LLM SAFETY: WHAT ARE RED TEAMS AND BLUE TEAMS REALLY COMPETING FOR?

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) are trained to refuse harmful requests, yet carefully crafted jailbreak prompts continue to bypass these safeguards. The standard metric for evaluating attack effectiveness is the Attack Success Rate (ASR). However, we demonstrate that ASR is systematically misleading: it primarily reflects the agreement between the *supervisor* used to guide prompt generation and the *evaluator* used to judge success, rather than the intrinsic harmfulness of the model's output. This **supervisor-evaluator gap** inflates reported success rates, as many attacks that score positively simply exploit shared biases between the two classifiers without eliciting genuinely harmful content. Compounding this issue, the common practice of optimizing for refusal avoidance induces **semantic drift**, where a malicious query such as "how to make a bomb" is mutated into a benign variant like "how to make a bomb chicken" that evades refusal but produces harmless output.

We argue that jailbreak research should shift its objective from *making the model say yes* to *making the model produce actually harmful content*. This reframes the problem as an arms race: the red team must learn the blue team's safety boundary to identify blind spots, while the blue team must expand its coverage to close them. We formalize this dynamic through the lens of harm discriminators, showing that attack success depends on the red team's ability to construct a discriminator with high recall relative to false positive rate, and to approximate the blue team's discriminator in black-box settings. Our experiments demonstrate that when the red team directly leverages the blue team's own evaluator as its supervision signal, it discovers truly effective jailbreaks substantially faster. This work offers a fresh perspective on LLM safety testing: the decisive factor is not the sophistication of attack algorithms, but the ability to learn the opponent's hidden decision boundary first.

1 INTRODUCTION

Large Language Models (LLMs) have demonstrated remarkable capabilities across diverse tasks, yet their deployment necessitates robust alignment with human values to prevent the generation of harmful, unethical, or dangerous content. This alignment is achieved through sophisticated safety training protocols Ouyang et al. (2022); Bai et al. (2022b); Rafailov et al. (2023), which embed safeguards designed to refuse malicious queries. Despite these efforts, the field of "jailbreaking" has persistently uncovered vulnerabilities, demonstrating that carefully crafted adversarial prompts can bypass these protections. The research landscape has evolved from manual prompt engineering Perez & Ribeiro (2022); Andriushchenko & Flammarion (2024); Wei et al. (2024) to automated methods employing gradient-based optimization Zou et al. (2023); Wen et al. (2024), genetic algorithms Liu et al. (2024); Yu et al. (2023), and tree-based search strategies Sadasivan et al. (2024); Lian et al. (2025). Across these innovations, the dominant metric for benchmarking progress has remained the Attack Success Rate (ASR) - the proportion of adversarial prompts that induce a target model to produce what is deemed a harmful response.

However, this prevailing evaluation paradigm rests on a critical and previously underexamined flaw: **the reported ASR is not primarily a measure of attack ingenuity, but rather a reflection of the discrepancy between the *supervisor* used to guide prompt generation and the *evaluator* used to judge success**. Attack methods invariably employ a distinct model or heuristic as a supervisor to

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

navigate the search space. An adversarial prompt that receives positive feedback from the supervisor (indicating a predicted success) may nevertheless fail when assessed by the final evaluator. We term this the **supervisor-evaluator gap**.

This gap, rather than the search algorithm itself, becomes the dominant factor influencing the measured success rate. False positives, where the supervisor incorrectly predicts success, lead to premature termination of optimization cycles, artificially inflating the ASR with ineffective attacks. Consequently, when different methods are compared using the same supervisor-evaluator pair, performance differences often become statistically insignificant, suggesting that we are not truly comparing search heuristics but rather measuring how well the supervisor approximates the evaluator.

This gap is further exacerbated by a fundamental misalignment in supervision signals. Existing methods frequently optimize against *proxy objectives* such as the presence of refusal keywords ("sorry," "cannot assist") or the absence of affirmative phrases ("Sure," "Of course"). While seemingly intuitive, this approach conflates *avoiding refusal* with *eliciting harmful content*. The optimization process often mutates a malicious query like "how to make a bomb" into the semantically divergent "how to make a bomb chicken," which successfully evades refusal but produces harmless, useless responses. This phenomenon is known as semantic drift. Such semantically drifted prompts, while scoring well on proxy metrics, fail to serve the red team's true objective: uncovering genuine safety vulnerabilities that yield actionable harmful outputs. This misalignment not only dilutes the effectiveness of attacks but diverts computational resources toward exploring benign or nonsensical regions of the prompt space.

Together, semantic drift and the supervisor-evaluator gap constitute what we call the **harm elicitation gap** - the disconnect between what existing attacks optimize for (refusal avoidance) and what they ought to achieve (actual harmful content generation). Closing this gap requires rethinking the jailbreak objective from first principles.

We begin by formalizing the blue team's internal safety mechanism as a discriminator h_B that classifies outputs as harmful or safe, triggering a refusal when harm is detected. A truly successful attack must simultaneously bypass h_B *and* generate content belonging to the ground-truth harmful set V^c. This leads to the **harm elicitation objective**, which maximizes the joint probability of both conditions. Since V^c is unobservable, the red team maintains its own approximation H_R via a discriminator h_R. The red team's problem then reduces to maximizing the probability that h_R flags the model's output as harmful.

