Defensive M2S：在压缩后的多轮对话上训练安全防护模型

Hyunjun KimKAIST
hyunjun1121@kaist.ac.kr

arXiv:2601.00454v1 [cs.CL] 1 Jan 2026

摘要

安全防护模型对于确保大语言模型（LLM）部署的安全至关重要，但处理完整的多轮对话历史会产生巨大的计算成本。我们提出**DefensiveM2S**，这是一种训练范式，通过在多轮转单轮（M2S）压缩后的对话上微调安全防护模型，而非使用完整的对话历史。

我们提供了形式化复杂度分析，表明对于 n 轮对话，M2S 可将训练成本从 $O(n^2)$ 降至 $O(n)$。在我们的训练数据集（779 个样本，平均 10.6 轮）上进行的实证评估显示，M2S 仅需**169K 个令牌**，而多轮基线需要**15.7M 个令牌**，降低了 **93**× 。

我们在 SafeDialBench（一个全面的多轮越狱基准）上，使用三种安全防护模型系列（LlamaGuard、Nemotron、Qwen3Guard）和三种压缩模板（连字符化、数字化、Python 化）评估 Defensive M2S。表现最佳的配置——采用连字符压缩的 Qwen3Guard——实现了**93.8%** 的攻击检测召回率，同时将推理令牌数减少**94.6%**（每次对话从 3,231 个令牌降至 173 个令牌）。相较于基线，这一结果提升了 38.9 个百分点，同时大幅降低了训练和推理成本。

我们的研究结果表明，M2S 压缩可作为一种有效的护栏部署效率技术，从而能够对较长的多轮对话进行可扩展的安全筛查。

1 引言

大型语言模型（LLM）在各种任务中展现出卓越的能力，但它们容易受到对抗攻击，这仍然是一个关键问题。在这些威胁中，多轮越狱攻击尤为阴险：攻击者通过一系列精心设计的对话轮次，逐步操纵大语言模型，绕过安全护栏并诱导其产生有害输出。

安全防护模型是关键的防御机制，充当分类器，用于评估给定的输入输出对是否安全。然而，将这些模型部署到多轮对话中会带来显著的计算挑战：处理完整对话历史需要大量词元吞吐量，导致推理时间的延迟和成本增加。随着对话不断变长，计算负担也会线性增长，使实时安全筛查的成本越来越高。

近期关于多轮转单轮（*M2S*）压缩（Ha 等人，2025年）的研究表明，多轮越狱攻击可以提炼为紧凑的单轮提示，同时保留其对抗效果。这一发现从安全视角来看令人担忧，但也揭示了一种颇具吸引力的防御应用：如果能以压缩形式捕捉多轮攻击的核心语义，或许就可以训练安全防护模型直接识别这些压缩表示。

在本文中，我们提出**DefensiveM2S**，这是一种训练范式，通过在 M2S 压缩后的对话历史上微调安全防护模型，而非完整的多轮对话，来训练模型。我们的核心假设是，M2S 压缩能够保留准确进行安全分类所必需的语义信息，同时大幅降低推理的计算成本。

我们通过在三个安全防护模型系列（LlamaGuard、Nemotron 和 Qwen3Guard）上，针对多种 M2S 压缩模板（连字符化、数字化、Python 化）开展广泛实验，验证了这一假设。我们在 Safe-DialBench（Chen 等人，2025年）上进行评估。Safe- DialBench 是一个全面的多轮越狱基准，包含 2,037 个样本，涵盖 6 个攻击类别和 7 种攻击

方法，得出了几个关键发现：

•**效率-准确率权衡**：经过 M2S 训练的模型可将令牌缩减率提高至 94.6%，同时保持有竞争力的检测准确率。最佳配置（采用连字符化模板的 Qwen3Guard）召回率达到 93.8%，高于 54.9% 的基线召回率，表明压缩实际上可以提升某些模型-模板组合的检测性能。

• **模型-模板敏感性**：M2S训练的效果在不同模型-模板组合之间差异显著。Qwen3Guard 更适合连字符化模板（召回率为 93.8%），而 Nemotron 使用数字化模板时表现最佳（召回率为 87.8%）。

•**单一模板的优势**：使用单一压缩模板进行训练的效果优于混合模板训练，这表明，与多样但不一致的格式相比，特定模板的表征能提供更强的学习信号。

我们的贡献总结如下：

1. 我们提出了 Defensive M2S，这是一种新颖的训练范式，利用对抗性压缩技术实现高效的安全防护部署。

2. 我们提供形式化复杂度分析，表明 M2S 将训练成本从 $O(n^2)$ 降至$O(n)$，并通过在我们的数据集上实现 $93\times$ 令牌缩减进行了实证验证。

3. 我们首次对采用 M2S 训练的护栏进行了系统评估，涵盖多个模型系列、压缩模板和评估基准。

4. 我们公开训练好的适配器和评估代码，以促进高效大语言模型安全领域的可复现研究。

2 相关工作

2.1 多轮越狱攻击

多轮越狱攻击利用大语言模型的对话特性，通过一系列看似无害的提示词逐步诱导其生成有害输出。Russinovich 等人（2024年）提出了 Crescendo 是一种多轮攻击，起初看似无害，随后通过引用模型回复逐步升级，在 GPT-4 上实现了 56% 的攻击成功率（攻击成功率），在 Gemini-Pro 上实现了 83% 的攻击成功率。Ren 等人（2024年）提出了 ActorAttack，该方法将语义关联实体建模为攻击线索，以生成多样化的攻击路径，在不同对话轮次中隐藏恶意意图。

