

没有免费的午餐有了护栏

Divyanshu Kuamr, Nitin Aravind Birur, Tanay Baswa, Sahil Agarwal & Prashanth Harshangi
 Enkrypt AI
 MA, USA
 {divyanshu, nitin, tanay, sahil, prashanth}@enkryptai.com

Abstract

随着大型语言模型 (LLMs) 和生成式 AI 被广泛采用, 护栏已成为确保其安全使用的关键工具。然而, 添加护栏并非没有权衡——更强的安全措施可能会降低可用性, 而更加灵活的系统可能留下对抗攻击的漏洞。在这项工作中, 我们探讨当前的护栏是否能够有效地防止误用并保持实际效用。我们介绍了一个框架来评估这些权衡, 测量不同的护栏如何平衡风险、安全性和可用性。

我们的研究证实了**没有免费的午餐带有护栏**加强安全往往以牺牲可用性为代价。为了应对这一问题, 我们提出了一种设计更好的护栏的蓝图, 该蓝图在保持可用性的前提下最小化风险。我们评估了各种行业护栏, 包括 Azure 内容安全¹、Bedrock 护栏²、OpenAI 的 Moderation API、Guardrails AI³ 和 Enkrypt AI 护栏⁴。此外, 我们还评估了在不同系统提示下的 LLM (如 GPT-4o、Gemini 2.0-Flash、Claude 3.5-Sonnet 和 Mistral Large-Latest) 的响应情况, 包括简单提示、详细提示以及带有连贯性思维 (CoT) 推理的详细提示。我们的研究提供了不同护栏性能的明确对比, 突显了在平衡安全性和可用性方面的挑战。

1 介绍

大型语言模型 (LLMs) 的部署增加引发了重要的安全问题, 特别是关于它们生成有害、有偏见或违反政策内容的能力。这些风险因对抗性输入和分布变化而加剧, 使 LLMs 容易被滥用。为此, 监控、过滤或修改模型输出的护栏系统已成为常见的缓解策略。

然而, 护栏引入了不可避免的权衡。更强的安全措施减少了剩余风险, 但可能会抑制合法或有价值的输出, 降低实用性和可用性。相反, 宽松的过滤器保留了输出流畅性, 但却增加了有害内容通过的风险。这种紧张关系在高风险领域尤为成问题, 在这些领域中, 安全违规和不必要的审查都可能导致严重后果。

在本文中, 我们将这一挑战形式化为**没有免费的午餐 (NFL)** 假设对于护栏。我们论证没有防护系统能够在现实的对抗性和自然条件下同时最小化剩余风险、保持效用并避免可用性损失。为了验证这一点, 我们在对抗性的提示和良性内容被误标为有害 (伪伤害) 的情境下, 对三种常见的防护架构——提供商 API、基于 BERT 的分类器和基于 LLM 的评估器进行了实证比较。

¹<https://contentsafety.cognitive.azure.com/>

²<https://aws.amazon.com/bedrock/guardrails/>

³<https://www.guardrailsai.com/>

⁴<https://app.enkryptai.com/guardrails>

1.1 相关工作

大型语言模型护栏 Ayyamperumal & Ge (2024) 的格局迅速多样化，涵盖了基于分类器的系统、由大型语言模型驱动的评估器以及专门的架构创新。基于分类器的方法在商业部署中仍然最为普遍，如 OpenAI 的 Moderation API、Azure 内容安全和 Amazon Bedrock Guardrails 提供了通过预训练的风险分类器进行快速、低延迟过滤的服务。这些系统在标准基准测试上展示了高吞吐量和有效性，但它们在对抗压力下的局限性、误报的敏感性和对任务效用的影响则研究得较少。

相比之下，由 LLM 驱动的评估者利用模型自身的推理能力来评估政策违规行为，通常通过基于提示的技术如思维链 (CoT) 来进行。先前的研究表明，这些方法可以提高对抗鲁棒性并提供可解释的拒绝理由 Kazemi Rad et al. (2025)，而类似 AGrail Luo et al. (2025) 这样的代理级别框架将这些能力扩展到长期动态任务中。然而，这些改进通常伴随着显著的计算成本，通常是分类器系统 5 – 10 倍 $\times$ 慢，引发了关于可扩展性和部署可行性的担忧。为了解决这种权衡，专门架构如 LoRA-Guard Elesedy et al. (2024) 和 SafeRoute Lee et al. (2025) 引入了参数高效适应和动态模型路由，实现了显著的延迟减少。同时，混合框架如 RigorLLM Yuan et al. (2024) 和多模态安全系统如 UniGuard Oh et al. (2024) 提高了对抗韧性并覆盖更多模态。尽管有这些进展，大多数研究仍然孤立地评估进度，没有系统地检查安全性、效用性和使用差距之间的权衡。