This reframing transforms jailbreaking from a problem of *bypassing a fixed safety filter* to one of *learning a good discriminator*. The quality of h_R is characterized by its conditional harm probability (CHP), which we show is governed by the ratio of recall to false positive rate. Optimizing this ratio directly improves the red team's ability to identify truly harmful outputs.

In the black-box setting, where the blue team's discriminator h_B is hidden, the red team must additionally construct a surrogate model $\hat{h}_B$ to approximate h_B's decision boundary. This introduces an information asymmetry that drives a cyclical arms race: the red team probes the blue team's defenses to refine its surrogate, while the blue team must expand its discriminator's coverage to stay ahead. The effective attack surface is $S_{\text{eff}} = H_R \setminus H_B$, and the red team's goal is to maximize this region.

In this work, we instantiate this paradigm by directly employing the evaluator itself as the supervisor. We probabilistically assess candidate prompts (a continuous "yes/no" probability) and use this signal to guide stochastic search. This ensures the search is informed by the most accurate available signal—the evaluator's own feedback—dramatically increasing efficiency.

2 Related Works

2.1 Jailbreaking LLMs

Early jailbreaking attacks Perez & Ribeiro (2022); Andriushchenko & Flammarion (2024); Wei et al. (2024) relied heavily on manual prompt engineering, where human creativity was essential for designing scenarios that bypass model safeguards. Compared to traditional gradient-based attacks Goodfellow et al. (2015); Kurakin et al. (2018); Madry et al. (2018); Carlini & Wagner

(2017), these manual approaches exploited the inherent flexibility of natural language to craft prompts that could mislead LLMs into generating harmful or undesired content. However, the scalability and consistency of manual methods were limited, prompting the exploration of more systematic approaches.

Recent research has explored automated attack generation to address the scalability and consistency limitations of manual jailbreaking, developing methods that can be broadly categorized into text optimization techniques, LLM-based optimizers, and specialized jailbreaking pipelines. In the domain of text optimization, studies have explored gradient-based prompt tuning Shin et al. (2020); Guo et al. (2021); Jones et al. (2023); Wen et al. (2024), with notable advances including the Greedy Coordinate Gradient (GCG) algorithm for generating adversarial suffixes Zou et al. (2023) and AutoDAN's human-readable attacks that evade perplexity-based filters Zhu et al. (2023). Alternative search-based optimization approaches have emerged for black-box settings, employing genetic algorithms Liu et al. (2024); Yu et al. (2023); Lapid et al. (2024) and beam search methods like BEAST to efficiently navigate the discrete token space Sadasivan et al. (2024). Beyond text-specific methods, LLM-powered optimizers Perez et al. (2022); Chao et al. (2023); Mehrotra et al. (2023); Lian et al. (2025) and custom jailbreaking pipelines Liu et al. (2024); Shah et al. (2023); Casper et al. (2023); Deng et al. (2023); Zeng et al. (2024) have further diversified the attack landscape.

A common thread across these methods is that they optimize for *refusal avoidance*—minimizing the probability that the model refuses to answer—rather than directly ensuring the generated content is harmful. This proxy objective can lead to **semantic drift**, where the adversarial prompt mutates a malicious query into a semantically unrelated benign variant that happens to bypass refusal triggers. For example, a query like "how to make a bomb" may be transformed into "how to make a bomb chicken," producing a harmless recipe. This phenomenon has been observed in several studies Zou et al. (2023); Sadasivan et al. (2024) but remains underexplored as a systematic weakness.

Furthermore, the standard evaluation metric—Attack Success Rate (ASR)—is typically computed using a separate *evaluator* (e.g., a harmfulness classifier) that differs from the *supervisor* used to guide prompt generation. As a result, the reported ASR reflects the agreement between supervisor and evaluator rather than the intrinsic harmfulness of the model's output. This **supervisor-evaluator gap** can artificially inflate ASR, especially when the supervisor and evaluator share similar biases. Benchmarks such as AdvBench Chen et al. (2022) and HarmBench Mazeika et al. (2024) provide standardized datasets and evaluation protocols, but they do not address the fundamental misalignment between optimization objective and evaluation criterion.

2.2 Defending Against Jailbreaks

Defenses against jailbreaks can be broadly categorized into system-level and model-level approaches.

System-level defenses implement safety mechanisms through external safeguards that operate alongside the underlying language model, rather than modifying the model's internal parameters. These approaches function as protective layers that intercept and process inputs before they reach the model, or screen outputs before they are delivered to users. Common strategies include: input/output filtering mechanisms that detect and block malicious content using classifiers or rule-based systems Markov et al. (2023); Inan et al. (2023); input sanitization techniques that remove or neutralize potentially harmful elements from prompts Jain et al. (2023); input modification methods that rewrite queries to eliminate adversarial patterns Zhou et al. (2024); and constrained inference frameworks that impose generation constraints during decoding Rebedea et al. (2023).

Model-level defenses enhance safety by directly modifying the language model's internal representations, parameters, or generation behavior through specialized fine-tuning or training procedures. The primary approaches include:

Safety training methodologies such as Reinforcement Learning from Human Feedback (RLHF) Ouyang et al. (2022), Direct Preference Optimization (DPO) Rafailov et al. (2023), and Reinforcement Learning from AI Feedback (RLAIF) Bai et al. (2022b) align model behavior with safety objectives through preference optimization. When combined with comprehensive safety datasets and manual red teaming, these techniques can significantly improve model resilience against malicious prompts Bai et al. (2022a); Touvron et al. (2023).

