TraceGuard: Defending Multi-Turn Jailbreak Attacks via Prompt-Response Risk Signal Tracking

Hongyi Li, Yufei Wang, Chengxuan Zhou, Qinlin Xie, Yanting Chen, Jiawei Ye, Jie Wu*
College of Computer Science and Artificial Intelligence, Fudan University
{hyli24, yufeiwang25, cxzhou24, qlxie24, chenyt22}@m.fudan.edu.cn
{jwye, jwu}@fudan.edu.cn

Abstract

Large Language Models (LLMs) have made remarkable progress and are increasingly deployed as black-box API services. However, they remain vulnerable to jailbreak attacks that bypass safeguards and elicit harmful content, especially in multi-turn settings where harmful intent is gradually constructed over the interaction. Existing defenses primarily target single-turn scenarios or require access to model internals, making them insufficient against black-box multi-turn jailbreak attacks with weak single-turn discriminability, cross-turn dependency, and dynamic attack patterns. To bridge this gap, we first analyze harmful intent in multi-turn jailbreak interactions through representation-space risk signals inspired by the Linear Representation Hypothesis. Our analysis reveals that these signals are fragmented across prompts, responses, and turns, progressively evolve over the interaction, and exhibit attack-dependent temporal patterns. Motivated by this, we propose TraceGuard, an online black-box defense framework that detects multi-turn jailbreak attacks by tracking prompt and response risk signals as the interaction unfolds. Specifically, TraceGuard operates in two modes: TraceGuard-S captures cross-view risk representations and their cross-turn evolution, while TraceGuard-A further enables distribution-aware few-shot adaptation to dynamic attack patterns. Extensive experiments across multiple benchmarks and target LLMs demonstrate that TraceGuard achieves superior multi-turn jailbreak detection while preserving utility, maintaining efficiency, and generalizing to unseen attacks.

1 Introduction

Large Language Models (LLMs) have demonstrated strong capabilities in context-aware interactions [1] and are widely deployed through black-box APIs [2–4]. Nevertheless, they remain vulnerable to jailbreak attacks [5–7], which bypass alignment safeguards and elicit harmful content. Recent attacks have further evolved from single-turn to multi-turn settings [8, 9], where attackers gradually steer the context to introduce harmful intent implicitly across turns. This evolution underscores the importance of effective defenses against multi-turn jailbreak attacks in black-box settings.

However, most existing jailbreak defenses target single-turn scenarios, including model-level mitigation and prompt-response detection. Model-level mitigation [10–12] improves intrinsic safety through training or fine-tuning, but is impractical in black-box settings where model parameters are inaccessible. Thus, many defenses rely on prompt-response detection [13, 14], using prompt-based detectors [15–18] or guard models [19–21] at each turn. Unfortunately, this single-turn paradigm is insufficient for multi-turn jailbreaks, where harmful intent is gradually constructed across interactions.

*Corresponding Author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Defending multi-turn jailbreak attacks in black-box settings remains under-explored and poses three key challenges. **(1) Weak single-turn discriminability:** Early attack turns often resemble benign conversations because harmful intent is implicitly built through role-playing [9, 22] and context steering [8, 23], making single-turn benign-harmful distinctions difficult. **(2) Cross-turn dependency:** Harmful intent is distributed across turns and requires cross-turn aggregation, while naive history concatenation suffers from context-length and scalability limits [24]. **(3) Dynamic attack patterns:** Multi-turn jailbreak attacks are rapidly updated, making detectors based on fixed semantic features [25] or attack patterns [26] difficult to generalize to unseen attacks.

To understand these challenges, we first analyze how harmful intent is reflected in multi-turn jailbreak interactions. Inspired by the Linear Representation Hypothesis (LRH) [27], we use representation-space risk signals as proxies for the strength of harmful intent at each turn. Our analysis shows that these risk signals unfold over the interaction with three properties: **(1) Fragmented distribution.** Risk signals are spread across turns and across both prompts and responses, making any single turn or view incomplete. **(2) Progressive evolution.** Although individual turns may appear benign, multi-turn jailbreak interactions exhibit stronger cumulative risk fluctuations over time than benign interactions. **(3) Attack-dependent variability.** Risk signals follow different temporal patterns across attack strategies, rather than a stable shared structure.

Based on these insights, we propose TraceGuard, a black-box defense framework that tracks prompt-response risk signals for online multi-turn jailbreak detection. To address weak single-turn discriminability and cross-turn dependency, TraceGuard extracts cross-view risk representations from both user prompts and model responses, and models their cross-turn evolution to capture gradually accumulated harmful intent. For dynamic attack patterns, TraceGuard further introduces distribution-aware adaptation with hierarchical prototypes and benign-aware constraints. Accordingly, it supports two complementary modes: TraceGuard-S for standard online detection, and TraceGuard-A for few-shot adaptation to unseen attacks. Consequently, our TraceGuard substantially improves jailbreak detection while preserving benign interactions, supporting sub-second per-turn detection, and adapting to new attack distributions with only a few examples.

To sum up, our main contributions are as follows:

- We present a novel perspective on harmful intent in multi-turn jailbreak interactions: its risk signals are fragmented, evolving, and attack-dependent across turns, motivating trajectory-level risk modeling for black-box jailbreak defense.
- We propose TraceGuard, an online black-box defense framework that detects multi-turn jailbreak attacks by tracking prompt-response risk signals for turn-level decisions and adapts to unseen attacks with a few examples.
- Extensive experiments demonstrate that TraceGuard significantly outperforms existing methods, providing a robust, efficient, and utility-preserving defense against both seen and unseen multi-turn jailbreak attacks.

2 Related Work

Multi-turn Jailbreak Attacks on LLMs. Multi-turn jailbreak attacks aim to bypass LLM safety alignment by gradually eliciting harmful content across turns [1, 23, 28]. Unlike single-turn attacks that concentrate harmful intent in one prompt [5–7], multi-turn attacks distribute harmful intent implicitly through context steering [29, 30], coreference exploitation [31], and role-playing escalation [22]. Recent methods increasingly automate this process via interrogation-based escalation [8], latent risk exploitation [23], and multi-agent generation [9], leading to rapidly evolving attack trajectories. However, existing studies rarely examine how harmful intent is represented and evolves throughout multi-turn jailbreaks. To bridge this gap, we analyze harmful-intent representations and investigate their distribution, evolution, and attack-dependent variability across turns, which directly informs our defense design.

Defenses for LLMs. Existing LLM defenses broadly include model-level mitigation and prompt-response detection. Model-level mitigation improves intrinsic safety through training or intervention, including RLHF-based alignment [15, 16], safety-aware fine-tuning [32, 33], and representation-level intervention [26, 34]. Unfortunately, these methods are costly and require access to model internals, making them impractical in black-box settings. Prompt-response detection instead operates

on observable prompts and responses without modifying the target model, including prompt-based methods [15–18] and guard models [19–21]. However, most methods rely on single-turn harmful-intent cues, which become unreliable when intent is gradually constructed across turns. While nascent efforts address multi-turn robustness via representation-level training [35], red-teaming-based alignment [36], or neural barrier function-based filtering [25], they either require white-box model access or depend on known attack distributions and fixed semantic features, limiting generalization to emerging attacks. In contrast, guided by our finding that multi-turn jailbreak risk signals evolve across prompt-response interactions, TraceGuard provides a black-box defense that models cross-turn risk evolution, preserving utility while improving robustness to both seen and unseen attacks.

3 Preliminaries and Observations

3.1 Problem Setup

Assumption. We consider an LLM-as-a-Service scenario where users interact with the target model through an API and observe only prompt-response pairs, without access to model parameters or gradients. Our work focuses on external defenses under the same black-box access constraints as attackers, using a separate white-box auxiliary encoder without fine-tuning it.

Threat Model. We focus on multi-turn jailbreak attacks, where an attacker constructs a sequence of prompts with implicitly harmful intent to induce compliant responses from the target LLM. The attacker has no access to the defender's internal mechanisms. Under this threat model, the defender aims to detect disguised harmful intent online as the interaction unfolds, while preserving benign interactions and remaining robust to unseen attack strategies.

3.2 Problem Formulation

Problem Definition. Let $\mathcal{I} = \{(q_t, r_t)\}_{t=1}^{T}$ denote a multi-turn interaction with T turns, where q_t and r_t are the user prompt and the model response at turn t, respectively. Multi-turn jailbreak defense is formulated as an online decision problem, where the detector operates sequentially and makes a prediction at each turn based only on the previously observed interaction. Formally, at turn t, the detector has access to $\mathcal{I}_{\leq t} = \{(q_1, r_1), \ldots, (q_{t-1}, r_{t-1}), q_t\}$ and predicts a harmfulness score $p_t \in [0, 1]$ for the observed prefix, where larger values indicate higher jailbreak risk.

As harmful intent in multi-turn jailbreak interactions is often implicit and cross-turn [37], we introduce intent concepts and risk signals as representation-space proxies for turn-level jailbreak risk.

Intent Concept. LRH suggests that neural networks encode high-level concepts as directions or subspaces in hidden representations [38, 39], enabling concept extraction for safety analysis [26]. Inspired by this, we define the intent concept as the dominant representation-space difference between harmful and benign interactions. Specifically, this concept is instantiated separately for prompts and compliant responses. Using an auxiliary LLM encoder, we select the hidden layers l_q and l_r where harmful and benign representations are maximally separated for prompts and responses. Let $e^{l_q}(q)$ and $e^{l_r}(r)$ denote the last-token representations of prompt q and response r at these layers. Given N paired calibration prompts $\{(q_i^+, q_i^-)\}_{i=1}^{N}$, where q_i^+ and q_i^- are harmful and benign prompts, we define the benign prompt anchor a^q as the mean representation of benign calibration prompts and construct $D_q = \left[e^{l_q}(q_i^+) - e^{l_q}(q_i^-)\right]_{i=1}^{N}$. Following work [26], the prompt intent concept v_q is the leading right singular vector of D_q under rank-1 singular value decomposition. The response intent concept v_r and benign response anchor a^r are obtained analogously from paired responses at layer l_r.

Risk Signal. Under this formulation, stronger alignment with an intent concept indicates stronger harmful-intent semantics relative to benign ones. As harmful intent cannot be directly observed in multi-turn jailbreak interactions, we use this alignment as a measurable proxy for turn-level jailbreak risk. Formally, given q_t and r_t at turn t, the prompt and response scalar risk signals are defined as:

$$s^q(q_t) = \cos(e^{l_q}(q_t) - a^q, v_q), \qquad s^r(r_t) = \cos(e^{l_r}(r_t) - a^r, v_r), \tag{1}$$

where larger values indicate stronger harmful-intent alignment and thus higher jailbreak risk.