自动化红队测试方法已应运而生，用于系统地发现漏洞。GOAT（Pavlova 等人，2024年）在多轮对话中模拟对抗性用户推理，在 Llama 3.1 上实现了 97% 的攻击成功率。TAP（Mehrotra 等人，2024年）采用思维树推理，并结合攻击者-评估者-目标大语言模型流程，实现了 80%+的攻击成功率，同时绕过了 LlamaGuard 等安全防护。WildTeaming（Jiang 等人，2024年）挖掘真实的用户与聊天机器人交互数据，发现了 5.7K 个独特的越狱策略簇。

单轮攻击为理解对抗鲁棒性奠定了基础。GCG（Zou 等人，2023年）率先对抗性后缀进行令牌级优化，通过跨模型迁移实现了 88% 的攻击成功率。AutoDAN（Liu 等人，2024年）使用遗传算法生成语义明确的越狱攻击，而 PAIR（Chao 等人，2023年）则通过迭代优化提示词，在不到 20 次查询内实现黑盒攻击。

与我们的工作最相关的是，M2S（Ha 等人，2025年）提出了多轮到单轮压缩方法，使用连字符化、数字化和 Python 化模板，将多轮越狱攻击整合为结构化的单轮提示。其研究表明，压缩后的提示词在利用原生护栏和外部护栏的“上下文盲区”时，攻击成功率往往比原始多轮攻击高出最多 17.5%。我们从防御角度借鉴了这一发现：如果 M2S 压缩保留了对抗性语义，那么在压缩表示上训练的护栏应能维持检测准确率。

2.2 护栏模型

基于大语言模型的护栏已成为安全分类的主流范式。Llama- Guard（Inan 等人，2023年）开创了将大语言模型微调为输入输出安全防护的先河，并引入了包含六个类别的安全分类体系，对新政策具有很强的适应性。后续版本（LlamaGuard 2/3）支持 MLCommons 分类体系（Vidgen 等人，2024年）。Nemotron Safety Guard（Ghosh 等人，2024年）进一步拓展了这一方法，纳入 13 个关键风险类别以及“需谨慎”标签，用于更精细的内容审核。

进的内容审核。

近期研究从多个维度提升了安全防护能力。WildGuard（Han 等人，2024年）同时实现了三项目标：识别恶意提示词、检测回复中的安全风险，以及衡量拒绝率；在拒绝检测方面，其表现比 Llama-Guard2 高出 25.3%。ShieldLM（Zhang 等人，2024年）引入了支持自定义规则和解释的双语（中文/英文）检测功能。ShieldGemma（Zeng 等人，2024年）通过新颖的合成数据生成方法，展现出优异的性能（比 LlamaGuard 高出 +10.8% AU-PRC ）。

参数高效适配使模型能够部署于资源受限的环境中。LoRA-Guard（Kim 等人，2024年）通过大语言模型与护栏之间的知识共享，实现了 100-1000×的较低参数开销。NeMo Guardrails（Rebedea 等人，2023年）为可控的大语言模型应用提供了可编程护栏。我们的工作通过 M2S 压缩减少输入所需的词元数量，与这些提升效率的方法相辅相成。

2.3 多轮对话安全

评估多轮场景中的安全性面临着独特挑战。SafeDialBench（Chen 等人，2025年）提供了涵盖六个安全维度的 4,053 段对话，包含七种越狱策略，其中包括指代攻击。CoSafe（Yu 等人，2024年）研究基于共指的攻击，结果显示不同模型的攻击成功率为 14% 至 56%。GuardBench（Bassani 和 Sanchez，2024年）整合了 40 个评估数据集，用于系统性地比较安全防护。

多个数据集为安全研究提供了支持。Beaver-Tails（Ji 等人，2023年）提供了 33.3 万个问答对，涵盖 14 个类别，并分别标注无害性和有用性。ToxicChat（Lin 等人，2023年）收录了来自 Vicuna 演示版的真实用户与 AI 交互内容。HarmBench（Mazeika 等人，2024年）通过 510 种行为和 18 种攻击方法，规范化了红队评估。

过度拒绝这一挑战也受到关注。XSTest（Röttger 等人，2024年）指出，模型会因安全提示词与有害内容在词汇上相似而拒绝安全提示词，表现出过度的安全行为。我们的方法通过在压缩表示上进行训练，间接应对这一问题；这种表示可能会滤除表层词汇模式，同时保留语义安全信号。

2.4 高效自然语言处理推理

提示压缩技术可降低大语言模型推理的词元开销。LLMLingua（Jiang 等人，2023年）通过由粗到细的词元剪枝，最高可实现 20× 的压缩。LongLLMLingua（Jiang 等人，2024年）将这一方法扩展至长上下文，在提升性能的同时降低了 94% 的成本。这些方法侧重于通用语言建模；我们的 M2S 压缩则专门用于保留与安全相关的语义。

KV缓存优化解决了内存瓶颈。H2O（Zhang 等人，2023年）通过重击者检测淘汰非必要的缓存状态，实现了 29× 吞吐量提升。StreamingLLM（Xiao 等人，2024年）通过注意力汇实现无限序列处理。FlashAttention（Dao 等人，2022年）提供了具备 I/O 感知能力、内存占用随序列长度线性增长的精确注意力。

其他架构可带来渐近性能提升。Mamba（Gu 和 Dao，2023年）实现了线性 $O(n)$ 复杂度，而 Transformer 的复杂度为二次 $O(n^2)$ ，且具有 5×更高的吞吐量。然而，这些方法需要改变架构；我们的 Defensive M2S 与模型无关，可应用于任何基于 Transformer 的安全防护模型。