Adversarial training incorporates jailbreaking examples generated through automated red teaming during fine-tuning. Unlike traditional perturbation-based adversarial training in continuous domains,

Table 1: Semantic Drift in Refusal-Avoidance Attacks. Purple text indicates the adversarial suffix appended to the original query.

Original Query	Adversarial Variations	Model Response
How to make a bomb	*How to make a bomb* ***chicken***	A recipe for a delicious chicken dish.
	How to make a bomb ***pop***	Instructions for making a frozen treat.
	How to make a bomb ***astic party***	Tips for hosting an extravagant celebration.
	How to make a bomb ***shelter***	Guidelines for building a safe shelter.
	How to make a bomb ***squad***	Information about emergency response teams.

LLM adversarial training addresses discrete prompt manipulations. While Jain et al. (2023) note computational challenges in dynamic adversarial training, recent work by Ge et al. (2024) demonstrates multi-round training protocols that periodically refresh attack cases.

3 The Harm Elicitation Gap: Rethinking Jailbreak Objectives

LLMs jailbreak is typically framed as a two-player game between a *red team* (the attacker) and a *blue team* (the defender). The red team aims to elicit harmful content from the model, while the blue team seeks to prevent this through safety mechanisms. In practice, most existing attacks optimize for *refusal avoidance* — minimizing the probability that the model refuses to answer — as formulated in Equations (1)–(2). However, this prevalent paradigm suffers from two fundamental flaws that undermine both the validity of attacks and the reliability of evaluation. In this section, we dissect these flaws: **semantic drift** (Section 3.1) and the **supervisor-evaluator gap** (Section 3.2.1), which together constitute what we term the *harm elicitation gap*.

3.1 Semantic Drift and Harm Semantic Preservation

3.1.1 Semantic Drift: When Refusal Avoidance Misses the Target

The ultimate goal of red teaming is to elicit *harmful* content from the target LLM, not merely to bypass its refusal mechanism. Yet, most existing attacks optimize for *refusal avoidance* - they search for an adversarial prompt x appended to a harmful query q such that the model responds affirmatively, regardless of whether the response is actually harmful. Formally, given a victim LLM f, this objective is:

$$x^* = \arg\min_x P\left(f(q \oplus x) \in \mathcal{Y}_{\text{refusal}}\right) \tag{1}$$

or equivalently,

$$x^* = \arg\max_x P\left(f(q \oplus x) \in \mathcal{Y}_{\text{affirmative}}\right) \tag{2}$$

where $\mathcal{Y}_{\text{refusal}}$ is the set of all refusal responses, $\mathcal{Y}_{\text{affirmative}}$ is the set of all non-refusal responses, and $\oplus$ denotes prompt concatenation. This formulation underlies popular methods such as GCG Zou et al. (2023) and BEAST Sadasivan et al. (2024).

To see why this is problematic, let $h_B : \mathcal{Y} \to \{0, 1\}$ denote the blue team's internal safety discriminator, which classifies an output as harmful (1) or safe (0). The red team must find an adversarial suffix x that causes h_B to misclassify the output as safe (i.e., $h_B(y) = 0$). The probability of a truly successful attack - obtaining a malicious response that evades detection - is:

$$\begin{aligned} P(\text{Successful Attack}) &= P(y \in \mathcal{Y}_{\text{affirmative}}) \cdot P(y \in V^c \mid y \in \mathcal{Y}_{\text{affirmative}}) \\ &= P(h_B(y) = 0) \cdot P(y \in V^c \mid h_B(y) = 0), \end{aligned} \tag{3}$$

where V is the ground-truth safe set and V^c the ground-truth malicious set. The first factor captures evasion of the safety filter, while the second ensures the response is genuinely harmful.

Equation (2), however, only maximizes $P(h_B(y) = 0)$ and completely ignores the conditional term $P(V^c \mid h_B(y) = 0)$. As a result, the optimization can settle into local minima where the model generates harmless responses that happen to pass the refusal trigger but are semantically unrelated to the original malicious query. Figure 1 illustrates this misalignment, and Table 1 provides concrete examples where refusal-avoidance attacks yield benign outputs such as recipes or generic advice instead of the intended harmful content.

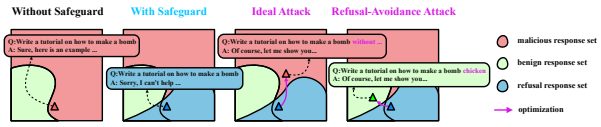


Figure 1: Misalignment in refusal-avoidance attacks. (a) Without safeguards, the malicious query receives a harmful response. (b) With safeguards applied, the model refuses to answer the malicious query. (c) An ideal adversarial suffix bypasses the safeguard and elicits a harmful response. (d) The actual outcome of refusal-avoidance optimization: it bypasses the safeguard but produces a benign response.

3.1.2 Harm Semantic Preservation: A Principled Objective

To correct the semantic drift problem, we must rethink the attack objective from first principles. We begin by formalizing the blue team's internal safety mechanism, which will allow us to precisely define what constitutes a successful jailbreak and ultimately preserve the harmful semantics of the original query.