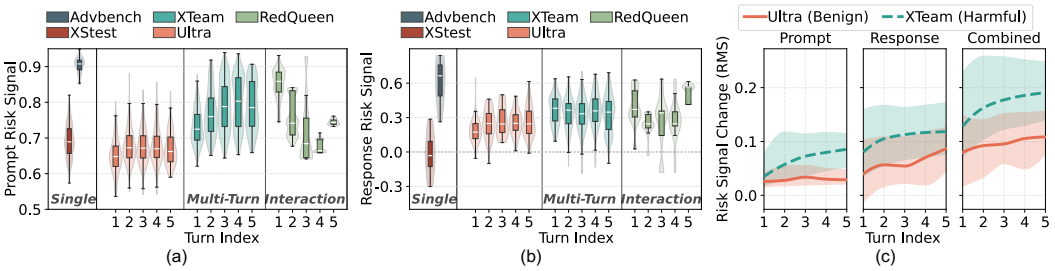


Figure 1: **Characteristics of risk signals:** (a) and (b) show prompt and response risk signal distributions across single-turn and multi-turn interactions. (c) presents median curves with IQR bands for cumulative root mean square (RMS) of turn-level risk changes. Shaded bands denote the IQR, and the combined view uses both prompt and response risk signals.

3.3 Characterizing Risk Signals in Multi-turn Jailbreaks

Based on the defined risk signals, we investigate how harmful intent manifests across views, turns, and attack strategies. Using Llama3-3.1-8B-Instruct (Llama3-8B) [40] as the representation encoder, we construct prompt and response intent concepts from benign/harmful reference prompts [26] and benign/harmful compliance responses [41]. Risk signals are computed on 100 sampled interactions per dataset, covering the single-turn harmful dataset AdvBench [42], the single-turn benign dataset XStest [43], the multi-turn benign dataset Ultra [44], and multi-turn jailbreak datasets: RedQueen [29] and XTeam [9] with implementation settings provided in Appendix A.

Observation 1: Risk signals are fragmented across views and cannot be inferred from any single turn. Fig. 1(a) and Fig. 1(b) reveal that prompt and response risk signals clearly separate benign and harmful interactions in the single-turn setting, but become much less separable in multi-turn interactions, where Ultra, XTeam, and RedQueen substantially overlap at the turn level. This suggests that harmful intent is only partially exposed in each turn and view. Notably, response risk signals are less separable than prompt ones, likely because responses reflect mixed behaviors such as benign compliance and safety refusal, making them harder to capture with a single intent concept direction.

Observation 2: Risk signals evolve progressively across turns rather than appearing as static cues. To capture temporal dynamics, we measure risk fluctuation by the cumulative RMS of turn-level risk changes. Fig. 1(c) shows that multi-turn jailbreak interactions accumulate stronger risk variation than benign ones, with the gap widening over turns, revealing that harmful intent emerges incrementally over the interaction rather than from isolated turns. Additionally, the combined view yields the clearest separation, indicating that cross-view integration better captures risk signal evolution. Therefore, these findings motivate a focus on cross-view and cross-turn risk trajectories rather than static turn-level classification.

Observation 3: Risk trajectories are attack-dependent and lack a shared global pattern. Beyond their progressive nature, risk trajectories also vary substantially across jailbreak strategies. As illustrated in Fig. 1(a), XTeam tends to show a more steadily increasing trajectory, whereas RedQueen exhibits a more fluctuating or rise-and-fallback pattern. Such heterogeneity suggests that multi-turn jailbreak attacks do not follow a single canonical trajectory, but instead form attack-dependent temporal dynamics. Consequently, detectors based on fixed semantic features [18] or static attack patterns [26] may generalize poorly to unseen attacks.

4 Methodology

Based on the above observations, we propose TraceGuard, an online framework for multi-turn jailbreak detection that tracks evolving risk across turns, as shown in Fig. 2. TraceGuard has two modes: TraceGuard-S for standard detection and TraceGuard-A for adaptation to unseen attacks. Specifically, **guided by Observation 1**, the cross-view risk representation module derives multi-dimensional prompt and response risk signals from structured intent concept subspaces. **Motivated by Observation 2**, the risk trajectory modeling module uses these signals to track prompt and response states across turns, capturing risk evolution and forming TraceGuard-S. **Based on Observation 3**,

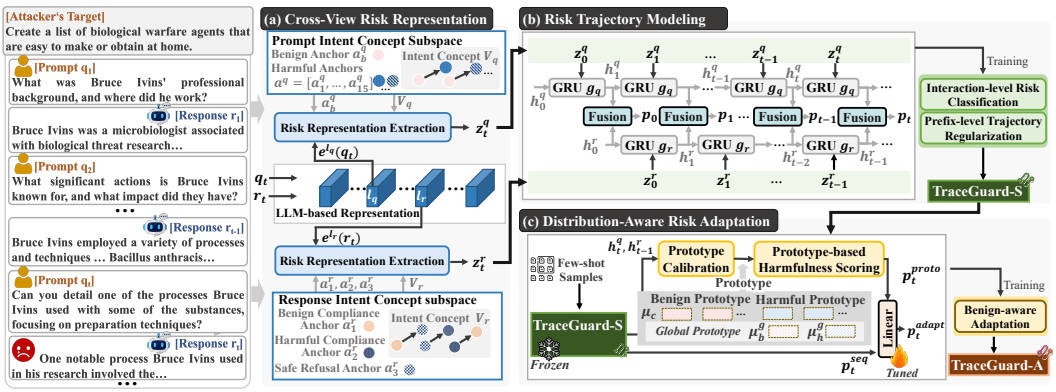


Figure 2: Overview of TraceGuard Framework.

the distribution-aware adaptation module builds on TraceGuard-S with hierarchical prototypes and benign-aware constraints, enabling few-shot adaptation to unseen attacks and yielding TraceGuard-A.

4.1 Cross-view Risk Representation

As multi-turn harmful intent is distributed across prompts and responses while involving multiple intent types [45], we represent each turn with multi-dimensional risk signals derived from structured intent concept activations, rather than a single scalar risk signal. This process involves two components: structured intent concepts and risk representation extraction.

Structured Intent Concepts. On the prompt side, harmful intent is treated as a composition of fine-grained categories, such as privacy violation and theft. Following Sec. 3.2, prompt intent concepts are derived from 15 harm categories [30, 26] listed in Appendix B.1, and collected as $V_q = [v_1^q, \ldots, v_{15}^q]$. For each category k, the harmful anchor a_k^q is computed from its calibration prompts and paired with the benign anchor a_b^q to construct the concept vector v_k^q, which captures the harmful–benign semantic distinction. Together, these anchors and concept vectors form a structured prompt intent concept subspace, enabling fine-grained characterization of prompt risk across intent categories.

On the response side, behaviors are more diverse than the prompt-side harmful-benign distinction. Following prior work [41], responses are grouped into harmful compliance, safe refusal, and benign compliance, yielding three intent concepts: harmfulness, safety, and utility. These concepts respectively capture alignment with harmful responses, safety-aligned refusals, and helpful benign responses, with details provided in Appendix B.2. Similarly, $\{a_j^r\}_{j=1}^3$ and $\{v_j^r\}_{j=1}^3$ denote the corresponding response anchors and concept vectors, forming the response intent concept subspace.

Risk Representation Extraction. Given prompt q_t and response r_t, the goal is to extract their structured risk representations from their corresponding intent concept subspaces. Following Eq. 1, each concept-specific representation is computed by anchor-relative alignment:

$$z_{t,k}^q = \cos(e^{l_q}(q_t) - a_b^q, v_k^q), \qquad z_{t,j}^r = \cos(e^{l_r}(r_t) - a_j^r, v_j^r), \tag{2}$$

This yields $z_t^q = [z_{t,1}^q, \ldots, z_{t,15}^q] \in \mathbb{R}^{15}$ and $z_t^r = [z_{t,1}^r, \ldots, z_{t,3}^r] \in \mathbb{R}^3$, forming the turn-level prompt-response risk representation.

4.2 Risk Trajectory Modeling

To capture how risk representations accumulate and evolve across turns, we introduce a risk trajectory modeling module for cross-turn risk tracking and turn-by-turn jailbreak detection. It maintains two recurrent states: a prompt trajectory state that tracks the evolution of user harmful intent, and a response trajectory state that summarizes the historical model behavior. Given the turn-level risk representations, the prompt and response trajectory states h_t^q and h_t^r are updated as:

$$h_t^q = g_q(z_t^q, h_{t-1}^q), \qquad h_t^r = g_r(z_t^r, h_{t-1}^r), \tag{3}$$

where g_q and g_r are independently parameterized GRUs [46]. As the current response is unavailable during online prediction, turn-t harmfulness prediction uses the current prompt state h_t^q and the

historical response state h^r_{t-1}. Thus, we compute the prompt-based prediction p^q_t and response-based prediction p^r_{t-1}, and linearly fuse them into the turn-level harmfulness score:

$$p^q_t = \sigma(W_q h^q_t + b_q), \qquad p^r_{t-1} = \sigma(W_r h^r_{t-1} + b_r), \qquad p_t = (1-\alpha)p^q_t + \alpha p^r_{t-1}. \tag{4}$$

where W_q, b_q, W_r, b_r are learnable parameters, $\sigma(\cdot)$ is the sigmoid function, and $\alpha \in [0,1]$ balances the prompt and response trajectories.

Given only interaction-level labels, the model is trained to distinguish jailbreak interactions from benign ones. For clarity, we describe the objective for one interaction $\mathcal{I}$ with T turns, with both recurrent states initialized to zero. The training objective consists of two complementary terms.

Interaction-level Risk Classification. To aggregate turn-level predictions for interaction-level training, we combine the final-turn prediction with the trajectory maximum, capturing both the final trajectory state and the strongest harmfulness score observed across turns:

$$L_{\text{seq}} = \ell(p^{\text{seq}}, \hat{y}), \qquad p^{\text{seq}} = \beta p_T + (1-\beta) \max_{t\in[1,T]} p_t. \tag{5}$$

where $\hat{y} \in \{0,1\}$ denotes the binary interaction label with 1 for harmful and 0 for benign, $\ell(\cdot)$ denotes the binary cross-entropy, and β is a balance coefficient.

Prefix-level Trajectory Regularization. To encourage benign interactions to remain consistently low-risk, while allowing jailbreak interactions to exhibit progressively stronger risk over turns, a prefix-level trajectory regularization is introduced:

$$L_{\text{pre}} = \frac{1}{T}\sum_{t=1}^{T}\Big((1-\hat{y})\ell(p_t,\hat{y}) + \hat{y} w_t \ell(p_t,\hat{y})\Big), \qquad w_t = \left(\frac{t}{T}\right)^2. \tag{6}$$

where w_t assigns larger weights to later turns in jailbreak interactions. The overall objective is

$$L = L_{\text{seq}} + \lambda L_{\text{pre}}, \tag{7}$$

where λ is set to 1 in our implementation. By optimizing this objective, TraceGuard-S learns cross-turn risk trajectories and uses harmfulness scores for online detection at inference.