研究空白： 尽管护栏研究取得了显著进展，但仍存在一个关键差距：在护栏设计中，尚未对安全、实用性和易用性之间的基本权衡进行系统评估。。大多数现有研究仅专注于优化单一目标，如对抗鲁棒性、计算效率或内容效用，而没有量化在某一轴上的收益可能如何损害其他方面。特别是，在伪伤害检测问题中，由于与有害语言表面相似，良性的内容被错误地标记为有害内容的问题仍未得到充分探索，尽管这个问题具有实际意义。基于 LLM 的审核系统对现实世界数据的初步评估显示了强大的真阴性性能，但在复杂违规行为上的准确率有限，真阳性率为 43.1% Kolla et al. (2024)，这凸显了当前自动化方法的局限性和需要更多细致、符合人类需求的审核策略。

我们的工作通过提供首个统一的实证框架来解决这些差距，该框架同时评估了在多种护栏架构中安全性、实用性和计算效率。通过对对抗条件和自然条件下这些权衡进行系统分析，我们全面理解了护栏设计与部署中存在的内在紧张关系。

2 预备知识

2.1 问题设置和护栏定义

我们考虑一个大型语言模型 (LLM)，表示为 $\mathcal{M}$ ，它根据输入提示 $x \in \mathcal{X}$ 生成输出 $y \in \mathcal{Y}$ ，从模型学习的条件分布中采样：

$$y \sim \mathcal{M}(y|x) \quad (1)$$

虽然 $\mathcal{M}$ 在连贯性和通用生成方面进行了优化，但在安全性、伦理一致性和符合下游应用程序政策等方面没有内在保证。因此，LLM 可能会生成事实错误、冒犯性、有偏见或以其他方式有害的内容。

为了缓解这些风险，部署管道中集成了一个护栏系统 $\mathcal{G}$ ，作为生成后的过滤器或修改层。最终面向用户的输出 y' 定义为：

$$y' = \mathcal{G}(\mathcal{M}(y|x)) \quad (2)$$

该护栏强制执行一个二元策略函数 $\pi: \mathcal{Y} \rightarrow \{0, 1\}$ ，其中 $\pi(y') = 1$ 表示接受内容为安全且符合政策，而 $\pi(y') = 0$ 则导致拒绝或抑制内容。

这一抽象概括了广泛的实际部署实践，包括基于 API 的内容审核、分类器驱动的过滤以及基于推理的评估。重要的是，这种框架将护栏视为独立于生成过程之外的部分，使我们能够孤立并独立于 $\mathcal{M}$ 的生成行为来评估它们的影响。

2.2 护栏行为的特征化

鉴于大型语言模型输出的随机性以及 $\mathcal{G}$ 的过滤决策，我们从三个关键维度对系统行为进行概率建模：剩余风险、效用影响和可用性损失。这些指标共同量化了任何护栏机制的有效性和副作用。

残余风险（假阴性）： 该指标捕捉到了生成有害内容并被护栏接受的概率。具体来说，它反映了在通过政策检查时， y' 属于有害内容空间 $\mathcal{Y}_{\text{harmful}}$ 的可能性：

$$\mathbb{P}(\pi(y') = 1 \mid y' \in \mathcal{Y}_{\text{harmful}}) \quad (3)$$

一个设计良好的护栏旨在将此概率降至最低，但由于自然语言的复杂性和模糊性，完全做到这一点是不可能的。

效用影响（信息损失）： 护栏可能会无意中降低大型语言模型输出的信息量、正确性或任务相关性。我们将这种降级量化为未过滤输出 y 和过滤后输出 y' 之间的预期效用分数差异，其中 $\mathcal{U}(\cdot)$ 是一个特定应用的评分函数：

$$\mathbb{E}[\mathcal{U}(y) - \mathcal{U}(y')] \quad (4)$$

这既涵盖了显式内容删除，也涵盖了由过滤引起的生成质量细微变化。

可用性损失（误报和延迟开销）： 除了过滤错误外，护栏还会施加计算成本，这直接影响系统可用性。可用性损失涵盖两个关键方面：(i) 拒绝良性或有价值内容的概率，特别是在合法讨论与有害模式重叠的敏感领域中；(ii) 护栏评估引入的延迟开销，这可能会降低交互式系统的用户体验。