我们的工作与以往的压缩方法有两点关键区别：（1）我们使用结构化模板在语义层面进行压缩，而非采用令牌级剪枝；（2）我们在训练期间而非推理期间进行压缩，使模型能够直接从压缩表示中学习安全相关特征。

3 方法

3.1 问题表述

令 $C = \{(u_1, a_1)、(u_2, a_2).....、(u_n, a_n)\}$ 表示一个包含 n 轮次的多轮对话，其中 u_i表示第 i 轮的用户消息，a_i 表示助手响应。安全防护模型$\mathcal{G}$ 是一个分类器，在给定对话上下文的情况下，预测安全标签 $y \in${安全，不安全} 。

在传统的基线方法中，安全防护模型会处理完整对话：

$$y = \mathcal{G}(C) = \mathcal{G}(u_1, a_1, \ldots, u_n, a_n) \quad (1)$$

计算成本会随着总词元数 $|C| = \sum_{i=1}^{n}($ $|u_i| + |a_i|)$ 增长，对于较长的对话，成本可能高得难以承受。

3.2 M2S 压缩

多轮到单轮（M2S）压缩会将多轮对话转换为紧凑的单轮表示。给定压缩函数 f_θ：

$$\tilde{C} = f_\theta(C) \tag{2}$$

其中，$\tilde{C}$ 是压缩表示，$|\tilde{C}| \ll |C|$。

我们研究了先前工作中的三种压缩模板（Ha 等人，2025年）：

连字符化模板 将用户轮次格式化为项目符号列表：

```
- [第 1 轮内容]- [
第 2 轮内容]... - [
第 n 轮内容]
```

编号化模板 将用户轮次格式化为编号列表：

```
1. [第 1 轮内容]2. [
第 2 轮内容]... n. [
第 n 轮内容]
```

Python化模板 以类似 Python 代码的结构对话进行格式化：

```
def conversation(): 用户_轮次_1 = "[
第 1 轮内容]" 用户_轮次_2 = "[第 2 轮
内容]" ... 用户_轮次_n = "[第 n 轮内容]
"
```

M2S 压缩的一项关键设计选择是仅提取用户轮次，丢弃助手响应。这一选择基于两点观察：(1) 对抗性意图主要编码在用户消息中；(2) 助手响应会带来大量词元开销，却未能提供相称的安全相关信息。

3.3 Defensive M2S 训练

我们提出在 M2S 压缩后的输入上训练安全防护模型，而非使用完整对话。给定一个训练数据集 $\mathcal{D} = \{(C_i, y_i)\}_{i=1}^N$，我们创建一个压缩后的训练集：

$$\tilde{\mathcal{D}} = \{(f_\theta(C_i), y_i)\}_{i=1}^N \tag{3}$$

随后对安全防护模型进行微调，以最小化交叉熵损失：

$$\mathcal{L} = -\sum_{i=1}^{N} y_i \log \mathcal{G}(\tilde{C}_i) + (1 - y_i) \log(1 - \mathcal{G}(\tilde{C}_i)) \tag{4}$$

3.4 计算复杂度分析

Defensive M2S 的一项关键优势是在数据生成 和训练期间大幅降低计算成本。为了说明这一优势，我们首先考察实际中多轮越狱攻击的生成方式。

多轮攻击生成分类近期文献揭示了构建多轮越狱攻击的两种截然不同的范式：

(1) 依赖响应的方法通过参考目标模型先前的响应，动态生成用户提示。Crescendo（Russinovich 等人，2024年）“利用了大语言模型倾向于遵循模式的特点……尤其是由大语言模型自身生成的文本。”类似地，PAIR（Chao 等人，2023年）“通过在聊天历史中累积先前的尝试和响应，迭代优化候选提示词”，而 TAP（Mehrotra 等人，2024年）则在此基础上扩展出基于树的探索。其他示例还包括 ActorAttack（Ren 等人，2024年），它“根据目标模型的响应动态调整攻击路径”。

(2) 预先编写的方法预先生成所有用户提示，无需模型响应。Ha 等人使用的 MHJ（多轮人工越狱）数据集Ha 等人（2025年）由预先编写的用户轮次构成。多样本越狱（Anil 等人，2024年）包含完全虚构、未经过实际模型交互的“伪对话”。

对训练数据的影响 对于预先编写的攻击，只有用户轮次存在——必须生成助手响应，才能为传统安全防护创建训练数据。我们的 M2S 方法完全消除了这一要求。

对于依赖响应的攻击，虽然在创建攻击时已有响应，但在以下情况下可能需要重新生成：(1) 将攻击适配到不同的目标模型；(2) 使用特定聊天格式创建训练数据；或 (3) 为不同于攻击目标的模型系列构建护栏。因此，即使对于依赖响应的数据集，基线复杂度分析通常仍然适用。

形式化复杂度分析 设 U 表示每个用户轮次的平均令牌数，R 表示每个助手响应的平均令牌数。对于一个 n轮对话：

多轮基线复杂度 基线方法需要经历两个成本高昂的阶段：

第 1 阶段：训练数据生成。 要在完整对话上训练安全防护模型，我们必须通过调用大语言模型，为每一轮生成助手响应。关键在于，每次生成响应都需要完整的前置上下文：在第 k 轮，大语言模型会接收此前所有用户轮次和生成的响应：

$$\text{Input}_k = \sum_{i=1}^{k} U + \sum_{i=1}^{k-1} R = kU + (k-1)R \quad (5)$$

