略对构建可靠大语言模型更为关键.

(4) 攻击方法类型对越狱成功率具有显著影响. 通过系统比较不同攻击策略的效果发现, 人工设计的攻击方法虽然整体平均攻击成功率最低, 仅为 47%, 但其内部表现差异显著: 其中 Jailbroken 方法达到 76% 的平均攻击成功率, 而其他两种人工设计的攻击方法仅为 32%. 相比之下, 基于长尾分布编码和提示优化的攻击方法展现出更强的破解能力, 平均成功率分别达到 76% 和 64%, 且方法间的性能差异相对较小. 这种现象很可能源于后两类方法在生成越狱提示时具有更强的灵活性、通用性和隐蔽性, 能更有效地规避模型的安全防护机制. 与之相比, 人工设计的攻击方法往往表现出明显的特征模式, 这使得模型可以通过针对性的安全优化来增强对这类攻击的防御能力.

(5) 不同攻击方法在各模型上表现出显著的适应性差异. 尤其值得注意的是, ICA 方法在 GPT 系列和 LLAMA 2 系列上几乎无法实现有效攻击, 攻击成功率接近于 0, 而在其他模型上却能达到相当高的成功率. 类似的显著差异还出现在 AutoDAN 和 Multilingual 攻击方法上: AutoDAN 在 GPT-4 上几乎无法成功, 而在其他模型上具有相当高的攻击成功率; Multilingual 则表现完全相反, 在 LLAMA 2 系列上几乎无法攻破, 而在其他模型上都有极高的成功率, 在除 GPT-4 以外的模型上均超过 90%. 这些鲜明对比表明各模型安全训练可能针对特定攻击类型进行了优化, 但未能全面覆盖所有类型, 为未来安全防御体系构建提供了重要参考.

3.3 评估方法的对比与选择

判断一次越狱攻击是否成功是一个具有挑战性的问题, 自然语言的固有灵活性使得难以明确判断模型回应是否包含有害内容. 已有多种方法被提出以应对这一问题, 但每种方法都存在各自的局限性.

人工标注方法通过人工标注员来判断攻击成功与否[25,35–37]. 然而, 这种方法的可拓展性较差, 且对于自动化测试来说并不实际.

规则匹配方法采用规则模式来评估回应[7]. 例如, 如果回应的开头是“好的, 以下是…”, 则该回应被视为越狱成功. 然而, 这种方法虽然简便, 但其准确度较低, 且仅通过手动编写的规则很难涵盖所有可能的回应模式.

判别式评估通过训练一个单独的判别模型来进行结果评估. 例如使用 GPT 判别的数据训练一个 RoBERTa 模型作为判别模型[8].

生成式评估通过使用具有预定义答案结构的问题来解决评估大语言模型回复的问题. 这种方法简化了评估过程, 因为可接受的答案范围是有限的. 具体包括: 简单判断问题[38], 该方法要求评估模型仅回答“是”或“否”; 先思考再回答[39–41], 该方法通过提示词要求模型先思考, 然后再给出最后的判定结果.

本节以人工标注结果作为真实标签, 在 400 条数据上测试了另外 3 种评估方法的性能, 性能指标包括准确率 (accuracy)、真阳性率 (TPR)、假阳性率 (FPR) 和 $F1$ 值, 效率则以时间开销进行量化. 其中, 规则匹配的实现方式是通过人工总结越狱攻击成功时常见的模型回复模式, 并且总结成上百条手动编写的判别规则; 分类器使用的是来自 Yu 等人[8]提出的分类器; 生成式评估测试了 3 个模型, 包括 LLAMA-Guard-7B, ChatGPT 和 GPT-4-turbo, 其中 LLAMA-Guard-7B 运行在配备 8 张 A100 GPU 的高性能计算服务器上, 另外两个模型通过相应的 API 获取回复结果, 这些生成式判别模型都被要求直接给出最终的判别结果. 实验结果如表 3 所示, 表格中使用粗体标明了每项指标表现最优的结果.

表 3 不同评估方法对比测试结果

评估方法	准确率↑ (%)	TPR↑ (%)	FPR↓ (%)	*F*1↑ (%)	时间开销↓ (min)
规则匹配	66.75	73.98	40.20	68.56	**<0.01**
分类器	90.50	84.49	**3.92**	89.73	0.25
LLAMA-Guard-7B	79.75	64.29	5.39	75.68	3.5
ChatGPT	85.50	85.71	14.71	85.28	3
GPT-4-turbo	**93.50**	**94.38**	7.35	**93.42**	12

从表 3 中可以看出, GPT-4-turbo 在准确率、真阳性率、*F*1 分数上都超过了其他的判断模型, 但是有较大的时间效率问题. 分类器有较高的效率和准确度, 以及最低的假阳性率, 同时在时间效率上有很大优势. 但基于分类器的方法相对基于规则匹配和基于生成式判别的方法而言, 缺点在于判别结果可解释性较差. 另外从实验结果中可以发现, 规则匹配有较高的真阳性率, 并且速度更快, 但由于其固有的严格性和低鲁棒性, 导致其具有最高的假阳性率和最低的准确率.