正式地，我们定义假正率如下：

$$\mathbb{P}(\pi(y') = 0 \mid y' \in \mathcal{Y}_{\text{safe}}) \quad (5)$$

此外，我们将预期的护栏引起的延迟表示为 $\mathbb{E}[\Delta t]$ ，捕捉到与原始模型生成相比 $\mathcal{G}$ 增加的额外时间。这种开销对于基于 LLM 的护栏（ $\mathcal{G}_{\text{LLM}}$ ）或多阶段评估来说尤为重要。

假阳性率和延迟惩罚都是关键的可用性考量因素。过度过滤会降低系统的实际效用，而增加的延迟可能会使应用程序不适合实时或面向用户的情景。

2.3 范围内的护栏架构

在实践中，护栏通过多种架构实现，每种架构在覆盖率、可解释性、计算成本和对对抗绕过的敏感度方面都有不同的权衡。本研究关注三种广泛应用于工业和研究中的代表性类别。

提供商部署的护栏（ $\mathcal{G}_P$ ）： 这些是由模型提供商作为 API 服务公开的黑盒安全层。给定一个大语言模型输出 y ，防护措施计算出一个标量风险分数 $S_P(y)$ ，并与预定义阈值 τ_P 进行

比较：

$$\mathcal{G}_P(y) = \begin{cases} y & \text{if } S_P(y) \leq \tau_P \\ \emptyset & \text{otherwise} \end{cases} \quad (6)$$

示例包括 OpenAI 的 Moderation API、Azure Content Safety 和 Amazon Bedrock Guardrails。这些系统高效且通用，但通常缺乏透明度，并且在处理细微且依赖上下文的内容时存在困难。

基于 BERT 的分类器护栏 ($\mathcal{G}_{\text{BERT}}$)： 这些护栏采用了经过内容审核微调的预训练掩码语言模型。每个输出 y 都由分类器预测的危害概率 $B(y)$ 进行评分，并且超过阈值 τ_B 的内容将被过滤：

$$\mathcal{G}_{\text{BERT}}(y) = \begin{cases} y & \text{if } B(y) \leq \tau_B \\ \emptyset & \text{otherwise} \end{cases} \quad (7)$$

基于 BERT 的模型提供了快速推理和一定程度的可解释性，但在处理复杂语境或对抗性模糊输入方面的能力有限。

基于 LLM 的护栏 ($\mathcal{G}_{\text{LLM}}$)： 一种日益常见的方法是将大型语言模型本身用作审核代理。一个监督的 $\text{LLM}\mathcal{M}_g$ 负责评估候选输出 y 并基于安全考虑做出接受或拒绝的决定：

$$\mathcal{G}_{\text{LLM}}(y) = \begin{cases} y & \text{if } \mathcal{M}_g(e, y) = \text{ACCEPT} \\ \emptyset & \text{otherwise} \end{cases} \quad (8)$$

其中 e 表示提供给 $\mathcal{M}_g$ 的审核指令或系统提示。

在实践中， $\mathcal{G}_{\text{LLM}}$ 的行为和有效性很大程度上取决于评估提示的设计。为了系统地探索这种敏感性，我们评估了三种代表推理复杂度逐渐增加的提示策略：(i) 一个简单的调制提示，(ii) 一个详细说明危害类别和决策启发式的提示，以及 (iii) 一个增强型连贯思维 (CoT) 提示，鼓励在判断前逐步推理。

每个中等提示变体的详细描述和示例见附录 ??。此设计使我们能够实证量化提示复杂性如何影响中等准确性、计算开销以及对推理失败的易感性。

2.4 无免费午餐假设

实现有效护栏的难度源于对抗威胁和自然内容变异性。对手可以设计旨在绕过护栏的输入，例如通过操纵模型行为或使用改写或编码技巧来掩饰恶意意图的提示注入或模糊攻击。

分布偏移和伪有害场景同样具有挑战性，其中良性内容在词汇或主题上与有害类别重叠。例如，由于与有害内容存在表面相似性，医学或法律讨论可能会被错误地标记。

我们将这些挑战形式化为免费午餐假设对于护栏：任何试图最小化残差风险（方程 3）的努力都将不可避免地增加效用退化（方程 4）或可用性损失（方程 5），特别是在对抗压力或模糊输入的情况下。基于大语言模型的评估器虽然有希望，但并不能摆脱这种权衡，并引入了自己的脆弱性。