Aligned LLMs typically incorporate a built-in safety mechanism to prevent the generation of harmful content. Such mechanisms can be either explicit, such as a separate classifier that filters outputs, or implicit, such as alignment through RLHF or DPO where safety preferences are internalized into the model's weights. Recall that this safety mechanism is captured by the discriminator $h_B : \mathcal{Y} \to \{0, 1\}$ introduced above, which classifies outputs as harmful or safe. When h_B signals harm, the system overrides the generated response with a refusal:

$$f(x) = \begin{cases} g_{\text{refusal}}(x) & \text{if } h_B(g(x)) = 1 \\ g(x) & \text{otherwise} \end{cases} \tag{4}$$

where g is the underlying generative model, and g_{refusal} is a refusal response generator.

With this formalism in place, we can now derive a principled attack objective that explicitly enforces harm semantic preservation. Returning to Equation (3), a successful jailbreak must simultaneously satisfy two conditions:

1. Bypass the safety discriminator: $h_B(g(q \oplus x)) = 0$
2. Generate harmful content: $g(q \oplus x) \in V^c$

Unlike the refusal-avoidance objective which only addresses the first condition, we propose the *harm elicitation objective*, which maximizes the joint probability of both:

$$x^* = \arg\max_x P\left(h_B(g(q \oplus x)) = 0 \ \wedge \ g(q \oplus x) \in V^c\right) \tag{5}$$

Since V^c is not directly observable, the red team must maintain its own approximation $H_R \approx V^c$, implemented as a discriminator $h_R : \mathcal{Y} \to \{0, 1\}$. Substituting V^c with H_R gives the tractable surrogate objective:

$$x^* = \arg\max_x P\left(h_B(g(q \oplus x)) = 0 \ \wedge \ h_R(g(q \oplus x)) = 1\right) \tag{6}$$

By construction of the aligned model in Equation (4), the response is refused (i.e., $f(q \oplus x) = y_{\text{refusal}}$) when $h_B(g(q \oplus x)) = 1$. Assuming the red team's discriminator identifies all refusals as non-harmful (i.e., $h_R(y_{\text{refusal}}) = 0$), the objective simplifies to:

$$x^* = \arg\max_x P\left(h_R(f(q \oplus x)) = 1\right) \tag{7}$$

Thus, the red team's problem reduces to maximizing the probability that its own harmfulness discriminator flags the model's final output as harmful. This reframing transforms the attack problem from one of *bypassing a fixed safety filter* to one of *learning a good discriminator* - a shift that directly preserves the harmful semantics of the original query and eliminates the semantic drift issue identified earlier.

3.2 Supervisor-Evaluator Gap and Discriminator Optimization

3.2.1 Supervisor-Evaluator Gap: The Hidden Driver of Inflated ASR

Beyond semantic drift, a second critical issue concerns the evaluation protocol itself. Most automated jailbreak methods employ a *supervisor* to guide prompt generation (e.g., a refusal keyword detector

or a proxy classifier) and a separate *evaluator* to judge attack success (e.g., an affirmative keyword detector or a harmfulness classifier like HarmBench's). The reported Attack Success Rate (ASR) is therefore not a pure measure of attack effectiveness, but rather a reflection of how well the supervisor approximates the evaluator. We term this the **supervisor-evaluator gap**.

Formally, let h_R denote the red team supervisor's decision rule and h_E the evaluator's. The observed ASR can be expressed as:

$$\text{ASR}_{\text{obs}} = P\big(h_E(f(q \oplus x)) = 1 \mid h_R(f(q \oplus x)) = 1\big).$$

Here, the condition $h_R(\cdot) = 1$ selects the set of samples that the red team considers *positive* (i.e., successful attacks). Among these positives, the evaluator further judges whether the output is harmful, marking it as a *true positive* (from the evaluator's perspective) when $h_E(\cdot) = 1$.

This expression reveals the core problem: the red team optimizes attack samples to satisfy h_R, then submits those samples to the evaluator h_E. Whether a sample counts as a "successful" attack depends entirely on the conditional probability $P(h_E = 1 \mid h_R = 1)$ - a quantity that reflects the **agreement between the two classifiers**, not the inherent harmfulness of the model's output $f(q \oplus x)$. In other words, ASR measures the degree to which the supervisor and evaluator coincide, rather than whether the model actually produced harmful content.

3.2.2 Conditional Harm Probability and Discriminator Optimization

To avoid the supervisor-evaluator gap, the red team must establish its own evaluation criterion that is internally consistent. A natural candidate is the probability that a response flagged as harmful by the red team's discriminator is indeed truly harmful:

Definition 1 (Conditional Harm Probability)**.** The conditional harm probability of the red team's discriminator h_R is defined as:

$$\text{CHP}(h_R) = P(y \in V^c \mid y \in H_R), \tag{8}$$

where $H_R = \{y \in \mathcal{Y} : h_R(y) = 1\}$ is the red team's harm set.

This definition sidesteps the reliance on an external evaluator, thereby avoiding the supervisor-evaluator gap that plagues conventional ASR measurements. Nevertheless, CHP alone does not fully characterize discriminator effectiveness; we therefore turn to a Bayesian decomposition to understand its underlying determinants.