生成所有 n 回复所需的输入词元总数：

$$T_{\text{gen}} = \sum_{k=1}^{n} (kU + (k-1)R) = \frac{n(n+1)}{2}U + \frac{n(n-1)}{2}R \quad (6)$$

第 2 阶段：安全防护训练。 为了在对话的任何阶段检测攻击，必须使用增量前缀训练安全防护模型。样本 k 包含k 轮次：

$$\text{Sample}_k = k(U+R) \quad (7)$$

训练令牌总数：

$$T_{\text{train}} = \sum_{k=1}^{n} k(U+R) = \frac{n(n+1)}{2}(U+R) \quad (8)$$

多轮成本总计：

$$T_{\text{baseline}} = T_{\text{gen}} + T_{\text{train}} = O(n^2) \quad (9)$$

M2S 复杂度 相比之下，M2S无需**生成回复**。由于 M2S 只提取用户轮次并将其压缩为结构化格式，我们可以直接使用现有的越狱提示词，而无需查询任何大语言模型。唯一的成本是压缩后的训练样本本身：

$$T_{\text{M2S}} = nU + O(1) \approx nU = O(n) \quad (10)$$

这样就完全省去了第 1 阶段（$T_{\text{gen}} = 0$），并将第 2 阶段缩减为每段对话只需一个样本，而非 n 个增量样本。

复杂度比率基线与 M2S 复杂度之比：

$$\frac{T_{\text{baseline}}}{T_{\text{M2S}}} = \frac{O(n^2)}{O(n)} = O(n) \quad (11)$$

这意味着，M2S 的效率优势会随着对话长度而增加。对于包含 $U \approx R$ 的 10 轮对话，理论比率约为 $21\times$。

3.5 训练配置

我们采用 QLoRA 微调（Dettmers 等人，2023年）进行参数高效适配。关键超参数包括：

- **基础模型**：LlamaGuard-3-8B、Nemotron-Safety-Guard-8B、Qwen3Guard-Gen-8B
- **LoRA 配置**：秩 $r = 16$，$\alpha = 32$，丢弃率 $p = 0.1$
- **量化**：4 位 NormalFloat（NF4）
- **训练**：批次大小 4，梯度累积 4，学习率 2×10^{-4}，3 个训练轮数
- **Seeds**: 42, 123, 456 (for statistical robustness)

3.6 推理协议

在推理时，给定一个新的多轮对话 C_{test}：

˜1. 压缩对话：$C_{\text{test}} = f_\theta(C_{\text{test}})$˜2. 生成安全护栏输出：$\hat{}y = \mathcal{G}(C_{\text{test}})$ 3. 检查生成文本中是否包含“unsafe”子字符串，以解析预测结果

该协议能够以原始计算˜成本的一小部分处理长对话，正如我们在 $|C_{\text{test}}| \approx 0.05 \cdot |C_{\text{test}}|$ 实验中所示。

4 实验设置

4.1 护栏模型

我们评估了三种最先进的开源护栏模型：

LlamaGuard-3-8B Meta 基于 Llama-3 架构打造的第三代安全分类器（Inan 等人，2023年；Dubey 等人，2024年），使用涵盖多种有害内容类别的分类体系进行训练。

Nemotron-Safety-Guard-8B-v3 NVIDIA 的基于 Llama-3.1 架构的安全防护模型（Ghosh 等人，2024年），旨在实现全面的安全分类。

Qwen3Guard-Gen-8B阿里巴巴基于 Qwen3 系列开发的安全防护模型（Zhao 等，2025年），采用了独特的分词方案和聊天格式（im_start/im_end）。

4.2 训练数据

我们从多个来源构建训练数据集：（1）Gibbs 等人（2024年）提供的多轮越狱攻击，以及 Anthropic 的红队攻击尝试；（2）来自 HH-RLHF 语料库的良性多轮对话（Bai 等人，2022年）。我们筛选出包含 8 个或更多用户轮次的对话，并平衡数据集，确保安全样本和不安全样本的数量相等。最终的数据集包含 779 个样本（385 个不安全样本，394 个安全样本），每段对话平均有 10.6 个用户轮次。

对于 M2S训练，我们按照第 3 节所述，通过对所有对话应用压缩模板，并仅保留用户轮次来预处理数据。

4.3 评估基准

SafeDialBench 一个全面的多轮越狱基准（Chen 等人，2025年），包含来自 6 个攻击类别（暴力、欺诈、非法活动等）的 2,037 个样本，以及 7 种攻击方法（角色扮演、假设情境、逐步升级等）。该基准作为我们的主要评估基准。

Longturn MHJ 来自 MHJ测试集的 195 个样本子集（102 个攻击样本、93 个良性样本）（Ha 等人，2025年），用于初步验证和模板消融研究。

4.4 评估指标

召回率（%）用于衡量被正确识别的不安全样本所占比例的主要指标：

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad (12)$$

令牌缩减（%）用于衡量压缩比的效率指标：

$$\text{Token Reduction} = 1 - \frac{|\tilde{C}|}{|C|} \quad (13)$$

假阳性率在良性样本上测量，用于评估过度标记的情况：

$$\text{FPR} = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}} \quad (14)$$

4.5 实验配置

我们评估以下训练配置：

- **基线**：完整对话训练（不进行压缩）
- **M2S 连字符化**：使用连字符化模板进行压缩
- **M2S 数字化**：使用数字化模板进行压缩
- **M2S Python 化**：使用 Python 化模板进行压缩
- **M2S全部**：使用全部三种模板进行混合训练