以下各节通过在对抗和伪伤害场景中测试具有代表性的护栏系统，实证评估这一假设，并量化安全、效用和易用性之间的内在权衡。

3 方法论

我们的方法旨在通过系统分析不同护栏架构在特定评估场景下的表现，实证验证无免费午餐（NFL）假设对于护栏。我们评估它们平衡三个相互竞争目标的能力：最小化残余风险、保持效用和避免可用性损失。本节详细介绍了数据集构建、评估协议、选定的度量标准以及指导研究的比较分析框架。

3.1 数据集设计和评估轴线

我们的评估依赖于两个专门设计的数据集，每个数据集针对在第 2 节定义的权衡的不同方面。它们共同使得能够全面评估防护栏行为在对抗压力、效用敏感任务以及可能降低可用性的过度过滤场景下的表现。

为了评估剩余风险——定义为有害内容绕过防护系统概率，我们整理了一个全面的对抗数据集，表示为 $\mathcal{D}_{\text{attack}}$ 。该数据集特别设计用于通过向每个防护措施呈现多样化和具有挑战性的有害输入来探究安全机制的故障模式。该集合包括多种攻击类型：(i) 直接危害查询会索要明显违反政策的输出，(ii) 混淆攻击利用改写、假设或编码提示来规避词汇过滤器，以及 (iii) 注入攻击明确旨在覆盖系统级指令和对齐约束。数据集中的每个样本都进行了标注以确定模型的完成是否构成政策违规行为，从而根据公式 3 进行未检测到的有害输出的精确估计。

我们的对抗基准来源于三个已建立的资源：SAGE 数据集 [Kumar et al. \(2024\)](#)，**野越狱** [Jiang et al. \(2024\)](#)，以及 XTRAM 的**安全防护提示注入**数据集⁵。为了扩展已知攻击之外的覆盖范围，我们还引入了两个新的对抗测试平台。首先，一个精选的目标集合**长上下文越狱提示**用于应对存在于许多防护措施中的上下文长度限制（例如 4096 个标记的截断），评估它们在关键指令被深入嵌入或分散时的行为。其次，我们编制了一个基准测试集**高级越狱提示**，其中包括能够成功绕过领先对齐调优模型安全层的已验证漏洞，这些模型包括 Claude (Anthropic)、GPT (OpenAI) 和 LLaMA (Meta)。这些补充确保了我们的评估涵盖了不断变化的安全威胁环境中的典型和前沿对抗行为。

同时，我们整理了 $\mathcal{D}_{\text{utility+usability}}$ 数据集，该数据集旨在共同评估护栏系统的两个关键属性：保持合法响应的效用的能力以及抵御过度过滤安全内容（通常表现为假阳性）的弹性。该数据集特意设计以捕捉在审核系统中信息性和谨慎性之间的内在张力。

具体来说，该数据集承担了两个相互关联的评估角色。首先，它包括一组多样化的知识密集型任务、编程挑战和多跳推理提示词样本，这些允许我们在内容审核下量化模型效用的下降。这些样本评估护栏是否不必要地抑制了与任务相关的内容，如公式 4 所形式化的那样。其次，它包含一组精选的伪和谐示例——从医疗保健、法律和社会政治讨论等敏感或高风险领域提取的良性提示。尽管这些内容是非恶意的，但由于它们经常包含与有害输入相似的关键字或语言模式，因此容易被误分类。根据公式 5，在这些情况下的护栏拒绝被视为可用性失败。

为了支持这种双重目的的评估，我们借鉴了多个来源。PH **检验**数据集 [An et al. \(2024\)](#) 提供了一个基准，用于设计触发误报的伪有害提示。我们进一步通过过滤掉明显有害的实例并手动验证其余部分来增强**防护栏 AI 检测越狱**⁶数据集，以模拟伪有害场景。最后，为了评估通

⁵<https://huggingface.co/datasets/xTRam1/safe-guard-prompt-injection>

⁶<https://huggingface.co/datasets/GuardrailsAI/detect-jailbreak>

用用途效用的保留情况，我们纳入了来自公共基准的提示，如**竞技场** Li et al. (2024)、**极棒的 ChatGPT 提示语**⁷ 和**无机器人**，这些涵盖了多样化的开放性用户意图和信息检索查询。

共同地， $\mathcal{D}_{\text{utility+usability}}$ 使得能够对保留有价值内容和避免过度审查之间的护栏权衡进行细致的原则性分析，特别是在挑战词汇或语义过滤器的边缘案例中。