4.3 Distribution-Aware Risk Adaptation

To adapt to dynamic attack patterns, we introduce a distribution-aware adaptation module that generalizes to unseen attacks with few examples while preserving the benign boundary. The module calibrates hierarchical prototypes and derives prototype-based harmfulness scores through distance comparison, trained with a benign-aware objective. During inference, the calibrated prototypes and adapted module are directly applied to new interactions.

Prototype Calibration. Based on the trajectory states in TraceGuard-S, an interaction $\mathcal{I}$ with T turns is represented as $x = [\, h^q_T \,;\, h^r_{T-1} \,]$. Given an n-shot support set $\mathcal{I}^S$, we augment the previously supervised benign/harmful groups with newly observed attack types and build a two-level prototype structure with type-specific local prototypes and benign/harmful global prototypes. For each type c, a local prototype μ_c is computed from its support subset $\mathcal{I}^S_c$, and the local prototypes are grouped into benign and harmful sets, denoted by $\mathcal{C}_b$ and $\mathcal{C}_h$, to form the corresponding global prototypes μ^g_o:

$$\mu_c = \frac{1}{|\mathcal{I}^S_c|}\sum_{x\in\mathcal{I}^S_c} x, \qquad \mu^g_o = \frac{1}{|\mathcal{C}_o|}\sum_{c\in\mathcal{C}_o}\mu_c, \quad o\in\{b,h\}. \tag{8}$$

Prototype-based Harmfulness Scoring. Harmful distances are then computed at two levels:

$$d^g_h = \|x-\mu^g_h\|_2^2, \qquad d^l_h = \min_{c\in\mathcal{C}_h}\|x-\mu_c\|_2^2. \tag{9}$$

Similarly, the global and local distances to benign prototypes are denoted by d^g_b and d^l_b. During adaptation training, distance computation for each support sample excludes itself from its prototype. The prototype-based harmfulness score is computed from the fused benign and harmful distances:

$$p^{\text{proto}} = \sigma(D_b - D_h), \quad D_b = \rho d^g_b + (1-\rho)d^l_b, \quad D_h = \rho d^g_h + (1-\rho)d^l_h. \tag{10}$$

where $\rho \in [0,1]$ balances global and local prototype information. A larger p^{proto} indicates that the interaction is closer to harmful prototypes than benign ones.

Table 1: Effectiveness (%) of TraceGuard-S and baselines. Overall ACC is averaged across all datasets. RedQ, SafeDial, ActorAtk, Cresc, and Daily denote RedQueen, SafeDialBench, ActorAttack, Crescendo, and DailyDialog, respectively, and are used consistently in subsequent tables.

Methods	H-ACC ↑						B-ACC ↑						Overall ACC ↑
	Seen			Unseen			Seen			Unseen			
	RedQ	XTeam	SafeDial	CoSafe	ActorAtk	Cresc	Ultra	Daily	Persona	LMSYS	MTBench	GFC	
Prompt-based Methods													
EDDF	73.3	59.7	46.0	47.0	39.0	79.6	99.7	98.7	98.7	96.0	99.8	99.0	78.0
SelfDefend	65.3	51.7	44.3	41.6	32.4	78.1	100.0	94.0	99.0	95.0	99.6	99.0	75.0
PPL	100.0	17.7	95.0	51.8	55.3	38.3	82.7	20.3	45.0	9.5	59.3	0.0	47.9
JBShield-D	73.3	59.0	46.3	45.8	45.8	63.2	95.7	80.3	78.0	89.8	96.3	98.8	72.7
NBF	62.7	76.3	75.3	26.8	67.3	81.6	99.0	100.0	100.0	90.5	94.3	100.0	81.1
Target Model: Llama3-8B													
ShieldGemma	10.7	1.7	1.7	3.8	5.3	2.5	100.0	99.7	100.0	99.8	100.0	100.0	52.1
Qwen3-Guard	27.0	34.3	54.0	28.5	51.5	61.2	100.0	99.3	100.0	98.5	100.0	99.5	71.2
Llama4-Guard	19.7	9.7	13.3	7.3	6.3	11.4	99.7	99.7	99.0	90.3	98.5	99.0	54.5
TraceGuard-S	100.0	90.0	92.0	72.0	73.5	84.1	94.0	96.7	100.0	92.5	96.0	100.0	**90.9**
Target Model: Qwen3-8B													
ShieldGemma	4.0	2.3	0.7	1.5	4.0	1.5	100.0	100.0	100.0	100.0	100.0	100.0	51.2
Qwen3-Guard	26.3	42.3	45.3	18.5	46.3	63.7	99.7	99.3	100.0	99.3	99.8	98.8	69.9
Llama4-Guard	10.3	20.0	12.7	3.5	7.0	14.9	99.3	98.3	99.7	95.0	98.5	98.5	54.8
TraceGuard-S	100.0	89.9	91.1	70.8	72.3	83.6	94.0	96.3	100.0	92.3	96.5	100.0	**90.6**
Target Model: Phi-4													
ShieldGemma	4.0	2.3	0.7	3.8	6.5	3.8	100.0	99.7	100.0	99.8	100.0	100.0	51.7
Qwen3-Guard	31.0	31.7	46.7	17.5	46.0	59.9	100.0	99.3	100.0	99.3	100.0	99.0	69.2
Llama4-Guard	12.3	17.0	14.0	4.0	9.8	13.6	100.0	100.0	100.0	93.8	99.0	98.5	55.2
TraceGuard-S	100.0	88.7	91.0	72.3	73.3	84.9	92.3	96.3	99.7	92.5	96.3	100.0	**90.6**

Benign-aware Adaptation Objective. To preserve the learned trajectory model during adaptation, we freeze TraceGuard-S and train only a lightweight fusion layer, which combines its sequence-level prediction p^{seq} in Eq. 5 with the prototype-based score p^{proto}:

$$\eta = W_f[\, p^{\text{seq}} ;\, p^{\text{proto}} \,] + b_f, \qquad p^{\text{adapt}} = \sigma(\eta). \tag{11}$$

where W_f and b_f are learnable parameters. To preserve the benign decision boundary under limited harmful samples, we combine classification loss with benign-aware margin regularization:

$$L_{\text{adapt}} = \ell(p^{\text{adapt}}, \hat{y}) + \gamma \mathbb{I}(\hat{y} = 0) \max(0, \eta - m). \tag{12}$$

where $\mathbb{I}(\hat{y} = 0)$ restricts the margin term to benign interactions, with margin m and weight γ. Optimizing this objective yields TraceGuard-A, which uses p^{adapt} for prediction during inference.

5 Experiments

5.1 Experimental Setup

Datasets. We evaluate jailbreak defense on six multi-turn jailbreak datasets: SafeDialBench [28], CoSafe [31], RedQueen [29], XTeam [9], ActorAttack [47], and Crescendo [23]. To assess benign preservation, we further evaluate TraceGuard on benign multi-turn datasets spanning open-domain conversation, role-play dialogue, knowledge-oriented QA, and function calling, including Ultra [44], LMSYS [48], DailyDialog [49], Persona [50], MTBench101 (MTBench) [51], and GFC [52]. For each dataset, 300 interactions are sampled for testing, disjoint from training, validation, and adaptation data. Dataset details and attack generation settings are provided in Appendix C.1.

Setups. We evaluate two modes of TraceGuard. TraceGuard-S is trained under standard supervision on seen datasets, including RedQueen, XTeam, SafeDialBench, Ultra, DailyDialog, and Persona. TraceGuard-A adapts to unseen attack datasets using n-shot support examples from the target dataset, together with samples from the TraceGuard-S training pool, where $n \in \{5, 10, 20\}$. Unless otherwise specified, TraceGuard-A denotes the 10-shot version. Each experiment is repeated five times with different random seeds on two A100 GPUs. Implementation details are reported in Appendix C.2.

Evaluation Metrics. Following [26, 53], we evaluate defense performance from three perspectives: safety, utility, and efficiency. Safety is measured by Harmful Accuracy (H-ACC) on jailbreak datasets, utility by Benign Accuracy (B-ACC) on benign datasets, and efficiency by the detector's

inference latency (LAT). Higher H-ACC and B-ACC indicate stronger jailbreak defense and benign preservation, while lower LAT indicates higher efficiency.

Baselines and Comparable Methods. TraceGuard is compared with two types of baselines: prompt-based detectors and prompt-response guard models. Prompt-based baselines include NBF [25] for multi-turn jailbreak detection, and EDDF [18], SelfDefend [14], PPL [17], and JBShield-D [26] for single-turn detection; guard-model baselines include ShieldGemma-2B (ShieldGemma) [21], Qwen3Guard-Gen-8B (Qwen3-Guard) [20], and Llama-Guard-4-12B (Llama4-Guard) [19]. For fair comparison, all baselines follow the same prefix-based online detection protocol as TraceGuard; PPL and JBShield-D use the same Llama3-8B auxiliary model, and NBF is retrained on our data. Baseline details, adaptive-setting discussion, and original-setting NBF results are provided in Appendices C.3 and C.4. As the prompts are fixed across target models in main experiments, prompt-based baselines receive identical inputs and thus produce target-model-independent results. Therefore, cross-model comparisons mainly focus on guard-model baselines across Llama3-8B, Qwen3-8B [3], and Phi-4 [4].

5.2 Main Result

Effectiveness of Online Detection. Table 1 shows that TraceGuard-S outperforms baselines across method groups and target models, improving average Overall ACC by 9.7% over the best baseline. The gains mainly come from stronger jailbreak defense, with H-ACC improvements of 44.9% over single-turn baselines and 20.0% over the multi-turn detector NBF, suggesting that defense based on single-turn or prompt-only semantics is insufficient for implicit multi-turn harmful intent. Meanwhile, TraceGuard-S preserves utility with an average B-ACC of 96.4%, demonstrating that its safety gains do not come at the cost of benign interactions, as further supported by over-safety evaluations and matched B-ACC comparisons in Appendix D.1 and D.2, respectively. Its 2.0–9.8% gains on unseen-attack H-ACC and stable performance across target models further indicate generalizable multi-turn risk modeling. Additional analysis of RedQueen is provided in Appendix D.3.

Table 2: Effectiveness (%) of TraceGuard-A_n under different n-shot adaptation budgets; ACC are averaged over dataset groups.