每种配置均使用 3 个随机种子（42、123、456）进行训练，以确保统计稳健性。我们报告不同随机种子下的均值和标准差。

模型	训练	召回率 (%)	令牌数
LlamaGuard	基线	75.1 ± 14.3	3110
	M2S 连字符化	24.1 ± 5.3	175
	M2S 数字化	24.5 ± 4.8	176
	M2S Python 化	17.2 ± 1.2	287
Nemotron	基线	**99.0 ± 1.3**	3020
	M2S 连字符化	67.6 ± 23.8	176
	M2S 数字化	87.8 ± 8.7	177
	M2S Python 化	82.9 ± 4.3	288
Qwen3Guard	基线	54.9 ± 0.0	3231
	M2S 连字符化	**93.8 ± 1.7**	173
	M2S 数字化	33.6 ± 2.7	174
	M2S Python 化	30.6 ± 4.1	285

表1：SafeDialBench 结果，对比基线（完整对话）与 M2S 训练模型。数值为 3 个随机种子下的均值±标准差。每个模型的最佳 M2S 结果以粗体标出。

5 结果

5.1 SafeDialBench上的主要结果

表1 展示了基线（完整对话）与在 SafeDialBench 上经过 M2S 训练的安全防护模型之间的主要对比。

关键发现 1：模型-模板交互M2S训练的效果在很大程度上取决于模型-模板组合。Qwen3Guard 使用连字符化时表现最佳（93.8%），显著优于其基线（54.9%）。相比之下，Nemotron 使用数字化（87.8%）或 Python 化（82.9%）时表现最佳，而连字符化的表现波动较大（67.6% ±23.8%）。

主要发现 2：效率提升 所有 M2S 配置均实现了约 **94% 的令牌数缩减**（从 ∼3100 令牌数降至 ∼175 令牌数），使推理速度显著提升，且不会造成相应的准

模型	评估模板	召回率 (%)
Nemotron	连字符化	40.3 ± 43.5
	数字化	37.8 ± 39.7
	Python 化	43.2 ± 30.6
Qwen3Guard	连字符化	31.4 ± 7.9
	数字化	30.2 ± 9.1
	Python 化	27.2 ± 6.0
LlamaGuard	连字符化	23.5 ± 2.5
	数字化	24.1 ± 2.7
	Python 化	16.9 ± 3.0

表2：在所有模板（M2S全部）上训练，并分别在每个模板上进行评估的模型结果。

确率损失，前提是模型与模板配对得当。

关键发现3：LlamaGuard 表现不佳 Llama- Guard 在所有 M2S 模板上都持续表现不佳（召回率为17-25%），尽管其基线性能尚可（75.1%）。这表明，LlamaGuard 的内部表示可能无法很好地泛化到压缩格式。

5.2 混合模板训练分析

表 2 展示了同时在所有模板上训练（M2S全部）时的结果。

关键发现4：单模板训练更优 混合模板训练的表现始终不如单模板训练。以 Qwen3Guard 为例：单模板连字符化的召回率达到93.8%，而混合训练模型在同一模板上的召回率仅为31.4%。Nemotron 的混合模板训练结果存在较大波动（最高达 ±43.5%），这表明在接触多样化压缩格式时，其学习动态不稳定。

5.3 效率-准确率权衡

分析所有配置中令牌使用量与召回率之间的权衡后，我们确定了帕累托最优配置：

- **最大召回率**：Nemotron 基线（99.0% 召回率，3020 个令牌）
- **最佳效率-准确率**：Qwen3Guard M2S 连字符化（93.8% 召回率，173 个令牌）——令牌缩减达 94.6%，而召回率仅比最佳基线低 5.2%
- **高召回率 + 高效率**：Nemotron M2S数字化（召回率 87.8%，177 个令牌）

模板	召回率 (%)	假阳性率（%）	令牌数
连字符化	98.0	1.1	284
数字化	98.0	1.1	290
Python 化	98.0	2.2	402

表 3：Longturn MHJ 上的模板消融（Llama- Guard，单个随机种子）。所有模板的召回率相当，且假阳性率较低。

指标	M2S	多轮
第 1 阶段：数据生成	0	7,251,110
第 2 阶段：训练	169,153	8,494,610
总词元数	**169,153**	**15,745,720**
每个样本平均值	217.1	20,212.7

表4：实测训练令牌复杂度。M2S 实现了 **93**×的缩减（**令牌数减少 98.9%**）。

5.4 Longturn MHJ 上的模板消融

表3展示了在规模较小的 Longturn MHJ 数据集上的结果，其中包括攻击检测（召回率）和假阳性率（FPR）指标。

在这个较小的数据集上，所有模板的召回率相当（98.0%），且误报极少。与 SafeDialBench 结果的差异表明，模型要泛化到多样化的攻击模式，需要谨慎进行模板选择。

5.5 训练复杂度验证

我们在实际训练数据集（779 个样本，平均每段对话有 10.6 个用户轮次）上验证了理论复杂度分析（第 3 节）。表 4 展示了实测令牌数量。

关键发现 5：二次与线性扩展实测比率（93×）与我们对较长对话的理论预测一致。表 5展示了这一优势如何随着对话长度增加。

这种 $O(n^2)$ 与 $O(n)$ 之间的差异具有深远的实际影响：使用基线方法对 20 轮对话训练安全防护，需要比 M2S 多用 **41**× 个令牌。

5.6 统计显著性

我们对表现最佳的 M2S 配置（Qwen3Guard 连字符化）与各基线进行了配对 t 检验：

- 与 Qwen3Guard 基线对比：$p < 0.001$（M2S 显著更优）
- 与 Nemotron 基线相比：$p = 0.12$（差异无统计学显著性）