重要的是，这种组合数据集结构反映了现实情况，在这种情况下，护栏必须同时保持输出质量并避免不必要的审查。除了审查准确性外，我们还衡量了每个护栏引入的计算开销，捕捉与审查相关的预期延迟 $\mathbb{E}[\Delta t]$ 。完整数据集的详细信息、收集方法和示例在附录??中提供。

3.2 评估方法论

我们的评估管道通过基础模型和每个护栏处理每个样本，系统地记录审核决策、效用分数和延迟测量。评估遵循以下结构化程序：

Algorithm 1 护栏评估管道

```

1: for each dataset  $\mathcal{D} \in \{\mathcal{D}_{\text{attack}}, \mathcal{D}_{\text{utility+usability}}\}$  do
2:   for each input sample  $x$  in  $\mathcal{D}$  do
3:      $y \leftarrow \mathcal{M}(x)$ 
4:     for each guardrail  $\mathcal{G}_i$  do
5:        $y' \leftarrow \mathcal{G}_i(y)$ 
6:       记录通过/拒绝决定
7:       测量并记录延迟  $\Delta t$ 
8:       计算特定任务的效用  $\mathcal{U}(y')$ 
9:       注释相对于 ground truth 的有害性或正确性
10:    end for
11:  end for
12: end for

```

对于基于 LLM 的护栏 ($\mathcal{G}_{\text{LLM}}$)，我们进一步研究了对系统提示设计的敏感性。具体来说，我们评估了三种提示策略：简单的调制指令、详细的特定任务说明以及链式思维 (CoT) 推理提示。这种变化使我们能够分析提示复杂度如何影响调制质量和计算成本。完整的提示设计方案见附录??。

3.3 评估指标

我们使用性能和效率指标的组合来评估所有护栏，以全面捕捉各种权衡。

加权 F1 分数。 我们报告加权 F1 分数作为评估在对抗性和实用及易用性基准上审核性能的主要指标。F1 分数提供了精确度（最小化假阳性）和召回率（捕捉真正有害案例）之间的平衡，这对评估实际护栏效果至关重要。与宏平均不同，加权变体调整了标签分布，为样本更多的类别赋予更大权重。这一点在现实世界设置中特别相关，在这种设置中，良性查询更为频繁，过度过滤可能会严重降用户体验。使用加权 F1 允许在不平衡数据集中进行细致的性能跟踪而不掩盖主要失败模式。

⁷<https://huggingface.co/datasets/flka/awesome-chatgpt-prompts>

延迟开销。 对于每个护栏，我们计算在调节过程中相对于未调节生成引入的平均额外评估时间 $E[\Delta t]$ 。该指标反映了计算可行性及对用户体验的影响，在实时应用中这一点尤为重要，因为过度延迟是不可接受的。

3.4 比较分析框架

我们的比较框架将这些指标应用于三种不同的护栏架构：提供商 API ($\mathcal{G}_P$)、基于 BERT 的分类器 ($\mathcal{G}_{\text{BERT}}$) 和基于 LLM 的评估器 ($\mathcal{G}_{\text{LLM}}$)。每个系统都在对抗攻击、以效用为中心的任务以及伪有害内容条件下进行测试。通过共同评估 F1 分数、延迟、PR 和 ROC 特性，我们提供了一个全面的观点，展示这些系统如何在安全、效用和可用性之间权衡取舍。

这种方法使我们能够实证观察每种护栏设计的结构限制。具体来说，它突出了减少残余风险的努力如何常常会降低效用或增加延迟，从而证实了 NFL 假设。我们的评估策略反映了理论和实际部署的关注点，为当前大型语言模型安全机制的状态及未来挑战提供了可操作的见解。

4 结果与讨论

我们提出了一种在三个核心维度上对护栏系统的比较评估：(i) 安全内容保留和效用保持，(ii) 对抗鲁棒性和越狱缓解，以及 (iii) 延迟和计算效率。我们的研究结果汇总在表 1 和 2 中，实证验证了本工作的中心论点的免费午餐假设对于护栏。

4.1 可用性和效用保留

护栏在保留良性、信息性和非有害内容的能力上存在很大差异。虽然大多数模型在通用任务（例如，竞技场，棒极了，无机器人）上的性能一直很高，但伪伤害数据集如 PH 检验和护栏人工智能揭示了过滤保守性方面的差异。