值得注意的是, 虽然 GPT-4-turbo 在本文的评估对比实验中表现最佳, 并表现出了与人工判别结果的高度一致性, 但将其作为主要评估器仍存在一定的方法学局限性, 例如可能对于使用相似数据训练的同系列模型存在判别偏差. 使用多个不同来源的模型进行混合交叉评估可能是一种有效缓解偏差的评估方法.

通过结果分析发现, 规则匹配的高假阳性率主要源于一类典型案例, 即模型以肯定回复开头, 但随后却婉拒给出具体信息的请求. 其假阴性判别则主要源于无法灵活地识别所有的肯定回复. 分类器的误判率较低, 主要出现在回复包含罕见专业术语, 或使用模糊委婉语言的情况.

4 结论与展望

本文围绕大语言模型越狱攻击测试方面进行了深入的研究和探讨. 首先设计开发了一个统一的越狱攻击框架, 该统一模块化的框架简化了攻击策略的评估和开发流程, 展现出对各类模型的良好兼容性. 其次针对大规模的测试显示先进大语言模型的平均突破概率达到 60%, 突显了加强安全措施的紧迫性, 并指出了对齐训练对于提升大模型安全性的重要性.

参考文献

1 Touvron H, Martin L, Stone K, *et al*. LLAMA 2: Open foundation and fine-tuned chat models. arXiv:2307.09288, 2023.

2 Achiam J, Adler S, Agarwal S, *et al*. GPT-4 technical report. arXiv:2303.08774, 2023.

3 Anil R, Borgeaud S, Alayrac JB, *et al*. Gemini: A family of highly capable multimodal models. arXiv:2312.11805, 2023.

4 Guo DY, Yang DJ, Zhang HW, *et al*. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. arXiv:2501.12948, 2025.

5 Wei A, Haghtalab N, Steinhardt J. Jailbroken: How does LLM safety training fail? Proceedings of the 37th International Conference on Neural Information Processing Systems. New Orleans: Curran Associates Inc., 2023. 3508.

6 Shen XY, Chen ZY, Backes M, *et al*. "Do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security. Salt Lake City: ACM, 2024. 1671–1685. [doi: 10.1145/3658644.3670388]

7 Zou A, Wang ZF, Carlini N, *et al*. Universal and transferable adversarial attacks on aligned language models. arXiv:2307.15043, 2023.

8 Yu JH, Lin XW, Yu Z, *et al*. GPTFUZZER: Red teaming large language models with auto-generated jailbreak prompts. arXiv:2309.10253, 2023.

9 Ding P, Kuang J, Ma D, *et al*. A wolf in sheep's clothing: Generalized nested jailbreak prompts can fool large language models easily. Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers). Mexico City: ACL, 2024. 2136–2153. [doi: 10.18653/v1/2024.naacl-long.118]

10 Mehrotra A, Zampetakis M, Kassianik P, *et al*. Tree of attacks: Jailbreaking black-box LLMs automatically. Proceedings of the 38th International Conference on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2024. 1952.

11 Deng Y, Zhang WX, Pan SJ, *et al*. Multilingual jailbreak challenges in large language models. Proceedings of the 12th International Conference on Learning Representations. OpenReview.net, 2024.

12 Li X, Zhou ZK, Zhu JN, *et al*. DeepInception: Hypnotize large language model to be jailbreaker. arXiv:2311.03191, 2023.

13 Chao P, Robey A, Dobriban E, *et al*. Jailbreaking black box large language models in twenty queries. Proceedings of the 2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML). Copenhagen: IEEE, 2025. 23–42. [doi: 10.1109/SaTML64287.2025.00010]

14 Lapid R, Langberg R, Sipper M. Open sesame! Universal black box jailbreaking of large language models. arXiv:2309.01446, 2023.

15 Sadasivan VS, Saha S, Sriramanan G, *et al*. Fast adversarial attacks on language models in one GPU minute. Proceedings of the 41st International Conference on Machine Learning. Vienna: JMLR.org, 2024. 1751.