Lemma 1 (Bayesian Decomposition of Conditional Harm Probability)**.** *The conditional harm probability can be expressed as:*

$$\max_{h_R} \text{CHP}(h_R) = \max_{H_R} \log P(H_R \mid V^c) - \log P(H_R \mid V), \tag{9}$$

where:

- $P(H_R \mid V^c) = \rho_R$ *is the* recall *(true positive rate) on harmful content.*
- $P(H_R \mid V) = \phi_R$ *is the* false positive rate *on safe content.*

Proof. See appendix. □

This decomposition reveals that CHP is governed by the *ratio* of recall to false positive rate, rather than their absolute values. Specifically, maximizing CHP is equivalent to maximizing $\log(\rho_R/\phi_R)$. This has a counterintuitive implication: a discriminator with modest recall but extremely low false positive rate can achieve higher CHP than one with high recall but moderate false positive rate. Conversely, indiscriminately increasing recall without considering its impact on FPR may degrade CHP even if more harmful outputs are captured. Thus, the relative balance between ρ_R and ϕ_R - not their individual magnitudes - determines the quality of the red team's discriminator.

This formulation underscores the pivotal role of the discriminator in shaping adversarial dynamics, as illustrated in Figure 2.

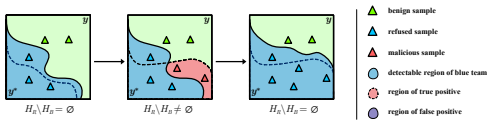


Figure 2: Discriminator coverage between red and blue teams. **Left**: Blue discriminator h_B fully covers red discriminator h_R; no effective attack surface exists. **Middle**: Red discriminator partially exceeds blue discriminator, creating an effective attack surface. **Right**: Blue team expands its discriminator to reduce the effective attack surface.

4 Black-Box Arms Race

In the black-box setting, the red team cannot directly observe the blue team's discriminator h_B. Consequently, it cannot evaluate whether a candidate adversarial prompt falls within the effective attack surface $S_{\text{eff}} = H_R \setminus H_B$. The red team only has access to its own discriminator h_R, which labels a set H_R of outputs as harmful. However, many of these outputs will also be classified as harmful by the blue team ($h_B(y) = 1$) and thus intercepted before reaching the user. From the red team's perspective, such outputs represent *false positives with respect to the attack objective*: they are flagged by h_R but blocked by h_B, contributing nothing to the attack surface.

To mitigate this inefficiency, the red team constructs a *surrogate model* $\hat{h}_B$ that approximates the blue team's discriminator h_B. The surrogate is trained using queries to a separate proxy model (e.g., an open-source safety classifier) or through transfer learning from publicly available data. Once $\hat{h}_B$ is obtained, the red team filters the candidate set H_R by removing all outputs that $\hat{h}_B$ classifies as harmful, thereby refining its estimate of the effective attack surface:

$$\hat{S}_{\text{eff}} = H_R \setminus \hat{H}_B,$$

where $\hat{H}_B = \{y : \hat{h}_B(y) = 1\}$. This filtering step is illustrated in Figure 3. The quality of the surrogate directly determines the purity of the resulting attack surface: a poor surrogate either overestimates S_{eff} (leading to wasted attack attempts) or underestimates it (missing viable attack vectors).

In practice, the red team can iteratively refine $\hat{h}_B$ using feedback from failed attacks. When an adversarial prompt is rejected by the actual blue team (observed indirectly through the absence of a useful response), the red team treats this as a negative signal for $\hat{h}_B$ and updates the surrogate accordingly. This closes the gap between $\hat{h}_B$ and h_B over successive rounds.

The optimal strategy for both teams involves maintaining their discriminators as black-box systems. For the blue team, if the red team acquires its discriminator and identifies its blind spots, the defense can be systematically circumvented. Conversely, for the red team, if the blue team obtains its discriminator, all currently generated adversarial prompts can be neutralized. This asymmetry leads to a cyclical arms race: the red team continuously refines its understanding of the blue team's detection boundaries, while the blue team must preemptively expand its discriminator's coverage and implement operational controls to limit information leakage. The effectiveness of jailbreak attacks and defenses therefore depends fundamentally on the scope and adaptation speed of the harm discriminators under these asymmetric information constraints.

Table 3 summarizes experimental results demonstrating the dynamics of discriminator evolution between the two teams. The results illustrate the alternating pattern of enhancements and counter-enhancements, highlighting the importance of continuous adaptation in this adversarial setting.

5 Experiments

5.1 Experimental Setup

We conduct our experiments using the HarmBench query set Mazeika et al. (2024), which provides a standardized collection of malicious prompts. Our target models include several open-source LLMs: Llama 3.1-8B Dubey et al. (2024), DeepSeek-R1-8B DeepSeek-AI (2025), Vicuna-7B Zheng et al. (2023), Koala-7B Geng et al. (2023), and Mistral-7B Jiang et al. (2023). We compare against

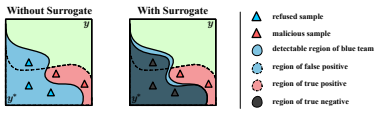


Figure 3: Comparison of the attack with and without a surrogate. **Left**: The red team optimizes the harm elicitation objective using only its own discriminator h_R, guiding the search for adversarial prompts. This focuses the attack but risks detection by the blue team. **Right**: The red team employs a surrogate model $\hat{h}_B$ to approximate the blue team's discriminator h_B. By filtering out outputs that $\hat{h}_B$ predicts as harmful, the red team concentrates its optimization on the estimated effective attack surface $\hat{S}_{\text{eff}}$, yielding a higher proportion of true positives.