静态分类器如 llama-guard 和思维守护在伪有害提示和效用提示上都实现了很强的 F1 分数，但会导致高延迟（高达 3.7 秒）。轻量级分类器如 iad-v3 和 vijil-mbert-prompt-injection 以牺牲一些准确性来换取低于 0.1 秒的延迟，在交互设置中提供了更好的可用性。像 Enkrypt AI 的 guardrail api⁸ 和 moderation-api 这样的提供商 API 达到了一个有利的平衡，尽管它们偶尔在更敏感的例子表现出不一致。

基于 LLM 的护栏通过详细的提示和 CoT 风格的评估显示了对边缘案例处理的改进。然而，它们的好处被增加了延迟所削弱。例如，克劳德-3.5-十四行诗（推理）在无机器人和精彩极了上得分接近完美，但代价是 7.88 秒的延迟，这使得它们不适合用于对延迟要求严格的应用程序。

4.2 对抗鲁棒性

护栏在对抗压力下的表现揭示了系统类别之间的结构差异。开源系统如 nemo-guard 和 iad-v3 在所有攻击类型中均表现出强大的 F1 分数，同时保持实用的延迟。vijil-mbert-prompt-injection 在注入式攻击上表现出色，但在长上下文提示下表现不佳。

提供商 API 如 enkrypt-api 在所有五个对抗数据集上表现出高可靠性，特别是在长提示和高级越狱上，同时保持最小的延迟。相比之下，像 Azure 这样的模型表现不一致，在长提示 (0.773) 上得分很高，但在其他方面泛化能力较低（例如，SAGE: 0.010）。

⁸<https://app.enkryptai.com/guardrails/>

基于 LLM 的通过 CoT 进行的内容审查展示了最先进的对抗检测性能。克劳德-3.5-十四行诗（推理）和 Gemini-2.0-Flash（推理）在 SAGE、XTRAM 和 AJ 上超越了静态系统，但平均每个查询需要超过 8 秒。这些结果再次证实，在内容审查中，可解释性和语境细微差别带来了显著的计算成本。

4.3 延迟和系统效率

延迟分析突显了实际护栏部署中的一个关键瓶颈。虽然开源分类器和基于 API 的系统在小于 0.1 秒的情况下运行良好，例如，iad-v3(0.038 秒)，enkrypt-api(0.053 秒)，基于大语言模型的方法引入了数量级的延迟。详细的和 CoT 增强的提示将延迟增加到 8.6 秒，对于像克劳德-3.5-十四行诗（推理）和旋风大型（推理）这样的模型来说，这使得它们不适合用于响应速度至关重要的交互式应用。

4.4 关键点

实证结果证实了我们的假设：没有一种护栏配置能够同时实现最优的安全性、实用性和易用性。我们观察到以下一致的权衡：

- 针对对抗性召回优化的模型（例如，克劳德-3.5-十四行诗 CoT）会遭受严重的延迟和升高的假阳性率。
- 针对速度和易用性调整的系统（例如，提示守护,Azure）经常会错过微妙或隐晦的对抗性提示。
- 平衡解决方案（例如，nemo-guard,enkrypt-api）提供了具有竞争力的性能，但在分布变化或上下文过载的情况下仍面临退化。

这些结果证实了没有免费午餐假设对于护栏：由于自然语言审核的概率性、对抗性和模糊性，一个维度上的任何提升都会在另一个维度上引入可测量的妥协。本文其余部分提供了如何在实践中应对这些权衡的方法。

我们的评估确认了实际的护栏部署需要特定任务的校准。没有任何系统能够同时优化安全、实用性和效率，这强调了在大语言模型的安全机制中“没有免费午餐”假说的经验现实。

5 结论

在这项工作中，我们实证验证了无免费午餐假设对于护栏，证明通过护栏提高大语言模型部署的安全性不可避免地会在效用和可用性方面产生权衡。我们对供应商 API、基于 BERT 的分类器以及基于 LLM 的护栏在对抗场景、效用场景和伪伤害场景中的系统评估显示，虽然某些系统在过滤有害内容方面表现出色，但它们往往过度过滤或引入延迟，影响了实际应用的可用性。值得注意的是，我们将伪伤害检测识别为一种关键但未充分探索的故障模式，在这种情况下，由于与有害文本存在表面相似之处，无害的内容被错误地拒绝。此外，虽然带有详细提示和链式思考推理的基于 LLM 的审查提供了改进的鲁棒性，但它引入了显著的计算开销，限制了其实用性。这些发现表明，没有一种护栏架构可以普遍优化安全、效用和效率，强调在未来的大语言模型部署中需要自适应且情境感知的审查策略。