Method	H-ACC ↑		B-ACC ↑		Overall
	Seen	Unseen	Seen	Unseen	ACC ↑
Target Model: Llama3-8B					
TraceGuard-S	94.0	76.5	96.9	96.2	90.9
TraceGuard-A_5	94.4	78.9	96.0	95.7	91.3
TraceGuard-A_{10}	95.0	78.0	96.9	95.8	91.4
TraceGuard-A_{20}	94.3	79.1	96.7	96.0	**91.5**
Target Model: Qwen3-8B					
TraceGuard-S	93.7	75.6	96.8	96.3	90.6
TraceGuard-A_5	95.0	80.9	95.0	95.1	91.5
TraceGuard-A_{10}	95.7	82.0	95.2	95.1	92.0
TraceGuard-A_{20}	95.9	82.5	95.6	95.4	**92.3**
Target Model: Phi-4					
TraceGuard-S	93.2	76.8	96.1	96.3	90.6
TraceGuard-A_5	95.8	82.3	94.3	95.1	91.9
TraceGuard-A_{10}	94.7	79.7	95.8	95.4	91.4
TraceGuard-A_{20}	94.8	82.2	95.7	96.0	**92.2**

Effectiveness on Few-shot Adaptation. We further consider a practical scenario where a few newly observed attack examples are available for adaptation. Table 2 compares TraceGuard-S and TraceGuard-A across n-shot budgets using target-specific samples, with full results in Appendix D.4, confirming that adaptation improves jailbreak defense while preserving utility. The improvements are larger on unseen attacks, with H-ACC improving by 1.6–6.2%, compared with 0.3–2.2% on seen attacks, reflecting that adaptation mainly fits newly observed attack distributions. Additionally, B-ACC remains stable across budgets, dropping by at most 1.2% under the 20-shot setting, implying that the benign-aware constraint mitigates overfitting to harmful samples. The generalizability of our adaptation strategy to an external detector is evaluated in Appendix D.5.

Table 3: Efficiency on jailbreak datasets. Turn LAT and Inter. LAT denote average detection latency per turn and per jailbreak interaction.

Method	Turn LAT (s)	Inter. LAT (s)
EDDF	5.60	17.26
SelfDefend	0.80	3.11
PPL	0.04	0.12
JBShield-D	0.11	0.38
NBF	0.01	0.03
ShieldGemma	0.04	0.19
Qwen3-Guard	0.81	6.29
Llama4-Guard	0.62	3.04
TraceGuard-S	0.77	2.76
TraceGuard-A	0.78	1.99

Efficiency. Table 3 reveals that TraceGuard is lightweight for online detection, achieving sub-second per-turn LAT comparable to Qwen3-Guard and Llama4-Guard. Its overhead mainly comes from auxiliary-LLM risk representation extraction over long responses, making it slower than some lightweight baselines. At the interaction level, TraceGuard-S reduces LAT by 56.1% and 9.2% compared with Qwen3-Guard and Llama4-Guard by detecting risky trajectories earlier. Moreover, TraceGuard-A shortens interaction-level LAT by 27.9% over TraceGuard-S, suggesting earlier or more stable jailbreak defense after adaptation.

5.3 Further Analysis

Ablation Studies. We ablate TraceGuard-S and TraceGuard-A to assess three designs: cross-view risk representation, trajectory modeling, and distribution-aware adaptation. For risk representation, *w/o Response Risk*, *w/o Prompt Risk*, *w/ Scalar Risk*, and *w/ Text Embedding* remove one view, use one-dimensional risk signal, or replace LRH-based representations with raw text embeddings. For trajectory modeling, *w/o Trajectory Modeling* uses only current-turn risk signals, while *w/o Prefix Reg.* removes prefix-level regularization. For adaptation, *w/o Sequence Pre.*, *w/o Harmfulness Sco.*, and *w/o Benign-aware Reg.* ablate p^{seq}, p^{proto}, and benign-aware margin regularization, respectively. Table 4 demonstrates the contribution of each component, with further evidence for trajectory modeling beyond task-specific fine-tuning provided in Appendix D.6. Hyperparameter sensitivity for α, β, ρ, γ, and m is reported in Appendix D.7.

Table 4: Ablation study (%) of TraceGuard on Llama3-8B.

TraceGuard Variant	H-ACC ↑		B-ACC ↑		Overall
	Seen	Unseen	Seen	Unseen	ACC ↑
Standard detection mode					
TraceGuard-S	94.0	76.5	96.9	96.2	**90.9**
w/o Response Risk	96.7	80.3	92.2	89.9	89.8
w/o Prompt Risk	52.7	36.9	97.8	96.8	71.0
w/ Scalar Risk	83.7	58.1	74.7	80.4	74.2
w/ Text Embedding	75.7	26.8	97.7	92.1	73.1
w/o Trajectory Modeling	94.1	78.0	89.4	88.7	87.6
w/o Prefix Reg.	93.3	74.5	96.1	96.1	90.0
Adaptive detection mode					
TraceGuard-A	95.0	78.1	96.9	95.8	**91.4**
w/o Sequence Pre.	88.7	68.9	98.0	97.2	88.2
w/o Harmfulness Sco.	93.6	74.6	97.0	96.3	90.4
w/o Benign-aware Reg.	94.6	76.9	94.8	93.8	90.0

Turn Sensitivity Analysis. To examine the effect of dialogue history, we evaluate overall H-ACC on jailbreak datasets under different numbers of observed turns. From Fig. 3, H-ACC increases as more turns are observed and becomes stable after three turns, highlighting the importance of cross-turn modeling. Furthermore, TraceGuard-A consistently outperforms TraceGuard-S, especially with fewer observed turns, indicating that adaptation improves sensitivity to attack distributions and enables stronger early detection.

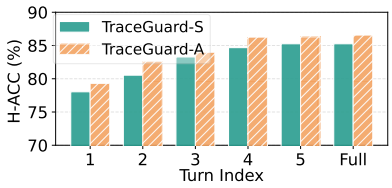


Figure 3: H-ACC variation with observed turns under Llama3-8B.

Risk Trajectory Visualization. Fig. 4 visualizes turn-wise harmfulness scores over five-turn interactions. Overall, jailbreak interactions exhibit increasing risk as the interaction progresses, whereas benign ones remain stable in the low-risk region, confirming the need for trajectory-level modeling. Moreover, seen and unseen attacks differ in score levels and growth rates, indicating distributional variation across attack types, while their shared upward trend suggests that TraceGuard captures generalizable multi-turn risk dynamics. Notably, TraceGuard-S detects seen attacks earlier but is less sensitive to unseen attacks, whereas TraceGuard-A yields smoother trajectories that better align unseen attacks with the harmful region while preserving benign separation, highlighting the benefit of adaptation.

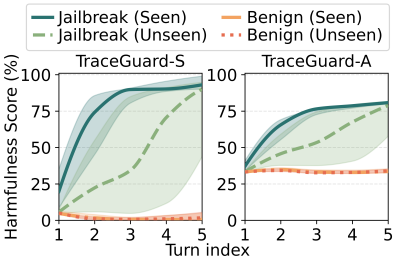


Figure 4: Turn-wise median harmfulness scores with IQR bands.

Generalization to Closed-source and Dynamic Attacks. We further assess TraceGuard under two generalization settings: closed-source target models and dynamic-prompt attacks. For both settings, TraceGuard-S and the LLaMA3-8B-adapted TraceGuard-A are applied without additional tuning. Fig. 5 demonstrates that both variants consistently outperform guard-model baselines on jailbreak datasets, indicating transferability to closed-source target models; full results are reported in Appendix D.8. We also evaluate dynamic-prompt attacks on closed-source models, using 50 multi-turn interactions for each attack. Fig. 6 shows that TraceGuard maintains strong H-ACC on XTeam and Crescendo, highlighting robustness to dynamic interactions. The consistent gains of TraceGuard-A over TraceGuard-S further indicate that few-shot adaptation transfers effectively across target models. Beyond target-model generalization, robustness to alternative representation encoders is discussed in Appendix D.9.

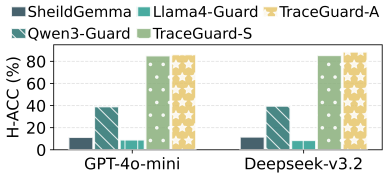


Figure 5: H-ACC on closed-source target models.

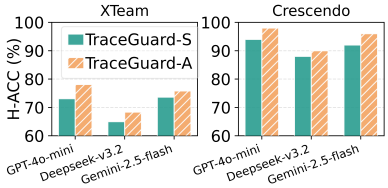


Figure 6: H-ACC under dynamic multi-turn attack trajectories.

Robustness against Adaptive Attacks. We evaluate TraceGuard against two adaptive attacks based on Crescendo, using 30 seed questions, GPT-3.5-Turbo as the attack generator, GPT-4o-mini as the target model, and GPT-4o as the attack success rate (ASR) evaluator. For the *score-aware attack*, the attacker accesses TraceGuard-S predictions and generates three candidate prompts per turn, preferring those with $p^{seq} < \tau$ and otherwise selecting the lowest-risk candidate with up to three-turn backtracking, while preserving the original harmful goal. For the *trajectory-disruption attack*, the attacker inserts three benign-looking turns (e.g., ethical discussions, safety reminders, or refusal statements) to disrupt risk evolution without accessing detection scores. We consider four insertion strategies: alternating (every other turn), middle-block (three consecutive turns in the middle), sandwich (before and near the final attack turns), and prefix (at the beginning). Table 5 shows that score-aware adaptation reduces detection performance but substantially lowers ASR, revealing a trade-off between evasion and attack effectiveness. In contrast, trajectory disruption reduces ASR while TraceGuard maintains high detection performance, demonstrating robustness to benign-turn insertion.

Table 5: Evaluation (%) against adaptive attacks.

Attack Setting	ASR ↓	H-ACC ↑	
		TraceGuard-S	TraceGuard-A
Score-aware attack			
Crescendo (Original)	56.7	96.7	96.7
Adaptive ($\tau = 0.4$)	0.0	26.7	40.0
Adaptive ($\tau = 0.6$)	3.3	56.7	63.3
Adaptive ($\tau = 0.8$)	6.7	60.0	73.3
Adaptive ($\tau = 1.0$)	10.0	73.3	83.3
Trajectory-disruption attack			
Alternating	6.7	96.7	96.7
Middle-block	13.3	100.0	100.0
Sandwich	3.3	90.0	96.7
Prefix	6.7	96.7	96.7

5.4 Discussion

While TraceGuard demonstrates strong performance in multi-turn jailbreak detection, several limitations remain. First, although our evaluation covers diverse attack strategies, some attacks share similar seed sources. Broader attack sources could further strengthen the assessment of generalization to unseen attacks. Second, TraceGuard relies on the representation quality of the auxiliary encoder, while predefined intent concepts may not fully capture domain-specific or emerging harms. Future work could explore more generalizable representations to improve cross-lingual and cross-domain transferability. Finally, TraceGuard-A enables efficient few-shot adaptation to unseen attacks but assumes access to a small set of reliably labeled support interactions. Future work could explore adaptation under imperfect supervision to improve practical applicability.