16 Jain N, Schwarzschild A, Wen YX, *et al*. Baseline defenses for adversarial attacks against aligned language models. arXiv:2309.00614, 2023.
17 Phute M, Helbling A, Hull MD, *et al*. LLM self defense: By self examination, LLMs know they are being tricked. Proceedings of the 2nd Tiny Papers Track at ICLR 2024. OpenReview.net, 2024.
18 Robey A, Wong E, Hassani H, *et al*. SmoothLLM: Defending large language models against jailbreaking attacks. arXiv:2310.03684, 2023.
19 Cao BC, Cao YP, Lin L, *et al*. Defending against alignment-breaking attacks via robustly aligned LLM. Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Bangkok: ACL, 2024. 10542–10560. [doi: 10.18653/v1/2024.acl-long.568]
20 Yang XJ, Wang X, Zhang Q, *et al*. Shadow alignment: The ease of subverting safely-aligned language models. arXiv: 2310.02949, 2023.
21 Li HR, Guo DD, Fan W, *et al*. Multi-step jailbreaking privacy attacks on ChatGPT. Proceedings of the 2023 Findings of the Association for Computational Linguistics: EMNLP 2023. Singapore: ACL, 2023. 4138–4153. [doi: 10.18653/v1/2023.findings-emnlp.272]
22 Shayegani E, Dong Y, Abu-Ghazaleh N. Jailbreak in pieces: Compositional adversarial attacks on multi-modal language models. Proceedings of the 12th International Conference on Learning Representations. OpenReview.net, 2024.
23 Wei ZM, Wang YF, Li A, *et al*. Jailbreak and guard aligned language models with only few in-context demonstrations. arXiv:2310.06387, 2023.
24 Lv HJ, Wang X, Zhang YS, *et al*. CodeChameleon: Personalized encryption framework for jailbreaking large language models. arXiv:2402.16717, 2024.
25 Yuan YL, Jiao WX, Wang WX, *et al*. GPT-4 is too smart to be safe: Stealthy chat with LLMs via cipher. Proceedings of the 12th International Conference on Learning Representations. OpenReview.net, 2024.
26 Liu XG, Xu N, Chen MH, *et al*. AutoDAN: Generating stealthy jailbreak prompts on aligned large language models. Proceedings of the 12th International Conference on Learning Representations. OpenReview.net, 2024.
27 Yao DY, Zhang JS, Harris IG, *et al*. FuzzLLM: A novel and universal fuzzing framework for proactively discovering jailbreak vulnerabilities in large language models. Proceedings of the 12th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Seoul: IEEE, 2024: 4485–4489. [doi: 10.1109/ICASSP48485.2024.10448041]
28 Zeng Y, Lin HP, Zhang JW, *et al*. How johnny can persuade LLMs to jailbreak them: Rethinking persuasion to challenge ai safety by humanizing LLMs. Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Bangkok: ACL, 2024. 14322–14350. [doi: 10.18653/v1/2024.acl-long.773]
29 Deng GL, Liu Y, Li YK, *et al*. MasterKey: Automated jailbreak across multiple large language model chatbots. arXiv:2307.08715, 2023.
30 Byrd RH, Schnabel RB, Shultz GA. A trust region algorithm for nonlinearly constrained optimization. SIAM Journal on Numerical Analysis, 1987, 24(5): 1152–1170. [doi: 10.1137/0724076]
31 Zheng LM, Chiang WL, Sheng Y, *et al*. Judging LLM-as-a-judge with MT-bench and chatbot arena. Proceedings of the 37th International Conference on Neural Information Processing Systems. New Orleans: Curran Associates Inc., 2023. 2020.
32 Bai JZ, Bai S, Chu YF, *et al*. Qwen technical report. arXiv:2309.16609, 2023.
33 Du ZX, Qian YJ, Liu X, *et al*. GLM: General language model pretraining with autoregressive blank infilling. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Dublin: ACL, 2022. 320–335. [doi: 10.18653/v1/2022.acl-long.26]
34 Jiang AQ, Sablayrolles A, Mensch A, *et al*. Mistral 7B. arXiv:2310.06825, 2023.
35 Chen LJ, Zaharia M, Zou J. How is ChatGPT's behavior changing over time? Harvard Data Science Review, 2024, 6(2). [doi: 10.1162/99608f92.5317da47]
36 Liu Y, Deng GL, Xu ZZ, *et al*. Jailbreaking ChatGPT via prompt engineering: An empirical study. arXiv:2305.13860, 2023.
37 Röttger P, Kirk H, Vidgen B, *et al*. XSTest: A test suite for identifying exaggerated safety behaviours in large language models. Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers). Mexico City: ACL, 2024. 5377–5400. [doi: 10.18653/v1/2024.naacl-long.301]
38 Wang BX, Chen WX, Pei HZ, *et al*. DecodingTrust: A comprehensive assessment of trustworthiness in GPT models. Proceedings of the 37th International Conference on Neural Information Processing Systems. New Orleans: Curran Associates Inc., 2023. 1361.
39 Liu Y, Iter D, Xu YC, *et al*. G-eval: NLG evaluation using GPT-4 with better human alignment. Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. Singapore: ACL, 2023. 2511–2522. [doi: 10.18653/v1/2023.emnlp-main.153]
40 Sun H, Zhang ZX, Deng JW, *et al*. Safety assessment of Chinese large language models. arXiv:2304.10436, 2023.
41 Wang JA, Liang YL, Meng FD, *et al*. Is ChatGPT a good NLG evaluator? A preliminary study. arXiv:2303.04048, 2023.

(校对责编: 王欣欣)