Type	Model	PHTest	GuardrailsAI	Arena	Awesome	NoRobots	Latency (s)
Open-Source	iad-v3	0.819	0.584	0.972	0.619	0.960	0.038
	nemo-guard	0.928	0.640	0.858	0.900	0.986	1.019
	llama-guard	0.971	0.860	0.975	0.972	0.995	1.540
	think-guard	0.945	0.895	0.994	0.995	0.994	3.692
	vijil-mbert-prompt-injection	0.841	0.686	0.944	0.980	0.958	0.076
Provider	guardrailsai	0.917	0.620	0.920	0.777	0.907	0.357
	moderation-api	0.896	0.863	0.995	0.998	0.955	0.259
	azure	1.000	0.811	1.000	0.995	1.000	0.068
	bedrock	0.938	0.553	0.950	0.112	0.978	0.197
	enkrypt-api	0.920	0.864	0.976	0.993	0.982	0.053
LLM-Based	gpt-4o	0.975	0.871	0.993	0.995	0.999	0.966
	gpt-4o-detailed	0.962	0.855	0.990	0.990	0.996	1.272
	gpt-4o-reasoning	0.956	0.860	0.985	0.988	0.999	4.295
	gemini-2.0-flash	0.963	0.874	0.996	0.998	0.999	0.579
	gemini-2.0-flash-detailed	0.957	0.887	0.995	0.998	0.998	0.421
	gemini-2.0-flash-reasoning	0.821	0.820	0.971	0.972	0.990	1.837
	mistral-large-latest	0.951	0.868	0.992	1.000	1.000	1.341
	mistral-large-latest-detailed	0.882	0.852	0.973	0.988	0.997	1.752
	mistral-large-latest-reasoning	0.825	0.780	0.940	0.956	0.986	8.856
	claude-3-5-sonnet-latest	0.873	0.836	0.985	0.990	0.997	2.161
	claude-3-5-sonnet-latest-detailed	0.867	0.832	0.984	0.987	0.994	4.013
	claude-3-5-sonnet-latest-reasoning	0.857	0.814	0.989	0.972	0.996	7.881

表 1: F1 分数和延迟时间在基准数据集 (PHTest、GuardrailsAI、Arena、Awesome ChatGPT Prompts、No Robots) 上用于评估模型使用性过滤。延迟表示平均响应时间，以秒为单位。(Nemo Guardrails 使用 gpt-4o-mini 内核模型进行评估)

Type	Model	SAGE	WildJailbreak	Long	AJ	XTRAM	Latency(s)
Open-Source	enkryptai-iad-v3	0.896	0.857	0.792	0.991	0.950	0.042
	nemo-guard	0.978	0.917	0.840	0.974	0.937	0.690
	llama-guard	0.897	0.636	0.494	0.622	0.868	0.959
	think-guard	0.963	0.772	0.457	0.748	0.913	0.524
	vijil-mbert-prompt-injection	0.938	0.887	0.451	0.851	0.996	0.056
Provider	guardrailsai	0.600	0.619	0.710	0.662	0.819	0.348
	moderation-api	0.835	0.608	0.432	0.706	0.702	0.283
	azure	0.010	0.162	0.773	0.077	0.816	0.070
	bedrock	0.922	0.785	0.819	0.994	0.854	0.203
	enkrypt-api	0.904	0.843	1.000	0.986	0.941	0.048
LLM-Based	gpt-4o	0.940	0.819	0.463	0.903	0.848	1.040
	gpt-4o-detailed	0.972	0.909	0.541	0.961	0.915	1.474
	gpt-4o-reasoning	0.964	0.897	0.506	0.949	0.922	4.816
	gemini-2.0-flash	0.925	0.778	0.432	0.823	0.846	0.420
	gemini-2.0-flash-detailed	0.967	0.862	0.482	0.915	0.874	0.593
	gemini-2.0-flash-reasoning	0.983	0.895	0.625	0.949	0.936	1.863
	claude-3-5-sonnet-latest	0.982	0.906	0.546	0.983	0.933	2.199
	claude-3-5-sonnet-latest-detailed	0.987	0.927	0.574	0.990	0.945	4.493
	claude-3-5-sonnet-latest-reasoning	0.986	0.895	0.769	0.988	0.960	8.588
	mistral-large-latest	0.961	0.782	0.512	0.839	0.908	1.277
	mistral-large-latest-detailed	0.978	0.929	0.584	0.963	0.932	1.644
	mistral-large-latest-reasoning	0.979	0.842	0.610	0.789	0.932	8.689