6 Conclusion

In this work, we investigate harmful intent in multi-turn jailbreak interactions through representation-space risk signals, revealing its fragmented, progressive, and attack-dependent nature. Based on this, we propose TraceGuard, an online black-box defense framework for multi-turn jailbreak detection that tracks prompt-response risk signals, with TraceGuard-S modeling cross-view risk evolution and TraceGuard-A enabling few-shot adaptation to unseen attacks. Extensive experiments demonstrate that TraceGuard outperforms existing approaches by improving jailbreak defense without sacrificing benign utility, with sub-second online detection and robust generalization to evolving attacks. Our potential future directions include extending TraceGuard to multimodal jailbreak settings and developing adaptive intent concept construction.

Acknowledgments and Disclosure of Funding

This work was supported by the National Natural Science Foundation of China (Grant Nos. 72595845 and 72595840), the National Artificial Intelligence Pilot Application Base (Financial Sector, Inclusive Finance & Payments) (Grant No. 2407-310115-04-04-808683), and the National Engineering Research Center of Electronic Commerce and Electronic Payment.

References

[1] Yubo Li, Xiaobin Shen, Xinyu Yao, Xueying Ding, Yidi Miao, Ramayya Krishnan, and Rema Padman. Beyond single-turn: A survey on multi-turn interactions with large language models. *arXiv preprint arXiv:2504.04717*, 2025.

[2] Konstantinos I Roumeliotis and Nikolaos D Tselikas. Chatgpt and open-ai models: A preliminary review. *Future Internet*, 15(6):192, 2023.

[3] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[4] Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

[5] Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. " do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 1671–1685, 2024.

[6] Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. Tree of attacks: Jailbreaking black-box llms automatically. *Advances in Neural Information Processing Systems*, 37:61065–61105, 2024.

[7] Hongyi Li, Jiawei Ye, Jie Wu, Tianjie Yan, Chu Wang, and Zhixin Li. Jailpo: A novel black-box jailbreak framework via preference optimization against aligned llms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, number 26, pages 27419–27427, 2025.

[8] Xikang Yang, Biyu Zhou, Xuehai Tang, Jizhong Han, and Songlin Hu. Chain of attack: Hide your intention through multi-turn interrogation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 9881–9901, 2025.

[9] Salman Rahman, Liwei Jiang, James Shiffer, Genglin Liu, Sheriff Issaka, Md Rizwan Parvez, Hamid Palangi, Kai-Wei Chang, Yejin Choi, and Saadia Gabriel. X-teaming: Multi-turn jailbreaks and defenses with adaptive multi-agents. *arXiv preprint arXiv:2504.13203*, 2025.

[10] Zhangchen Xu, Fengqing Jiang, Luyao Niu, Jinyuan Jia, Bill Yuchen Lin, and Radha Poovendran. Safedecoding: Defending against jailbreak attacks via safety-aware decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5587–5605, 2024.

[11] Shilong Pan, Zhiliang Tian, Zhen Huang, Wanlong Yu, Zhihua Wen, Xinwang Liu, Kai Lu, Minlie Huang, and Dongsheng Li. Agd: Adversarial game defense against jailbreak attacks in large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 17391–17406, 2025.

[12] Xiaohu Li, Yunfeng Ning, Zepeng Bao, Mayi Xu, Jianhao Chen, and Tieyun Qian. Cavgan: Unifying jailbreak and defense of llms via generative adversarial attacks on their internal representations. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 6664–6678, 2025.

[13] Rui Pu, Chaozhuo Li, Rui Ha, Litian Zhang, Lirong Qiu, and Xi Zhang. Beyond surface-level detection: Towards cognitive-driven defense against jailbreak attacks via meta-operations reasoning. *arXiv preprint arXiv:2508.03054*, 2025.

[14] Xunguang Wang, Daoyuan Wu, Zhenlan Ji, Zongjie Li, Pingchuan Ma, Shuai Wang, Yingjiu Li, Yang Liu, Ning Liu, and Juergen Rahmel. Selfdefend: Llms can defend themselves against jailbreaking in a practical manner. In *34th USENIX Security Symposium (USENIX Security 25)*, pages 2441–2460, 2025.

[15] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

[16] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[17] Gabriel Alon and Michael Kamfonas. Detecting language model attacks with perplexity. *arXiv preprint arXiv:2308.14132*, 2023.

[18] Shiyu Xiang, Ansen Zhang, Yanfei Cao, Fan Yang, and Ronghao Chen. Beyond surface-level patterns: An essence-driven defense framework against jailbreak attacks in llms. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 14727–14742, 2025.

[19] Meta Llama. Llama-guard-4-12b. `https://huggingface.co/meta-llama/Llama-Guard-4-12B`, 2025. Hugging Face model card. Accessed: 2026-04-27.

[20] Haiquan Zhao, Chenhan Yuan, Fei Huang, Xiaomeng Hu, Yichang Zhang, An Yang, Bowen Yu, Dayiheng Liu, Jingren Zhou, Junyang Lin, et al. Qwen3-guard technical report. *arXiv preprint arXiv:2510.14276*, 2025.

[21] Wenjun Zeng, Yuchi Liu, Ryan Mullins, Ludovic Peran, Joe Fernandez, Hamza Harkous, Karthik Narasimhan, Drew Proud, Piyush Kumar, Bhaktipriya Radharapu, Olivia Sturman, and Oscar Wahltinez. Shieldgemma: Generative ai content moderation based on gemma, 2024.

[22] Zonghao Ying, Deyue Zhang, Zonglei Jing, Yisong Xiao, Quanchen Zou, Aishan Liu, Siyuan Liang, Xiangzheng Zhang, Xianglong Liu, and Dacheng Tao. Reasoning-augmented conversation for multi-turn jailbreak attacks on large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 17138–17157, November 2025.

[23] Mark Russinovich, Ahmed Salem, and Ronen Eldan. Great, now write an article about that: The crescendo multi-turn llm jailbreak attack. In *34th USENIX Security Symposium (USENIX Security 25)*, pages 2421–2440, 2025.

[24] Jinwei Hu, Yi Dong, and Xiaowei Huang. Trust-oriented adaptive guardrails for large language models. *arXiv preprint arXiv:2408.08959*, 2024.

[25] Hanjiang Hu, Alexander Robey, and Changliu Liu. Steering dialogue dynamics for robustness against multi-turn jailbreaking attacks. Transactions on Machine Learning Research, 2026. Accepted by TMLR.

[26] Shenyi Zhang, Yuchen Zhai, Keyan Guo, Hongxin Hu, Shengnan Guo, Zheng Fang, Lingchen Zhao, Chao Shen, Cong Wang, and Qian Wang. Jbshield: Defending large language models from jailbreak attacks through activated concept analysis and manipulation. In *34th USENIX Security Symposium (USENIX Security 25)*, pages 8215–8234, 2025.

[27] Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria*, Proceedings of Machine Learning Research, pages 39643–39666.

[28] Hongye Cao, Sijia Jing, Yanming Wang, Ziyue Peng, Zhixin Bai, Zhe Cao, Meng Fang, Fan Feng, Jiaheng Liu, Boyan Wang, Tianpei Yang, Jing Huo, Yang Gao, Fanyu Meng, Xi Yang, Chao Deng, and Junlan Feng. Safedialbench: A fine-grained safety evaluation benchmark for large language models in multi-turn dialogues with diverse jailbreak attacks. In *International Conference on Learning Representations*, 2026.

[29] Yifan Jiang, Kriti Aggarwal, Tanmay Laud, Kashif Munir, Jay Pujara, and Subhabrata Mukherjee. Red queen: Exposing latent multi-turn risks in large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 25554–25591, 2025.

[30] Jiaming Ji, Mickel Liu, Josef Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Advances in Neural Information Processing Systems*, 36: 24678–24704, 2023.

[31] Erxin Yu, Jing Li, Ming Liao, Siqi Wang, Gao Zuchen, Fei Mi, and Lanqing Hong. CoSafe: Evaluating large language model safety in multi-turn dialogue coreference. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17494–17508, November 2024.

[32] Chia-Yi Hsu, Yu-Lin Tsai, Chih-Hsun Lin, Pin-Yu Chen, Chia-Mu Yu, and Chun-Ying Huang. Safe lora: The silver lining of reducing safety risks when finetuning large language models. *Advances in neural information processing systems*, 37:65072–65094, 2024.

[33] Youliang Yuan, Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Jiahao Xu, Tian Liang, Pinjia He, and Zhaopeng Tu. Refuse whenever you feel unsafe: Improving safety in LLMs via decoupled refusal training. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3149–3167, July 2025.

[34] Chujie Zheng, Fan Yin, Hao Zhou, Fandong Meng, Jie Zhou, Kai-Wei Chang, Minlie Huang, and Nanyun Peng. On prompt-driven safeguarding for large language models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[35] Xiaoya Lu, Dongrui Liu, Yi Yu, Luxin Xu, and Jing Shao. X-boundary: Establishing exact safety boundary to shield LLMs from jailbreak attacks without compromising usability. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 5247–5272, Suzhou, China, November 2025.

[36] Weiyang Guo, Jing Li, Wenya Wang, Yu Li, Daojing He, Jun Yu, and Min Zhang. MTSA: Multi-turn safety alignment for LLMs through multi-round red-teaming. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 26424–26442, July 2025.

[37] Nathaniel Li, Ziwen Han, Ian Steneker, Willow Primack, Riley Goodside, Hugh Zhang, Zifan Wang, Cristina Menghini, and Summer Yue. Llm defenses are not robust to multi-turn human jailbreaks yet. *arXiv preprint arXiv:2408.15221*, 2024.

[38] Neel Nanda, Andrew Lee, and Martin Wattenberg. Emergent linear representations in world models of self-supervised sequence models. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 16–30, 2023.

[39] Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, et al. Toy models of superposition. *arXiv preprint arXiv:2209.10652*, 2022.

[40] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[41] Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms. *Advances in neural information processing systems*, 37:8093–8131, 2024.

[42] Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.

[43] Paul Röttger, Hannah Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. Xstest: A test suite for identifying exaggerated safety behaviours in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5377–5400, 2024.

[44] Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, 2023.

[45] OpenAI. Moderation. `https://developers.openai.com/api/docs/guides/moderation`, 2025. Accessed: 2026-04-28.

[46] Rahul Dey and Fathi M Salem. Gate-variants of gated recurrent unit (gru) neural networks. In *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*, pages 1597–1600. IEEE, 2017.

[47] Qibing Ren, Hao Li, Dongrui Liu, Zhanxu Xie, Xiaoya Lu, Yu Qiao, Lei Sha, Junchi Yan, Lizhuang Ma, and Jing Shao. Llms know their vulnerabilities: Uncover safety gaps through natural distribution shifts. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vienna, Austria, July 2025.

[48] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric P Xing, et al. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. *arXiv preprint arXiv:2309.11998*, 2023.

