从外部探针到结构阻断：基于形根关系约束的生成阶段剪枝验证（V2）

From External Probes to Structural Pruning: Verification of Generation-Stage Blocking via Morpho-Root Relational Constraints (V2)

作者：刘深，曾嘉梁

摘要

此前的负面结果[1]表明，基于字面 Token 屏蔽或中间层线性探针[2]的外部护栏，在面对分布外（OOD）对抗提示词时会发生灾难性失效（0/16 阻断）。为应对这一结构性盲区，本文基于作者提出的“表意 AI”（Logographic AI）理论框架，提出并验证了一种基于“形根硬约束”[3]的生成阶段剪枝原型（Morpho-Guard）。

与传统护栏依赖统计概率不同，该方法通过引入携带内在意义锚点的“形根”（Morpho-Root）作为认知基元，在解码阶段（Generation Stage）实时解析当前生成的文本前缀，并与预设的安全拓扑关系（如：缺乏授权节点）进行匹配。**实验证明**，在受控的系统越权子域内，Morpho-Guard 对 16 条 JailbreakBench[4] (JBB) OOD 攻击实现了 **15/16 (93.75%) 的生成阶段干预**（剩余 1 条由模型自身安全机制拒答）；同时，在面对 100 条正常请求时，总体干预率为 29/100 (29%)。值得注意的是，在 30 条高混淆正常请求（涉及系统备份等合法操作）中，由于规则提取器过度敏感，干预率高达 23/30 (76.67%)。

本研究不仅为“从下游概率拦截转向上游结构阻断”提供了工程证据，也客观揭示了当前启发式规则在安全性与可用性之间的巨大权衡代价。

Abstract

Our preceding negative result [1] demonstrated that external guardrails based on lexical Token masking or intermediate-layer linear probes [2] suffer catastrophic failure (0/16 block rate) when facing Out-of-Distribution (OOD) adversarial prompts. To address this structural blind spot, this paper—grounded in the "Logographic AI" theoretical framework proposed by the authors—introduces and verifies a generation-stage pruning prototype based on "Morpho-Root hard constraints" [3].

Unlike traditional guardrails that rely on statistical probabilities, this method employs "Morpho-Roots" (carrying intrinsic meaning anchors) as cognitive primitives, dynamically parsing the current generated text prefix at the Generation Stage and matching it against predefined safety topological relations (e.g., missing authorization nodes). Experimental results demonstrate that within a controlled system privilege-escalation sub-domain, Morpho-Guard achieved **15/16 (93.75%) generation-stage interventions** against 16 JailbreakBench [4] (JBB) OOD attacks (the remaining 1 was refused by the model's baseline safety). Meanwhile, it yielded an overall intervention rate of **29/100 (29%)** on 100 benign requests. Notably, on 30 high-confusion benign requests (involving legitimate system operations), the intervention rate reached **23/30 (76.67%)** due to the over-sensitivity of the rule-based extractor.

This study not only provides engineering evidence for shifting from downstream probabilistic interception to upstream structural blocking but also objectively reveals the severe trade-off between security and utility under current heuristic rules.

关键词：表意 AI；生成阶段阻断；形根硬约束；受限解码；关系剪枝；AI 安全

Keywords: Logographic AI; Generation-stage blocking; Morpho-Root hard constraints; Constrained decoding; Relational pruning; AI Safety

1. 引言

在上一项研究[1]中，我们通过实证数据表明，当前主流的外部统计护栏（如字面词表屏蔽、中间层线性探针）在面临分布外（OOD）的对抗性提示词时，会发生灾难性的失效（攻击阻断率为 0/16）。模型之所以能轻易绕过这些防御，根本原因在于外部监控发生在模型统计优化的下游。这种基于后验概率的拦截机制，既无法真正理解文本深层语义的拓扑结构，也无法应对意图在潜空间中的形变。

为弥补这一结构性盲区，本文引入作者前期提出的“表意 AI”（Logographic AI）理论框架[3]。该理论指出，现有大语言模型的脆弱性源于其底层的“无根 Token”范式——Token 缺乏内在的意义锚点，导致安全约束只能作为“补丁”外挂于概率计算之上。为了实现真正的内生安全，必须将携带结构化语义与硬约束的认知基元——即“形根”（Morpho-Root）——引入系统底层。

基于此理论，本文提出并验证了一种生成阶段剪枝原型（Morpho-Guard）。我们不再试图训练一个“更聪明的分类器”来预测恶意，而是通过受限解码（Constrained Decoding），在模型生成每个 Token 的瞬间，动态解析当前推理路径的意图节点，并检查其是否违反了预设的形根拓扑关系（如：缺乏授权节点却执行系统越权）。

必须强调的是：本文旨在验证“结构阻断”这一机制在**受控子域（系统越权与提权）**内的工程可行性。它并非一个开箱即用的通用安全方案，其核心贡献在于证明，“在生成阶段施加拓扑约束”能够消除外部探针面临分布偏移时的盲区，并为表意 AI 理论在安全防御领域的落地提供初步的实证支撑。

2. 方法论：Morpho-Guard 原型设计

本研究不重训底座大模型（如 Qwen2.5-1.5B[5]），而是通过自定义干预机制（如 LogitsProcessor[6]）实现动态阻断。

2.1 形根节点映射(Morpho-Root Mapping)