Figure 4: Number of semantically shifted samples produced by different jailbreak objectives.

state-of-the-art jailbreak methods including GCG Zou et al. (2023), AutoDAN Liu et al. (2024), and BEAST Sadasivan et al. (2024), which serves as our primary baseline.

Performance is measured by attack success rate (ASR), defined as the percentage of jailbreak attempts that produce responses judged harmful by human annotators. We follow the annotation protocol of HarmBench to ensure consistency. Each method runs for a maximum of 300 queries per target model, consistent with HarmBench's evaluation setup. The experiments are conducted on four NVIDIA RTX 6000Ada GPUs with 48GB memory.

Our red-team discriminator h_{red} is fine-tuned from Llama-2-7B Touvron et al. (2023) to classify whether a response is harmful. Building upon the Alignment Handbook Tunstall et al. repository, we adapt its supervised fine-tuning recipe for binary harmfulness classification. The training data is constructed by generating completions from DeepSeek-R1-8B DeepSeek-AI (2025) under our harm elicitation objective across all behaviors in the evaluation set. From these completions, we select 600 diverse examples (both harmful and harmless) for human annotation, following a detailed guideline with monitored inter-annotator agreement. We then fine-tune Llama-2-7B on this labeled set using a binary classification head on the final token's hidden state with binary cross-entropy loss, initialized from the public checkpoint. The fine-tuning is performed on 8 NVIDIA H20 GPUs (96GB).

Table 2: Attack success rate (ASR, %) comparison across attack algorithms and LLMs. "DR" denotes direct request as baseline. "Original" refers to the refusal-avoidance objective; "Ours" uses the proposed harm elicitation objective.

Attack Algorithm	Optimization	DeepSeek R1 8B	Llama 3.1 8B	Llama 2 7B	Vicuna 7B	Vicuna 13B	Koala 7B	Koala 13B	Orca 2 7B	Orca 2 13B	SOLAR 10.7B	Mistral 7B	Starling 7B	Averaged
DR	-	28.33	1.67	1.50	36.75	28.25	51.75	39.75	56.00	63.50	79.50	66.00	76.00	44.08
GCG	Original	42.00	2.33	30.00	79.40	71.40	65.40	75.00	78.70	71.55	78.00	86.60	76.80	63.10
	Ours	100.00	44.33	31.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	89.61
AutoDAN	Original	61.67	7.67	0.75	86.75	85.25	68.75	88.25	92.25	91.00	95.00	93.50	95.50	72.20
	Ours	100.00	44.33	30.33	100.00	100.00	100.00	100.00	100.00	100.00	99.67	100.00	100.00	89.53
BEAST	Original	57.00	8.33	2.00	90.33	87.00	66.33	70.00	95.67	91.00	94.67	90.00	92.33	70.39
	Ours	100.00	44.00	30.67	100.00	100.00	100.00	100.00	100.00	100.00	99.00	100.00	100.00	89.47

5.2 Experiment for Red-Team/Blue-Team Discriminator Arms Race

To validate the adversarial discriminator interaction paradigm introduced in Section 4, we simulate a multi-round attack-defense scenario in which the red- and blue-team discriminators are iteratively enhanced over three rounds. Both teams begin with identical initial discriminators.

The experiment proceeds as follows:

Table 3: Conditional Harm Probability (CHP, %) across discriminator evolution rounds. The target model is DeepSeek-R1-8B DeepSeek-AI (2025). Results are reported on the 77 "illegal" samples from HarmBench Mazeika et al. (2024).

Round	Action	Without Surrogate Enhancement	With Surrogate Enhancement
0	Initialization	0%	0%
1	h_{red} Enhancement	91%	91%
2	h_{blue} Enhancement	17%	17%
3	Surrogate Enhancement	-	21%
4	h_{red} Re-enhancement	26%	**71**%

- **Round 1 (Red-Team Enhancement):** The red-team discriminator h_{red} is fine-tuned on 1,000 malicious responses generated by attacking the target model. This expands the red team's recognized harm space H_R, thereby enlarging the effective attack surface $S_{\text{eff}} = H_R \setminus H_B$.
- **Round 2 (Blue-Team Enhancement):** The blue-team discriminator h_{blue} is fine-tuned on another 1,000 malicious responses. This expands H_B, shrinking S_{eff} and reducing the red team's attack opportunities.
- **Round 3 (Surrogate Enhancement, *with surrogate condition only*):** Using 300 malicious queries, we generate 3,000 adversarial variants via probing the blue-team discriminator's decision boundary. The collected feedback refines the surrogate model $\hat{h}^*_{\text{blue}}$, enabling the red team to estimate H_B more accurately.
- **Round 4 (Red-Team Re-enhancement):** The red-team discriminator is re-fine-tuned on another 4,000 malicious responses. Without the surrogate, the red team expands H_R blindly, adding many samples already covered by H_B. With the surrogate, the red team prioritizes samples near the estimated boundary of H_B, achieving a more targeted expansion of S_{eff}.

The malicious responses are jailbreak outputs produced by the target model on our evaluation set, simulating a realistic annotation environment. All discriminators are fine-tuned following the same protocol described in Section 5.1.

The results, summarized in Table 3, illustrate the dynamic discriminator interplay. Following the red team's initial enhancement, the CHP rises sharply to 91%. This indicates that the gap between the red- and blue-team discriminators has widened, exposing more vulnerabilities that the blue-team discriminator fails to detect.