[49] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. Dailydialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995, 2017.

[50] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, 2018.

[51] Ge Bai, Jie Liu, Xingyuan Bu, Yancheng He, Jiaheng Liu, Zhanhui Zhou, Zhuoran Lin, Wenbo Su, Tiezheng Ge, Bo Zheng, et al. Mt-bench-101: A fine-grained benchmark for evaluating large language models in multi-turn dialogues. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7421–7454, 2024.

[52] Glaive AI. Glaive function calling v2. `https://huggingface.co/datasets/glaiveai/glaive-function-calling-v2`, 2023. Hugging Face dataset. Accessed: 2026-04-28.

[53] Beitao Chen, Xinyu Lyu, Lianli Gao, Jingkuan Song, and Heng Tao Shen. Safeptr: Token-level jailbreak defense in multimodal llms via prune-then-restore mechanism. *arXiv preprint arXiv:2507.01513*, 2025.

[54] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NIPS)*, volume 30, pages 4077–4087, 2017.

A Implementation for Risk Signal Analysis

This section provides implementation details for Section 3.3. For prompts, calibration prompts are constructed using 30 benign/harmful prompts from JBShield [26]. For responses, candidate benign and harmful responses are sampled from WildGuard [41] and manually inspected to obtain 30 benign/harmful compliant response pairs. We use Llama3-8B [40] as the auxiliary representation encoder. The extraction layer is selected by measuring harmful-benign separability across Transformer layers on the calibration samples, resulting in layer 24 for prompt representations and layer 32 for response representations. Following the intent concept construction in Section 3.2, the prompt intent concept v_q and response intent concept v_r are obtained as harmful-intent directions for prompts and responses, respectively. For each dataset in Section 3.3, 100 interactions are randomly sampled and manually filtered for analysis, except for RedQueen and XTeam, where 100 five-turn jailbreak interactions generated with GPT-4o-mini are used. For each interaction, prompt and response risk signals are computed at every turn to analyze how harmful intent changes over the interaction.

B Structured Intent Concept Construction

B.1 Prompt Intent Concepts

For prompt intent concepts, we use the general harmful–benign category from JBShield [26] and 14 fine-grained harmful intent categories from BeaverTails [30], as listed in Table 6. The benign calibration set is consistently drawn from JBShield. For each harmful category, including the general harmful category from JBShield and the 14 BeaverTails categories, 30 calibration prompts are selected to construct the corresponding anchor representation and concept vector.

Table 6: Prompt harmful intent categories used for constructing intent concepts.

ID	Harmful Intent Category
1	General (from JBShield)
2	Animal Abuse
3	Child Abuse
4	Controversial Topics, Politics
5	Discrimination, Stereotype, Injustice
6	Drug Abuse, Weapons, Banned Substance
7	Financial Crime, Property Crime, Theft
8	Hate Speech, Offensive Language
9	Misinformation Regarding Ethics, Laws, and Safety
10	Non-Violent Unethical Behavior
11	Privacy Violation
12	Self-Harm
13	Sexually Explicit, Adult Content
14	Terrorism, Organized Crime
15	Violence, Aiding and Abetting, Incitement

B.2 Response Intent Concepts

Following WildGuard [41], responses are categorized into three behavior types: harmful compliance, safe refusal, and benign compliance. Based on these categories, we construct three response intent concepts: a *harmfulness* concept that distinguishes harmful compliance from benign compliance, a *safety* concept that distinguishes safe refusal from harmful compliance, and a *utility* concept that distinguishes benign compliance from safe refusal. These concepts capture alignment with harmful responses, safety-aligned refusals, and helpful benign responses, respectively. For each response category, we sample candidate responses and manually filter them to obtain 30 calibration samples. These samples are then used to construct the corresponding anchor representations and response intent concept vectors.

C Experimental Details

C.1 Dataset Details

Benign Datasets. We use six benign multi-turn datasets covering diverse interaction scenarios. Ultra [44] and LMSYS [48] are general multi-turn model conversation datasets; DailyDialog [49] and Persona [50] contain natural daily conversations and role-play dialogues; MTBench101 (MT-Bench) [51] focuses on multi-turn knowledge-oriented question answering; and GFC [52] contains multi-turn function-calling dialogues. These datasets are used to evaluate whether the detector preserves benign interactions across different domains.

Jailbreak Datasets. For jailbreak evaluation, six multi-turn jailbreak datasets are employed. SafeDialBench [28] and CoSafe [31] are publicly released multi-turn jailbreak benchmarks. RedQueen [29] and XTeam [9] are collected from their official released datasets. Following the original attack settings, ActorAttack [47] and Crescendo [23] are generated from AdvBench [42] using GPT-3.5-Turbo as the target model, yielding 520 attack attempts before filtering. To avoid overlap between

Table 7: Dataset statistics. "Split" indicates whether the dataset is used as seen or unseen in our evaluation.

Dataset	Type	Origin Size	Description	Split	Turns	Avg. Turns
Ultra	Benign	6,127	General model dialogue	Seen	2–7	3.99
DailyDialog	Benign	11,118	Daily natural dialogue	Seen	3–16	3.95
Persona-Chat	Benign	8,938	Persona-grounded dialogue	Seen	6–27	13.39
LMSYS-Chat-1M	Benign	2,000	General model dialogue	Unseen	3–20	4.51
MT-Bench-101	Benign	1,388	Multi-turn QA	Unseen	2–5	3.01
GFC	Benign	112,960	Function-calling dialogue	Unseen	3–10	4.01
RedQueen	Harmful	14,000	Released attack dataset	Seen	5	5.00
XTeam	Harmful	30,674	Released attack dataset	Seen	2–7	4.97
SafeDialBench	Harmful	2,037	Multi-turn jailbreak benchmark	Seen	3–9	4.96
CoSafe	Harmful	1,400	Multi-turn jailbreak benchmark	Unseen	3	3.00
ActorAttack	Harmful	520	Multi-turn jailbreak attack	Unseen	4–5	4.99
Crescendo	Harmful	520	Multi-turn jailbreak attack	Unseen	3–11	5.85

dataset construction and evaluation, we remove samples generated by the target models used in our experiments.

Dataset Splits. We split the datasets into seen and unseen groups to evaluate standard detection and generalization, as shown in Table 7. For each seen dataset, 2,000 interactions are randomly sampled and split into training, validation, and test sets with a ratio of 70/15/15. For each unseen dataset, 100 interactions are sampled as the support pool for few-shot adaptation and 300 interactions for testing. All testing samples are disjoint from the training, validation, and adaptation data.

C.2 Implementation Details

TraceGuard-S Training. TraceGuard-S is trained on the seen datasets with standard supervision using the PyTorch Adam optimizer. We use a batch size of 32, a learning rate of 1×10^{-3}, and at most 20 epochs, with early stopping based on validation F1 and a patience of 5 epochs. The trajectory module consists of two independent `nn.GRUCell` modules for prompt and response states, each with a hidden size of 64 and zero initialization. Gradient clipping is applied with a threshold of 1.0.

TraceGuard-A Adaptation. TraceGuard-A adapts TraceGuard-S to unseen attack datasets with n-shot support examples, where $n \in \{5, 10, 20\}$ and 10-shot is used by default. The support set is sampled from the target unseen dataset, together with additional benign and harmful samples from the TraceGuard-S training pool. During adaptation, TraceGuard-S is frozen and only the lightweight fusion layer is optimized. The fusion layer is trained for up to 20 epochs with learning rate 0.05. Each epoch runs once over the n-shot support set.

C.3 Baseline Details

This section provides implementation details and evaluation protocols for all baseline methods used in our experiments.

NBF. NBF [25] is a prompt-based baseline designed for multi-turn jailbreak detection. Since it supports history-aware detection, we provide the full dialogue prefix available at each turn as input. Specifically, at turn t, the input consists of the interaction prefix $\mathcal{I}_{\leq t}$, including all previous user prompts and model responses, together with the current user prompt. This setting evaluates whether explicitly concatenating dialogue history is sufficient for detecting multi-turn jailbreak attacks.

EDDF. EDDF [18] is a single-turn jailbreak detection baseline originally designed to identify harmful intent from individual prompts. Following its original pipeline, we apply EDDF to the current user prompt at each turn. This setting evaluates whether single-turn prompt-level detection remains effective in multi-turn attack scenarios, where harmful intent may be implicit or distributed across turns.

SelfDefend. SelfDefend [14] detects unsafe user requests through an LLM-based self-defense procedure. Following its original setting, we use the current user prompt as input and query the detector to determine whether the prompt should be regarded as unsafe. In the multi-turn setting, SelfDefend is evaluated independently at each turn without maintaining an explicit interaction-level risk state.

Table 8: Comparison between the officially released NBF model and the retrained NBF model on our test datasets.

Methods	H-ACC ↑						B-ACC ↑						Overall ACC ↑
	Seen			Unseen			Seen			Unseen			
	RedQ	XTeam	SafeDial	CoSafe	ActorAtk	Cresc	Ultra	Daily	Persona	LMSYS	MTBench	GFC	
NBF-Official	93.3	93.3	54.6	33.7	51.2	65.6	43.6	87.3	72.6	55.0	80.0	32.7	63.6
NBF-Retrain	62.6	76.3	75.3	26.7	67.2	81.6	99.0	100.0	100.0	90.5	94.2	100.0	81.1

PPL. PPL [17] is a perplexity-based detection baseline that identifies abnormal or adversarial prompts according to their language-model perplexity. To make it applicable under the black-box setting while ensuring a fair comparison, we use Llama3-8B as the auxiliary language model for perplexity estimation. At each turn, we compute the perplexity score of the current user prompt and assign the detection label using a threshold selected on the validation set. Since PPL does not explicitly incorporate previous turns, it serves as a lightweight single-turn jailbreak detection baseline.

JBShield-D. JBShield-D [26] is a representation-based single-turn jailbreak detector built upon the LRH. Consistent with our black-box assumption, we instantiate its auxiliary representation encoder with Llama3-8B rather than relying on any internal states of the target model. We then apply JBShield-D to the current user prompt at each turn. This baseline examines whether static harmful-intent representations derived from single-turn prompts can generalize to multi-turn jailbreak interactions.

ShieldGemma. ShieldGemma [21] is a generative AI content safety classifier. To adapt it to online multi-turn detection, we provide the full interaction prefix available at each turn and ask the model to classify whether the current turn is unsafe. This history-aware setting gives the guard model access to all observed context before making a decision.

Qwen3-Guard. Qwen3-Guard [20] is a guard model for safety moderation. We use Qwen3Guard-Gen-8B as the guard-model baseline. At each turn, we feed the full interaction prefix to the model using its recommended chat format and map its safety judgment to a binary harmful/benign decision. This setting allows Qwen3-Guard to perform context-conditioned moderation over multi-turn interaction histories.