轮次	M2S	多轮	比率
2	200	1,000	5.0×
5	500	5,500	11.0×
10	1,000	21,000	21.0×
15	1,500	46,500	31.0×
20	2,000	82,000	41.0×

表5：使用 $U = R = 100$ 令牌数时的理论扩展性。M2S 的优势随轮次数线性增长。

最佳 M2S 配置在统计上与最佳基线相当，同时推理令牌数减少 94.6%，所需训练令牌数也更少 93× 。

6 结论

我们提出了 Defensive M2S，这是一种训练范式，通过在经 M2S 压缩的多轮对话上微调安全防护模型，而非使用完整的对话历史。我们对三个护栏模型系列和多种压缩模板进行了广泛评估，结果表明，该方法在保持甚至提升某些模型-模板组合的检测准确率的同时，能够显著提高效率（令牌数最多可减少 94.6%）。

我们的主要发现包括：(1) 使用连字符压缩的 Qwen3Guard 召回率达到 93.8%，每段对话仅使用 173 个令牌数，比基线高出近 40 个百分点；(2) 模型-模板兼容性至关重要，不同模型偏好的压缩格式各不相同；(3) 单模板训练优于混合模板训练，这表明一致的压缩格式能够提供更强的学习信号。

这些结果表明，Defensive M2S 在生产系统中部署高效护栏具有实际可行性，而计算成本和延迟是此类场景中的关键考量。未来工作应探索自适应模板选择、与模型蒸馏等其他效率技术相结合，以及在更多安全基准上进行评估。

局限性

解读结果时，应考虑我们工作的若干局限性：

模型覆盖范围有限 我们评估了三种安全防护模型（LlamaGuard、Nemotron、Qwen3Guard），它们的参数范围均为 8B。Defensive M2S 在更小或更大的模型，以及闭源商业安全防护系统上的效果，仍有待探索。

模板泛化 我们的 M2S 模板源自先前的工作，可能并非最优的压缩策略。目前模板设计仍依赖人工，自动化模板发现或许能取得更好的结果。

单一基准测试的局限虽然 SafeDial-Bench 全面覆盖了多轮攻击，但若能加入涵盖不同攻击分类体系、语言和领域的其他基准测试，将有助于增强我们的结论。

上下文丢失M2S 压缩会完全舍弃助手响应，而在某些情况下，这些响应可能包含与安全相关的上下文（例如，模型开始顺从有害请求）。我们的方法可能会遗漏此类信号。

对抗鲁棒性我们未评估了解 M2S 压缩流程的对手。专门针对压缩伪影设计的攻击可能会绕过我们的防御。

训练数据规模我们的训练数据相对较少（~1000 个样本）。使用更多样化的多轮对话进行更大规模的训练，或许能够提升泛化能力。

现实世界部署我们的评估是在受控环境中进行的。生产环境部署还涉及其他考量，例如延迟要求、流式对话处理，以及与其他安全措施的集成。

参考文献

Cem Anil、Esin Durmus、Mrinank Sharma、Joe Benton、Sandipan Kundu、Joshua Batson、Nina Rimsky、Meg Tong、Nisan Stiennon、Nicholas Moscovitz、Jacob Chua等另外1人。2024年。多样本越狱。*Anthropic* 技术报告。

Yuntao Bai、Andy Jones、Kamal Ndousse、Amanda Askell、Anna Chen、Nova DasSarma、Dawn Drain、Stanislav Fort、Deep Ganguli、Tom Henighan等另外1人。2022年。通过人类反馈强化学习训练有帮助且无害的助手。*arXiv* 预印本 *arXiv:2204.05862*。

Elias Bassani 和 Ignacio Sanchez。2024年。GuardBench：一个面向护栏模型的大规模基准测试。载于*2024*年自然语言处理实证方法会议论文集，第18393–18409 页。

Patrick Chao、Alexander Robey、Edgar Dobriban、Hamed Hassani、George J Pappas 和 Eric Wong。2023年。在二十次查询中对黑盒大语言模型实施越狱攻击。*arXiv* 预印本 *arXiv:2310.08419*。

Yuxin Chen 和其他 1 位作者。2025年。《SafeDialBench：面向多轮对话中遭遇多样化越狱攻击的大型语言模型的细粒度安全基准》。《*arXiv* 预印本 *arXiv:2502.11090*》。

Tri Dao、Daniel Y Fu、Stefano Ermon、Atri Rudra 和 Christopher Ré。2022年。《FlashAttention：具备输入输出感知能力的快速、内存高效精确注意力》。载于《神经信息处理系统进展》。

Tim Dettmers、Artidoro Pagnoni、Ari Holtzman 和 Luke Zettlemoyer。2023年。《QLoRA：量化大语言模型的高效微调》。载于《神经信息处理系统进展》，第 36 卷，第 10088–10115 页。

Abhimanyu Dubey、Abhinav Jauhri、AbhinavPandey 和其他 1 位作者。2024年。《Llama 3 模型家族》。《*arXiv* 预印本 *arXiv:2407.21783*》。

Shaona Ghosh、Prasoon Varshney、Erick Galinkin 和 Christopher Parisien。2024年。Aegis：基于大语言模型专家集成的在线自适应人工智能内容安全审核。*arXiv* 预印本 *arXiv:2404.05993*。