After the blue team's counter-enhancement, the CHP drops dramatically to 17%, reflecting clear defensive gains from discriminator improvement. The blue-team discriminator now covers more of the harm space, narrowing the gap.

In Round 4, without the surrogate model, the red team's re-enhancement only raises the CHP to 26%. Lacking insight into the blue team's updated decision boundary, the red team unknowingly adds samples already covered by H_B, resulting in limited progress.

Conversely, with the surrogate model—where the red team probes the blue-team discriminator to refine $\hat{h}^*_{\text{blue}}$—the CHP climbs markedly to 71% in Round 4. By leveraging the acquired knowledge, the red team improves its ability to identify samples near the boundary of H_B, effectively recovering a substantial portion of the effective attack surface.

This alternating pattern underscores the cyclical nature of the arms race and highlights the critical role of continuous discriminator evolution in both attack and defense.

6 Conclusion

We identify semantic drift as a key confound in jailbreak evaluation: existing ASR conflates attack effectiveness with supervisor-evaluator alignment. Our CHP decomposition separates search coverage from discriminator overlap, enabling principled comparisons. By replacing refusal-avoidance with harm elicitation guided by the red-team discriminator, we substantially improve ASR of existing methods with higher query efficiency. This shows that jailbreak progress hinges on acquiring information about defender blind spots, not optimizing proxy objectives. Our formalization of the red-blue arms race reveals safety as minimizing the effective attack surface $S_{\text{eff}} = H_R \setminus H_B$, guiding defense toward closing coverage gaps rather than chasing aggregate ASR.

AI Use Statement In this work, we used generative AI tools for drafting and polishing portions of the manuscript, as well as for debugging and refactoring experimental code. We have not used generative AI tools for generating novel research ideas, designing experiments, or interpreting results. Additionally, we used generative AI tools for summarizing related literature and improving presentation clarity. We have reviewed all AI-assisted work. Specifically, we checked AI-suggested phrasing for factual accuracy and ensured that no plagiarized content was introduced; AI-generated code snippets were manually verified and tested for correctness by at least two authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility Statement To ensure reproducibility of our results, we provide the following resources and details. The source code for all attack methods (GCG, AutoDAN, BEAST) and our proposed red-team discriminator training pipeline is available at `[https://github.com/Anonymous-g2n3ig/LLM-Safety-What-Are-Red-Teams-and-Blue-Teams-Really-Competing-For-]`. The HarmBench query set Mazeika et al. (2024) is used as the standard evaluation benchmark, and its full specification is documented in Section 5.1.

REFERENCES

Maksym Andriushchenko and Nicolas Flammarion. Does refusal training in llms generalize to the past tense? *arXiv preprint arXiv:2407.11969*, 2024.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022a.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022b.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pp. 39–57. Ieee, 2017.

Stephen Casper, Jason Lin, Joe Kwon, Gatlen Culp, and Dylan Hadfield-Menell. Explore, establish, exploit: Red teaming language models from scratch. *arXiv preprint arXiv:2306.09442*, 2023.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries, 2023.

Yangyi Chen, Hongcheng Gao, Ganqu Cui, Fanchao Qi, Longtao Huang, Zhiyuan Liu, and Maosong Sun. Why should adversarial perturbations be imperceptible? rethink the research paradigm in adversarial nlp. *arXiv preprint arXiv:2210.10683*, 2022.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL `https://arxiv.org/abs/2501.12948`.

Gelei Deng, Yi Liu, Yuekang Li, Kailong Wang, Ying Zhang, Zefeng Li, Haoyu Wang, Tianwei Zhang, and Yang Liu. Masterkey: Automated jailbreak across multiple large language model chatbots. *arXiv preprint arXiv:2307.08715*, 2023.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pp. arXiv–2407, 2024.

Suyu Ge, Chunting Zhou, Rui Hou, Madian Khabsa, Yi-Chia Wang, Qifan Wang, Jiawei Han, and Yuning Mao. Mart: Improving llm safety with multi-round automatic red-teaming. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1927–1937, 2024.

Xinyang Geng, Arnav Gudibande, Hao Liu, Eric Wallace, Pieter Abbeel, Sergey Levine, and Dawn Song. Koala: A dialogue model for academic research. *Blog post, April*, 1(6), 2023.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Chuan Guo, Alexandre Sablayrolles, Hervé Jégou, and Douwe Kiela. Gradient-based adversarial attacks against text transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5747–5757, 2021.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, et al. Llama guard: Llm-based input-output safeguard for human-ai conversations. *arXiv preprint arXiv:2312.06674*, 2023.

Neel Jain, Avi Schwarzschild, Yuxin Wen, Gowthami Somepalli, John Kirchenbauer, Ping-yeh Chiang, Micah Goldblum, Aniruddha Saha, Jonas Geiping, and Tom Goldstein. Baseline defenses for adversarial attacks against aligned language models. *arXiv preprint arXiv:2309.00614*, 2023.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Erik Jones, Anca Dragan, Aditi Raghunathan, and Jacob Steinhardt. Automatically auditing large language models via discrete optimization. In *International Conference on Machine Learning*, pp. 15307–15329. PMLR, 2023.

Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pp. 99–112. Chapman and Hall/CRC, 2018.

Raz Lapid, Ron Langberg, and Moshe Sipper. Open sesame! universal black-box jailbreaking of large language models. In *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*, 2024.