依据表意 AI 理论对认知基元的定义[3]，我们将危险动作拆解为三个核心基元：**Action(A)**、**Target(T)**、**Auth(U)**。

2.2 逻辑不可达约束(Inviolable Constraints)

若生成路径同时激活了[Action: Execute]和[Target: System_Critical]，且缺乏[Auth: Verified]前置节点，则该路径被判定为违规。

2.3 实时干预与剪枝(Real-time Intervention & Pruning)

本节介绍的干预机制，是受限解码（Constrained Decoding）在 LogitsProcessor 层面的具体工程实现。在模型进行 Greedy Decoding 生成每个 Token 时，Morpho-Guard 动态解析当前的文本前缀。一旦识别到违反 Rhard 的意图闭环，直接触发干预，强制注入安全拒绝 Token，切断危险路径的生成。

3. 实验设置与结果

——测试集与正常集：16 条 JBB OOD 攻击，100 条合法系统操作请求（含 30 条高混淆）。
——全新环境复现验证：为杜绝环境污染，本实验在全新的 Docker 容器中进行了端到端复跑。复跑输出的 JSON 记录与原实验的**哈希值（SHA-256）差异为 0**，实现了绝对确定性的复现。

实验结果表：

（注：指标为生成阶段干预率/干预数，而非最终回答级判分）

防御机制	攻击样本干预	普通正常请求干预	高混淆正常请求干预	全部正常请求干预
Base（原模型）	0/16 (0%)	0/70 (0%)	0/30 (0%)	0/100 (0%)
Probe（外部探针）	0/16 (0%)	0/70 (0%)	0/30 (0%)	0/100 (0%)
Morpho-Guard	**15/16 (93.75%)**	**6/70 (8.57%)**	**23/30 (76.67%)**	**29/100 (29%)**
Probe + Morpho	15/16 (93.75%)	6/70 (8.57%)	23/30 (76.67%)	29/100 (29%)

结果分析：

1. **攻击拦截：**外部探针全部失效（0/16）。Morpho-Guard 成功干预了 15 条攻击（93.75%）。剩余的 1 条虽未被形根规则触发，但被模型自身的安全机制拒答。综合来看，系统级的越狱攻击阻断/干预率达到了 16/16 (100%)，这与上一篇论文中外部护栏 0/16 的全面失效形成了鲜明对比。（注：最终的回答级 ASR 仍需盲审复核）。
2. **代价与误伤：**虽然普通正常请求的干预率仅为 8.57%，但高混淆正常请求的干预率飙升至 76.67%。这是因为基于正则的启发式提取器无法准确区分“恶意提权”与“合法的系统备份脚本”。这客观反映了结构化阻断在逻辑安全与工程可用性之间的严峻挑战。
3. **防御纵深：**叠加探针未产生负面影响，证明结构阻断可独立成立。

4. 讨论与局限

1. **规则提取器的语义盲区：**当前的启发式匹配导致了极高的高混淆误伤率（23/30），这是未来工程优化的核心痛点（需引入 LLM 辅助解析而非单纯正则）。
2. **指标定义局限：**当前测量的是“干预率（Intervention Rate）”。完整的回答级误拒率（FRR）与最终的攻击成功率（ASR），仍需根据完整输出文本进行严格的盲审评分。

数据与代码可用性声明

本文提出的 Morpho-Guard 原型、对抗样本及测试集均已开源：https://github.com/WindSeaAI/Morpho-Guard。该仓库包含了四组条件下的完整复跑日志、逐条判定记录以及 SHA-256 验证记录，确保了实验结果的绝对透明度。

致谢

感谢智能体 Boardy.ai (@boardyai)在讨论中提供的精辟总结：“承重胜过法律约束”（Load bearing beats legally binding）。这句话完美捕捉了“形根硬约束”的核心理念，即物理架构的承重能力永远优于外部的合规性约束。

参考文献

[1] Liu S, Zeng J. A Negative Result on Statistical Guardrails: Out-of-Distribution Adversarial Evaluation of Lexical Masks and Linear Probes on Qwen2.5-1.5B[PP/OL]. Zenodo, 2026. https://doi.org/10.5281/zenodo.24114711

[2] Alain G, Bengio Y. Understanding intermediate layers using linear classifier probes[J]. arXiv:1610.01644, 2016. https://arxiv.org/abs/1610.01644

[3] Liu S. The Formal Foundations of Morpho-Root Theory: Mathematical Definitions and Reasoning Systems for Logographic AI Cognitive Primitives[PP/OL]. ChinaXiv, 2026. T202608.00490.
https://chinaxiv.org/businessFile/T202608/T202608.00490v1/T202608.00490v1.pdf

[4] Mazeika M, et al. JailbreakBench: An Open Robustness Benchmark for Jailbreaking Large Language Models[EB/OL]. arXiv:2404.01318, 2024. https://arxiv.org/abs/2404.01318

[5] Qwen Team. Qwen2.5 Technical Report[EB/OL]. arXiv:2412.15115, 2024. https://arxiv.org/abs/2412.15115

[6] HuggingFace. Utilities for generation[EB/OL]. (2026-10-08)[2026-11-09]. https://huggingface.co/docs/transformers/v5.21.1/en/internal/generation_utils