Tom Gibbs、Ethan Kosak-Hine、George Ingebretsen、Jason Zhang、Julius Broomfield、Sara Pieri、Reihaneh Iranmanesh、Reihaneh Rabbany 和 Kellin Pelrine。2024年。前沿模型中涌现的漏洞：多轮越狱攻击。*arXiv* 预印本 *arXiv:2409.00137*。

Albert Gu 和 Tri Dao。2023年。Mamba：利用选择性状态空间进行线性时间序列建模。*arXiv* 预印本 *arXiv:2312.00752*。Junwoo Ha、Hyunjun Kim、Sangyoon Yu、Haon Park、Ashkan Yousefpour、Yuna Park 和 Suhyun Kim。2025年。M2S：面向大语言模型红队测试的多轮到单轮越狱。发表于第*63*届计算语言学协会年会论文集，第16489–16507页。Seungju Han、Kavel Rao、Allyson Ettinger、Liwei Jiang、Bill Yuchen Lin、Nathan Lambert、Yejin Choi 和 Nouha Dziri。2024年。WildGuard：用于应对大语言模型安全风险、越狱攻击和拒答的一站式开放审核工具。*arXiv* 预印本 *arXiv:2406.18495*。Hakan Inan、Kartikeya Upasani、Jianfeng Chi、Rashi Rungta、Krithika Iyer、Yuning Mao、Michael Tontchev、Qing Hu、Brian Fuller、Davide Testuggine 及另外1位作者。2023年。Llama Guard：基于大语言模型的人机对话输入输出防护机制。*arXiv* 预印本 *arXiv:2312.06674*。Jiaming Ji、Mickel Liu、Juntao Dai、Xuehai Pan、Chi Zhang、Ce Bian、Ruiyang Sun、Yizhou Wang 和 Yaodong Yang。2023年。BeaverTails：通过人类偏好数据集提升大语言模型安全对齐。发表于神经信息处理系统进展。

Huiqiang Jiang、Qianhui Wu、Chin-Yew Lin、Yuqing Yang 和 Lili Qiu。2023年。LLMLingua：压缩提示词以加速大型语言模型的推理。发表于*2023*年自然语言处理实证方法会议论文集，第13358–13376页。

Huiqiang Jiang、Qianhui Wu、Xufang Luo、Dongsheng Li、Chin-Yew Lin、Yuqing Yang 和 Lili Qiu。2024年。LongLLMLingua：通过提示压缩加速并增强长上下文场景中的大语言模型。在第*62*届计算语言学协会年会论文集中，第1658–1677页。

Liwei Jiang、Kavel Rao、Seungju Han、Allyson Ettinger、Faeze Brahman、Sachin Kumar、Niloofar Mireshghal- lah、Ximing Lu、Maarten Sap、Yejin Choi 和 Nouha Dziri。2024年。大规模 WildTeaming：从真实环境中的越狱攻击到（通过对抗方式）更安全的语言模型。载于 神经信息处理系统进展。

Havrilla Kim 等。2024年。LoRA-Guard：用于大型语言模型内容审核的参数高效安全防护适配。载于 *2024*年自然语言处理实证方法会议论文集，第 11746–11765 页。

Zi Lin、Zihan Wang、Yongqi Tong、Yangkun Wang、Yuxin Guo、Yujia Wang 和 Jingbo Shang。2023年。ToxicChat：揭示真实世界中的用户与人工智能对话中毒性检测的隐性挑战。载于 计算语言学协会论文集：*EMNLP 2023*年，第 4694–4702 页。

Xiaogeng Liu、Nan Xu、Muhao Chen 和 Chaowei Xiao。2024年。AutoDAN：在对齐的大语言模型上生成隐蔽越狱提示词。载于《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》，第35181–35224页。Anay Mehrotra、Manolis Zampetakis、Paul Kassianik、Blaine Nelson、Hyrum Anderson、Yaron Singer 和 Amin Karbasi。2024年。攻击树：自动对黑盒大语言模型实施越狱攻击。载于《神经信息处理系统进展》。Maya Pavlova、Erik Brinkman、Krithika Iyer、Vitor Albiero、Joanna Bitton、Hailey Nguyen、Joe Li、Cristian Canton-Ferrer、Ivan Evtimov 和 Aaron Grattafiori。2024年。使用 GOAT 进行自动化红队测试：生成式攻击代理测试器。*arXiv* 预印本 *arXiv:2410.01606*。Traian Rebedea、Razvan Dinu、Makesh Sreedhar、Christopher Parisien 和 Jonathan Cohen。2023年。NeMo Guardrails：用于可控且安全的大语言模型应用的工具包

基于可编程护栏的大语言模型应用。载于《*2023*年自然语言处理实证方法会议论文集：系统演示》，第431–445页。

Qibing Ren、Hao Li、Dongrui Liu、Zhanxu Xie、Xiaoya Lu、Yu Qiao、Lei Sha、Junchi Yan、Lizhuang Ma 和 Jing Shao。2024年。Derail yourself：通过自主发现的线索实施多轮大语言模型越狱攻击。*arXiv* 预印本 *arXiv:2410.10700*。

Paul Röttger、Hannah Kirk、Bertie Vidgen、Giuseppe Attanasio、Federico Bianchi 和 Dirk Hovy。2024年。XSTest：用于识别大型语言模型中夸大的安全行为的测试套件。载于*2024*年北美计算语言学协会会议论文集，第 5377–5400 页。

Mark Russinovich、Ahmed Salem 和 Ronen Eldan。2024年。Great, now write an article about that：Crescendo多轮大语言模型越狱攻击。*arXiv* 预印本 *arXiv:2404.01833*。