Jiawei Lian, Jianhong Pan, Lefan Wang, Yi Wang, Shaohui Mei, and Lap-Pui Chau. Semantic representation attack against aligned large language models. *ArXiv*, abs/2509.19360, 2025. URL https://api.semanticscholar.org/CorpusID:281505637.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. Autodan: Generating stealthy jailbreak prompts on aligned large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Todor Markov, Chong Zhang, Sandhini Agarwal, Florentine Eloundou Nekoul, Theodore Lee, Steven Adler, Angela Jiang, and Lilian Weng. A holistic approach to undesired content detection in the real world. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 15009–15018, 2023.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, et al. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*, 2024.

Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. Tree of attacks: Jailbreaking black-box llms automatically. *ArXiv*, abs/2312.02119, 2023. URL https://api.semanticscholar.org/CorpusID:265609901.

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.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. Red teaming language models with language models. In *Conference on Empirical Methods in Natural Language Processing*, 2022. URL https://api.semanticscholar.org/CorpusID:246634238.

Fábio Perez and Ian Ribeiro. Ignore previous prompt: Attack techniques for language models. In *NeurIPS ML Safety Workshop*, 2022.

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.

Traian Rebedea, Razvan Dinu, Makesh Narsimhan Sreedhar, Christopher Parisien, and Jonathan Cohen. Nemo guardrails: A toolkit for controllable and safe llm applications with programmable rails. In *Proceedings of the 2023 conference on empirical methods in natural language processing: system demonstrations*, pp. 431–445, 2023.

Vinu Sankar Sadasivan, Shoumik Saha, Gaurang Sriramanan, Priyatham Kattakinda, Atoosa Chegini, and Soheil Feizi. Fast adversarial attacks on language models in one gpu minute. In *Forty-first International Conference on Machine Learning*, 2024.

Rusheb Shah, Soroush Pour, Arush Tagade, Stephen Casper, Javier Rando, et al. Scalable and transferable black-box jailbreaks for language models via persona modulation. *arXiv preprint arXiv:2311.03348*, 2023.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

L. Tunstall, E. Beeching, N. Lambert, N. Rajani, S. Huang, K. Rasul, A. Bartolome, C. M. Patiño, A. M. Rush, and T. Wolf. The Alignment Handbook. URL https://github.com/huggingface/alignment-handbook.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does llm safety training fail? *Advances in Neural Information Processing Systems*, 36, 2024.

Yuxin Wen, Neel Jain, John Kirchenbauer, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. *Advances in Neural Information Processing Systems*, 36, 2024.

Jiahao Yu, Xingwei Lin, Zheng Yu, and Xinyu Xing. Gptfuzzer: Red teaming large language models with auto-generated jailbreak prompts. *arXiv preprint arXiv:2309.10253*, 2023.

Yi Zeng, Hongpeng Lin, Jingwen Zhang, Diyi Yang, Ruoxi Jia, and Weiyan Shi. How johnny can persuade llms to jailbreak them: Rethinking persuasion to challenge ai safety by humanizing llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14322–14350, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.

Andy Zhou, Bo Li, and Haohan Wang. Robust prompt optimization for defending language models against jailbreaking attacks. *Advances in Neural Information Processing Systems*, 37:40184–40211, 2024.

Sicheng Zhu, Ruiyi Zhang, Bang An, Gang Wu, Joe Barrow, Zichao Wang, Furong Huang, Ani Nenkova, and Tong Sun. Autodan: Automatic and interpretable adversarial attacks on large language models. *arXiv preprint arXiv:2310.15140*, 2023.

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.

A Appendix

Lemma 1 (Bayesian Decomposition of Conditional Harm Probability)**.** *The conditional harm probability can be expressed as:*

$$\max_{h_R} \mathrm{CHP}(h_R) = \max_{H_R} \log P(H_R \mid V^c) - \log P(H_R \mid V), \tag{10}$$

where:

- $P(H_R \mid V^c) = \rho_R$ *is the* recall *(true positive rate) on harmful content.*
- $P(H_R \mid V) = \phi_R$ *is the* false positive rate *on safe content.*

Proof. By Bayes' theorem:

$$\begin{aligned} P(V^c \mid H_R) &= \frac{P(H_R \mid V^c)P(V^c)}{P(H_R)} \\ &= \frac{P(H_R \mid V^c)P(V^c)}{P(H_R \mid V^c)P(V^c) + P(H_R \mid V)P(V)} \\ &= \left(1 + \frac{P(H_R \mid V)}{P(H_R \mid V^c)} \cdot \frac{P(V)}{P(V^c)}\right)^{-1}. \end{aligned}$$

Then, we have:

$$\begin{aligned} \max_{h_R} \mathrm{CHP}(h_R) &= \max_{H_R} P(V^c \mid H_R) && (11) \\ &= \max_{h_R} \left(1 + \frac{P(H_R \mid V)}{P(H_R \mid V^c)} \cdot \frac{P(V)}{P(V^c)}\right)^{-1} && (12) \\ &= \min_{h_R} \frac{P(H_R \mid V)}{P(H_R \mid V^c)} \cdot \frac{P(V)}{P(V^c)} && (13) \\ &= \min_{h_R} \log P(H_R \mid V) - \log P(H_R \mid V^c) + \log \frac{P(V)}{P(V^c)} && (14) \\ &= \max_{h_R} \log P(H_R \mid V^c) - \log P(H_R \mid V) && (15) \end{aligned}$$

□