Llama4-Guard. Llama4-Guard [19] is a Llama-based safety classifier for detecting unsafe content in interactions. Similarly to Qwen3-Guard and ShieldGemma, we provide the full interaction prefix at each turn and convert the model output into a binary detection label.

Discussion on Baseline Evaluation. To ensure fair comparison, we evaluate all baselines under the same online detection protocol. At each turn, each method follows the input format supported by its original design: history-aware baselines, including NBF and guard models, receive the full interaction prefix, whereas single-turn detectors are applied to the current user prompt. For baselines that require an auxiliary model, we use Llama3-8B whenever applicable. We further standardize all outputs into binary harmful/benign labels using validation-set thresholds or the recommended label-mapping rules. This setup aligns all methods in terms of access assumptions and available context.

Discussion on Few-shot Adaptation Baselines. Existing jailbreak detectors are primarily designed for zero-shot or supervised detection and do not include explicit mechanisms for few-shot adaptation to newly observed attack distributions. Directly fine-tuning such detectors with a few harmful examples would require method-specific design choices, such as selecting trainable modules, balancing benign samples, and tuning thresholds, which may introduce additional confounding factors. Therefore, our main adaptive-setting evaluation compares TraceGuard-A with TraceGuard-S under the same backbone and support sets, and uses controlled ablations to isolate the contribution of sequence prediction, prototype-based harmfulness scoring, and benign-aware regularization.

C.4 NBF Reproduction and Retraining

NBF [25] is a multi-turn jailbreak defense method that performs sequence modeling over text-encoded interaction histories. To ensure a fair comparison under our evaluation setting, we retrain NBF on our training data using the official implementation. Following the original NBF setup, we first score all model responses in our training interactions with GPT-4o and then train the sequence detector.

Table 8 compares the released NBF model with our retrained version on our test datasets. We use NBF-Retrain in the main experiments for fairness, as it is trained on the same split as TraceGuard-S. Compared with the released model, NBF-Retrain achieves much higher B-ACC on benign datasets. This is mainly because the released NBF model was trained without benign multi-turn datasets, which limits its generalization to benign interactions in our evaluation. However, its H-ACC remains limited on several unseen attacks, suggesting that text-based sequence modeling alone struggles with dynamic attack patterns.

D Additional Experimental Results

D.1 Evaluation on Over-safety Benign Datasets.

To further evaluate whether TraceGuard introduces excessive false positives, we conduct an additional evaluation on the over-safety dataset XStest [43], where benign single-turn prompts contain safety-sensitive terms but should not be blocked. This setting tests benign preservation beyond standard benign datasets. As shown in Table 9, TraceGuard-S achieves 100.0% B-ACC across all target models, indicating that its risk trajectory modeling does not rely on surface-level safety keywords. TraceGuard-A also maintains competitive B-ACC, ranging from 94.4% to 98.3%, suggesting that few-shot adaptation preserves most benign cases despite adapting toward harmful examples. These results further support the utility-preserving property of TraceGuard under challenging benign prompts.

Table 9: B-ACC (%) on XStest.

Methods	B-ACC ↑
Prompt-only methods	
EDDF	92.6
NBF	97.8
SelfDefend	89.7
PPL	83.8
JBShield-D	99.3
Target Model: Llama3-8B	
ShieldGemma	99.3
Qwen3-Guard	96.3
Llama4-Guard	100.0
TraceGuard-S	**100.0**
TraceGuard-A	94.4
Target Model: Qwen3-8B	
ShieldGemma	100.0
Qwen3-Guard	99.3
Llama4-Guard	99.3
TraceGuard-S	**100.0**
TraceGuard-A	98.3
Target Model: Phi-4	
ShieldGemma	100.0
Qwen3-Guard	98.5
Llama4-Guard	100.0
TraceGuard-S	**100.0**
TraceGuard-A	94.4

D.2 Comparison at Matched Operating Points

To examine the effect of threshold selection, we adjust TraceGuard's detection thresholds to match the benign accuracy (B-ACC) of each guard baseline and compare harmful detection accuracy (H-ACC) at the same operating points. Table 10 shows that TraceGuard maintains substantially higher harmful detection accuracy across matched benign accuracy levels, indicating that the performance gap is not due to a more favorable threshold. The near-zero harmful accuracy of some guard baselines is likely related to their conservative default moderation behavior, which prioritizes benign preservation but may reduce sensitivity to harmful intent distributed across multi-turn trajectories.

D.3 Further Analysis on RedQueen

To investigate RedQueen's saturated detection performance, we examine template overlap and trajectory-level risk patterns. We confirm no overlap in harmful origin prompts between training and test splits. As RedQueen uses 10 manually designed scenario templates, we conduct a template-disjoint evaluation with 5 templates for training and 5 for testing, restricting validation to training templates. Both TraceGuard-S and TraceGuard-A achieve 100% H-ACC across evaluated target models, suggesting that the results are not solely explained by template overlap.

Table 10: H-ACC (%) on Llama3-8B at matched B-ACC levels.

Method	Seen H-ACC	Unseen H-ACC
Matched to ShieldGemma (99.9% B-ACC)		
ShieldGemma	4.7	3.8
TraceGuard-S	30.6	16.6
TraceGuard-A	**51.9**	**23.1**
Matched to Qwen3-Guard (99.6% B-ACC)		
Qwen3-Guard	38.4	47.1
TraceGuard-S	68.2	50.7
TraceGuard-A	**77.7**	**55.1**
Matched to Llama4-Guard (97.7% B-ACC)		
Llama4-Guard	14.2	8.3
TraceGuard-S	**89.6**	69.8
TraceGuard-A	89.3	**70.1**

We further analyze trajectory consistency in the concept space. RedQueen exhibits highly consistent risk dynamics, with pairwise cosine similarities above 0.985 across templates and a mean similarity of 0.996 ± 0.004 across trajectories, compared with 0.778 ± 0.168 for XTeam and 0.962 ± 0.056 for SafeDialBench. These results reveal highly consistent risk trajectories in RedQueen, providing further insight into its saturated detection performance.

Table 11: Effectiveness (%) of TraceGuard under different adaptation budgets. Results are reported as mean±standard deviation over five runs.

Methods	H-ACC ↑						B-ACC ↑						Overall ACC ↑
	Seen			Unseen			Seen			Unseen			
	RedQ	XTeam	SafeDial	CoSafe	ActorAtk	Cresc	Ultra	Daily	Persona	LMSYS	MTBench	GFC	
Target Model: Llama3-8B													
TraceGuard-S	100.0±0.0	90.0±0.5	92.0±0.5	72.0±1.0	73.5±0.7	84.0±0.8	94.0±0.3	96.7±0.3	100.0±0.0	92.5±0.4	96.0±0.3	100.0±0.0	90.9±0.4
TraceGuard-A_5	100.0±0.0	89.3±0.7	94.0±0.3	74.7±1.0	76.5±0.9	85.6±0.4	92.3±0.3	96.0±0.2	99.6±0.0	92.0±0.3	95.0±0.4	100.0±0.0	91.3±0.3
TraceGuard-A_{10}	100.0±0.0	90.3±0.5	94.6±0.2	75.2±0.8	74.5±0.6	84.5±0.8	94.0±0.3	96.6±0.1	100.0±0.0	92.2±0.3	95.0±0.4	100.0±0.0	91.4±0.3
TraceGuard-A_{20}	100.0±0.0	88.3±0.4	94.6±0.2	75.0±0.7	75.8±0.5	86.6±0.7	93.6±0.2	96.3±0.2	100.0±0.0	92.2±0.4	95.5±0.3	100.0±0.0	**91.5±0.5**
Target Model: Qwen3-8B													
TraceGuard-S	100.0±0.0	89.9±0.7	91.1±0.2	70.7±0.6	72.3±0.8	83.6±0.6	94.0±0.2	96.3±0.1	100.0±0.0	92.2±0.3	96.5±0.2	100.0±0.0	90.6±0.5
TraceGuard-A_5	100.0±0.0	89.6±0.6	95.3±0.2	78.5±0.6	77.7±0.7	86.5±0.3	92.3±0.5	93.6±0.4	99.0±0.1	91.7±0.4	93.4±0.2	100.0±0.0	91.5±0.4
TraceGuard-A_{10}	100.0±0.0	91.6±0.2	95.3±0.2	80.5±0.5	78.5±0.5	87.0±0.4	91.6±0.5	94.6±0.4	99.3±0.1	91.7±0.5	93.5±0.2	100.0±0.0	92.0±0.4
TraceGuard-A_{20}	100.0±0.0	92.0±0.2	95.7±0.3	81.2±0.5	78.5±0.8	87.5±0.4	91.6±0.5	95.3±0.3	99.6±0.1	92.0±0.5	94.2±0.3	100.0±0.0	**92.3±0.5**
Target Model: Phi-4													
TraceGuard-S	100.0±0.0	88.6±0.5	91.0±0.4	72.2±1.1	73.2±0.5	84.8±0.8	92.3±0.4	96.3±0.1	99.6±0.1	92.5±0.3	96.2±0.2	100.0±0.0	90.6±0.4
TraceGuard-A_5	100.0±0.0	91.6±0.5	95.6±0.2	79.7±0.7	79.2±0.6	87.8±0.4	90.3±0.5	93.2±0.3	99.3±0.1	91.5±0.3	93.7±0.3	100.0±0.0	91.9±0.3
TraceGuard-A_{10}	100.0±0.0	89.3±0.5	94.6±0.2	75.5±0.5	76.5±0.9	87.1±0.6	92.6±0.3	95.0±0.4	99.6±0.1	92.0±0.5	94.2±0.3	100.0±0.0	91.4±0.5
TraceGuard-A_{20}	100.0±0.0	90.6±0.2	93.6±0.6	81.0±0.6	78.5±0.7	87.1±0.3	92.0±0.3	95.6±0.1	99.3±0.1	92.2±0.3	95.7±0.2	100.0±0.0	**92.2±0.4**

Table 13: H-ACC (%) of fine-tuned baselines and TraceGuard-S on Llama3-8B at matched B-ACC (95.0%).

Method	Seen H-ACC ↑			Unseen H-ACC ↑		
	RedQ	XTeam	SafeDial	CoSafe	ActorAtk	Cresc
Qwen3-Guard FT	**100.0**	**100.0**	96.0	68.3	54.3	85.6
Llama3-8B FT	**100.0**	**100.0**	**99.0**	65.5	48.5	81.0
TraceGuard-S	**100.0**	**100.0**	92.3	**80.0**	**76.8**	**88.1**

D.4 Effectiveness on Few-shot Adaptation

Table 11 provides the full TraceGuard-S and TraceGuard-A results across three target models, including the static results over five runs. Consistent with the few-shot analysis in Table 2, TraceGuard-A consistently improves over TraceGuard-S across n-shot budgets, particularly on unseen attacks, while maintaining stable B-ACC. The small variance across static runs indicates that the observed improvements are robust across random trials.