表 2: F1 分数和平均延迟时间用于五种攻击类型 (SAGE、WildJailbreak、Long、AJ、XTRAM) 的对抗鲁棒性测试。延迟表示以秒为单位的平均响应时间。(Nemo Guardrails 使用 gpt-4o-mini 核心模型进行评估)

参考文献

Bang An, Sicheng Zhu, Ruiyi Zhang, Michael-Andrei Panaitescu-Liess, Yuancheng Xu, and Furong Huang. Automatic Pseudo-Harmful Prompt Generation for Evaluating False

- Refusals in Large Language Models. arXiv, September 2024. doi: 10.48550/arXiv.2409.00598.
- Suriya Ganesh Ayyamperumal and Limin Ge. Current state of LLM Risks and AI Guardrails. arXiv, June 2024. doi: 10.48550/arXiv.2406.12934.
- Hayder Elesedy, Pedro M Esperanca, Silviu Vlad Oprea, and Mete Ozay. LoRA-guard: Parameter-efficient guardrail adaptation for content moderation of large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pp. 11746–11765, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.656. URL <https://aclanthology.org/2024.emnlp-main.656/>.
- Liwei Jiang, Kavel Rao, Seungju Han, Allyson Ettinger, Faeze Brahman, Sachin Kumar, Niloofar Mireshghallah, Ximing Lu, Maarten Sap, Yejin Choi, and Nouha Dziri. Wildteaming at scale: From in-the-wild jailbreaks to (adversarially) safer language models, 2024. URL <https://arxiv.org/abs/2406.18510>.
- Melissa Kazemi Rad, Huy Nghiem, Sahil Wadhwa, Andy Luo, and Mohammad Shahed Sorower. Refining input guardrails: Enhancing LLM-as-a-judge efficiency through chain-of-thought fine-tuning and alignment. In AAAI 2025 Workshop on Preventing and Detecting LLM Misinformation (PDLIM). AAAI, Feb 2025. URL <https://openreview.net/forum?id=UNPzbCKovl>.
- Mahi Kolla, Siddharth Salunkhe, Eshwar Chandrasekharan, and Koustuv Saha. LLM-Mod: Can Large Language Models Assist Content Moderation? In ACM Conferences, pp. 1–8. Association for Computing Machinery, New York, NY, USA, May 2024. doi: 10.1145/3613905.3650828.
- Anurakt Kumar, Divyanshu Kumar, Jatan Loya, Nitin Aravind Birur, Tanay Baswa, Sahil Agarwal, and Prashanth Harshangi. SAGE-RT: Synthetic Alignment data Generation for Safety Evaluation and Red Teaming. arXiv, August 2024. doi: 10.48550/arXiv.2408.11851.
- Seanie Lee, Dong Bok Lee, Dominik Wagner, Minki Kang, Haebin Seong, Tobias Bocklet, Juho Lee, and Sung Ju Hwang. SafeRoute: Adaptive Model Selection for Efficient and Accurate Safety Guardrails in Large Language Models. arXiv, February 2025. doi: 10.48550/arXiv.2502.12464.
- Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. From Live Data to High-Quality Benchmarks: The Arena-Hard Pipeline. arXiv, April 2024. URL <https://lmsys.org/blog/2024-04-19-arena-hard/>.
- Weidi Luo, Shenghong Dai, Xiaogeng Liu, Suman Banerjee, Huan Sun, Muhao Chen, and Chaowei Xiao. AGrail: A Lifelong Agent Guardrail with Effective and Adaptive Safety Detection. arXiv, February 2025. doi: 10.48550/arXiv.2502.11448.
- Sejoon Oh, Yiqiao Jin, Megha Sharma, Donghyun Kim, Eric Ma, Gaurav Verma, and Srijan Kumar. UniGuard: Towards Universal Safety Guardrails for Jailbreak Attacks on Multimodal Large Language Models. arXiv, November 2024. doi: 10.48550/arXiv.2411.01703.

Zhuowen Yuan, Zidi Xiong, Yi Zeng, Ning Yu, Ruoxi Jia, Dawn Song, and Bo Li. RigorLLM: Resilient guardrails for large language models against undesired content. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), Proceedings of the 41st International Conference on Machine Learning, volume 235 of Proceedings of Machine Learning Research, pp. 57953–57965. PMLR, 21–27 Jul 2024. URL <https://proceedings.mlr.press/v235/yuan24f.html>.