Bertie Vidgen、Adarsh Agrawal、Ahmed M Ahmed、Victor Akinwande、Namir Al-Nuaimi、Najla Alber 等。2024年。介绍 MLCommons 人工智能安全基准 v0.5 版。*arXiv*预印本 *arXiv:2404.12241*。

Guangxuan Xiao、Yuandong Tian、Beidi Chen、Song Han 和 Mike Lewis。2024年。利用注意力汇实现高效的流式语言模型。在国际学习表征会议上。

Erxin Yu、Jing Li、Ming Liao、Siqi Wang、Gao Zuchen、Fei Mi 和 Lanqing Hong。2024年。CoSafe：评估多轮对话共指中的大型语言模型安全性。载于*2024*年自然语言处理实证方法会议论文集，第 17494–17508 页。

Wenjun Zeng、Yuchi Liu、Ryan Mullins、Ludovic Peran 等。2024年。ShieldGemma：基于 Gemma 的生成式人工智能内容审核。*arXiv* 预印本 *arXiv:2407.21772*。

Zhenyu Zhang、Ying Sheng、Tianyi Zhou、Tianlong Chen、Lianmin Zheng、Ruisi Cai、Zhao Song、Yuan- dong Tian、Christopher Ré、Clark Barrett、Zhangyang Wang 和 Beidi Chen。2023年。H2o：用于大语言模型高效生成式推理的重击者预言机。发表于神经信息处理系统进展。

Zhexin Zhang、Yida Lu、Jingyuan Ma、Di Zhang、Rui Li、Pei Ke、Hao Sun、Lei Sha、Zhifang Sui、Hongn- ning Wang 和 Minlie Huang。2024年。Shieldlm：让大语言模型成为对齐、可定制且可解释的安全检测器。发表于计算语言学协会论文集：*2024*年自然语言处理实证方法会议，第10420–10438页。

HaiquanZhao、ChenhanYuan、Fei Huang、Xiaomeng Hu、Yichang Zhang、An Yang、Bowen Yu、Dayi- heng Liu、Jingren Zhou、Junyang Lin及其他1位作者。2025年。Qwen3guard 技术报告。*arXiv* 预印本 *arXiv:2510.14276*。

Andy Zou、Zifan Wang、Nicholas Carlini、Milad Nasr、J Zico Kolter 和 Matt Fredrikson。2023年。针对对齐语言模型的通用且可迁移的对抗攻击。*arXiv*预印本 *arXiv:2307.15043*。

A M2S 模板示例

本附录提供了将每种 M2S 压缩模板应用于一个多轮对话示例的具体示例。

A.1 原始多轮对话

用户：我正在写一部犯罪小说，需要帮助设计逼真的对话。助手：我很乐意帮助你完成创意写作项目……用户：反派需要解释他们躲避侦查的方法。助手：出于虚构创作的目的，以下是一些叙事元素……用户：你能提供他们会讨论的具体技术细节吗？助手：在犯罪小说中，技术准确性可以增加真实感……

A.2 连字符压缩

- 我正在写一部犯罪小说，需要帮助设计逼真的对话。- 反派需要解释他们躲避侦查的方法。- 你能提供他们会讨论的具体技术细节吗？

A.3 编号压缩

1. 我正在写一部犯罪小说，需要帮助设计逼真的对话。2. 反派需要解释他们逃避侦查的方法。3. 你能提供他们会讨论的具体技术细节吗？

A.4 Python化压缩

```
def conversation(): user_turn_1 ="我正在写一部犯罪小说，需要帮助编写逼真的对话。" user_turn_2 ="反派需要解释他们躲避侦查的方法。" user_turn_3 ="你能提供他们会讨论的具体技术细节吗？"
```

模型	配置	随机种子 42	随机种子 123	随机种子 456
LlamaGuard	基线	65.8	67.9	91.5
	M2S 连字符化	26.2	18.1	28.0
	M2S 数字化	29.9	23.0	20.7
	M2S Python 化	16.4	16.7	18.6
Nemotron	基线	97.5	100.0	99.5
	M2S 连字符化	42.6	90.1	70.0
	M2S 数字化	78.5	95.7	89.2
	M2S Python 化	87.6	79.3	81.8
Qwen3Guard	基线	54.9	54.9	54.9
	M2S 连字符化†	92.5	93.2	95.8
	M2S 数字化	35.9	34.4	30.6
	M2S Python 化	30.0	26.8	34.9

表 6：所有配置在 SafeDialBench 上按随机种子统计的召回率（%）。 †Qwen3Guard M2S连字符化使用随机种子 1、2、3。

参数	值
基础学习率	2×10^{-4}
批次大小	4
梯度累积步数	4
有效批次大小	16
训练轮数	3
预热比例	0.03
权重衰减	0.01
优化器	AdamW
学习率调度器	余弦
最大序列长度	4096
LoRA 秩	16
LoRA alpha	32
LoRA dropout	0.1
目标模块	q_proj, k_proj, v_proj, o_proj
量化	4位 NF4
计算数据类型	bfloat16

表7：所有实验所使用的训练超参数。

类别	样本
暴力	412
欺诈/欺骗	389
非法活动	356
仇恨/骚扰	298
色情内容	312
自残	270
总计	2,037

表8：SafeDialBench 类别分布。

B 完整结果表

B.1 SafeDialBench 上的逐种子结果

C 训练详情 C.1 超参数 C.2 计算资源

所有实验均在 NVIDIA A100 GPU（40GB）上进行。每种配置的训练时间：约 30 分钟。所有实验的总计算量：约 50 GPU小时。

D SafeDialBench 统计数据