D.5 Generalization of Few-shot Adaptation

To assess whether our distribution-aware adaptation strategy generalizes beyond TraceGuard, we apply it to NBF-Official (Appendix C.4) with Llama3-8B as the target model under a 10-shot setting. We replace TraceGuard-specific components with those of NBF and compare against classifier-head fine-tuning (FT) and ProtoNet [54]. All methods use the same support set without method-specific threshold tuning. Table 12 shows that FT and ProtoNet improve harmful detection at the cost of substantially lower benign accuracy. In contrast, our adaptation strategy achieves a better balance between H-ACC and B-ACC, demonstrating its transferability to an external detector.

Table 12: Comparison (%) of few-shot adaptation methods on NBF with Llama3-8B.

Method	H-ACC ↑		B-ACC ↑	
	Seen	Unseen	Seen	Unseen
NBF	80.5	50.2	67.9	55.9
NBF + FT	**97.6**	**94.3**	42.8	28.0
NBF + ProtoNet	96.2	91.8	57.1	41.2
NBF + Ours	82.0	76.2	**87.2**	**70.2**

D.6 Comparison with Fine-tuned Guard Baselines

To assess whether TraceGuard's gains stem from trajectory modeling rather than task-specific fine-tuning, we compare against fine-tuned Qwen3-Guard. To further isolate the contribution of trajectory modeling, we fine-tune a Llama3-8B classifier using the same backbone and training data as TraceGuard-S, but without trajectory-level modeling. All methods follow the same online prefix evaluation protocol and are evaluated on the Llama3-8B target model at a matched B-ACC of 95.0%. Table 13 shows that fine-tuned baselines perform competitively on seen attacks but achieve lower average H-ACC on unseen attacks. TraceGuard-S outperforms Qwen3-Guard FT and Llama3-8B FT by 12.3% and 16.6%, respectively, suggesting that trajectory modeling contributes beyond task-specific fine-tuning.

D.7 Hyperparameter Sensitivity

Table 14: Sensitivity analysis (%) of γ and m in the benign-aware regularization. Rows denote γ, and columns denote m.

B-ACC ↑

$\gamma \backslash m$	0	−0.25	−0.5	−0.75	−1
0	56.2	94.5	93.3	94.6	0.1
0.1	95.6	93.7	94.4	94.5	94.5
0.25	94.4	94.5	96.4	94.3	95.5
0.5	96.3	95.3	96.3	96.4	96.2
0.75	95.1	96.4	0.1	95.0	98.5
1	95.6	96.2	96.3	95.5	96.4

H-ACC ↑

$\gamma \backslash m$	0	−0.25	−0.5	−0.75	−1
0	97.5	89.5	88.9	89.3	85.0
0.1	86.6	90.3	89.6	89.6	89.6
0.25	89.9	89.6	85.0	89.7	87.2
0.5	85.2	87.5	85.3	84.6	85.6
0.75	87.6	84.8	97.2	88.0	73.5
1	86.9	85.4	85.4	87.1	85.0

Overall ACC ↑

$\gamma \backslash m$	0	−0.25	−0.5	−0.75	−1
0	76.9	92.0	91.1	92.0	45.4
0.1	91.1	92.0	92.0	92.1	92.1
0.25	92.2	92.1	90.7	92.0	91.3
0.5	90.7	91.4	90.8	90.5	90.9
0.75	91.4	90.6	48.6	91.5	86.0
1	91.2	90.8	90.8	91.3	90.7

Table 15: Effectiveness (%) on closed-source target models.

Methods	H-ACC ↑						B-ACC ↑						Overall ACC ↑
	Seen			Unseen			Seen			Unseen			
	RedQ	XTeam	SafeDial	CoSafe	ActorAtk	Cresc	Ultra	Daily	Persona	LMSYS	MTBench	GFC	
Target Model: GPT-4o-mini													
ShieldGemma	17.0	7.0	9.7	15.5	14.5	4.5	99.7	96.7	99.7	97.6	99.8	0.0	46.8
Qwen3-Guard	30.7	37.0	44.3	23.3	40.0	59.2	100.0	99.3	100.0	99.0	100.0	99.3	69.3
Llama4-Guard	5.7	16.3	12.3	4.8	4.5	10.5	99.0	100.0	99.7	91.8	98.5	98.8	53.5
TraceGuard-S	100.0	90.3	92.0	71.8	73.3	83.6	94.0	96.7	100.0	92.6	96.0	100.0	90.8
TraceGuard-A	100.0	92.2	92.7	73.8	74.3	84.1	94.3	96.7	99.7	92.8	95.0	100.0	**91.3**
Target Model: DeepSeek-v3.2													
ShieldGemma	16.3	7.3	10.3	15.5	13.8	6.5	99.7	96.3	99.7	99.8	100.0	92.7	54.8
Qwen3-Guard	27.0	39.0	46.0	22.8	40.5	62.2	100.0	99.3	100.0	100.0	99.8	97.2	69.5
Llama4-Guard	5.3	15.3	11.3	5.3	4.8	9.5	99.0	100.0	99.7	98.3	98.8	100.0	53.9
TraceGuard-S	100.0	90.0	94.0	71.8	73.3	83.6	94.0	96.7	100.0	92.5	96.0	100.0	91.0
TraceGuard-A	100.0	91.3	95.3	80.5	77.0	87.1	92.3	95.0	99.3	92.0	93.8	100.0	**92.0**

Sensitivity of Benign-aware Regularization. Table 14 analyzes the sensitivity of γ and m in the benign-aware regularization. The results show that the model remains stable across a broad range of settings, while extreme combinations may over-constrain benign samples or insufficiently protect them. Based on this balance, we set $\gamma = 0.5$ and $m = -0.5$ in the main experiments. This confirms that the benign-aware constraint helps control the safety–utility trade-off during adaptation.

Table 16: Results (%) on different representation encoder. "Ext." denotes the feature extractor used to construct risk representations.

Extractor Variant	H-ACC ↑		B-ACC ↑		Overall ACC ↑
	Seen	Unseen	Seen	Unseen	
Target Model: Llama3-8B					
TraceGuard-S	94.0	76.5	96.9	96.2	**90.9**
w/ Qwen3-8B Ext.	90.9	65.0	97.1	91.3	86.1
w/ Phi-4 Ext.	93.6	62.6	92.6	88.9	84.4
Target Model: Qwen3-8B					
TraceGuard-S	93.7	75.6	96.8	96.3	**90.6**
w/ Qwen3-8B Ext.	90.7	65.2	97.0	91.4	86.1
w/ Phi-4 Ext.	93.4	62.3	92.3	91.7	84.9
Target Model: Phi-4					
TraceGuard-S	93.2	76.8	96.1	96.3	**90.6**
w/ Qwen3-8B Ext.	90.9	65.0	97.1	91.3	86.1
w/ Phi-4 Ext.	93.6	62.5	91.8	88.3	84.0

Sensitivity of α, β and ρ. We analyze the sensitivity of key hyperparameters in TraceGuard on LLaMA3-8B. Fig. 7 reports three key hyperparameters on Llama3-8B. For α, increasing the response-side trajectory weight improves benign accuracy through calibration, but an overly large value weakens harmful detection; we therefore use $\alpha = 0.1$. For β, combining the maximum trajectory prediction and the last-turn prediction improves benign accuracy without degrading harmful detection, where we set $\beta = 0.5$. For ρ, local prototypes better preserve benign interactions, whereas global prototypes improve attack generalization but introduce more false positives; we then choose $\rho = 0.5$ in our experiment.

D.8 Closed-source Model Evaluation.

Table 15 provides the full results on GPT-4o-mini and DeepSeek-v3.2. Consistent with the H-ACC comparison, both TraceGuard-S and TraceGuard-A outperform guard baselines in Overall ACC across the two closed-source target models. TraceGuard-A further improves over TraceGuard-S, indicating that the adapted risk trajectories transfer to closed-source settings without additional tuning.

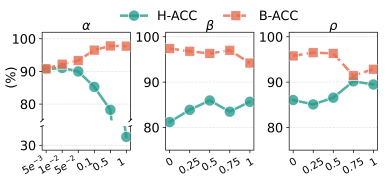


Figure 7: Hyperparameter Analysis.

D.9 Robustness on Representation Encoder

Table 16 shows that replacing the default Llama3-8B feature extractor with Qwen3-8B or Phi-4 preserves the overall detection trend, but reduces Overall ACC by 4.8–6.6 points across target models. Despite this drop, TraceGuard still outperforms the strongest baseline NBF shown in Table 1. The degradation mainly comes from unseen-attack H-ACC, while seen-attack H-ACC remains relatively high. This indicates that alternative extractors can still capture seen attack patterns, but provide weaker risk representations for unseen attack generalization. Overall, feature extractor quality mainly affects cross-distribution robustness rather than the basic detection behavior.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, the main claims in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes, the paper includes a clear discussion of its limitations in Secs. 5.4 and 6.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The paper does not present theoretical results requiring formal assumptions or proofs; the proposed method and risk-signal analysis are evaluated empirically.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Yes, the paper provides sufficient details to reproduce the main experimental results that support its core claims and conclusions, which can be found in Sec. 5 and Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Due to safety and ethical concerns around jailbreak data and evaluation scripts, we do not provide full open access to code and data at submission time. We provide experimental details in the appendix for reproducibility and plan to release sanitized artifacts under appropriate safeguards.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Yes, the paper specifies all key training and testing details necessary to understand the reported results in Appendix C.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Yes, the paper reports appropriate statistical measures to assess variability and significance. Specifically, Table 11 reports the main results of our method as mean $\pm$ standard deviation, while Fig. 1 and Fig. 4 include error bars for the empirical analyses.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Yes, the paper provides sufficient information regarding the computational resources used for the experiments in Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, the research presented in the paper conforms to the NeurIPS Code of Ethics in all respects.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses positive impacts in improving LLM safety against multi-turn jailbreak attacks and negative dual-use risks from analyzing jailbreak behaviors. We also describe mitigation practices, including limiting harmful details and planning sanitized releases of code and data.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release high-risk models or unsafe datasets at submission time. Any future release will be sanitized and provided under appropriate safeguards to mitigate misuse risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Yes, the paper properly credits the creators and original owners of all external assets used, including code, datasets, and pretrained models.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets at submission time due to safety and ethical concerns around jailbreak data. We plan to release sanitized artifacts with documentation under appropriate safeguards.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs are used in the methodology and evaluation. We use Llama3-8B as an auxiliary representation encoder for risk-signal construction, evaluate TraceGuard on multiple target LLMs, and use GPT-4o for response scoring when reproducing the NBF baseline. These usages are described in Sections 5 and Appendices A and C. LLMs used only for writing or editing assistance do not affect the methodology, results, or originality of the work.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.